SEARCH DATA PROCESSOR

This application claims benefit of the filing date of provisional application No. 60/094,694 filed Jul. 30, 1998.

This invention was made under U.S. Government Contract NROXXX-96-G-3006. The Government has certain rights in the invention.

TECHNICAL FIELD

This invention relates generally to the field of search techniques used on information management system or on the global information network ("the World Wide Web"). More specifically, the present invention is a method and system for refining and improving search queries and for organizing the results of a search query by different and overlapping criteria.

BACKGROUND OF THE INVENTION

The blossoming of the World Wide Web in the 1990s has given computer users access to vast quantities of information, an estimated 100–300 million Web pages, many terabytes of data. The user provides the Uniform Resource Locator ("URL") of a page to the browser, the browser retrieves the page from the Internet and displays it to the user. When the user knows the URL of the page, the procedure is simple. However, to find information on the Web, the user must access a search engine. The user submits a query and the search engine returns a list of URL's of pages that satisfy the query together with a summary of each page. The continuing exponential growth of the Web makes the task of finding the relevant information exceedingly difficult. This effort is further aggravated by the unorganized and extremely dynamic nature of the Web.

There are two paths to searching for information on the Web. One path is consulting a manually compiled Web catalog, such as Yahoo. Any manual catalog of the Web necessarily suffers two drawbacks: the nature of the information on the Web makes any cataloging efforts necessarily limited and incomplete, and the catalog offers no help to a user interested in a subject that happens not to be covered by the catalogers.

The other path to searching for information on the Web is using a Web engine. The major ones as of January 1998 are AltaVista, Excite, HotBot, InfoSeek, Lycos, NorthernLight, and Web Crawler, plus a number of branded versions of these. These engines send out programs called robots, or crawlers, which automatically peruse the Web and gather Web pages they discover. The collected pages are automatically indexed and collected into a data base. In this process, known as indexing, Internet URLs are associated with relevant words from the page they identify. Many search engines store page summaries along with URLs. Page summarization varies from one search engine to another. Some search engines store the first fifty words of a document. Other engines, try to understand the content of the pages. They attempt to define relevant "ideas" based on associations of words within documents and they summarize the Web Pages by storing these "ideas". The users can query the indices for pages meeting certain criteria. For example, a user can request all the Web pages found by the search engine that have the phrase "cryptography software" somewhere in the text. There are two major problems with using the search engines: 1) incomplete coverage and 2) difficulty of effective use. Not a single engine contains a complete index of the Web; they index anywhere from 2 million pages by WebCrawler to 100 million pages by AltaVista. Given the explosive growth of the Web and the limitation of time and space faced by search engines, it is unlikely that full coverage of the Web is forthcoming.

Most users feel the incompleteness of the indices only indirectly, since they can not miss a web page if they do not know it exists. The more pressing problem is that using the search engines can be a frustrating, time-consuming, and often unsuccessful process for the user. In most search sessions, the user's needs are well enough formulated in her head that only a small number of web pages would exactly meet her need. The problem then, is getting the search engine to understand the user's needs. Unfortunately, the state of the art in human-machine interaction is far from meeting such a goal. Many user queries produce unsatisfactory results, yielding thousands of matching documents. The search engine indices support many basic information retrieval queries, but the users are offered little guidance in determining which keywords and in which combination would yield the desired content. Typically, the user ends up alternating between specifying too few keywords which yield too many matching documents, and supplying too many keywords which yield no matches. Many search engines lack efficiency in eliminating duplicate URLs from their indices. As a consequence, redundant information is sometimes returned to users, and can create a lot of frustration.

While a number of tools have been developed to help the user search more intelligently, by allowing selection of additional search criteria, none of them offers useful analysis of the query results that could give guidance to the user in reformulating a more appropriate query. Some search engines group and display results based on the popularity of the site. While others attempt to do some type of organization. One such search engine, Northern Light, organizes all the query results into at most 10 folders based on subject, type, source and language. While this is a step in the right direction, the user is not given any information on how the categories are derived or on how many results are in each folder.

SUMMARY OF THE INVENTION

The present invention is embodied in a simple and effective method for improving the searching of an information management system using a search engine and for refining and organizing the search results.

The present invention provides for a query tuner, allowing a user to effectively reformulate a query in order to find a reasonable number of matching documents from the search engine by automatically and selectively modifying individual query terms in the user's query to be weaker or stronger.

One aspect of the present invention provides for a dynamic filter, using a dynamic set of record tokens to restrict the results of a search query to include only records which correspond to the record tokens.

Another aspect of the present invention provides for a results organizer, to aid the user in organizing and understanding a large number of matching documents returned in response to a search query by clustering like items returned from the search.

Another aspect of the present invention provides for a search history, to allow the user to save, organize and search the queries and the documents that best satisfy the query.

It is to be understood that both the foregoing general description and the following detailed description are exemplary, but are not restrictive, of the invention.

DESCRIPTION OF THE DRAWING

The invention is best understood from the following detailed description when read in connection with the accompanying drawings. Included in the drawing are the following Figures:

FIG. **1**A is a flowchart illustrating a high level chart of the invention;

FIG. **1**B is an example of a data processing system in which the invention may be implemented;

FIG. **1**C is an example of another data processing system in which the invention may be implemented;

FIG. **2**A is a portion of a flow chart illustrating an exemplary implementation of the query tuner operation shown in FIG. **1**A;

FIG. **2**B is a portion of a flow chart illustrating an exemplary implementation of the dynamic filter operation shown in FIG. **1**A;

FIG. **2**C is a portion of a flow chart illustrating an exemplary implementation of the result organizer operation shown in FIG. **1**A;

FIG. **3** is a further illustration of the user's operating environment illustrated in FIG. **1**B;

FIG. **4** is an example of a graphical display of a search query according to a first exemplary embodiment of the present invention;

FIG. **5** is an example of a graphical display of a search query according to a second exemplary embodiment of the present invention.

FIG. **6** is an example of a graphical display of a selected search result of FIG. **5**.

FIG. **7** is a Venn diagram of the theoretical operation according to a third exemplary embodiment of the present invention;

FIG. **8** is a functional block diagram of an exemplary implementation of the third exemplary embodiment of the invention;

FIG. **9**(*a*) is a hierarchy tree according to a fourth exemplary embodiment of the invention;

FIG. **9**(*b*) is another hierarchy tree according to the fourth exemplary embodiment;

FIG. **9**(*c*) is yet another hierarchy tree according to the fourth exemplary embodiment;

FIG. **10**(*a*) is a further hierarchy tree according to the fourth exemplary embodiment;

FIG. **10**(*b*) is a further hierarchy tree according to the fourth exemplary embodiment;

FIG. **11** is an example of an implementation of the query tuner;

DETAILED DESCRIPTION OF THE INVENTION

FIG. **1**A shows an overview of the search data processing system. The search data processing system is a computer program which may reside on a carrier such as a disk, diskette or modulated carrier wave. The system, in step **5**, begins processing when a user initiates or continues a search session. In step **10** the user enters a search query. If the user is continuing a prior search session, then the history is retreived as shown in step **11** and the previous search's keywords are added to the search query. Next, in step **12**, the system determines which of the following processing options are to be performed:

1—Query Tuner Option—Reformulation of a query

2—Dynamic Filter Option—Restriction of the results from a query

3—Results Organizer Option—Organization of the results from a query

The system then begins to process each option individually. First, the system checks, in step **14**, if the query tuner option has been selected. If the option has been selected then, in step **16**, the query refinement process is initiated and the query is modified prior to the search being performed. The search is then performed as shown in step **18**.

The system, in step **20**, checks for the existence of additional processing options to be performed. If the system determines, in step **22**, that the dynamic filter option has been selected, then the dynamic filter process is performed in step **24**. The system, in step **26**, determines if the result organizer option has been selected. If this option has been selected, then in step **28**, the results organization process is performed. Next, after all options have been processed, the system displays the results in step **30**. The system concludes with the user selection of the results as shown in step **32** and, optionally, the user saves the results of the query at step **34**.

An example of a data processing system which can use the search data processing system to search the Web is shown in FIG. **1**B. In FIG. **1**B, the Web server **41**, executes the invention and provides the users **43** access to the Web. The users **43** send their queries over the Lan **45** to the Web server **41**. FIG. **3** further illustrates a typical user's interaction with the Web when performing a search. The Web Server relays a users query to a search engine to perform the search.

Although the invention is illustrated in terms of an Internet browser searching pages on the World Wide Web, it is contemplated that it may be generally applied to any information management system. This implementation of the invention is shown in FIG. **1**C, where the user **42** executes the invention and information management system **49** is the information management system to be searched. Alternatively, the information management system may be a distributed information management system including both of the information management systems **49** and **49'**. In applying the searching techniques described below, it may be desirable to substitute information management system records for the documents and web pages described below and to substitute record tokens or some other identifying field from an information management system record for the URL of the web page.

FIG. **2**C provides the details of the results organizer from step **28** in FIG. **1**A. The results organizer processes the documents that match the query and cluster them according to common themes. Clustering may be accomplished, for example, by removing all the common stop-words from the documents and then hashing phrases of different lengths (referred to hereafter as clean phrases), such as phrases consisting of single words, pairs of consecutive words and long sequences of words, to determine which phrases occur in multiple documents that were returned by the search operation.

The hashing function takes all the text fields contained in the documents, deletes all the common stop-words, and then hashes all the clean phrases into a particular position in a hash table. Typically, a hash address value for a particular item is generated by applying an algorithm (the hashing function) directly to the item. The hashing function generates different hash table addresses for different items while generating the same hash table address for identical items.

While the exemplary embodiments of the invention are described as using a hashing function to cluster the query

results, it is contemplated that other methods of clustering, may be used instead of the hashing function. One such alternate method might be to form a concordance. Clean phrases in each document may be alphabetically sorted as they are received to form a list of all of the words in the combined documents. Each item in the list may include the clean phrase, a list of the documents in which the clean phrase occurs and the offset in each document at which the clean phrase occurs. This concordance may be used to cluster clean phrases in the documents based on the occurrence of single words or on the near occurrences of groups of words in the documents. Another alternate method might be to form a vector for each document in the multidimensional space defined by all the clean phrases in the documents. Each dimension of this space can correspond to a single clean phrase in the document collection, and the corresponding position in a document's vector is set to 1 if the document contains the clean phrase and to 0 otherwise. Any number of geometric clustering algorithms can then be used to cluster the vectors into a small number of clusters so as to minimize a geometric measure of the cluster, such as the volume of the cluster or the cluster's diameter.

As illustrated in FIG. **2**C, after the documents have been hashed in step **63**, the hash tables are analyzed to identify the clusters as shown in step **65**. The results of the clustering are then displayed in step **67** and shown by example in FIG. **4**. FIG. **4** is an example of a graphical display of a search query according to a first embodiment of the present invention.

In the exemplary embodiment, the clustering algorithm is implemented in the language Perl, which includes a non-collision hashing function. An exemplary embodiment hashes each clean phrase from the document title, URL, and summary to any entry in the hash table (also known as a hash bucket) using the hashing function in Perl. The exemplary hash table entry includes counts of the number of documents that contain the hashed clean phrase. At the end of the hashing process, each entry in the table may or may not represent a cluster. The entries are analyzed to determine the best clusters by weighing both the number of documents that contain the common clean phrase and the length of the clean phrase. The best clusters are output to the user.

FIG. **4** is an illustration of a clustered display for a query. For example, the query produced over 400 matching documents. The system discovers a small number of interesting patterns by using pattern matching and clustering algorithms. The results organizer produced clusters **410**, **420**, **430** and **440** for this sample set of documents. The partition of the documents into only 4 clusters is not intended to limit to scope of the invention rather it is shown for simplicity and illustrative purposes only. For each cluster, the system displays the number of documents that are in the cluster, the common clean phrase, and a representative document from the cluster. For example, cluster **410** contains 23 documents whose common theme is the phrase "www.quickaid.com/airports". For a URL, any characters found between consecutive slashes are interpreted as a word in the text. For example, a URL http://www.quickaid.com/airports/newark/ewr**0444**/dayd.html would cause the following "words" to be hashed: www.quickaid.com, airports, newark, ewr**0444** and dayd. In addition to single "words", the following two-word phrases and long phrases would also be hashed: www.quickaid.com/airports, airports/newark, newark/ewr**0444**, ewr**0444**/dayd and www.quickaid.com/airports/newark/ewr**0444**/dayd.

The user may choose to view any of the discovered clusters; the system, then, displays the documents that appear in the selected cluster. For example, as shown in FIG. **4**, if the user were to choose cluster **410**, the system would display the 23 documents that contain "www.quickaid.com/airports".

FIG. **5** shows an example of a graphical display of a search query for a second exemplary embodiment of the invention. The clustering lenses interface consists of a display of the title, URL, content and age lenses and a Combination window. For each lens, the corresponding part of each matching document is analyzed. As a result, a small number of interesting patterns are discovered and presented to the user by using pattern matching and clustering algorithms. Users also have the option of specifying their own patterns. More specifically, each tool takes one field at a time and partitions all the documents returned by the search engine according to a pattern found in that field. The documents may be partitioned into 1 to 5 clusters or more. Since the pattern analysis is performed on each field separately, it corresponds to viewing the documents through a lens that only displays the field of interest and hides the other fields.

FIG. **5** shows an illustration of a display for a query about New Jersey restaurants. For example, this query produces 100 matching documents. Title lens **500** partitions the documents found into 3 clusters corresponding to cells **502**, **504** and **506**. Title Lens **500** considers similarities in the titles of the matching documents. Searching for similarity in both format and words does the partitioning. For example, a format similarity is documents with "No Title" or documents whose title begins with "Re:". A word similarity refers to any common subsequence of words in the title. The strongest word similarity is identical titles; a weaker word similarity is an identical phrase within titles or identical words separated by other words, e.g. "Jane K. Doe' and Jane Katherine Doe".

Title Lens **500** finds that 40 titles contain the phrase "NJWeb: Dining in New Jersey" corresponding to a cluster in cell **502**. In cell **504**, title lens **500** finds 20 titles that start with the word "Yahoo!". In cell **106**, title lens **500** finds that the remaining 40 titles do not have any interesting patterns. In this exemplary embodiment of the invention, the width of each cell in the display is proportional to the number of documents the cell represents. The partition of the documents into only 3 clusters is not intended to limit to scope of the invention rather it is shown for simplicity and illustrative purposes only.

Also shown in FIG. **5** is URL Lens **510** which partitions the 100 documents found into four clusters corresponding to cells **512**, **514**, **516** and **518**. URL Lens **510** considers similarities in the matching documents' Web addresses. For example, if there are many files with "pub/biblio" as part of the pathname, they may form a cluster. In general, any nontrivial contiguous part of the file path is mined for patterns. URL lens **510** finds 40 URLs that contain the term "www.njweb.com/dining" corresponding to cell **512**. In cell **514**, URL lens **510** finds 20 URLs that contain the term "yahoo.com". In cell **516**, URL lens **510** finds 20 URLs that contain the term "metrocast.com". In cell **518**, URL lens **510** finds 20 URLs that have no patterns. Furthermore, the 40 URLs having "www.njweb.com/dining" as a substring are exactly those with titles "NJWeb: Dining in New Jersey". Such a fact is indicated by the edges **550** joining cells **502** and **512**. Edges **552** indicate that the documents clustered in Cell **504** are exactly those documents clustered in cell **514**.

Further, FIG. **5** shows Content Lens **520** with the 100 documents found partitioned into 4 clusters corresponding to cells **522**, **524**, **526** and **128**. Content lens **520** considers similarity in the short excerpts of the matching documents.