

University of Alberta

Library Release Form

Name of Author: Hang Cui

Title of Thesis: Web Search Result Re-organization with Ontologies

Degree: Master of Science

Year this Degree Granted: 2001

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

Hang Cui

Date: _____

Take It to the Limit One More Time

University of Alberta

WEB SEARCH RESULT RE-ORGANIZATION WITH ONTOLOGIES

by

Hang Cui

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta
Fall 2001

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Web Search Result Re-organization with Ontologies** submitted by Hang Cui in partial fulfillment of the requirements for the degree of **Master of Science**.

Dr. Osmar R. Zaïane
Supervisor

Dr. Dekang Lin

Dr. Connie K. Varnhagen

Date: _____

To my lovely wife *Jean*

Abstract

In this research, we have designed and implemented a Web search tool that categorizes Web documents returned from Web search engines based on the idea of ontological categorization. It aims at improving the browsability of the long ranked list presentation of search results, which is widely adopted by the industry. The result is a hierarchical organization of Web search results with a mechanism to allow the users to navigate the hierarchy to locate information organized in clusters in a progressive and interactive manner. We have shown a proof of concept to adopt the combination of hierarchical organization and progressive navigation scheme to improve the browsability of Web search results using WordNet as an initial semantic network. Our preliminary evaluation results strongly support the concept behind the design philosophy. We believe that such an approach of Web document re-organization will be the future for the new generation of Web search tools.

Acknowledgements

I would like to take this opportunity to express my gratefulness to many people who have inspired and helped me in many ways in the past two years. Thanks to my committee members for the time they spent in discussing and revising of my thesis. Special thanks goes to my Supervisor Dr. Osmar R. Zaïane for his supervision, patience, and encouragement. His professionalism and dedication inspires me, and definitely makes great impact on my work. Last but not least, I'd like to thank my good friend and fellow student Tingshao Zhu with whom I really enjoy and appreciate all the discussions.

Table of Contents

1	Introduction	1
2	Searching the Web	5
2.1	How Web Search Engines Work	5
2.1.1	Web Search Engine	5
2.1.2	Web Crawling and Indexing	6
2.1.3	Search Result Ranking	8
2.1.4	MetaSearch Engine	12
3	Related Work	15
3.1	Clustering Approach	16
3.1.1	Flat Clustering of Web Documents	16
3.1.2	Hierarchical clustering of Web Documents	17
3.2	Machine Learning Approach	18
3.2.1	Web Document Classification	18
3.2.2	Hierarchical Classification of Web Documents	19
3.3	Web Search Result Visualization	19
4	WordNet	22
4.1	WordNet - an Online Lexical Database	22
4.2	Structure of WordNet	22
4.3	Syntactic Categories in WordNet	23
4.3.1	Nouns in WordNet:	23
4.3.2	Verbs in WordNet:	26
4.3.3	Adjectives in WordNet:	27
5	Architecture Design and Implementation	29
5.1	Issues and Approach Concept	29
5.2	System Model	31
5.2.1	Independent, MetaSearch Approach	31
5.2.2	Ontological Approach for Search Result Organization	33
5.2.3	User's Navigation Scheme	34
5.3	System Architecture Design	35
5.4	Implementation Issues	39

5.4.1	MetaSearch Module	39
5.4.2	System Engine Module	40
6	Case Study and System Evaluation	45
6.1	Case Study	45
6.2	System Evaluation	56
6.2.1	Evaluation of Information Retrieval System	56
6.2.2	System Evaluation Scheme	58
6.2.3	Evaluation Criteria	60
6.2.4	Experiments and Result Analysis	63
7	Conclusion and Future Work	69
7.1	Conclusion	69
7.2	Future Work	70
	Bibliography	72
	Appendix	77

List of Figures

2.1	A Typical Web Crawler - Indexer Architecture [YAT99]	7
2.2	MetaCrawler Flow Control [SEL96]	13
4.1	Top 25 Categories(represented by SynSet) of Nouns in WordNet .	24
5.1	System Component Modeling	32
5.2	A Brief Layout of Top-level Concept Hierarchy in WordNet	34
5.3	System Modularity	36
5.4	System Architecture Design	38
5.5	An Example of Path Optimization	43
6.1	User Interface Page for Querying the System	46
6.2	The Result Page in Response to Query	48
6.3	Initial Folder Tree Display with Top Categories	50
6.4	Performing Drill Down on the Folder Tree	50
6.5	Cluster “hockey” at the Bottom of Navigating Path	51
6.6	Drilling Through Operation for Web Site Display	52
6.7	Hierarchical Folder Tree for Query “amazon”	55
6.8	The Relationship Between Cluster Properties and Clustering Threshold	65
6.9	The Relationship Between Cluster Precision/Topic Relevance and Clustering Threshold	66

List of Tables

5.1	Algorithm: Web Document Cluster Mapping	42
5.2	Algorithm: prune(tree)	43
6.1	Cluster Topics and Sizes of Search Result for “Edmonton Oilers” .	49
6.2	Differences Between Data Retrieval and Information Retrieval . . .	56
6.3	Queries and Their Domains	64
6.4	The Relationship Between Cluster Properties and Clustering Thresh- old	65
6.5	Cluster Property Analysis of Search Results from 13 Queries . . .	67
1	Cluster Property Analysis for Query “artificial intelligence”	77
2	Cluster Property Analysis for Query “Adidas”	78
3	Cluster Property Analysis for Query “amazon”	79
4	Cluster Property Analysis for Query “amino acid”	80
5	Cluster Property Analysis for Query “cheddar cheese”	81
6	Cluster Property Analysis for Query “Edmonton Oilers”	82
7	Cluster Property Analysis for Query “George Bush”	83
8	Cluster Property Analysis for Query “inline skates”	84
9	Cluster Property Analysis for Query “mutual fund”	85
10	Cluster Property Analysis for Query “Napster”	86
11	Cluster Property Analysis for Query “sodium benzoate”	87
12	Cluster Property Analysis for Query “Stephen King”	88
13	Cluster Property Analysis for Query “University of Alberta” . . .	89

Chapter 1

Introduction

With the wide use of the Internet, and the exponential growth of the World Wide Web, information retrieval and resource discovery from the Web is becoming more challenging. Today, the Web has rapidly become an alternative of traditional information repository, and the base for e-commerce business model. Digital libraries have replaced paper index cards for reference search; images, music, video clips are all made available on-line. More companies put their product/service databases on the Web for customer services and business transactions. The Web has become the key for the revolution of doing business, which aims at efficiency improvement and cost reduction.

The revolution that the Web has brought to information access is not so much due to the availability of information, but rather the increased efficiency of accessing information. The emergence of Web search engines has been helping Web users for information discovery from the Web. It is estimated that eighty five percent of Web users rely on search services to locate Web pages, and sixty percent of Web users use Web directories [LAW98]. However, the current crawling - indexing - ranking - querying by keywords model that all of today's search engines (such as AltaVista, Google, NorthernLight, Yahoo, etc.) adopt does not guarantee a perfect answer to the user's query. This is primarily due to the humongous size of the Web, which is still growing exponentially.

The motivation of our research in this project is from the observations of the following facts with regard to the status of today's Web search engines: First,

there is not a single Web search engine that is able to index the whole Web. Usually, a Web search engine crawls the Web starting from a few seed pages, and recursively follow the hyperlinks to download Web pages for indexing until such a hyperlink chain is exhausted. It is estimated that the largest capacity for a single search engine today covers about thirty percent of the Web [LAW98]. Since each Web search engine has its own seed pages for Web crawling, the Web coverage of different search engines are different. Therefore, if a single Web search engine is used for Web searching, a large chunk of information resource is never explored. Secondly, the long ranked list presentation of search results has become the *de facto* standard approach for organizing and displaying search result set. The intention of this approach is to return the ranked list of matches to the user, and leave it to the user to navigate and further search the results. However, depending on the search engine used, and user's query, the ranked list of document set returned by search engine could easily exceed ten thousands. The user would have to sift through this large document collection to find information relevant to his/her query. Even though search engine results usually have high recall by retrieving relevant pages from an index database, the precision is low. According to a survey, a majority of search engine users never go to the second page of search engine results [NMC01]. If the target information is among the rest of the list, chances are that some relevant information is buried in the long ranked list of document set, and never reaches the user. Therefore, the browsability of search engine results has to be improved in order to meet the increasing quality demand of Web users, and the rapid growth of the Web itself. In summary, today's crawler-dependent Web search engines face two major issues: the capacity to cover the Web; and the browsability of search engine results.

Since the size of the Web is beyond the coverage of a single Web search engine, Metasearch engines such as MetaCrawler [SEL96], Mamma, etc. have been designed to alleviate this issue. The concept of metasearch is somewhat different from a traditional Web search engine. The major difference lies in that metasearch engine never conducts Web crawling and indexing. It directs a user's query to multiple search engines, combines the search results, eliminates the overlap, and

re-ranks them before the final result is presented to the user. Metasearch engines significantly increase the coverage of the Web without taking their own resources for indexing the Web pages [LAW99]. In this project, we also adopted the same concept for larger Web coverage. However, metasearch approach does not touch the browsability issue caused by large search result set.

Although the long ranked list presentation of search engine results has become the *de facto* standard in Web search engine industry, the volume of matches that search engine is capable of answering to a query is overwhelming to the user. Obviously, the browsability of a ranked list is much reduced for a large dataset. Web search engines are relying heavily on the ranking algorithm to highlight the most relevant Web pages. However, a ranking algorithm would fail to satisfy the user with the same searching query but different intentions.

One way to approach this issue is by grouping search engine results into different categories. Each category can be considered as a sub-topic under the query term. It is up to the user which category he/she wants to browse. The significance of this approach is that it directs a user to his/her target information set by browsing categories instead of each individual match returned by Web search engine. In this project, we tried to apply the concept of such an approach for Web search engine results categorization. The key contribution of this research work is the exploration of constructing a hierarchical organization for Web document categorization using existing ontological knowledge. Also, we implemented a navigation scheme as the supplement of such a hierarchical organization of Web documents in support of user's interactive and progressive browsing of Web search results.

The rest of the thesis is organized as the following: Since the system we designed and implemented is ultimately a Web searching tool, Chapter 2 outlines the mechanism of Web search engines including metasearch engine, and the functionality of the major components of a Web search engine. Because a good Web search engine relies heavily on its unique ranking algorithm to distinguish itself from others and bring relevant Web pages to the top of the list, the major factors

that affect ranking algorithm are also analyzed in this chapter. Chapter 3 summarizes some related research work on Web document clustering and classification to improve the browsability of search results. The pros and cons of these related works are analyzed. Chapter 4 gives a brief introduction of WordNet - an online lexical database which is used as the initial ontological network for the construction of the hierarchical organization of Web search results in our project. Chapter 5 focuses on the system model, architecture design and implementation issues of our system. Chapter 6 uses some examples to illustrate the system behavior and major functions of the system. Also, this chapter describes the preliminary experiments we conducted for system evaluation and result analysis. Finally, Chapter 6 draws the conclusions from our research experience, and recommends some future work strategies with regard to the directions of research on this topic.

Chapter 2

Searching the Web

2.1 How Web Search Engines Work

2.1.1 Web Search Engine

A Web search engine is a software utility or program that is used to find information on the Internet, related to a specific subject that is being sought by the user [CHO98].

With the rapid growth of the World Wide Web, and its popularity, one of the major problems that Web users encounter is how to find target information from the Web efficiently and accurately. Web search engines have become one of the most successful and popular Web services since the very early stage of the Web development. Yahoo, Excite, Infoseek and Lycos are among the most successful Web search tools from the first generation of Web search engines. Today, the Web search engine industry is much more diversified. Major search engines like AltaVista, Ask Jeeves, Google, NorthernLight, and MetaCrawler all have their own specialty features to meet the needs of different types of users. In addition to general purpose search engines, there are also a large number of specialty search engines which were designed for searching specific domain areas [KIN00], such as news groups, multimedia, scientific papers, financial information, legal matters, etc. It is hard to tell how many Web search engines are in use nowadays. But their basic functionality and design fundamental are similar.

The term “search engine” is often used to describe both true search engines and Web directories. However, the difference lies in how listings are compiled. Today,

all major search engines provide both a Web search engine and a Web directory. A Web directory depends on humans for its listings [SHE00]. If one wants to get a Web page listed under a Web search engine's directory, he/she submits a short description of the Web site to the directory for review. If the editor of the Web search engine's directory likes the content and quality of the web site, it is chosen to be listed. A directory search is a hierarchical search that starts with a general subject heading and follows with a succession of increasingly more specific sub-headings. It is also known as a subject search.

In contrast to directory, search engine creates its listings automatically. A server or a collection of servers dedicated to indexing Internet Web pages, storing the results and returning lists of pages which match particular queries. A search engine has three major elements. The first element is a robot, also referred to as spider, crawler, or wanderer. The robot is a program that traverses the World Wide Web to discover, gather, index, and verify documents and links to be included in a database. The robot crawls the Web by starting from visiting a Web page, reading it, and then following links to other pages. The second element of a search engine is a database that is an indexed collection of information gathered by the robot. A robot records all indexed information in its database which may include Web addresses, titles, headers, words, snippets, abstracts, sizes or even full texts of documents. The third element of search engine is a software that takes the user's query, sifts through the pages recorded in the index to find matches, and ranks them in the order of what it believes is most relevant [SUL98].

2.1.2 Web Crawling and Indexing

A robot is a program that automatically traverses the Web's hypertext structure by retrieving a document, and recursively retrieving all documents that are referenced [CHO98]. A Web robot does not itself move from Web site to Web site. It simply visits Web sites by requesting documents from them. Web robots can be used for a number of purposes such as indexing, link validation, update monitoring, and mirroring. The major use of Web robots is indexing. A Web robot usually starts from a list of seed URLs which are the most popular Web sites with a lot of links,

and crawls the Web in a broad range [EAG96]. A robot downloads a page from the Web and scans it for links to other sites. Then it chooses a URL and jumps to another Web site. If the robot encounters a page that does not have any link, it goes one level back, and selects another URL. This recursive navigation process is conducted automatically by the robot to cover a broad range of the Web. Since Web search engines choose different seed URLs for Web crawling, they usually give different Web coverage with a certain degree of overlap among them. Once the Web robot downloads a web document, it parses the HTML document, indexes the Web document, and saves the index in a database for searching. Some robots parse the whole HTML document, and index all words appearing in the document. Other Web robots only index the HTML TITLE tag, META tag, or the first few paragraphs of the text [CHA99]. Figure 2.1 shows a typical architecture of Web crawler and indexer.

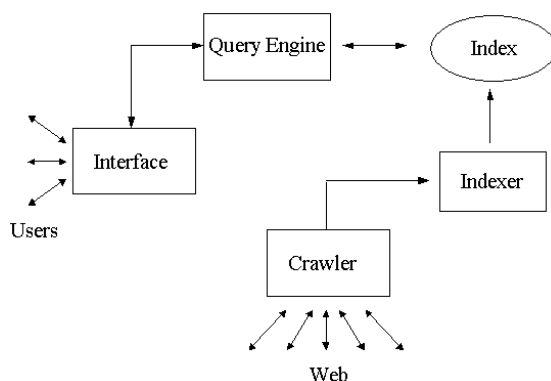


Figure 2.1: A Typical Web Crawler - Indexer Architecture [YAT99]

The frequent visits of Web robots can contribute tremendous network traffic increase to a Web site. Such a Web server overload may significantly affect the access speed of the normal visitors to the Web site. In order to alleviate such a problem, Web Robots offer facilities for Web site Administrators and content providers to limit what the robot does. This is achieved through two mechanisms [WEB01]. One of the mechanisms is the robot exclusion protocol. When a robot visits a Web site, it first checks for `http://.../robots.txt`. If it can find

this document, it will analyze its contents to see if it is allowed to retrieve the document. A Web site administrator can indicate which parts of the site should not be visited by a robot in the robots.txt file. The other way to prevent the Web site from indexed is using the robot META tag. A Web author can indicate if a page may or may not be indexed through the use of a specific HTML META tag.

2.1.3 Search Result Ranking

Ranking is very important to a Web search engine in order to enhance the quality of its search result. Nowadays, it is very common for the user to receive a result list with a few thousand matching Web pages in response to a particular query. However, the users' browsing behavior indicates that only the very top part of the search result list is examined, and the rest of the list is never browsed. Therefore, bringing the most relevant Web pages to the top of the search result list through ranking is crucial to meet the user's satisfaction. Each different Web search engines have their own ranking methods. The ranking method of a specific search engine is rarely disclosed in detail as it is considered top corporate secret. In general, the following factors are often used by Web search engines to create their own ranking algorithm.

Term Frequency and Location Ranking

At their core, the major search engines use location/frequency method for determining relevancy to a user's query [SUL98]. In fact, most search engines use the location and frequency of keywords on a Web page as the basis of ranking it in response to a query. A document that contains the keywords towards the top of the document -title, headline, or the first few paragraphs of text, and has the keywords repeated several times throughout the document is more likely to be deemed relevant to the query [YAT99]. Therefore, the document is more likely to have a higher ranking. Some Web site owners use spamming techniques that attempt to "spam" the search engines in order to improve their ranking. Spamming techniques are various. One common technique is "stacking" or "stuffing" words

on a page that allows a word to be repeated many times in a row in invisible text or tiny text. All major search engines detect spams. If they spot spamming technique, they may downgrade a page's ranking or exclude it from listings.

However, location and frequency are not the only factors used. Each search engine has a blend of techniques that go into their algorithms [ZHA00]. But location and frequency have tended to be the dominant factors. In addition to location and frequency, some search engines may give a page a relevancy boost based on the following factors.

Meta Tags Boosts Ranking

Some search engines use META tags as one component of their ranking formulas. A META tag is an optional line of HTML code in the HEAD section of the document. The tag's actual content gives a concise summary of the Web site. Since the content in META tags provides descriptive information about the Web site, search engines use META tags to help determine the content and value of the Web page content. Thus, search engines that support META tags often give pages that include tags a ranking boost if the inquiry terms appear in this area. Description and Keywords are the two most important META attributes to search engines. Each search engine algorithm evaluates META tags differently and some even do not consider them at all when ranking web sites [WEB01]. In order to avoid spam, some search engines give penalties to web sites that abuse the META tags for higher ranking. Among the major search engines, only Go and Inktomi seem to rank web pages based on meta tags [SUL01].

Reviewed Status Boosts Ranking

Some search Engines review Web sites, and choose some of them in a related directory. If the retrieved Web site is listed in the directory, the search engines assign a higher ranking to it. Both Excite and Infoseek use Reviewed Status to boost ranking of Web sites.

Link Popularity Boosts Ranking

Search engines can determine the popularity of a page by analyzing how many links there are to it from other indexed pages, and give pages with either lots of links or links from important sites a relevancy boost [HEN00]. Link popularity boost is especially effective to improve search engine's ability to move the official home page of a company or organization to the top of the ranked list of results.

Before the emergence of the Web, information retrieval techniques developed were mostly word based. Hyperlinks which are popularly used in Web documents provide a valuable source of information for Web information retrieval. Because of the widely adopted hyperlinks among Web pages, the Web can be viewed as a well connected graph with each Web page as a node, and each hyperlink as an edge. Such a graph presentation of the Web provides the fundamental for the use of link analysis. Therefore, link analysis becomes a very important ranking technique for today's search engine.

A hyperlink is a reference of a Web page B that is contained in Web page A. Web page B can be accessed from Web page A by clicking on the hyperlink. Typically, Web document authors tend to create hyperlinks either for navigational aid purpose or pointing to high quality pages that might be on the same topic as the page containing the link. Based on this observation, two assumptions are made with regard to the hyperlink. First, if there is a hyperlink between Web pages, it is more likely that the contents of these two pages share a similar topic; secondly, a link in Web page A to page B means a recommendation of Web page B by the author of page A. These two assumptions imply that a Web page is better and more important than others if it has a larger number of hyperlinks pointing to it. However, this theory fails to identify the importance of two Web pages if both of them have the same number of incoming hyperlinks. In reality, the Web page has better quality and is more influential if its referring pages themselves are more popular [BRI98]. Such an observation played a key role in the creation of the query-independent ranking technology PageRank. To deal with this issue, the PageRank of a Web page is computed by weighting each hyperlink proportionally to the quality of the page containing the hyperlink. The quality of a referring page is determined by using its PageRank recursively. PageRank is very efficient

to distinguish high quality Web pages from low quality pages [HEN00].

Direct Hits Boosts Ranking

Direct Hits is a system that measures what the users click on the search results in order to refine relevancy rankings [DIR01]. It “watches” what the users search for, then records which pages they visit from the normal search results. Over time, it develops enough data to know which pages are popular among the users, and adjusts the ranking accordingly. One problem with direct hit boost is spamming [SUL98]. Web site owners can click their own pages to boost rankings. Another problem is the fact that many users do not search deep. They usually only browse the pages listed in the top of ranked list of search results. Therefore, Web pages that are initially ranked beyond that have much less to be clicked by users, and thus are unlikely to be boosted.

Different search engines have different formulas to assign weights to different ranking factors mentioned above. Search engines keep their ranking algorithm secret as they are considered the most crucial technology for search engines.

NorthernLight is reported to use the following measures for ranking Web pages [SUL98]: statistical measures such as query term frequency, inverse document frequency of the term, and length of the document; word context information such as whether or not a word or phrase queried occurs in the document title; document classification based on NorthernLight’s patented technology for automatically classifying the Web and organizing results into Custom Search Folder; natural language analysis of the query; link popularity; and Document date.

The heart of Google search engine is PageRank, which relies on the uniquely democratic nature of the Web by using its vast link structure as an indicator of an individual page’s value [BRI98]. It uses weighted link popularity as a primary part of its ranking mechanism. Each page has a rank, based on the number of other pages linking to it and the importance of those pages. Google also makes extensive use of the text within hyperlinks. This text is associated with the pages to which the links point. In addition, Google provides some ranking boosts on

page characteristics. The appearance of terms in bold text, or in header text, or in a large font size is all taken into account. None of them are dominant factors, but they are figured into the overall equation for ranking score calculation.

2.1.4 MetaSearch Engine

One of the major challenges of Internet information retrieval is that an Internet search must be as complete as possible. Considering the size of the Web, and its exponential growth rate, there is not a single search engine that is able to index the whole Web. Google, the world's largest search engine allows its users to search 1.3 billion Web pages. However, it can only index about 560 million pages using full text, and the rest Web pages are partially indexed [GOO01]. According to a NEC research report [LAW99] of seven major Web search engines, the coverage of the engines is increasing slower than the size of the Web. If a single search engine can not conduct a complete Web search, one obvious way is to query multiple search engines. Metasearch engine technology was developed to deal with this issue.

Unlike search engines, metasearch engines do not crawl the Web themselves to build listings. It resides on top of the indexing based Web search engines in the information herbivore. It takes advantage of the infrastructure built by indexing based search engines, and moves up the information food chain by providing the users better search results with higher efficiency and less consumption of resources [ETZ96]. Metasearch engines allow searches to be send to several search engines all at once. The results are then blended together onto one page. Search result ranking is just as important to metasearch engines as it is to indexing based Web search engines. However, a metasearch engine has to rank its search results returned from multiple indexing based search engines. Therefore, the ranking scheme is different. MetaCrawler calculates a confidence score as the ranking of a retrieved Web page to show how close this Web page matches the query [SEL96]. A higher confidence score indicates a more relevant document. To calculate the confidence score for a retrieved Web page, MetaCrawler first distributes the confidence score returned by each Web search engine into the range from 0 to 1000 with the top pick having a confidence score of 1000. Then, MetaCrawler eliminates duplicates, and adds

the confidence score of the eliminated Web page to the score of the Web page that represents all of its duplicates. This way, MetaCrawler allows Web search engines that it uses for metasearch to vote for the best match. A Web page retrieved by multiple Web search engines most likely has a higher total confidence score than a Web page returned by only one Web search engine. The first metasearch engine MetaCrawler was designed by E. Selberg and O. Etzioni [SEL95], and its flow control chart is shown below.

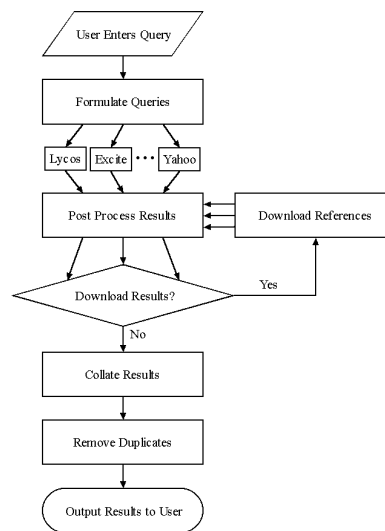


Figure 2.2: MetaCrawler Flow Control [SEL96]

There are two types of metasearch engine: one is client-based metasearch engine, which is running as a client on the user's PC; the other type is server-based metasearch engine, which is operating as a server, and answering queries of many users. Since client-based metasearch engines encounter the last-mile-problem and the update-problem [SBW98], which lead to slower search result return and unnecessary network traffic, server-based metasearch engine is heavily favored. In order to be qualified as metasearch engine, the following criteria [SBW98] have to be matched:

1. The metasearch engine should direct a user's query to selected search engines in parallel.

2. The results of the different search engines should be merged.
3. Duplicates from different search engines should be eliminated.
4. The specification of the selected search engines should be hidden to the user.
5. The merged search results from multiple Web search engines should be ranked.

Chapter 3

Related Work

Web search engines have been working hard to keep up with the rapid growth of the Web by improving the way web documents are indexed, searched and ranked. However, the presentation style of a search result - a long ranked list has not changed much since the emergence of Web search engines. The philosophy behind this approach is that all the matches from searching should be presented to the user, and the freedom should be given to the user to target his/her interest by browsing the whole set of possible matches. Such a presentation style suffers from two major drawbacks. First, even though search engine can achieve high recalls by returning all the possible matches to the user, the precision is low. There is only a small part of the retrieved documents that shares the user's interest. Secondly, the user has to sift through the long list of research results from search engine to find out the target of his/her particular interest. Considering the low precision of search engine results, such a task is certainly a big load of burden to the user. Even worse, instead of browsing through the whole ranked list, most of search engine users only check out the very top part of the list. If the user's target is not highly ranked by the search engine, chances are it will never reach the user. The necessity of improving the browsability of search engine results has increasingly drawn attention of researchers. Some promising and interesting approaches are presented in the following:

3.1 Clustering Approach

3.1.1 Flat Clustering of Web Documents

Zamir [ZAM97] proposed to present Web search engine results with clusters. Each of the clusters is assigned a topic to summarize the documents included in the cluster. Therefore, the users only need go to the cluster(s) with the topics of their interests instead of sifting through the full list of the documents returned by search engine. In this approach, an agent is built on top of search engine(s) to process the snippets from search engine results into clusters on the fly. A suffix tree structure [UKK95] was used for this purpose. A suffix tree clustering algorithm can efficiently identify shared phrases among documents [ZAM97]. It provides a mechanism to take advantage of the sequential relationship of words in documents to create meaningful clusters to assist online browsing. The documents with the shared phrase are grouped into one cluster, and the shared phrase is used as the topic of the cluster as a summary of all the documents in the cluster. Clusters containing similar set of documents will be merged to reduce small clusters, and thus simplify the presentation of final search result. However, there is a tendency that suffix tree clustering produces too many clusters with single-word topics. Compared with multi-word phrase topic, in most of the cases, a single-word topic tells much less about the content in the cluster, and thus much reduces the significance of such an approach.

Modha and Spangler used a similar concept for hypertext clustering. Documents are considered similar if they share the same words, out-links or in-links. The document similarity is measured by a geometric hypertext clustering algorithm [MOD00]. Each document cluster is annotated by summary, keywords, review, references, citations and breakthrough, which contain high quality, typical information resources, and efficient for document navigating. The summary and keywords of clusters are derived from words in documents; reviews and references are from out-links; and breakthrough and citations are from in-links. The approach of this algorithm is very similar to Hypursuit. But instead of generating hierarchies of clusters, it simply returns some document clusters given a set of hypertext document from search engine in response to a query.

3.1.2 Hierarchical clustering of Web Documents

Another application worth mentioning is the Scatter/Gather system [CUT93], which introduced clustering as a document browsing method. It used a linear time clustering algorithm (Buckshot) to cluster a corpus of documents and presented these clusters to the user. The user selected one or more clusters for further investigation, and the documents in this subcollection were then reclustered in an iterative and hierarchical manner. It was the first to use fast, linear time algorithms and thus was able to cluster sufficiently fast a reasonable number of search results. A similar approach was also used by a system from www.vivisimo.com. The merit of such an approach is that it is able to create clusters with very intuitive topics for browsing. However, since the shared words or phrases were used as the base for clustering, this approach would produce excessively many top-level clusters. Another drawback of this approach is that even though the search results are arranged hierarchically, there are no logical and semantic relationships whatsoever among clusters either horizontally or vertically.

Principal Direction Divisive partitioning algorithm [BOL98] was proposed to automatically generate hierarchical topical taxonomy of a document set. The algorithm constructs a binary tree hierarchy of clusters starting with a single cluster encompassing the entire document collection, and recursively split clusters based on a linear discriminant function derived from the principal direction until a desired number of clusters is reached.

Hypersuit, a hierarchical network search engine adopted a content-link clustering algorithm for Web document categorization [WEI96]. Content-link clustering is based on the semantic information embedded in hyperlink structures and document contents to construct clustering hierarchies. This algorithm clusters hypertext documents using a similarity function that relies on both term similarity and hyperlink similarity factors. The hyperlink similarity between two documents is measured under the following notions: 1. The similarity of the two documents varies inversely with the length of the shortest path between them; 2. the similarity between two documents is proportional to the number of ancestors and descendants respectively that the two documents both have in common. The term similarity

is measured based on traditional method of using weighted terms. The hierarchy of document clusters can also be used for interactive information browsing.

3.2 Machine Learning Approach

3.2.1 Web Document Classification

Another type of approach for improving browsability of web documents retrieved by search engine is classifying web documents into different categories. Instead of presenting the long ranked list of retrieved documents to the user, categories, which these documents belong to, are given to the user for easy browsing. The major difference between web document classification and clustering is that the former approach requires a machine learning process. Classifiers are trained beforehand, and used for categorizing web documents. Classification of Web documents is different from categorizing classical document collections. Whereas documents of classical collection all have a similar structure, Web documents are rather heterogeneous. Therefore, classification algorithms, which perform well on classical document collections, will have problems when applied to Web documents.

One research project in this area used the idea of category-specific centroid vectors [CPK00]. The centroid of all document vectors belonging to the same category is used as a presentation of this category. In order to cope with heterogeneity of Web documents, the texts of all Web documents in a category are concatenated to achieve a so-called metadocument. Then, a vector is derived for this metadocument by applying the standard tf-idf method [JOA97]. For a given collection of Web documents, the total number of metadocuments equals to the number of categories. To classify a Web document, first all terms of this document are extracted, and the n best terms are selected as query for retrieving the most similar metadocuments. However, the classification accuracy of this approach still need to be improved, and it does not support the hierarchical structure of the classification scheme.

3.2.2 Hierarchical Classification of Web Documents

The classification scheme mentioned above ignores the hierarchical structure and treats the topics as separate classes. As a result, a large number of classes will be created, and a huge number of relevant features needed to distinguish them. Most recently, a machine learning algorithm has been investigated to generate a hierarchy of classes for Web document classification. Naive Bayesian classifiers were trained to map Web documents into categories of Yahoo's hierarchy [KOL97]. In this approach, each Web document is represented as feature-vectors using a bag of words representation including not only single words, but also up to 5-words. Also, Feature selection technique is used to increase the accuracy [DUN98]. The hierarchical structure decomposes the learning task into a set of sub-tasks corresponding to individual categories. For each sub-task, a classifier is constructed. During training, the positive examples in a particular category node are propagated to its higher-level category node. The final result of learning is a set of specialized classifiers. However, the training process is lengthy, and the accuracy for document classification needs to be improved.

3.3 Web Search Result Visualization

In information retrieval, set of documents are stored and categorized in order to allow for search and retrieval. The approaches described in the previous sections were designed to deal with large dataset of retrieved Web documents from search engines. Retrieved Web documents are grouped according to their relevance computed one way or another. The knowledge or metadata commonly shared by all the documents within a group is extracted, and used to represent the group of documents. Such a modeling of document categorization aims at efficiently managing a large amount of information using extracted higher-level textual metadata. Compared with such a traditional approaching concept, search result visualization has been studied to manage and display information in a graphic environment. Such an effort is to make complex information easier to understand and browsing. Some of the related work in this area is summarized below.

Bead is a system for the graphically-based exploration of information [CHA93].

The underlying notion of the system is using spatial proximity to represent similarity of documents. The relationships between documents in a corpus is represented by their relative spatial positions with similar documents close to one another and dissimilar ones further apart. The emergent structure is a model of the corpus: a landscape or map of the information within the document set. A landscape-like scene is used to visualize patterns in the high-dimensional information space, and thus make interaction with a database of information more graphically oriented. The model used in Bead emphasizes patterns of thematic similarity as estimated by similarities in word usage [CHA96]. Individual documents are categorized into the patterns according to the particular words used within them. The open landscape gives an overview of the patterns and structure of the corpus. Documents matching the query term are marked in a particular color. Matching documents with high relevance are placed close to the query term. Documents with low relevance lie on the periphery of the island-shaped landscape.

XFIND is another tool for search result visualization [AND01]. XFIND adopts a gatherer-indexer-broker architecture. The gatherer is responsible for gathering information from the Web, pre-processing the HTML documents, and extracting metadata from retrieved documents. The metadata include title, snippet, keywords, headings, and links etc. The generated metadata are sent to indexer for indexing. The XFIND broker provides search functionality for user to find target information. In addition to ranked list, XFIND also present search results in an interactive scatterplot. Documents are plotted according to two of their metadata attributes corresponding to the x and y axes. Due to the interactive nature of the plot, most of the metadata attributes generated by XFIND gatherer can be mapped to either axis or to icon size or color. Another functionality of XFIND is its dynamic thematic clustering of search result set. The search result set is first clustered. The cluster centroids are then distributed randomly in the viewing window. The documents belonging to a particular cluster are placed in a ring around the centroid. By this way, documents similar to one another are attracted towards each other.

Umap is a metasearch engine that provides a visualization tool for overview of

hundreds of search engine results [UMP01]. In addition to display the ranked list of search result set, and a list of keywords for each search result, Umap adopts a dynamic map technology to integrate all the keywords extracted in a dynamic graphical interface. The map allows the user to visualize and understand the nature, the context and the coherence of the texts along with associated themes.

ThemeScape is another software that gathers information from the Web, extracts themes and topics, and creates an interactive map of information [THE99]. The topographical map shows the overview of the content within large quantities of Web documents. In the graphical map, major themes rise from the surface indicating a cluster of documents around a given topic. The distance between themes indicates how close they are related. All the visual cues shown in the map attempt to help the user understand the context of the information.

Chapter 4

WordNet

4.1 WordNet - an Online Lexical Database

WordNet [WOR01], an electronic lexical database was designed by a group of researchers at Princeton University. It is based on psycholinguistic and computational theories of human lexical memory, and is considered as the most important resource available to researchers in computational linguistics, text analysis, and other related areas. Presently, WordNet contains approximately 95,600 different word forms (51,500 simple words and 44,100 collocations) organized into some 70,100 word meanings, or sets of synonyms. The most obvious difference between WordNet and a standard dictionary is that WordNet divides the lexicon into five categories: nouns, verbs, adjectives, adverbs, and function words. Nouns are organized as topical hierarchies, verbs are organized by a variety of entailment relations, and adjectives and adverbs are organized as N-dimensional hyperspaces. The most ambitious feature of WordNet is its attempt to organize lexical information in terms of word meanings, rather than word forms [MIL95].

4.2 Structure of WordNet

Words and Concepts: There are two kinds of building blocks in WordNet: word forms and concepts. A word form is essentially a word in its original form. a concept represents a set of synonyms (synsets) that share the same sense or meaning. Therefore, a word meaning or synset in WordNet generates a concept that represents a set of words, and may cover a group of other concepts semantically.

Basic relations of WordNet: WordNet makes distinguish between conceptual - *semantic relations*, which link concepts, and *lexical relations*, which link individual word forms. The smallest unit is the word/sense pair identified by a sense key. Word/sense pairs are linked through WordNet's basic lexical relation, synonymy, which is expressed by grouping word/sense pairs into synonym sets or synsets. Each synset represents a concept. Two words are synonyms if they have a designated relationship to the same concept. Synsets (concepts) are the basic building blocks for hierarchies and other conceptual structures in WordNet [MIL95].

Synonymy is the most important lexical relation that WordNet uses to organize word forms. Two expressions are synonymous if the substitution of one for the other never changes the truth value of a sentence in which the substitution is made. Based on the synonymy relationship, words are organized into synsets, and thus concepts are generated accordingly. Wordnet is organized by the semantic relations of concepts. Unlike the lexical relation, semantic relations in WordNet differ depending upon syntactic categories.

4.3 Syntactic Categories in WordNet

WordNet partitions English words into four syntactic categories: noun, verb, adjective and adverb. Word forms in each syntactic category are organized into synsets, each representing one underlying lexicalized concept. Different semantic relations link the synsets. The definition of synonymy in terms of substitutability makes it necessary to partition WordNet into nouns, verbs, adjectives, and adverbs. That is to say, if concepts are represented by synsets and if synonyms must be interchangeable, then words in different syntactic categories can not be synonyms.

4.3.1 Nouns in WordNet:

Definitions of common nouns typically give a superordinate term plus distinguishing features. This information provides the basis for organizing nouns in WordNet. In WordNet, nouns are categorized into a hierarchical organization. First, synonym sets (synsets) are created for noun word forms according to their lexical

relations. Each of these groups of nouns (synset) is considered as a unique concept in WordNet, and the semantic relationships are established among synsets based upon hyponymy relation between two concepts. Thus, a hierarchical organization of nouns is created by applying the lexical relation (synonymy) among words forms, and semantic relation (hyponymy) among concepts. In principle, all nouns are contained in a single hierarchy, with the topmost, or the most generic level being empty. Since this high level, abstract generic concept carries little semantic information, and has little practical use, WordNet actually partitions all nouns into 25 high level top categories [MIL90]. Each of these categories is assigned a generic concept, and treated as a separate hierarchy. These separate hierarchies vary in size and are not mutually exclusive. They cover distinct conceptual and lexical domains. The depth of the hierarchy is limited within a dozen levels.

- | | |
|----------------------------|---------------------------|
| 1. {act, action, activity} | 14. {natural object} |
| 2. {animal, fauna} | 15. {natural phenomenon} |
| 3. {artifact} | 16. {person, human being} |
| 4. {attribute, property} | 17. {plant, flora} |
| 5. {body, corpus} | 18. {possession} |
| 6. {cognition, knowledge} | 19. {process} |
| 7. {communication} | 20. {quantity, amount} |
| 8. {event, happening} | 21. {relation} |
| 9. {feeling, emotion} | 22. {shape} |
| 10. {food} | 23. {state, condition} |
| 11. {group, collection} | 24. {substance} |
| 12. {location, place} | 25. {time} |
| 13. {motive} | |

Figure 4.1: Top 25 Categories(represented by SynSet) of Nouns in WordNet

Figure 4.1 shows the 25 categories of nouns in WordNet, each category is represented by the synset of the top level concept. Starting from the top level of each category, hyponyms at different sub-levels can be found. Although hyponymy gives rise to the WordNet hierarchy, other features such as meronymy (parts of) and holonymy(is part of ...) that distinguish one concept from another can also be found in WordNet.

Given a noun using “*dog*” as an example, WordNet is able to return all senses

and their corresponding synsets as shown below:

1. *dog, domestic dog, Canis familiaris* – (a member of the genus *Canis* (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds; “the dog barked all night”)
2. *frump, dog* – (a dull unattractive unpleasant girl or woman; “she got a reputation as a frump”; “she’s a real dog”)
3. *dog* – (informal term for a man: “you lucky dog”)
4. *cad, bounder, blackguard, dog, hound, heel* – (someone who is morally reprehensible; “you dirty dog”)
5. *pawl, detent, click, dog* – (a hinged device that fits into a notch of a ratchet to move a wheel forward or prevent it from moving backward)
6. *andiron, firedog, dog, dogiron* – (metal supports for logs in a fireplace; “the andirons were too hot to touch”)

If the first synset (*dog, domestic dog, Canis familiaris*) is chosen to find the hypernyms, which is “a kind of” like relation of this synset, WordNet will return a path to the top level of the category with a gradual generalization of concept.

dog, domestic dog, Canis familiaris

⇒ *canine, canid*

⇒ *carnivore*

⇒ *placental, placental mammal, eutherian mammal*

⇒ *mammal*

⇒ *vertebrate, craniate*

⇒ *chordate*

⇒ *animal, beast, brute, creature, fauna*

⇒ *life form, organism, being, living thing*

⇒ *entity, something*

For a hyponym search of the synset - dog, domestic dog, *Canis familiaris*, WordNet will give all the concepts that have a "... is a kind of" relation to the synset. Each of the resulting synset is a specialization of the synset - dog, domestic dog, *Canis familiaris*.

dog, domestic dog, Canis familiaris

⇒ *pooch, doggie, doggy, bow-wow*

⇒ *cur, mongrel, mutt*

⇒ *lapdog*

⇒ *toy dog, toy*

⇒ *hunting dog*

⇒ *working dog*

⇒ *dalmatian, coach dog, carriage dog*

⇒ *basenji*

⇒ *pug, pug-dog*

⇒ *Newfoundland*

⇒ *Great Pyrenees*

⇒ *spitz*

⇒ *griffon, Brussels griffon, Belgian griffon*

⇒ *corgi, Welsh corgi*

⇒ *poodle, poodle dog*

⇒ *Mexican hairless*

4.3.2 Verbs in WordNet:

The relationships between verbs are all forms of a broadly defined entailment relationship: troponymy establishes a verb hierarchy (verb A has the troponym verb B if B is a particular way of doing A). WordNet contains over 21,000 verb word forms and 8,400 word meanings (synsets). WordNet divides verbs into 15 entailment on the basis of semantic criteria. The 15 entailments are listed as below: *verbs of bodily function and care; verbs of change; verbs of communication; competition verbs; consumption verbs; contact verbs; cognition verbs; creation verbs; motion verbs; emotion verbs; stative verbs; perception verbs; verbs of possession; verbs of*

social interaction; and weather verbs [FEL93B].

Like nouns in WordNet, verbs are organized into synsets to form a concept based upon lexical relations within each entailment. Unlike the organization of nouns, verb concepts are not organized into hierarchical structure. The different relations that organize verbs can be cast in terms of lexical entailment. Entailment is a semantic relation. It refers to the relation between two verbs V1 and V2 that holds when the sentence someone V1 logically entails the sentence someone V2. Thus, WordNet can capture two semantic relationships of verbs: hypernymy and troponymy. Hypernymy relation of verbs is a “verb is one way to ...” like relation. Troponymy relation of verb is a “particular ways to verb” like relation.

Use synset “*go, go away, depart*” as an example, its hypernyms would be:

go, go away, depart
 \Rightarrow *exit, go out, get out, leave*
 \Rightarrow *move*

and its troponym would be:

go, go away, depart
 \Rightarrow *shove off, shove along, blow*

4.3.3 Adjectives in WordNet:

WordNet contains approximately 19,500 adjective word forms, organized into 10,000 word meanings (Synsets). WordNet divided adjectives into two major classes: *descriptive adjectives* and *relational adjectives* [FEL93A]. Each of these two classes is distinguished by the particular semantic and syntactic properties of its adjectives.

A *descriptive adjective* is one that ascribes a value of an attribute to a noun. The basic organization of descriptive adjective among descriptive adjectives is antonymy. Another kind of adjective is *relational adjective*, which means “relating/pertaining to, or associated with” some noun. Relational adjectives differ from descriptive adjectives in that they do not relate to an attribute, and relational adjectives do not have antonyms. Thus, relational adjectives are incorporated into clusters that characterize descriptive adjectives.

WordNet system is an online lexical database system that has four parts: the WordNet lexicographer's source files, the software to convert these files into the WordNet lexical database; the WordNet lexical database; and a suite of software tools used to access the database [BEC90]. The software programs and tools were written using C language, Unix utilities, and shell scripts. Applications can access the rich lexical database of WordNet through its function library for the purpose of text retrieval improvement [GON98]. In this project, we have used the ontological organization of noun words of WordNet, and its corresponding function libraries to help us categorize Web documents into a hierarchical organization.

Chapter 5

Architecture Design and Implementation

5.1 Issues and Approach Concept

The development of Web search engine has been one of the most dynamic area in the past several years due to its popularity among Web users, and strong demand for high quality online search services. Today, the largest Web search engine is capable of indexing more than one billion Web pages compared with only a few million Web pages three years ago [GOO01]. Web searching is usually very fast. Most of Web search engines can answer a user's query within a few seconds. Thanks to the progress of Web document ranking technology, search engines have been trying to bring the most widely visited and most referenced Web pages to the users in the first displayed result page. Hypertext document ranking has gone far beyond the tf-idf similarity measurement which is widely used by traditional information retrieval system. The structure of the Web [BRI98], metadata information within HTML document, the content and reference of hypertext linkage [HEN00], and the user's browsing behaviour [SUL98] all become part of the equation for search engine ranking strategy.

However, one of the major issues that today's Web search engine is facing lies in the organization and presentation style of its returning search results. Almost all major Web search engines adopt the ranked list of matching document format, which generally contains title, snippet, URL, and size of each Web page. A snippet is a short description of a Web page that is returned by a Web search engine as

a summary of the Web page. The ranked list organization style for search results has been used from almost the very beginning. Clearly, as the Web grows larger by the second, the dominating ranked list scheme is creating a bottleneck for Web search engines to help users locate target information. Any Web search engine user would tell you how overwhelming it is when search engine returns a list with few thousands matching Web pages. This is especially problematic to the users because of the following facts. First, in many of the cases, there is a certain degree of gap between the user's query term(s) and the true intention behind the query term used for searching. This does not just apply to the naive search engine users. Such difference and ambiguity is also hard to express or sometimes inexpressible even to experienced search engine users. Secondly, Web search engine satisfies user's query by returning a ranked list of all matches from its indexing database. Even though the recall is very high, the precision against the user's querying intention is low [ZAM99]. Despite Web search engine reliance on ranking to bring the most relevant and popular Web pages to the top of the search result list, in many cases, search engine's ranking algorithm can not reflect the user's querying intention. Therefore, the target pages are scattered and mingled in the pile of results. Thirdly, according to a survey of user's browsing behaviour, up to seventy-five percent of search engine users never go to the second page of search engine results [NMC01]. A large portion of target information never finds its way to reach the users, even if it is included within the ranked list of results. The consequence of the issue resulting from the combination of these facts is that the advantage of high technology developed for indexing, searching, and ranking is not fully reflected, and thus never fully taken advantage of by search engine users. As the impact of the Web grows larger, we believe it will increasingly compromise the role that Web search engines play in the Internet era. Therefore, changing the way a Web search engine presents its search results to improve its browsability is becoming inevitable.

Instead of the flat arrangement of a ranked list of search results, we believe that the most desired paradigm for organizing Web search engine results, making it comprehensive to the users, is by categorizing different documents according

to their topics, where topics are organized in a hierarchy of increasing specificity. Such an approach not only applies the concept of abstraction to managing a large amount of information, but also build the foundation for the mechanism of interactive user browsing.

5.2 System Model

Based on such an approach, an independent software agent is proposed. This software agent takes a user's query term(s) from a Web browser, then sends the query to the server to trigger a server side script, and the server side script distributes the user's query to multiple Web search engines for a metasearch. The search results are collected and pre-processed to create index files. The index files is used to construct a hierarchical organization of search results based on an ontological approach. The final results are posted back to the user's Web browser. The user can interact with the tree-like organization of search results through OLAP operation, which supports progressive navigation.

Ideally, such an agent should possess the following functionalities [CUI01]:

1. A metasearch engine on top of multiple Web search engines;
2. an independent entity operating separately from Web search engine;
3. a hierarchical organization of search results based on ontological approach;
4. a navigating scheme allowing user's interaction in a progressive manner;
5. and, a friendly user interface.

There are two major functional components of the proposed software agent as shown in Figure 5.1. The first component(Web Search Component) takes the user's query, and performs Web searching. The second component(Main Processing Component) is responsible for reorganizing search results, and providing a mechanism for browsing through user's interactive navigation.

5.2.1 Independent, MetaSearch Approach

In this research, we create a metasearch engine on top of multiple Web search engines. It sends user's query to several user specified Web search engines, such as

AltaVista, Google, Yahoo, etc., and take the combined search results as input for further processing. Since we mainly focus on the reorganization and browsability of Web search results, metasearch seems a natural fit for the project.

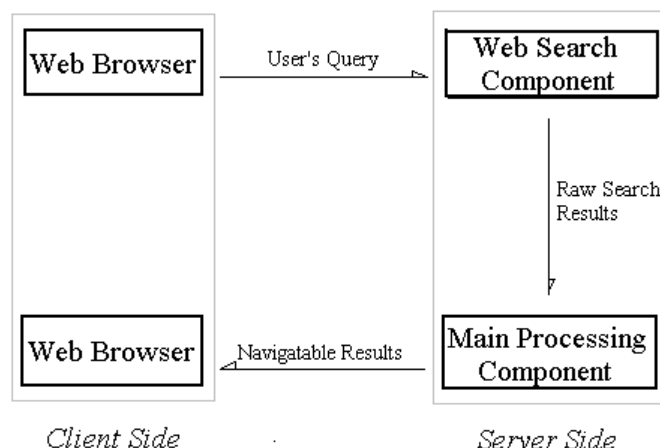


Figure 5.1: System Component Modeling

The Internet has become so large so fast that any sophisticated search engine can only covers part of the Web's vast information reservoir. According to a study, the Web is about five hundred times larger than the maps provided by popular search engines like AltaVista, Lycos, and Yahoo [TAP00]. Each Web search engine has its own Web coverage with a certain amount of overlap among them. Using a metasearch approach by taking advantage of different Web coverage of multiple search engines, we would have a larger representation of the Web. Also, metasearch approach provides the users the convenience to browse search results from multiple search engines in a unified interface [SBW98].

In addition, we believe that this agent should be independent from Web search engines for efficiency purpose. Web search engines handle millions of user's queries per day, the resources committed to each individual query is very limited. Thus, the proposed agent should operate on a separated machine, which receives search engine results as input, creates its own presentation of results, and presents it to the user.

5.2.2 Ontological Approach for Search Result Organization

How to create a hierarchical organization of search results is an interesting issue of this research. As mentioned in the last section, Web document clustering and classification have been investigated to achieve this goal. In the clustering approach [CUT92], Web documents that share the same phrase or word are categorized into the same cluster, and this process is recursively conducted to create a hierarchy of multi-layer clusters. Since the construction of the hierarchy is simply based on text processing, there is no semantic ties among different levels of clusters in the hierarchy. The classification approach [KOL97], which used ontological knowledge to build such a hierarchy tends to create a much clear logic for navigating. Classifiers were trained, and used to map Web documents into the hierarchy. Since the knowledge used for hierarchical classification is extracted from the learning algorithm, the training process is lengthy, and the accuracy is not quite reliable, especially considering the diverse nature of Web documents. The hierarchy created tends to be overly spread, and lacks depth.

An ontology is an agreement about shared conceptualization [MEE00]. It is a set of semantic information represented in a form which can be manipulated by software components. A good ontology is hard to built through automatic machine learning process since the linguistic issues that involved themselves are conceptual and abstract, and therefore hard to capture and model.

Here, we propose an ontological approach that uses WordNet - an online lexical database to help us construct such a hierarchical organization of Web search results. More specifically, we are taking advantage of the ontology of nouns provided by WordNet to categorize Web documents. In WordNet, nouns are organized as topical hierarchies with 25 top-level categories (Figure 5.2), and twelve levels in depth. There is a very strong semantic relationship (synonymy/hyponymy) between a parent and a child in the topical hierarchy of nouns in WordNet. For a given noun, WordNet provides an API to find its semantically related concepts. Thus, we can utilize the mechanism to map search engine results into the created-on-the-fly hierarchy. The detailed cluster mapping algorithm is discussed in Section 5.4.2.

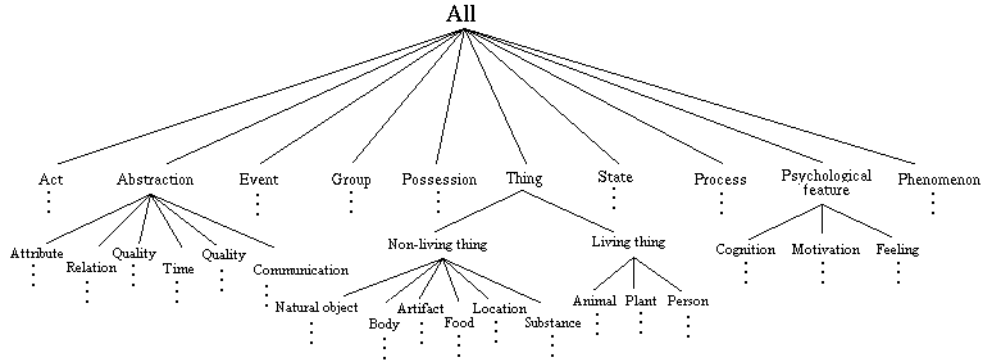


Figure 5.2: A Brief Layout of Top-level Concept Hierarchy in WordNet

5.2.3 User's Navigation Scheme

The hierarchical organization of search results is a tree-like structure with Web documents categorized in the leaf nodes. Each internal node of the tree represents a higher-level concept that semantically includes all of its child concepts. Thus, by following a path from the root to a leaf in this hierarchy, the user can start off a generalized concept, and gradually get to more specialized information. The philosophy behind the hierarchical organization of search results is information abstraction [HAN94], which assumes that most of the users prefer to browse general description of information instead of overwhelming details of information in the first view when searching for target information. Therefore, the most generalized concepts located in the very top level of the hierarchy will always be presented to the user first, and the system provides a mechanism to allow the user to set his/her own path towards the target information for details in a progressive manner.

Such a navigating scheme involves user's interaction, and thus gives the flexibility to the user to make his decisions along the way of navigation. In order to achieve such a desired goal, we introduce OLAP operations of data warehouse into our navigating scheme. Four major functions are implemented in order for the users to conduct interactive and progressive searching and browsing.

Drill Down: During the interactive searching process, the Drill Down opera-

tion allows the user to reach all the topics of more specialized information one level lower from a given topic. It gives the user a structural overview of the information one level below the current topic, and thus provides a guide to the user for in-depth searching.

Roll Up: Roll Up operation is the opposite of Drill Down. It allows the user to reach the parent concept from the current concept level. It is especially useful when the user wants to modify the searching path. By rolling up, the user is able to select a different but relevant topic for drilling down.

Show Detail: By default, the system is designed to return a simple and generalized top level categories of the search results to the user based on the philosophy of information abstraction. The Show Detail operation gives the user an option to see the detailed information including Title, Snippet and URL of each Web page categorized under the tree node wherever this tree node is in the hierarchy.

Drill Through: This operation allows the user to leave the hierarchical organization of search results, and access the actual Web page of his/her interest. However, the user has the option to come back where she/he left.

5.3 System Architecture Design

We adopted the client-server architectural style for system design. The major functionality of the system is operated on the server. The user can access the system through a thin client - Web browser, which allows the user to send queries, and navigate the search results. On the server side, the system is partitioned into four functional modules as shown in Figure 5.3: MetaSearch Module, Pre-processing Module, System Engine Module, and User Interface Module.

The MetaSearch Module is responsible for taking the user's query from Web browser through CGI, distributing the user's query term(s) to multiple pre-selected Web search engines, and collecting search results from different search engines. For each match returned by Web search engines, its URL, Title, and snippet are retrieved, and routed to Pre-processing Module for further processing.

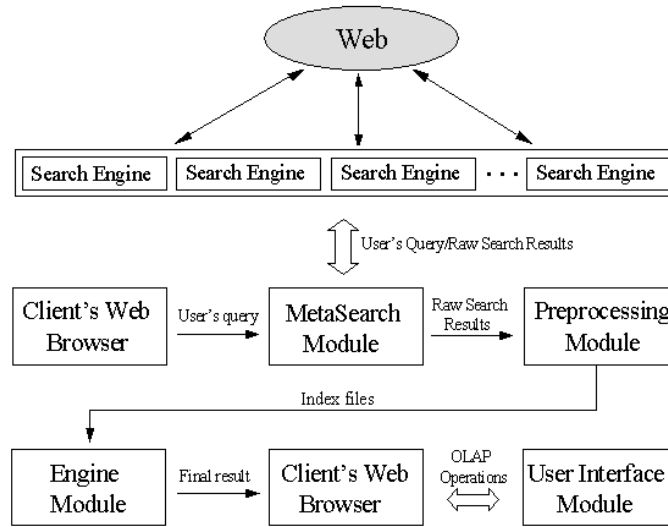


Figure 5.3: System Modularity

The major functionality of Pre-processing Module is to prepare the items retrieved, and create indices for the use of System Engine Module. During pre-processing, the retrieved information of each query goes through the following processing:

1. Eliminate all the duplicates of matches retrieved by more than one Web search engines. Since there is a Web coverage overlap among search engines, this step prevents the same Web page from being processed more than once by the system.
2. Remove all punctuation marks, numbers, and HTML tags.
3. Remove all the stop-words from each Title and Snippet. Since stop-words frequently appear in almost all text documents, they are not very useful to represent the content of the document. Therefore, they should be removed to reduce noise. The standard stop-word list containing 425 stop-words is used.
4. Stem the stop-word removed text. The purpose of stemming is to convert the words with different surffix but similar meanings to their base form, and therefore can be treated as one word. The Porter's algorithm is used.
5. Create an index file. Each retrieved URL is indexed using a vector of

keywords. The index file is our internal representation of the Web page.

6. Create an inverted index file. Given an extracted keyword, the inverted index file stores the frequency of its appearance, and all pointers as references of the location(s) that it appears among the whole retrieved items. The inverted file is used as the input of the System Engine Module for the construction of hierarchical organization of search results.

System Engine Module is the core of the system. The main functionality of the System Engine Module is constructing the hierarchical organization of search results, and optimizing the tree-like structure. Once the indices are generated, the System Engine Module is triggered. All records go through the Filter. Only those records with nouns as keywords, and with frequencies of appearance above a user specified clustering threshold are sent to the Mapping Manager for hierarchy construction. For each of the qualified record, the Mapping Manager takes the keyword(noun), refers to the ontological organization of nouns in WordNet, finds its precedents(hypernyms) at different levels of the ontology. Then, the keyword itself and its hypernyms are used to construct a unique path of the hierarchy with the most generalized hypernym mapped directly under the root (which contains the query term), and the keyword as the leaf of the path. After the completion of hierarchy construction, all keywords from inverted file are mapped into the leaves of the hierarchy with corresponding retrieved Web pages attached. The optimizer then prunes the tree to eliminate all the internal tree nodes with only one single direct child. This reduces the depth of the tree, and makes it more efficient for navigation.

The User Interface Module written in JavaScript is responsible for describing the tree structure according to its hierarchical organization, and translating it into HTML format for display. The user interface is a visual folder tree as shown in Figure 6.5. Each internal node is a folder of higher-level topical concept, which contains all the sub-topical concepts. Retrieved information is stored in the leaf nodes. Another function of User Interface Module is performing OLAP operations, which allows the users to navigate the folder tree in a progressive and interactive manner.

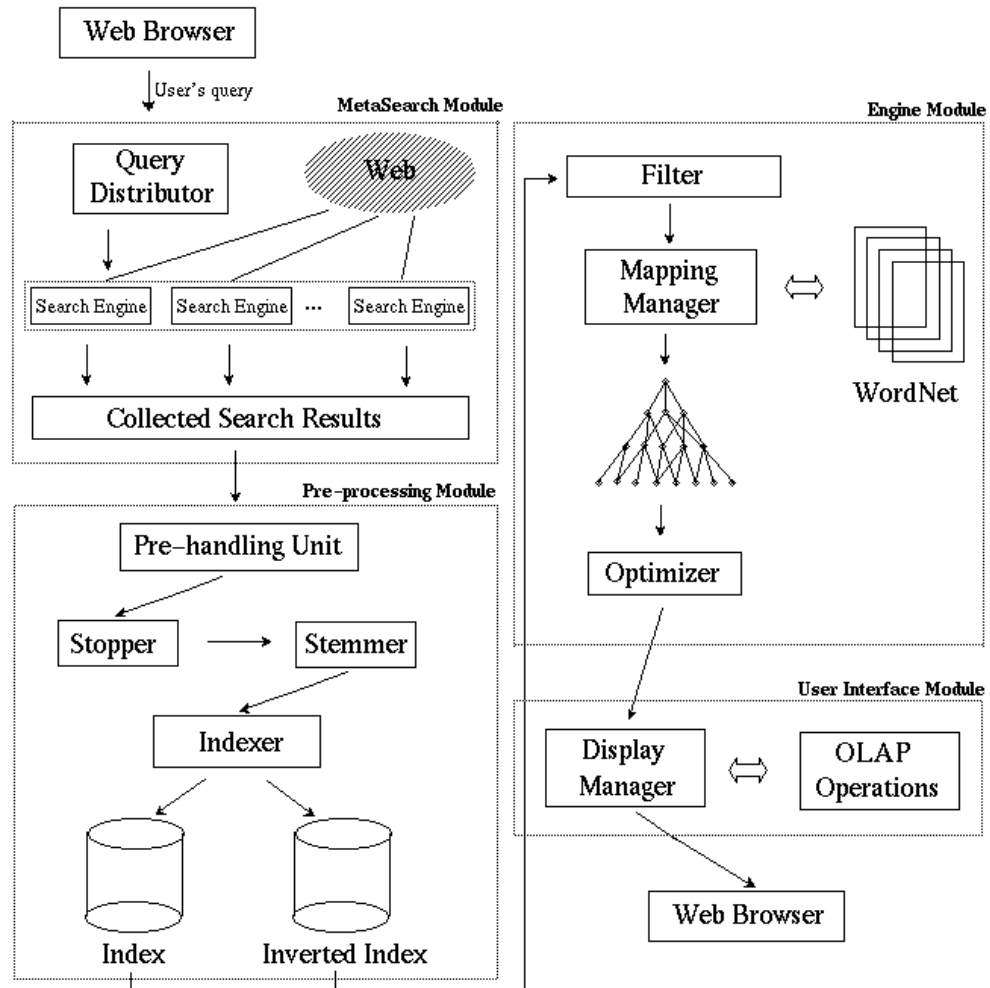


Figure 5.4: System Architecture Design

5.4 Implementation Issues

Based on the system model and architectural design, the implementation phase is much more detail oriented. There are several implementation issues that are worth mentioning. The dealings and tradeoffs of these issues aimed at making the system more efficient, accurate, and convenient to use.

5.4.1 MetaSearch Module

Indexing: Since our system was built on top of multiple Web search engines, we were able to direct the user's queries to these Web search engines, and collected and merged search results from them for our own use without doing any off-line web crawling, indexing, and maintaining a huge database of Web pages. However, we did need to build an index of the search results returned by the selected Web search engines for hierarchical clustering use later on. Since the metasearch is conducted online in response to a user's query, such an indexing process, which relies on the returned search results, also needs to be performed online, and requires speed as well as accurate representation of the content for each Web page. Instead of using the whole textual content of the Web pages for indexing, which would give a more accurate representation of the actual Web page at the cost of a much slower process, we considered partial indexing for each Web page. As a trade-off, it would significantly reduce the amount of time for indexing to achieve the speed required for online services. In order to minimize the accuracy compromise, we chose the Title and Snippet of each Web pages returned by the search engines for indexing. Since the title of a Web page is the reflection of its theme, and snippet is widely considered as the highlight of its content, we believe that the combined text of title and snippet is a good summary of a Web page, and thus is our best choice for indexing a Web page. Our experiment data (in Chapter 6) show that such an approach for indexing can generate clusters of Web pages with very good qualities.

Selection of Web Search Engines: Another issue is the selection of Web search engines that were used for metasearch. Intuitively, as many Web search engines as possible should be included since the purpose of metasearch is increasing Web coverage. Snippet quality is a crucial factor for Web search engine selection in our implementation. However, the quality of snippets from different search engines are vary. Generally, Web search engines generate Snippets by extracting the content of the META tag in the HTML file. If no information is found there, some search engines simply grab the first few lines from the Web page, and other search engines tend to extract a few lines of text from the Web page, where the query term appears. Also, some search engines return shorter snippet than others. Overall, we found that Google, Excite, and Yahoo return better quality snippets.

5.4.2 System Engine Module

Cluster Mapping: Clusters of metasearch results are generated based on the information stored in the inverted index file which is produced by the metasearch module. Each cluster has a topic as the summary of all the Web pages in the cluster. The construction of hierarchical organization of these clusters relies on cluster topic mapping using the hierarchical framework of noun’s organization provided by WordNet. Since verbs and adjectives in WordNet are not organized into a semantic hierarchy as nouns do, only those cluster topics that are nouns are considered by the Mapping Manager for building the hierarchy. For each of the noun cluster topics (T), the higher level concept (C1) of T is retrieved according to the “IS A” semantic relationship of WordNet. By the same token, the higher level concept (C2) of C1 is retrieved from WordNet. This process continues recursively until the most generalized concept (Cn) of T is retrieved. Then, the retrieved concepts Cn, Cn-1, ... C2, C1 and T in that order are used to construct a hierarchical tree path from the root. The cluster topic T is mapped at the bottom of the path, then the cluster of Web pages related to this topic is allocated into the hierarchical tree accordingly. However, WordNet is a very rich lexical database. For each noun, it usually returns multiple senses. For each sense, we can generate an

unique tree path following the procedure. As shown in the following example, the noun word “*song*” has five senses referring to in WordNet. For each of the five senses as shown below, there is an unique path starting from the most generalized category towards more specialized concepts, and ending up with the word “song” generated according to the hierarchical “IS A” relationship organization of nouns from WordNet.

Five senses of “*song*”:

Sense 1: *A short musical composition with words.*

Path: *entity, something \Rightarrow object, physical object \Rightarrow artifact, artefact \Rightarrow creation \Rightarrow musical composition \Rightarrow **song***

Sense 2: *A distinctive or characteristic sound.*

Path: *event \Rightarrow happening, occurrence, natural event \Rightarrow sound \Rightarrow **song***

Sense 3: *The act of singing.*

Path: *act, human action, human activity \Rightarrow activity \Rightarrow diversion, recreation \Rightarrow music \Rightarrow vocal music \Rightarrow **song***

Sense 4: *Birdcall, call, birdsong, song.*

Path: *abstraction \Rightarrow relation \Rightarrow social relation \Rightarrow communication \Rightarrow signal, signaling, sign \Rightarrow animal communication \Rightarrow **song***

Sense 5: *A very small sum.*

Path: *possession \Rightarrow transferred property, transferred possession \Rightarrow acquisition \Rightarrow purchase \Rightarrow nargain, buy, steal \Rightarrow **song***

If all the senses of a noun were considered, we would have allocated the same cluster into multiple places in the hierarchy. Such a scenario would be ideal if we could distinguish its senses among different Web pages. Since we did not have a mechanism to identify the sense of the noun in a particular Web page, we chose to

map a particular cluster topic into the hierarchy only once using its most widely used sense from WordNet. In the above example, path: *entity, something* \Rightarrow *object, physical object* \Rightarrow *artifact, artefact* \Rightarrow *creation* \Rightarrow *musical composition* \Rightarrow *song* would be used to map cluster topic “*song*” during the construction of the hierarchical organization of search results. Such an approach is a trade-off between processing simplification and path accuracy. Since the path used for this purpose was always the one with the most popular sense of the noun, the path accuracy was satisfactory.

Require: The inverted file F generated from Pre-processing Module; threshold; WordNet
Ensure: A hierarchical tree of clusters containing Web search results

```

for all record  $r$ ,  $r \in F$  do
   $kwd \leftarrow r.keyword$ ;
   $freq \leftarrow r.frequency$ ;
   $urls \leftarrow r.URLs$ ;
  if  $kwd \in noun$  and  $freq \geq threshold$  then
     $path \leftarrow \emptyset$ ;
    repeat
      add_to_path( $kwd$ ); add  $kwd$  to a linked list called path
       $kwd \leftarrow get\_hypernym(kwd, WordNet)$ ; get the direct parent of  $kwd$  from Word-
      Net
    until  $get\_hypernym(kwd, WordNet) == NULL$ 
    for all node in path do
      if ! in_Tree( $node, tree$ ); if node is not built in tree then
        add_in_tree( $node, tree$ ); add the node into the tree structure
      end if
      if node ==  $kwd$  then
        attach_urls_to_treeNode( $urls, node, tree$ ); attach the cluster of Web documents
        to the tree node
      end if
    end for
  end if
end for

```

Table 5.1: Algorithm: Web Document Cluster Mapping

Path Optimization: Getting the hierarchical tree of search result clusters constructed is not enough. The resulting tree tends to be loose, and some of the tree paths can be as deep as a dozen levels. In order to reduce the user’s effort for navigation, the hierarchical organization of concepts and clusters should be optimized towards a smaller-sized tree. The pruning procedure we used for tree

path optimization is the following: for each of the internal tree node, if it has only one direct child node, this internal node should be pruned, and its leftmost child node would be used to replace its position. This pruning process is conducted recursively from the tree root, and ended at the leaf node of each path. The following example illustrates the pruning process. Since internal tree node B1 only has one direct child C1 along the tree path $A \Rightarrow B1 \Rightarrow C1 \Rightarrow D11$, node B1 is pruned, and replaced by node C1. By the same token, node C2 is replaced by D2.

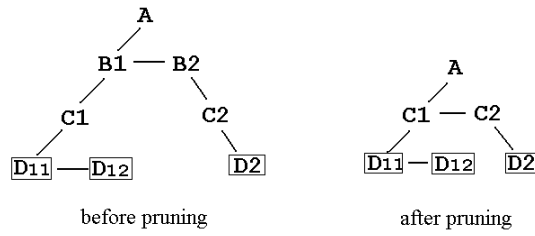


Figure 5.5: An Example of Path Optimization

The result of the path optimization is a denser tree with shorter path length, and thus leads to less user's effort for navigation of the hierarchical tree.

Require: tree root

Ensure: A optimized hierarchical tree of clusters containing Web search results

```

for all child node cn of root do
  if cn != leaf node then
    if ccn is the only child node of cn then
      remove(cn, root); remove the tree node cn from the tree
      ccn  $\Leftarrow$  cn;
    end if
    root  $\Leftarrow$  cn;
    prune(root);
  else
    return;
  end if
end for

```

Table 5.2: Algorithm: prune(tree)

Ranking: Since a mainstream Web search engine returns all relevant Web

pages to a query in a form of flat list, and such a list of search results could contain more than a few thousand matching pages, ranking becomes crucial to the success of a Web search engine. It has to have the ability to bring the most relevant and popular Web pages to the top of the list to draw user's attention. Otherwise, they will be buried in a large pile of matching pages, and probably never be reached by the user. However, ranking retrieved Web pages according to their relevance to the user's query is a very difficult, and subjective matter. The ontological approach of search result organization in our system downplays the importance of ranking as it is for a mainstream Web search engine. Instead, it categorizes the large set of returned Web pages into many smaller clusters, and provides a semantic hierarchical framework and an interactive navigation scheme to find target cluster(s) of Web pages in a progressive manner. Since the size of each cluster is much smaller (the actual size of a cluster depends on the settings of searching parameters), we assume that if the user reaches a particular cluster through the screening of navigation, he/she should at least sift through all the pages in the cluster for the best match(es). Thus, we did not rank Web pages containing in each cluster other than presenting them in the relative natural order from the search engines used for metasearch. However, for all the clusters categorized under a specific top level concept in the hierarchical tree, we did rank these clusters according to the descending order of their size. The logic behind it is quite simple. If a theme represented by the topic of a cluster is shared by more Web pages than another theme, it is considered as a more popular theme, and thus the cluster of this theme should be ranked higher than the other one. We believe that such an approach may help user's navigation in some degree especially when the user does not have a strong sense of target. However, a higher ranked cluster in our system does not indicate that it is more relevant to the user's query. We believe that such a relevance is best determined by the user during the interactive navigation process.

Chapter 6

Case Study and System Evaluation

6.1 Case Study

The system prototype is fully implemented, and is named *J-Walker*. The name is somewhat a reflection of its design concept and navigation scheme, which is ultimately aimed at giving the user the convenience and flexibility to find target information in a highly structured organization.

From the user's point of view, the system behaves similarly to a mainstream Web search engine. We believe that we can give users the much needed convenience for on-line searching basing on their previous experience and habits of Web search engine usage in order to promote the popularity of the system. Under such an assumption, we hide all the technical maneuvers behind the scene, and came up with an user interface that has the same format as other Web search engines for taking a user's query, and displaying the results. The major difference is the addition of a hierarchical folder tree, which is essential to allow the user's interaction with the system in a progressive manner. Even the hierarchical folder tree, which reminds users the Windows File Manager in both appearance and document arrangement concept, should be very familiar to the mainstream users.

Figure 6.1 is the query page to take a user's query term for Web searching. The query term can be a single English word or a multiple word phrase. Boolean operators are also supported for query generation. Before conducting a search, the user has the flexibility to define the following parameters for his/her Web search.

1. Select which Web search engine(s) to be included in the metasearch. Currently, AltaVista, Google, Lycos, NorthernLight, and Yahoo are integrated into the system for selection.
2. Specify how many matches returned by each Web search engine the user wants to take for consideration. The specified number of matches will be retrieved from top of the ranked list of Web search engine results. The range of this selection is from 50 to 200 with a step of fifty.
3. Define minimum cluster size. The higher the threshold, the higher percentage of cluster topic relevance will be against the query term, however, at the cost of a narrower range of cluster topics.

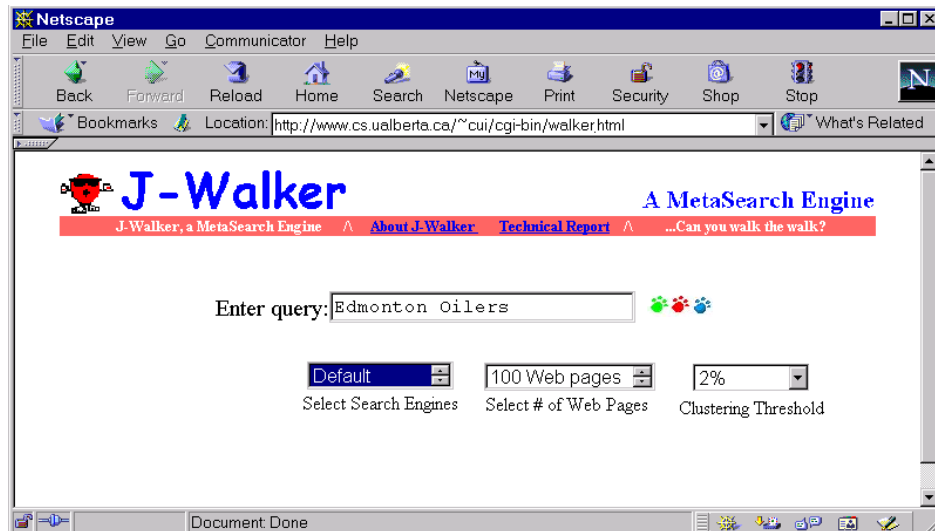





Figure 6.1: User Interface Page for Querying the System

The result page returned by the system in response to the user's query is illustrated in Figure 6.2. In the top frame of the browser is the query form that allows users to conduct another Web search. The bottom-left frame is a navigable hierarchical folder tree with a root node containing user's query term. The initial result page only displays the top level categories(folders) containing the most generalized concepts in the hierarchical folder tree. Each folder itself is a sub folder-tree. Given a concept in the hierarchical folder tree, the bottom-right frame displays

the number of matches found, as well as a list of matching Web pages categorized under the current folder(concept). For each matching Web page, its Title, Snippet and URL are listed.

For example, if you would like to browse all the retrieved Web pages in response to a query, you can click on the  icon of the root folder in the bottom-left frame to open the folder (an opening folder icon  will replace the closed folder icon ), and correspondingly, a full list of the retrieved Web pages will be displayed in the bottom-right frame.

Here, a querying example is given to illustrate the system behavior and some characters of our system design concept.

Since NHL Playoff games are currently under way, and Edmonton Oilers is among one of the playoff contenders, we choose “Edmonton Oilers” as the query term for Web search. The searching parameters are set as the following:

Web search engines specified for metasearch: AltaVista and Google. Matching results taken from each search engine: 100. Minimum cluster size: 2.

The returning result for the query is shown in Figure 6.2. The bottom-right frame gives a full list of retrieved Web pages. The bottom-left frame shows a hierarchical folder tree that displays the top level categories directly under the root - query term: “Edmonton Oilers”. The hierarchical folder tree sorts all the retrieved Web pages into 41 different topics. Each topic is a cluster which is located at the bottom of the folder tree, and has a unique path in the folder tree. Table 6.1 lists the topics and sizes of the 41 clusters. Each internal node of the folder tree is a semantic concept that includes all the topics of clusters under it according to the semantic relationship from WordNet.

Based on our design concept, the user will not see the 41 clusters at first. Instead, the adopted ontological approach creates a hierarchically structured organization of search results, and only displays the top level categories to which the topics of the 41 clusters belongs. Table 6.1 shows that the 41 clusters are included into 13 top level categories.

The navigation of the folder tree requires the user’s interaction. Depending

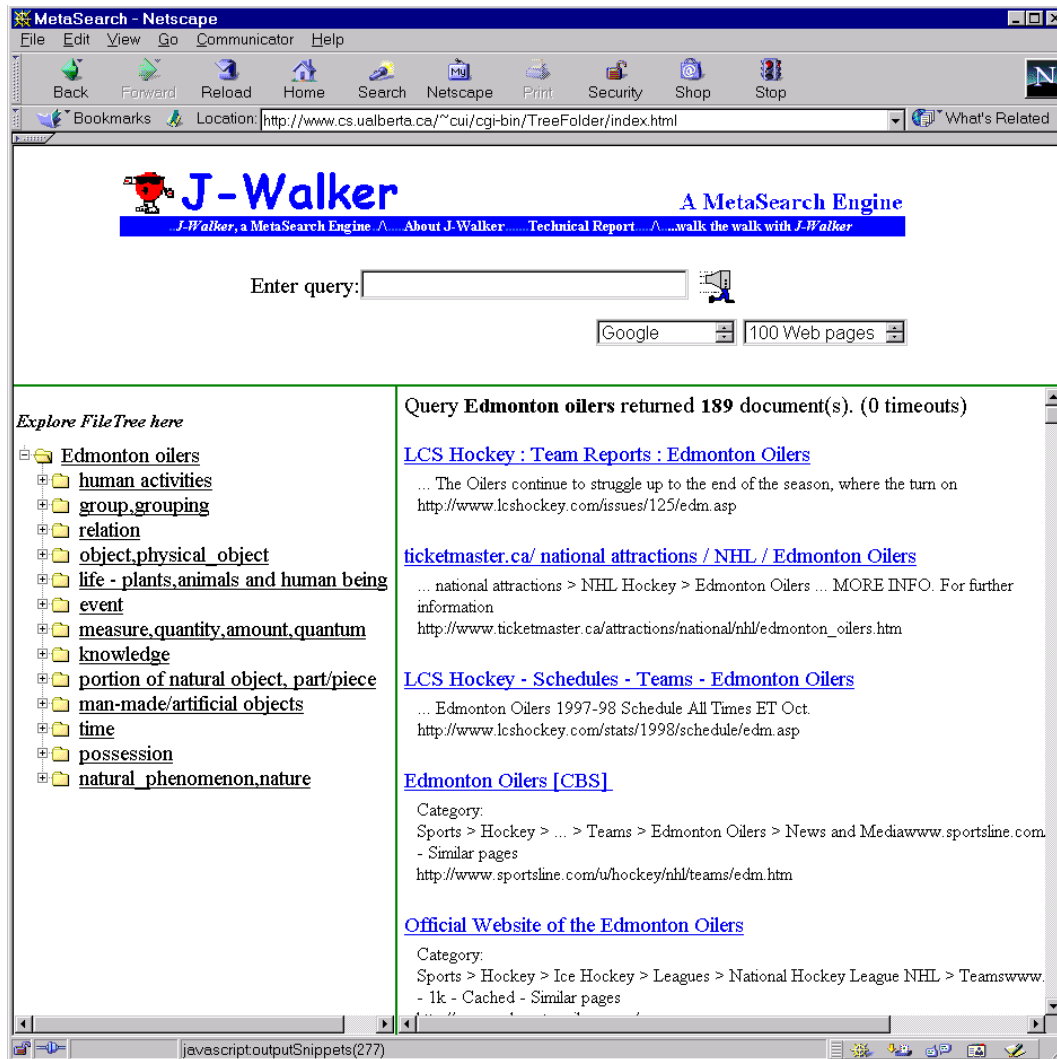




Figure 6.2: The Result Page in Response to Query

Query: Edmonton Oilers					
Cluster Topic	Size	Cluster Topic	Size	Cluster Topic	Size
hockey	36	Minnesota	3	February	2
sport	19	play	3	goaltender	2
team	18	report	3	January	2
news	10	season	3	lineup	2
game	9	statistics	3	Monday	2
slam	8	tattoo	3	northwest	2
player	6	calendar	2	outdoor	2
headline	6	chat	2	product	2
Wednesday	6	coliseum	2	return	2
league	5	Dallas	2	roster	2
recap	4	fan	2	ticket	2
schedule	4	fantasy	2	wild	2
sun	3	Friday	2	vote	2
March	3	former	2		

Table 6.1: Cluster Topics and Sizes of Search Result for “Edmonton Oilers”

upon the user’s interest, a particular top level category can be selected for navigation. By clicking on the  icon in front of the folder icon, the user performs Drill Down operation which further displays more specialized concepts under the selected category. If the user is interested in hockey activities related to Oilers for example, Drill Down should be performed along the path: Edmonton Oilers \Rightarrow human activities \Rightarrow activity \Rightarrow diversion, recreation \Rightarrow sport, athletics \Rightarrow hockey as shown in Figure 6.3 and Figure 6.4. Each concept along the drilling down path is a more specialized semantic concept of its precedent, but a more generalized one of its succedent. This type of semantic relationship provides a logic thread to help the user locate target information in a interactive and progressive manner.

Once drilling down reaches the cluster at the bottom of the tree path, all the titles of hyper-linked Web pages containing in the cluster are displayed for browsing as shown in Figure 6.5. In the example, the consecutive Drill Down operations brings the user to the cluster with topic “hockey”. If, for example, the user has a particular interest in the Web page with a title “Official Website of Edmonton Oilers” as listed in the fifth Web page in the cluster, he/she can click on the  icon to perform Drill Through operation to access the Web page. The Web page will be displayed in the bottom-right frame of the Web browser as shown in Figure 6.6. The Drill Through operation should not affect continuing navigation

- Edmonton oilers
 - human activities
 - group, grouping
 - relation
 - object, physical object
 - life - plants, animals and human being
 - event
 - measure, quantity, amount, quantum
 - knowledge
 - portion of natural object, part/piece
 - man-made/artificial objects
 - time
 - possession
 - natural phenomenon, nature

Figure 6.3: Initial Folder Tree Display with Top Categories

- Edmonton oilers
 - human activities
 - activity
 - diversion,recreation
 - sport,athletics
 - hockey
 - sport
 - attempt,effort,endeavor,endeavour,try
 - group_action
 - group_grouping
 - relation
 - object,physical_object
 - life - plants,animals and human being
 - event
 - measure,quantity,amount,quantum
 - knowledge
 - portion of natural object, part/piece
 - man-made/artificial objects
 - time
 - possession
 - natural_phenomenon,nature

Figure 6.4: Performing Drill Down on the Folder Tree

Explore FileTree here

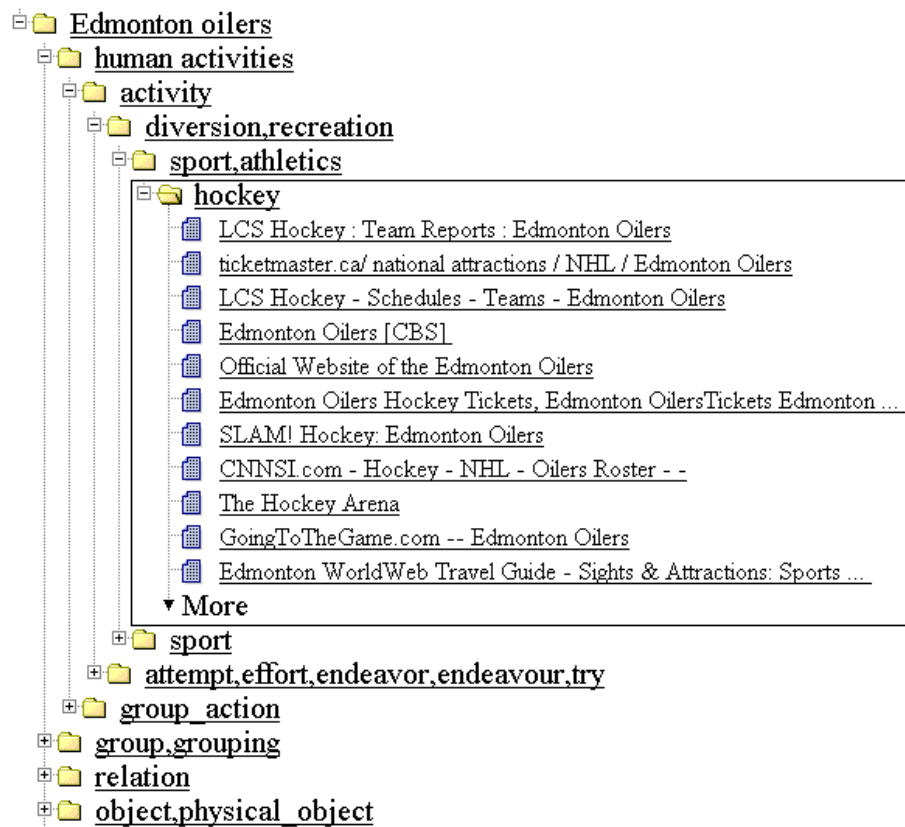


Figure 6.5: Cluster “hockey” at the Bottom of Navigating Path

of the hierarchical folder tree, and the user can explore other paths for relevant information under different topics.

Similarly, the user may find Oilers' game information by following the path: *Edmonton oilers* \Rightarrow *event* \Rightarrow *social event* \Rightarrow **game**; or Oilers related hockey merchandises through path: *Edmonton Oilers* \Rightarrow *object, physical object* \Rightarrow *commodity, goods* \Rightarrow **product**.

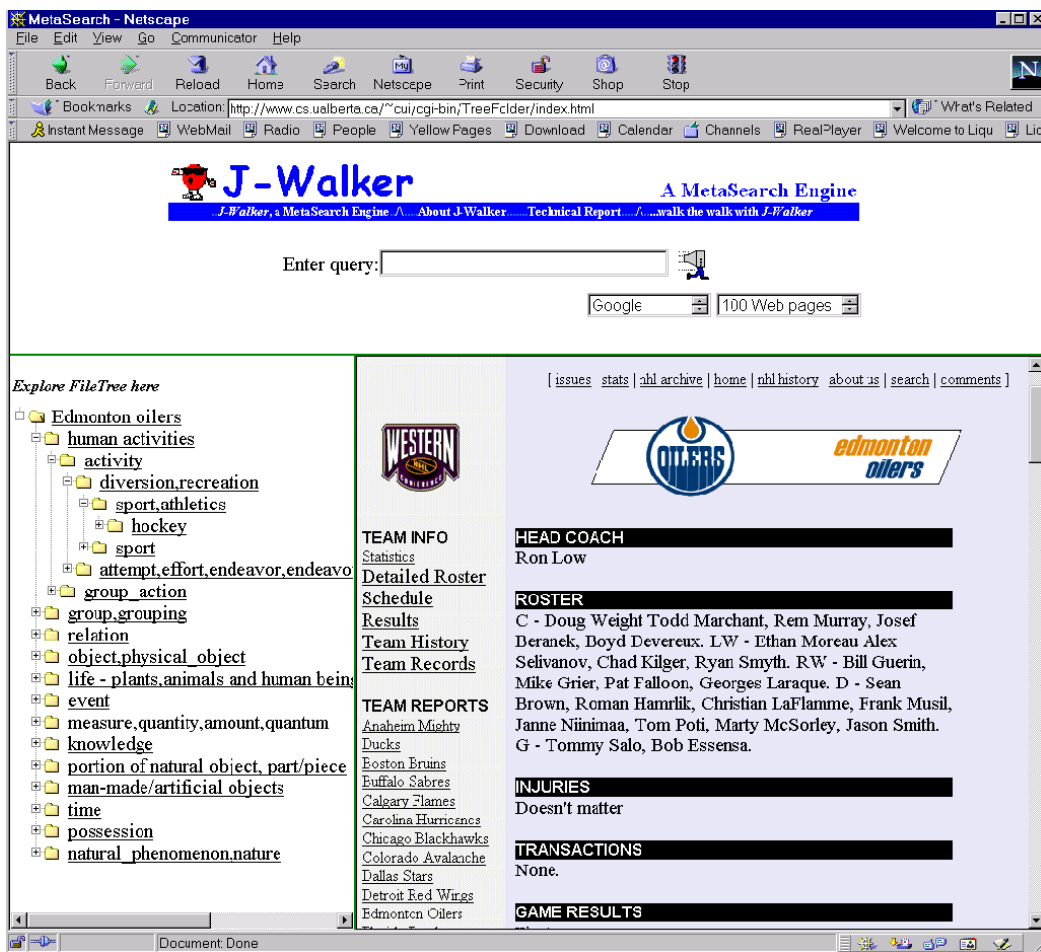


Figure 6.6: Drilling Through Operation for Web Site Display

While the hierarchical organization of search results provides a genuine solution

to manage a large number of Web documents for convenient and flexible user navigation, it also possesses some nice features that are far beyond the reach of long ranked list presentation of search results.

One advantage of such a hierarchical organization is that it is much simpler for users to find all retrieved Web pages sharing the same topic. However, this is not the case with the long ranked list presentation of search results. This point can be well illustrated in the “Oilers” example. By following the path: *Edmonton Oilers* \Rightarrow *life - plants, animals and human being* \Rightarrow *player* \Rightarrow *athlete,jock* \Rightarrow **goaltender**, the user can find the cluster containing two Web pages about Oilers’ goalie Tommy Salo. However, these two pages are listed separately at No.65 and No.81 in the ranked list of search results from Google. Most users would miss both of the pages from Google’s return if only top 30 matches are browsed.

Another advantage of the hierarchical organization of search results is that it is very helpful to users to discover the so-called “know-it-when-I-see-it” type of relevant resources. In many cases, users may not specify his/her interests very well in the query term. By navigating the hierarchical folder tree, the user may capture some relevant information while navigating for his/her target. This is also illustrated in the example. While the user follows the path: *Edmonton Oilers* \Rightarrow *knowledge* \Rightarrow *content* \Rightarrow **schedule** to find Oiler’s game schedule, he may also captures Oilers’ historical game statistics along the way under another sub-path of knowledge (*knowledge* \Rightarrow *information* \Rightarrow **statistics**). The numerous connections within the hierarchical organization of search results make the discovery of closely related information much easier.

Also, such an ontological approach for search result organization is very good at grouping result pages into different clusters according to the content of Web pages. This is especially elegant when the query term has double meanings, and users might have different search intentions given the same query term. This edge can be reflected in the following example.

For the query term “amazon”, it could mean two different things. One user might refer it to amazon.com, the well known on-line book seller; and for another user, it might indicate the world famous Amazon River. Therefore, the query

“amazon” would create an ambiguity for a mainstream Web search engine, and the consequence is a ranked list of search results with pages related to amazon.com and pages about Amazon River mingled together. Users who are interested in the Amazon River would have to sift through the ranked list to hand pick pages about Amazon River one by one, which is very time consuming. On the contrary, using *J-Walker* to search for query “amazon”, users would experience a totally different browsing practice. Pages that are related to amazon.com would be grouped into a cluster identified as “book”; and pages that are associated to Amazon River would be categorized into a cluster with topic “river”. These two clusters can be reached by tracing through the following two paths respectively as shown in Figure 6.7. Such an approach we believe would be much elegant and efficient from the user’s point of view.

amazon \Rightarrow *man-made/artificial object* \Rightarrow *creation* \Rightarrow *product, production* \Rightarrow ***book***

amazon \rightarrow *object, physical object* \rightarrow *body of water, water* \rightarrow ***river***

However, the downside of the system is that it is sometimes tricky to find the correct navigating path for target information. Some of the top level concepts in the hierarchical folder tree are not intuitive enough for users to figure out what are included, and thus might take a few path navigation to adjust the right direction. This is especially true for first time users who are not quite familiar with the structural connections of nouns in WordNet. In the “Oilers” example, users are required to follow the path: *Edmonton Oilers* \Rightarrow *relation* \Rightarrow *social relation* \Rightarrow *written communication* \Rightarrow ***ticket*** to find Web pages about Oilers’ game ticket information. While it might make strict linguistic sense to establish the “is a” inclusive relationship between the concepts in two adjacent hierarchical levels along this navigating path, it is obviously ambiguous and somewhat incomprehensible for mainstream users.

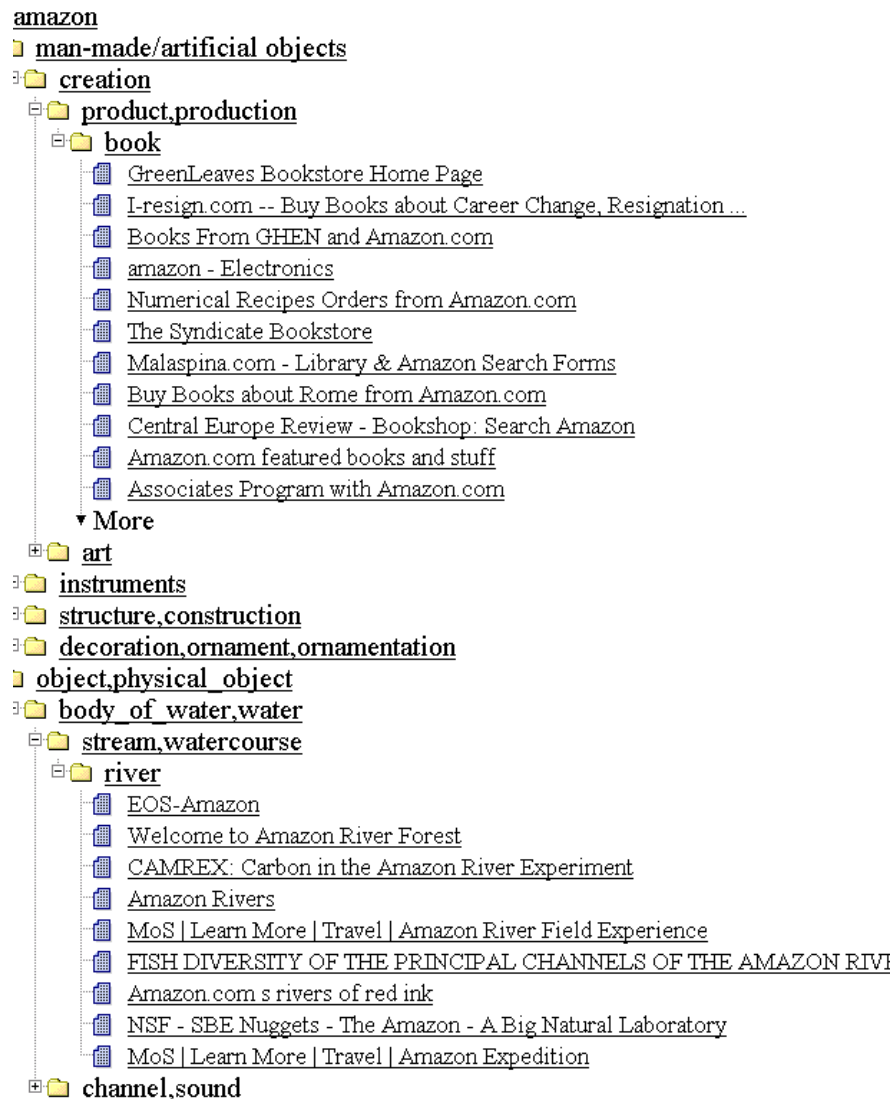


Figure 6.7: Hierarchical Folder Tree for Query “amazon”

6.2 System Evaluation

6.2.1 Evaluation of Information Retrieval System

Information retrieval is a loosely defined term, and may lead to misinterpretation in some circumstances. However, one should distinguish the differences between data retrieval and information retrieval in terms of their tasks and functionalities. Rijsbergen summarized the differences in the following table [RIJ99].

	Data Retrieval	Information Retrieval
Matching	exact match	partial match, best match
Inference	deduction	induction
Model	deterministic	probabilistic
Classification	monothetic	polythetic
Query language	artificial	natural
Query specification	complete	incomplete
Items wanted	matching	relevant
Error response	sensitive	insensitive

Table 6.2: Differences Between Data Retrieval and Information Retrieval

The fundamental difference between data and information retrieval is that data retrieval looks for exact match(es) to the query, while information retrieval in general tries to find the most relevant match(es) against the query. Thus, data retrieval demands a complete and error-free result set, and even a small error in the retrieved data set would mean a failure of the system. As for information retrieval, one can only request an effective and accurate output with a certain degree of tolerance for completion, relevance, and error.

The purpose of information retrieval is to reach the maximum effectiveness in a sense that such a system is designed to retrieve all relevant documents at the same time retrieving as few of the non-relevant documents as possible. Compared with data retrieval system, the nature of information retrieval system determines that the evaluation of such a system would be extremely difficult to conduct, and therefore the evaluation scheme for such a task would be subjective. Despite a volume of research work on system evaluation of information retrieval, a general theory of evaluation has not emerged yet [RIJ99].

Probably the best way to evaluate an information retrieval system is a well-defined, and complete survey of user's satisfaction of the system. However, most

of the systems in the experimental stage have used some sort of measurements for preliminary system evaluation. Cleverdon listed six measurable quantities for information retrieval system evaluation. The six quantities are: 1. recall of the system; 2. precision of the system; 3. the coverage of the collection; 4. the form of presentation of the output; 5. the user's effort for a search; and 6. the time log of a search [CLE66].

Among the six measurements, two criteria - Recall and Precision have been widely used for information retrieval system evaluation traditionally. Precision measures the proportion of retrieved documents that are relevant. Recall measures the proportion of relevant documents that are retrieved [CLE72]. The combination of precision and recall reflects the effectiveness, which is purely a measure of the ability of the system to satisfy the user in terms of the relevance of documents retrieved. However, as the World Wide Web is becoming a major alternative of traditional information repository, the traditional evaluation scheme shows its limit, and does not suit the task well anymore. The reason behind such a challenge lies in the conceptual differences between the Web and a traditional information repository. Compared with a traditional information repository, the Web has much larger scale and more diversified content and information sources. Also, unlike the relatively static, and well-managed traditional information repository, the Web is extremely dynamic, and not very well structured as well. Anybody can publish just about anything on the Web and can withdraw it from the Web anytime. Furthermore, because of such a dynamic nature of the Web, querying scheme for information retrieval from the Web is loose, which has been mainly relying on keyword(s) queries. Consequently, the volume of retrieved information itself is so huge that it is beyond the user's manageable scope. Web search engines have been heavily relying on ranking to alleviate this problem. Therefore, evaluation of such a system also goes beyond precision and recall. Under such an evaluation scheme, a high score does not necessarily give an accurate reflection of the quality of such a system if a relevant Web page does not rank high enough to draw the user's attention.

Unlike a mainstream Web search engine or a metasearch engine, which use a

ranked list to represent the output of retrieved document set, our system groups retrieved Web pages in a number of clusters, and maps the clusters into a hierarchical organization. The recall/precision scheme does not apply to the evaluation of a system with such an unique presentation and organization of retrieved results. Therefore, we come up with our own scheme for system evaluation.

6.2.2 System Evaluation Scheme

Here we briefly describe the mechanism and behavior of the *J-Walker* system before we get into the system evaluation scheme. It will help us comprehend the overall design of our evaluation scheme.

J-Walker is a metasearch engine, which takes the user's query, and sends it to multiple search engines for Web searching. The search result sets returned from Web search engines are merged, and duplicates are eliminated. The merged result set is then grouped into smaller clusters according to semantic information extracted from the results. Then, the clusters obtained are mapped into a hierarchical organization of lexical concepts with the most generalized concepts at the top level of the hierarchical organization, and each cluster residing at the bottom of each path.

Unlike a mainstream Web search engine that returns a flat ranked list of result in response to the user's query, *J-Walker* responds to the user's query by returning a hierarchical organization of concepts with clusters of search results located at the leaf nodes of the structure. Only the most generalized concepts at the top level of the hierarchical organization are displayed to the user. The user may start the navigation for target information by drilling down from a top level concept. Concepts at different levels provide supplemental hints for further navigation down the path until a particular cluster is reached. Since our research focus is the re-organization of search engine results, and the nature of the presentation and organization of J-Walker's search result is totally different from the widely adopted ranked-list presentation, the traditional evaluation scheme does not serve it well under such a circumstance.

Instead of using precision and recall for effectiveness evaluation, we intend to

evaluate two aspects of the system: the retrieval effectiveness of the system and the user’s effort to find target information. We believe that the combination of these two approaches can give us a better overall evaluation of the system.

In the hierarchical organization of search results in the *J-Walker* system, clusters and paths are two components that are the fundamentals to achieve the user’s progressive and interactive navigation for target information.

Cluster is the basic unit of retrieved document categorization. It functions as a container that holds a set of retrieved documents which possesses an unique and commonly-shared theme. The shared theme which is a phrase or a word is used as the topic to represent and identify the content of the cluster. Therefore, the quality of the cluster in terms of its content and identity is very important, and should be taken into account for system evaluation. Here, we propose a measurement *Cluster Precision* to evaluate the quality of the content in a cluster; and another measurement *Topic Relevance* to evaluate the quality of the cluster identification.

In the hierarchical presentation of search results, paths are the threads to connect the clusters together. A path consists of a sequence of concepts with a “IS A” relationship between two adjacent concepts along the path. The most generalized concept resides at the top, and a cluster at the bottom of a path. Because of the tree-like structure of paths in the hierarchical organization of search result, and the semantic relationships among concepts along a particular path, paths establish the essential logic threads of the system to help user’s navigation. Concepts at a particular level of the hierarchy offer the user semantic clues to choose the correct next-level concept to reach the target cluster of information. Therefore, ideally, the path should be intuitive and short enough in order to help user’s navigation. In our evaluation scheme, we propose *Path Length* to measure the steps that the user has to go through to reach target information, and *Path Intuitiveness* to measure how easy a path can be followed by the user from a semantic point of view.

In sum, we propose a system evaluation scheme that evaluates both the effectiveness of the system and the hierarchical presentation style of search results. In our system, retrieved Web pages are grouped into clusters, and the clusters are

mapped at the bottom of the paths in the hierarchy. Therefore, we propose to use cluster precision and topic relevance to measure the effectiveness of the system, and use path length and path intuitiveness to measure how easy the hierarchical presentation may help user’s result navigation. Detailed definitions and descriptions of these evaluation criteria are described in the following section.

6.2.3 Evaluation Criteria

Since clusters and paths are integrated parts of the hierarchical organization of search result in *J-Walker* system, and play such crucial rules in user’s result navigation process, our system evaluation is designed to assess both cluster properties including cluster size, cluster precision and topic relevance, and path properties covering path length and path intuitiveness.

Cluster Properties

Cluster Size: Cluster size is defined as the number of Web pages contained in a cluster. In a way, the size of a cluster is a reflection of the quality of the cluster as we will see in the experiment section. While too big a cluster would reduce the browsability of the cluster, small clusters have the tendency to reduce the overall cluster precision and topic relevance of the search result given a query. Therefore, the minimum cluster size for cluster generation is used and adjusted in our experiment to study cluster and path properties.

Cluster Precision: Cluster Precision measures the percentage of the Web pages contained in a cluster which are considered as relevant to both the topic of the cluster and the original query term. Deciding whether a Web page is relevant to the topic of the cluster and the query term is rather subjective, and the content of the Web page should be evaluated for this purpose. If the overall content of the Web page is reasonably related or linked to the cluster topic and the query term, the Web page should be considered as relevant. By such a standard, the Web page would not be considered relevant if the word or phrase contained in the cluster topic or query term merely appears in the Web page, but the content itself has nothing do to with either of them.

$$Precision(\%) = (Nb. \text{ of relevant Web pages} / \text{total Nb. of Web pages in the}$$

$cluster) \times 100$

Topic Relevance: Unlike cluster precision that measures how its content is relevant to the query term and the cluster topic, topic relevance measures how its identification is relevant to the user's query term. The topic of a cluster is a theme that is used to label or represent the Web pages in the cluster. Therefore, ideally, all the clusters generated should be relevant to the query term from different angles or aspects in order to cope with the user's individual need. In reality, not all topics of clusters are relevant to the query term. Topic relevance is used to measure such a relationship.

For an individual cluster, its topic relevance is graded as *VR-very relevant*; *R-relevant*; or *NR-not relevant*. A cluster topic is considered as *very relevant* to the query term if the cluster topic has a very clear and straight forward connection to the query term. For example, the cluster topic is very relevant to the query term if it's the synonym of the query term, or closely related to its properties, specifications, or characters. A cluster is considered *relevant* if it is reasonably related to the query term one way or another logically, semantically or in some cases by common sense. A cluster topic is *not relevant* to the query term if it has no apparent and legitimate tie to the query term. The standard used for such a grading scheme is subjective, but the measurement itself is a good reflection of the quality of clusters. If handled carefully and consistently, the error can be reduced to minimum level.

Topic Relevance Grading:

VR - very relevant;

R - relevant;

NR - not relevant.

The overall topic relevance for the search result given a query term is calculated as the percentage of the number of relevant cluster topics over the total number of clusters generated.

$Topic\ Relevance(\%) = (Nb.\ of\ relevant\ cluster\ topics / total\ Nb.\ of\ clusters\ per\ query) \times 100$

Path Properties

While cluster properties measure the effectiveness of the information retrieval system, path properties are used to measure the search result presentation and the user's effort to find target information.

Path Length: Path length is defined as the number of conceptual levels from a top level concept to a cluster located at the bottom of a particular path. Path length can also be treated as number of mouse clicks the user needs to conduct in order to reach a particular cluster from a top level concept in the hierarchy assuming that the user follows a correct path during the navigation. Therefore, a short path length means less user's effort of navigation. More importantly, a short path length implies less decision making on the user's part along the path navigation, which reduces the chance to select a wrong path when looking for his/her target information. However, path length is an internal property of the hierarchical organization of search result, and can not be altered at will.

Path Intuitiveness: Like path length, path intuitiveness also reflects the amount of user's effort to find target information. In the hierarchical organization of search result, a path consists of a sequence of concepts at different levels of the hierarchy. Given a cluster topic, path intuitiveness measures the overall difficulty to locate the target cluster starting from the top level categories. Path intuitiveness grading largely relies on the perception of the evaluator. In order to quantify path intuitiveness and reduce human error in the process, the following grading scheme is established based on some preliminary experience and observations of the system.

Path Intuitiveness grading standard (0 - 10 scales):

Scale 0: The topic of the cluster is allocated in a semantically wrong path, and can not be located using the semantic information provided by the system.

Scale 1 - 3: The cluster is difficult to find using the semantic clues provided by the semantic hierarchical organization. But the semantic information provided by the system does refine user's navigation scope. However, finding the target cluster would take a few path navigations, and thus create confusion to the user.

Scale 4 - 5: The cluster is not easy to find using the semantic clues provided by the semantic hierarchical organization. In addition to the situation that is

mentioned in the description of Grade 6-7, the same situation is also encountered at least a couple of more times along the way of path navigation through roll-up/drill-down operations.

Scale 6 - 7: The cluster is somewhat easy to find using the semantic clues provided by the semantic hierarchical organization. Among the top level concepts, the user might find that the target cluster topic could be allocated under any path of two or three top level concepts, and need to guess the correct path (among the two or three candidates) at the top concept level. But, in worst case, this should not take the extra work of more than two or three drilling down operations to get back on track.

Scale 8 - 9: The cluster is relatively easy to find using the semantic clues provided by the semantic hierarchical organization. The top level concepts provide very clear semantic clues to predict under which path it belongs to, but the concepts at different levels along the path might create some slight confusion for the user when selecting concept(s) for drilling down operation along path navigation. But such a confusion should be minor, and usually takes one or two rolling ups for correction.

Scale 10: The cluster is very easy to find using the semantic clues provided by the semantic hierarchical organization. The top level concepts provide very clear semantic clue to predict under which path it belongs to, and the concepts at different levels along the path provide very clear and supplemental clues to make correct and quick decisions for drilling down operations to locate the target cluster.

6.2.4 Experiments and Result Analysis

We ran thirteen actual queries on *J-Walker* system, and data were collected from the search result of each query for system evaluation. The system parameters were set as the following: Web search engines covered: AltaVista and Google; number of search results extracted from each Web search engine: 100. The same parameter settings were used for search of all the eleven queries.

The thirteen queries were carefully chosen in an effort to cover a diverse and broad range of domains. The purpose of our wide coverage of query selections was

to minimize testing errors caused by skewed samples, and also as a way to show the robustness of the system functionality. The thirteen queries used for system evaluation are: Amino Acid, Adidas, Artificial Intelligence, Amazon, Cheddar Cheese, Edmonton Oilers, George Bush, Inline Skates, Mutual Fund, Napster, Sodium Benzoate, Stephen King, and University of Alberta. These thirteen queries can be concluded into five general domains as shown in Table 6.3.

Domain	Queries
Leisure	Napster(music related), Stephen King(literature related)
Substance	Amino Acid, Sodium Benzoate(chemicals), Cheddar Cheese, Adidas, Inline Skates (commodities)
Education	Artificial Intelligence(knowledge domain), University of Alberta(institution)
People	George Bush(politician), Stephen King(novelist)
Sports	Adidas(sports brand), Inline Skates(sports gear) Edmonton Oilers(sports team)
Investment	Mutual Fund (investing venue)
Misc.	amazon

Table 6.3: Queries and Their Domains

We ran each query through the *J-Walker* system, and data were taken from the returned search result. For each of the clusters in the search result, the topic of the cluster, cluster size and path length were recorded; then cluster precision, cluster topic relevance, and path intuitiveness were evaluated through navigation, and graded accordingly. The cluster size, path length, cluster precision, topic relevance and path intuitiveness from all clusters of the search results were averaged respectively, and the averaged measurements were used to indicate the overall quality of search result from a particular query. The detailed data collection for all thirteen queries was listed in the Appendix, and the summarized result analysis for these queries are shown in Table 6.5.

As mentioned in the previous section, the minimum cluster size was used as clustering threshold when constructing the hierarchical organization of search result. We also tested the correlation between clustering threshold and cluster properties. Table 6.4 presents the overall cluster properties calculated from the search results of the thirteen queries at different clustering threshold levels ranging from two percent to five percent.

The results indicate that the clustering threshold has little impact on the path properties: path length and path intuitiveness as indicated in Figure 6.8. This comes no surprise since path properties are largely determined by the ontology of nouns in WordNet. Therefore, the path intuitiveness which is a major indication of the amount of user's navigation effort for target information is directly related to the concept hierarchy adopted in the system, and can only be improved with the improvement or alternation of the concept hierarchy.

Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	3.6	3.7	76.0	80.2	6.0
3%	5.3	3.8	82.1	85.2	6.5
4%	7.4	3.8	88.9	94.2	6.8
5%	9.1	3.9	91.9	96.1	7.5

Data compiled from the search results of 13 queries as shown in Appendix

Table 6.4: The Relationship Between Cluster Properties and Clustering Threshold

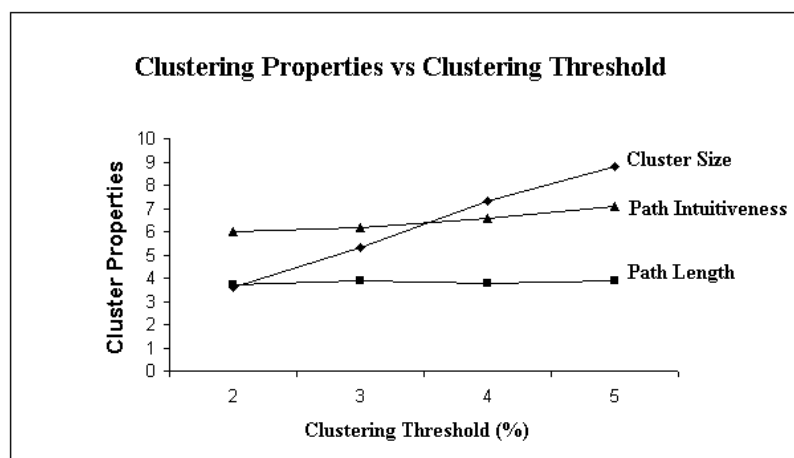


Figure 6.8: The Relationship Between Cluster Properties and Clustering Threshold

The averaged cluster size is raised from 3.6 to 9.1 as clustering threshold is increased from 2% to 5%. Ideally, the cluster size should not exceed 50% of the total retrieved Web pages since a very large cluster means low browsability. Also, an overly large cluster implies that the cluster topic is not effective enough to

categorize retrieved Web pages by theme, and thus should not be generated. As for the lower end of clustering threshold, it is a tradeoff situation when determining the minimum clustering threshold. The increase of clustering threshold would raise of the overall cluster size, but at the cost of a narrower topic coverage of clusters. However, this does not necessarily all bad. Based on our observation, non-relevant clusters (in terms of both lower cluster precision and topic relevance) are mainly contributed by those of smaller clusters. Eliminating these clusters with smaller size through clustering threshold setting would significantly improve the overall cluster precision and topic relevance as shown in Figure 6.8. So, clustering threshold selection is a balancing process between topic coverage and overall quality of clusters. Our testing data indicated (Figure 6.9) that there is a steep jump for both topic relevance and cluster precision when the clustering threshold is raised from 3% to 4%. Consequently, we selected four percent as the optimal clustering threshold.

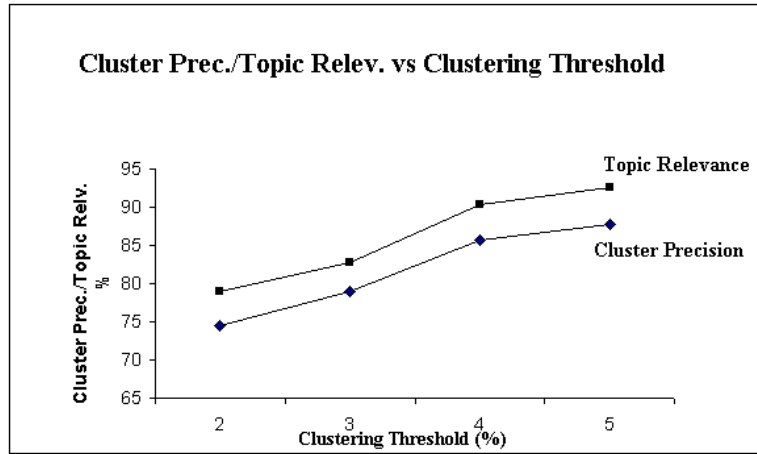


Figure 6.9: The Relationship Between Cluster Precision/Topic Relevance and Clustering Threshold

Table 6.5 gives the statistics of cluster properties for each of the search results from the thirteen queries, and the averaged overall evaluation of cluster properties with a clustering threshold of 4%.

The data indicate that on average, it takes the user about four clicks (four drill-down operations) along a particular path to reach a target cluster, which is rather efficient in term of user's navigating effort thanks to the optimization of the constructed hierarchy during the implementation of the System Engine Module. Since all the nodes along a path in the hierarchy, which only have one single direct succedent, were pruned during optimization, the result is an optimized hierarchy with a much denser overall structure.

The average cluster size is about 7 to 8. It is highly optimal since smaller group organization of documents is convenient for browsing, and this is especially true when a topic is also provided as the summary of all the documents within the cluster. Another advantage of the smaller cluster size is that we do not need to emphasize ranking of retrieved Web pages other than giving the approximate natural order from the Web search engines used. The assumption behind this is that all the Web pages within the cluster are supposed to be relevant to the query and the cluster topic, and the user should browse all the Web pages in the cluster if he/she gets to this point through navigation. The 88.9% cluster precision from our experiments shows a fairly strong support to the assumption.

Query Term(s)	Cluster Size	Path Length	Cluster Precision%	Topic Relevance%	Path Intuitiveness
Adidas	5.9	4.0	83.4	87.5	7.5
Amazon	8.6	3.4	95	100	7.8
Amino Acid	5.5	3.5	100	100	7.5
Artificial Intelligence	7.8	3.7	100	100	6.4
Cheddar cheese	7.1	4.8	66.0	91.0	6.2
Edmonton Oilers	9.6	3.3	100	100	8.0
George Bush	9.8	3.3	100	100	6.8
Inline Skates	7.6	3.3	80.8	88.2	6.1
Mutual Fund	7.6	3.9	80.4	100	6.3
Napster	7.9	4.3	85.7	85.7	7.7
Sodium Benzoate	5.5	3.9	100	92.3	7.0
Stephen King	6.6	3.4	80.0	80.0	4.8
University of Alberta	6.7	4.1	84.6	100	6.9
Average	7.4±1.21	3.8±0.46	88.9±13.16	94.2±13.07	6.8±0.78

clustering threshold: 4%

Table 6.5: Cluster Property Analysis of Search Results from 13 Queries

In addition to cluster precision, topic relevance also gives high marks on the

quality of clusters. The overall topic relevance of cluster from our experiments is over 94% with a range from the low of 80.0% to the high of 100% depending upon individual queries. One explanation of this is that the snippet quality of returned search results tends to fluctuate from query to query, thus affects the quality of cluster topics extracted from the snippets. The negative effect of snippet quality on topic relevance is not significant as shown in our experiments. Among the thirteen queries we chose for testing, all have a topic relevance above 80%, and more than half of them are over 90%.

The only measurement in our experiments that gave a disappointing result is Path Intuitiveness. The overall score of path intuitiveness from our experiments is 6.8 in a scale of 0 to 10. The interpretation of such a score level implies that the semantic clues provided by the paths of the hierarchical organization can definitely help users find target information among search results. But the logic connections and semantic relationships within the paths are not intuitive enough, and may cause confusions to users at certain points of their navigation process. Such a phenomenon has something to do with the philosophy behind the concept hierarchy construction. In our project, we adopted the concept hierarchy of nouns from WordNet, and it was built based on strict linguistic relationships among English nouns. While the semantic connections within it make perfect sense from a linguistic point of view, it lacks the intuitiveness among the senses of concepts that ordinary users face in everyday life. Nevertheless, this negative factor can be simply eliminated by replacing WordNet concept hierarchy with a more user-friendly one such as Yahoo's Web Directory, and such an enhancing process should be easy to conduct without involving any overhaul of the rest of the system.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this research work, we have designed and implemented a system that categorizes Web documents returned from Web search engines using an ontological approach. The result is a hierarchical organization of Web search results with a mechanism to allow users to navigate the hierarchy to find target information organized in clusters. Such an approach aims at improving the browsability of long ranked list presentation of search engine result, which has been widely using by the Web search engine industry. Metasearch concept is also used in the system to increase Web coverage. The system classifies Web documents by mapping them on an existing concept hierarchy, and the resulting tree-like structure is later optimized to keep the essential paths for user's navigation.

In such a framework, a cluster is a Web document container, and treated as an existing unit of a group of Web documents represented by a theme. The theme is generated from the commonly shared phrase or word of the Web documents. The themes representing clusters of Web documents are classified into the hierarchical organization according to the semantic relationships provided by WordNet concept hierarchy. The constructed hierarchical organization provides the fundamental for user's navigation scheme based on information abstraction philosophy. Therefore, user's Web search result browsing is a process of progressive and interactive navigation of concepts from the most generalized ones down to more and more specialized ones.

We have shown a proof of the concept to adopt the combination of hierarchical organization and progressive navigation scheme to improve the browsability of Web search results using WordNet as an initial semantic network. Our preliminary evaluation results strongly support the concept behind our designing philosophy. The experiment results from thirteen queries achieved very good cluster quality with both high cluster precision and topic relevance. The averaged path length of cluster in the hierarchical organization was also optimal from user's navigation point of view. However, we have not achieved high score in path intuitiveness, which was used to reflect the amount of information about the sub-concepts that a user can perceive from a specific concept along a particular path. This is due to the nature of WordNet, rich with linguistic terms. The resulting tree is often full with elaborate terminology at the different nodes. The system however, can be done with any given concept hierarchy.

7.2 Future Work

As the Web keeps growing, the volume of Web pages returned by a search engine in response to a query is becoming unmanageable to search engine users. The long ranked list presentation of search results can no longer meet user's satisfaction. NorthernLight has started to use a hierarchically organized Custom Search Folders [NOR01] for display of the search results along with the traditional ranked list presentation. While NorthernLight's Custom Search Folders has a lot of room to improve, there is no doubt that hierarchical clustering of Web documents is a very interesting and useful way to improve the browsability of Web search engine results. Based on this research work, we believe that breakthroughs in the following two research directions would be significantly meaningful.

One aspect of our system that needs to improve is path intuitiveness. Path intuitiveness is very tightly related to the concept hierarchy used for hierarchical clustering of Web documents in our approach. Thus, a proper replacement of WordNet seems to be a legitimate approach. One interesting candidate of such a concept hierarchy is the Web Directory that is provided by most of Web search engines like Yahoo. Such a Web directory was constructed manually according

to very pragmatic standard for document categorization following human's browsing behavior, thus is very intuitive and convenient for navigation. However, the real challenge is how to classify a given Web document into such a hierarchy automatically. While term matching approach might compromise the accuracy of classification, a training process involving in machine learning algorithm would be much complicated. More research need to be conducted on this area to quantify the accuracy and complexity of such a strategy.

In addition to using an existing concept hierarchy for Web document clustering, other algorithms should also be investigated for this purpose. Currently, we are exploring the use of association rule mining to discover frequent terms and phrases in large Web document collection for clustering, and use the learned knowledge to recursively construct a hierarchical organization of Web document clusters. Such an approach would not rely on any given concept hierarchy of thesaurus, and therefore should be intuitive for navigation, but at a potential cost of losing the logic thread along a vertical path. However, such an approach is very interesting, and would be a good alternative to the ontological approach for Web document categorization.

Bibliography

- [AND01] K. Andrews, C. Gutl, J. Moser, and V. Sabol, "Search Result Visualization with xFIND," in *Proceedings of 2nd International Workshop on User Interfaces to Data Intensive Systems*, pp. 50 - 58, Zurich, Switzerland, June, 2001.
- [BEC90] R. Beckwith, G. Miller and R. Tengi, "Implementing a lexical network," *International Journal of Lexicography*, 3(4), pp. 302 - 312, 1990.
- [BRI98] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine," in *Proceedings of the Seventh International World Wide Web Conference*, pp. 107 - 117, 1998.
- [CHA93] M. Chalmers, "Using a landscape metaphor to represent a corpus of documents," in *Spatial Information Theory, Proc. COSIT'93*, pp. 377 - 390, Boston, MA, USA.
- [CHA96] M. Chalmers, "Adding imageability features to information displays," in *Proc. UIST'96*, Seattle, WA, USA, November, 1996.
- [CHA99] S. Chakrabarti, M. Berg, and B. Dom, "Focused Crawling: a new approach to topic-specific Web resource discovery," *the 8th World Wide Web Conference*, Toronto, Canada, May 1999.
- [CHO98] J. Cho, H. Garcia-Molina, and L Page, "Efficient crawling through URL ordering," *the 7th International WWW Conference*, Brisbane, Australia, April 14-18, 1998.
- [CLE66] C.W. Cleverdon, J. Mills, and M. Keen, "Factors determining the performance of indexing systems," *ASLIB Cranfield Project*, Cranfield, 1966.
- [CLE72] C.W. Cleverdon, "On the inverse relationship of recall and precision," *journal of Documentation*, 28, pp. 195 - 201, 1972.
- [CUI01] H. Cui and O. R. Zaiane, "Hierarchically structural approach to improving the browsability of Web search engine results," Accepted by *The First International Workshop on Digital Libraries in conjunction with the 12th International Conference on Database and Expert Systems Applications*, Munich, Germany, Sept. 3-7, 2001.
- [CPK00] Claus-Peter Klas, Norbert Fuhr, "A new effective approach for categorizing Web documents," *22nd Annual Colloquium on Information Retrieval Research*, Cambridge, England, April 2000.

- [CUT92] D. Cutting, D.R. Karger, J. Redersen and J. Turey, "Scatter/Gather: A cluster based approach to browsing large document collections," in *Proceedings of the 15th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 318-329, 1992.
- [CUT93] D. Cutting, D.R. Karger, J. Redersen and J. Tukey, "Constant interaction time Scatter/Gather browsing of large document collections," in *Proceedings of the 16th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 126-135, 1993.
- [DIR01] <http://www.directhit.com>.
- [DUM98] S. Dumais, J. Platt, D. Heckerman et al., "Inductive learning algorithms and representations for text categorization," in *Proceedings of the 1998 ACM 7th International Conference on Information and Knowledge Management*, pp. 148 - 155, 1998.
- [DUN98] Dunja Mladenic, "Turning Yahoo into an automatic Web-page classifier," in *Proceedings of the 13th European Conference on Artificial Intelligence ECAI'98*, pp. 473-474, 1998.
- [EAG96] A. Eagan, and L. Bender, "Spiders and Worms and Crawlers, oh my: searching on the World Wide Web," <http://www.library.ucsb.edu/untangle/eagan.html>, 1996.
- [ETZ96] O. Etzioni, "Moving up the information food chain: deploying softbots on the World Wide Web," In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996.
- [FEL93A] C. Fellbaum, D. Gross, and K. Miller, "Modifiers in WordNet," in *An Electronic Lexical Database, ISBN 0-262-06197-X*, The MIT Press, Cambridge, MA, USA, 1998.
- [FEL93B] C. Fellbaum, D. Gross, and K. Miller, "English verbs as a semantic net," in *An Electronic Lexical Database, ISBN 0-262-06197-X*, The MIT Press, Cambridge, MA, USA, 1998.
- [FIS87] D. H. Fisher, "Knowledge acquisition via incremental conceptual clustering," *Machine Learning*, 2: pp. 139 - 172.
- [FRI96] N. Friedman and M. Goldszmidt, "Building classifiers using bayesian networks", in *Proceedings of AAAI 96*, pp. 1277 - 1284.
- [GON98] J. Gonzalo, F. Verdejo, I. Chugur, and J. Cigarran, "Indexing with WordNet synsets can improve text retrieval," *cmp-lq/9808002*, August 5, 1998.
- [GOO01] "Google", <http://www.google.com>, March 2001.
- [HAN92] J. Han, Y. Cai and N. Cercone, "Knowledge Discovery in Databases: An Attribute-oriented Approach," in *Proceedings of the 18th VLDB Conference*, Vancouver, BC, Canada, 1992.

- [HAN94] J. Han, O. Zaiane and Y. Fu, "Resource and knowledge discovery in global information systems: a scalable multiple layered database approach," *Technical Report*, <ftp://ftp.fas.sfu.ca/pub/cs/techreports/1994/CMPT94-10.ps.Z>, 1994.
- [HEN00] M. Henzinger, "Link analysis in Web information retrieval," *IEEE Data Engineering Bulletin*, November 2000.
- [JOA97] T. Joachims, "A probabilistic analysis of the Rocchio algorithm with TFIDF for text classification," in *Proceedings of the 14th International Conference on Machine Learning ICML97*, pp. 170-178, 1997.
- [KIN00] D. King, "Specialized search engines: alternatives to the big guys," *Online*, May 2000.
- [KOL96] D. Koller and Y. Shoham, "Information agents: a new challenge for AI," *IEEE Expert*, pp. 8 - 10, June 1996.
- [KOL97] D. Koller and M. Sahami, "Hierarchically classifying documents using very few words," in *Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 170 - 178, Nashville, TN, July 1997.
- [LAW98] S. Lawrence and C. Giles, "Searching the World Wide Web," *Science*, Volume 280, No.5360, pp. 98-100, 1998.
- [LAW99] S. Lawrence and C. Giles, "Search the Web: General and Scientific Information Access," *IEEE Communications*, 37(1), pp. 116-122, 1999.
- [LES69] M.E. Lesk and G. Salton, "Relevance assessments and retrieval system evaluation," *Information Storage and Retrieval*, 4, pp. 343 - 359, 1969.
- [MAC98] S.A. Macskassy, A. Banerjee, B. Davison, and H. Hirsh, "Human performance on clustering Web pages: a preliminary study," in *Proceedings of the Fourth International Conference on knowledge discovery and data mining*, New York, USA, 1998.
- [MEE00] R. Meersman, "Can Ontology learn from database semantics," <http://www.starlab/vub.ac.be/staff/robert>.
- [MIL90] G. Miller, "Nouns in WordNet: a lexical inheritance system," *International Journal of Lexicography*, 3(4), pp. 245 - 264, 1990.
- [MIL95] G. Miller, "WordNet: a lexical database for English," *Communications of the ACM*, 38 (11), pp. 39 - 41, November, 1995.
- [MIL98] R. Miller and K. Bharat, "SPHIN: a framework for creating personal, site-specific Web crawlers," <http://www.cs.cmu.edu/afs/cs/user/rcm/WWW/papers/www7/www7.html>.
- [MOD00] D. Modha and W. Spangler, "Clustering hypertext with applications to web searching," in *Proceedings of ACM Hypertext Conference*, San Antonio, Texas, May 30 - June 3, 2000.

- [MUR99] T. Murrata, "Machine discovery based on the co-occurrence of references in a search engine," in *Proceedings of the Second International Conference on Discovery Science (DS'99), Lecture Notes in Artificial Intelligence 1721*, pp. 220 - 229, 1999.
- [NMC01] <http://www.netmechanic.com/powerpack/tracker.html>, May, 2001.
- [NOR01] <http://www.northernlight.com>.
- [PIR96] P. Pirolli, P. Schank, M. Hearst, and C. Diehl, "Scatter/Gather Browsing Communicates the Topic Structure of a Very Large Text Collection," in *Proceedings of CHI'96*, pp. 213 -220, Vancouver BC, Canada, April 1996.
- [RIJ99] C.J. Rijsbergen, "Information Retrieval, Second Edition," <http://www.dcs.gla.ac.uk/Keith/Preface.html>.
- [SBW98] W. Sander-Beuermann and M. Schomburg, "Internet information retrieval - The further development of Meta-searchengine technology," in *Proceedings of the Internet Summit*, July 22-24, 1998.
- [SEL95] E. Selberg and O. Etzioni, "Multi-service search and comparison using the MetaCrawler," in *Proceedings of the 1995 World Wide Web Conference*, Darmstadt, Germany, April 10-13, 1995.
- [SEL96] E. Selberg and O. Etzioni, "The MetaCrawler architecture for resource aggregation on the Web," *IEEE Expert*, 12(1), pp. 8-14, 1997.
- [SHE00] C. Sherman, "The future revisited: what's new with Web search," *Online*, May 2000.
- [SUL98] D. Sullivan, "The Search Engine Update," <http://www.searchenginewatch.com>, August 4, 1998.
- [SUL01] <http://www.searchenginewatch.com>.
- [TAP00] "Study says Web is 500 times larger then major search engines now show'," <http://www.cnn.com>, July 27, 2000.
- [THE99] <http://www.infoday.com/it/jan99/news9.htm>, June, 2001.
- [UKK95] E. Ukkonen, "On-line construction of suffix trees," *Algorithmica*, 14: pp. 249-260, 1995.
- [UMP01] <http://www.umap.com/highuk/index.htm>, May, 2001.
- [WEB01] <http://www.webmasterGate.com>., April. 2001.
- [WEI96] R. Weiss, B. Velez, and M. Sheldon, "HyPursuit: A hierarchical network search engine that exploits content-link hypertext clustering," in *Proceedings of the Seventh ACM Conference on Hypertext*, Washington, DC, March 1996.
- [WOR01] <http://www.cogsci.princeton.edu/wn/> .
- [YAT99] R. Baeza-Yates and B. Ribeiro-Neto, "Chapter 3: Retrieval Evaluation; Chapter 13: Searching the Web," *Modern Information Retrieval*, ACM Press, New York, USA, 1999.

- [ZAM97] O. Zamir, O. Etzioni, O. Madani, and R. Karp, "Fast and intuitive clustering of Web documents," in *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, pp. 287-290, 1997.
- [ZAM99] O. Zamir and O. Etzioni, "Grouper: a dynamic clustering interface to Web search results," in *Proceedings of the 8th International World Wide Web Conference*, Toronto, Canada, May 1999.
- [ZHA00] D. Zhang and Y. Dong, "An efficient algorithm to rank Web resources," *the 8th International Conference*, Amsterdam, Netherlands, May 15 - 19, 2000.

APPENDIX

Data Collection for Search Result Analysis

QUERY: ARTIFICIAL INTELLIGENCE					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
artificial intelligence	18	4	100	VR	8
research	13	4	100	R	7
university	9	5	100	R	8
laboratory	8	4	100	R	0
computer	7	3	100	R	8
institute	7	5	100	R	8
science	6	3	100	R	8
conference	5	5	100	R	8
resource	5	2	100	R	7
guide	4	3	100	R	0
society	4	3	100	R	8
edinburgh	3	4	100	R	8
department	3	6	100	R	7
journal	3	3	100	R	7
project	3	2	100	R	8
system	3	2	67	R	6
technology	3	3	100	R	5
philosophy	2	3	100	R	8
programming	2	3	100	R	7
division	2	6	50	R	3
generation	2	3	0	NR	2
repository	2	3	100	R	6
library	2	3	100	R	4
directory	2	3	50	NR	5
european	2	2	100	R	8
austrian	2	3	100	R	8
language	2	2	0	NR	6
application	2	3	100	R	0
uncertainty	2	2	100	R	7
game	2	2	100	R	7
joint	2	2	0	NR	2
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	4.3	3.3	86.0	87.1	5.9
3%	6.1	3.6	98.1	100	6.5
4%	7.8	3.7	100	100	6.4
5%	8.7	3.9	100	100	6.9

Table 1: Cluster Property Analysis for Query “artificial intelligence”

QUERY: ADIDAS					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
sport	8	3	100	VR	10
shoe	8	4	100	VR	8
soccer	8	6	100	R	9
golf	6	5	100	R	9
team	5	3	100	NR	8
apparel	4	3	67	R	8
Germany	4	5	0	R	8
nike	4	3	100	R	0
tennis	3	5	100	R	9
clothing	3	3	67	NR	7
ball	3	4	67	R	8
store	3	4	67	R	6
game	3	2	100	R	7
club	3	4	50	NR	7
ad	2	4	100	R	7
art	2	3	100	NR	7
boot	2	4	50	R	8
code	2	3	0	NR	7
gear	2	4	100	R	7
image	2	3	50	NR	5
junior	2	4	100	R	0
merchandise	2	3	100	R	9
running	2	2	0	R	0
sponsor	2	4	50	R	7
shopping	2	3	100	R	7
performance	2	3	100	NR	6
revolution	2	3	0	NR	7
sock	2	5	100	R	7
uniform	2	4	100	R	7
wrestling	2	3	100	R	8
play	2	4	0	R	6
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	3.2	3.6	73.2	74.2	6.7
3%	4.6	3.9	80.0	78.6	7.4
4%	5.9	4.0	83.4	87.5	7.5
5%	7.0	4.2	100	80	8.8

Table 2: Cluster Property Analysis for Query “Adidas”

QUERY: AMAZON					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
news	14	5	100	R	7
book	12	3	100	R	9
river	8	3	100	R	9
bookstore	5	3	100	R	8
technology	4	3	75	R	6
software	3	5	100	NR	8
foundation	3	2	100	R	5
commerce	3	4	100	R	7
boycott	3	4	100	R	9
border	3	5	0	NR	9
privacy	3	2	0	NR	6
business	3	5	100	R	7
association	3	3	0	NR	6
report	2	5	100	R	7
comment	2	5	50	NR	7
patent	2	5	100	NR	7
education	2	3	100	R	7
retail	2	6	100	R	8
garden	2	7	100	R	6
finance	2	6	100	R	7
travel	2	4	50	R	8
shopping	2	3	100	R	7
story	2	3	100	NR	7
brazil	2	4	100	R	8
company	2	5	100	R	7
forest	2	4	100	R	7
content	2	3	0	NR	6
associate	2	4	0	NR	6
unit	2	3	100	NR	6
rain	2	3	100	R	9
program	2	3	100	NR	7
asset	2	2	50	NR	8
buy	2	3	100	R	6

Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	3.2	3.9	86.7	79.5	4.3
3%	5.2	3.6	84.4	75	7.4
4%	8.6	3.4	95	100	7.8
5%	9.8	3.5	100	100	8.3

Table 3: Cluster Property Analysis for Query “amazon”

QUERY: AMINO ACID					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
acid	10	4	100	VR	9
composition	7	3	100	VR	7
metabolism	6	2	100	R	7
property	5	2	100	R	7
arginine	4	6	100	VR	9
structure	4	2	100	R	7
sequence	4	3	100	R	6
analysis	4	6	100	R	8
letter	3	5	0	NR	6
database	3	5	67	R	5
block	3	2	67	R	8
research	3	5	100	R	7
disappointing	3	5	0	NR	3
eration	3	2	100	R	3
essential	3	1	100	R	10
supplement	3	5	100	R	0
quiz	3	3	100	R	5
lysine	2	6	100	VR	9
peptide	2	5	100	VR	9
glycine	2	5	100	VR	9
version	2	3	0	NR	7
oxidase	2	5	100	R	9
synthetic	2	4	100	R	9
organic	2	4	100	R	9
bonding	2	3	100	R	7
tool	2	3	100	R	7
glossary	2	3	100	R	7
course	2	4	100	R	7
server	2	2	0	NR	0
laboratory	2	4	100	R	0
pronunciation	2	5	100	R	5
chain	2	4	100	R	7
collection	2	2	100	R	8
identification	2	6	100	R	5
similarity	2	5	100	R	7
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	3.1	3.8	86.7	88.6	6.5
3%	4.2	3.6	84.4	88.2	6.3
4%	5.5	3.5	100	100	7.5
5%	7.0	2.8	100	100	7.5

Table 4: Cluster Property Analysis for Query “amino acid”

QUERY: CHEDDAR CHEESE					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
soup	18	5	100	R	9
recipe	17	5	100	R	6
ingredient	7	4	100	R	6
vermont	6	5	0	R	8
bread	5	6	100	R	8
apple	5	7	0	R	8
pie	4	7	100	R	8
sharp	4	5	0	NR	0
taste	4	4	100	VR	3
market	4	3	25	R	5
price	4	2	100	VR	7
loaf	3	7	100	R	7
chicken	3	6	100	R	8
farmhouse	3	4	100	R	7
company	3	5	100	R	6
cream	3	3	100	R	0
biscuit	2	8	100	R	8
texas	2	4	0	NR	8
product	2	4	100	R	8
island	2	3	0	NR	7
catalog	2	4	100	R	7
intelligence	2	3	0	NR	6
agriculture	2	3	0	NR	0
specialty	2	3	100	R	2
sponsor	2	4	0	NR	2
gourmet	2	4	100	R	6
cook	2	4	50	R	7
cranberry	2	4	100	R	8
cauliflower	2	4	100	R	8
village	2	5	0	NR	6
aging	2	2	100	VR	10
traveler	2	4	0	NR	8
date	2	3	0	NR	7
eggplant	2	7	100	R	7
sauce	2	7	100	R	7
mix	2	6	50	R	7
milk	2	6	0	NR	7
food	2	3	100	R	8
dairy	2	4	100	R	0
onion	2	3	100	NR	0
settlement	2	5	100	R	2
serving	2	3	0	R	5
october	2	4	0	NR	8
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	3.4	4.5	63.4	72.1	5.9
3%	5.8	4.9	76.6	93.8	6.0
4%	7.1	4.8	66.0	91.0	6.2
5%	9.7	5.3	66.7	100	7.5

Table 5: Cluster Property Analysis for Query “cheddar cheese”

QUERY: EDMONTON OILERS					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
hockey	28	4	100	VR	10
sport	18	3	100	VR	10
team	15	4	100	VR	8
news	8	4	100	R	7
league	6	4	100	R	8
game	5	2	100	VR	9
headline	4	3	100	R	7
schedule	4	3	100	VR	6
product	4	3	100	R	8
player	4	3	100	R	7
slam	3	3	0	NR	5
poster	3	4	100	R	6
playoff	3	2	100	R	9
statistic	3	3	100	R	6
score	3	3	100	R	4
star	3	3	100	R	0
skate	3	4	0	R	7
fan	3	4	100	R	0
wednesday	3	3	0	NR	7
alberta	2	6	0	R	8
canada	2	5	0	R	8
dallas	2	4	0	NR	8
directory	2	4	0	NR	6
memorabilia	2	4	100	NR	5
draft	2	5	100	R	0
lineup	2	5	100	R	4
forum	2	3	100	NR	6
march	2	3	0	NR	7
standing	2	2	100	R	10
writer	2	3	0	NR	6
picture	2	4	100	R	6
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	4.7	3.5	71.0	74.2	6.4
3%	6.5	3.3	84.2	89.5	6.5
4%	9.6	3.3	100	100	8.0
5%	13.3	3.5	100	100	8.7

Table 6: Cluster Property Analysis for Query “Edmonton Oilers”

QUERY: GEORGE BUSH					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
president	16	4	100	VR	9
biography	10	4	100	VR	4
united states	8	3	100	VR	7
issue	5	2	100	R	7
republican	3	4	100	VR	9
candidate	3	4	100	R	9
governor	3	4	100	VR	9
american	3	3	100	R	6
address	3	4	100	R	7
politics	3	2	100	VR	7
texas	3	4	0	VR	8
real	3	5	67	NR	0
june	3	3	0	NR	7
election	3	4	100	R	8
webster	2	3	0	R	0
guardian	2	2	0	NR	6
quote	2	5	100	R	7
statement	2	4	50	R	8
report	2	5	100	R	7
news	2	5	100	R	7
profile	2	4	100	R	3
america	2	4	100	R	8
houston	2	3	100	R	8
forty	2	5	100	NR	8
war	2	4	100	R	8
vote	2	4	100	R	8
inaugural	2	3	100	R	8
glimpse	2	3	0	NR	7
school	2	3	100	R	8
law	2	3	100	R	0
life	2	3	100	R	0
dark	2	3	100	NR	8
library	2	3	100	R	4
encyclopedia	2	4	100	NR	7
dance	2	4	100	R	7
airport	2	3	100	R	9
campaign	2	2	100	VR	7
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	3.1	3.6	84.2	81.1	6.5
3%	4.9	3.6	83.4	85.7	6.9
4%	9.8	3.3	100	100	6.8
5%	9.8	3.3	100	100	6.8

Table 7: Cluster Property Analysis for Query “George Bush”

QUERY: INLINE SKATES					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
skate	19	4	100	VR	9
skating	11	5	100	VR	9
hockey	10	6	100	R	9
size	10	3	100	VR	8
toy	9	2	89	R	8
sport	8	3	100	R	8
game	8	3	25	R	7
accessory	7	4	100	VR	0
shop	6	3	100	VR	7
sale	6	3	100	R	6
speed	6	3	100	R	3
ice	6	4	0	NR	7
fitness	5	3	100	R	7
price	5	3	100	VR	7
regular	5	3	60	R	0
adult	5	2	100	R	9
mission	4	2	0	NR	0
tour	3	4	100	NR	2
bearing	3	3	100	VR	4
buy	3	3	100	VR	4
quad	3	3	100	R	0
kid	3	3	100	R	9
guide	3	3	100	R	0
dealer	3	3	100	VR	8
retailer	3	2	100	VR	8
skateboard	3	3	100	VR	7
gear	2	4	100	R	7
round	2	4	0	NR	0
shoe	2	4	100	R	8
clothes	2	3	100	R	8
catalog	2	3	100	VR	4
searching	2	3	100	R	8
recreation	2	3	100	VR	9
shopping	2	5	100	VR	7
selection	2	3	100	R	6
question	2	3	100	NR	4
flash	2	3	50	NR	8
item	2	3	100	R	6
brand	2	3	100	R	0
sizing	2	4	100	R	0
product	2	3	100	VR	9
voyager	2	3	0	NR	7
junior	2	3	100	R	0
expert	2	2	100	R	8
men	2	3	100	R	6
reason	2	2	0	NR	8
fit	2	2	100	R	0
feature	2	2	50	VR	7

Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	4.2	3.1	84.9	83.3	5.5
3%	6.0	3.2	87.5	88.5	5.6
4%	7.6	3.3	80.8	88.2	6.1
5%	7.9	3.4	85.9	93.8	6.5

Table 8: Cluster Property Analysis for Query “inline skates”

QUERY: MUTUAL FUND					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
fund	29	4	100	VR	8
investing	12	4	100	VR	7
investor	8	4	100	VR	8
investment	6	4	100	VR	7
load	6	3	100	VR	0
company	5	4	40	R	6
newsletter	5	6	100	R	7
stock	4	4	100	VR	8
management	4	4	0	R	7
market	4	3	50	R	5
business	4	4	75	R	7
account	4	3	100	VR	6
service	3	4	100	R	7
india	3	5	0	R	7
research	3	5	67	R	7
advisor	3	5	33	R	7
portfolio	3	4	100	VR	5
glossary	3	5	100	R	7
calculator	3	5	100	R	3
cost	2	3	100	VR	7
email	2	3	0	NR	7
manager	2	4	50	R	7
issue	2	3	0	NR	0
plan	2	4	50	R	7
retirement	2	3	50	R	6
economy	2	3	0	R	0
institute	2	4	100	R	4
tool	2	4	100	R	7
letter	2	4	0	NR	6
american	2	5	0	R	6
screening	2	5	100	R	0
performance	2	5	50	R	3
family	2	5	100	R	7
canada	2	5	0	R	8
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	4.2	4.1	63.7	91.2	5.7
3%	5.9	4.2	77.1	100	6.3
4%	7.6	3.9	80.4	100	6.3
5%	10.1	4.1	91.4	100	6.1

Table 9: Cluster Property Analysis for Query “mutual fund”

QUERY: NAPSTER					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
news	15	5	100	R	7
music	14	4	100	VR	9
February	8	4	0	NR	8
industry	6	5	100	R	8
company	4	5	100	R	9
offer	4	3	100	R	6
court	4	4	100	R	7
file	3	5	33	NR	0
pc	3	5	33	R	8
camp	3	4	0	NR	8
billion	3	4	0	R	7
people	3	2	67	R	8
network	3	3	0	NR	0
label	3	4	100	NR	5
artist	3	4	100	VR	8
chaos	3	3	100	R	7
notice	2	5	0	NR	6
standard	2	5	100	R	0
recording	2	4	100	VR	3
software	2	5	100	VR	7
sharing	2	3	100	VR	8
filter	2	4	100	R	8
settle	2	4	100	R	0
story	2	4	100	R	7
technology	2	4	100	R	5
entertainment	2	4	100	VR	7
ruling	2	5	100	R	7
lawsuit	2	5	100	VR	7
deal	2	4	100	R	7
decision	2	3	50	NR	3
judge	2	5	100	R	8
guide	2	5	100	NR	0
user	2	4	100	R	7
lawyer	2	4	100	R	7
client	2	4	100	R	0
CEO	2	4	100	R	10
clone	2	2	100	R	0
freedom	2	2	100	R	6
animation	2	3	0	NR	0
revolution	2	4	100	R	7
opinion	2	2	100	R	6
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	3.2	4.0	80.1	78.0	5.6
3%	5.1	4.0	64.6	68.8	6.6
4%	7.9	4.3	85.7	85.7	7.7
5%	10.8	4.5	75.0	75.0	8.0

Table 10: Cluster Property Analysis for Query “Napster”

QUERY: SODIUM BENZOATE					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
product	9	3	100	R	8
acid	8	4	100	R	9
caffeine	7	6	100	R	9
water	6	4	100	R	9
potassium	6	4	100	VR	9
chemical	6	4	100	VR	8
brand	5	3	100	R	0
bicarbonate	4	6	100	VR	8
spice	4	5	100	NR	8
ingredient	4	3	100	R	8
description	4	4	100	R	5
purity	4	2	100	VR	7
offer	4	3	100	R	3
specification	3	5	100	VR	7
fructose	3	7	100	NR	8
soda	3	5	100	R	0
sugar	3	6	100	NR	9
food	3	3	33	R	8
nutrition	3	2	100	R	0
encyclopedia	3	3	100	R	7
cellulose	2	7	100	NR	8
benzine	2	6	100	R	8
syrup	2	6	100	NR	8
vinegar	2	6	100	NR	9
juice	2	4	0	NR	8
UK	2	4	100	NR	8
mountain	2	3	0	NR	8
commodity	2	2	100	R	8
supplier	2	4	100	R	7
producer	2	4	100	R	4
gourmet	2	4	0	NR	7
corn	2	3	0	NR	9
formula	2	5	100	R	0
safety	2	2	100	R	3
flavor	2	3	100	R	0
resistance	2	3	100	R	2
manufacturer	2	4	100	R	7
drug	2	2	0	NR	8
property	2	2	100	VR	6
price	2	3	100	R	7
effect	2	1	100	R	10
chemistry	2	3	0	R	8

Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	3.2	3.9	84.1	71.4	6.5
3%	4.6	4.1	96.7	85.0	6.5
4%	5.5	3.9	100	92.3	7.0
5%	6.7	4.0	100	100	7.4

Table 11: Cluster Property Analysis for Query “sodium benzoate”

QUERY: STEPHEN KING					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
horror	10	3	100	VR	9
news	9	5	100	R	7
book	6	4	100	VR	8
fan	4	3	100	R	0
domain	4	2	0	NR	0
revue	3	5	0	NR	3
movie	3	4	100	R	6
novel	3	5	100	VR	6
check	3	5	0	NR	0
bookstore	3	4	100	R	5
club	3	5	0	NR	7
review	3	2	100	R	8
maine	3	5	100	VR	8
mailing	3	4	100	NR	5
author	3	4	100	VR	7
name	3	4	0	NR	3
fear	2	2	100	R	9
bibliography	2	6	50	R	2
description	2	5	0	NR	5
literature	2	4	100	R	8
art	2	3	100	R	8
cover	2	3	100	R	5
gallery	2	4	100	R	7
system	2	2	100	NR	6
portland	2	4	100	VR	8
hit	2	3	100	R	6
story	2	4	100	R	6
interview	2	4	100	R	6
author	2	4	100	VR	8
rose	2	2	0	NR	9
resource	2	2	50	R	7
February	2	4	0	NR	8
form	2	5	100	R	2
writer	2	5	50	R	7
father	2	4	50	R	7
past	2	2	50	R	9
future	2	2	100	R	9
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	2.9	3.7	71.6	73.0	6.1
3%	4.1	4.0	68.8	62.5	5.1
4%	6.6	3.4	80.0	80.0	4.8
5%	8.3	4.0	100	100	8.0

Table 12: Cluster Property Analysis for Query “Stephen King”

QUERY: UNIVERSITY OF ALBERTA					
Cluster Topic	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance	Path Intuitiveness
department	21	4	100	VR	7
faculty	15	3	100	VR	8
science	8	3	100	VR	8
Canada	8	5	0	R	8
engineer	8	4	100	R	2
service	7	4	57	R	5
research	7	5	100	VR	7
edmonton	7	5	0	R	9
student	5	4	100	VR	9
resource	5	2	100	R	5
school	4	4	100	VR	8
program	4	4	100	R	6
dentistry	4	6	100	R	9
forestry	4	6	100	R	9
chemistry	4	4	100	R	8
economics	4	4	100	R	9
education	4	4	100	VR	9
agriculture	4	3	100	R	6
server	4	4	50	R	0
biology	3	5	100	R	9
datum	3	3	100	R	8
america	3	5	0	NR	8
north	3	4	0	NR	8
bear	3	3	100	VR	6
library	3	3	100	VR	4
health	3	3	100	R	5
business	2	4	100	R	7
association	2	3	100	R	8
people	2	2	100	R	7
law	2	3	100	R	0
study	2	6	100	R	7
course	2	4	100	VR	7
athletics	2	4	100	R	8
mining	2	3	100	R	0
campus	2	5	100	VR	3
petroleum	2	4	100	R	7
food	2	3	100	R	7
undergraduate	2	5	100	VR	9
graduate	2	4	100	VR	9
alumnus	2	4	100	R	8
human	2	2	0	NR	9
map	2	3	100	R	4
opportunity	2	3	100	R	0
employment	2	2	100	R	3
serving	2	2	100	NR	0
message	2	2	100	NR	0
Averages (per cluster)					
Cluster Size Threshold	Cluster Size	Path Length	Cluster Precision(%)	Topic Relevance(%)	Path Intuitiveness
2%	4.1	3.7	52.2	89.1	6.2
3%	5.7	4.0	81.0	92.3	6.9
4%	6.7	4.1	84.6	100	6.9
5%	9.1	3.9	75.7	100	6.8

Table 13: Cluster Property Analysis for Query “University of Alberta”

...So put me on a highway
And show me a sign
And take it to the limit one more time...

- *The Eagles*