SEARCHING AND MINING THE WEB FOR PERSONALIZED AND SPECIALIZED INFORMATION

by

Michael Chiu-Lung Chau

A Dissertation Submitted to the Faculty of the

COMMITTEE ON BUSINESS ADMINISTRATION

In Partial Fulfillment of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY
WITH A MAJOR IN MANAGEMENT

In the Graduate College

THE UNIVERSITY OF ARIZONA

2003

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

THE UNIVERSITY OF ARIZONA ®
GRADUATE COLLEGE

As members of the Final Examination Committee, we certify that we have read the dissertation prepared by Michael Chiu-Lung Chau entitled Searching and Mining the Web for Personalized and Specialized Information

and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy

Hsinchun Chen, Ph.D.	4/11/03 Date
Olivia R. Liu Sheng, Ph.D.	4/11/03 Date
Daniel D. Zeng, Ph.D.	4/11/03 Date
D. Terence Langendoen, Ph.D.	4/11/03 Date
Simin Karimi, Ph.D.	4-11-03 Date

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copy of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

Hsinchun Chen, Ph.D. Dissertation Director	4-14-03 Date

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: MICHAEL CHAN

ACKNOWLEDGEMENTS

First of all, I would like to thank my dissertation advisor and mentor, Professor Hsinchun Chen, for his guidance and encouragement throughout my five years at the University of Arizona. It has been an invaluable opportunity for me to work in the Artificial Intelligence Lab under his direction. I also thank my major committee members, Dr. Olivia R. Liu Sheng and Dr. Daniel D. Zeng, and my minor committee members in the Department of Linguistics, Dr. D. Terence Langendoen and Dr. Simin Karimi, for their guidance and encouragement. I also thank all the faculty members in the Department of MIS for their support. In addition, I am grateful to my undergraduate advisors Dr. Jerome Yen, Dr. Lester Yee and Dr. Christopher C. Yang for introducing me to MIS research.

My dissertation has been partly supported by the National Science Foundation (#9817473, "High-performance Digital Library Systems: From Information Retrieval to Knowledge Management" and #9800696, "An Intelligent CSCW Workbench: Analysis, Visualization, and Agents") and the National Institute of Health (#1-R01-LM06919-1A1, "UMLS Enhanced Dynamic Agents to Manage Medical Knowledge"). Most projects discussed in this dissertation have been supported by other AI Lab members. Wojciech Wyzga, Haiyan Fan, Harry Li, Andy Clements, David Hendriawan, Ye Fang, Hadi Bunnalim, Ming Yin, Bill Oliver, Kristin Tolle, Chikit Chan, and Esther Chou contributed to the CI Spider and the Meta Spider systems. Haiyan Fan, Bill Oliver, Andy Lowe, Fiona Sung, Xiaoyun Sun, Hui Liu, and Ann Lally participated in the Cancer Spider. Kalai Chinnaveerappan helped in developing Nano Spider. David Hendriawan, Michael Huang, and Yin Ming contributed to the Collaborative Spider project. Wojciech Wyzga, Haiyan Fan, Yin Ming, Yohanes Santoso, and Gondy Leroy contributed to the Hopfield Net Spider. I appreciate not only their efforts but also all the good time we shared.

I am fortunate to meet an excellent group of doctoral students in the department, who share their insights and ideas, as well as laughters and tears with me. I would like to especially thank Dorbin Ng, Kristin Tolle, Bin Zhu, Chienting Lin, Thian-Huat Ong, Dmitri Roussinov, Rosie Hauck, Gondy Leroy, Xiao Fang, Poh-Kim Tay, Taeha Kim, Dongsong Zhang, Zan Huang, Jennifer Xu, Jing Zhang, Haidong Bi, Wingyan Chung, Jinwei Cao, Ming Lin, Jialun Qin, Yilu Zhou, Yiwen Zhang, Dan McDonald, Byron Marshall, Gang Wang, Rong Zheng, and Jason Li, for the good times we spent together on and off campus. I also worked with excellent colleagues in the AI Lab. Besides the people mentioned, I greatly enjoyed working with Homa Atabakhsh, David Gonzalez, Yang Xiang, Alan Yip, Lihua Cao, Xue Wei, Fei Guo, Ravi Parimi, Pei He, Yi Qin, Yintak Lam, Shan Fu, Suresh Nandiraju, Mark Chen, Lu Tseng, Yon Hsu, Tailong Ke, Tim Petersen, and Jenny Schroeder. I also thank Barbara Sears for editing my papers.

Lastly but most importantly, I greatly appreciate the support from my parents and my family, who have always been giving me unfailing love and care. I am also extremely grateful to Fiona, who has walked a long way with me in my life, nourishing me with support, care, friendship, hope, and love.

DEDICATION

I dedicate this dissertation in memory to my best friend, Karen Cheng (1976-2002). Karen is a very sweet, caring, and intelligent girl who has been an angel to most people around her. I feel myself to be very fortunate to have known her and become a very close friend of her since we were young. We grew up together, we hanged out together, we talked about our dreams and secrets, we laughed and cried together, and we cared for each other like brother and sister. Not a single day has passed since she left that I didn't think of her. Sometimes, I still cannot believe that she is not here with us any more. But, no matter what, I do believe that the warmth and lights that she has shed on our lives will never dim, the happiness and memories that she has left in our minds will never fade, and the friendship and love that she has planted in our hearts will never cease, till we meet again.

TABLE OF CONTENTS

TABLE OF CONTENTS - *Continued*

TABLE OF CONTENTS - *Continued*

TABLE OF CONTENTS - *Continued*

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

With the rapid growth of the Web, users are often faced with the problem of information overload and find it difficult to search for relevant and useful information on the Web. Besides general-purpose search engines, there exist some alternative approaches that can help users perform searches on the Web more effectively and efficiently. Personalized search agents and specialized search engines are two such approaches. The goal of this dissertation is to study how machine learning and artificial intelligence techniques can be used to improve these approaches.

A system development research process was adopted as the methodology in this dissertation. In the first part of the dissertation, five different personalized search agents, namely CI Spider, Meta Spider, Cancer Spider, Nano Spider, and Collaborative Spider, were developed. These spiders combine Web searching with various techniques such as noun phrasing, text clustering, and multi-agent technologies to help satisfy users' information needs in different domains and different contexts. Individual experiments were designed and conducted to evaluate the proposed approach and the experimental results showed that the prototype systems performed better than or comparable to traditional search methods.

The second part of the dissertation aims to investigate how artificial intelligence techniques can be used to facilitate the development of specialized search engines. A Hopfield Net spider was proposed to locate from the Web URLs that are relevant to a

given domain. A feature-based machine-learning text classifier also was proposed to perform filtering on Web pages. A prototype system was built for each approach. Both systems were evaluated and the results demonstrated that they both outperformed traditional approaches.

This dissertation has two main contributions. Firstly, it demonstrated how machine learning and artificial intelligence techniques can be used to improve and enhance the development of personalized search agents and specialized search engines. Secondly, it provided a set of tools that can facilitate users in their Web searching and Web mining activities in various contexts.

CHAPTER 1: INTRODUCTION

With more than two billion pages contributed by millions of Web page authors and organizations, the World Wide Web is a rich, enormous knowledge base that can be useful to many applications. The knowledge comes not only from the content of the pages themselves, but also from the unique characteristics of the Web, such as its hyperlink structure and its diversity in content and languages. These characteristics often reveal interesting patterns and new knowledge that can be very useful to various applications. First, such knowledge can be used to improve users' efficiency and effectiveness in searching for information on the Web. Second, knowledge obtained from the Web also can be used for other applications that are not related to the Web, such as decision-making support or business management.

Due to the Web's large size, its unstructured and dynamic content, and its multilingual nature, extracting useful knowledge from it has become a challenging research problem. Although development of general-purpose search engines, such as Google (http://www.google.com/), AltaVista (http://www.altavista.com/) and Lycos (http://www.lycos.com/), has alleviated the problem to a great extent, exponential growth of the Web is making it impossible for these search engines to collect and index all the Web pages and refresh their indexes frequently enough to keep them up-to-date. Most search engines present to users search results that are incomplete or outdated, usually leaving users confused and frustrated.

Another problem with general search engines is the poor retrieval performance when only a single search engine is used. It has been estimated that none of the search engines available indexes more than 16% of the total Web that could be indexed (Lawrence & Giles, 1999). Even worse, each search engine maintains its own searching and ranking algorithm as well as query formation and freshness standard. Unless the different features of each search engine are known, searches will be inefficient and ineffective. From the user's point of view, dealing with an array of different interfaces and understanding the idiosyncrasies of each search engine is too burdensome. The development of meta-search engines has alleviated this problem. However, how the different results are combined and presented to the user greatly affects the effectiveness of these tools.

In addition, given the huge number of daily hits, most search engines are not able to provide enough computational power to satisfy each user's information need. Analysis of search results, such as verifying that the Web pages retrieved still exist or clustering of Web pages into different categories, are not available in most search engines. Search results are usually presented in a ranked list fashion; users cannot get a whole picture of what the Web pages are about until they click on every page and read the contents. This can be time-consuming and frustrating in a dynamic, fast-changing electronic information environment.

Two possible approaches have been proposed to address the above problems. The first approach is personalized search agents (also known as spiders or crawlers). Search agents can provide users with customized and real-time search and analysis. Because these

programs usually run on the client-machine, more computational power is available for the search process and more functionalities are possible. The second approach is the use of domain-specific search engines, also known as specialized search engines or vertical search engines. These search engines keep search indexes only in particular domains. Because they only focus on a small subset of the Web, they are often able to build a more comprehensive index in the domains of interest and they usually provide customized features. For example, BuildingOnline (http://www.buildlingonline.com/) specializes in searching in the building industry domain on the Web, and LawCrawler (http://www.lawcrawler.com/) specializes in searching for legal information on the Internet.

There are some challenges to these approaches. A search agent approach needs to effectively search for personalized information which is relevant to a user's search queries and provide sophisticated, customized analysis. It also needs to search the Web in real time, using dynamic searching or meta-searching methods. On the other hand, specialized search engines are not easy to build. In the process, the system needs to identify URLs on the Web that point to relevant, high-quality Web pages. It also needs to learn to automatically classify relevant pages from irrelevant ones.

This dissertation mainly focuses on how machine learning and artificial intelligence techniques can be used to achieve and improve these two approaches. The rest of the dissertation is organized as follows. In Chapter 2, I review some relevant literature in information retrieval, machine learning, Web mining, search agents, and specialized

search engines. Chapters 3 to 5 are devoted to personalized search agents. Chapter 3 discusses the design, implementation, and evaluation of two personalized search agents called CI Spider and Meta Spider (Chau et al., 2001b; Chen et al., 2001; Chen et al., 2002). These two agents combine Web searching, natural language processing, and text clustering techniques to provide advanced searching for Web users. In Chapter 4, two specialized search agents are discussed. These two agents, called Cancer Spider and Nano Spider, provide customized search features for users looking for information in cancer research and nanotechnology respectively (Chau et al., 2002a). Chapter 5 presents a multi-agent architecture that supports collaborative Web searching and Web mining (Chau et al., 2003). By utilizing multi-agent and collaborative technologies, a collaborative spider was built to allow users to share their search sessions and leverage on their search findings. In Chapters 6 and 7, I present my research on specialized search engine development. Chapter 6 investigates how artificial intelligence techniques can be used to locate Web pages relevant to a particular domain (Chau & Chen, 2003). In particular, a Hopfield Net Spider designed based on spreading activation is presented. In Chapter 7, I propose a feature-based text classification approach that can be applied to Web page filtering during the development of specialized search engines (Chau, 2002). Finally, in Chapter 8, I summarize the contributions of my dissertation and suggest some possible future directions.

CHAPTER 2: LITERATURE REVIEW AND RESEARCH FORMULATION

Information retrieval has been studied extensively since the 1970's. Various techniques, such as automatic indexing, text classification and clustering, natural language processing, and machine learning, have been applied to helping users improve the effectiveness and efficiency when they are searching for information. Since the advent of the Web, the problem of information overload has become much more important. It has been estimated that there are more than two billion documents on the Web, and the number is still growing at a rapid rate (Lyman & Varian, 2000). Much time and effort is required for Web users to search for the relevant information on the Web and then analyze the information collected in the correct context.

Machine learning techniques represent one possible approach to the problem of information overload on the Web. Artificial intelligence and machine learning techniques have been applied in many important and useful applications in both scientific and business domains, and data mining research has become a significant subfield in this area. Machine learning techniques also have been used in information retrieval and text mining applications. The various activities and efforts in this area are referred to as *Web mining*. The term Web mining was coined by Etzioni (1996) to denote the use of data mining techniques to automatically discover Web documents and services, extract information from Web resources, and uncover general patterns on the Web. Over the years, Web mining research has been extended to cover the use of data mining as well as other similar techniques to discover resources, patterns, and knowledge from the Web and

Web-related data (such as Web usage data or Web server logs). In this dissertation, I have adopted a broad definition that considers Web mining to be "the discovery and analysis of useful information from the World Wide Web" (Cooley et al., 1997).

Web mining research overlaps substantially with other areas, including data mining, text mining, information retrieval, and Web retrieval. A possible classification of research in these areas is shown in Table 2.1. The classification is based on two aspects: the purpose and the data sources. *Retrieval* research focuses on retrieving relevant, existing data or documents from a large database or document repository, while *mining* research focuses on discovering new information or knowledge from data. For example, data retrieval techniques mainly concern improving the speed of retrieving data from a database, while data mining techniques analyze the data and try to identify interesting patterns. It should be noted, however, that the distinction between information retrieval and text mining is at times unclear. Many applications, such as text classification and text clustering, are often considered both information retrieval and text mining (e.g., Voorhees & Harman, 1998; Trybula, 1999). In fact, almost all text mining techniques have been investigated by the information retrieval community such as the Text REtrieval Conference (TREC). Because information retrieval research has the primary goals of indexing and searching, I consider areas such as document clustering to be an instance of text mining techniques that is also part of the retrieval process. Similarly, Web retrieval and Web mining share many similarities. Web document clustering has been studied both in the context of Web retrieval and that of Web mining. On the other hand, however, Web mining is not simply the application of information retrieval and text mining techniques to Web pages; it also

involves non-textual data such as Web server logs and other transaction-based data. From this point of view, Web retrieval and Web mining are considered two overlapping areas, in which the main criterion for classification of an application is the specific purpose of the application.

It is also interesting to note that although Web mining relies heavily on data mining and text mining techniques, not all techniques applied to Web mining are based on data mining or text mining. Some techniques, such as Web link structure analysis, are unique to Web mining. In general, it is reasonable to consider Web mining as a subfield of data mining, but not a subfield of text mining, since some Web data are not textual (e.g., Web log data).

Table 2.1: A classification of retrieval and mining techniques and applications.

		Data/information sources		
		Any data	**Textual data**	**Web-related data**
Purpose	**Retrieving known data or documents efficiently and effectively**	Data Retrieval/ Database	Information Retrieval	Web Retrieval
	Finding new patterns or knowledge previously unknown to the system	Data Mining	Text Mining	Web Mining

As can be seen, Web mining research is at the intersection of several established research areas, including information retrieval, Web retrieval, machine learning, database, data mining, and text mining. Most previous papers have viewed Web mining from a database or data mining perspective (e.g., Cooley et al., 1997; Chakrabarti, 2000; Han & Chang, 2002). On the other hand, research in machine learning and information retrieval also has

played a very important role in Web mining research. Machine learning is the basis for most data mining and text mining techniques, and information retrieval research has largely influenced the research directions of Web mining applications. In this chapter, I will review several relevant and overlapping areas, including machine learning and their applications in information retrieval, Web mining research, Web search agents, and specialized search engines.

2.1 Machine Learning Techniques and Information Retrieval

Since the invention of the first computer in the 1940's, researchers have been attempting to create knowledgeable, learnable, and intelligent computers. Many knowledge-based systems have been built for various applications such as medical diagnosis, engineering troubleshooting, and business decision-making (Hayes-Roth & Jocobstein, 1994). However, most of these systems have been designed to acquire knowledge manually from human experts, which can be a very time-consuming and labor-intensive process. To address the problem, machine learning algorithms have been developed to acquire knowledge automatically from examples or source data. Simon (1983) defines machine learning as "any process by which a system improves its performance." Mitchell (1997) gives a similar definition, which considers machine learning as "the study of computer algorithms that improve automatically through experience." Machine learning algorithms can be classified as supervised learning or unsupervised learning. In supervised learning, training examples consist of input/output pair patterns. The goal of the learning algorithm is to learn how to predict the output values of new examples, based on their input values. In unsupervised learning, training examples contain only the input patterns only and no

explicit target output is associated with each input. The learning algorithm needs to generalize from the input patterns and discover the output values.

2.1.1 Machine Learning Paradigms

Over the past decades, many machine learning systems have been developed. Langley and Simon (1995) identify five major areas of machine learning research, namely neural networks, cased-based learning, genetic algorithms, rule induction, and analytic learning. Chen (1995) identifies three classes of machine learning techniques, which include symbolic learning, neural networks, and evolution-based algorithms. Based on the two classifications and a review of the field, I have adopted a similar framework that combines the two and have identified the following five major paradigms: (1) probabilistic models, (2) symbolic learning and rule induction, (3) neural networks, (4) evolution-based models, and (5) analytic learning and fuzzy logic. In this section, I will briefly review research in the five areas.

2.1.1.1 Probabilistic Models

The use of probabilistic models was one of the earliest attempts to perform machine learning. Of this type of models, the most popular example is the Bayesian method, which originated in research in the field of pattern recognition (Duda & Hart, 1973). Often used in the context of classifying different objects into predefined classes based on a set of features, a Bayesian model stores the probability of each class, the probability of each feature, and the probability of each feature given each class based on the training data. When a new instance is encountered, it can be classified according to these

probabilities (Langley et al., 1992). A variation of the Bayesian model, called the Naïve Bayesian model, assumes that all features are mutually independent within each class. Because of its simplicity, the Naïve Bayesian model has been widely used in different applications and different domains (Fisher, 1987; Kononenko, 1993).

2.1.1.2 Symbolic Learning and Rule Induction

Symbolic learning can be classified based on the underlying learning strategies such as rote learning, learning by being told, learning by analogy, learning from examples, and learning from discovery (Cohen & Feigenbaum, 1982; Carbonell et al., 1983). Among these, learning from examples appears to be the most promising symbolic learning technique for knowledge discovery and data mining by applying algorithms that attempt to induce a general concept description that best describes the different classes of the training examples. Numerous algorithms have been developed, each using one ore more different techniques to identify patterns that are useful in generating the concept description. Among these algorithms, Quinlan's ID3 decision-tree building algorithm (Quinlan, 1983) and its variations such as C4.5 (Quinlan, 1993) have become one of the most widely used symbolic learning techniques. Given a set of objects, ID3 produces a decision tree that attempts to classify all the given objects correctly. At each step, the algorithm finds the attribute that best divides the objects into the different classes by minimizing entropy (information uncertainty). After all objects have been classified or all attributes have been used, the results can be represented by a decision tree or a set of production rules.

2.1.1.3 Neural Networks

Artificial neural networks attempt to achieve human-like performance by modeling the human nervous systems. A neural network is a graph of many active nodes (neurons) that are connected with each other by weighted links (synapses). While knowledge is represented by symbolic descriptions such as decision tree and production rules in symbolic learning, knowledge is learned and remembered by a network of interconnected neurons, weighted synapses, and threshold logic units (Rumelhart et al., 1986a; Lippmann, 1987). Learning algorithms can be applied to adjust the connection weights based on learning examples such that the network can predict or classify unknown examples correctly. Activation algorithms over the nodes can then be used to retrieve concepts and knowledge from the network (Belew, 1989; Kwok, 1989; Chen & Ng, 1995).

Many different types of neural networks have been developed, among which the feedforwrd/backpropagation model is the most widely used. Backpropagation networks are fully connected, layered, feed-forward networks in which activations flow from the input layer through the hidden layer, then to the output layer (Rumelhart et al., 1986b). The network usually starts with a set of random weights and adjusts its weights according to each learning example. Each learning example is passed through the network to activate the nodes. The network's actual output is then compared to the target output and the error estimates are then propagated back to the hidden and input layers. The network updates its weights incrementally according to these error estimates until the network stabilizes. Other popular neural network models include Kohonen's self-organizing map

and the Hopfield network Self-organizing maps have been widely used in unsupervised learning, clustering, and pattern recognition (Kohonen, 1995). Hopfield networks have been used mostly in search and optimization applications (Hopfield, 1982).

2.1.1.4 Evolution-based Algorithms

Another class of machine learning algorithms consists of evolution-based algorithms that rely on analogies to natural processes and Darwinian survival of the fittest. Fogel (1994) identifies three categories of evolution-based algorithms: genetic algorithms, evolution strategies, and evolutionary programming. Among these, genetic algorithms have had rising popularity and have been successfully applied to various optimization problems. Genetic algorithms were developed based on the principle of genetics (Goldberg, 1989; Michalewicz, 1992). A population of individuals in which each individual represents a potential solution is first initiated. This population undergoes a set of genetic operations known as crossover and mutation. Crossover is a high-level process that aims at exploitation while mutation is a unary process that aims at exploration. The individuals strive for survival based on a selection scheme that is biased toward selecting fitter individuals (individuals that represent better solutions). The selected individuals form the next generation and the process continues. After some number of generations the program converges and the optimum solution is represented by the best individual.

2.1.1.5 Analytic Learning and Fuzzy Logic

Analytic learning represents knowledge as logical rules, and performs reasoning on such rules to search for proofs. Proofs can be compiled into more complex rules to solve

similar problems with a smaller number of searching required. For example, Samuelson and Rayner (1991) use analytic learning to represent grammatical rules that improve the speed of a parsing system.

While traditional analytic learning systems depend on hard computing rules, there is usually no clear distinction between values and classes in the real world. To address this problem, fuzzy systems and fuzzy logic have been proposed. Fuzzy systems allow the values of False or True to operate over the range of real numbers from 0 to 1 (Zedah, 1965). Fuzziness has been applied to allow for imprecision and approximate reasoning.

2.1.1.6 Hybrid Approaches

As Langley and Simon (1995) have pointed out, the reasons for differentiating the paradigms are "more historical than scientific." The boundaries between the different paradigms are usually unclear and there are many systems that have been built to combine different approaches. For example, fuzzy logic has been applied to rule induction and genetic algorithms (e.g., Mendes et al., 2001), genetic algorithm has been combined with neural network (e.g., Maniezzo, 1994), and neural network has a close resemblance to probabilistic model and fuzzy logic and can be easily combined (e.g., Paass, 1990).

2.1.2 Applications of Machine Learning Techniques in Information Retrieval

Learning techniques had been applied in information retrieval (IR) applications long before the recent advances of the Web. In a recent ARIST article, Cunningham et al.

(1999) provided an extensive review of applications of machine learning techniques in IR. In this section, I will briefly survey some of the research in this area, covering the use of machine learning in information extraction, relevance feedback, information filtering, text classification, and text clustering.

Information extraction is one area in which machine learning is applied in IR, by means of techniques designed to identify useful information from text documents automatically. Named-entity extraction is one of the most widely studied sub-fields. It refers to the automatic identification from text documents of the names of entities of interest, such as persons (e.g., "John Doe"), locations (e.g., "Washington, D.C."), and organizations (e.g., "National Science Foundation"). It also includes the identification of other patterns, such as dates, times, number expressions, dollar amounts, email addresses, and Web addresses (URLs). The Message Understanding Conference (MUC) series has been the major forum where researchers in this area meet and compare the performance of their entity extraction systems (Chinchor, 1998). Machine learning is one of the major approaches. Machine learning-based entity extraction systems rely on algorithms rather than human-created rules to extract knowledge or identify patterns from texts. Examples of machine learning algorithms include neural networks, decision tree (Baluja et al., 1999), Hidden Markov Model (Miller et al., 1998), and entropy maximization (Borthwick et al., 1998). Instead of relying on a single approach, most existing information extraction systems combine machine learning with other approaches (such as a rule-based approach or a statistical approach). Many systems that used a combined approach were evaluated at the MUC-7 conference. The best systems were able to achieve over 90% in both precision

and recall rates in extracting persons, locations, organizations, dates, times, currencies, and percentages from a collection of New York Times news articles (Chinchor, 1998).

Relevance feedback is a well-known method used in IR systems to help users conduct searches iteratively and reformulate search queries based on evaluation of previously retrieved documents (Ide, 1971; Rocchio, 1971). The main assumption is that documents relevant to a particular query are represented by a set of similar keywords (Salton, 1989). After a user rates the relevance of a set of retrieved documents, the query can be reformulated by adding a set of terms from the relevant documents and subtracting a set of terms from the irrelevant documents. It has been shown that a single iteration of relevance feedback can significantly improve search precision and recall (Salton, 1989). Probabilistic techniques have been applied to relevance feedback by estimating the probability of relevance of a given document to a user. Using relevance feedback, a model can learn the common characteristics of a set of relevant documents in order to estimate the probability of relevance for the remaining documents in a collection (Fuhr & Buckley, 1991; Fuhr & Pfeifer, 1994). Various machine learning algorithms, such as genetic algorithms, ID3, and simulated annealing, have been used in relevance feedback applications (Kraft et al., 1995; 1997; Chen et al., 1998d).

Similarly to relevance feedback, *information filtering and recommendation* techniques use user evaluation to improve IR system performance. The main difference is that while relevance feedback helps users reformulate their search queries, information filtering techniques try to learn about users' interests based on their evaluations and actions, and

then to use this information to analyze new documents. Information filtering systems are usually designed to alleviate the problem of information overload in IR systems. The NewsWeeder system allows users to give an article a rating from 1 to 5. After a user has rated enough articles, the system learns the user's interests from these examples and identifies USENET news articles that the system predicts to be interesting to the user (Lang, 1995). Decision tree also has been used for news-article filtering (Green & Edwards, 1996). Another approach is called collaborative filtering or recommender systems, in which collaboration is achieved as the system allows users to help one another perform filtering by recording their reactions to documents they read (Goldberg et al., 1992). One example is the GroupLens system, which performs collaborative filtering on USENET news articles (Konstan et al., 1997). GroupLens recommends articles that may be of interest to a user based on the preferences of other users who have shown interests similar to those of this user. Many personalization and collaborative systems have been implemented as software agents to help users in information systems (Maes, 1994).

Text classification and *text clustering* have been studied extensively in traditional IR literature. Text classification is the classification of textual documents into predefined categories (supervised learning), while text clustering groups documents into categories dynamically defined based on their similarities (unsupervised learning). Although their usefulness is still under constant debate (Voorhees, 1985; Hearst & Pedersen, 1996; Wu et al., 2001), the use of classification and clustering is based on the Cluster Hypothesis: "closely associated documents tend to be relevant to the same requests" (van Rijsbergen,

1979). Machine learning is the basis of most text classification and clustering applications. Text classification has been extensively reported at SIGIR conferences and evaluated on standard testbeds. For example, the Naïve Bayesian method has been widely used (e.g., Koller & Sahami, 1997; Lewis & Ringuette, 1994; McCallum et al., 1999). Using the joint probabilities of words and categories calculated by using all documents, this method estimates the probability that a document belongs to a given category. Documents with a probability above a certain threshold are considered relevant. The k-nearest neighbor method is another widely used approach to text classification. For a given document, the *k* neighbors that are most similar to a given document are first identified (Iwayama & Tokunaga, 1995; Masand et al., 1992). The categories of these neighbors are then used to decide the category of the given document. A threshold is also used for each category. Neural network programs also have been applied to text classification, usually employing the feedforward/backpropagation neural network model (Wiener et al., 1995; Ng et al., 1997; Lam & Lee, 1999). Term frequencies or TF*IDF scores (term frequency multiplied by inverse document frequency) of the terms are used to form a vector (Salton, 1989) which can be used as the input to the network. Based on learning examples, the network will be trained to predict the category of a document. Another new technique used in text classification is called support vector machine (SVM), a statistical method that tries to find a hyperplane that best separates two classes (Vapnik, 1995; 1998). Joachims first applied SVM to text classification (Joachims, 1998). It has been shown that SVM achieved the best performance on the Reuters-21578 data set for document classification (Yang & Liu, 1999).

Similarly to text classification, text clustering tries to assign documents into different categories based on their similarities. However, in text clustering, there are no predefined categories; all categories are dynamically defined. There are two types of clustering algorithms, namely hierarchical clustering and non-hierarchical clustering. The *k*-nearest neighbor method and Ward's algorithm (Ward, 1963) are the most widely used hierarchical clustering methods. Willet (1988) provided an excellent review of hierarchical agglomerative clustering algorithms for document retrieval. For non-hierarchical clustering, one of the most common approaches is the K-means algorithm. It uses the technique of local optimization, in which a neighborhood of partitions is defined for each partition. The algorithm starts with an initial set of clusters, examines each document and searches through the set of clusters, and moves to that cluster for which the distance between the document and the centroid is minimum. The centroid position is recalculated every time a document is added. The algorithm stops when all documents have been grouped into the final required number of clusters (Rocchio, 1966). The Single-Pass method (Hill, 1968) is also widely used. However, its performance depends on the order of the input vectors and it tends to produce large clusters (Rasmussen, 1992). Suffix Tree Clustering, a linear time clustering algorithm that identifies phrases common to groups of documents, is another incremental clustering technique (Zamir & Etzioni, 1998). Kraft et al. (1999) and Chen et al. (2000) also have proposed an approach to applying fuzzy clustering to information retrieval systems.

Another approach often used in recent years is the neural network approach. For example, Kohonen's self-organizing map (SOM), a type of neural network that produces a 2-

dimensional grid representation for n-dimensional features, has been widely applied in IR (Lin et al., 1991; Kohonen, 1995; Orwig et al., 1997). The self-organizing map can be either multi-layered or single-layered. First, the input nodes, output nodes, and connection weights are initialized. Each element is then represented by a vector of N terms and is presented to the system. The distance d_j between the input and each output node j is computed. A winning node with minimum d_j is then selected. After the network stabilizes, the top phrase from each node is selected as the label, and adjacent nodes with the same label are combined to form clusters.

2.2 Web Mining

Web mining research can be classified into three categories: Web content mining, Web structure mining, and Web usage mining (Kosala & Blockeel, 2000). Web content mining refers to the discovery of useful information from Web contents, including text, images, audio, video, etc. Web content mining research includes resource discovery from the Web (e.g., Cho et al., 1998; Chakrabarti et al., 1999b), document categorization and clustering (e.g., Zamir & Etzioni, 1999; Kohonen et al., 2000), and information extraction from Web pages (e.g., Hurst, 2001). Web structure mining studies the model underlying the link structures of the Web. It usually involves the analysis of in-links and out-links information of a Web page, and has been used for search engine result ranking and other Web applications (e.g., Brin & Page, 1998; Kleinberg, 1998). Web usage mining focuses on using data mining techniques to analyze search logs or other activity logs to find interesting patterns. One of the main applications of Web usage mining is its use to learn user profiles (e.g., Armstrong et al., 1995; Wasfi et al., 1999).

There are several major challenges for Web mining research. First, most Web documents are in HTML (HyperText Markup Language) format and contain many markup tags, mainly used for formatting. While Web mining applications must parse HTML documents to deal with these markup tags, the tags also can provide additional information about the document. For example, a bold typeface markup (<b>) may indicate that a term is more important than other terms that appear in normal typeface. Such formatting cues have been widely used to determine the relevance of terms (Arasu, 2001).

Second, while traditional IR systems often contain structured and well-written documents (e.g., news articles, research papers, metadata), this is not the case on the Web. Web documents are much more diverse in terms of length, document structure, writing style, and many Web pages contain many grammatical and spelling errors. Web pages are also very diverse in terms of languages and domains; one can find almost any language and any topic on the Web. In addition, the Web has many different types of content, including text, images, audios, videos, and executables. There are numerous formats, such as HTML, XML, PDF, MS Word, mp3, wav, ra, rm, avi, just to name a few. Web applications have to deal with these different formats and retrieve the desired information.

Third, while most documents in traditional IR systems tend to remain static over time, Web pages are much more dynamic; they can be updated every day, every hour or even every minute. Some Web pages do not even have a static form; they are dynamically generated on request, with content varying according to the user and the time of the

request. Such dynamics make it much more difficult for retrieval systems such as search engines to keep an up-to-date search index of the Web.

Another characteristic of the Web, perhaps the most important one, is the hyperlink structure. Web pages are hyperlinked to each other, and it is through hyperlink that a Web page author "cites" other Web pages. Intuitively, the author of a Web page places a link to another Web page if he or she believes that it is contains a relevant topic or is of good quality (Kleinberg, 1998). Anchor text, the clickable text of an outgoing link in a Web page, also provides a good description of the target page because it represents how other people linking to the page actually describe it. Several studies have tried to make use of anchor text or the text nearby to predict the content of the target page (Amitay, 1998; Rennie & McCallum, 1999).

Lastly, the size of the Web is larger than traditional data sources or document collections by several orders of magnitude. The number of indexable Web pages has exceeded two billion, and has been estimated to be growing at a rate of roughly one million pages per day (Lawrence & Giles, 1999; Lyman & Varian, 2000). Collecting, indexing, and analyzing these documents presents a great challenge. Similarly, the population of Web users is much larger than that of traditional information systems. Collaboration among users can be more feasible because of the availability of a large user base, but it can also be more difficult because users are more diverse.

In this section, I review how machine learning techniques for traditional IR systems have been improved and adopted in Web mining applications, based on the characteristics of the Web. Significant work has been done both in academia and industry. However, because most commercial applications do not disclose any technical or algorithmic details, my review will mainly focus on academic research.

2.2.1 Web Content Mining

Web content mining is mainly based on research in information retrieval and text mining, such as information extraction, text classification and clustering, and information visualization. However, it also includes some new applications, e.g., Web resource discovery. Some important Web content mining techniques and applications are reviewed in this subsection. They are classified as text mining for Web documents, multilingual Web mining, Web visualization, and the Semantic Web. Web spiders or Web intelligent agents are often considered Web content mining, and will be reviewed in detail in Section 2.3.

2.2.1.1 Text Mining for Web Documents

As discussed earlier, text mining is often considered a sub-field of data mining and refers to the extraction of knowledge from text documents (Hearst, 1999; Chen, 2001). As the majority of documents on the Web are text documents, text mining for Web documents can be considered a sub-field of Web mining or, more specifically, Web content mining. Information extraction, text classification, and text clustering are some examples of text mining applications that have been applied to Web documents.

While information extraction techniques have been applied to plain text documents, extracting information from HTML Web pages can present a quite different problem. As described earlier, HTML documents contain many markup tags that may be useful to identify useful information. However, Web pages are also comparatively unstructured. Instead of a document consisting of paragraphs, a Web page can be a document composed of a sidebar with navigation links, tables with textual and numerical data, capitalized sentences, and repetitive words. The format of these structures is very diverse across the Web. If a system could parse and understand such structures, it would effectively acquire additional information for each piece of text. For example, a set of links with a heading "Link to my friends' homepages" may indicate a set of people's names and corresponding personal homepage links. The header row of a table can also provide additional information about the text in the table cells. On the other hand, if these tags are not processed correctly and simply stripped off, the document may become much noisier.

Chang and Lui (2001) used a PAT tree to construct automatically a set of rules for information extraction. The system, called IEPAD, reads an input Web page and looks for repetitive HTML markup patterns. After patterns not wanted have been filtered out, each pattern is used to form an extraction rule in regular expression. IEPAD has been tested in an experiment to extract search results from different search engines and achieved high retrieval rate and accuracy. Wang and Hu (2002) used decision tree and SVM to learn the patterns of table layouts in HTML documents. Layout features, content type features and word group features are combined and used as a document's features.

Experimental results show that both decision tree and SVM can detect tables in HTML documents with high accuracy. Borodogna and Pasi (2001) proposed a fuzzy indexing model which allows users to retrieve sections of structured documents such as HTML and XML. Doorenbos et al. (1997) also have applied machine learning in the ShopBot system to extract product information from Web pages. There are also some commercial applications that extract useful information from Web pages. For instance, FlipDog (http://www.flipdog.com/), developed by the Whizbang! Labs (http://www.inxight.com/whizbang/), crawls the Web to identify job openings on various employer Web sites. Lencom Software (http://www.lencom.com/) also developed several products that can extract email addresses and image information from the Web.

While information extraction analyzes individual Web pages, text classification and text clustering analyze a set of Web pages. Again, Web pages consist mostly of HTML documents and are often noisier and more unstructured than traditional documents such as news articles and academic abstracts. In some applications, the HTML tags are simply stripped from the Web documents and traditional algorithms are then applied to perform text classification and clustering. However, some useful characteristics of Web page design, which also differs from that for traditional IR applications, would be ignored. For example, Web pages are hyperlinked, co-referencing, and many pages contain common phrases such as "Home", "Click here" and "Contact us" which should not be included as a document's features. This creates a unique problem for performing text classification and clustering of Web documents, because the format of HTML documents and the structure of the Web provide additional information for analysis. For example, text from

neighboring documents has been used in an attempt to improve classification performance. However, experimental results show that this method does not improve performance because there are often too many neighbor terms and too many cross-linkages between different classes (Chakrabarti et al., 1998; Yang et al., 2002). Uses of other information of neighboring documents have been proposed. Examples of such information include the predicted category of neighbors (Chakrabarti et al., 1998; Oh et al., 2000), the anchor text pointing to a document (Furnkranz, 1999), or the outgoing links to all other documents (Joachims et al., 2001). It has been shown that using such additional information improves classification results.

Similarly, text clustering algorithms have been applied to Web applications. In the Grouper system, Zamir and Etzioni (1998; 1999) applied the Suffix-Tree Clustering algorithm described earlier to the search results of the HuskySearch system. He et al. (2002) use a combination of content, hyperlink structure, and co-citation analysis in Web document clustering. Two Web pages are considered similar if they have similar content, they point to a similar set of pages, or many other pages point to both of them.

The large number of documents available has made the Web an excellent resource for linguistic studies. The digital library project groups of the University of California, Berkeley and Stanford University performed an analysis on 88 million Web pages and calculated the document frequency of the 113 million terms found in those pages (UC Berkeley, 2002). Roussinov and Zhao (2003) use the Web as a resource for finding

phrases with high co-occurrences. Another example is the Strand system (Resnik, 1998), which attempts to identify bilingual parallel corpora on the Web.

2.2.1.2 Multilingual Web Mining

The number of non-English documents on the Web keeps increasing — more than 30% of Web pages are in a language other than English. In order to extract non-English knowledge from the Web, Web mining systems have to deal with issues in language-specific text processing. One might think that this should not be a problem because the base algorithms behind most machine learning systems are language-independent. Most algorithms, e.g., text classification and clustering, need only to take a set of features (a vector of keywords) for the learning process. However, the algorithms usually depend on some phrase segmentation and extraction programs to generate a set of features or keywords to represent Web documents. Many existing extraction programs, especially those that employ a linguistic approach (e.g., Church, 1988), are language-dependent and work only with English texts. In order to perform analysis on non-English documents, Web mining systems must use the corresponding phrase extraction program for each language. Other learning algorithms such as information extraction and entity extraction also have to be tailored for different languages.

Some segmentation and extraction programs are language-independent. These programs usually employ a statistical or a machine learning approach. For example, the mutual-information-based PAT-Tree algorithm is a language-independent technique for key phrase extraction and has been tested on Chinese documents (Chien, 1997; Ong & Chen,

1999). Similarly, Church and Yamamoto (2001) use suffix arrays to perform phrase extraction. Because these programs do not rely on specific linguistic rules, they can be easily modified to work with different languages.

2.2.1.3 Web Visualization

Because it is often difficult to extract useful content from the Web, visualization tools have been used to help users maintain a "big picture" of a set of retrieval results from search engines, particular Web sites, a subset of the Web, or even the whole Web. Various techniques have been developed in the past decade. For example, many systems visualize the Web as a tree structure based on the outgoing links of a set of starting nodes (e.g., Huang et al., 1998). The most well known example of using the tree-metaphor for Web browsing is the hyperbolic tree developed by Xerox PARC (Lamping & Rao, 1996). The hyperbolic tree employs the "focus+context" technique to show Web sites as a tree structure using a hyperbolic view. Users can focus on the document they are looking at and still keep an overview picture of the context at the same time. Map is another metaphor widely used for Web visualization. The ET-Map provides a visualization of the manually catalogued Entertainment hierarchy of Yahoo as a 2-dimensional map (Chen et al., 1996). 110,000 Web pages are clustered into labeled regions on the map based on self-organizing map (SOM), in which larger regions represent more important topics, and regions close to each other represent topics that are similar (Lin et al., 2000). The WEBSOM system also utilizes the SOM algorithm to cluster over a million Usenet newsgroup documents (Kohonen, 1995; Lagus et al., 1999). Other examples of Web visualization include WebQuery, which uses a bullseye's view to visualize Web search

results based on link structure (Carrière & Kazman, 1997), WebPath, which visualizes a user's trail as he or she browses the Web (Frécon and Smith, 1998), and 3-dimensional models such as Natto View (Shiozawa & Matsushita, 1997) and Narcissus (Hendley et al., 1995). Dodge and Kitchin (2001) provide a comprehensive review of the maps of the cyberspace, called cybermaps, generated since the inception of the Internet.

In these visualization systems, machine learning techniques are often used to determine how Web pages should be placed in the 2- or 3-dimensional space. One example is the SOM algorithm described earlier (Chen et al., 1996). Web pages are represented as vectors of keywords and used to train the network that contains a two-dimensional grid of output nodes. The distance between the input and each output node is then computed and the node with the minimum distance is selected. After the network is trained through repeated presentation of all inputs the documents are submitted to the trained network and each region is labeled by the phrase which is the key concept that most represents the cluster of documents in that region. Multidimensional scaling (MDS) is another method that can be used to position documents on a map. It tries to map high dimensionality (e.g., document vectors) to low dimensionality (usually 2-dimensional) by solving a minimization problem (Cox & Cox, 1994). It has been tested with document mapping and the results are encouraging (McQuaid et al., 1999).

2.2.1.4 The Semantic Web

A recent significant extension of the Web is the Semantic Web (Berners-Lee et al., 2001), which tries to add metadata to describe data and information on the Web, based on such

standards as RDF (Resource Description Framework) and XML (Extensible Markup Language). The idea is that Web documents will no longer be unstructured text; they will be labeled with meaning that can be understood by computers. Machine learning can play three important roles in the Semantic Web. First, machine learning can be used to automatically create the markup or metadata for existing unstructured textual documents on the Web. It is very difficult and time-consuming for Web page authors to generate Web pages manually according to the Semantic Web representation. To address this problem, information extraction techniques, such as entity extraction, can be applied to automate or semi-automate tasks such as identifying entities in Web pages and generating the corresponding XML tags. Second, machine learning techniques can be used to create, merge, update, and maintain ontologies. Ontology, the explicit representation of knowledge combined with domain theories, is one of the key pieces in the Semantic Web (Berners-Lee et al., 2001; Fensel & Musen, 2001). Maedche & Staab (2001) propose a framework for knowledge acquisition using machine learning. In that framework, machine learning techniques, such as association rule mining or clustering, are used to extract knowledge from Web documents in order to create new ontologies or improve existing ones. Third, machine learning can understand and perform reasoning on the metadata provided by the Semantic Web in order to extract knowledge from the Web more effectively. The documents in the Semantic Web are much more precise, more structured, and less "noisy" than the general, syntactic Web. The Semantic Web also provides context and background information for analyzing Web pages. It is believed that

the Semantic Web can greatly improve the performance of Web mining systems (Berendt et al., 2002).

2.2.2 Web Structure Mining

In recent years, Web link structure has been widely used to infer important information about Web pages. Web structure mining has been largely influenced by research in social network analysis and citation analysis (bibliometrics). Citations (linkages) among Web pages are usually indicators of high relevance or good quality. The term *in-links* is used to indicate the hyperlinks pointing to a page and the term *out-links* to indicate the hyperlinks found in a page. Usually, the larger the number of in-links, the better a page is considered to be. The rationale is that a page referenced by more people is likely to be more important than a page that is seldom referenced. As in citation analysis, an often-cited article is considered better than one never cited. In addition, it is reasonable to give a link from an authoritative source (such as *Yahoo!*) a higher weight than a link from an unimportant personal homepage.

By analyzing the pages containing a URL, it is also possible to obtain the anchor text that describes it. Anchor text represents how other Web page authors annotate a page and can be useful in predicting the content of the target page. Several algorithms have been developed to address this issue.

Among various Web structure mining algorithms, PageRank and HITS are the two most widely used. The PageRank algorithm is computed by weighting each in-link to a page

proportionally to the quality of the page containing the in-link (Brin & Page, 1998). The qualities of these referring pages also are determined by PageRank. Thus, the PageRank of a page p is calculated recursively as follows:

$$PageRank(p) = (1-d) + d \times \sum_{\substack{all\ q\ linking \\ to\ p}} \left(\frac{PageRank(q)}{c(q)} \right)$$

where *d is a damping factor between 0 and 1,*

c(q) is the number of out-going links in a page q.

A Web page has a high PageRank score if it is linked from many other pages, and the scores will be even higher if these referring pages are also good pages (pages that have high PageRank scores). It is also interesting to note that the PageRank algorithm follows a random walk model — the PageRank of a page is proportional to the probability that a random surfer clicking on random links will arrive at that page.

Kleinberg (1998) proposed the HITS (Hyperlink-Induced Topic Search) algorithm, which is similar to PageRank. In the HITS algorithm, authority pages are defined as high-quality pages related to a particular topic or search query. Hub pages are those that are not necessarily authorities themselves but provide pointers to other authority pages. A page to which many others point should be a good authority, and a page that points to many others should be a good hub. Based on this intuition, two scores are calculated in

the HITS algorithm for each Web page: an authority score and a hub score, which are calculated as follows:

$$AuthorityScore(p) = \sum_{\substack{all\ q\ linking \\ to\ p}} (HubScore(q))$$

$$HubScore(p) = \sum_{\substack{all\ r\ linking \\ from\ p}} (AuthorityScore(r))$$

In other words, a page with a high authority score is one pointed to by many good hubs, and a page with a high hub score is one that points to many good authorities.

Following the success of the PageRank and HITS algorithms, other similar algorithms also have been proposed. Examples include the SALSA algorithm (Lempel & Moran, 2001) and the PHITS algorithm (Cohn & Chang, 2000). Web structure mining techniques are often used to enhance the performance of Web applications. For instance, PageRank has been shown to be very effective for ranking search results in the commercial search engine Google (http://www.google.com/) (Brin & Page, 1998). It also has been used as a measure to guide search engine spiders, where URLs with higher PageRank are visited earlier (Cho et al., 1998). The HITS algorithm also has been used in various Web applications. One example is the Clever search engine (Chakrabarti et al., 1999a), which achieves a higher user evaluation than the manually compiled directory of Yahoo. Bharat and Henzinger (1998) have added several extensions to the basic HITS algorithm, such as modifying how much a node influences its neighbors based on a relevance score. One of

the major drawbacks shared by most Web structure analysis algorithms is their high computational requirement, because the scores often have to be calculated iteratively (Kleinberg, 1998; Haveliwala, 1999).

Another application of Web structure mining is to understand the structure of the Web as a whole. Broder et al. (2000) performed analysis on the graph structure of a collection of 200 million Web pages and 1.5 billion links. Their results suggest that the core of the Web is a strongly connected component and that the Web's graph structure is shaped like a bowtie. The strongly connected component (SCC) comprises around 28% of the Web. Another group that consists of 21% of Web pages are called IN, in which every Web page contains a direct path to the SCC. Another 21% of Web pages are in the group OUT. For every page in OUT there is a direct path from SCC linking to it. 22% of Web pages are in the group TENDRILS, which consists of pages hanging off from IN and OUT but without direct path to SCC. The remaining Web pages, accounting for around 8% of the Web, are isolated, disconnected components that are not connected to the other 4 groups. Such analysis and similar findings are very useful and provide insights to other Web-related research and applications.

2.2.3 Web Usage Mining

Web servers, Web proxies, and client applications can quite easily capture data about the usage of the Web. Web server log contains information about every visit to the pages hosted on the server. Some of the useful information includes what files have been requested from the server, when they were requested, the Internet Protocol (IP) address of

the request, the error code, the number of bytes sent to user, and the type of browser used. Web servers can also capture referrer logs, which show on what page a visitor is located when making the next request. Client-side applications, such as Web browsers or personal agents, also can be designed to monitor and record a user's actions. By performing analysis on Web usage data (sometimes referred to as *clickstream analysis*), Web mining systems can discover useful knowledge about a system's usage characteristics and the users' interests. Such knowledge has various applications, such as personalization and collaboration in Web-based systems, marketing, Web site design, Web site evaluation, and decision support (e.g., Chen & Cooper, 2001; Marchionini, 2002).

2.2.3.1 Pattern Discovery and Analysis

One of the major goals of Web usage mining is to reveal interesting trends and patterns from Web usage data. Such patterns and statistics can often provide important knowledge about the customers of a company or the users of a system. Srivastava et al. (2000) provided a framework for Web usage mining. The framework consists of 3 major steps, namely preprocessing, pattern discovery, and pattern analysis. As in other data mining applications, preprocessing involves data cleansing. However, one of the major challenges faced by Web usage mining applications is that Web server log data are anonymous, making it difficult to identify users and user sessions from the data. Techniques like Web cookies and user registration have been used in some applications, but each method has its own shortcomings (Pitkow, 1997). In pattern discovery and analysis, generic machine learning and data mining techniques, such as association rule

mining, classification, and clustering, often can be applied. For instance, Yan et al. (1996) performed clustering on Web log data to identify users who have accessed similar Web pages.

Web usage mining has been used for various purposes. For example, Buchner and Mulvenna (1998) proposed a knowledge discovery process for mining marketing intelligence information from Web data. Data such as Web traffic patterns also can be extracted from Web usage logs in order to improve the performance of a Web site (Cohen et al., 1998). Many commercial products have been developed to support analysis and mining of Web site usage and Web log data. Examples of these applications include WebTrends developed by NetIQ (http://www.netiq.com/webtrends/), WebAnalyst by Megaputer (http://www.megaputer.com/products/wa/), NetTracker by Sane Solutions (http://www.sane.com/products/NetTracker/), and NetGenesis by CustomerCentric (http://www.customercentricsolutions.com/content/solutions/ent_Web_analytics.cfm). While most Web usage analysis applications focus on single Web sites, the advertising company DoubleClick (http://www.doubleclick.com/), selling and administrating two billions of online advertisements per day, collects gigabytes of clickstream data across different Web sites.

Search engine transaction logs also provide valuable knowledge about user behavior on Web searching. Various analyses have been performed on the transaction logs of the Excite search engine (http://www.excite.com/) (Jansen et al., 2000; Spink & Xu, 2000; Spink et al., 2001). Silverstein et al. (1999) also conducted a study of 153 million unique

search queries collected from the AltaVista search engine (http://www.altavista.com/). Some of the interesting findings from these analyses include the set of most popular words used by the public in Web search queries, the average length of a search query, the use of Boolean operators in queries, and the average number of result pages viewed by users. Such information is very useful to researchers trying to reach a better understanding of users' Web searching and information seeking behavior and hoping to improve the design of Web search systems.

2.2.3.2 Personalization and Collaboration

In addition to the research in Web spiders discussed earlier, other agent techniques also have been used in Web applications. Many Web applications aim to provide personalized information and services to users. Web usage data provide an excellent way to learn about users' interest (Srivastava et al., 2000). WebWatcher (Armstrong et al., 1995) and Letizia (Lieberman, 1995) are two early examples. In WebWatcher, a user needs to specify the information needs, and the traversal links of the user will be captured. These data are then used to generate recommendations for the user using some simple learning algorithms. The Letizia system tries to learn the user's interests on-the-fly, using some heuristics based on a user's actions, such as following a link or saving a document. The system explores neighborhood Web pages of potential interest using a best-first search algorithm.

The exponential growth of the Web has greatly increased the amount of Web usage data in server logs. Web logs usually consist of the usage data of more than one user. Web

usage mining can help identify users who have accessed similar Web pages. The patterns that emerge can be very useful in collaborative Web searching and collaborative filtering. In the Fab system, Web pages are recommended to users based on the Web pages visited by other users having similar interests (Balabanovic & Shoham, 1997). Similarly, Amazon.com (http://www.amazon.com/) uses collaborative filtering to recommend books to potential customers based on the preferences of other customers having similar interests or purchasing histories. Huang et al. (2002) used Hopfield Net to model user interests and product profiles in an online bookstore in Taiwan. Spreading activation and association rule mining are used to search the network in order to provide recommendations to users.

2.3 Searching the Web

General-purpose Web search engines and Web directories, such as Google (http://www.google.com/), AltaVista (http://www.altavista.com/), Lycos (http://www.lycos.com/), Infoseek (http://infoseek.go.com/), FAST (http://www.alltheweb.com/), and Yahoo (http://www.yahoo.com), provide the most popular way for users to look for information on the Web. Many users begin their Web activities by submitting a query to a search engine. A simple search engine architecture is shown in Figure 2.1. *Spiders*, also referred to as Web robots, crawlers, worms, or wanderers, are programs behind a search engine that retrieve Web pages by recursively following URL links in pages using standard HTTP protocols (Cheong, 1996). After these pages have been collected, they are processed by an *indexer* to build the underlying index of the search engine. An indexer tokenizes each page into words and records their

occurrences. The resulting index is stored into a database. When a user performs a search through a Web interface, a *query engine* retrieves the search results from the index database and a *ranking engine* ranks and presents the results to the user (Arasu, 2001).

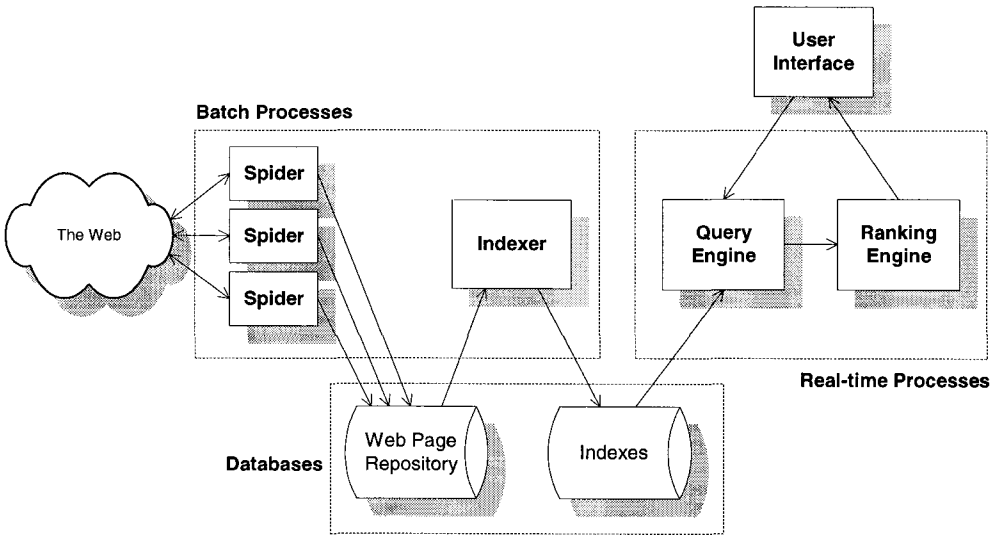

Figure 2.1: Typical search engine architecture

As the size of the Web is growing exponentially, however, it is more difficult for search engines to keep an up-to-date and comprehensive search index, resulting in low precision and low recall rates. Users often find it difficult to search for useful and high-quality information on the Web using general-purpose search engines, especially when they are searching for specific information on a given topic. Alternative approaches include personalized search agents and specialized search engines. In this section, research in these two areas are reviewed.

2.3.1 Web Agents

Client-side Web search tools, often known as spiders, crawlers, Web robots, worms, or wanderers, have been developed to help individual users search for personalized information on the Web. Personalized information can be defined as any kind of "tailored" information provided to an individual (Mulvenna et al., 2000). These tools are often considered as software agents known as Web agents or Internet agents (Cheong, 1996). As Web agents usually run on the client machine, more CPU time and memory can be allocated to the search process and more functionalities are possible. They also allow users to have more control and options during the search process.

tueMosaic is an early example of personal Web spiders (DeBra & Post, 1994). Using tueMosaic, users can enter keywords, specify the depth and width of search for links contained in the current Web pages displayed, and request the spider agent to fetch pages connected to the current Web page. tueMosaic uses the "fish search" algorithm, a modified best first search method. Since its introduction, many more powerful personal spiders have been developed.

Some spiders have been designed to provide additional functionalities. The TkWWW robot is a program integrated in the TkWWW browser (Spetka, 1994). It can be dispatched from the browser and search Web neighborhoods to find relevant pages and returns a list of links that look promising. SPHINX, a spider written in Java, allows users to perform breadth-first search and view the search results as a 2-dimensional graph (Miller & Bharat, 1998).

In other studies, spiders use machine learning techniques or other advanced algorithms during the search process. The Itsy Bitsy Spider searches the Web using a best-first search and a genetic algorithm approach (Chen et al., 1998a; Chen et al., 1998b). Each URL is modeled as an individual in the initial population. Crossover is defined as extracting the URLs that are pointed to by multiple starting URLs. Mutation is modeled by retrieving random URLs from Yahoo. Because genetic algorithm is an optimization process, it is well-suited to finding the best Web pages according to particular criteria. Webnaut is a later spider that also applies genetic algorithm (Zacharis & Panayiotopoulos, 2001). Other advanced search algorithms also have been used in personal spiders. Yang et al. (2000) apply hybrid simulated annealing in a personal spider application. Focused Crawler locates Web pages relevant to a pre-defined set of topics based on example pages provided by the user. In addition, it also analyzes the link structures among the Web pages collected (Chakrabarti et al., 1999b). Context Focused Crawler uses a Naïve Bayesian classifier to guide the search process (Diligenti et al., 2000). Relevance feedback also has been applied in spiders (Balabanovic & Shoham, 1995; Vrettos & Stafylopoatis, 2001).

Many commercial Web spiders are also available. WebRipper, WebMiner, and Teleport are examples of software that helps users download specified files from given Web sites. Excalibur's RetrievalWare and Internet Spider (http://www.excalib.com/) collect, monitor and index information from text documents on the Web as well as graphic files. Autonomy's products (http://www.autonomy.com/) support a wide range of information collection and analysis tasks, which include automatic searching and monitoring

information sources in the Internet and corporate Intranets, and classifying documents into categories predefined by users or domain experts. Verity's knowledge management products (http://www.verity.com/), such as Agent Server, Information Server and Intelligent Classifier, also perform similar tasks in an integrated manner.

Another important category of personal spiders is composed of meta spiders, programs that connect to different search engines to retrieve search results. Lawrence and Giles showed that the best search engine covered only about 16% of Web sites in 1999 (Lawrence & Giles, 1999). Therefore, combining results from different search engines achieves more comprehensive coverage. MetaCrawler was the first meta spider reported (Selberg & Etzioni, 1995; Selberg & Etzioni, 1997). It provides a single interface allowing users to search simultaneously from six different search engines, namely Lycos, WebCrawler, Infoseek, Galaxy, Open Text, and Yahoo. MetaCrawler, with much more search options available, is still in service now.

Following the success of MetaCrawler, many meta search services have been developed. Profusion allows users to choose among a list of six search engines that can be queried (Gauch et al., 1996). 37.com (http://www.37.com/) connects with 37 different search engines. Dogpile (http://www.dogpile.com/) provides meta-search service for news, Usenet, white pages, yellow pages, images, etc., in addition to Web pages. SavvySearch connects with a large number of general and topic-specific search engines and selects those likely to return useful results based on past performance (Howe & Dreilinger, 1997). Similarly, Yu et al. (1999) use link analysis to decide which are the appropriate search

engines to be used, and Chen and Soo (2001) use domain ontology in meta search agents to assist users in query formulation. Blue Squirrel's WebSeeker (http://www.bluesquirrel.com/) and Copernic's software (http://www.copernic.com/) connect with other search engines, monitor Web pages for any changes, and schedule automatic search. Grouper, an extension of MetaCrawler, clusters the search results from various search engines based on a suffix-tree clustering algorithm (Zamir & Etzioni, 1999).

In addition to getting a list of URLs and summaries returned by other search engines, some meta spiders download and analyze all the documents in the result set. Inquirus, also known as the NECI meta search engine, downloads actual result pages and generates a new summary of each page based on the search query. Pages which are no longer available (dead links) or which no longer contain the search terms are filtered from the search results (Lawrence & Giles, 1998a; Lawrence & Giles, 1998b). Another similar system is TetraFusion, which performs hierarchical and graph-based classification on the result set (Crimmins et al, 1999). Focused Crawler (Chakrabarti et al., 1999b) and Fetuccino (Ben-Shaul et al., 1999) use search results from popular search engines and expand the result set based on these URLs. Dwork et al. (2001) proposed the use of Markov chain to combine search results from different search engines and achieved promising experimental results.

Recently, search spiders also have been developed on the basis of peer-to-peer (P2P) technology, following the success of other P2P applications such as Napster. JXTA

Search (search.jxta.org), formerly known as InfraSearch, uses Gnutella as its backbone and links to other computers when a request is received from a user. The request is passed to neighbor computers to see if any of them can fulfill the request. Each computer can have its own strategy on how to respond to the request (Waterhouse et al., 2002).

2.3.2 Specialized Search Engines

Because of the large number of documents on the Web, it has become more difficult for general-purpose search engines to maintain up-to-date, comprehensive search indexes. Many vertical search engines, or domain-specific search engines, have been developed to address the problem by keeping search indexes only in particular domains. Examples include LawCrawler (http://www.lawcrawler.com/), BuildingOnline (http://www.buildingonline.com/), and SciSeek (http://www.sciseek.com/). Vertical search engines allow users to perform searches in particular domains and they usually provide customized features. However, it is also difficult to build these search engines as it is difficult to locate relevant and high-quality Web pages. Finding a way to collect from two billion Web pages a subset of high-quality ones relevant to a desired domain becomes a great challenge. During the collection of Web pages, an intelligent search engine spider (search agent) should "predict" whether a URL pointed to a relevant Web page, before actually fetching the page. In addition, the spider should first visit pages with higher probability of having relevant content to improve efficiency, and it should determine the quality and reputation of pages to avoid irrelevant or bad quality ones.

2.4 Research Formulation and Design

As the Web continues to grow, general-purpose search engines can no longer satisfy users' information needs. Alternative approaches need to be explored to facilitate more efficient and effective searching and analysis capabilities. To address these issues, three research questions have been posed:

- How can search agents improve user performance in performing personalized search and analysis on the Web?
- How can search agents facilitate knowledge sharing for Web search and analysis?
- How can existing machine learning and artificial intelligence techniques be used to collect domain-specific knowledge effectively and efficiently from the Web?

The system development research process described in Nunamaker et al. (1991) has been adopted to study these research questions. Following this methodology, several prototype systems were developed and evaluated to investigate the research questions by studying how users interact with these systems and the techniques incorporated. The following five Chapters will present several system prototypes that have been designed, implemented, and evaluated to address different issues related to Web search agents and specialized search engines. The structure of the dissertation is shown in Figure 2.1. In Chapters 3, 4, and 5, I present my work in personalized Web search agents that study the first two research questions. In Chapters 6 and 7, I describe my work in developing specialized search engines to investigate the third research question.

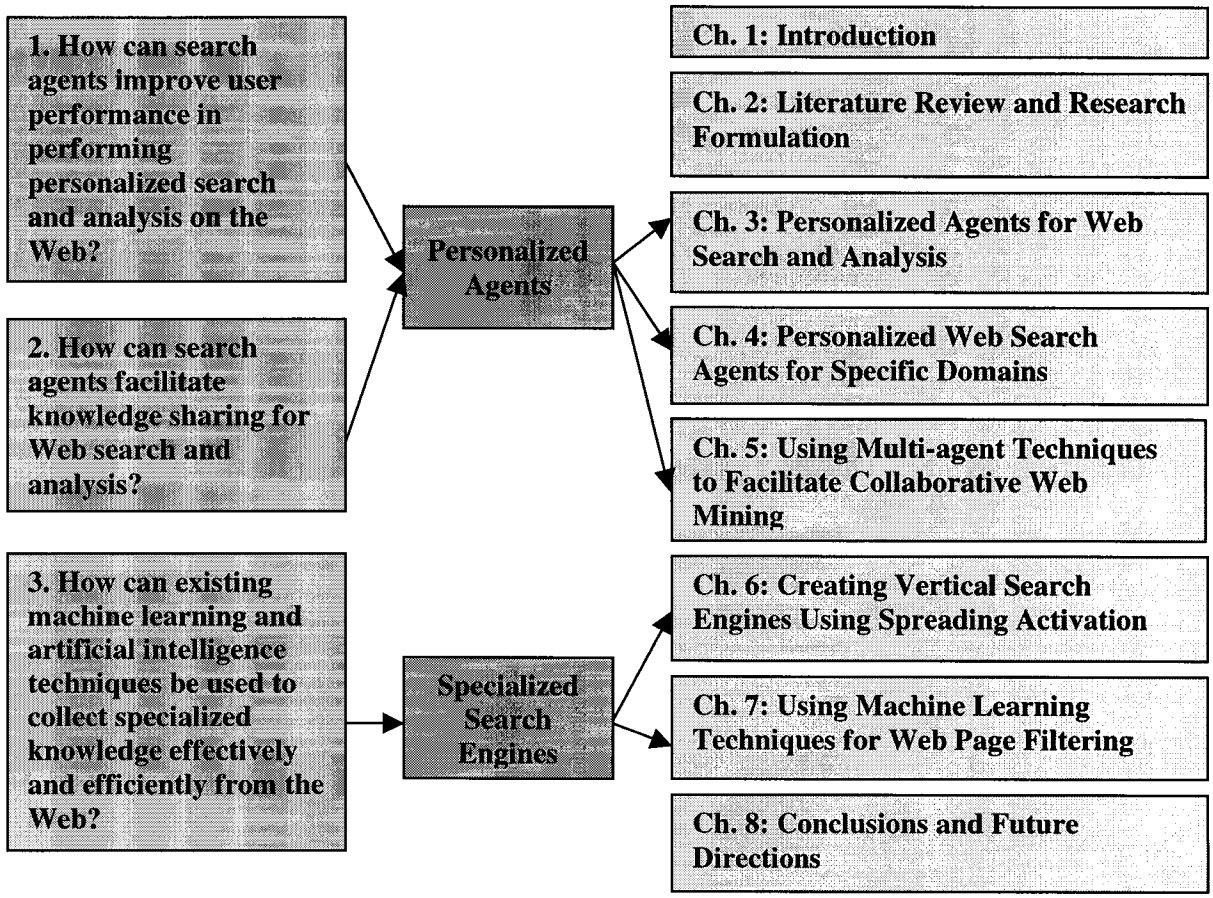

Figure 2.2: Dissertation structure

CHAPTER 3: PERSONALIZED AGENTS FOR WEB SEARCH AND ANALYSIS

3.1 Background

As discussed in Chapters 1 and 2, the Web has become the largest source of information that has ever existed. However, it also has become increasingly difficult to search for useful information on it, due to its dynamic, unstructured nature and its fast growth rate. Although development of Web search and analysis tools such as search engines has alleviated the problem to a great extent, exponential growth of the Web is making it impossible to collect and index all the Web pages and refresh the index frequently enough to keep it up-to-date. Most search engines present search results to users that are incomplete and outdated, usually leaving users confused and frustrated.

A second problem that Internet users encounter is the difficulty in searching information on a particular Web site, e.g., looking for information related to a certain topic in the Web site http://www.phoenix.com/. Among the popular commercial search engines, only a few offer the search option to limit a search session to a specified Web site. Because most search engines only index a certain portion of each Web site, the recall rate of these searches is very low, and sometimes even no documents are returned. Although most large Web sites nowadays have their built-in internal search engines, these engines index the information based on different schemes and policies and users may have difficulty in uncovering useful information. In addition, most of the Web sites on the Internet are small sites that do not have an internal search feature.

A third problem is the poor retrieval performance when only a single search engine is used. It has been estimated that none of the search engines available indexes more than 16% of the total Web that could be indexed (Lawrence & Giles, 1999). Even worse, each search engine maintains its own searching and ranking algorithm as well as query formation and freshness standard. Unless the different features of each search engine are known, searches will be inefficient and ineffective. From the user's point of view, dealing with an array of different interfaces and understanding the idiosyncrasies of each search engine is too burdensome. The development of meta-search engines has alleviated this problem. However, how the different results are combined and presented to the user greatly affects the effectiveness of these tools.

In addition, given the huge number of daily hits, most search engines are not able to provide enough computational power to satisfy each user's information need. Analysis of search results, such as verifying that the Web pages retrieved still exist or clustering of Web pages into different categories, are not available in most search engines. Search results are usually presented in a ranked list fashion; users cannot get a whole picture of what the Web pages are about until they click on every page and read the contents. This can be time-consuming and frustrating in a dynamic, fast-changing electronic information environment.

In order to alleviate the above problems, I propose a personalized and integrated approach to Web search. In this Chapter, I present a client-side Web search tool that applies various artificial intelligence techniques. I suggest that a search tool that is more

customizable would help users locate useful information on the Web more effectively. The client-based architecture also allows for greater computation power and resources to provide better searching and analysis performance. Two experiments were conducted to evaluate the performance of different prototypes built according to this architecture.

3.2 Related Work

In recent years, many client-side Web spiders have been developed. Because the software runs on the client machine, more CPU time and memory can be allocated to the search process and more functionalities are possible. Also, these tools allow users to have more control and personalization options during the search process. For example, Blue Squirrel's WebSeeker (http://www.bluesquirrel.com/) and Copernic 2000 (http://www.copernic.com/) connect with different search engines, monitor Web pages for changes, and schedule automatic search. Focused Crawler (Chakrabarti et al., 1999b) locates Web pages relevant to a pre-defined set of topics based on example pages provided by the user. In addition, it also analyzes the link structures among the Web pages collected.

3.2.1 Monitoring and Filtering

Because of the fast changing nature of the Internet, different tools have been developed to monitor Web sites for changes and filter out unwanted information. Push Technology is one of the emerging technologies in this area. The user first needs to specify some areas of interest. The tool will then automatically push related information to the user. Ewatch (http://www.ewatch.com/) is one such example. It monitors information not only from

Web pages but also from Internet Usenet groups, electronic mailing lists, discussion areas and bulletin boards to look for changes and alert the user.

Another popular technique used for monitoring and filtering employs a software agent, or intelligent agent (Maes, 1994). Personalized agents can monitor Web sites and filter information according to particular user needs. Machine learning algorithms, such as an artificial neural network, are usually implemented in agents to learn the user's preferences.

3.2.2 Indexing and Categorization

There have been many studies in textual information analysis of information retrieval and natural language processing. In order to retrieve documents based on given concepts, the documents have to be indexed. Automatic indexing algorithms have been used widely to extract key concepts from textual data. It having been shown that automatic indexing is as effective as human indexing (Salton, 1986), many proven techniques have been developed. Linguistics approaches such as noun phrasing also have been applied to perform indexing for phrases rather than just words (Tolle & Chen, 2000). These techniques are useful in extracting meaningful terms from text documents not only for document retrieval but also for further analysis.

Another type of analysis tool is categorization. These tools allow a user to classify documents into different categories. Some categorization tools facilitate the human categorization process by simply providing a user-friendly interface. Tools that are more

powerful categorize documents automatically, allowing users to quickly identify the key topics involved in a large collection of documents (e.g., Hearst & Pedersen, 1996; Rasmussen, 1992; Zamir & Etzioni, 1999).

In document clustering, there are in general two approaches. In the first approach, documents are categorized based on individual document attributes. An attribute might be the query term's frequency in each document (Hearst, 1995; Veerasamy & Belkin; 1996). NorthernLight, a commercial search engine, is another example of this approach. The retrieved documents are organized based on the size, source, topic or author of each document. Other examples include Envision (Fox et al., 1993) and GRIDL (Shneiderman et al., 2000).

In the second approach, documents are classified based on inter-document similarities. This approach usually includes some kind of machine learning algorithms. For example, the Self-Organizing Map (SOM) approach classifies documents into different categories which are defined during the process, using neural network algorithm (Kohonen, 1995). Based on this algorithm, the SOM technique automatically categorizes documents into different regions based on the similarity of the documents. It produces a data map consisting of different regions, where each region contains similar documents. Regions that are similar are located close to each other. Several systems utilizing this technique have been built (Lin et al., 1991; Chen et al., 1996; Kohonen, 1997; 2000).

3.3 Proposed Approaches

Two different prototypes based on the proposed architecture have been built. Competitive Intelligence Spider, or CI Spider, collects Web pages on a real-time basis from Web sites specified by the user and performs indexing and categorization analysis on them, to provide the user with a comprehensive view of the Web sites of interest (Chau et al., 2001b; Chen et al., 2002). The second tool, Meta Spider, has similar functionalities as the CI Spider, but instead of performing breadth-first search on a particular Web site, connects to different search engines on the Internet and integrates the results (Chau et al., 2001b; Chen et al., 2001). The architecture of CI Spider and Meta Spider is shown in Figure 3.1. There are 4 main components, namely (1) User Interface, (2) Internet Spiders, (3) Noun Phraser, and (4) Self-Organizing Map (SOM). These components work together as a unit to perform Web search and analysis. Shneiderman (1997) developed a four-phase framework for searching on the Web that allows better design and analysis of search systems. The four phases are (1) *formulation* (expressing the search), (2) initiating *action* (launching the search), (3) review of *results* (reading messages and outcomes) and (4) *refinement* (formulating the next step). The proposed tools are designed to assist users in the first three phases. Sample user sessions with CI Spider and Meta Spider are shown in Figures 3.2 and 3.3 respectively.

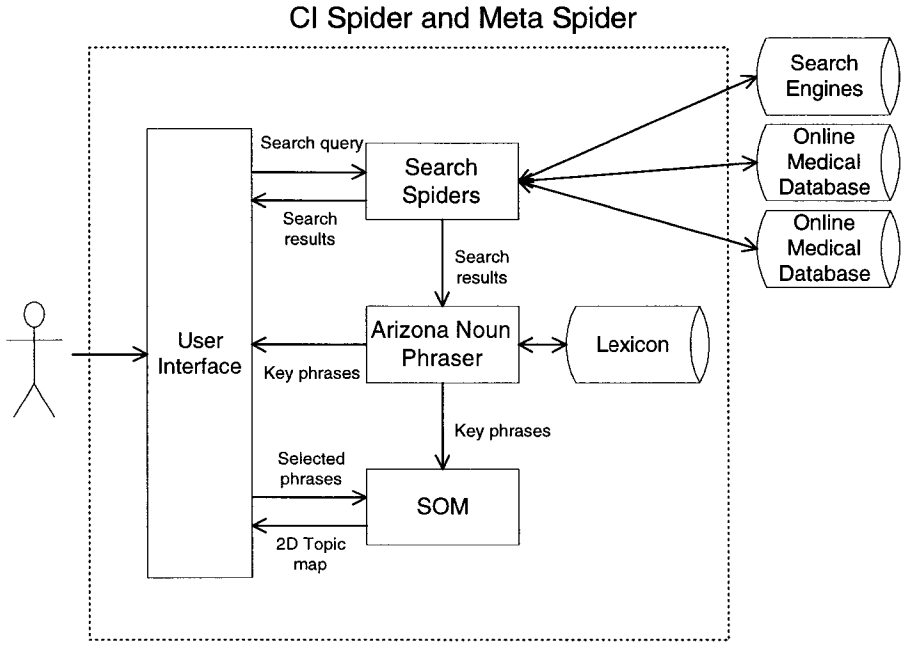

Figure 3.1: System architecture of CI Spider and Meta Spider

3.3.1 Internet Spiders

In CI Spider, the Internet Spiders are Java spiders that start from the URLs specified by the user and follow the outgoing links to search for the given keywords, until the number of Web pages collected reaches a user-specified target. The spiders run in multi-thread such that the fetching process will not be affected by slow server response time. Robots exclusion protocol is also implemented such that the spiders will not access sites where the Web master has placed a text file in a host or a meta-tag in a Web page, indicating that robots are not welcome to these sites.

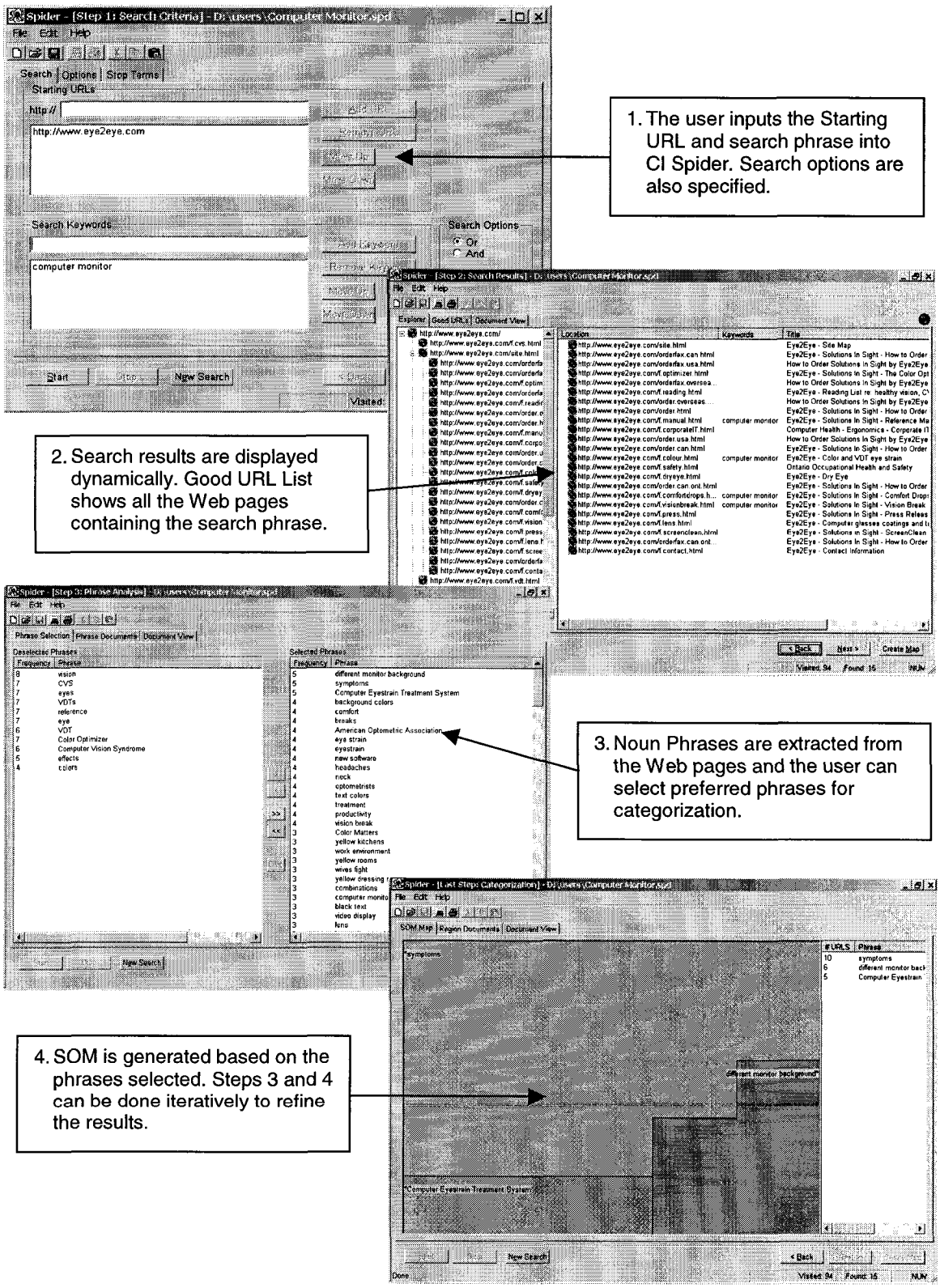

Figure 3.2: Example of a user session with CI Spider

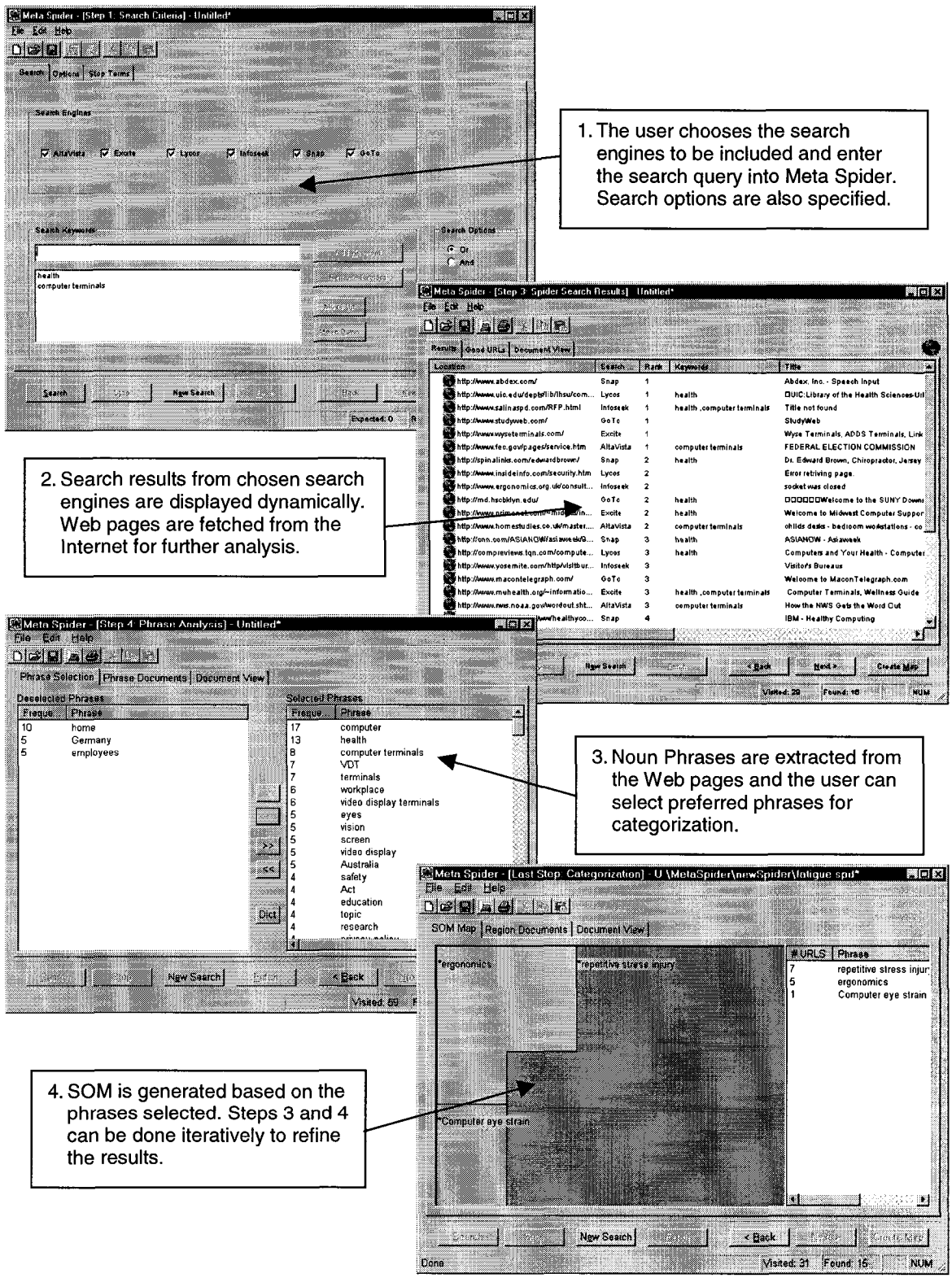

Figure 3.3: Example of a user session with Meta Spider

In the case of Meta Spider, the Internet Spiders first send the search queries to the search engines chosen. After the results are obtained, the Internet Spiders attempt to fetch every result page. Deadlinks and pages which do not contain the search keyword are discarded.

Whenever a page is collected during the search, the link to that page is displayed dynamically. The user can click on any link displayed and read its full content without having to wait for the whole search to be completed. The user can also switch to the Good URL List to browse only the pages that contain the search keyword. When the number of Web pages collected meets the amount specified by the user, the spiders will stop and the results will be sent to the Noun Phraser for analysis.

3.3.2 Noun Phraser

The Arizona Noun Phraser developed at the University of Arizona is the indexing tool used to index the key phrases that appear in each document collected from the Internet by the Internet Spiders. It extracts all the noun phrases from each document based on part-of-speech tagging and linguistic rules (Tolle & Chen, 2000). The Arizona Noun Phraser has three components. The tokenizer takes Web pages as text input and creates output that conforms to the UPenn Treebank word tokenization rules by separating all punctuation and symbols from text without interfering with textual content. The tagger module assigns a part-of-speech to every word in the document. The last module, called the phrase generation module, converts the words and associated part-of-speech tags into noun phrases by matching tag patterns to a noun phrase pattern given by linguistic rules. The frequency of every phrase is recorded and sent to the User Interface. The user can

view the document frequency of each phrase and link to the documents containing that phrase. After all documents are indexed, the data are aggregated and sent to the Self-Organizing Map for categorization.

3.3.3 Self-Organizing Map (SOM)

In order to give users an overview of the set of documents collected, the Kohonen SOM employs an artificial neural network algorithm to automatically cluster the Web pages collected into different regions on a 2-dimensional map (Kohonen, 1995; Chen et al., 1996). Each document is represented as an input vector of keywords and a two-dimensional grid of output nodes is created. After the network is trained, the documents are submitted to the network and clustered into different regions. Each region is labeled by the phrase which is the key concept that most accurately represents the cluster of documents in that region. More important concepts occupy larger regions, and similar concepts are grouped in a neighborhood (Lin et al., 2000). The map is displayed through the User Interface and the user can view the documents in each region by clicking on it.

3.3.4 Personalization Features

Because both CI Spider and Meta Spider have been designed for personalized Web search and analysis, a user has been given more control during the search process.

In the Options Panel, the user can specify how the search is to be performed. This is similar to the "Advanced Search" feature of some commercial search engines. The user can specify number of Web pages to be retrieved, domains (e.g. .gov, .edu or .com) to be

included in the search results, number of Internet Spiders to be used, and so on. In CI Spider, the user can also choose either Breadth-First Search or Best-First Search to be the algorithm used by the Internet Spiders.

The SOM also is highly customizable in the sense that the user can select and deselect phrases for inclusion in the analysis and produce a new map at any time. If the user is not satisfied with the map produced, he can always go back to the previous step to discard some phrases that are irrelevant or too general and generate a new map within seconds. The systems also let each user store a personalized "dictionary" which contains the terms that the user does not want to be included in the results of the Arizona Noun Phraser and the SOM.

Another important functionality incorporated in the system is the Save function. The user can save a completed search session and open it at a later time. This feature allows the user to perform a Web search and review it in the future. This also helps users who want to monitor Web pages on a particular topic or Web site.

3.4 Experimental Design

Two separate experiments have been conducted to evaluate CI Spider and Meta Spider. Because the two spider systems were designed to facilitate both document retrieval and document categorization tasks, traditional evaluation methodologies would not have been appropriate. These methodologies treat document retrieval and document categorization separately. In the experiments, the experimental task was therefore so designed as to

permit evaluation of the performance of a combination of their functionalities in identifying the major themes related to a certain topic being searched.

3.4.1 Evaluation of CI Spider

In the experiment, CI Spider was compared with the usual methods that Internet users use to search for information on the Internet. General users usually use popular commercial search engines to collect data on the Internet, or they simply explore the Internet manually. Therefore, these two search methods were compared with the CI Spider. The first method evaluated was Lycos, chosen because it was one of the few popular search engines that offered the functionality to search for a certain keyword in a given Web domain (e.g., http://www.ibm.com/). The second method was "within-site" browsing and searching. In this method the subject was allowed to freely explore the contents in the given Web site using an Internet browser. When using CI Spider, the subject was allowed to use all the components including Noun Phraser and SOM.

Each subject first tried to locate the pages containing the given topic within the given Web host using the different search methods described above. The subject was required to comprehend the contents of all the Web pages relevant to that keyword, and to summarize the findings as a number of themes. In the experiment, a theme was defined as "a short phrase which describes a certain topic." Phrases like "success of the 9840 tape drive in the market" and "business transformation services" are examples of themes in the experiment. By examining the themes that the subjects came up with using different search methods, it is possible to evaluate how effectively and efficiently each method

helped a user locate a collection of documents and gain a general understanding of the response to a given search query on a certain Web site. Web sites with different sizes, ranging from small sites such as http://www.eye2eye.com/ to large sites such as http://www.ibm.com/ were chosen for the experiments.

Six search queries were designed for the experiment, based on suggestions given by professionals working in the field of competitive intelligence. For example, one of the search tasks was to locate and summarize the information related to "merger" on the Web site of a company called Phoenix Technologies (http://www.phoenix.com/). Two pilot studies were conducted in order to refine the search tasks and experiment design. During the real experiment, thirty subjects, mostly information systems management students, were recruited and each subject was required to perform three out of the six different searches using the three different search methods. At the beginning of each experiment session, the subject was trained in using these search methods. Each subject performed at least one complete search session for each of the 3 search methods until he felt comfortable with each method. Rotation was applied such that the order of search methods and search tasks tested would not bias the results. Some sample documents used in the experiment can be found in Appendix A.

3.4.2 Evaluation of Meta Spider

Meta Spider was compared with MetaCrawler and NorthernLight. MetaCrawler (http://www.metacrawler.com/) is a renowned, popular meta-search engine and has been recognized for its adaptability, portability and scalability (Selberg & Etzioni, 1997).

NorthernLight (http://www.northernlight.com/), being one of the largest search engines on the Web, provides clustering functionality to classify search results into different categories. When using Meta Spider, the subject was allowed to use all the components including Noun Phraser and SOM.

Each subject was required to use the different search tools to collect information related to the given topic. As in the CI Spider experiment, each subject was required to summarize the Web pages collected as a number of themes. The search topics were chosen from TREC 6 topics. The TREC series was sponsored by the National Institute of Standards and Technology (NIST) and the Defense Advanced Research Projects Agency (DARPA) to encourage research in information retrieval from large text collections. Because the TREC topics were not especially designed for Web document retrieval, care was taken to make sure each search topic was valid and retrievable on the Internet. Thirty undergraduate students from an MIS class at The University of Arizona were recruited to undertake the experiment. Training and rotation similar to those used in the CI Spider experiment were applied. Some sample documents used in the experiment are attached in Appendix B.

3.5 Experimental Results and Discussions

Two graduate students majoring in library science were recruited as experts for each experiment. They employed the different search methods and tools being evaluated and came up with a comprehensive set of themes for each search task. Their results were then

aggregated to form the basis for evaluation. Precision and recall rates for themes were used to measure the effectiveness of each search method.

The time spent for each experiment, including the system response time and the user browsing time, was recorded in order to evaluate the efficiency of the 3 search methods in each experiment. During the studies, the subjects were encouraged to talk about the search method used and their comments were recorded. Finally, each subject filled out a questionnaire to record further comments about the 3 different methods.

3.5.1 Experiment Results of CI Spider

The quantitative results of the CI Spider experiment are summarized in Table 3.1. Four main variables for each subject have been computed for comparison: precision, recall, time, and ease of use. Precision rate and recall rate were calculated as follows:

$$precision = \frac{number\ of\ correct\ themes\ identified\ by\ the\ subject}{number\ of\ all\ themes\ identified\ by\ the\ subject}$$

$$recall = \frac{number\ of\ correct\ themes\ identified\ by\ the\ subject}{number\ of\ all\ themes\ identified\ by\ the\ expert\ judges}$$

The time recorded was the total duration of the search task, including both response time of the system and the browsing time of the subject. Usability was calculated based on subjects' responses to the question "How easy/difficult is it to locate useful information using [that search method]?" Subjects were required to choose a level from a scale of 1 to 5, with 1 being the most difficult and 5 being the easiest.

In order to see whether the differences between the values were statistically significant, *t*-tests were performed on the experimental data. The results are summarized in Table 3.2. As can be seen, the precision and recall rates for CI Spider both were significantly higher than those of Lycos at a 5% significant level. CI Spider also was given a statistically higher value than Lycos and within-site browsing and searching in usability.

Table 3.1: Experiment results of CI Spider

	CI Spider	Lycos	Within-Site Browsing/ Searching
Precision: Mean	**0.708**	0.477	0.576
Variance	0.120	0.197	0.150
Recall: Mean	**0.273**	0.163	0.239
Variance	0.027	0.026	0.033
Time(min): Mean	10.02	9.23	**8.60**
Variance	11.86	44.82	36.94
Usability*: Mean	**3.97**	3.33	3.23
Variance	1.34	1.13	1.29

*Based on a scale of 1 to 5, where 1 being the most difficult to use and 5 being the easiest.

Table 3.2: *t*-test results of the CI Spider experiment

	CI Spider vs Lycos	CI Spider vs Within-Site B/S	Lycos vs Within-Site B/S
Precision	***0.029**	0.169	0.365
Recall	***0.012**	0.459	0.087
Time	0.563	0.255	0.688
Usability	***0.031**	***0.016**	0.126

* The mean difference is significant at the 0.05 level.

3.5.2 Experiment Results of Meta Spider

Three variables, namely precision, recall, and time, have been computed for comparison in the Meta Spider experiment and the results are summarized in Table 3.3. The *t*-test results are summarized in Table 3.4. In terms of precision, Meta Spider performed better than MetaCrawler and NorthernLight, and the difference with NorthernLight was

statistically significant. For recall rate, Meta Spider was comparable to MetaCrawler and better than NorthernLight.

Table 3.3: Experiment results of Meta Spider

	Meta Spider	Meta-Crawler	Northern-Light
Precision: Mean	**0.815**	0.697	0.561
Variance	0.281	0.315	0.402
Recall: Mean	0.308	**0.331**	0.203
Variance	0.331	0.291	0.181
Time(min): Mean	**10.93**	11.13	11.00
Variance	4.04	4.72	5.23

Table 3.4: *t*-test results of the Meta Spider experiment

	Meta Spider vs Meta-Crawler	Meta Spider vs Northern-Light	Meta-Crawler vs Northern-Light
Precision	0.540	***0.013**	0.360
Recall	1.000	0.304	0.139
Time	1.000	1.000	1.000

* The mean difference is significant at the 0.05 level.

3.5.3 Strength and Weakness Analysis

3.5.3.1 Precision and Recall

The *t*-test results show that CI Spider performed statistically better in both precision and recall than Lycos, and Meta Spider performed better than NorthernLight in precision. In terms of precision, I suggest that the main reason for the high precision rate of CI Spider and Meta Spider is their ability to fetch and verify the content of each Web page in real time. That means the Spiders can ensure that every page shown to the user contains the keyword being searched. On the other hand, it was found that indexes in Lycos and NorthernLight, like most other search engines, were often outdated. A number of URLs returned by these two search engines were irrelevant or dead links, resulting in low

precision. Subjects also reported that in some cases two or more URLs returned by Lycos pointed to the same page, which led to wasted time verifying the validity of each page.

The high recall rate of CI Spider is mainly attributable to the exhaustive searching nature of the spiders. Lycos has the lowest recall rate because, like most other commercial search engines, it samples only a number of Web pages in each Web site, thereby missing other pages that contain the keyword. For within-site browsing and searching, a user is more likely to miss some important pages because the process is mentally exhausting.

3.5.3.2 Display and Analysis of Web Pages

In the CI Spider study, subjects believed it was easier to find useful information using CI Spider (with a score of 3.97/5.00) than using Lycos domain search (3.33) or manual within-site browsing and searching (3.23). Three main reasons may account for this. The first is the high precision and recall discussed above. The high quality of data saved users considerable time and mental effort. Second, the intuitive and useful interface design helped subjects locate information they needed more easily. Third, the analysis tools helped subjects form an overview of all the relevant Web pages collected. The Arizona Noun Phraser allowed subjects to narrow and refine their searches as well as provided a list of key phrases that represented the collection. The Self-Organizing Map generated a 2-dimensional map display on which subjects could click to view the documents related to a particular theme of the collection.

In the post-test questionnaires in the CI Spider experiment, it was found that 77% of subjects found the Good URL List useful for their analyses, while 40% of subjects found either the Noun Phraser or the SOM useful. This suggests that while many subjects preferred traditional search result list, a significant portion of subjects were able to gain from the use of advanced analysis tools. Similar results were obtained in the Meta Spider experiment, in which 77% of subjects found the list display useful and 45% found either the Noun Phraser or the SOM useful.

3.5.3.3 Speed

The *t*-test results demonstrated that the three search methods in each experiment did not differ significantly in time requirements. As discussed in the previous section, the time used for comparison is total searching time and browsing time. Real-time indexing and fetching time, which usually takes more than 3 minutes, also was included in the total time for CI Spider and Meta Spider. Therefore, it is anticipated that the two Spiders can let users spend less time and effort in the whole search process, because the users only need to browse the verified results.

3.6 Conclusion

The results of the two studies are encouraging. They indicate that the use of CI Spider and Meta Spider can potentially facilitate the Web searching process for Internet users with different needs by using a personalized approach. The results also demonstrated that powerful AI techniques such as noun phrasing and SOM can be processed on the user's personal computer to perform further analysis on Web search results, which allows the

user to understand the search topic more correctly and more completely. I believe that many other powerful techniques can possibly be implemented on client-side search tools to improve efficiency and effectiveness in Web search as well as other information retrieval applications.

CHAPTER 4: PERSONALIZED WEB SEARCH AGENTS FOR SPECIFIC DOMAINS

4.1 Background

Searching in particular domains is another problem often faced by Web users. It is important for users to find specific information in different domains, such as medicine, information technology, country music, and so on. Because it is difficult for general-purpose search engines to keep up with the increasing size of the Web, they are often not able to keep a comprehensive and up-to-date index of Web pages in each domain. To make it worse, because a user cannot specify a search domain, a search query may bring up Web pages both within and outside the desired domain. For example, a user searching for "cancer" may get Web pages related to the disease as well as those related to the Zodiac sign. As a result, the user has to browse through the list of results to identify relevant Web pages, a task which requires significant mental effort. Directory services such as Yahoo! (http://www.yahoo.com/) provide users with a hierarchy of classified topics. While the precision is high, recall rate suffers as each page included in the results has to be manually evaluated. In this chapter, I look at the Web searching problems in two domains – healthcare and nanotechnology.

Healthcare is an information-intensive business. Hersh (1996) classified textural health information into two main categories. The first type is patient-specific information. The second type is knowledge-based information, which can be further divided into the following three layers. Primary knowledge-based information contains original research reported in journals, books, reports and other sources. Secondary knowledge-based

information consists of indexes that catalog primary literature. The most widely known of these in the health fields is *Index Medicus*. MEDLINE, a widely used medical computer database, is the computerized version of *Index Medicus*. Tertiary literature consists of summaries of research in review articles, books, etc. The data volume of these information resources is overwhelming. According to a TIME magazine special issue on online health (March 2001), 26,000 Web sites provide health-related information. MEDLINE itself contains over 11 million references to journal articles in life sciences. LOCATOR*plus*, NLM's online catalog includes over 800,000 catalog records for books, audiovisuals, journals, computer files, and other materials in the Library's collections.

Nanotechnology, the study of science and technology at the scale of nanometer – a billionth of a meter, is another interesting domain to study specialized Web searching. As nanotechnology is young and fast growing, it is important for researchers and practitioners to find up-to-date information on the Web. However, information sources and quality on the Web are diverse (Lawrence & Giles, 1999). Useful nanotechnology information can include scientific papers, journals, technical reports, and Web pages of widely-varied quality, among others. As a result, users need to go to multiple information sources (Web sites, search engines, online academic databases, etc.) and conduct significant manual analysis to identify quality, time-critical information. In addition, the nanotechnology domain encompasses a diversity of research perspectives and application areas such as nanoscale physics, nanoscale medicine, and nanoscale electronics. This has resulted in terminology and vocabulary differences (Furnas et al., 1987; Chen, 1994). Researchers in different disciplines use different terminologies to describe their findings

and user search terms may not be the same as the indexing terminologies used in databases. In addition, there also exists a dichotomy between academia and industry in the field. The first seeks long-term research issues and the second focuses on short-term development. Language differences make the problem worse, because researchers in the discipline come from a variety of regions and countries. Research findings written in a language other than English can be difficult to search for on the Web.

4.2 Related Work

4.2.1 Searching in the Healthcare Domain

Kiley (1999) summarized two main ways of searching for medical information on the Internet: (1) by using a free-text search engine or Web directory service to interrogate a database of Internet resources, and (2) by browsing/searching through human-evaluated sources of information.

Free text searching refers either to browsing the Web sites or to utilizing an Internet search service. Some examples of reputable medical Web sites are *MayoClinic.com* (http://www.mayoclinic.com/), *WebMD* (http://www.webmed.com/), *Inteli-Health* (http://www.intelihealth.com/), and *DrKoop* (http://www.drkoop.com/). A few Web portals also have emerged as popular sources of reliable health information. Among these are *Yahoo Health* pages (http://www.yahoo.com/Health/) and *CBSHealthWatch* (http://www.cbshealthwatch.com/). Kiley pointed out that the weakness of free-text searching lies in the indiscriminate method for document retrieval and non-evaluation of content, both of which contribute to lack of structure and information overload. An

important consideration of IR in health care is information quality control and quality assurance. A substantial number of Web-based information sources are not suitable for direct clinical application because of poor data organization, questionable validity and uncertain reliability (Westberg & Miller, 1999). It also has been shown that many medical search engines return Web pages that are not even medical related (Bin & Lun, 2001).

Evaluated subject catalogs or peer-reviewed Web directories deal with these information quality issues, as they are compiled by health professionals and provide points of access to relevant and authoritative sources of healthcare-related content. Some well-known examples are NLM's MEDLINE-based literature databases; Medical Matrix (http://www.medicalmatrix.org/), compiled by the Internet Working Group of the American Medical Informatics Association; Organising Medical Networked Information, or OMNI (http://omni.ac.uk/), created by a core team of information specialists and subject experts based at the University of Nottingham Greenfield Medical Library; and CliniWeb (http://www.ohsu.edu/cliniweb/) developed by the Oregon Health Sciences University.

Although evaluated subject catalogs have gained enthusiastic acceptance in practice, new problems have arisen with respect to effectively retrieving information from them. Studies have shown that MEDLINE and its derivatives can be helpful in answering clinical questions, but finding specific answers can be time-consuming, in part because of

the effort required to sift through large collections of relevant publications (Westberg & Miller, 1999).

4.2.2 Searching in the Nanotechnology Domain

Compared to the medical domain, the number of search engines available in the nanotechnology domain is very small. To the best of my knowledge, there only exist two specialized search engines dedicated to the domain, namely NanoSpot (http://www.nanospot.org/) and Nanotechnology.com (http://www.nanotechnology.com/). Both search engines only provide simple searching functions without any post-retrieval capabilities.

As nanotechnology is a very diverse field, a lot of useful information can be retrieved from other data sources. For example, several academic abstract and full-text journal databases, such as Scirus, Medline, IEEE Xplore and ACM Digital Library, contain some useful information for nanotechnology research. The database of the US Patent and Trademark Office also provides some documents relevant to the nanotechnology domain. However, such information often constitutes only a small portion of the corresponding databases. It is a time-consuming and tedious process for users to go through all the different databases and perform searching one by one.

4.3 Proposed Approaches

To address the above problems, the Meta Spider architecture outlined in Chapter 3 is customized for domain-specific Web searching. The diagram of the new architecture is

shown in Figure 4.1. The major difference is that instead of connecting with multiple general-purpose search engines, the new architecture connects with a set of different search engines, particular domain-specific ones. Two tools, namely Cancer Spider and Nano Spider, were developed based on the architecture.

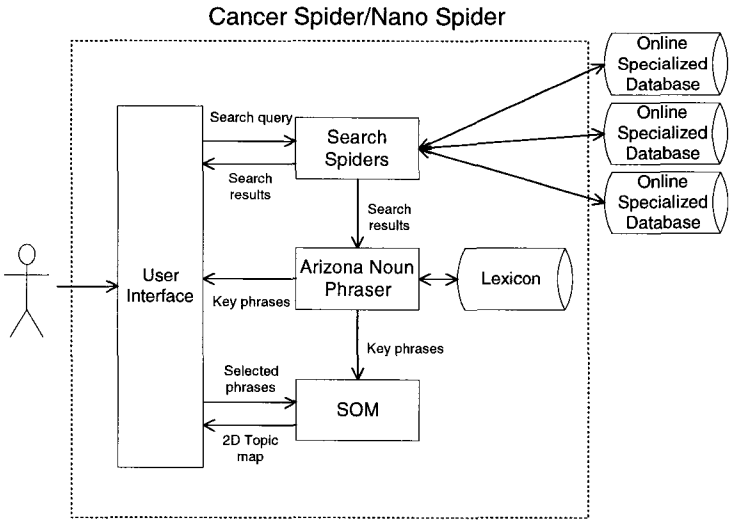

Figure 4.1: System architecture of Cancer Spider and Nano Spider

4.3.1 Cancer Spider

Intended end users of Cancer Spider are cancer researchers, physicians and medical librarians. The design of Cancer Spider builds on the ability to offer value-added capabilities that assist users in effectively weeding out irrelevant Web pages and locating useful information. The major functions of the Cancer Spider are similar to that of Meta Spider. The functions include: (1) searching (2) filtering, (3) phrase selection, and (4) visualization using self-organizing map (SOM). Currently, Cancer Spider connects to three databases: MEDLINE, CANCERLIT and PDQ. MEDLINE, or Medical Literature, Analysis, and Retrieval System Online (http://www.ncbi.nlm.nih.gov/entrez/query.fcgi)

is the U.S. National Library of Medicine's (NLM) premier bibliographic database that contains over 11 million references to journal articles in life sciences with a concentration on biomedicine. Produced by the National Cancer Institute (NCI), CANCERLIT (http://cnetdb.nci.nih.gov/cancerlit.shtml) contains references to the vast realm of cancer literature published from the 1960's to the present. PDQ (http://cnetdb.nci.nih.gov/pdqsrch.shtml), NCI's comprehensive cancer database contains peer-reviewed summaries on cancer treatment, screening, prevention, genetics, and supportive care. A sample user session of Cancer Spider is shown in Figure 4.2. Readers are referred to Chapter 3.3 for a more detailed discussion of the architecture.

4.3.2 NanoSpider

In NanoSpider, a much wider range of data sources were incorporated (Chau et al., 2002a). There are four categories of search engines in NanoSpider. The first category is general-purpose search engines (AltaVista, Excite, Google, Goto, Lycos) which locate pages from the whole Web, aiming at high coverage. The second category involved nanotechnology-specific search engines (NanoSpot) and aims at high precision by searching only for pages in the nanotechnology domain. The third category consists of academic literature search engines (IEEE Xplore, MEDLINE) which tries to locate journal articles and abstracts that may not be covered by the other search engines. The fourth category is news search engines (USA Today, ABC News) trying to retrieve the latest, breaking information by searching online news articles. These data sources were added to NanoSpider rather easily, as could many other data sources. A sample user session is shown in Figure 4.3.

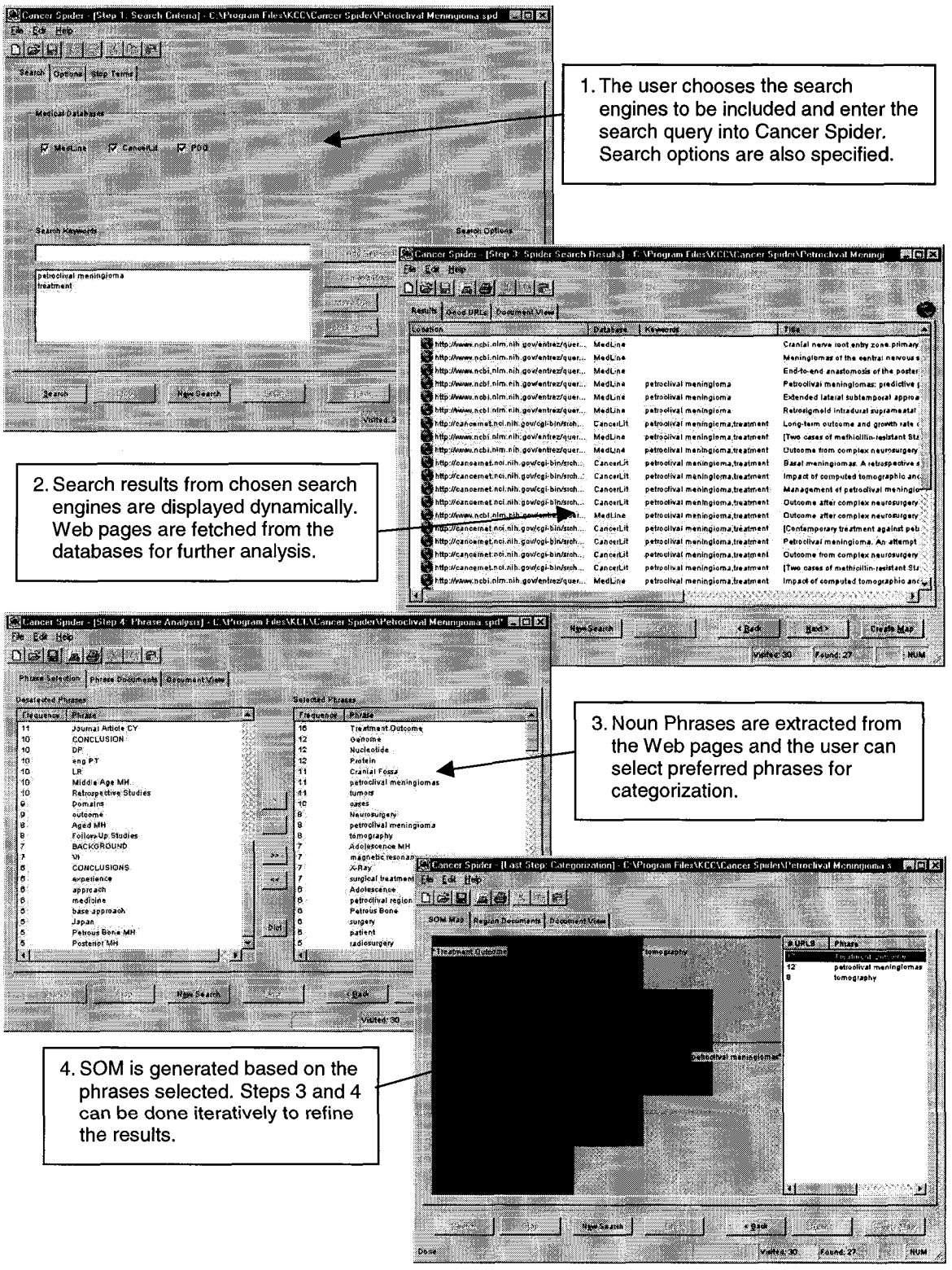

Figure 4.2: Example of a user session with Cancer Spider

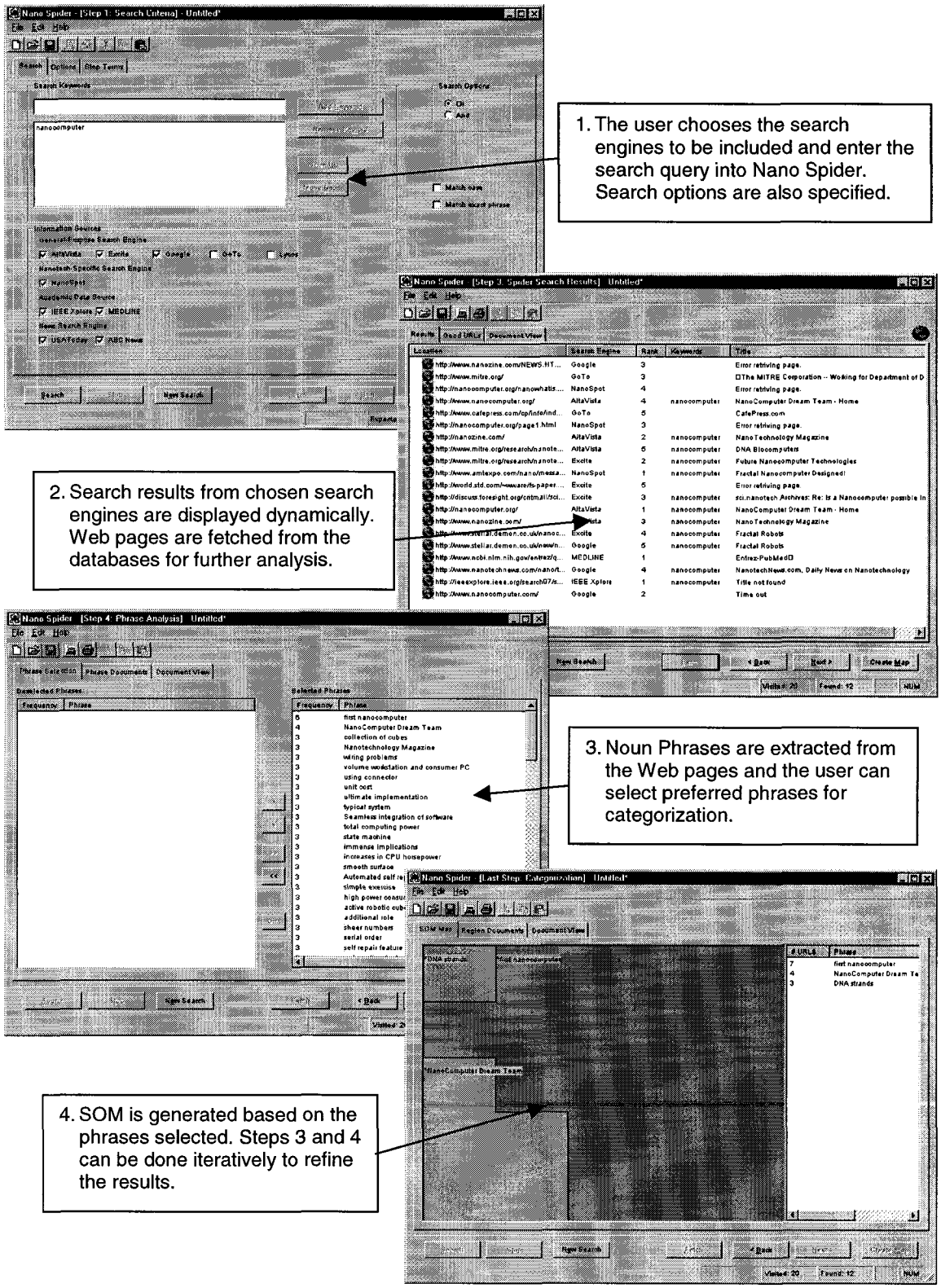

Figure 4.3: Example of a user session with Nano Spider

4.4 Experimental Design

4.4.1 Comparison Base

A user study was designed and conducted to evaluate the proposed approach of meta searching and categorization implemented in the Cancer Spider system. In the experiment, Cancer Spider was compared with the NLM Gateway (http://gateway.nlm.nih.gov/gw/Cmd?GMBasicSearch), the portal search engine to NLM's multiple literature databases.

The NLM Gateway is a Web-based system that lets users search simultaneously multiple retrieval systems at the U.S. National Library of Medicine (NLM) through a unified Web interface. The Gateway connects users with multiple NLM retrieval systems, which include MEDLINE/PubMed, OLDMEDLINE, LOCATOR*plus*, AIDS Meetings, HSR Meetings, HSRProj, MEDLINE*plus* and DIRLINE. Released to the public in October 2000, the NLM Gateway provides advanced search capabilities such as terminology suggestion and search history archiving. For users who are not familiar with medical terminology, the system is particularly helpful in providing definitions and related terms of the targeted search, based on NLM's MeSH (Medical Subject Heading) and the UMLS (Unified Medical Language Systems) Metathesaurus. The purpose of the UMLS is to develop *knowledge sources* that can be used by a wide variety of applications programs to overcome retrieval problems caused by differences in terminology and the scattering of relevant information across many databases.

4.4.2 Theme-based Evaluation

Because Cancer Spider has been designed to facilitate and integrate both document retrieval and automated categorization, traditional evaluation methodologies that treat document retrieval and categorization completely separately are not directly applicable. A new evaluation framework was adopted based on theme identification. Instead of tasks that focus on specific information search, the subjects were given relatively open-ended information search tasks and ask him or her to summarize search results in the form of a number of themes. This is similar to the open-ended soft queries created by the National Institute of Standards and Technology (NIST) for the TREC-6 ad hoc task.

In the experiment, a theme was defined as "a short phrase that summarizes a specific aspect of the search results." Phrases like *stereo tactic radio surgery such as gamma knife* and *diagnostic and therapeutic use of radioactive iodine as treatment* are examples of such themes. Within this theme-based framework, protocols were designed to permit evaluation of the extent to which combined document retrieval and categorization facilitate users' identification of major themes related to a certain topic. Section 4.5 presents a detailed discussion of how this theme-based framework was used in the user study.

4.4.3 Experiment Hypotheses

The overall research goal is to examine how effectively and efficiently Cancer Spider supports the user's ability to locate useful information and to understand retrieved documents. The specific hypotheses examined in the user study are:

H1 Compared to the NLM Gateway, Cancer Spider achieves higher precision when assisting the user in theme identification.

H2 Compared to the NLM Gateway, Cancer Spider achieves higher recall when assisting the user in theme identification.

H3 Compared to the NLM Gateway, Cancer Spider achieves better performance as evaluated by F-measure.

H4 Compared to the NLM Gateway, Cancer Spider requires less time for users to locate useful information and to understand retrieved documents.

H5 Compared to the NLM Gateway, Cancer Spider requires less manual browsing effort from its users.

4.4.4 Experiment Tasks

In the user study, six search questions were selected from a list of over one hundred real-world research questions compiled by medical librarians at the Arizona Health Sciences Library. These questions include questions asked by library users at the reference desk at the medical library, as well as questions raised among physicians or cancer researchers in their practice. The six search questions used in the experiment were as follows:

1. What role do antioxidants play in reducing cancer risk and which neoplasms are most affected?
2. How can chemotherapy be used to cure ALL (acute lymphocytic leukemia)?
3. What are the treatment options for petroclival meningioma?
4. What are the connections between breast cancer and iodine?

5. What are the symptoms of pancreatic neoplasm?
6. What is the role of folic acid in the prevention of colon cancer?

These questions were chosen because they allowed subjects to explore a given topic, as opposed to locating specific answers for a closed-ended question. They represent a balanced combination of general, exploratory, and focused searching. From the perspective of cancer medicine, the selection represents a good mixture of both clinically-oriented questions and research questions at the molecular level.

4.4.5 Experimental Subjects and Expert Evaluators

Thirty cancer researchers, including graduate students of cancer-related majors, medical students, and lab technicians from the Arizona Cancer Center were recruited to participate in the experiment. This subject body represented a good combination of cancer research and clinical background. Subjects were assigned information search tasks and required to jot down the themes they had identified after searching on a given IR system. For each IR system, each subject was assigned a search task and was instructed to perform a search on the system. To avoid a potential fatigue effect, the order in which each IR system was evaluated was rotated. Although subjects were not given a specific time frame within which to perform the searches, they were encouraged to stop after 20 minutes (most subjects took less than 20 minutes to finish the tasks).

Two senior Ph.D. students at the Arizona Cancer Center majoring in Cancer Biology and Molecular and Cellular Biology, respectively, were recruited as content experts to

evaluate all subjects' answers. First of all, the two experts performed all six searches independently and identified themes for each search task. Then they compared and converged their own answers into one final standard answer set, against which all subjects' answers were evaluated. To ensure the comprehensiveness of the standard answer set, the experts were not restricted to using only the two IR systems under investigation but could choose to utilize any type of available information resources they felt most suitable for answering a particular question. Finally, the experts worked together to evaluate all subjects' answers.

It is worth mentioning that the evaluation was based on semantic meanings rather than exact phrasing. For example, Question 1 was "*What role do antioxidants play in reducing cancer risk and which neoplasms are most affected?*" The experts recognized that antioxidants protect cells from varying kinds of oxidative damage. Answers that mentioned only protection of lipid oxidation were recognized as a valid theme, since lipid oxidation is one type of damage which is prevented by antioxidants. However, answers that listed several types of oxidative damage, e.g., lipid per oxidation, protein cross-linking, and DNA modification, each as a separate theme, were considered to be one valid theme, as all are merely different forms of oxidative damage prevented by antioxidants.

4.4.6 Experiment Measurements

Both quantitative and qualitative data were collected and examined. For quantitative data, the primary interests were in performance and efficiency of the IR systems under

investigation. Performance was evaluated by theme-based precision and recall rates, whereas efficiency was measured by searching time, precision effort, and the total number of documents browsed.

Precision rate was computed as the number of correct themes identified by the subject divided by the total number of themes in the subject's answer set. Recall rate was calculated as the number of correct themes identified by the subject divided by the total number of themes in the standard answer compiled by the experts. Searching time was recorded as number of minutes spent on the searching, including both the response time of the system and subjects' browsing time. Time elapsed while subjects wrote their answers on the answer sheet were not included.

Precision effort is a new user-oriented measure developed in this study, and is defined as the ratio between the number of relevant documents the user found helpful and the number of documents examined in an attempt to find the useful documents. The number of documents browsed records the number of articles subjects viewed in a search session seeking answers to the search questions.

Qualitative data were drawn from user search logs and questionnaires. The search log created for each subject recorded major observations of user behaviors, as well as the user's *think aloud* discourse during searching. To enable qualitative comparison of the two systems, subjects were required to fill out questionnaires at the end of their search sessions. The questionnaire investigated subject's experiences on five different

dimensions: (1) user interface, (2) usefulness of the information retrieved in answering the search questions, (3) subjects' level of certainty about their answers, (4) user satisfaction of the search experience, and (5) the amount of knowledge obtained after the search. Based on each subject's search log and completed questionnaire, a verbal protocol analysis was conducted to discover emerging themes or trends.

Samples of documents used in the experiment can be found in Appendix C.

4.5 Experimental Results and Discussions

4.5.1 Performance

Measured by theme-based precision and recall, the performances of the two IR systems, Cancer Spider and the NLM Gateway were on a par with each other. The main statistics are summarized in Table 4.1. Both the mean precision and the mean recall of Cancer Spider were comparable with those of NLM Gateway. On average, the NLM Gateway seemed to be slightly superior to Cancer Spider, but the difference was not statistically significant, as suggested by the p-value of the pair-wise *t*-test. The F-measure, which was the integral measure of precision and recall, also reveals the same result. These findings indicated that hypotheses *H1*, *H2*, and *H3* were not confirmed.

The relative comparability of the two IR systems can be explained by the fact that each system has its unique, differentiating features. First, NLM Gateway, being the portal site to many NLM administered medical literature databases, has the advantage of

comprehensive data coverage. In comparison, the current version of Cancer Spider connects to only three databases, considerably smaller than that of the NLM Gateway.

Table 4.1: System performance of Cancer Spider and NLM Gateway

	Sample size	Cancer Spider mean	Cancer Spider variance	NLM Gateway mean	NLM Gateway variance	p-value of *t*-test
Precision	30	0.803	0.117	0.826	0.122	0.7572
Recall	30	0.533	0.112	0.539	0.121	0.9523
F-measure	30	0.612	0.105	0.622	0.110	0.9056

Second, the use of UMLS Metathesaurus in the NLM Gateway greatly improved the system's usability and performance. Of the 30 subjects in the experiment, 20 utilized UMLS Metathesaurus to find related terminology, and these users generated positive feedback. Cancer Spider does not provide domain-specific, high quality ontological assistance to its users.

Third, the traditional Web-enabled interface of NLM Gateway provides a user-friendly navigating environment. The document summary presents to the users not only full-text titles, but also authors, resources and publication dates of journal articles. In contrast, Cancer Spider, as a research prototype, provides limited graphic options and navigating capability that is different from typical Web-based interfaces. This unfamiliar user interface and lack of certain contextual information (e.g., Cancer Spider does not show author and other meta-level resource information to the user) may have affected searching performance.

Despite these disadvantages, Cancer Spider's performance was comparable to the NLM Gateway. I hypothesize that this was largely due to Cancer Spider's value-added capabilities in post-search processing and categorization. First of all, it was found that users liked the key phrase extraction feature of Cancer Spider. This feature provides immediate feedback to the user, whether the given document contains the search terms or not. As the system supports multiple search terms, the user can first roughly judge the relevance of a returned article by eyeballing the number of search terms it contains. Then, by clicking on the *Rank* bar on the top of the *Search* panel, the user can rank all the returned documents sorted by the number of search terms they contain.

Second, the categorization function of Cancer Spider helps users narrow down the search scope by focusing on the noun phrases (key topics) in which a user is interested. At the *Phrase Selection* stage, the user can click on any noun phrases of interest that have been extracted from the full-text documents to find a subset of articles that focus on the particular topic. For example, one subject saw the value of the Cancer Spider clustering function when working on the topic *using aspirin in skin cancer prevention.* He was quoted as saying "it helps me to focus on different aspects of the topic, e.g., how aspirin is related to this particular gene in the molecular pathway."

Third, the interaction between the system and users is comparatively active and dynamic, unlike traditional search services in which users passively take whatever the system generates. Subjects indicated that they liked the flexible, interactive design of Cancer Spider, which gives them a sense of "user-in-control." It has been shown that users

perform better and have increased subjective satisfaction when they can view and control the search (Koenemann & Belkin, 1996). This benefit is illustrated by the following comment made by one of the subjects: "I like the system because I have the control over how the documents can be arranged by keywords, as opposed to other search engines, where the user cannot manipulate the result." At the searching stage, users can rank the returned documents by the number of search terms contained so that they can directly visit those most relevant without sifting blindly through document summaries. At the clustering stage, users can sort noun phrases by the frequency of their appearance in the returned documents or simply in the alphabetic order. Such handy tools give users a sense of control and are helpful in assisting them to quickly target their search focus.

4.5.2 Efficiency

In the experiment, search time, the number of documents browsed, and precision effort were used to measure the efficiency of the two medical IR systems under investigation. The key experimental findings are summarized in Table 4.2.

4.5.2.1 Time

The mean search time for Cancer Spider (10.22 minutes) was significantly shorter than that of the NLM Gateway (14.00 minutes), with a p-value of 0.0003. This has confirmed hypothesis *H4* at the 1% level that Cancer Spider requires less time than NLM Gateway to locate useful information and to understand retrieved documents.

Table 4.2: Searching time and effort

	Sample size	Cancer Spider mean	Cancer Spider variance	NLM Gateway mean	NLM Gateway variance	p-value of *t*-test
Time (in minutes)	30	10.22	15.64	14.00	23.18	0.0003*
No. of documents browsed	30	4.53	3.57	6.23	12.46	0.0067*
Precision effort	30	0.65	0.05	0.43	0.07	0.0009*

* The difference is statistically significant at the 1% level.

A main reason for this result is that Cancer Spider provides its categorization function. First of all, based on search terms, Cancer Spider filters out all irrelevant documents. Second, by grouping a large retrieved document set into small subtopics, clustering helps users understand the structure of the topic and expedites the process of reading documents and understanding the topic as a whole.

4.5.2.2 Number of Documents Browsed and Precision Effort

The mean number of documents browsed using Cancer Spider (4.533) was significantly lower than that using NLM Gateway (6.233), with a p-value of 0.0067. The mean precision effort of Cancer Spider (0.648) is significantly higher than that of NLM Gateway (0.436), with a p-value of 0.0009. Both of these measures represent the amount of manual browsing effort. For both of them, the differences between the two systems are significantly at the 1% level and the findings have confirmed hypothesis *H5* that, compared to NLM Gateway, Cancer Spider requires less manual browsing effort from its users.

The key phrase extraction feature of Cancer Spider allows users to quickly judge the relevance of a given document. It saves users a significant amount of time not to have to go through all the document summaries. Furthermore, Cancer Spider groups all returned

documents into sub-topics based on a clustering algorithm, letting users focus only on sub-topics of interest to them, as opposed to opening and reading every single document. Users need to browse fewer documents. Similarly, the higher precision effort associated with Cancer Spider is closely related to Cancer Spider's key phrase extraction and categorization capability. As commented by one of the subjects, "It (Cancer Spider) returned the highest amount of usable citations compared to other search engines, which usually only return 20-30% of what I need. It is the best search engine I've ever used."

4.5.2.3 Questionnaire Results

The questionnaire was designed primarily to discover users' subjective experience with the medical IR systems under study and was comprised of questions relating to five different dimensions of the user experience: user interface, usefulness of the information retrieved in answering the search questions, users' level of certainty about their answers, user satisfaction of the search experience, and the amount of knowledge obtained after the search.

Table 4.3: Subjects' ratings of Cancer Spider and NLM Gateway

	Sample size	Cancer Spider		NLM Gateway		p-value of
		mean	variance	mean	variance	*t*-test
Interface	30	3.55	0.68	3.72	1.14	0.5018
Usefulness	30	3.93	0.71	3.41	1.04	0.0831**
Certainty	30	4.07	0.57	3.72	1.06	0.0479*
Satisfaction	30	3.79	0.74	3.14	0.91	0.0079*
Knowledge obtained	30	3.72	0.64	3.07	1.50	0.0139*

* The difference is statistically significant at the 5% level.
** The difference is statistically significant at the 10% level.

On a scale of 1 to 5 (5 being the most desirable), the result of the questionnaire is shown in Table 4.3. The data show that NLM Gateway scored higher in user interface, but the difference was not significant. Cancer Spider scored higher in all the other four categories. The differences in three of these categories were statistically significant at the 5% level, and the remaining one (Usefulness) is significant at the 10% level. The smaller variances in the evaluation of Cancer Spider also show that Cancer Spider performs more consistently for subjects with different backgrounds.

The interface of Cancer Spider is one of the dominant themes in the verbal protocol analysis. Although subjects showed strong interest in the interactive interface, they made many valuable suggestions for improvement. For instance, tabs such as *Good URLs* and *Document View* were designed to cut through different windows within a panel. Subjects who were accustomed to simple Web browser navigation found the distinction between within-panel and inter-panel switching confusing.

4.6 Conclusion

In this chapter, I demonstrated the feasibility of building specialized search agents such as Cancer Spider and Nano Spider. The results of the Cancer Spider experiment are encouraging. Both Cancer Spider and the NLM Gateway have strengths and weaknesses. The NLM Gateway, as a portal site to many high-quality NLM databases, is a conventional medical search engine. Cancer Spider's data coverage is not as comprehensive as that of the NLM Gateway. However, Cancer Spider draws its strength from post-retrieval processing and categorization capability.

With respect to system performance measured by theme-based precision and recall, Cancer Spider and the NLM Gateway achieved comparable results. When measured by system efficiency, Cancer Spider demonstrated statistically significant superiority in search time, the number of documents browsed, and precision effort. Cancer Spider's key phrase extraction feature and categorization tools have been shown helpful in assisting users to locate useful information.

While the reported user study focused on the post-retrieval processing capabilities of Cancer Spider, a separate evaluation of the visualization component will be an interesting research issue. It will be interesting to evaluate how users perceive search results categorized into a topic-map compared to a traditional ranked-list. It is also interesting to study how the different visualization metaphors affect a user's search effectiveness and efficiency.

Another interesting area to research into is UMLS-enhanced semantic parsing. Semantic parsing involves natural language processing capabilities that go beyond simple part-of-speech tagging and try to achieve a deeper understanding of semantic types and relationships between concepts that appear in the queries or documents. User queries will not be limited to the exact key phrases that a user enters. By including other semantically relevant search terms, documents containing closely related concepts also will be retrieved, thus solving the problem of vocabulary differences, a major issue in medical IR.

CHAPTER 5: USING MULTI-AGENT TECHNIQUES TO FACILITATE COLLABORATIVE WEB MINING

5.1 Background

Most existing Web search engines or tools are designed for individual users. It is interesting to see how user collaboration can improve users' performance in Web searching and analysis. Without collaboration, users must start from scratch every time they perform a search task, even if similar searches have been done or relevant search strategies have been identified by other users. In this chapter, a multi-agent approach for collaborative information retrieval and Web mining is proposed. This approach is implemented in a system called *Collaborative Spider*. The main research issues explored include: (a) the impact on users' search and analysis performance and efficiency of the volume of collaborative information available, and (b) different types of user collaboration behavior in the context of Web mining. To my knowledge, this research is the first to develop a Web search system that supports collaboration by sharing complete search sessions based on post-retrieval analyses.

5.2 Related Work

5.2.1 Collaborative Information Retrieval and Collaborative Filtering

In order to alleviate the information overload problem that has resulted from the overwhelming amount of available resources on the Internet, collaborative information retrieval techniques have been proposed and studied in the context of *computer supported cooperative work* (CSCW) that allows multiple users to perform search collaboratively or to share their past search and analysis experiences. Sharing search results, or better,

sharing the data about a whole search sessions, are considered the basic requirements for a collaborative information retrieval system (Karamuftuoglu, 1998).

There are in general two approaches to collaborative information retrieval. The first approach is concerned with situations where several people utilize CSCW tools to support collaboration in the information retrieval process. Users collaboratively search for answers to the same query. Individual findings are then aggregated and unified by the users (Baeza-Yates & Pino, 1997).

The second approach is what has been called *collaborative filtering* or *recommender systems*. Goldberg et al. (1992) defines collaborative filtering as collaboration in which people help one another perform filtering by recording their reactions to Web documents they read. Examples of collaborative filtering and recommender systems include Amazon.com, GroupLens (Konstan et al., 1997), Fab (Balabanovic & Shoham, 1997), Ringo (Shardanand & Maes, 1995), and Do-I-Care (Starr et al., 1996). When a user performs a search, these systems will recommend a set of documents or items that may be of interest based on this user's profile and other users' interests and past actions. For example, Amazon.com uses collaborative filtering to recommend books to potential customers based on the preferences of other customers who have similar interests or purchasing histories. Annotations in free-text or predefined formats are also incorporated in systems such as AntWorld (Kantor et al., 2000), Annotate! (Ginsburg, 1998), and CIRE (Romano et al., 1999) to facilitate collaboration retrieval among users.

Collaborative information retrieval systems can also help users with different backgrounds to share information more effectively. For example, the Worm Community System (Chen, 1994) helps users from different backgrounds to solve the vocabulary difference problem. Shank (1993) also points out that for interdisciplinary participation on the Internet, not only information but also world views are shared among users, creating a perfect environment for knowledge creation.

One of the major issues for collaborative information retrieval system is the users' willingness to share information. Orlikowski (1992) observed that Lotus Notes was not well utilized because workers had little or no incentive to share information. The situation, however, becomes less problematic for Web search, which consists mostly of voluntary contributions (Romano et al., 1999). Users are more willing to contribute in exchange of pride and popularity. In addition, many systems try to minimize extra user effort by capturing user profiles and patterns automatically (Armstrong et al., 1995; Starr et al., 1996).

5.2.2 Multi-agent Systems

As discussed in Section 2.3, software agents have been widely used in Web applications. A multi-agent system is one where a number of agents cooperate and interact with each other in a complex and distributed environment. In a typical multi-agent system, each agent has incomplete information or capabilities. The agents work together to achieve a global objective based on distributed data and control. In most cases, the interaction is

asynchronous and decentralized. Jennings et al. (1998) provide a detailed review of the field.

Multi-agent systems have been developed for a variety of application domains, including electronic commerce, air traffic control, workflow management, transportation systems, and Web applications, among others (Sycara & Zeng, 1996; Sycara, 1998). To enable effective inter-agent communication and coordination, agents that work together have to use an interoperable, platform-independent, and semantically unambiguous communication protocol. The two most widely-used agent communication languages (ACL) are the Knowledge Query and Manipulation Language (KQML) and the FIPA ACL. KQML, developed as part of the ARPA Knowledge Sharing Effort, is a language and protocol for exchanging information and knowledge among software agents. In KQML, each expression is a *speech act* described by a *performative* (Finin et al., 1992; Finin et al., 1994). The FIPA ACL was developed by the Foundation for Intelligent Physical Agents (FIPA). Similarly to KQML, the FIPA ACL is based on speech act theory and the two languages have similar syntax.

5.3 Proposed Approaches

There are two major problems with current Web searching and mining approaches. First, although real-time post-retrieval analysis has been proven effective for Web searching, very few systems perform it. For commercial Web search engines, this kind of analysis can be prohibitively expensive from a computational viewpoint. As discussed in Chapter

3, recent research prototypes are starting to incorporate such analysis capability into client-side Web computing.

Second, for systems that do perform post-retrieval analysis, the analysis is based entirely on individual searches. In these systems, search sessions are not shared by the users. Search strategies, which may have taken a significant amount of time and effort to formulate and test, are lost when the related search session is complete. As a result, users are essentially on their own when they perform search tasks; they re-invent the wheels quite often, denying potentially much improved Web searching and mining experience that could make collaboration among users possible.

5.3.1 Collaborative Spider

To address these problems, the Collaborative Spider system was proposed (Chau et al., 2003). It incorporates post-retrieval analysis and collaboration based on search session sharing. The main research issues explored include (a) the impact on users' search and analysis performance and efficiency of the volume of collaborative information available, and (b) different types of user collaboration behavior in the context of Web mining.

I suggest that a collaborative Web information retrieval and mining environment that performs in-depth post-retrieval analysis leads to improved search effectiveness and efficiency. This idea is embodied in a multi-agent system, called the Collaborative Spider system. The system architecture is shown in Figure 5.1. Collaborative Spider consists of three types of software agents, namely, *User Agent*, *Collaborator Agent*, and *Scheduler*

Agent. In a typical system setup, each individual user will have his/her own personalized User Agent. Each user group (e.g., all participants of a research project or members of a new product design team) will share one Collaborator Agent and one Scheduler Agent. The User Agent is mainly responsible for retrieving pages from the Web, performing post-retrieval analysis, and interacting with the users. The Collaborator Agent facilitates the sharing of information among different User Agents. The Scheduling Agent keeps a list of monitoring tasks and is responsible for carrying out these tasks based on users' schedules.

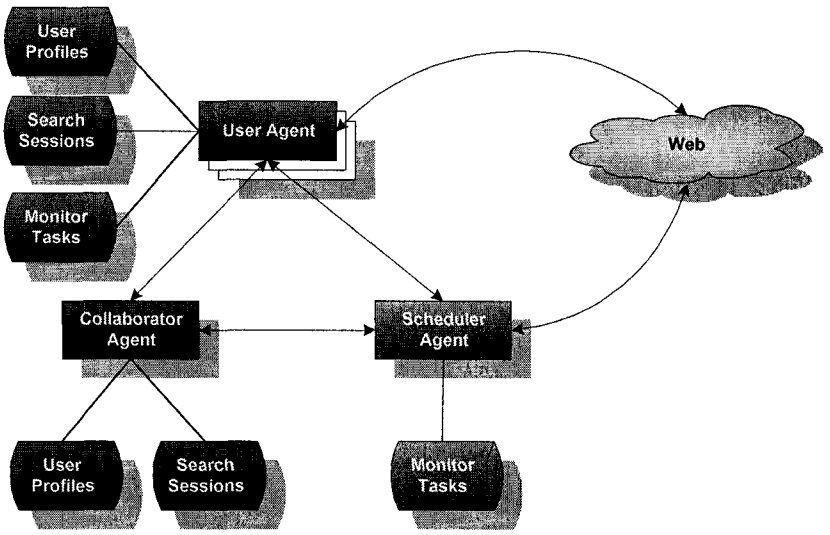

Figure 5.1: Architecture of Collaborative Spider

The proposed architecture differs from traditional information retrieval systems or recommender systems, in that collaboration is based on users' searches and analyses, rather than Web pages rated (Balabanovic & Shoham, 1997), news articles viewed

(Konstan et al., 1997), or items purchased (Amazon.com). The functionalities of each type of agent are discussed in details in the following sections.

5.3.1.1 User Agent

The User Agent is developed based on Competitive Intelligence (CI) Spider discussed in Section 3.3. CI Spider is a personalizable Web search tool designed to help the user search within a given Web site and perform post-retrieval analysis on the set of Web pages retrieved. Technically, the User Agent consists of four main components: *Internet Spiders*, *Arizona Noun Phraser, Self-Organizing Map*, and *Knowledge Dashboard*. The Internet Spiders perform breadth-first search or best-first search on Web sites specified by the user. The Web pages are then fetched to the user machine and passed to the *Arizona Noun Phraser* (AZNP) for further analysis. Developed at the University of Arizona, AZNP extracts and indexes all noun phrases from each document collected by the Internet Spiders based on part-of-speech tagging and linguistic rules (Tolle & Chen, 2000). The noun phrases are then presented to the user, ranked in descending order of occurrence frequencies. If further analysis is desired, the data are aggregated and sent to the *Self-Organizing Map* (SOM) for automatic categorization and visualization. The SOM employs an artificial neural network algorithm to cluster the Web pages automatically into different regions on a 2-dimensional topic map (Chen et al., 1998c). In SOM, more important concepts occupy larger regions and similar concepts are grouped together in a neighborhood (Lin et al., 2000). This provides the user with a convenient and intuitive way to browse the most important topics in the result set.

The main interface component of the user agent is called a *Knowledge Dashboard*, which enables collaborative information search. It shows the information shared among users including detailed search results, analysis, and users' annotations. In addition to browsing past search sessions, the user can also launch new search and Web site monitoring tasks by choosing a combination of any information items and search criteria on the dashboard. More details are discussed in Section 5.4.

5.3.1.2 Scheduler Agent

The Scheduler Agent runs continuously on the background listening for monitoring requests sent by the User Agent. It keeps a complete list of the monitoring tasks for every user and is responsible for carrying out each one of those retrieval and analysis tasks according to a user-given schedule. The Scheduler Agent also performs load-balancing to avoid overloading the same Web server when there are a large number of scheduled tasks using simple heuristics (e.g., prevent launching two scheduled search tasks simultaneously on a same Web site). Whenever a new search session is completed, the Scheduler Agent will store the session and forward it to the corresponding User Agent and Collaborator Agent.

5.3.1.3 Collaborator Agent

The Collaborator Agent is the central piece of Collaborative Spider. It functions as a mediator and regulates the interactions between the User Agents and the Scheduler Agent. It also maintains a collective data repository which stores all user search and monitoring sessions as well as user profiles. To ensure system robustness, each User Agent continues

to store a subset of data associated with its user such that the system will be responsive even if the Collaborator Agent or the Scheduler goes down. In addition to user search sessions and profiles, the Collaborator Agent keeps track of user annotations and comments that can be attached to any documents that the user has browsed or studied. These annotations are accessible to other users.

One of the key functionalities of the Collaborator Agent is to recommend Web documents to potentially interested users, based on profile matching. While different types of recommendation strategies may be used, a simple approach is used in the current implementation. Recommendations are made based on users' areas of interest. When a user performs a search, the search topic needs to be explicitly specified. This search session will then be shared with other users who have selected the corresponding area of interest in their profiles. Section 4 will provide more detailed examples to illustrate how the Collaborator Agent facilitates Web searching and mining experience sharing among multiple users.

5.3.1.4 Data Repository Design

Aimed at simplicity, the system stored all high-level data in plain-text file format. The system follows a simple relationship database design and all the tables followed a certain degree of database normalization. There are three main entities in the data repository, namely *User Profiles*, *Search Sessions*, and *Monitor Tasks*. User Profiles hold information about the users of the system, including name, user id, email, and areas of interest, among other personal information. Search Sessions store the information about

each search session performed. This information includes user id, session id, the area each session belongs to, date and time of the search, whether the search is shared, starting URLs and search terms used and other search options, and the comments and annotations by other users. Monitor Tasks store all the Web site monitoring tasks specified by the users. The data stored include user id, session id, date, and the frequency for revisit. One should note that the lists of information discussed are not exhaustive; they can be easily expanded to accommodate future system enhancement (e.g., to support anonymity).

5.3.1.5 Agent Communication Language

The Knowledge Query and Manipulation Language (KQML) was used as the communication language by the agents in the Collaborative Spider system. JATLite (Java Agent Template, Lite) was used as the KQML implementation platform. Developed at the Center for Design Research at Stanford University, JATLite is a Java implementation of KQML. Agents send and receive ASCII KQML messages in the system through a *message router* using TCP/IP protocol (Petrie, 1996; Jeon et al., 2000). The use of agent templates facilitates agent development by aggregating common functions and services across all agent classes into a few abstract classes.

5.3.2 Sample User Sessions Using Collaborative Spider

Detailed examples are provided in this section to illustrate how a user interacts with Collaborative Spider.

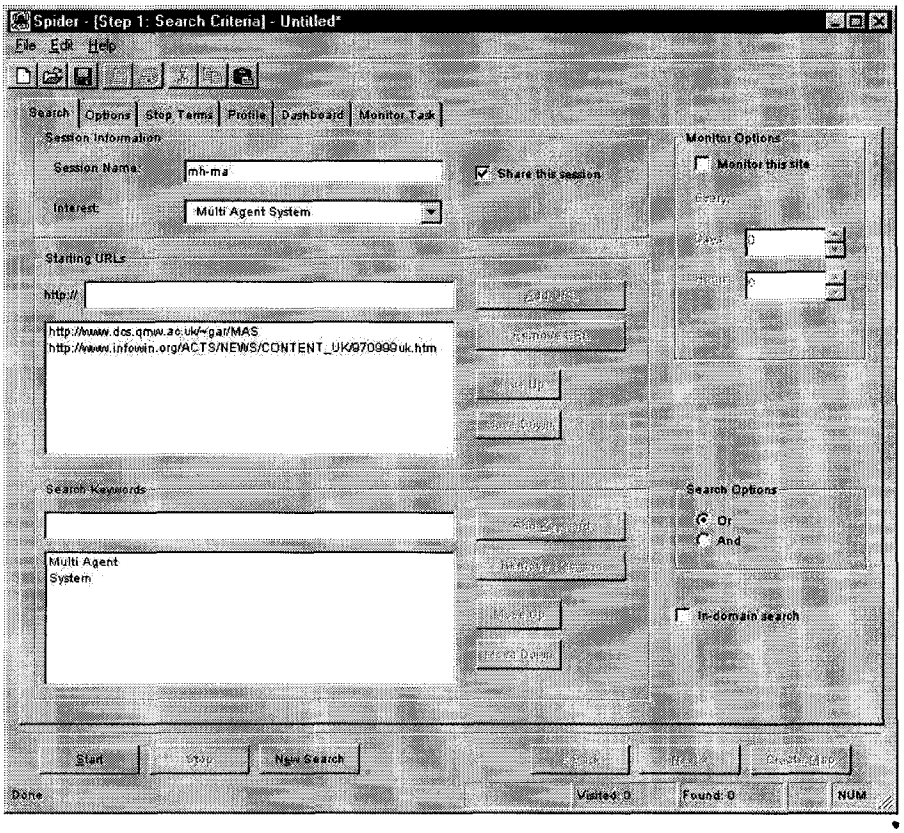

Figure 5.2: Specifying starting URLs, search terms, and sharing options in Collaborative Spider

5.3.2.1 User Registration

First-time users are required to register with Collaborative Spider through a User Agent. The user must select at least one area of interest from the *Areas of Interest* panel. Examples areas of interest include *Information Visualization*, *Expert System*, and *Data Mining*. After receiving the user input, the User Agent updates its local data source as well as the collective data repository managed by the Collaborator Agent.

5.3.2.2 Web Search and Analysis

After user registration and profile specification, the user is ready to perform searches. As shown in Figure 5.2, the user can specify a *session name*, select the proper *areas of*

interest, and have the option of whether to share this session with others who have the same interests. The next step is to add the *starting URLs* and *query terms* such that the User Agent can perform a search based on the given information. The starting URLs specify the Web sites that the user wants to analyze.

The following searching and analysis processes are similar to those of CI Spider. The system searches the starting URLs for the query terms, downloads the pages for user browsing, analyzes the Web pages using the Arizona Noun Phraser, and visualizes the search results by the Self-Organizing Map. Readers are referred to Section 3.3 for more details.

5.3.2.3 Accessing Other Users' Search Sessions

The *Knowledge Dashboard* panel can be used by a user to access other users' search sessions (see Figure 5.3). Search sessions which are in the user's areas of interest will be displayed to the user. In this case, the users *Michael Huang*, *Michael Chau,* and *Daniel Zeng* are all interested in the area *Multi-Agent System.* Should the user decide to use some of the *URLs* or *search terms* from past search sessions, he or she can click on the items preferred (a red check mark will appear) and then press the *Add to Query* button. All the selected items will be added to the search panel along with what the user has already input. The user can also start a brand new search with the help of the collaborative information obtained from the dashboard.

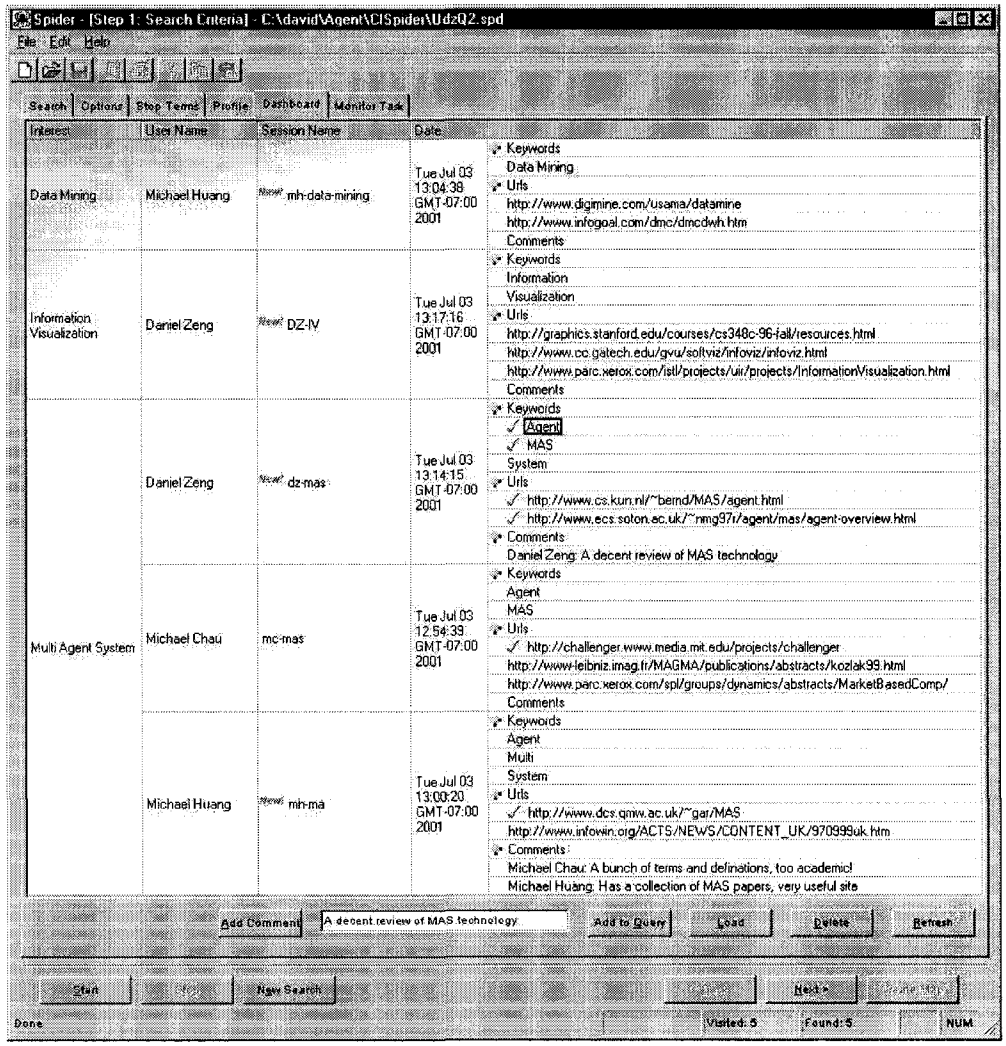

Figure 5.3: Knowledge Dashboard

In addition to using other users' URLs or Keywords, the user can also choose to load a complete session performed by another user from the dashboard. The user can identify the session of interest based on the session names entered by other users. When descriptive name is not available (as in the example), the user can choose the session based on other users' reputation or the keywords and URLs used by them. A search session can be retrieved by highlighting the session of interest and clicking the *Load*

button. As shown in Figure 5.3, the user also has the option to add comments to his or her own session or any other sessions shown on the dashboard.

In the current system, the identity of a user is automatically captured by the system and shown to other users. In future implementation, a user would be allowed to choose whether his/her name should be hidden from other, as some users may prefer to be anonymous.

5.3.2.4 Saving and Sharing Search Sessions

After finishing a search session, the user may save it in a binary file and share it with other users. Saving a search session will trigger the User Agent to send a message along with the real content of the saved session to the Collaborator Agent. The Collaborator Agent will then forward the metadata to all other users interested in the same area. Any user connected (logged on) to Collaborative Spider can immediately view this new session on the knowledge dashboard panel.

5.3.2.5 Monitoring Sessions

The user can request the system to make regular visits to certain Web sites. Such requests are forwarded to the Scheduler Agent. The Scheduler Agent constantly checks the monitor tasks list and looks for outstanding tasks. If an outstanding task is found, the Scheduler Agent activates a number of Internet Spiders to conduct the search. After the search is completed, the Scheduler Agent sends a message to the Collaborator Agent to

inform it of the new search session. The Collaborator Agent in turn forwards the metadata of the search session to all interested users.

5.4 Experimental Design

The main focus of this research is to answer the following questions: (1) Does a collaborative Web information retrieval and mining environment that performs in-depth post-retrieval analysis lead to improved search effectiveness and efficiency? (2) How does the size of shared repository affect a user's Web search and mining effectiveness and efficiency? The evaluation study has been designed to find the answers to these questions.

In order to study the effect of collaboration on Web search and analysis, a user study has been conducted. A key research question being explored is to understand the impact of the size of the shared repository on a user's Web search and mining effectiveness and efficiency. I hypothesize that a user's performance will gradually improve for the first few sessions when more search sessions are made available. Beyond a certain number of search sessions, I hypothesize that the marginal benefit of additional search sessions will approach zero. To test the hypotheses, the User Agent and the Collaborator Agent, but not the Scheduler Agent, were evaluated in the user study.

The general design of the experiment was based on the theme-based evaluation framework previously developed in the evaluation of the CI Spider and the Meta Spider systems as discussed in Chapter 3. Each subject was given several relatively open-ended

information search tasks and asked to summarize search results in the form of a number of themes, rather than to find specific information. In the experiments, a theme was defined as "a short phrase that summarizes a specific aspect of the search results." Within this theme-based framework, protocols were designed to permit evaluation of the extent to which post-retrieval analysis and collaboration facilitate users' identification of major themes related to a certain topic.

Fifty undergraduate students, most of them majoring in management information systems, were recruited in the experiment. Six of the fifty topics used in the Sixth Text Retrieval Conference (TREC-6) ad hoc task were selected and modified for use in the experiment in the context of Web searching (Voorhees & Harman, 1998). Each subject was given 3 search topics and asked to perform search and analysis to identify the major themes related to each of them.

The six topics used in the experiments were:

1. Hubble telescope achievement
2. Implant dentistry
3. Radio waves and brain cancer
4. Undersea fiber optic cable
5. New fuel sources
6. Health and computer terminals

For each search task, a subject could choose one or more search strategies from the following list to perform the analysis using the Collaborative Spider system:

1. The subject could start a completely new search by entering new starting URLs and search terms, and analyze the search results. The subject could obtain the starting URLs by any search engine of his or her choice.
2. The subject could start a new search by using a combination of URLs and search terms from the knowledge dashboard, plus the subject's own starting URLs (obtained from any preferred search engine) and search terms.
3. The subject could start a new search by using URLs and search terms from the knowledge dashboard if available. These URLs and search terms will be those previously used by other subjects.
4. The subject could browse the search sessions performed by other subjects (if available) and come up with the findings without actually doing a Web search.

The 50 subjects were equally divided into 5 groups. Each group had a different number of previous sessions made available to them. Each subject in Group 0 had no access to search sessions performed by other subjects and was required to perform the search individually. A subject in Group 1 had access to 1 previous search session for the same topic by another subject for each search topic. These were search sessions performed by subjects in Group 0. Similarly, subjects in Group 2 had access to 2 search sessions performed and saved by subjects in Group 0 and Group 1, and subjects in Group 3 had access to 3 sessions, etc. Based on a rotation scheme, half of the subjects within each

group were given topics 1, 2, and 3 and half of the subjects were given topics 4, 5, and 6. This allowed us to control the amount of collaborative information available and measure its effect on the efficiency and effectiveness on Web search and analysis. (An alternative design would be to assign each subject to different groups for different search tasks. Although such design might decrease the effect of subject bias on the data, it was not chosen because it would make it much more difficult to control the experiment environment). Some sample documents used in the experiment are attached in Appendix D.

5.4.1 Performance Measures

Two graduate students majoring in library science were recruited as expert judges in the experiment. The expert judges individually performed extensive searches on the 6 search topics and summarized their findings into topic themes. Their results were then condensed and combined to form the basis for evaluation. Precision and recall rates for the number of themes were used to measure the effectiveness of each search performed and were calculated as follows:

$$precision = \frac{number\ of\ correct\ themes\ identified\ by\ the\ subject}{total\ number\ of\ themes\ identified\ by\ the\ subject}$$

$$recall = \frac{number\ of\ correct\ themes\ identified\ by\ the\ subject}{total\ number\ of\ themes\ identified\ by\ expert\ judges}$$

A well-accepted single measure that tries to balance recall and precision called F-measure was also used in the evaluation and calculated as follows (van Rijsbergen, 1979):

$$F\text{-}measure = \frac{recall * precision}{(recall + precision)/2}$$

The F-measure value was calculated for each search session and the average value was used for each group.

The amount of time each subject spent for each search topic was recorded. During the experiment, the subjects were encouraged to tell us about the search method used and their comments were recorded. The experimenter also closely observed each test subject during the experiment and filled out an observation log to record the user actions. At the end of the experiment, each subject filled out a questionnaire to offer further comments on the search system and the search strategies used.

5.5 Experimental Results and Discussions

5.5.1 Quantitative Results

The average times spent on the set of the search tasks by the subjects in different groups are summarized in Table 5.1. No significant difference in total search time was observed among the groups. The result shows that the subjects were not able to reduce the total time by browsing or using other users' sessions. By looking at the amount and percentage of time spent on each particular type of search and mining activity, it was found that subjects across the groups spent comparable amounts of time on these activities, except for Group 0, to which other users' sessions were not available.

Table 5.1: Average time spent on each search task

Group	0	1	2	3	4
Total search time (minutes)	16.1 (100%)	15.3 (100%)	15.8 (100%)	16.6 (100%)	16.3 (100%)
Time spent for each subtask (minutes)					
Getting URLs from search engines	6.8 (42.2%)	4.3 (28.2%)	5.4 (34.3%)	5.6 (34.0%)	5.5 (33.7%)
Browsing metadata on the dashboard	0.0 (0.0%)	1.2 (7.7%)	1.2 (7.4%)	1.3 (7.6%)	1.8 (10.8%)
Browsing other users' session(s)	0.0 (0.0%)	1.3 (8.3%)	1.0 (6.1%)	0.5 (3.6%)	1.7 (10.6%)
Performing own search and analysis	9.3 (57.8%)	8.5 (55.8%)	8.3 (52.2%)	9.1 (55.2%)	7.3 (44.9%)

The results on average precision, recall, and F-measures are summarized in Table 5.2. Each group represents a sample size of 30 subject-task combinations. A corresponding chart is shown in Figure 5.4. Group 0 (where subjects had no access to any collaborative information) was used as a reference group for comparison with other groups. A series of *t*-tests were conducted, the results were shown in Table 5.3.

Table 5.2: Analysis of effectiveness on different groups

Group	0	1	2	3	4
Sample Size	30	30	30	30	30
Average Precision	0.474	0.462	0.503	0.557	0.569
Average Recall	0.243	0.217	0.211	0.280	0.312
Average F-measure	0.299	0.275	0.282	0.355	0.387

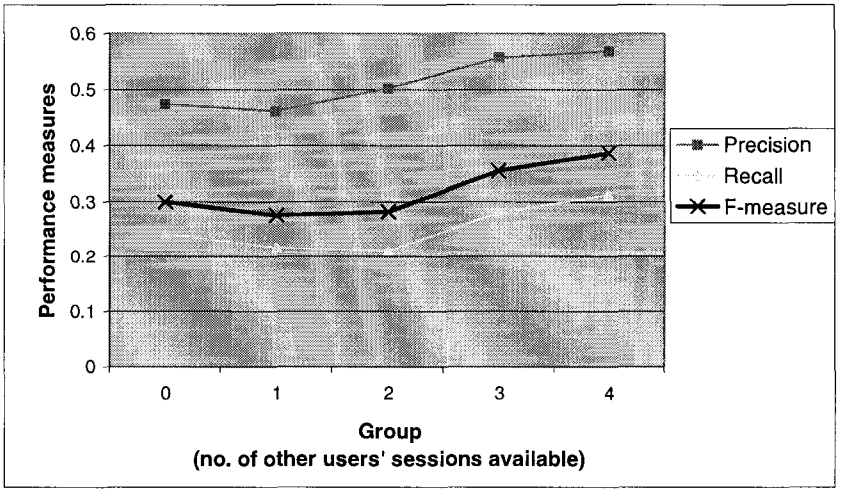

Figure 5.4: Performance measures vs. number of other users' sessions available

Table 5.3: *p*-values of *t*-tests on groups' performances

	Group 0 vs. Group 1	Group 0 vs. Group 2	Group 0 vs. Group 3	Group 0 vs. Group 4
Precision	0.859	0.752	0.268	0.150
Recall	0.527	0.487	0.362	*0.078
F-measure	0.569	0.750	0.246	*0.070

*The difference is statistically significant at the 10% level.

Comparing the performance of Group 0 and Group 4, it was found that the average precision, recall, and F-measure climbed to 0.569, 0.312, 0.387 in Group 4 where subjects had most collaborative information, from 0.474, 0.243, 0.299 in Group 0 where subjects had no collaboration at all, respectively (see Table 5.2). From the *t*-test results in Table 5.3, it can be seen that the differences between the recall rates and the F-measures of the two groups are statistically significant at the 10% level, indicating that the collaboration provided by the system was able to improve users' search performance.

In order to study the effect of size of the shared repository on search performance, I analyzed the performance of each group in details. In Figure 5.4, the three leftmost data points indicate the performance of subjects who had no access to any other subjects' search sessions (Group 0). Search and analysis performance started to decline for subjects having access to one other user's search session (Group 1) and for those having access to two other users' search sessions (Group 2). As shown in Table 5.3, the precision, recall, and F-measure of these two groups were not statistically different from those of Group 0. The results demonstrate that subjects were unable to gain improvement from having access to 1 or 2 search sessions by other users. It was observed that when subjects had only a little amount of collaborative information, the normally expected improvement in

performance did not occur. It was believed that this was because the amount of information available was so small (only 1 or 2 search sessions with a few URLs and search terms) that the subjects were not able to benefit considerably from the collaboration. The efforts and attentions that the subjects had spent on collaboration counteracted any performance gain from the limited amount of collaborative information. Sometimes, the subjects might even have been distracted or confused by this information. As a result, in Groups 1 and 2, the performance actually dropped slightly below that of Group 0, where subjects had no collaborative information at all.

Proceeding from Group 2 to Group 3, it can be seen that there were notable improvements in all the three performance measures, and the search performance further increased and reached a plateau for Group 4. This may be explained by the fact that subjects started to benefit from other users' searches when they had access to 3 or more sessions. Previous sessions could be compared and benefits derived from the best of them, including help in deciding which sessions to explore, based on the annotations made by other subjects. I suggest that at this point the benefit of accessing other users' sessions started to outweigh the overhead cost of time and effort spent reading other users' sessions. Nevertheless, it is suspected that after a certain threshold, the marginal benefit of having more past search sessions available will diminish.

5.5.2 Collaboration Behavior

In this section, I create a taxonomy of various types of collaboration behavior based on the search strategies observed during the experiment. Some qualitative data and technical insights obtained from the post-search questionnaires are also discussed.

Because the subjects in Group 0 did not have access to other users' search sessions, I focused the analysis on the 120 cases (10 subjects from each group from Group 1 to Group 4, each subject performing 3 search queries) in which subjects could browse or load other users' sessions. It was observed that most subjects did not attempt to rely solely on the search sessions available; they tried to combine other users' URLs and keywords with their own and launch new searches. In fact, of the 120 cases, in only 4 cases (1.7%) did the subject simply load other users' search sessions and come up with findings without actually performing a search. There were 34 cases (28.3%) in which subjects combined findings in their own searches and other users' search. In 57 cases (47.5%), subjects combined other users' starting URLs and search terms and launched new searches, using only the other users' starting points but not their findings. This method was found to be the one most commonly used in the experiment. It is interesting to note that although the subjects realized the value of sharing and collaboration and gained from obtaining some useful information from other users, they preferred to perform their own search sessions and draw their own conclusions. In the remaining 25 cases (20.8%), subjects performed their own searches without using any of the information available from other search sessions. To summarize, I categorize the types of collaboration behavior observed in the experiment as follows:

A. Use own starting points (URLs and search terms) and perform new analysis.

B. Combine other users' starting points with own starting points and perform new analysis.

C. Use starting points of other users and perform new analysis.

D. Use other users' starting points and their analysis.

The performance for each type of collaboration are summarized and the results are summarized in Table 5.4.

Table 5.4: Performance analysis of different types of collaboration behavior

Type	A	B	C	D
Total number of cases	25 (20.8%)	57 (47.5%)	34 (28.3%)	4 (1.7%)
Number of subjects who considered this behavior as most efficient	6 (15.0%)	26 (65.0%)	7 (17.5%)	1 (2.5%)
Precision	0.522	0.486	0.516	0.321
Recall	0.268	0.228	0.252	0.163
F-measure	0.334	0.294	0.321	0.215

From Table 5.4, it was found that the performance measures were comparable for Types A, B and C (the performance of Type D was not considered given the small sample size). However, Type B, i.e., using a combination of their own starting points (URLs and search terms) and those of other users, was the most preferred search strategy. This strategy is in fact similar to the follow-up searches triggered by search results as discussed by O'Day and Jeffries (1993). Such follow-up searches often intend to probe

more deeply in the same topic area. I believe that the use of other users' starting points reassured the subject that he or she was not off-track in performing the search. At the same time, subjects also liked to add their own starting points rather than relying entirely on other users' findings. These preferences provide a productive balance between having individual control over the search task and gaining help from other users.

5.6 Conclusion

In this chapter I present my work on the design and evaluation of a multi-agent collaborative Web mining system. The main contribution of this research is to develop and evaluate the first Web search system that supports collaboration by sharing complete search sessions based on post-retrieval analysis. I demonstrated the feasibility of using a multi-agent architecture to build a collaborative Web searching and mining environment with session sharing. An initial evaluation study was designed and conducted to investigate the correlation between search performance and the amount of collaborative information available. The study showed that subjects' performance was slightly degraded from those of individual search situations when they had access to 1 or 2 other users' search sessions but improved significantly when subjects had access to 3 or more search sessions of other users. I suggest that having 3 or more sessions available is the point at which the gain from collaboration outweighed the overhead of browsing and comprehending other users' sessions. In summary, it is likely that the system's *point of insufficiency* have been found, at which less than that the amount of collaborative information will be considered insufficient to provide the performance gain that will exceed the overhead expended to acquire it. However, the *point of sufficiency*, beyond

which performance gain from collaboration will become stable even if more collaborative information is available, was not found in the study. It will be interesting to find out if the performance continuously improves as the number of previous sessions increased beyond 4, or the performance will reach an optimal point.

I also cataloged the different types of collaboration behavior observed in the user study. It was noticed that a large proportion of the subjects enjoyed using some starting Web sources obtained from other users' past searches as guidance but retaining control over their own search process and drawing their own conclusions. Nevertheless, the different behaviors did not have a considerable impact on subjects' performance.

The nature of the tasks the subjects performed may have had a substantial impact on their collaborative behavior and effectiveness. I am currently planning a user study in which the subjects will be asked to search for specific answers to well-defined questions, as opposed to the open-ended soft search queries discussed in this Chapter. Because subjects can probably gain from easy access to concrete answers obtained by other users, it will be interesting to examine whether users will demonstrate collaboration behavior different from what was observed in this experiment.

Another future research plan is to perform data mining on user search activities such that user profiles can be learned automatically. Currently, users have to explicitly specify their areas of interest in order to access shared search sessions. It will be interesting to enhance

the Collaborative Spider system with more sophisticated content-based or collaborative-based information recommendation functionalities using data mining algorithms.

Although the findings of the study and evaluation methodology may not be applicable to collaboration systems in general, it is believed that the experimental results are interesting and useful for related research, and that the research issues identified should be further studied in other collaborative environments.

CHAPTER 6: CREATING VERTICAL SEARCH ENGINES USING SPREADING ACTIVATION

6.1 Background

Besides the use of various types of search agents described in Chapters 3, 4, and 5, specialized, domain-specific search engines are another approach to effective searching on the Web. Many specialized search engines have been developed to address the problem by keeping search indexes only in particular domains. Vertical search engines allow users to perform searches in particular domains and they usually provide customized features. However, it is also difficult to build these search engines as it is difficult to locate relevant and high-quality Web pages. Finding a way to extract from two billion Web pages a subset of high-quality ones relevant to a desired domain becomes a great challenge. During the collection of Web pages, an intelligent search engine spider (search agent) should "predict" whether a URL pointed to a relevant Web page, before actually fetching the page. In addition, the spider should first visit pages with higher probability of having relevant content to improve efficiency, and it should determine the quality and reputation of pages to avoid irrelevant or bad quality ones. This Chapter investigated the issue of how good search and analysis algorithms can improve the performance of these spiders and proposes new approaches to address the problem.

6.2 Related Work

6.2.1 Search Engine Spiders

The spiders behind most search engines collect Web pages by using simple algorithms such as breadth-first search. Without control, spiders can possibly fetch Web pages from any field. There are two popular ways to control the quality of a spider.

- The spiders can be restricted to stay in particular Web domains, because many Web domains have specialized contents (e.g., most Web pages within the domain http://www.toyota.com would be relevant to automobiles).
- The collected pages can be processed by small filtering programs that decide whether to remove a page from a collection based on its content (e.g., the number of relevant keywords contained, or the level of similarity to particular example pages).

Both approaches have some disadvantages, however. One problem with the first approach is that it cannot discover potentially relevant Web sites that are not within the original list. It also does not work for Web sites having more diverse contents. On the other hand, the second approach has the problem of inefficiency. Without a good spidering algorithm, it is possible that more than half of the pages collected are irrelevant. It would be much more efficient if a spider could predict whether a page is desired before downloading it to the local database.

6.2.2 Graph Search Algorithms

Traditional graph search algorithms have been extensively studied in the field of computer science. Since most researchers view the Web as a directed graph with a set of nodes (pages) connected with directed edges (hyperlinks), some of these algorithms have been applied in Web applications. In this section, I review three categories of graph search algorithms that are relevant to the study, namely, (1) uninformed search, (2) informed search, and (3) parallel search.

6.2.2.1 Uninformed Search

The first category of graph search algorithms consists of simple algorithms such as breadth-first search and depth-first search. They are also known as *uninformed search* as they do not make use of any information to guide the search process. Breadth-first search, one of the most popular methods used in Web search spiders, collects all pages on the current level before proceeding to the next level. Although these algorithms are easy to implement and use in different applications, they are usually not very efficient because of their simplicity.

6.2.2.2 Informed Search

The second category is *informed search*, in which some information about each search node is available during the search process. Such information is used as the heuristics to guide the search. Best-first search is one example that is widely used. Best-first search explores the most promising node at each step. This class of algorithms has been studied in different search engine spiders or search agent systems with different variations (Chen et al., 1998a; Cho et al., 1998). Different metrics, such as number of in-links, PageRank

score, keyword frequency, and similarity to search query, have been used as guiding heuristics.

6.2.2.3 Parallel Search

Another category is *parallel search.* Algorithms in this category try to explore different parts of a search space in parallel. One example is the spreading activation algorithm used in artificial neural network models, which tries to achieve human-like performance by modeling the human nervous systems. A neural network is a graph of many active nodes (neurons) that are connected with each other by weighted links (synapses). Neural network uses activations over the nodes to represent and retrieve concepts and knowledge (Kwok, 1989; Chen & Ng, 1995). Although these algorithms are powerful and extensively used in other applications, they have not been widely applied in Web applications.

6.3 Proposed Approaches

As discussed earlier, traditional search engine spiders have a few problems. Most existing approaches are not systematic and not very efficient. While simple graph search algorithms such as breadth-first search and best-first search are widely used in spiders, there is little research on using more powerful search algorithms in spider applications.

Web content and link analysis techniques provide good heuristics in the spidering process. However, most of these techniques, such as PageRank and HITS, are computationally expensive. Also, link-based analysis and content-based analysis usually are not effectively combined. Even when the two are combined, only a driving search query or

search topic is used (e.g., Bharat & Henzinger, 1998; Cho et al., 1998). Domain knowledge should be applied in a better and more effective way.

Based on my review, I believe that these two classes of techniques can be integrated more effectively. In this study, I pose the following research questions:

- How can content-based analysis and link-based analysis be combined effectively?
- How can existing graph search algorithms and Web analysis techniques be integrated to create a collection of Web pages of high relevance and good quality for a vertical search engine?

Aimed at combining different Web content and structure analysis techniques with traditional graph search techniques to build spider programs for vertical search engines, I developed and compared three versions of Web spiders, namely, (1) Breadth-First Search Spider, (2) PageRank Spider, and (3) Hopfield Net Spider. In this section, I describe the designs and approaches adopted in the study.

6.3.1 Breadth-First Search Spider

The Breadth-First Search Spider (or BFS Spider) follows a basic breadth-first search algorithm which has been widely used by commercial search engines. The intuition behind it is that if a URL is relevant to a target domain, it is likely that the Web pages in its neighborhood are also relevant. It has been shown that breadth-first search can discover high-quality pages early on in a spidering process. As the most important pages

have many links to them from numerous hosts, those links usually can be found at an early stage in the process (Najork & Wiener, 2001).

6.3.2 PageRank Spider

The PageRank Spider was adapted from the algorithm reported by Cho et al. (1998). Aiming to combine link-based analysis and a heuristics-based traversal algorithm, it is designed to perform best-first search using PageRank (as described earlier) as the heuristics. URLs with higher PageRank scores would be visited earlier.

In each step, the spider gets the URL with the highest PageRank score, fetches the content, and extracts and enqueues all the outgoing links in the page. It runs until the required number of pages have been collected. The PageRank score is calculated iteratively using the algorithm described above until convergence is reached. The damping factor d is set to 0.90 in the implementation. The Hot Queue approach used in the original study also has been adopted for anchor text analysis in the PageRank Spider. Two priority queues are established: hot_queue and normal_queue. The URLs within each queue are ordered by PageRank score in descending order. The spider first dequeues from the hot_queue. If the hot_queue is empty, the spider dequeues from the normal_queue. In the adopted design, a URL will be placed in the hot_queue if the anchor text pointing to this URL contains a relevant term.

6.3.3 Hopfield Net Spider

The Web can be viewed as a large network structure of massive, distributed knowledge composed of pages and hyperlinks, contributed by all Web page authors. This can be viewed as a neural network — a graph of many active nodes (neurons) that are connected with each other by weighted links (synapses). Neural network uses activations over the nodes to represent and retrieve concepts and knowledge (Kwok, 1989; Chen & Ng, 1995). In this approach the Web is modeled as a Hopfield Net, which is a single-layered, weighted neural network (Hopfield, 1982). Nodes are represented by pages and links are simply represented by hyperlinks. Nodes are activated in parallel and activation values from different sources are combined for each individual node until the activation scores of nodes on the network reach a stable state (convergence).

Based on this spreading activation algorithm, which was shown to be effective for knowledge retrieval and discovery in a Hopfield Net, I developed the Hopfield Net Spider to perform searching on this network (Chau & Chen, 2003). The aim of this approach is to combine a parallel search algorithm with content-based and link-based analysis. My implementation incorporated the basic Hopfield Net spreading activation idea, but significant modification was made to take into consideration the unique characteristics of the Web.

The Hopfield Net Spider starts with a set of seed URLs represented as nodes, activates neighboring URLs, combines weighted links, and determines the weights of newly

discovered nodes. The process repeats until the required number of URLs have been visited. The algorithm adopted is as follows:

1. Initialization with Seed URLs

An initial set of seed URLs is given to the system and each of them is represented as a node with a weight of 1. $\mu_i(t)$ is defined as the weight of node i at iteration t.

$$\mu_i(0) = 1, \text{ for all seed URLs } i$$

The spider fetches and analyzes these seed Web pages in iteration 0. The new URLs found in these pages are added to the network.

2. Activation, Weight Computation, and Iteration

Proceeding to the next iteration, the weight of each node is calculated as follows:

$$\mu_i(t+1) = f_s\left(\sum_{\substack{\text{every known} \\ \text{parent } h \text{ of } i}} w_{h,i}\mu_h(t)\right)$$

where $w_{h,i}$ is the weight of the link between two nodes and f_s is the SIGMOID transformation function that normalized the weight to a value between 0 and 1.

$w_{h,i}$ estimates whether a URL i pointed to from a Web page h is relevant to the target domain, based on the use of anchor text. This score can be calculated based on a function

of the number of words relevant to the target domain used in the anchor text of the hyperlink used in a Web page *h*.

A slightly modified SIGMOID function was adopted as follows:

$$f_s(x) = \left(\frac{1}{1+e^{-x}} - 0.5\right) \times 2$$

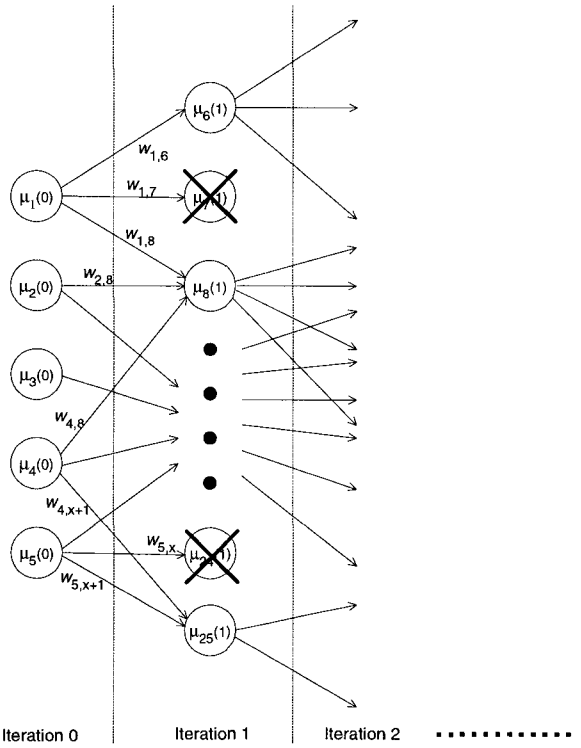

Figure 6.1: Spreading activation: Starting with a set of seed URLs, the Hopfield Net Spider activates neighbor URLs, combines weighted links, and determines the weights of newly discovered nodes. Nodes with a low weight (e.g., node 7 and node 24) are discarded.

After the weights of all the nodes in the current iteration are calculated, the set of nodes (URLs) in the current iteration are then activated (visited) and fetched from the Web in descending order of weight. In order to filter out low-quality URLs, nodes with a weight smaller than a threshold θ are not visited. The activation process is illustrated in Figure 6.1.

After all the pages with a weight > θ have been visited and downloaded, the weight of each node in the new iteration is updated to reflect the quality and relevance of the downloaded page content as follows:

$$\mu_i(t+1) = f_s[\mu_i(t+1) \times p_i]$$

where p_i is a weight that represents the relevance of the textual content of a page i. This score is a function of the number of phrases contained in a page's content that are relevant to the target domain. A page with more relevant phrases will receive a higher score.

3. Stopping Condition

The above process is repeated until the required number of Web pages have been collected or until the average weight of all nodes in an iteration is smaller than a maximum allowable error (a small number).

6.4 Experimental Design

In order to evaluate and compare the performances of the three proposed approaches, they were implemented as the backend spiders for a medical search engine called HelpfulMed (http://ai.bpa.arizona.edu/helpfulmed/) (Chen et al., 2003). Medical Web pages are highly distributed, of varying quality, and difficult to locate. It is important for a large number of users working in the medical industry to find important and high-quality information on the Web in order to make potentially significant medical-related decisions. It is also especially important to distinguish between Web pages of good and poor quality, as an online international journal article is likely to be more authoritative than a company's product Web site. Designed to address these issues by collecting high-quality medical Web pages and providing them to users, HelpfulMed provided an ideal testbed for the spiders in the medical domain.

6.4.1 Domain Knowledge

To facilitate content-based analysis, a set of *domain knowledge* for the medical domain was developed. The Unified Medical Language System (UMLS) was used to develop a medical lexicon, called the *Good Phrase List*. UMLS is developed under a long-term research and development effort by the National Library of Medicine, aiming to provide medical professionals and researchers with various medical knowledge sources. To develop the lexicon, a medical librarian reviewed the semantic types of the UMLS Metathesaurus and extracted the best terms from the entire collection. After careful selection and filtering, about 300,000 medical phrases were extracted. The medical librarian also manually compiled a *Bad Phrase List* that contains 118 words that

frequently appear in unwanted, non-medical Web pages (e.g., "Job Posting" and "Contact Us").

The medical librarian also identified a set of 354 Web sites that were either good hubs or good authorities in the medical domain. For example, it is believed that the information and data contained in the Web site of the U.S. National Library of Medicine (http://www.nlm.nih.gov/) should be highly credible, so it was included in the list. From the list, the medical librarian further identified 5 high-quality hub pages as *seed URLs*, which served as the starting points in the experiment.

6.4.2 Creating the Testbed

In order to prevent variations in network load and traffic from affecting the systems' performance, I set up a controlled environment for the experiments by creating a local repository of a portion of the Web relevant to the study, similar to the design described in Cho et al. (1998). The spider systems could then perform *virtual spidering* on the local repository, meaning that when a spider needed to fetch the content of a page, it would obtain it from the local repository rather than from the Web. This ensured that the experiments were not affected by changes in actual Web pages or fluctuations in network traffic and servers' response time.

The repository was created by running a *random-first search*. The 5 seed URLs identified by the medical expert were used as starting points and then all new links were fetched in a random order. The resulting testbed consisted of 1,040,388 valid, unique Web pages

and 6,904,026 links. The repository contained pages both inside and outside the starting Web sites as well as both medical and non-medical Web pages, allowing us to test the performances of the spiders. I also ran the Arizona Noun Phraser (Tolle & Chen, 2000) to extract noun phrases from each page in the testbed, and calculated the total number of phrases and the number of phrases that appeared in the Good Phrase List for each page.

6.4.3 The Experiments

After the testbed had been created, I designed and conducted two experiments to compare the three spiders. The first experiment was a simulation to analyze the spidering processes and their speed and quality. The second experiment was designed to study how domain experts rated the Web pages collected by each system.

6.4.3.1 Simulation

In the first experiment, each spider was executed to perform *virtual spidering* on the testbed. Although the local repository contained information of all the pages, each spider could access only information based on pages it already had visited. The same set of seed URLs used to create the testbed were used as starting points for each spider, which ran until 100,000 Web pages had been visited. (To ensure that each of the three spiders collected the same number of pages, I did not make use of the convergence property of the Hopfield Net Spider in the experiment.)

To compare the performances of the three spiders, the quality of each Web page visited had to be determined. I introduced the notion of *Good Page*, which estimated a Web

page's relevance to the medical domain automatically. Based on a previous experiment on a similar but smaller collection (with about 100,000 Web pages), Web page was considered as a *Good Page* if the number of medical phrases divided by the total number of phrases found in the page was greater than a certain threshold. In the previous experiment, a set of randomly-sampled Web pages were classified by this method, and the error rate of this simple classification method was 5.0% for the medical domain. Using this classification, the testbed in the current experiment contained 171,405 Good Pages.

Using the notion of Good Page, the precision and recall rates of the spiders were defined as follows:

$$Precision\ rate = \frac{number\ of\ Good\ Pages\ visited\ by\ the\ spider}{number\ of\ all\ pages\ visited\ by\ the\ spider}$$

$$Recall\ rate = \frac{number\ of\ Good\ Pages\ visited\ by\ the\ spider}{number\ of\ Good\ Pages\ in\ the\ testbed}$$

Because the total number of Good Pages in the test bed was 171,405 and the total number of all pages visited by a spider was fixed at 100,000, the precision rate and the recall rate were directly proportional to each other for each spider. In the remainder of this Chapter, I focus my discussion on the precision rate. The time used by each spider was also measured to compare efficiency.

6.4.3.2 User Study

In addition to the simulation, a user study was conducted in which two senior graduate students with medical training were recruited to judge the quality of the pages collected by each spider. Each expert was assigned 100 Web pages randomly drawn from the collection of pages fetched by each spider during the simulation. The source of each Web page was not disclosed to the experts in order to eliminate any possible bias. The experts were asked to judge independently each page's quality and relevance to the medical domain. They were asked to give a score in the range of 1 (lowest) to 4 (highest) to each page.

6.5 Experimental Results and Discussions

6.5.1 Experimental Results of the Simulation

The results of the simulation experiment are summarized in Table 6.1. The results in Table 6.1 show that the Hopfield Net Spider retrieved 40,014 Web pages (40.0% of all pages visited), compared with 36,307 (36.3%) by BFS Spider and 19,630 (19.6%) by PageRank Spider. In terms of time, BFS Spider took 12.7 minutes, Hopfield Net Spider 12.6 minutes, and PageRank Spider a significantly slower 1183.6 minutes.

Table 6.1: Summary of simulation results

	Total no. of pages visited	No. of Good Pages visited	Precision	Time (minutes)
BFS Spider	100,000	36,307	36.3%	12.7
PageRank Spider	100,000	19,630	19.6%	1183.6
Hopfield Net Spider	100,000	40,014	40.0%	12.6

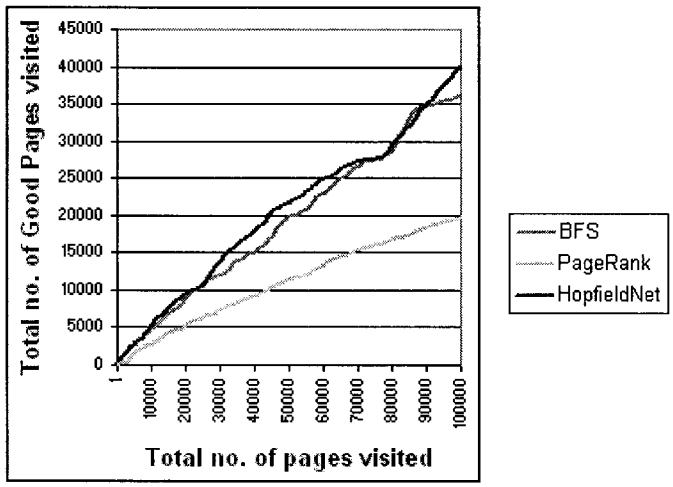

Figure 6.2: Total number of Good Pages visited

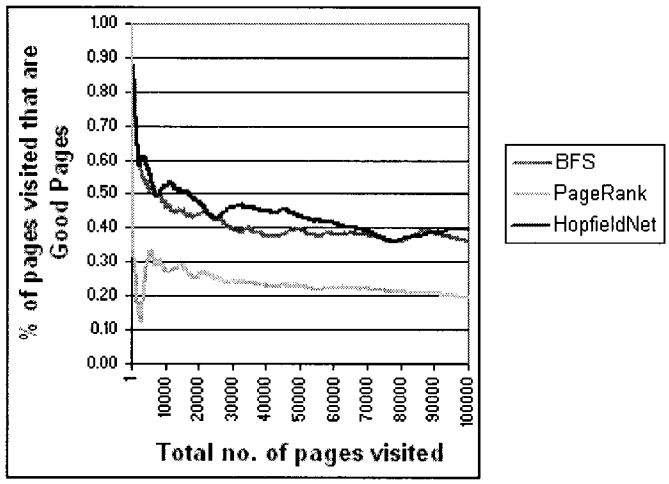

Figure 6.3: Percentage of pages visited that are Good Pages

In addition to the final collection, the performance of the spiders during different stages of the process was also studied. Figures 6.2 and 6.3 show the total number and the percentage of Good Pages during the spidering process for each system. It can be seen that the Hopfield Net Spider consistently achieved the best performance during the process. The BFS Spider was slightly less effective than the Hopfield Net Spider, and the PageRank Spider had a considerably lower performance level.

To further analyze the data, the 100,000 pages visited by each spider were divided into 1,000 equal portions, each containing 100 consecutive pages according to the original visiting order of each spider. Within each portion, the percentage of Good Pages (precision) was calculated. As there were 100,000 pages, 1,000 data points were obtained for each spider. Paired *t*-tests were conducted on these data and the results are summarized in Table 6.2.

Table 6.2: *t*-tests on simulation results

Comparison	*p*-value
BFS Spider vs PageRank Spider	< 0.0001*
BFS Spider vs Hopfield Net Spider	0.0021*
PageRank Spider vs Hopfield Net Spider	< 0.0001*

*The difference is statistically significant at the 1% level.

The *t*-test results show that the Hopfield Net Spider had significantly better precision than both the BFS Spider and the PageRank Spider at the 1% level. The BFS Spider also did significantly better than the PageRank Spider at the 1% level.

In terms of precision rate, the Hopfield Net Spider performed best, followed by the BFS Spider. The PageRank Spider had the worst performance. The promising results of the Hopfield Net Spider show that the algorithm effectively combined the use of Web link structure analysis and page content analysis to locate Web pages relevant to the medical domain. The fact that bad pages were filtered out also increased the precision rate of the spider.

The performance of the PageRank Spider was rather unexpected because it had been anticipated to perform at least as well as the BFS Spider, which did not use any heuristics.

After analyzing the data in detail, it was found that the PageRank Spider visited more irrelevant URLs than the other two spiders during the early stage of the search (the first 3,000 pages in Figure 6.3), because those URLs had high PageRank scores. Such pages tended to point to other pages that were also irrelevant, contributing to the low performance level of the PageRank Spider. One prominent example is the Adobe Web site. Many of the first few hundred pages visited by the PageRank Spider contained a link to the home page of the Adobe Web site (http://www.adobe.com/) which provided information on how to open and read PDF files. Because of the large number of referring pages, the abovementioned URL obtained a high PageRank score and would be visited by the spider at an early stage of the spidering process. After this page had been visited, such high score was propagated to other URLs contained in the page because of the recursive nature of the PageRank algorithm. As a result, a large number of irrelevant pages were visited. It is believed that although the PageRank algorithm is robust for large collections of pages, its scores can be misleading for small collections, especially during the early stage of a spidering process. On the other hand, the Hopfield Net Spider did not suffer from such problem because it incorporated domain-specific content analysis into the scoring function. The BFS Spider also was able to obtain high-quality pages simply by visiting the URLs close to the starting URLs, which tended to point to pages that were also relevant.

For execution time, I found that the PageRank Spider spent much more time than the other 2 spiders, due to its heavy computational requirements. It has been shown that the PageRank algorithm took several hours to calculate the scores of a set of 19 million pages

because of its recursive nature (Haveliwala, 1999). For the PageRank Spider, the problem was worse, as the PageRank scores were not calculated only once but had to be recalculated whenever any new URLs were found during the spidering process. Consequently, the PageRank Spider became exponentially slower as the number of visited URLs increased. While PageRank is a good measure for ranking Web search results, it would be impractical for use in the spidering process for large collections.

6.5.2 Experimental Results of the User Study

The results of the user study are summarized in Table 6.3. The results agreed with those of the simulation experiment in general. The Hopfield Net Spider had the highest relevance score (2.30) among the three. The BFS Spider score was 2.13 and that of the PageRank Spider was 1.78. *t*-tests were also performed and the results were shown in Table 6.4. The Hopfield Net Spider and the BFS Spider both scored significantly better than the PageRank Spider at the 5% level. The Hopfield Net Spider also had a higher relevance score than the BFS Spider, but the difference was not statistically confirmed in the experiment.

Table 6.3: User study results

	Average relevance score
BFS Spider	2.13
PageRank Spider	1.78
Hopfield Net Spider	2.30

Table 6.4: *t*-tests on user study results

Comparison	*p*-value
BFS Spider vs PageRank Spider	0.0307*
BFS Spider vs Hopfield Net Spider	0.3127
PageRank Spider vs Hopfield Net Spider	0.0188*

*The difference is statistically significant at the 5% level.

As in the simulation experiment, the PageRank Spider did not perform well in the user study. The expert judges found that the PageRank Spider collected many non-medical pages from large sites such as Yahoo and eBay (http://www.ebay.com/). As discussed earlier, the pages on these sites tended to have higher PageRank scores and were visited by the PageRank Spider during an early stage of the spidering process. As a result, the average relevance of the collection built by the PageRank Spider was not as high as the other two collections.

6.6 Conclusion

Given the continuous growth of the Web, it has become increasingly important to build vertical search engines of high quality. This research has provided some insights into development of search engine spiders that collect domain-specific Web pages using both content-based and link-based analysis. The combination of content and link structure analysis can be beneficial not only to Web spidering systems but also to other Web applications.

I also demonstrated how the Web could be modeled as a neural network. The Web can be viewed as a large collection of distributed yet interconnected knowledge which can be activated and retrieved by different neural network algorithms such as spreading activation. As neural network techniques have been widely studied, it would be very interesting to apply these techniques in Web applications, such as Web page clustering.

CHAPTER 7: USING MACHINE LEARNING TECHNIQUES FOR WEB PAGE FILTERING

7.1 Background

In Chapter 6, various methods for locating relevant URLs on the Web are reviewed. While these methods have different levels of performance in efficiency and effectiveness, in most cases the resulting collection is still noisy and needs further processing. Filtering programs are needed to filter irrelevant and low-quality pages from the collection to be used in specialized search engines. In this Chapter, I review related work in text classification, and propose a new approach to Web page filtering by applying machine learning-based text classification and Web analysis techniques.

7.2 Related Work

7.2.1 Web Page Filtering

Web page filtering is an essential process in creating specialized search engines. The filtering techniques used can be classified into the following four categories:

- Domain experts manually determine the relevance of each Web page (e.g., Yahoo).
- In the simplest automatic way, the relevance of a Web page can be determined by the occurrences of particular keywords (e.g., *computer*) (Cho et al., 1998). Web pages are considered relevant if they contain the specified keyword, and considered irrelevant otherwise.

- TF*IDF (term frequency * inverse document frequency) is calculated based on a lexicon created by domain experts. Web pages are then compared with a set of relevant documents, and those with a similarity score above a certain threshold are considered relevant (Baujard et al., 1998).
- Text classification techniques such as the Naive Bayesian classifier also have been applied to Web page filtering (Chakrabarti et al., 1998; McCallum et al., 1999).

Surprisingly, it appears that many vertical search engines do not perform filtering; they assume that most pages found in the starting domains (or at a specified depth) are relevant.

7.2.2 Text Classification

Text classification is the study of classifying textual documents into predefined categories. Please refer to Chapter 2.1.2 for a review of text classification techniques.

In addition to general text documents, classification of Web pages also has been studied. Web pages are often noisy, but they provide additional information about each document. For example, text from neighborhood Web pages has been used in attempt to improve classification performance. However, it turns out to worsen performance because there are often too many neighbor terms and too many cross-linkages between different classes (Chakrabarti et al., 1998; Yang et al., 2002). Use of other information about neighborhood Web pages has been proposed. Examples of such information include the predicted category of a page's neighbors (Chakrabarti et al., 1998; Oh et al., 2000),

anchor text pointing to a page (Furnkranz, 1999), or a page's outgoing links to all other documents (Joachims et al., 2001). It has been shown that using such additional information improves classification results.

7.3 A Feature-based Approach

Based on the review, several problems with traditional approaches to Web page filtering were identified. Firstly, a manual approach, like the one used by Yahoo, is extremely labor-intensive and time-consuming. Although such approach can achieve high quality, it is usually not feasible under limited resources. The keyword-based and the TFIDF-based approaches can automate the process, but they both have shortcomings. A simple keyword-based approach cannot deal with problem of *polysemy*, i.e., words having more than one semantic meaning. For example, a Web page containing the word "cancer" might well be a medical report about treatment for lung cancer or the horoscope for people born under the zodiac sign of cancer. As a result, this approach can easily fail to eliminate irrelevant pages. On the other hand, as people often use different terms to refer to the same concept, e.g., "lung cancer" and "lung neoplasm", this approach also can easily miss out relevant pages. The TFIDF approach alleviates the problem by considering all terms in the documents. However, TFIDF calculation can be biased by the collection; if the collection is "noisy", irrelevant terms can possibly get a very high IDF score and thus a high TFIDF score. In addition, both the keyword-based and the TFIDF approaches do not robustly resist spamming, a popular practice in which Web page authors manipulate the page content to boost ranking.

Using text classifiers for Web page filtering seem to be the most promising approach, given their good performance in traditional text classification. However, a major problem is that most classifiers were evaluated using at least 2/3 of the data for training in the hold-out sampling method. The problem becomes even worse in other evaluation methods such as k-fold cross-validation and leaving-one-out (Stone, 1974; Kohavi, 1995), in which (100-k)% of the data and all but one instance, respectively, were used for training. It is not feasible to obtain so large a set of training data in vertical search engine creation because only a small number of documents (hundreds) can be tagged for classifying a large number of documents (millions). Also, most existing text classification techniques do not make use of domain knowledge, which is important in vertical search engines.

In this study, the following research questions were investigated: (1) Can Web structure analysis techniques be used to help create a vertical search engine? (2) Can domain knowledge be used to enhance Web page filtering for a vertical search engine? (3) Can Web page classification be applied to a large collection (e.g., a million documents) with only a small number (a few hundred) of training examples?

In order to address the problems with current approaches in Web page filtering, a feature-based approach is proposed. Instead of representing each document as a bag of words, each Web page is represented by a limited number of content and link features. This reduces the dimensionality of the classifier and thus the number of training examples needed. The characteristics of Web structure also can be incorporated into these features.

In general, the relevance and quality of a Web page can be reflected in the following aspects: (1) the content of the page itself, (2) the content of the page's neighbor documents, and (3) the page's link information. Several features are defined for each aspect.

7.3.1 Page Content

The content of a page is probably the primary factor in determining whether a page is relevant to a given domain. As mentioned earlier, the content of each page was represented by a set of feature scores rather than a vector of words. An automatic approach was adopted to extract all the terms from a page and compare them with a domain lexicon, similarly to the method used in Baujard et al. (1998). Both the number of relevant terms that appear in the page title and the TFIDF scores of the terms that appear in the body of the page were considered. Two feature scores were defined:

1. Title(p) = Number of terms in the title of page p found in the domain lexicon
2. TFIDF(p) = Sum of TFIDF of the terms in page p found in the domain lexicon

7.3.2 Page Content of Neighbors

To incorporate the page content of the neighbors of a page, a score from each neighborhood document can be used instead of including all the terms from neighborhood documents, which appears to be more harmful than helpful (Chakrabarti et al., 1998; Yang et al., 2002). In my approach, three types of neighbors were considered: incoming, outgoing, and sibling (Chakrabarti et al., 1998). Two content scores (title and

TFIDF scores) of the neighborhood documents were determined similarly to those created in the previous aspect. Six features were used: the averages of the two scores for all incoming neighbors, the averages for all outgoing neighbors, and the averages for all siblings.

1. InTitle(p) = Average(number of terms in the title of page q found in the domain lexicon) for all incoming pages q of p
2. InTFIDF(p) = Average(sum of TFIDF of the terms in page q found in the domain lexicon) for all incoming pages q of p
3. OutTitle(p) = Average(number of terms in the title of page r found in the domain lexicon) for all outgoing pages r of p
4. OutTFIDF(p) = Average(sum of TFIDF of the terms in page r found in the domain lexicon) for all outgoing pages r of p
5. SiblingTitle(p) = Average(number of terms in the title of page s found in the domain lexicon) for all sibling pages s of p
6. SiblingTFIDF(p) = Average(sum of TFIDF of the terms in page s found in the domain lexicon) for all sibling pages s of p

7.3.3 Link Analysis

Connectivity (link analysis) is used to determine the quality of a page. Link analysis, such as number of in-links, HITS and PageRank, have been useful in many Web applications such as search result ranking (Brin & Page, 1998; Chakrabarti et al., 1999a), but have not been used in text classification. To incorporate link analysis scores in the filtering

approach, six scores, namely hub score, authority score, PageRank score, number of in-links, number of out-links, and number of relevant terms in the anchor texts, are used as features.

1. Hub(p) = Hub score of page p calculated by the HITS algorithm
2. Authority(p) = Authority score of page p calculated by the HITS algorithm
3. PageRank(p) = PageRank score of page p
4. Inlinks(p) = Number of incoming links pointing to p
5. Outlinks(p) = Number of outgoing links from p
6. Anchor(p) = Number of terms in the anchor texts describing page p found in the domain lexicon

7.3.4 Text Classifiers: FF/BP NN and SVM

In total, 14 features have been identified and can be used as the input values to a classifier. A neural network (NN) (Chen, 1995; Lippmann, 1987) and a support vector machine (SVM) (Vapnik, 1995; 1998) were used. A feedforward/backpropagation neural network (FF/BP NN) had been adopted because of its robustness and wide usage in classification (e.g., Wiener et al., 1995; Ng et al., 1997; Lam & Lee, 1999). The algorithm used is summarized as follows:

Initializing the network. A neural network was first created with three layers, namely the input layer, the hidden layer, and the output layer. The input layer of the neural network consisted of a threshold unit and 14 nodes that corresponded to the 14 feature scores of each page. The output layer consisted of a single output

node which determined the relevance of a page (whether or not a Web page should be included in the vertical search engine or not). The number of nodes in the hidden layer was set at 16 and the learning rate at 0.10. These parameters had been set based on some initial experimentation.

Training and tuning the network. The training documents were passed to the network for learning (the method of selecting the training set will be discussed in Section 5.5). The training documents were then further divided into two sets: 80% of the documents were used for training and 20% were used for tuning. The 14 features of each training document, as well as a binary score representing whether the document was relevant, were presented to the network. Each feature score was normalized to a value between 0 and 1 using the sigmoidal function. The network then updated the weights of its connection based on the training documents. After all training documents had passed through the network once, the tuning documents were presented to the network and the mean square error (MSE) of the network was recorded. The whole process was repeated 3,000 times (i.e., 3,000 epochs) and the network with the lowest MSE was selected.

Testing. Each testing document was presented to the trained network, which tried to predict whether the document should be accepted or not. The predictions were recorded and used to calculate the performance measures.

In order to allow for better comparison, a support vector machine also was used because of its outstanding performance in traditional text classification (Yang & Liu, 1999). It was used to perform classification based on the same set of feature scores. The SVM classifier involved the following steps:

Model Selection. A linear kernel function was chosen for the classifier because it is simple, learns quickly, and has been shown to achieve performance comparable to that of nonlinear models like polynomial classifiers and radial basis functions in text classification applications (Dumais et al., 1998; Joachims, 1998).

Training. Each training example was represented as a vector of the 14 features selected and presented to the SVM to learn the feature weights.

Testing. Similarly to the neural network algorithm, the SVM tries to predict based on its classification model whether each document in the testing set was relevant to the chosen domain. The results were recorded and used for evaluation.

7.4 Experimental Design

7.4.1 Experiment Testbed

In order to evaluate the proposed approach, two experiments that compared the proposed approaches with traditional approaches were conducted. The medical field was chosen as the domain for evaluation because many diverse users (including medical doctors, researchers, librarians and general public) seek important and high-quality information on

health topics on the Web. It is also important for them to distinguish between Web pages of good and poor quality (Chen et al., 2003).

A Web page testbed and a medical lexicon created in previous research were used (Chau & Chen, 2003). The Web page testbed was built by running a random-first search that started with 5 URLs in the medical domain and traversed the Web following random outgoing links. The random-first search was run until 1 million pages had been collected and indexed. The testbed represented a typical collection from simple Web spiders, and consisted of 1,040,388 valid, unique Web pages.

The medical lexicon was created based on the Metathesaurus, part of the Unified Medical Language System (UMLS) developed by the National Library of Medicine. About 600,000 medical phrases were extracted from the Metathesaurus. The lexicon was manually edited by a medical librarian and two filtering programs were developed and applied to revise the lexicon. The resulting lexicon has 300,442 unique terms.

To evaluate the proposed Web page classification approaches, 1,000 documents were randomly chosen from the testbed. Each of these documents was processed automatically to calculate its feature scores and keyword vector. All other Web pages in the testbed were accessible for content, neighbor and link analysis during the process. Two graduate students with medical training were also recruited to classify each document manually as either "acceptable" or "not acceptable" for a medical search engine.

7.4.2 Benchmark Approaches

The proposed feature-based neural network approach and feature-based support vector machine approach were compared against two benchmark approaches: (1) a TFIDF approach, and (2) a keyword-based support vector machine approach. The TFIDF approach was chosen because it is fast and has been used in various information retrieval applications. The keyword-based SVM approach was selected because it has been shown to achieve the best performance in traditional text classification problems (Yang & Liu, 1999).

The TFIDF approach was adopted from Baujard et al. (1998). TFIDF score is calculated for those terms found in the medical lexicon. The scores for all documents in the training set were calculated and a threshold that divided these training documents into the two classes (relevant and irrelevant) with the highest accuracy was determined. This threshold was then used for testing.

The second benchmark approach was a keyword-based SVM approach adopted from Joachims (1998). In the pre-processing stage, each document was first tokenized into single words. Common functional terms that did not bear a significant semantic meaning (e.g., a, of, and is) then were filtered based on a pre-defined stop-word list. In order to reduce the number of unique words and the vector size, I also followed Joachims's design by applying suffix-stripping (stemming) to the words, using the Porter stemmer (Porter, 1980). After the pre-processing, each document was represented as a keyword vector, which was used as the input to the SVM for training and testing.

7.4.3 Implementation

All four approaches were implemented in order to test their performances. In the TFIDF approach, the Arizona Noun Phraser (AZNP) was used to extract noun phrases from each document and these phrases were compared with the domain lexicon. The AZNP is a tool that extracts all valid noun phrases from a document, based on part-of-speech tagging and linguistic rules (Tolle & Chen, 2000). For the two approaches that rely on a support vector machine, the SVM-light package was used (Joachims, 1999). As mentioned earlier, the linear model was used for the kernel function of the SVM. All other programs, including the feature score calculation and the neural network algorithm, were implemented in Java.

7.4.4 Hypotheses

The experiment sought to compare the two feature-based approaches with the two benchmark approaches. I posed the following hypotheses:

- H1: The keyword-based SVM approach will perform with higher effectiveness than the TFIDF approach. The reason is that a keyword-based approach should be able to make better classification decisions by relying on more keyword information.

- H2: The two proposed feature-based approaches will perform with comparable effectiveness. I hypothesized that the two feature-based approaches should perform similarly because both neural network and support vector machine have

been widely used in text classification applications and should achieve comparable performance.

- H3: The two feature-based approaches will perform with higher effectiveness than the two benchmark approaches, i.e., the keyword-based SVM approach and the TFIDF approach. This hypothesis tests the main thesis of this study by verifying whether the proposed feature-based approaches perform better than the traditional approaches.

- H4: The TFIDF approach and the two feature-based approaches require significantly fewer training data than the keyword-based approach to achieve a satisfactory performance. I suggest the TFIDF-based and the feature-based approaches require fewer training data because they only rely on score(s) that should be similar across Web pages. Only a small number of training samples would be needed for the classifiers to learn the importance of the scores. On the other hand, the traditional keyword-based approach needs the occurrence of certain keywords in order to classify a document. When the number of training documents is small, it is likely that many words in the testing documents have not been seen before and hence provide no information for classification.

7.4.5 Experiment Setup

Each of the four approaches was evaluated using cross-validation, a widely-used evaluation methodology for machine learning and text classification systems (Stone,

1974; Kohavi, 1995). A 50-fold cross validation was adopted, in which the 1,000 documents in the data set were divided into 50 equal portions, with 20 documents each. Testing was performed for 50 iterations, in each of which 49 portions of the data (980 documents) were used for training and the remaining portion (20 documents) was used for testing. The data were rotated during the process such that each portion was used for testing in exactly one iteration.

I measured the effectiveness of each system using precision, recall, F-measure, and accuracy. Precision measures the fraction of the documents correctly classified as relevant, while recall measures the fraction of relevant documents retrieved from the data set. F-measure is a single measure that tries to combine precision and recall. Accuracy measures simply the prediction correctness of the classifiers. These measures are commonly used in text classification evaluation and have been adopted as follows:

$$precision = \frac{number\ of\ documents\ correctly\ classified\ as\ positive\ by\ the\ system}{number\ of\ all\ documents\ classified\ as\ positive\ by\ the\ system}$$

$$recall = \frac{number\ of\ documents\ correctly\ classified\ as\ positive\ by\ the\ system}{number\ of\ positive\ documents\ in\ the\ testing\ set}$$

$$F\text{-}measure = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{number\ of\ documents\ correctly\ classified\ by\ the\ system}{number\ of\ all\ documents\ in\ the\ testing\ set}$$

There are two popular ways to calculate the averages across the data for these metrics, namely, macro-averaging and micro-averaging (e.g., Joachims, 1998; Yang & Liu, 1999; Chai et al., 2002). In macro-averaging, the performance metrics are calculated for each

iteration, and the average of all iterations is obtained. In micro-averaging, the average is calculated across all the individual classification decisions made by a system. Both averages were calculated in the experiment. In addition to effectiveness, the time used by each classifier was also recorded in order to measure their efficiencies.

7.5 Experiment Results and Discussions

7.5.1 System Performance

The performances of the four approaches are summarized in Table 7.1.

Table 7.1: Experiment results

	Accuracy	Precision (macro/micro)	Recall (macro/micro)	F-measure (macro/micro)
TFIDF	80.80%	63.40% / 63.95%	60.52% / 62.50%	0.6005 / 0.6322
Keyword-based SVM	87.80%	87.97% / 94.94%	55.08% / 56.82%	0.6646 / 0.7109
Feature-based NN	89.40%	81.38% / 82.38%	76.19% / 76.14%	0.7614 / 0.7913
Feature-based SVM	87.30%	85.35% / 86.24%	61.99% / 61.74%	0.7049 / 0.7196

Because F-measures represent a balance between precision and recall, I focus the discussion on accuracy and F-measure. The results demonstrated that the TFIDF-based approach in general, did not perform as well as the other approaches; it achieved the lowest accuracy and F-measure. The feature-based NN approach achieved the highest accuracy and F-measure.

In order to study whether the differences among the different approaches were statistically significant, two statistical tests were adopted. The first was a micro sign-test that looks at all the classification decisions individually and uses a binomial distribution to determine whether the decisions made by any two approaches of interest are significantly different (Cohen, 1995). The number of observations n was defined to be the

number of times that the two systems made different classification decisions. The second test was a macro *t*-test that takes the performance of each iteration as an individual observation in order to determine whether the performances of two approaches are significantly different (Yang & Liu, 1999). The number of observations was 50, since there were 50 iterations of testing for each approach. The macro *t*-test was applied to both accuracy and F-measure.

Table 7.2: Micro sign-test results

vs.	Keyword-based SVM	Feature-based NN	Feature-based SVM
TFIDF	<0.00001**	<0.00001**	<0.00001**
Keyword-based SVM		0.0972*	0.3044
Feature-based NN			0.0095**

*The difference is statistically significant at the 10% level.
**The difference is statistically significant at the 1% level.

The p-values of the micro sign-tests are shown in Table 7.2. The results show that hypothesis H1 was supported as the keyword-based SVM approach performed better than the TFIDF approach with a p-value less than 0.00001. H3 was partly supported as both the feature-based SVM and NN approaches performed significantly better than the TFIDF approach, and the feature-based NN approach also performed significantly better than keyword-based SVM approach. H2 was not supported as the feature-based NN approach performed better than the feature-based SVM approach.

The p-values of the macro *t*-tests on accuracy and F-measure are shown in Tables 7.3 and 7.4 respectively. The results obtained were similar to those of the micro sign-test. In general, H1 was supported as the keyword-based approach performed significantly better than TFIDF approach. H3 also was partly supported as the feature-based NN approach

performed better than both benchmark approaches, but the feature-based SVM approach only performed better than the TFIDF approach but no better than the keyword-based approach. H2 was not supported as the feature-based NN approach performed better than the feature-based SVM approach. One possible reason is that because of the limitation in resources and time, it was more practical to use the linear model in the SVM. It is possible that the performance of both SVM approaches might improve if a non-linear model could be adopted.

Table 7.3: Macro *t*-test results on accuracy

vs.	Keyword-based SVM	Feature-based NN	Feature-based SVM
TFIDF	<0.00001**	<0.00001**	<0.0001**
Keyword-based SVM		0.1627	0.6091
Feature-based NN			0.0216*

*The difference is statistically significant at the 5% level.
**The difference is statistically significant at the 1% level.

Table 7.4: Macro *t*-test results on F-measure

vs.	Keyword-based SVM	Feature-based NN	Feature-based SVM
TFIDF	0.0827*	<0.00001**	0.0041**
Keyword-based SVM		0.0024**	0.2446
Feature-based NN			0.0033**

*The difference is statistically significant at the 10% level.
**The difference is statistically significant at the 1% level.

7.5.2 Analyzing the Importance of the Three Aspects

While the feature-based NN approach achieved the best performance in the experiment, it would be interesting to know which features contributed more to the good performance of the classifier. To study this issue, I performed another experiment to test the performance of the feature-based NN approach by varying the set of features used by the classifier. Seven different settings were used with different combinations of the three

aspects, namely page content, page content of neighbors, and link analysis. The seven different settings are as follows:

1. Features from Page Content only
2. Features from Page Content of Neighbors only
3. Features from Link Analysis only
4. Features from Page Content and Page Content of Neighbors only
5. Features from Page Content and Link Analysis only
6. Features from Page Content of Neighbors and Link Analysis only
7. Features from all three aspects (all 14 features)

A 50-fold cross validation was run for each setting. The results are shown in Table 7.5.

Table 7.5: Comparison of the three aspects

	Page Content	Page Content of Neighbors	Link Analysis	Accuracy	F-measure (macro/micro)
1	✓			80.80%	0.5961 / 0.6265
2		✓		89.50%	0.7998 / 0.8000
3			✓	73.50%	0.0067 / 0.0075
4	✓	✓		88.20%	0.7631 / 0.7731
5	✓		✓	81.90%	0.5820 / 0.6173
6		✓	✓	89.10%	0.7542 / 0.7900
7	✓	✓	✓	89.40%	0.7674 / 0.7913

The table shows that among the seven settings, the classifier performed the best when only features derived from the page content of neighbors were used (setting 2), achieving an accuracy of 89.50% and a macro F-measure of 0.7998. It is surprising to find that it performed even better than the other settings in which additional information like page content (setting 4), link analysis (setting 6), or a combination of both (setting 7) was used

together with the page content of neighbors. The result shows that the page content of neighbor documents is the most important factor in determining the relevance of a page.

On the other hand, the classifier only achieved a macro F-measure of 0.5961 when page content was used alone (setting 1). This value is similar to the F-measure obtained by the TFIDF approach. This finding is reasonable because in this setting the feature-based NN approach only relied on two feature scores (the Title score and the TFIDF score), similarly to the TFIDF approach. The performance degraded slightly when link analysis features were added and used together with page content features (setting 5), achieving an F-measure of 0.6173. The F-measure was improved to 0.7731 when page content features were used with the features derived from the page content of neighbors (setting 4).

The classifier obtained a very low macro F-measure of 0.0067 when only link analysis was used (setting 3). The result is disappointing as it was expected that the link analysis score would be able to help in determining the relevance of a page. One possible reason for its failure is that since the link analysis scores were not content-specific, many Web pages that had high link analysis scores (such as PageRank score and HITS scores) might be pages that were linked by many other pages but were not relevant to the selected domain at all. Because the NN classifier was designed to minimize the mean squared error (MSE) of the system, it predicted almost all pages to be negative (i.e., not relevant) in order to maintain a resonable accuracy of 73.50%. As a result, the classifier performed poorly in precision and recall.

7.5.3 Efficiency

The time needed for each system to perform the 50-fold cross validation (including both training and testing time) was recorded. The data are shown in Table 7.6. As can be seen, the keyword-based SVM approach required the longest time. The reason is that each document was represented as a large vector of keywords, which created a high dimensionality for the classifier. In the experiment, there were more than 6,000 unique words after stop-word removal and stemming. The classifier had to learn the relationships between all these attributes and the class attribute, thus requiring more time. The TFIDF approach used the least time, as it needed only to calculate the TFIDF score for each document and determine the threshold, both of which did not require complex processing. Comparing the two feature-based approaches, the NN classifier required a longer time than the SVM classifier because the neural network had to be trained in multiple epochs, i.e., in each iteration the training data set had to be presented to the network thousands of times in order to improve the network's performance.

Table 7.6: Time needed for 50-fold cross validation

	Time (minutes)
TFIDF	7.45
Keyword-based SVM	382.55
Feature-based NN	103.45
Feature-based SVM	37.60

7.5.4 Effect of the Number of Training Examples

In order to analyze the effect of the number of training examples on the performance, the experiments were run on the systems with varied number of training data. I started with 20 documents in the first run, and increased the number of training documents by 20 in

each subsequent run. There were thus 49 runs in total (from 20 to 980 training documents). In each run, a 50-fold cross validation similar to the one described above was used, and 20 documents were used for testing with rotation. The macro-averaged F-measure for each iteration was recorded and the results are shown in Figure 7.1.

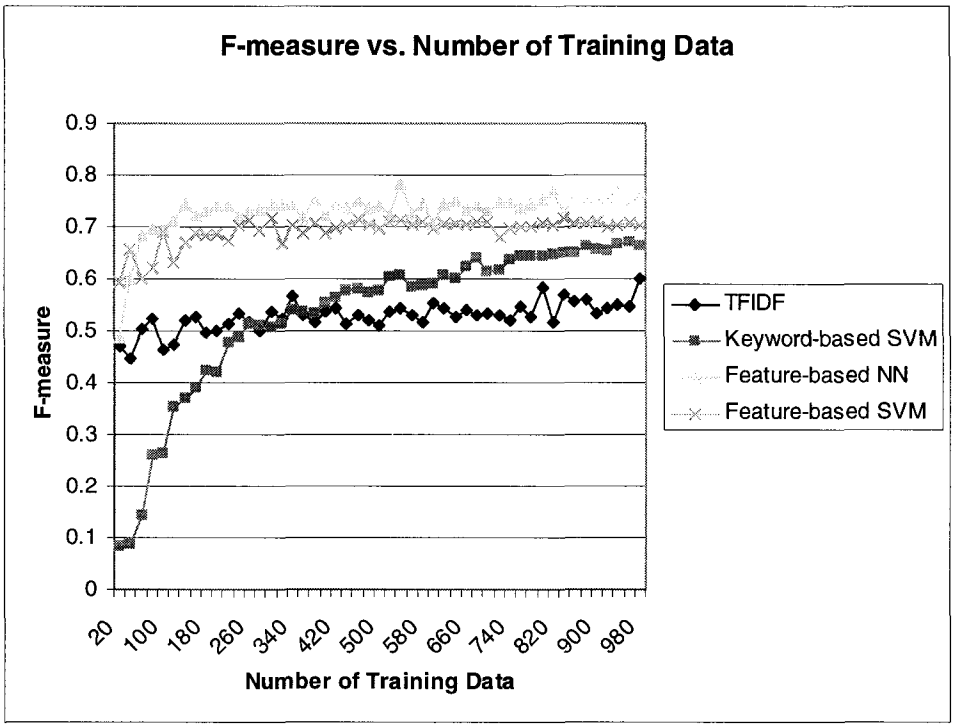

Figure 7.1: F-measure vs. number of training data

From the graph, it can be seen that the performances of the TFIDF approach and the two feature-based approaches became relatively stable after approximately 300, 140, and 260 training documents were used respectively. For the keyword-based approach, however, performance was unstable until about 700 training documents had been used. This supported hypothesis H4 that fewer documents were needed for the TFIDF approach or the feature-based approaches to achieve a satisfactory performance. As mentioned earlier,

this finding is especially important for building vertical search engines, since a large number of training documents are often not available. The graph also shows that traditional keyword-based approaches often require about 2/3 of the available data to be used for training in order to achieve a satisfactory and stable performance.

7.6 Conclusion

In this Chapter, I described a feature-based approach to Web page classification that combines Web content analysis and Web structure analysis. The proposed approaches were compared with traditional text classification methods and the experimental results were encouraging. The results showed that the proposed approaches are useful for various Web applications, especially for vertical search engine development.

While the feature-based approaches are promising, it is interesting to study which of the 14 features used are more important than the others in determining the relevance of a page. I plan to apply factor analysis techniques to the data set to investigate the features in detail. Another future direction will be to study whether a combined keyword-based and feature-based approach will perform better than using either the keywords or the features alone, as were in this study. I believe that a combined approach may potentially acquire the strengths of both approaches and perform better by allowing the classifier to rely on the feature scores when the number of training documents is small, but to rely more on the unique keyword attributes in the vector when the number of training documents reaches a certain level. Finally, I am also investigating how the proposed

classification systems can be applied to other applications, such as knowledge management and Web content management.

CHAPTER 8: CONCLUSIONS AND FUTURE DIRECTIONS

This dissertation investigates how machine learning and artificial intelligence techniques can be used to enhance Web searching using a personalized search agent approach and a specialized search engine approach. Several prototype systems have been developed and evaluated, and the experimental results are encouraging. In this final Chapter, I review the major contributions of this dissertation, discuss its relevance to business and MIS research, and suggest some possible future research directions.

8.1 Contributions

As this dissertation involves several different studies and disciplines, it makes several contributions, both in research and applications. The dissertation has the following theoretical contributions:

- The CI Spider and Meta Spider experiments showed that post-retrieval analysis was able to improve users' search performance.

- The study on Cancer Spider and Nano Spider demonstrated that the personalized agent approach can be easily customized across different domains.

- The Collaborative Spider study proposed a new multi-agent approach to collaborative Web searching and Web mining. The experiment conducted also verified that this approach could improve users' search performance. The

architecture has later been applied in a law enforcement application (Chau et al., 2001a; Zeng et al., 2003).

- A spreading activation algorithm was used in a Hopfield Net spider to locate URLs relevant to given domains. This study showed how useful, relevant resources can be identified automatically and dynamically on the Web.

- A feature-based text classification system was developed to facilitate the creation of specialized search engines. The study proposed a new feature-based approach to text classification and Web page filtering.

This dissertation also has the following technical research contributions:

- The study on personalized search agents demonstrated how noun phrasing and document clustering techniques can be combined efficiently and effectively in Web searching and mining applications.

- In the Hopfield Spider study, I proposed a way to apply the spreading activation algorithm efficiently to Web searching. I also showed how the Web could be modeled as an artificial neural network.

- The study on Web page filtering demonstrated how Web content analysis and Web structure analysis can be combined in Web page classification applications using a feature-based approach.

The dissertation also contributes to various science, medicine, and management applications in the following ways:

- The CI Spider system allows business managers in various companies and organizations (including IBM, AlliedSignal, and the US Army) to perform competitive intelligence analysis on the Web. Business managers can use the tool to analyze the key strengths of a competitor, to survey the landscape of a particular industry, or to support knowledge management activities.

- The Meta Spider system provides general Web users with a tool to search the Web in an alternative way, by meta-searching various engines and combining their search results. This tool also can be used by managers to support business intelligence analysis and knowledge management by searching and analyzing Web pages.

- Cancer Spider and Nano Spider allow scientists, physicians, practitioners, researchers, and students to perform better searching and analysis in their domains of interest. Users can use these tools to locate Web pages relevant to the chosen domains and perform post-retrieval analysis on the Web pages collected.

- The Collaborative Spider system allows users to perform collaborative Web searching and Web mining by sharing complete search sessions. This

functionality can enhance the collaboration among Web users and increase their effectiveness in Web searching.

- The Hopfield Net spider and the Web page filtering algorithm enable users to develop specialized search engines with minimal manual effort. Several search engines, including the HelpfulMed for the medical domain (Chen et al., 2003) and the NanoPort for the nanotechnology domain (Chau et al., 2002b), have been built based on these algorithms.

8.2 Relevance to Business, Management, and MIS

The Web has become a major resource that can provide useful and important information to support various business and management activities such as knowledge management, business intelligence, and decision making. It has become increasingly important for business managers to make effective use of such online information in order to remain competitive in the global business environment. However, given the large amount of information available on the Web, business managers are often faced with the problem of information overload. General-purpose search engines usually cannot satisfy the information needs of business managers.

The evaluation studies reported in this dissertation demonstrated that the proposed approaches could be very useful in helping business executives locate, retrieve, and analyze information on the Web effectively and efficiently. As mentioned earlier, business managers can use CI Spider and Meta Spider to support business intelligence

and knowledge management. For example, they can use CI Spider to support competitive intelligence by conducting a thorough analysis of a competitor's Web site. The CI Spider system can identify the key products and strategies used by the competitor. Business managers can also use Meta Spider to survey the landscape or new trends in an industry by searching for Web pages relevant to the industry and performing post-retrieval analysis on the retrieved pages. The important topics discussed in these pages can be identified as noun phrases and visualized in a two-dimensional topic map. The Collaborative Spider system also allows business managers working in different divisions within an organization to share their Web search findings in a collaborative environment. The Collaborative Spider also can help create an organizational knowledge repository by supporting knowledge creation and knowledge management.

In addition, the techniques and algorithms reported in this dissertation also can be generalized and applied to other applications. For example, besides Web pages, the noun phrasing and SOM clustering tools can be used for other types of documents of an organization, such as email messages, outputs of electronic meeting systems, or product review documents. Analysis of such documents is very useful for various business functions such as knowledge management or customer relationship management. The Web page spidering and filtering techniques also can be used to support Web-based business intelligence. For example, the spidering and filtering techniques can be used to develop a Web portal that supports business intelligence by allowing business managers to perform searches and analyses on the Web in order to keep track of the latest industry trends and changes in the competitive environment.

The Web is the largest information system that has ever existed, with the largest number of users, the busiest traffic, and the most information. At the same time, the Web has become an essential part of most management information systems. The evaluation studies reported in this dissertation investigated how users interacted with various Web-based information systems using different techniques. The evaluation results also can be applied to other similar Web-based systems.

8.3 Future Directions

The Web has become the world's largest knowledge repository. Extracting knowledge from the Web efficiently and effectively is becoming increasingly important for various applications.

Most current Web mining applications only scratch the surface of the Web's "knowledge mine." Most Web mining activities are still in their early stages and should continue to develop as the Web evolves. One future research direction for Web mining is multimedia data mining. Besides textual documents, which have been the main focus of this dissertation, there are a large number of multimedia documents on the Web, such as images, audios, and videos. While textual documents are comparatively easy to index, retrieve and analyze, operations on such multimedia files are much more difficult to perform. Because amount of multimedia content on the Web is growing rapidly, Web mining on multimedia content has become a challenging problem. Various machine learning techniques have been employed to address this issue. As can be predicted, research in pattern recognition and image analysis has been adapted for study of

multimedia documents on the Web, such as video (Wactlar et al., 1999; Christel et al., 2002) and music (McPherson & Bainbridge, 2001). Relevant text that describes a multimedia file, such as the "alt" text (alternative text), anchor text, HTML headings, table headings, image and video captions and descriptions, also have been used for analyzing multimedia documents (Rowe, 2002). However, these techniques currently are used primarily for information retrieval on the Web, rather than for Web mining. As a picture is worth a thousand words, I believe that multimedia data contain such rich knowledge that Web mining applications should not ignore them. It would be interesting to apply various Web searching and analysis techniques to multimedia data on the Web.

In addition to becoming more diverse in content types, the Web has also become more international and multi-cultural. Non-English Web content has experienced strong growth over the past few years, and the globalization and e-commerce trend has created extensive multilingual content. Current research in multilingual analysis used in Web applications include Web page translations, such as the AltaVista Babel Fish (http://babelfish.altavista.com/), and cross-language information retrieval in which a search query is entered in one language to retrieve Web pages in another language. Similarly to the situation for multimedia content, these techniques are often used only for information retrieval. Future Web mining applications should attempt to extract and infer knowledge from a set of multilingual documents. The Web agents and specialized search engines proposed in this dissertation also should be adapted to and tested in other languages.

Another important area is the Wireless Web. Although it is likely that the majority of Web content will still be traditional Web pages such as HTML documents, more and more documents on the Web will be written in formats designed for handheld devices such as PDAs (Personal Digital Assistants) and cellular phones. WML (Wireless Markup Language) and HDML (Handheld Device Markup Language) are examples of such formats. The wireless portion of the Web is also quite different from the traditional Web. The information contained in the Wireless Web is often more concise, more location-specific, and more time-critical. In addition, because of the nature of wireless devices, the usage patterns for the Wireless Web are also quite different from that of the traditional Web. It would be interesting to apply Web mining techniques to the Wireless Web as well as to use such techniques to improve wireless information delivery by methods such as information personalization.

The Hidden Web, also known as the Invisible Web or the Deep Web, has given rise to another issue facing Web mining research. The Hidden Web refers to the portion of documents on the Web which are dynamic and not accessible from general search engines, because most search engine spiders can access only the publicly indexable Web (or the visible Web). Most of the documents in the Hidden Web, including pages hidden behind search forms, specialized databases, and dynamically generated Web pages, are not accessible by general Web mining applications. If, as has been estimated, the Hidden Web is 400 to 550 times larger than the visible Web (Lyman & Varian, 2000), extracting information and knowledge from it has become a great challenge for search engines as well as Web mining applications. One interesting future research topic will be to study

how the proposed Web agents like CI Spider and Meta Spider can be enhanced to collect documents in the Hidden Web.

As discussed earlier, the Semantic Web provides great prospects for Web mining research. However, the Semantic Web is not without its weaknesses, the major one being that it requires the support of a substantial amount of Web users for it to be successful. If Web page authors do not see a great benefit for themselves in migrating to the Semantic Web, they will be reluctant to do metadata markup in Web pages, which requires significant work. As the Semantic Web is still in its infancy, Web mining researchers should pay close attention to its development and see how it affects Web mining applications as it matures.

The Web has become the largest knowledge base ever to have existed. However, without appropriate knowledge representation and knowledge discovery algorithms, the Web is just like a human being with extraordinary memory but no ability to think and reason. Research in machine learning and Web mining are promising as well as challenging, and both fields will help develop applications that can more effectively and efficiently utilize the Web of knowledge of the humankind.

APPENDIX A: DOCUMENTS FOR THE CI SPIDER EXPERIMENT

A.1 Instructions for Experiment Participants

Imagine yourself as in a consultant group who has been hired to do some research or investigation on a certain subject in a certain company. You might or might not have the domain knowledge, but it does not really matters. By searching or browsing the web pages and utilizing search engines, you should strive to gain a decent understanding and overview about the subject matter.

While searching and browsing, you should

- Notice the "themes" of the subjects you found in the retrieved documents
- Use a short phrase to describe each theme found

Since your clients are busy business executives who are only interested in a short, high-level briefing, your deliverable should be ***a list of the themes*** you found that would capture your search results. Make sure that the themes you gave are based on what you read in the webpages. That means you should NOT come up with something not mentioned in the webpages, even if you know by common sense that it's true.

A theme is a "short phrase" (not a sentence) which describes a certain topic. Try to come up with not only subtopics under the given topic, but also some other related topics. A theme should be concise and precise, and you don't need to draw a conclusion. For example, "risk in using mobile phone" is a good theme, while "The risk in using mobile phone is that it is emitting radioactive ray" is not.

There is no time constraint on your search. You can stop when you feel you have got a good understanding of the subject.

You will be given an answer sheet with topics on it. `<website>` is the website we are investigating. `<concept>` is the subject you need to search for.

We strongly encourage you to think aloud your like and dislike about the search method while searching. After all the search, you are requested to complete a short questionnaire about each search method.

Do you have questions about:

- What a theme is?
- What your task is?

Good luck and good surfing!

Instructions for CI Spider

To use CI Spider, follow these steps:

1. Enter Starting URL in the *URL* box.
2. Enter queries in the *Keyword* box. Try different query combinations using *"or"* or *"and"* logic operator.
3. Click on *Search* button. Searching is not finished until the *Back* and *Next* button pops up. Meanwhile you can click on any web pages already fetched and view the contents.
4. The globe icon with a cross on it indicates that the web page fetched does NOT contain the keywords. Click on *Good URL* tab, which only shows web pages of desired content. Click on any web page, you can browse the web site.
5. After fetching, click *Next* button, it shows you all the noun phrases extracted from the good web pages. Then click on *Create Map* button, you will see a 2-D map of the noun phrases. The size of the color block indicates the importance of the noun phrase in the retrieved documents; the proximity indicates the relationship between two concepts. Click on any color block, you will see those web pages containing that noun phrase (keyword).
6. You can modify the map by clicking on the *Back* button to the Noun Phraser page and deselecting noun phrases that you think are not relevant enough to be included in the map.
7. We strongly encourage you to fully utilize the functionality provided by Noun Phraser and the map.

Instructions for Lycos Advanced Search

1. Enter your query in the search box
2. Check the radio button "Selected Web Site"
3. Enter web domain in the text box "Selected Web Site"
4. Submit the search query
5. View the ranked list page
6. Click on any of the list to browse the web page

Instructions for Manual Browsing

1. Enter the starting URL in the address box
2. Click on links to go to different pages
3. You may use search engine within a website

Search Topics

1. <website> http://www.ibm.com/
   <concept> consulting

2. <website> http://www.eye2eye.com/
   <concept> computer monitor

3. <website> http://www.stortek.com/
   <concept> 9840

4. <website> http://www.phoenix.com/
   <concept> merger

5. <website> http://www.ei.com/
   <concept> mass media, advertising

6. <website> http://www.zoom.com/
   <concept> Hayes

Post-Search Questionnaire

Part A Manual Browsing

Please evaluate the IR system on the following dimensions:

1. How easy / difficult is it to locate **useful** information when manually browsing a site?

   1.....2.....3.....4.....5
difficult easy

2. Do you learn anything about the subject you searched for in the "manual browsing" experiment?

   1.....2.....3.....4.....5
very little a lot

3. Did you use the domain search engine in the "manual browsing" experiment?

   Yes No

Part B Lycos Advanced Search

4. The user interface of Lycos Advanced Search is

   1.....2.....3.....4.....5
confusing easy

5. How easy / difficult is it to locate **useful** information using Lycos Advanced Search?

   1.....2.....3.....4.....5
difficult easy

6. Do you learn anything about the subject you searched for in the "Lycos Advanced Search" experiment?

   1.....2.....3.....4.....5
very little a lot

Part C CI Spider

7. The user interface of CI Spider is

1.....2.....3.....4.....5
confusing easy

8. How easy / difficult is it to locate **useful** information using CI Spider?

1.....2.....3.....4.....5
difficult easy

9. Do you learn anything about the subject you searched for in the "CI Spider" experiment?

1.....2.....3.....4.....5
very little a lot

10. Which of the following component(s) in the CI Spider best assist you in searching for **useful** information? (Circle one or more)

1. Tree display of webpages
2. Noun Phraser
3. Categorization Map

That's the end of the experiment. Thank you very much!

A.2 Instructions for Experimenters

Data to collect:

1. Quantitative: answer sheet with experiment numbers on it (to be assigned). Record time spent on each search.

2. Qualitative:

- Prompt users to think aloud for their like and dislike about the systems including CI Spider, Lycos Advanced Search, and Manual Browsing. Write down their comments on each system if applicable.
- Notice and record major changes in user search behavior including the number of documents browsed.

Turn In:
Each individual subject's *answer sheet, his/her user log* and *questionnaires*. You do NOT need to staple them.

Procedures:

1. Greetings and confirming names
2. Give out instructions and answer sheet, remind them to do think-aloud
3. Let your subject read the instruction. Ask them if they have any questions
4. Query task:
   - training
   - searching by themselves, **record start time**
   - while they were searching, record both the qualitative and quantitative data on the user log
   - **record end time**
5. Give your subject the questionnaire to complete
6. Complete user log by yourself.
7. Repeat step 4-6 for all three query tasks.
8. Turn in individual package

- **CI: CI Spider**
- **LY: Lycos Advanced Search**
- **MB: Manual Browsing**

1. (query number)_____ (IR system)_____

User Comments:

Time Expended: _____Minutes

Number of documents browsed: _____

NOTE:

2. (query number)_____ (IR system)_____

User Comments:

Time Expended: _____Minutes

Number of documents browsed: _____

NOTE:

3. (query number)_____ (IR system)_____

User Comments:

Time Expended: _____Minutes

Number of documents browsed: _____

NOTE:

A.3 Rotation of Search Tools and Search Topics

Search tools: CI: CI Spider
LY: Lycos Advanced Search
MB: Manual Browsing

Subject	*First Experiment*	*Second Experiment*	*Third Experiment*
1	CI1	LY2	MB3
2	LY3	MB4	CI2
3	MB5	CI3	LY4
4	CI4	LY5	MB6
5	LY6	MB1	CI5
6	MB2	CI6	LY1
7	CI2	LY3	MB4
8	LY4	MB5	CI3
9	MB6	CI4	LY5
10	CI5	LY6	MB1
11	LY1	MB2	CI6
12	MB3	CI1	LY2
13	CI3	LY4	MB5
14	LY5	MB6	CI4
15	MB1	CI5	LY6
16	CI6	LY1	MB2
17	LY2	MB3	CI1
18	MB4	CI2	LY3
19	CI4	LY5	MB6
20	LY6	MB1	CI5
21	MB2	CI6	LY1
22	CI1	LY2	MB3
23	LY3	MB4	CI2
24	MB5	CI3	LY4
25	CI5	LY6	MB1
26	LY1	MB2	CI6
27	MB3	CI1	LY2
28	CI2	LY3	MB4
29	LY4	MB5	CI3
30	MB6	CI4	LY5

APPENDIX B: DOCUMENTS FOR THE META SPIDER EXPERIMENT

B.1 Instructions for Experiment Participants

Imagine yourself as in a consultant group who has been hired to do some research or investigation on a certain subject. You might or might not have the domain knowledge, but it does not really matters because you will be given Information Retrieval tools: search engines to accomplish your goal. By searching or browsing on search engines, you should strive to gain a decent understanding and overview about the subject matter.

While searching and browsing, you should

- Notice the aspects and constituent themes of the subjects you found in the retrieved documents
- Select and record a short phrase describing each theme found

Since your clients are busy business executives who are only interested in a short, high-level briefing, your deliverable should be a list of the themes and aspects you found that would capture your search results.

There is no time constraint on your search. The only criterion is that when you feel you have got a good understanding of the subject.

You will be given an answer sheet with topics on it. *<title>* is the subject you need to search for. *<description>* provides specific guidelines for searching, to which your searching should comply with. *<suggested query format>* provides candidate queries to enter. Feel free to use your own, but we suggest using more than one query for each search for better results.

We strongly encourage you to think out loud your like and dislike about the search tool where searching. After each search, you are requested to complete a short questionnaire about each searching system.

Do you have questions about:

- What themes and aspects are?
- What your task is?

Good luck and good surfing!

Instructions for MetaSpider

To use MetaSpider, follow the steps:

8. Enter queries in the *keyword* box. Try different query combinations using *or* logic operator.
9. Click on *search* button.
10. Click on *fetch* button. There is a time delay in this phase. Fetching is not finished until the *back* and *next* button pops up. Meanwhile you can click on any web pages already fetched and displayed.
11. The globe icon with a cross on it indicates that the web page fetched does NOT contain the keywords. Click on *Good URL* tab, it only shows web pages of desired content. Click on any web page, you can browse the web site.
12. After fetching, click *next* button, it shows you all the noun phrases extracted from the good web pages. Then click on *create map* button, you will see a 2-D map of the noun phrases. The size of the color block indicates the importance of the noun phrase in the retrieved documents; the proximity indicates the relationship between two concepts. Click on any color block, you will see those web pages containing that noun phrase.
13. You can modify the map by clicking on the *back* button to the noun phraser page and deselecting noun phrases that you think are not relevant enough to be included in the map.
14. We strongly encourage you to fully utilize the functionality provided by noun phraser and the map.

Instructions for MetaCrawler

1. Enter your query in the search box
2. View the ranked list page
3. Click on any of the list to browse the web page

Instructions for NorthernLight

1. Enter your query in the search box
2. View the ranked list page
3. "Customer Search Folder" on the left side of the page shows you documents belong to that concept cluster.
4. Click on those blue folders to view more detailed breakdown
5. Click on any of the list to browse the web page

Search Topics

1. <title> Hubble Telescope Achievements
<description> Identify positive accomplishments of the Hubble telescope since it was launched in 1991.
<suggested query format> hubble space telescope, hubble telescope achievements, hubble telescope

2 <title> Implant Dentistry
<description> What are the advantages and/or disadvantages of tooth implants?
<suggested query format> implant dentistry, dentistry

3. <title> Radio Waves and Brain Cancer
<description> Evidence that radio waves from radio towers or car phones affect brain cancer occurrence.
<suggested query format> radio waves and brain cancer, radio waves, brain cancer

4. <title> Undersea Fiber Optic Cable
<description> Fiber optic link around the globe (Flag) will be the world's longest undersea fiber optic cable. Who is involved and how extensive is the technology on this system. What problems exist?
<suggested query format> undersea fiber optic cable, fiber optic cable

5. <title> New Fuel Sources
<description> What research is ongoing for new fuel sources.
<suggested query format> new fuel sources, fuel sources

6. <title> Health and Computer Terminals
<description> Is it hazardous to the health of individuals to work with computer terminals on a daily basis? What are the potential problems?
<suggested query formats> health and computer terminals, health

Post-Search Questionnaire

Subject number: _____ IR System: _____

Please evaluate the IR system on the following dimensions:

1. The user interface is

   1……2…..3…..4…..5
confusing easy

2. How easy / difficult to locate **useful** information using this IR system?

   1…..2…..3…..4…..5
difficult easy

3. (for MetaSpider only) Which of the following component(s) best assist you in searching for **useful** information?

   1. fetch page
   2. document view (web pages)
   3. noun phraser
   4. the map

4. Do you learn anything about the subject you searched for?

   1…..2…..3…..4…..5
very little A lot

Subject number: _____ IR System: _____

Please evaluate the IR system on the following dimensions:

5. The user interface is

1…...2…..3…..4…..5
confusing easy

6. How easy / difficult to locate **useful** information using this IR system?

1…..2…..3…..4…..5
difficult easy

7. (for MetaSpider only) Which of the following component(s) best assist you in searching for **useful** information?

5. fetch page
6. document view (web pages)
7. noun phraser
8. the map

8. Do you learn anything about the subject you searched for?

1…..2…..3…..4…..5
very little A lot

Subject number: _____ IR System: _____

Please evaluate the IR system on the following dimensions:

9. The user interface is

1.....2.....3.....4.....5
confusing easy

10. How easy / difficult to locate **useful** information using this IR system?

1.....2.....3.....4.....5
difficult easy

11. (for MetaSpider only) Which of the following component(s) best assist you in searching for **useful** information?

9. fetch page
10. document view (web pages)
11. noun phraser
12. the map

12. Do you learn anything about the subject you searched for?

1.....2.....3.....4.....5
very little A lot

B.2 Instructions for Experimenters

Data to collect:

1. quantitative: answer sheet with experiment numbers on it (to be assigned). After subjects complete answers on the computer, print out the answer sheet. Second, record time spent on each search.
2. qualitative:
   * prompt users to do thinkout loud for their like and dislike about the systems including MetaSpider, MetaCrawler, and NorthernLight. Write down their comments on each system if applicable.
   * notice and record major changes in user search behavior including the number of documents browsed, number of times switching between ranked list and document pages.

Turn In:
Each individual subject's *answer sheet* stapled with his/*her user log* and questionnaire (next page)

Procedures:

1. Greetings and confirming names
2. Give out instructions and answer sheet, remind them to do think-aloud
3. Let your subject to read the instruction, ask them if they have any questions
4. Query task:
   - training
   - searching by themselves, **record the starting time**
   - while they were searching, record the both of the qualitative and quantitative data on the user log, **record end time**
5. Give your subject the questionnaire to complete
6. Complete user log by yourself.
7. Repeat step 4-6 for all three query tasks.
8. Turn in stapled individual package

- **MS: MetaSpider**
- **MC:MetaCrawler**
- **NL: NorthernLight**

Subject number: _____

Search Sequence:
1. (query number)_____ (IR system)_____

User Comments:

Time Expended:_____Minutes

Number of documents browsed: _____

Number of times switching between ranked list page and document pages:

_____

NOTE:

2. (query number)_____ (IR system)_____

User Comments:

Time Expended: _____Minutes

Number of documents browsed: _____

Number of times switching between ranked list page and document pages:

_____

NOTE:

3. (query number)_____ (IR system)_____

User Comments:

Time Expended: _____Minutes

Number of documents browsed: _____

Number of times switching between ranked list page and document pages:

_____

NOTE:

APPENDIX C: DOCUMENTS FOR THE CANCER SPIDER EXPERIMENT

C.1 Instructions for Experiment Participants

As a prospective health professional, you may find that medical and health-related resources available on the Internet are vast. The phenomenon of information overload leaves you sunk in the ocean of information, yet thirsty for real knowledge. This experiment gives the opportunity to explore with CancerSpider and NLM Gateway. With the help of the built-in knowledge-forming mechanisms, you should find searching on the web easier.

You are to perform 2 search tasks on two different systems, namely **CancerSpider** and **NLM Gateway**. After searching and browsing the documents with the help of categorization tool, you should have a decent understanding of your search topic. Then **write down a list of themes** that best capture your search result in the blank space provided below.

A theme is a short sentence or long, descriptive phrase that summaries a specific aspect of the given search task.
e.g. a list of themes of a search task "What is the impact of usage of computer terminals on health?" may look like:

1. causes vision(short eye sight, blurry vision) problems
2. ergonomics because of repetitive stress
3. radiation from the computer monitor

We strongly encourage you to think out loud your like and dislike about the search tool where searching. After each search, you are requested to complete a short questionnaire about each searching system.

Do you have questions about:

- What theme is ?
- What your task is?

Good luck and good surfing!

Instructions for CancerSpider

To use CancerSpider, follow the steps:

1. Enter queries in the ***keyword*** box. Hit RETURN or ***Add Keyword*** to enter one keyword at a time. If you have 3 keywords, you need to repeat the process 3 times.
2. Click on ***search*** button.
3. Click on ***fetch*** button. Fetching is not completed until the ***back*** and ***next*** button pops up.
4. The globe icon with a red cross on it indicates that the web page fetched does NOT contain the keywords. Click on ***Good URL*** tab on the upper left side, it only shows web pages of desired content.
5. View the keyword list. The more keywords extracted out, the higher chance the particular document is a good candidate to read on. Click on any web page, you can view the document itself.
6. **To narrow down your search**, click ***next*** button, it shows you all *the noun phrases* extracted from the good web pages. Click on any noun phrase, you will see a list of documents that contain that noun phrase.
7. Then click on ***create map*** button, you will see a 2-D map of the noun phrases. The size of the color block indicates the importance of the noun phrase in the retrieved documents; the proximity indicates the relationship between two concepts. Click on any color block, you will see those web pages containing that noun phrase.
8. You can modify the map by clicking on the ***back*** button to the noun phraser page and deselecting noun phrases that you think are not relevant enough to be included in the map.

You are required to go through all CancerSpider search steps.

Instructions for NLM Gateway

1. Enter your query in the ***search*** box. Separate multiple queries by comma, "and" or "or" logic operators.
2. If you need help with terminology, click on ***Find Terms*** for help. Type in your keywords in the box near the ***Find Terms*** button. Click on ***Find Terms*** button. Select the suggested terms that are applicable to you by clicking on the combo box right next to each term, and then click on ***Add to Search*** button.
3. Click on ***Search***
4. The result page shows you the total number of documents retrieved. To read Journal Citations, click on the first **Display Results**
5. Browse the document list. Do NOT view more than 40 documents (2 pages in the document list).
6. To view a particular document of interest, click on ***Expand***

Answer Sheet

Subject number: _____

Query number: _____

IR system: _____

Your list of themes goes here:

1.

2.

3.

4.

5.

6.

7.

8.

9.

10.

Query number: _____

IR system: _____

Your list of themes goes here:

1.

2.

3.

4.

5.

6.

7.

8.

9.

10.

Post-Search Questionnaire

Subject number: _____ IR System: _____

Please evaluate the IR system on the following dimensions:

1. The user interface is

1…...2…..3…..4…..5
confusing easy

2. How easy / difficult to locate **useful** information using this IR system?

1…..2…..3…..4…..5
difficult easy

3. How certain are you about the answers you just provided

1…..2…..3…..4…..5
least most

4. How satisfied are you with the search you just performed

1…..2…..3…..4…..5
least most

5. Do you learn anything about the subject you searched for?

1…..2…..3…..4…..5
very little A lot

6. (**for CancerSpider only**) Which of the following component(s) best assist you in searching for **useful** information?
   1. Good URLs
   2. Phrase Selection
   3. Phrase Document
   4. SOM map
   5. Region Document

7. Comment: _____

C.2 Instructions for Experimenters

Procedures:

1. Before subject comes in, bring up the two systems to desktop ready for use.
2. Greetings and have subjects fill out the payment sheet (1min)
3. Briefly briefing the subject the purpose of the experiment and what the experiment is about – **have them help us evaluate two cancer-related search engines**. (1min)
4. Give out instruction sheet, allow 3 minutes for subjects to read the task description
5. Quick demo using the same search query **colon cancer**
6. Subjects test use the system following the instruction step by step. Make sure subjects themselves be the driver. Explain the functionality of each component, particularly NP, SOM and the interactive process. Make sure subjects use the "suggested term" function of NLM Gateway http://gateway.nlm.nih.gov/gw/Cmd?GMBasicSearch (10min)
7. Give out question sheet, remind them to do think aloud.
8. Searching (40 min)
9. Set up the right system for any given search task (specification on subjects' question sheet)
   - **record the starting time**
   - while they were searching, record the both of the qualitative and quantitative data on the user log, **record end time**
10. Complete user log by yourself.
11. Repeat step 4-6 for another search task.
12. Give your subject the questionnaire to complete (5 min)
13. Turn in stapled individual package

Turn In:
Each individual subject's *answer sheet* including themes written and questionnaire, stapled with *user log* you completed.

- **CS: CancerSpider**
- **GW: NLM Gateway**

User Log

Subject number: _____ Query number: _____

IR system: _____ Search time: _____Minutes

CancerSpider					
Search Component	User-friendliness (Y/N)	Useful-ness (y/N)	subject's comment	# of documents browsed	# of documents helpful
Query Page				N/P	N/P
Search Page					
Fetch Page					
Phrase Selection				N/P	N/P
Phrase Document					
SOM Map				N/P	N/P
Region Document					

User Log

Subject number: _____ Query number: _____

IR system: _____ Search time: _____Minutes

NLM Gateway					
Use of **Find Terms**	User-friendliness (Y/N)	Usefulness (Y/N)	subject's comment	# of documents browsed	# of documents helpful

C.3 Rotation of Search Tools and Search Topics

Search tools: CS: Cancer Spider
NLM: NLM Gateway

Subject	*First Experiment*	*Second Experiment*
1	CS1	NLM2
2	NLM1	CS2
3	CS2	NLM3
4	NLM2	CS3
5	CS3	NLM4
6	NLM3	CS4
7	CS4	NLM5
8	NLM4	CS5
9	CS5	NLM6
10	NLM5	CS6
11	CS6	NLM1
12	NLM6	CS1
13	CS1	NLM3
14	NLM1	CS3
15	CS2	NLM4
16	NLM2	CS4
17	CS3	NLM5
18	NLM3	CS5
19	CS4	NLM6
20	NLM4	CS6
21	CS5	NLM1
22	NLM5	CS1
23	CS6	NLM2
24	NLM6	CS2
25	CS1	NLM4
26	NLM1	CS4
27	CS2	NLM5
28	NLM2	CS5
29	CS3	NLM6
30	NLM3	CS6

APPENDIX D: DOCUMENTS FOR THE COLLABORATIVE SPIDER EXPERIMENT

D.1 Instructions for Experiment Participators

Imagine yourself as in a consultant group who has been hired to do some research or investigation on a certain subject. You might or might not have the domain knowledge, but it does not really matters because you will be given Information Retrieval tools: search engines to accomplish your goal. By searching or browsing on search engines, you should strive to gain a decent understanding and overview about the subject matter.

While searching and browsing, you should

- Notice the aspects and constituent themes of the subjects you found in the retrieved documents
- Select and record a short phrase describing each theme found

Since your clients are busy business executives who are only interested in a short, high-level briefing, your deliverable should be a list of the themes and aspects you found that would capture your search results.

There is no time constraint on your search. The only criterion is that when you feel you have got a good understanding of the subject.

You will be given an answer sheet with topics on it. *<title>* is the subject you need to search for. *<description>* provides specific guidelines for searching, to which your searching should comply with. *<suggested query format>* provides candidate queries to enter. Feel free to use your own, but we suggest using more than one query for each search for better results. In some cases, you can also review the search sessions performed by other users.

We strongly encourage you to think out loud your like and dislike about the search tool where searching. After each search, you are requested to complete a short questionnaire about the searching system.

Do you have questions about:

- What themes and aspects are?
- What your task is?

Good luck and good surfing!

Post-Search Questionnaire

Subject number: _____

Please answer the following questions about the experiment.

1. How easy / difficult is it to locate useful information using this search system?
   1.....2.....3.....4.....5
   difficult easy

2. Do you learn anything about the subject you searched for?
   1.....2.....3.....4.....5
   very little a lot

3. Do you think it is useful to view the search sessions of other users?
   1.....2.....3.....4.....5
   not useful very useful

4. Do you think it is useful to use the *URLs* of other users?
   1.....2.....3.....4.....5
   not useful very useful

5. Do you think it is useful to use the *keywords* of other users?
   1.....2.....3.....4.....5
   not useful very useful

6. Do you think other users' *comments* on search sessions are useful?
   1.....2.....3.....4.....5
   not useful very useful

7. Which approach do you tend to try first in your search?
   A. Use my own starting URLs and keywords
   B. Use my own starting URLs and keywords, plus some starting URLs and keywords from previous session
   C. Use other people's starting URLs and keywords and do a new search
   D. Just load the previous search sessions and view the results

8. Which approach do you do think is the most efficient?
   A. Use my own starting URLs and keywords
   B. Use my own starting URLs and keywords, plus some starting URLs and keywords from previous session
   C. Use other people's starting URLs and keywords and do a new search
   D. Just load the previous search sessions and view the results

9. Please describe your search strategy.

_____

_____

_____

_____

10. Is there any other additional comment you want to make?

_____

_____

_____

_____

Thank you very much for your participation!

D.2 Instructions for Experimenters

Subject number: _____ Experimenter: _____
Machine Used: ai24 ai25 ai35

First Query

Query number: _____ Number of Previous Sessions: _____

Search Engine(s) Used: _____

Dashboard

1. How many times did the subject click the "Start" button on CI Spider?
   0 1 2 3 4 5 6 7 8 9 10
2. How many times did the subject obtain the URLs from Search Engine?
   0 1 2 3 4 5 6 7 8 9 10
3. How many times did the subject add URLs from dashboard?
   0 1 2 3 4 5 6 7 8 9 10
4. How many times did the subject add keywords from dashboard?
   0 1 2 3 4 5 6 7 8 9 10
5. How many times did the subject load search sessions of other subjects?
   0 1 2 3 4 5 6 7 8 9 10

Time

6. Total search time: _____ minutes
7. How much time did the subject spend on getting URLs from Search Engine?
   _____ minutes
8. How much time did the subject spend on browsing the dashboard?
   _____ minutes
9. How much time did the subject spend on browsing previous search sessions?
   _____ minutes
10. How much time did the subject spend on browsing his/her own search session(s)?
    _____ minutes

(Note: Part (6) = Sum of Part (7) to Part (10))

Additional Notes:

_____

_____

_____

Second Query

Query number: _____ Number of Previous Sessions: _____

Search Engine(s) Used: _____

Dashboard

1. How many times did the subject click the "Start" button on CI Spider?
   0 1 2 3 4 5 6 7 8 9 10
2. How many times did the subject obtain the URLs from Search Engine?
   0 1 2 3 4 5 6 7 8 9 10
3. How many times did the subject add URLs from dashboard?
   0 1 2 3 4 5 6 7 8 9 10
4. How many times did the subject add keywords from dashboard?
   0 1 2 3 4 5 6 7 8 9 10
5. How many times did the subject load search sessions of other subjects?
   0 1 2 3 4 5 6 7 8 9 10

Time

6. Total search time: _____ minutes
7. How much time did the subject spend on getting URLs from Search Engine?
   _____ minutes
8. How much time did the subject spend on browsing the dashboard?
   _____ minutes
9. How much time did the subject spend on browsing previous search sessions?
   _____ minutes
10. How much time did the subject spend on browsing his/her own search session(s)?
    _____ minutes

(Note: Part (6) = Sum of Part (7) to Part (10))

Additional Notes:

_____

_____

_____

_____

Third Query

Query number: _____ Number of Previous Sessions: _____

Search Engine(s) Used: _____

Dashboard

1. How many times did the subject click the "Start" button on CI Spider?
0 1 2 3 4 5 6 7 8 9 10
2. How many times did the subject obtain the URLs from Search Engine?
0 1 2 3 4 5 6 7 8 9 10
3. How many times did the subject add URLs from dashboard?
0 1 2 3 4 5 6 7 8 9 10
4. How many times did the subject add keywords from dashboard?
0 1 2 3 4 5 6 7 8 9 10
5. How many times did the subject load search sessions of other subjects?
0 1 2 3 4 5 6 7 8 9 10

Time

6. Total search time: _____ minutes
7. How much time did the subject spend on getting URLs from Search Engine?
_____ minutes
8. How much time did the subject spend on browsing the dashboard?
_____ minutes
9. How much time did the subject spend on browsing previous search sessions?
_____ minutes
10. How much time did the subject spend on browsing his/her own search session(s)?
_____ minutes

(Note: Part (6) = Sum of Part (7) to Part (10))

Additional Notes:

_____

_____

_____

_____

D.3 Rotation of Search Groups and Search Topics

	No previous sessions (no collab.)	1 previous sessions	2 previous sessions	3 previous sessions	4 previous sessions	5 previous sessions
Topic 1	Subject 1	Subject 11	Subject 21	Subject 31	Subject 41	Subject 51
Topic 2	Subject 1	Subject 11	Subject 21	Subject 31	Subject 41	Subject 51
Topic 3	Subject 1	Subject 11	Subject 21	Subject 31	Subject 41	Subject 51
Topic 4	Subject 2	Subject 12	Subject 22	Subject 32	Subject 42	Subject 52
Topic 5	Subject 2	Subject 12	Subject 22	Subject 32	Subject 42	Subject 52
Topic 6	Subject 2	Subject 12	Subject 22	Subject 32	Subject 42	Subject 52
Topic 1	Subject 3	Subject 13	Subject 23	Subject 33	Subject 43	Subject 53
. . .	. . .	. . .	. . .	. . .	. . .	. . .
Topic 6	Subject 10	Subject 20	Subject 30	Subject 40	Subject 50	Subject 60

REFERENCES

Amitay, E. (1998). "Using Common Hypertext Links to Identify the Best Phrasal Description of Target Web Documents," in *Proceedings of the ACM SIGIR'98 Post-Conference Workshop on Hypertext Information Retrieval for the Web*, Melbourne, Australia, 1998.

Arasu, A., Cho, J., Garcia-Molina, H., Paepcke, A. and Raghavan, S. (2001). "Searching the Web," *ACM Transactions on Internet Technology*, 1(1), 2-43.

Armstrong, R., Freitag, D., Joachims, T. and Mitchell, T. (1995). "WebWatcher: A Learning Apprentice for the World Wide Web," in *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments,* Mar 1995.

Baeza-Yates, R. and Pino, J. A. (1997). "A First Step to Formally Evaluate Collaborative Work," in *Proceedings of the of the International ACM SIGGROUP Conference on Supporting Group Work: The Integration Challenge*, Phoenix, Arizona, Nov 1997.

Balabanovic, M. and Shoham, Y. (1995). "Learning Information Retrieval Agents: Experiment with Web Browsing," in *Proceedings of the AAAI-95 Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments*, Stanford, California, USA, 1995.

Balabanovic, M. and Shoham, Y. (1997). "Fab: Content-based, Collaborative Recommendation," *Communications of the ACM*, 40(3), 66-72.

Baluja, S., Mittal, V. and Sukthankar, R. (1999). "Applying Machine Learning for High Performance Named-entity Extraction," in *Proceedings of the Conference of the Pacific Association for Computational Linguistics*, 1999.

Baujard, O., Baujard, V., Aurel, S., Boyer, C., and Appel, R. D. (1998). "Trends in Medical Information Retrieval on the Internet," *Computers in Biology and Medicine*, 28, pp. 589-601.

Belew, R. K. (1989). "Adaptive Information Retrieval: Using a Connectionist representation to Retrieve and Learn about Documents," in *Proceedings of the 12th ACM-SIGIR Conference*, Cambridge, MA, June 1989.

Ben-Shaul, I., Herscovici, M., Jacovi, M., Maarek, Y. S., Pelleg, D., Shtalhaim, M., Soroka, V., and Ur, S. (1999). "Adding Support for Dynamic and Focused Search with Fetuccino," in Proceedings of the 8th World Wide Web Conference, Toronto, May 1999.

Berendt, B., Hotho, A. and Stumme, G. (2002) "Towards Semantic Web Mining," in *Proceedings of the First International Semantic Web Conference*, Sardinia, Italy, Jun 2002, pp. 264-278, Springer, 2002.

Berners-Lee, T., Hendler, J. and Lassila, O. (2001). "The Semantic Web," *Scientific American*, 284(5), 35-43.

Bharat, K. and Henzinger, M. R. (1998). "Improved Algorithms for Topic Distillation in a Hyperlinked Environment," *Proceedings of the 21st ACM-SIGIR Conference*, Melbourne, Australia, 1998.

Bin, L. and Lun, K. C. (2001). "The Retrieval Effectiveness of Medical Information on the Web," *International Journal of Medical Informatics*, 62, 155-163.

Borodogna, G. and Pasi, G. (2001). "A User-adaptive Indexing Model of Structured Documents," in *Proceedings of the 10th IEEE International Conference on Fuzzy Systems Vol. 2,* Piscataway, NJ: IEEE, pp. 984-989.

Borthwick, A., Sterling, J., Agichtein, E., and Grishman, R. (1998). "NYU: Description of the MENE Named Entity System as Used in MUC-7," in *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, April 1998.

Brin, S. and Page, L. (1998). "The Anatomy of a Large-Scale Hypertextual Web Search Engine", *Proceedings of the 7th WWW Conference*, Brisbane, Australia, Apr 1998.

Broder, A., Kumar, R., Maghoul, F., Raghavan, P., Rajagopalan, S., Stata, R., Tomkins, A., and Wiener, J. (2000). "Graph Structure in the Web," in *Proceedings of the 9th International World Wide Web Conference*, Amsterdam, Netherlands, May 2000.

Buchner, A. and Mulvenna, M. D. (1998). "Discovering Internet Marketing Intelligence through Online Analytical Web Usage Mining," *SIGMOD Record*, 27(4), 54-61.

Carbonell, J. G. Michalski, R. S., Mitchell, T. M. (1983). "An Overview of Machine Learning," in R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine Learning, An Artificial Intelligence Approach*, Palo Alto, CA: Tioga.

Carrière, J. and Kazman R. (1997). "WebQuery: Searching and Visualizing the Web through Connectivity," *Proceedings of the 6th World Wide Web Conference*, Santa Clara, California, Apr 1997.

Chakrabarti, S. (2000). "Data Mining for Hypertext: A Tutorial Survey," SIGKDD Explorations, 1(1), 1-11.

Chakrabarti, S., Dom, B., and Indyk, P. (1998). "Enhanced Hypertext Categorization Using Hyperlink," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, Seattle, Washington, USA, Jun 1998.

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

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

Chang, C. H. & Lui, S. C. (2001). "IEPAD: Information Extraction based on Pattern Discovery," in *Proceedings of the 10th World Wide Web Conference*, Hong Kong, May 2001.

Chau, M. (2002). "Spidering and Filtering Web Pages for Vertical Search Engines," in *Proceedings of The Americas Conference on Information Systems*, AMCIS 2002 Doctoral Consortium, Dallas, Texas, August 8-11, 2002.

Chau, M., Atabakhsh, H., Zeng, D., and Chen, H. (2001a). "Building an Infrastructure for Law Enforcement Information Sharing and Collaboration: Design Issues and Challenges," in *Proceedings of the National Conference for Digital Government Research (dg.o 2001)*, Los Angeles, California, May 2001.

Chau, M. and Chen, H. (2003) "Comparison of Vertical Search Spiders: BFS, PageRank, Hopfield Net," *IEEE Computer*, forthcoming.

Chau, M., Chen, H., Qin, J., Zhou, Y., Qin, Y., Sung, W. K., McDonald, D. (2002a). "Comparison of Two Approaches to Building a Vertical Search Tool: A Case Study in the Nanotechnology Domain," in *Proceedings of The Second ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'02)*, Portland, Oregon, July 2002, pp. 135-144.

Chau, M., Chen, H., Qin, J., Zhou, Y., Sung, W. K., Chen, M., Qin, Y., McDonald, D., Lally, A., Landon, M. (2002b). "NanoPort: A Web Portal for Nanoscale Science and Technology," in *Proceedings of The Second ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'02)*, Portland, Oregon, July 2002.

Chau, M., Zeng, D., and Chen, H. (2001b). "Personalized Spiders for Web Search and Analysis," in *Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'01)*, Roanoke, Virginia, June 2001, pp. 79-87.

Chau, M., Zeng, D., Chen, H., Huang, M., and Hendriawan, D. (2003). "Design and Evaluation of a Multi-agent Collaborative Web Mining System," *Decision Support Systems*, Special Issue on Web Retrieval and Mining, 35(1), 167-183.

Chen, H. (1994). "Collaborative Systems: Solving the Vocabulary Problem," *IEEE Computer*, Special Issue on Computer-Supported Cooperative Work (CSCW), 27(5), 58-66.

Chen, H. (1995). "Machine Learning for Information Retrieval: Neural Networks, Symbolic Learning, and Genetic Algorithms," *Journal of the American Society for Information Science*, 46(3), 194-216, 1995.

Chen, H. (2001). *Knowledge Management Systems: A Text Mining Perspective*, The University of Arizona.

Chen, H., Chau, M., and Zeng, D. (2002). "CI Spider: A Tool for Competitive Intelligence on the Web," *Decision Support Systems*, 34(1), 1-17.

Chen, H., Chung, Y. Ramsey, M., and Yang, C. (1998a). "A Smart Itsy-bitsy Spider for the Web," *Journal of the American Society for Information Science*, Special Issue on AI Techniques for Emerging Information Systems Applications, 49(7), 604-618.

Chen, H., Chung, Y. Ramsey, M., and Yang, C. (1998b). "An Intelligent Personal Spider (Agent) for Dynamic Internet/Intranet Searching," *Decision Support Systems*, 23 (1998), 41-58.

Chen, H., Fan, H., Chau, M., and Zeng, D. (2001). "MetaSpider: Meta-Searching and Categorization on the Web," *Journal of the American Society for Information Science and Technology*, 52(13), 1134-1147.

Chen, H., Houston, A., Sewell, R. and Schatz, B. (1998c). "Internet Browsing and Searching: User Evaluations of Category Map and Concept Space Techniques," *Journal of the American Society for Information Science*, Special Issue on "AI Techniques for Emerging Information Systems Applications," 49(7), 582-603.

Chen, H., Lally, A. M., Zhu, B. and Chau, M. (2003). "HelpfulMed: Intelligent Searching for Medical Information over the Internet," *Journal of the American Society for Information Science and Technology (JASIST)*, 54(7), 683-694.

Chen, H. and Ng, T. (1995). "An Algorithmic Approach to Concept Exploration in a Large Knowledge Network (Automatic Thesaurus Consultation): Symbolic Brand-and Bound Search vs. Connectionist Hopfield Net Activation," *Journal of the American Society for Information Science*, 46(5), pp. 348-369.

Chen, H., Schuffels, C. and Orwig, R. (1996) "Internet Categorization and Search: A Machine Learning Approach," *Journal of Visual Communication and Image Representation*, Special Issue on Digital Libraries, 7(1), 88-102.

Chen, H., Shankaranarayanan, G., Iyer, A. and She, L. (1998d). "A Machine Learning Approach to Inductive Query by Examples: An Experiment Using Relevance Feedback, ID3, Genetic Algorithms, and Simulated Annealing," *Journal of the American Society for Information Science*, 49(8), 693-705, June 1998.

Chen, H. M. and Cooper, M. D. (2001). "Using Clustering Techniques to Detect Usage Patterns in a Web-Based Information System," *Journal of the American Society for Information Science and Technology*, 52(11), 888-904.

Chen, J., Mikulcic, A., and Kraft, D. H. (2000). "An Integrated Approach to Information Retrieval with Fuzzy Clustering and Fuzzy Inferencing," in O. Pons, M. A. Vila, and J. Kacprzyk (Eds.), *Knowledge Management in Fuzzy Databases*, Heidelberg, Germany: Physica-Verlag, pp.247-260.

Chen, Y. J. and Soo, V. W. (2001). "Ontology-based Information Gathering Agents," in *Proceedings of the 1st Asia-Pacific Conference on Web Intelligence*, Maebashi City, Japan, Oct 2001, pp. 423-427.

Cheong, F. C. (1996). *Internet Agents: Spiders, Wanderers, Brokers, and Bots*, New Riders Publishing, Indianapolis, Indiana, USA.

Chien, L. F. (1997). "PAT-Tree-based Adaptive Keyphrase Extraction for Intelligent Chinese Information Retrieval," in *Proceedings of the 20th ACM-SIGIR Conference*, Philadelphia, PA, 1997, pp. 50-58.

Chinchor, N. A. (1998). "Overview of MUC-7/MET-2," in *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, April 1998.

Cho, J., Garcia-Molina, H., Page, L. (1998). "Efficient Crawling through URL Ordering," *Proceedings of the 7th WWW Conference*, Brisbane, Australia, Apr 1998.

Christel, M. G., Cubilo, P., Gunaratne, J., Jerome, W., O, E. J., Solanki, S. (2002). "Evaluating a Digital Video Library Web Interface," in *Proceedings of the 2nd ACM-IEEE Joint Conference on Digital Libraries*, Portland, Oregon, USA, July 2002.

Church, K. (1988). "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Glasgow, 1989.

Church, K. and Yamamoto, M. (2001). "Using Suffix Arrays to Compute Term Frequency and Document Frequency for All Substrings in a Corpus," *Computational Linguistics*, 27(1), 1-30.

Cohen, E., Krishnamurthy, B. and Rexford, J. (1998). "Improving End-to-end Performance of the Web using Server Volumes and Proxy Filters," in *Proceedings of the ACM SIGCOMM*, 1998, pp. 241-253.

Cohen, P. R. and Feigenbaum, E. A. (1982). *The Handbook of Artificial Intelligence: Volume III*, Reading, MA: Addison-Wesley.

Cohen, W. W. (1995). "Text Categorization and Relational Learning," in *Proceedings of the 12th International Conference on Machine Learning (ICML'95)*. Morgan Kaufmann.

Cohn, D. and Chang, H. (2000) "Learning to Probabilistically Identify Authoritative Documents," in *Proceedings of the 17th International Conference on Machine Learning*, Stanford, CA, USA, Jun-Jul 2000.

Cooley, R., Mobasher, B. and Srivastava, J. (1997). "Web Mining: Information and Pattern Discovery on the World Wide Web," in *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence*, Newport Beach, CA, USA, Nov 1997, pp. 558-567.

Cox, T. F. and Cox, M. A. A. (1994). *Multidimensional Scaling*, Chapman & Hall, London, 1994.

Crimmins, F., Smeaton, A. F., Dkaki, T., and Mothe, J. (1999). "TetraFusion: Information Discovery on the Internet," *IEEE Intelligent System*, 1999 Jul-Aug, 55-62.

Cunningham, S. J., Witten, I. H., & Littin, J. (1999). "Applications of Machine Learning in Information Retrieval," in M. E. Williams (ed), *Annual Review of Information Science and Technology*, 34, 1999, Medford, NJ: Information Today, pp. 341-384.

DeBra, P. and Post, R. (1994). "Information Retrieval in the World-Wide Web: Making Client-based Searching Feasible," in *Proceedings of the First International World Wide Web Conference*, Geneva, Switzerland, 1994.

Diligenti, M., Coetzee, F., Lawrence, S., Giles, C. L., and Gori, M. (2000). Focused Crawling using Context Graphs. In *Proceedings of the 26th International Conference on Very Large Databases* (VLDB 2000), Cairo, Egypt, pp. 527–534.

Dodge, M. and Kitchin, R. (2001). *Atlas of Cyberspace*. Addison-Wesley.

Doorenbos, R. B., Etzioni, O., and Weld, D. S. (1997). "A Scalable Comparison-Shopping Agent for the World-Wide Web," in *Proceedings of the First International Conference on Autonomous Agents*, Marina del Rey, California, USA, Feb 1997, pp. 39-48.

Duda, R. and Hart, P. (1973). *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

Dumais, S. T., Platt, J., Heckerman, D. and Sahami, M. (1998). "Inductive Learning Algorithms and Representations for Text Categorization," in *Proceedings of the ACM*

Conference on Information and Knowledge Management, Bethesda, Maryland, Nov 1998, pp. 148-155.

Dwork, C., Kumar, R., Noar, M. and Sivakumar, D. (2001). "Rank Aggregation Methods for the Web," in *Proceedings of the 10th International World Wide Web Conference*, Hong Kong, May 2001.

Etzioni, O. (1996). "The World Wide Web: Quagmire or Gold Mine," *Communications of the ACM*, 39(11), 65-68.

Fensel, D. and Musen, M. A. (2001). "The Semantic Web: A Brain for Humankind," *IEEE Intelligent Systems*, 16(2), 24-25.

Finin, T., Fritzson, R., and McKay, D. (1992). "A Language and Protocol to Support Intelligent Agent Interoperability," in *Proceedings of the CE and CALS Washington 92 Conference*, Jun 1992.

Finin, T., Fritzson, R., McKay, D., and McEntire, R. (1994). "KQML as an Agent Communication Language," in *Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94)*, Nov 1994.

Fisher, D. H. (1987). "Knowledge Acquisition via Incremental Conceptual Clustering," *Machine Learning*, 2, 139-172.

Fogel, D. B. (1994). An Introduction to Simulated Evolutionary Optimization. *IEEE Transactions on Neural Networks*, 5, 3-14.

Fox, E., Hix, D., Nowell, L. T., Brueni, D. J., Wake, W. C., Lenwood, S. H. and Rao, D. (1993). "Users, User Interfaces, and Objects: Envision, A Digital Library," *Journal of the American Society for Information Science*, 44(8), 480-491.

Frécon, E. and Smith, G. (1998). "WebPath - A Three-dimensional Web History," *Proceedings of the IEEE Symposium on Information Visualization*, Chapel Hill, NC, USA.

Fuhr, N. & Buckley, C. (1991). "A Probabilistic Learning Approach for Document Indexing," *ACM Transactions on Information Systems*, 9, 223-248.

Fuhr, N. & Pfeifer, U. (1994). "Probabilistic Information Retrieval as a Combination of Abstraction, Inductive Learning, and Probabilistic Assumption," *ACM Transactions on Information Systems*, 12(1), 92-115.

Furnas, G. W., Landauer, T. K., Gomez, L. M., and Dumais, S. T. (1987). "The Vocabulary Problem in Human-System Communication," *Communications of the ACM*, 30(11), 964-971.

Furnkranz, J. (1999). "Exploiting Structural Information for Text Categorization on the WWW," in *Proceedings of the 3rd Symposium on Intelligent Data Analysis (IDA'99)*, Amsterdam, Netherlands, pp. 487-497.

Gauch, S., Wang, G., and Gomez, M. (1996). "Profusion: Intelligent Fusion from Multiple Different Search Engines," *Journal of Universal Computer Science*, 2(9).

Ginsburg, M. (1998). "Annotate! A Tool for Collaborative Information Retrieval," in *Proceedings of the 7th IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE'98)*, IEEE CS, Los Alamitos, California, 1998, 75-80.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley.

Goldberg, D., Nichols, D., Oki, B. and Terry, D. (1992). "Using Collaborative Filtering to Weave an Information Tapestry," *Communications of the ACM*, 35(12), 61-69.

Green, C. L. and Edwards, P. (1996). "Using Machine Learning to Enhance Software Tools for Internet Information Management," in *Proceedings of the AAAI-96 Workshop on Internet-Based Information Systems*, Menlo Park, CA: AAAI, pp. 48-55.

Han, J. and Chang, K. C. (2002). "Data Mining for Web Intelligence," *IEEE Computer*, 35(11), 64-70.

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

Hayes-Roth, F. & Jacobstein, N. (1994). "The State of Knowledge-based Systems," *Communications of the ACM*, 37, 27-39.

He, X., Zha, H., Ding, C., and Simon, H. (2002). "Web Document Clustering Using Hyperlink Structures," *Computational Statistics and Data Analysis*, forthcoming.

Hearst, M. (1995). "TileBars: Visualization of Term Distribution Information in Full Text Information Access," in *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI'95)*, 59-66.

Hearst, M. (1999). "Untangling Text Data Mining," in *Proceedings of ACL'99: the 37th Annual Meeting of the Association for Computational Linguistics*, Maryland, June 20-26.

Hearst, M. A. and Pederson, J. O. (1996). "Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results," in *Proceedings of the 19th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR'96)*, Zurich, Switzerland, Aug 1996.

Hendley, R. J., Drew, N. S., Wood, A. and Beale, R. (1995). "Narcissus: Visualizing Information," *Proceedings of the 1995 Information Visualization Symposium*, Atlanta, GA, 90-96, 1995.

Hersh, W. R. (1996). *Information Retrieval: A Health Care Perspective*. Berlin, Germany: Springer-Verlag.

Hill, D. R. (1968). "A Vector Clustering Technique," in Samuelson (Ed.), *Mechanized Information Storage, Retrieval and Dissemination*, North-Holland, Amsterdam, 1968.

Hopfield, J. J. (1982). "Neural Network and Physical Systems with Collective Computational Abilities," *Proceedings of the National Academy of Science,* USA, 1982, 79(4), pp. 2554-2558.

Howe, A. E. and Dreilinger, D. (1997). "SavvySearch: A Meta-Search Engine that Learns which Search Engines to Query," *AI Magazine*, 18(2), 19 - 25.

Huang, M. L., Eades, P. and Cohen, R. F. (1998). "WebOFDAV - Navigating and Visualizing the Web On-line with Animated Context Swapping," in *Proceedings of the 7th WWW Conference*, Brisbane, Australia, Apr 1998.

Huang, Z., Chung, W., Ong, T. H. and Chen, H. (2002). "A Graph-based Recommender System for Digital Library," in *Proceedings of the 2nd ACM-IEEE Joint Conference on Digital Libraries*, Portland, Oregon, USA, July 2002.

Hurst, M. (2001). "Layout and language: Challenges for Table Understanding on the Web," in *Proceedings of the 1st International Workshop on Web Document Analysis*, Seattle, WA, USA, September 2001, pp. 27-30.

Ide, E. (1971). "New Experiments in Relevance Feedback," in G. Salton (ed.), *The SMART Retrieval System – Experiments in Automatic Document Processing*, Englewood Cliffs, NJ: Prentice-Hall, pp. 337-354.

Iwayama, M. and Tokunaga, T. (1995). "Cluster-based Text Categorization: A Comparison of Category Search Strategies," in *Proceedings of the 18th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR'95)*, Seattle, Washington, July 1995, pp. 273-281.

Jansen, B. J., Spink, A., and Saracevic, T. (2000). "Real Life, Real Users and Real Needs: A Study and Analysis of Users Queries on the Web," *Information Processing and Management*, 36(2), 207-227.

Jeon, H., Petrie C., and Cutkosky, M. (2000). "JATLite: A Java Agent Infrastructure with Message Routing," *IEEE Internet Computing*, 4(2), 87-96.

Jennings, N., Sycara, K., and Wooldridge, M. (1998). "A Roadmap of Agent Research and Development," *Autonomous Agents and Multi-Agent Systems*, 1, 7-38.

Joachims, T. (1998). "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," in *Proceedings of the European Conference on Machine Learning*, Berlin, 1998, pp. 137-142.

Joachims, T. (1999). "Making large-Scale SVM Learning Practical," in B. Schölkopf and C. Burges and A. Smola (eds.), *Advances in Kernel Methods - Support Vector Learning*, MIT-Press.

Joachims, T., Chistianini, N., Shawe-Taylor, J. (2001). "Composite Kernels for Hypertext Categorization," in *Proceedings of the 18th International Conference on Machine Learning (ICML'01)*, 2001.

Kantor, P. B.,. Boros, E., Melamed, B., Meñkov, V., Shapira, B. and Neu, D. J. (2000). "Capturing Human Intelligence in the Net," *Communications of the ACM*, 43(8), 112-115.

Karamuftuoglu, M. (1998). "Collaborative Information Retrieval: Toward a Social Informatics View of IR Interaction," *Journal of the American Society for Information Science*, 49(12), 1070-1080.

Keonemann, J. and Belkin, N. (1996). "A Case for Interaction: a Study of Interactive Information Retrieval Behavior and Effectiveness," in *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI'96)*, pp. 205-212.

Kiley, R. (1999). *Medical Information on the Internet: A Guide for Health Professionals*. London: Churchill Livingstone.

Kleinberg, J. (1998). "Authoritative Sources in a Hyperlinked Environment," in *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, California, USA, Jan 1998, pp. 668-677.

Kohavi, R. (1995). "A Study of Cross-validation and Bootstrap for Accuracy Estimation and Model Selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, San Francisco, CA, 1995, Morgan Kaufmann, pp. 1137-1143.

Kohonen, T. (1995). *Self-organizing Maps*, Springer-Verlag, Berlin.

Kohonen, T. (1997). "Exploration of Very Large Databases by Self-Organizing Maps," in *Proceedings of the IEEE International Conference on Neural Networks*, 1-6.

Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V., and Saarela, A. (2000). "Self Organization of a Massive Document Collection," *IEEE Transactions on*

Neural Networks, Special Issue on Neural Networks for Data Mining and Knowledge Discovery, 11(3), 574-585. May 2000.

Koller, D. and Sahami, M. (1997). "Hierarchically Classifying Documents Using Very Few Words," in *Proceedings of the 14th International Conference on Machine Learning (ICML'97)*, 1997, pp. 170-178.

Kononenko, I. (1993). "Inductive and Bayesian learning in medical diagnosis," *Applied Artificial Intelligence*, 7, 317-337, 1993.

Konstan, J. A., Miller, B., Maltz, D., Herlocker, J., Gordon, L. and Riedl, J. (1997). "GroupLens: Applying Collaborative Filtering to Usenet News," *Communications of the ACM*, 40(3), 77-87.

Kosala, R. and Blockeel, H. (2000). "Web Mining Research: A Survey," *ACM SIGKDD Explorations*, 2(1), 1-15.

Kraft, D. H., Bordogna, G., and Pasi, G., "Fuzzy Set Techniques in Information Retrieval," in Bezdek, J. C., Didier, D. and Prade, H. (Eds.), *Fuzzy Sets in Approximate Reasoning and Information Systems*, 3, The Handbook of Fuzzy Sets Series, Norwell, MA: Kluwer Academic Publishers, 1999

Kraft, D. H., Petry, F. E., Buckles, B. P. and Sadasivan, T. (1995). "Applying Genetic Algorithms to Information Retrieval Systems via Relevance Feedback," in P. Bosc & J. Kacprzyk (Eds.), *Fuzziness in Database Management Systems*, Heidelberg, Germany: Physica-Verlag, pp. 330-344.

Kraft, D. H., Petry, F. E., Buckles, B. P. and Sadasivan, T. (1997). "Genetic Algorithms for Query Optimization in Information Retrieval: Relevance Feedback," in E Sanchez, T. Shibata, & L. A. Zadeh (Eds.), *Genetic Algorithms and Fuzzy Logic Systems*, Singapore: World Scientific, pp. 155-173.

Kwok, K. L. (1989). "A Neural Network for Probabilistic Information Retrieval," *Proceedings of the 12th ACM-SIGIR Conference on Research and Development in Information Retrieval*, Cambridge, Massachusetts, USA, Jun 1989, pp.21-30.

Lagus, K., Honkela, T., Kaski, S., and Kohonen, T. (1999). "WEBSOM for Textual Data Mining," *Artificial Intelligence Review*, 13(5/6), 345-364.

Lam, S.L.Y, and Lee, D.L. (1999). "Feature Reduction for Neural Network Based Text Categorization," in *Proceedings of the International Conference on Database Systems for Advanced Applications (DASFAA '99)*, Hsinchu, Taiwan, Apr 1999.

Lamping, J. and Rao, R. (1996). "Visualizing Large Trees Using the Hyperbolic Browser," in *Proceedings of the ACM CHI '96 Conference on Human factors in Computing Systems*, Vancouver, Canada, Apr 1996.

Lang, K. (1995). "NewsWeeder: Learning to Filter Netnews," in *Proceedings of the 12th International Conference on Machine Learning*, San Francisco, California, 1995.

Langley, P., Iba, W. and Thompson, K. (1992). "An Analysis of Bayesian Classifiers," in *Proceedings of the 10th National Conference on Artificial Intelligence*, AAAI Press and MIT Press, pp. 223-228.

Langley, P. and Simon, H. (1995). "Applications of Machine Learning and Rule Induction," *Communications of the ACM*, 38(11), 55-64.

Lawrence, S. and Giles, C. L. (1998a). "Inquirus, the NECI Meta Search Engine," in *Proceedings of the 7th International World Wide Web Conference*, Brisbane, Australia, Apr 1998.

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

Lawrence, S. and Giles, C. L. (1999). "Accessibility of Information on the Web," *Nature*, 400, 107-109.

Lempel, R. and Moran, S. (2001). "SALSA: The Stochastic Approach for Link-structure Analysis," *ACM Transactions on Information Systems*, 19(2), 131-160, 2001.

Lewis, D. D. and Ringuette, M. (1994). "Comparison of Two Learning Algorithms for Text Categorization," in *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94)*, 1994.

Lieberman, H. (1995). "Letizia: An Agent that Assists Web Browsing," in *Proceedings of the 1995 International Joint Conference on Artificial Intelligence*, Montreal, Canada, 1995.

Lin, C., Chen, H. and Nunamaker, J. F. (2000). "Verifying the Proximity Hypothesis for Self-Organizing Maps," *Journal of Management Information Systems*, 16(3), 57-70.

Lin, X., Soergel, D., and Marchionini, G. (1991). "A Self-Organizing Semantic Map for Information Retrieval," in *Proceedings of the 14th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIRIR'91)*, 262-269, 1991.

Lippmann, R. P. (1987). An Introduction to Computing with Neural Networks. *IEEE Acoustics Speech and Signal Processing Magazine*, 4, 4-22.

Lyman, P. and Varian, H. R. (2000). "How Much Information," available at http://www.sims.berkeley.edu/how-much-info/

Maedche, A. and Staab, S. (2001). "Ontology Learning for the Semantic Web," *IEEE Intelligent Systems*, 16(2), 72-79.

Maes, P. (1994). "Agents that Reduce Work and Information Overload," *Communications of the ACM*, 37(7), 31-40.

Maniezzo V. (1994). "Genetic Evolution of the Topology and Weight Distribution of Neural Networks," *IEEE Transactions on Neural Networks*, 5(1), 39-53.

Marchionini, G. (2002). "Co-Evolution of User and Organizational Interfaces: A Longitudinal Case Study of WWW Dissemination of National Statistics," *Journal of the American Society for Information Science and Technology*, 53(14), 1192-1209.

Masand, B., Linoff, G., and Waltz, D. (1992). "Classifying News Stories Using Memory Based Reasoning," in *Proceedings of the 15th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR'92),* pp. 59-64.

McCallum, A., Nigam, K., Rennie, J., and Seymore, K. (1999). "A Machine Learning Approach to Building Domain-specific Search Engines," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-99)*, 1999, pp. 662-667.

McPherson, J. and Bainbridge, D. (2001). "Usage of the MELDEX Digital Music Library," in *Proceedings of the 2nd Annual International Symposium on Music Information Retrieval*, Bloomington, Indiana, USA, Oct 2001.

McQuaid, M., Ong, T. H., Chen, H. and Nunamaker, J. F. (1999). "Multidimensional Scaling for Group Memory Visualization," *Decision Support Systems*, 27(1-2), 163-176, November 1999.

Mendes, R. R. F., Voznika, F. B., Freitas, A. A. & Nievola, J. C. (2001). "Discovering Fuzzy Classification Rules with Genetic Programming and Co-evolution," *Principles of Data Mining and Knowledge* Discovery, *Lecture Notes in Artificial Intelligence*, 2168, pp. 314-325. Springer-Verlag, 2001.

Michalewicz, Z. (1992). *Genetic Algorithms + Data Structures =Evolution Programs.* Berlin: Springer-Verlag.

Miller, S., Crystal, M., Fox, H., Ramshaw, L. Schwartz, R., Stone, R., Weischedel, R., and the Annotation Group (1998). "BBN: Description of the SIFT system as used for MUC-7," in *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, Washington, DC, April 1998.

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

Mitchell, T. (1997). *Machine Learning*, McGraw Hill, 1997.

Mulvenna, M. D., Anand, S. S., and Büchner, A. G. (2000). "Personalization on the Web Using Web Mining," *Communications of the ACM*, 43(8), 123-125.

Najork, M. and Wiener, J. L. (2001). "Breadth-first Search Crawling Yields High-quality Pages," in *Proceedings of the 10th WWW Conference*, Hong Kong, May 2001.

Ng, H. T., Goh, W. B., and Low, K. L. (1997). "Feature Selection, Perceptron Learning, and a Usability Case Study for Text Categorization," in *Proceedings of the 20th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR'97)*, 1997, pp. 67-73.

Nunamaker, J. F., Chen, M., and Purdin, T. D. M. (1991). "Systems Development in Information Systems Research," *Journal of Management Information Systems*, 7(3), 89-106.

O'Day, V. L. and Jeffries, R. (1993). "Information Artisans: Patterns of Result Sharing by Information Searchers," in *Proceedings of the ACM Conference on Organizational Computing Systems (COOCS'93)*, Milpitas, California, Nov 1993, 98-107.

Oh, H. J., Myaeng, S. H., and Lee, M. H. (2000). "A Practical Hypertext Categorization Method Using Links and Incrementally Available Class Information," in *Proceedings of the 23rd Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR'00)*, 2000, pp. 264-271.

Ong, T. and Chen, H. (1999). "Updateable PAT-Tree Approach to Chinese Key Phrase Extraction Using Mutual Information: A Linguistic Foundation for Knowledge Management," in *Proceedings of the Second Asian Digital Library Conference*, Taipei, Taiwan, November 1999, pp. 63-84.

Orlikowski, W. (1992). "Learning from Notes: Organizational Issues in Groupware Implementation," in *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW'92)*, 1992, 362-369.

Orwig, R., Chen, H. and Nunamaker, J.F. (1997). "A Graphical Self-organizing Approach to Classifying Electronic Meeting Output," *Journal of the American Society for Information Science*, 48(2), 157-170, 1997.

Paass, G. (1990), "Probabilistic Reasoning and Probabilistic Neural Networks," in *Proceedings of the 3rd International Conference on Information Processing and Management of Uncertainty*, pp.6-8.

Petrie, C. (1996). "Agent-based Engineering, the Web, and Intelligence," *IEEE Expert*, 11(6), 24-29.

Pitkow, J. (1997). "In Search of Reliable Usage Data on the WWW," in *Proceedings of the 6th International World Wide Web Conference*, Santa Clara, CA, USA, 1997.

Porter, M. F. (1980). "An algorithm for suffix stripping," *Program*, 14(3), pp. 130-137.

Quinlan, J. R. (1983). "Learning Efficient Classification Procedures and Their Application to Chess End Games," in R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine Learning, An Artificial Intelligence Approach*, Palo Alto, CA: Tioga.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*, Los Altos, CA: Morgan Kaufmann.

Rasmussen, E. (1992). *Clustering Algorithms*, Englewood Cliffs, NJ: Prentice Hall, 1992.

Rennie, J. and McCallum, A. K. (1999). "Using Reinforcement Learning to Spider the Web Efficiently," in *Proceedings of the 16th International Conference on Machine Learning (ICML-99)*, Bled, Slovenia, pp. 335-343.

Resnik, P. (1999). "Mining the Web for bilingual text," in *Proceedings of the 37th Annual Meeting of the Association of Computational Linguistics*, College Park, Maryland, USA, Jun 1999.

Rocchio, J. J. (1966). *Document Retrieval Systems - Optimization and Evaluation*. Ph.D. Thesis, Harvard University.

Rocchio, J. J. (1971). "Relevance Feedback in Information Retrieval," in G. Salton (ed.), The SMART Retrieval System – Experiments in Automatic Document Processing, Englewood Cliffs, NJ: Prentice-Hall, pp. 337-354.

Romano, N., Roussinov, D., Nunamaker, J. F., and Chen, H. (1999). "Collaborative Information Retrieval Environment: Integration of Information Retrieval with Group Support Systems," in *Proceedings of the 32nd Hawaii International Conference on System Sciences (HICSS-32)*, 1999.

Roussinov, D., and Zhao, L. (2003). "Automatic Discovery of Similarity Relationships through Web Mining," *Decision Sup Systems*, 35(1), 149-166.

Rowe, N. (2002). "A High-recall Self-improving Web Crawler that Finds Images using Captions," *IEEE Intelligent Systems*, 17(4), 8-14.

Rumelhart, D. E., Hinton, G. E., and McClelland, J. L. (1986a). "A General Framework for Parallel Distributed Processing," in D. E. Rumelhart, J. L. McClelland, and the PDP Research Group (Eds.), *Parallel Distributed Processing*, pp. 45-76, Cambridge, MA: The MIT Press.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986b). "Learning Internal Representations by Error Propagation," in D. E. Rumelhart, J. L. McClelland, and the PDP Research Group (Eds.), *Parallel Distributed Processing*, pp. 318-362, Cambridge, MA: The MIT Press.

Salton, G. (1986). "Another Look at Automatic Text-retrieval Systems," *Communications of the ACM*, 29(7), 648-656.

Salton, G. (1989). *Automatic Text Processing*, Reading, MA: Addison-Wesley.

Samuelson, C. and Rayner, M. (1991). "Quantitative Evaluation of Explanation-based Learning as an Optimization Tool for a Large-scale Natural Language System," in *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, Sydney, Australia, 1991, pp. 609-615.

Selberg, E. and Etzioni, O. (1995). "Multi-Service Search and Comparison using the MetaCrawler," in *Proceedings of the 4th World Wide Web Conference*, Boston, MA USA, December 1995.

Selberg, E. and Etzioni, O. (1997). "The MetaCrawler Architecture for Resource Aggregation on the Web," *IEEE Expert*, 12(1), 11-14.

Shank, G. (1993). "Abductive Multiloguing, the Semiotic Dynamics of Navigating the Net," *The Arachnet Electronic Journal on Virtual Culture*, 1(1), Mar 1993.

Shardanand, U. and Maes, P. (1995). "Social Information Filtering: Algorithms for Automating 'Word of Mouth,'" in *Proceedings of the ACM Conference on Human Factors and Computing Systems*, Denver, Colorado, May 1995.

Shiozawa, H. and Matsushita Y. (1997). "WWW Visualization Giving Meanings to Interactive Manipulations," *Proceedings of HCI International '97*, San Francisco, CA, 791-794, August 1997.

Shneiderman, B. (1997). "Designing information-abundant Web sites: issues and recommendations," *International Journal of Human-Computer Studies*, 47, 5-29.

Shneiderman, B., Feldman, D., Rose, A. and Grau, X. F. (2000). "Visualizing Digital Library Search Results with Categorical and Hierarchical Axes," in *Proceedings of 5th ACM Conference on ACM 2000 Digital Libraries*, San Antonio, Texas.

Silverstein, C., Henzinger, M., Marais, H. and Moricz, M. (1999) "Analysis of a Very Large Web Search Engine Query Log," *ACM SIGIR Forum*, 33(1), 6-12.

Simon, H. A. (1983). "Why Should Machine Learn?" In R. S. Michalski, J. Carbonell, and T. M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach*,. Palo Alto, CA: Tioga Press.

Spetka, S. (1994). "The TkWWW Robot: Beyond Browsing," in *Proceedings of the 2nd International World Wide Web Conference*, Chicago, Illinois, USA, 1994.

Spink, A., Wolfram, D., Jansen, B. J., and Saracevic, T. (2001). "Searching the Web: The Public and Their Queries," *Journal of the American Society for Information Science and Technology*, 52(3), 226-234.

Spink, A. and Xu, J. (2000) "Selected Results from a Large Study of Web Searching: The Excite Study," *Information Research*, 6(1), available at http://InformationR.net/ir/6-1/paper90.html

Srivastava, J., Cooley, R., Deshpande, M., and Tan, P. N. (2000) "Web Usage Mining: Discovery and Applications of Web Usage Patterns from Web Data," *ACM SIGKDD Explorations*, 1(2), 12-23.

Starr, B., Ackerman, M. and Pazzani, M. (1996). "Do-I-Care: A Collaborative Web Agent," in *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'96)*, 273-274.

Stone, M. (1974). "Cross-validation Choices and Assessment of Statistical Predictions," *Journal of the Royal Statistical Society*, 36, 111-147.

Sycara, K. (1998). "Multi Agent Systems," *AI Magazine*, 19(2), 79-92.

Sycara, K. and Zeng, D. (1996). "Coordination of Multiple Intelligent Software Agents," *International Journal of Cooperative Information Systems*, 5(2&3), 181-211.

Trybula, W. J. (1999). "Text Mining," in M. E. Williams (ed), *Annual Review of Information Science and Technology*, 34, 1999, Medford, NJ: Information Today, pp. 385-419.

UC Berkeley (2002). "Web Term Document Frequency and Rank" [Online]. Available at http://elib.cs.berkeley.edu/docfreq/

van Rijsbergen, C. J. (1979). *Information Retrieval*. Butterworths, London, 1979. Second Edition.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer, 1995.

Vapnik, V. (1998). *Statistical Learning Theory*, Wiley, Chichester, GB, 1998.

Veerasamy, A. and Belkin, N. J. (1996). "Evaluation of a Tool for Visualization of Information Retrieval Results," in *Proceedings of the 19th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'96)*, 85-92.

Voorhees, E. M. (1985). "The Cluster Hypothesis Revisited," in *Proceedings of the 12th ACM-SIGIR Conference*, 1985, pp. 188-196.

Voorhees, E., & Harman, D. (1998). "Overview of the Sixth Text Retrieval Conference (TREC-6)," in *Proceedings of the Sixth Text Retrieval Conference (TREC-6)*, Gaithersburg, Maryland: National Institute of Standards and Technology, pp. 1-24.

Vrettos, S. and Stafylopatis, A. (2001). A Fuzzy Rule-based Agent for Web Retrieval-Filtering. In *Proceedings of the 1st Asia-Pacific Conference on Web Intelligence*, Maebashi City, Japan, Oct 2001, pp. 448-453.

Wactlar, H. D., Christel, M. G., Gong, Y., and Hauptmann, A. G. (1999). "Lessons Learned from the Creation and Deployment of a Terabyte Digital Video Library," *IEEE Computer*, 32(2), 66-73.

Wang, Y. and Hu, J. (2002). "A Machine Learning Based Approach for Table Detection on the Web," in *Proceedings of the 11th World Wide Web Conference*, Honolulu, Hawaii, May 2002.

Ward, J. (1963). "Hierarchical Grouping to Optimize an Objection Function," *Journal of the American Statistical Association*, 58, 236–244.

Wasfi, A. M. A. (1999). "Collecting User Access Patterns for Building User Profiles and Collaborative Filtering," in *Proceedings of the 1999 International Conference on Intelligent User Interfaces (IUI'99)*, 57-64, 1999.

Waterhouse, S., Doolin, D. M., Kan, G., and Faybishenko, Y. (2002). "Distributed Search in P2P Networks," *IEEE Internet Computing*, 6(1), 68-72.

Westberg, E. and Miller R. (1999). "The Basis for Using the Internet to Support the Information Needs of Primary Care," *Journal of the American Medical Informatics Association*, 6, 6-25.

Wiener, E., Pedersen, J. O., and Weigend, A. S. (1995). "A Neural Network Approach to Topic Spotting," in *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95)*, 1995.

Willet, P. (1988). "Recent Trends in Hierarchical Document Clustering: A Critical Review," *Information Processing & Management*, 24, 577-597.

Wu, M., Fuller, M. and Wilkinson, R. (2001). "Using Clustering and Classification Approaches in Interactive Retrieval," *Information Processing and Management*, 37, 459-484.

Yan, T., Javobsen, J., Garcia-Molina, H., and Dayal, U. (1996). "From User Access Patterns to Dynamic Hypertext Linkage," in *Proceedings of the 5th World Wide Web Conference*, Paris, France, May 1996.

Yang, C. C., Yen, J., and Chen, H. (2000). "Intelligent Internet Searching Agent Based on Hybrid Simulated Annealing," *Decision Support Systems*, 28, 269-277.

Yang, Y. and Liu, X. (1999). "A Re-examination of Text Categorization Methods, in *Proceedings of the 22nd Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR'99)*, 1999, pp. 42-49.

Yang, Y., Slattery, S. and Ghani, R. (2002). "A Study of Approaches to Hypertext Categorization," *Journal of Intelligent Information Systems*, 18(2), March 2002.

Yu, C., Meng, W., and Liu, K. L. (2001). "Efficient and Effective Metasearch for Text Databases Incorporating Linkages among Documents," in *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, Dallas, Texas, May 2001, 187-198.

Zacharis, Z. N. and Panayiotopoulos, T. (2001). "Web Search Using a Genetic Algorithm," *IEEE Internet Computing*, 5(2), 18-26.

Zadeh, L. A. (1965). "Fuzzy sets," *Information and Control*, 8, 338-353.

Zamir, O. and Etzioni, O. (1998) "Web Document Clustering: A Feasibility Demonstration," *Proceedings of the 21st Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR'98)*, Melbourne, Australia, Aug 1998, pp. 46-54.

Zamir, O. and Etzioni, O. (1999). "Grouper: A Dynamic Clustering Interface to Web Search Results," in *Proceedings of the 8thWorld Wide Web Conference*, Toronto, May 1999.

Zeng, D., Chen, H., Daspit, D., Shan, F., Nandiraju, S., Chau, M., and Lin, C. (2003). "COPLINK Agent: An Architecture for Information Monitoring and Sharing in Law Enforcement," in *Proceedings of the NSF/NIJ Symposium on Intelligence and Security Informatics*, Tucson, Arizona, June 2003.