Hierarchical Structural Approach to Improving the Browsability of Web Search Engine Results

Hang Cui and Osmar R. Zaïane
University of Alberta
Edmonton, AB, T6H 2E1 CANADA
{cui,zaiane}@cs.ualberta.ca

Abstract

Web users have been mainly relying on Web search engines to find information of interest on the Web. However, two key issues remain with traditional Web search engines: the browsability of searching results and the capacity of Web coverage. The long ranked list presentation of search results, which is widely adopted by the industry adds a layer of confusion to users, especially when the number of matches returned from search engines can easily exceed ten thousand level. In this research, we designed an agent system based on hierarchically structural approach for organizing Web search results coupled with a metasearch approach for Web searching. The metasearch approach would help us extract the best of the Web from a larger Web coverage; and our ontological approach is aimed at providing a mechanism to categorize search results in a semantic hierarchical organization, and allow users to find target information in a progressive and interactive manner.

Introduction

With the wide use of the Internet, and the exponential growth of the size of World Wide Web, information retrieval and resource discovery from the Web is becoming more challenging. The revolution that the Web has brought to information access is not so much due to the availability of information, but rather the increased efficiency of accessing information. The emergency of Web search engines has been helping Web users for information discovery from the Web. It is estimated that eighty five percent of Web users rely on search services to locate Web pages and sixty percent of Web users use Web directories. However, the current crawling - indexing - ranking – querying By Keywords model that most of today's search engines adopt does not guarantee a perfect answer to the user's query. This is mainly due to the huge size of the Web, which is still growing exponentially. First, there is not a single Web search engine that is able to index the whole Web. It is estimated that the largest capacity for a single search engine today covers about thirty percent of the Web [LAW98]. Therefore, if a single Web search engine is used for Web searching, a large chunk of information resource is never explored. Secondly, the long ranked list presentation of search results has become the de facto standard way of organizing and displaying the search results to the user. The intention of this approach is to return the full ranked list of matches to the user, and leave the task of navigating and further searching to the user. However, the ranked list of the document set returned by search engine could easily exceed the thousands. The user would have to sift through this large document collection to find information relevant to his/her query. The assumption made by search engines is that the ranking strategy suits the user needs. However, chances are that some relevant information is buried in the long ranked list of the document set, and never reaches to the user. Therefore, the browsability of search engine results has to be improved in order to meet the increasing quality demand of Web users, and the rapid growth of the Web itself. This is also true for existing metasearch engines that compile results for different search engines.

Since the size of the Web is beyond the coverage of a single Web search engine, Metasearch engines such as MetaCrawler [SEL96], Mamma, etc. have been designed to alleviate this issue. Metasearch engines significantly increase the coverage of the Web without indexing any Web page [LAW99]. In this project, we also adopt the same concept for larger Web coverage. Whether it is a single or metasearch engine, the volume of matches that a search engine is capable of answering to a query is overwhelming to the user. Obviously, the browsability of ranked list is much reduced for large datasets, and it relies heavily on the ranking algorithm to highlight the most relevant Web pages. However, ranking algorithms would fail to satisfy users with the same searching query but different intentions. One way to approach this issue is by grouping search engine results into different categories. Each category can be considered as a sub-topic under the query term. It is up to the user which category he/she wants to browse.

Related Work

The necessity of improving the browsability of search engine results has increasingly drawn attention of researchers. Some promising and interesting approaches are presented in the following:

Web Document Clustering: Zamir et al. proposed to group Web search engine results into clusters [ZAR97]. In this approach, a suffix tree structure is used to identify shared phrases among snippets of text associated with search engine results. It provides a mechanism to take advantage of the sequential relationship of words in documents to create meaningful clusters to assist online browsing. The documents with the shared phrase are grouped into one cluster, and the shared phrase is used as the topic of the cluster. Clusters containing similar sets of documents are merged to reduce small clusters. However this grouping does not present a hierarchical classification. Another application worth mentioning is the Scatter/Gather system, which introduced clustering as a document browsing method before even the inception of the Web [CUT93]. It used a linear time clustering algorithm (Buckshot) to cluster a corpus of documents and presented these clusters to the user. The user selected one or more clusters for further investigation, and the documents in this sub-collection were then clustered again in an iterative and hierarchical manner. However, the relationship in the hierarchy layers did not follow any semantic logic and is thus difficult to browse. Principal Direction Divisive partitioning algorithm [BOL98] was proposed to automatically generate hierarchical topical taxonomy of a document set. The algorithm constructs a binary tree hierarchy of clusters starting with a single cluster encompassing the entire document collection, and recursively splits clusters based on a linear discriminant function derived from the principal direction until a desired number of clusters is reached. The resulting tree is not only binary, but like [CUT93] it doesn't hold a semantic meaning in the parent-child relationship.

Web Document Classification: Another type of approach for improving browsability of web documents retrieved by search engine is classifying web documents into pre-defined categories. However, classifiers need to be trained beforehand in order to use them for categorizing web documents. Our document classification approach, as we shall see later, does not require training. Recently, a machine-learning algorithm has been investigated to generate a hierarchy of classes for web document classification. Naïve Bayesian classifiers were trained to map Web documents into categories of Yahoo's hierarchy [DAP97] [DUN98]. The final result of learning is a set of specialized classifiers. This research showed very promising scheme to map Web documents into categories of a hierarchical organization. However, the classification accuracy of this approach still needs significant improvements.

Issues and Approach Concept

One of the major issues that today's Web search engines are facing lies in the organization and presentation style of their returning search result. Almost all major Web search engines adopt the ranked list of matching document format. Clearly, as the Web grows larger, the dominating ranked list scheme is creating a bottleneck for Web search engines to help users locate target information. Users are tied to the specific ranking algorithm of the search engine used. This is especially problematic to the users due to the following facts. First, in many cases, there is a certain degree of gap between the user's query term(s) and the true intention behind the query term used for searching. This does not just apply to naïve search engine users. Secondly, Web search engines satisfy user's query by returning a ranked list of all matches from their indexing database. Even though the recall could be very high, the precision against user's querying intention is very low. In many situations unfortunately, search engine's ranking algorithm cannot reflect user's querying intention. Therefore, the target pages are scattered and mingled in the pile of results. Thirdly, according to a survey of user's browsing behaviour, more than eighty-five percent of search engine users only sift through the top 30 results returned by the search engine. A large portion of the targeted information never finds a way to the users. As the Web grows larger, we believe it will increasingly compromise the role that Web search engines play in the Internet. Therefore, changing the way Web search engines present their search results to improve browsability is becoming necessary.

Instead of the flat arrangement of a ranked list of search results, we believe that the most desired paradigm for organizing Web search engine results, making it comprehensive to users, is by categorizing different documents according to their topics, where topics are organized in a hierarchy of increasing specificity. Such an approach not only applies the concept of abstraction to manage a large amount of information, but also builds the foundation for the mechanism of interactive user browsing.

System Model

We propose an independent software agent, which takes the user query term(s) from Web browser, and sends it to the server to trigger a server side script; the server side script will distribute user's query to multiple Web search engines for a metasearch. The search results are collected and pre-processed to create index files. The index files are used to construct a hierarchical organization of search results based on an ontological approach. The final results are posted back to user's Web browser. The user can interact with the tree-like organization of search results through on-line analytical processing-like (OLAP) operations, which support progressive navigation.

Ideally, such an agent should possess the following functionalities:

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

There are two major functional components of the proposed software agent as shown in Figure 1. The first component takes user's query and performs Web searching. The second component is responsible for reorganizing search results, and providing a mechanism for browsing through interactive user navigation.

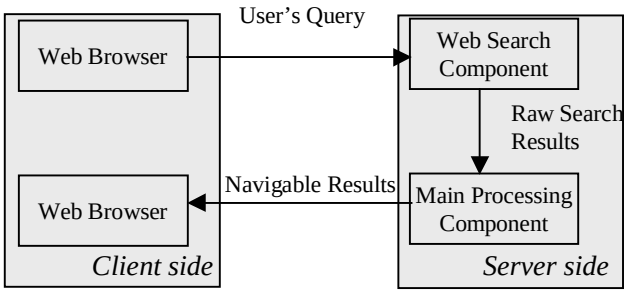

Figure 1: System Component Modeling

Independent, metasearch approach:

In this research, we create a metasearch engine on top of multiple Web search engines. It sends the user's query to several user-specified Web search engines, such as AltaVista, Google, Yahoo, etc., and takes the combined search results as input for further processing. Since we mainly focus on the reorganization and browsability of Web search results, metasearch seems a natural fit for the project.

Ontological approach for search result organization

How to create a hierarchical organization of search result is an interesting issue for this research. As mentioned in the last section, Web document clustering and classification have been investigated to achieve this goal. An ontological approach is proposed in our research for this purpose.

Ontology is a set of semantic information represented in a form that can be manipulated by software components [MEE00]. It is an agreement about shared conceptualization. A good ontology is hard to build through automatic machine learning process since the linguistic issues that involve themselves are conceptual and abstract, and therefore hard to capture and model. Here, we propose an ontological approach that uses WordNet - an online lexical database, to help us construct such a hierarchical organization of Web search results. More specifically, we are taking advantage of the ontology of nouns provided by WordNet to categorize Web documents.

WordNet is an online lexical reference system built by linguists from Princeton University [BEC90]. In WordNet, lexicons are divided into five categories: nouns, verbs, adjectives, adverbs and function words. Presently, it contains approximately 96,000 different words organized into about 70,000 word meanings. WordNet organizes lexical information in terms of word meanings, rather than word forms. The smallest unit is the word/sense pair. Word/sense pairs are linked through WordNet's lexical relation, synonymy, which is expressed by grouping word/sense pairs into synonym sets or synsets. Each synset represents a concept. Concepts are linked through conceptual - semantic relations. Under this designing philosophy, nouns are organized as topical hierarchies with 25 top-level categories, and twelve levels in depth. There is a very strong semantic relationship (synonymy/hyponymy) between a parent and a child in the topical hierarchy of nouns in WordNet. For a given noun, WordNet provides an API (application programming interface) to find its semantically related concepts [BEC90]. Thus, we can utilize such a mechanism to build a specific ontology, and map search engine results into the created-on-the-fly hierarchy.

User's Navigation Scheme

The hierarchical organization of search results is a tree-like structure with Web documents categorized in the leaf nodes. Each internal node of the tree represents a higher-level concept that semantically includes all of its child concepts. Thus, by following a path from the root to a leaf in this hierarchy, users can start with a generalized concept, and gradually get to more specialized information. The philosophy behind the hierarchical organization of search results is information abstraction [HAN92], which assumes that most of the users prefer to browse general description of information instead of the overwhelming details of information in the first glance when searching for target information. Therefore, the most generalized concepts located in the very top level of the hierarchy will always be presented to the user first, and the system provides an mechanism to allow user to set his/her own path towards the target information for details in a progressive manner by either drilling-down or rolling-up along the concept hierarchy, or drilling-through from the tree node to the documents subsumed by the concept in the node. Such a navigating scheme involves user's interaction, and thus gives the flexibility to the user to make his/her decisions along the way of navigation.

System Architecture Design and Implementation

We adopted the client–server architectural style for system design. The major functionality of the system is operated on the server. The user can access the system through a thin client – Web browser, which allows to sending a query and navigating the search result.

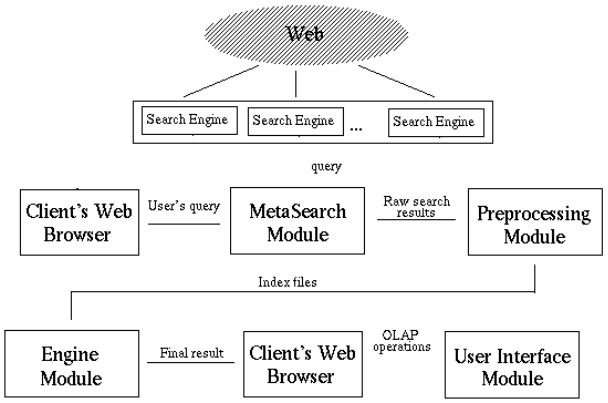

Figure 2: System Modularity

On the server side, the system is partitioned into four functional modules as shown in Figure 2: MetaSearch Module, Preprocessing Module, Engine Module, and User Interface Module.

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

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

1. Eliminate all the duplicates of matches retrieved by more than one Web search engines.
2. Remove all the punctuation marks, numbers and HTML tags.
3. Remove all the stop-words from each Title and Snippet.
4. Stem the stop-word removed text.
5. Create an index file. Each retrieved URL is indexed using a vector of keywords.
6. Create an inverted index file. Given an extracted keyword, the inverted index file stores the frequency of its appearance, and all pointers as references of the location(s) that it appears among the whole retrieved items.

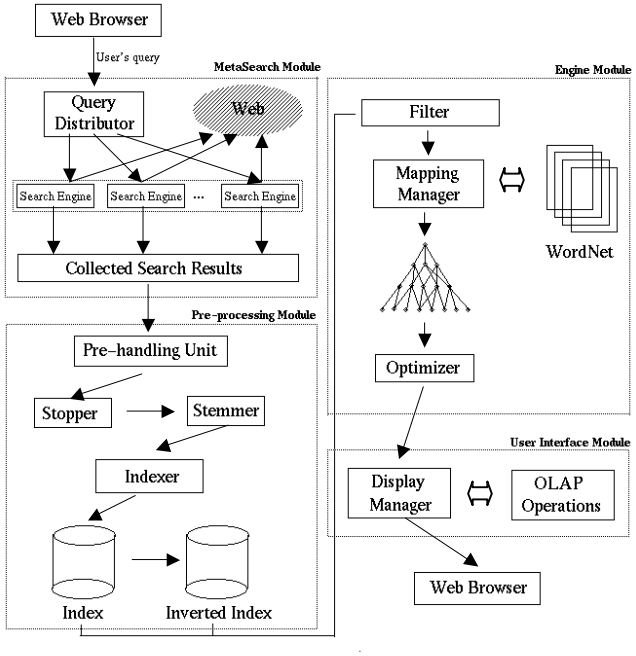

Figure 3: System Architecture Design

The Engine Module is the core of the system. The main functionality of the Engine Module is constructing the hierarchical organization of search results, and optimizing the tree-like structure. Once the indices are generated, the Engine Module is triggered. All records go through the Filter. Only those records with nouns as keywords, and with frequencies of appearance above the threshold of 3% are sent to the Mapping Manager for hierarchy construction. For each of the qualified record, the Mapping Manager takes the keyword (noun), refers to the ontological organization of nouns in WordNet, finds its precedents (hypernyms) at different levels of the ontology, then, the keyword itself and its hypernyms are used to construct a unique path of the hierarchy with the most generalized hypernym mapped directly under the root (which contains the query term), and the keyword as the leaf of the path. After the completion of hierarchy construction, all keywords from the inverted file are mapped into leaves of the hierarchy with corresponding retrieved Web pages attached. The Optimizer prunes the tree to eliminate all the internal tree nodes with only one single child. This will reduce the depth of the tree, and more efficient for navigating.

The User Interface Module is responsible for describing the tree structure according to its hierarchical organization, and translating it into HTML as a visual folder tree. Each

internal node is a folder of higher-level topical concept, which contains all the sub-topical concepts. Retrieved information is stored in the leaf nodes. Another function of User Interface Module is allowing interactive drilling-down and rolling-up operations to navigate the folder tree.

Figure 4: The result page in response to query: "Edmonton Oilers"

Case Study

Querying examples were given to illustrate the system behaviour and some characters of our system design concept [CUI01]. Here, we chose "Edmonton Oilers" as the query term for Web search, and the result page is shown in Figure 4. The bottom-right frame gives a full list of retrieved Web pages. The bottom-left frame shows the hierarchical folder tree that displays the top-level categories directly under the root – "Edmonton Oilers". The hierarchical folder tree sorts all the retrieved Web pages into 41 different topics. Each topic is a cluster, which is located at the bottom of the folder tree, and has a unique path in the folder tree. The navigation of the folder tree requires user's interaction. If the user is interested in hockey activities related to Oilers for example, path navigation should be followed along Edmonton Oilers ➔ human activities ➔ activity ➔ diversion, recreation ➔ sport, athletics ➔ hockey as shown in Figure 4 to reach the particular cluster. If the user is interested in a specific Web page in the cluster, clicking on its title with hyperlink can access the Web page. Following paths in the presented ontology can lead to the discovery of interesting information. For instance, the Oilers' historical game statistics can be found in path Edmonton Oilers ➔ knowledge ➔ information ➔ statistics.

Conclusions and Future Work

We have designed an implemented a system to categorize web documents from a search engine result list. The system classifies the documents based by mapping them on an existing concept hierarchy. The resulting tree is later trimmed to keep the essential paths and presented to the user to navigate through the large result list and easily find the relevant documents. We have shown a proof of concept using WordNet as an initial semantic network. Because of the nature of WordNet, rich with linguistic terms, the resulting tree is often full with elaborate terminology at the different nodes. The system however, can be done with any given concept hierarchy. We are currently investigating the use of association rule mining to discover frequent terms and phrases in the document collection but not present in the given concept hierarchy of thesaurus. The combination of both approaches would give even better results.

References

[BEC90] R. Beckwith, C. Fellbaum, D. Gross, K. Miller, G.A. Miller, R. Tengi, Five Papers on WordNet, Special Issue of the *International Journal of Lexicography* 3(4), 1990, pp. 235-312.

[CUI01] H. Cui. The architecture design and implementation of J-Walker, a metasearch engine based on ontological approach. M.S. Thesis, Dept. of Computing Science, Univ. of Alberta, May 2001.

[CUT92] D. Cutting, D.R. Karger, J. Redersen and J. Turey. Scatter/Gather: A cluster based approach to browsing large document collections. In Proceedings of the 15th International ACM SIGIR Conference on Research and Development in Information Retrieval, 318-329, 1992.

[CUT93] D. Cutting, D.R. Karger, J. Redersen and J. Tukey. Constant interaction time Scatter/Gather browsing of large document collections. In Proceedings of the 16th International ACM SIGIR Conference on Research and Development in Information Retrieval, 126-135, 1993.

[DAP97] D. Koller, M. Sahami. Hierarchically classifying documents using very few words. In proc. of the 14th Intl. Conf. on Machine Learning, pp 170-178, Nashville, TN, July 1997.

[DUN98] Dunja Mladenic. Turning Yahoo into an automatic Web-page classifier. In Proceedings of the 13th European Conference on Artificial Intelligence ECAI'98, pp. 473-474.

[HAN92] J. Han, Y Cai and N. Cercone. Knowledge Discovery in Databases: An Attribute-oriented Approach. In Proceedings of the 18th VLDB Conference, Vancouver, BC, Canada, 1992.

[LAW98] S. Lawrence and C. Giles. Searching the World Wide Web. Science, Volume 280, No.5360, pp. 98-100, 1998.

[LAW99] S. Lawrence and C. Giles. Search the Web: General and Scientific Information Access. In IEEE Communications, 37(1), pp. 116-122, 1999.

[MEE00] R. Meersman. Can Ontology learn from database semantics. http://www.starlab/vub.ac.be/staff/robert

[SEL96] E. Selberg and O. Etzioni, The MetaCrawler architecture for resource aggregation on the Web. IEEE Expert, 12(1), 1997.

[ZAM97] O. Zamir, O. Etzioni, O. Madani, and R. Karp. Fast and intuitive clustering of Web documents. In Proc. of the 3rd Intl. Conf. on Knowledge Discovery and Data Mining, 1997.