SEMANTIC BASED MULTIPLE WEB SEARCH ENGINE

MS.S.LATHA SHANMUGAVADIVU,

ASSISTANT PROFESSOR, ECE DEPARTMENT,
TAMILNADU COLLEGE OF ENGINEERING,
COIMBATORE.

DR.M.RAJARAM,

PROFESSOR AND HEAD, EEE DEPARTMENT,
GOVERNMENT COLLEGE OF ENGINEERING,
TIRUNELVELI.

ABSTRACT

With the tremendous growth of information available to end users through the Web, search engines come to play ever a more critical role. Nevertheless, because of their general-purpose approach, it is always less uncommon that obtained result sets provide a burden of useless pages. The next-generation Web architecture, represented by the **Semantic Web**, provides the layered architecture possibly allowing overcoming this limitation. Several search engines have been proposed, which allow increasing information retrieval accuracy by exploiting a key content of Semantic Web resources, that is, relations. To make the Semantic Web work, well-structured data and rules are necessary for agents to roam the Web [2]. XML and RDF are two important technologies: we can create our own structures by XML without indicating what they mean; RDF uses sets of triples which express basic concepts [2]. DAML is the extension of XML and RDF The aim of this project is to develop a search engine based on ontology matching within the Semantic Web. It uses the data in Semantic Web form such as DAML or RDF. When the user input a query, the program accepts the query and transfers it to a machine learning agent. Then the agent measures the similarity between different ontology's, and feedback the matched item to the user.

Keywords: semantic web, clustering, ontology, web crawling, multiple web search engine

INTRODUCTION

The Semantic Web is known for being a web of Semantic Web documents; however, little is known about the structure or growth of such a web. Search engines such as Google have transformed the way people access and use the web and have become a critical technology for finding and delivering information. Most existing search engines, however, provide poor support to accessing the web of result's and make no attempt to take advantage of the structural and semantic information encoded in SWDs.The Semantic Web will offer the way for solving this problem at the architecture level. In fact, in the Semantic Web, each page possesses semantic metadata that record additional details concerning the Web page itself. My research work is designed to serve the research activities in Semantic Web community, especially the following:

(i) Multiple Search Engine for single user query
(ii) Apply Clustering Method
(iii) Unwanted pages in the result set would force him or her to perform a post processing on retrieved information to discard unneeded ones. Today, search engines constitute the most helpful tools for organizing information and extracting knowledge from the Web. However, it is not uncommon that even the most renowned search engines return result sets including many pages that are definitely useless for the user this is mainly due to the fact that the very basic relevance criterions underlying their information retrieval strategies rely on the presence of query keywords within the returned pages. When a user enters a query composed by the following keywords "hotel," "Rome," and "historical center" (or "hotel," "Roma," and "centro storico") in the Italian version of the well-known Google search engine. He or she would not be astonished probably by finding that the result set actually includes several hotels located in the historical center of Rome, as expected small town at some distance from the Rome city center is also included. However, two hotels located in the historical center of other main Italian cities are also displayed. Finally, three hotels named Roma are included among the 10 most relevant results even if they have nothing to do with the selected city. Only 4 out the 10 results presented to the user satisfy user needs. (Even if they seem to satisfy the user query, based on the strategy adopted to process it).

Currently, the Semantic Web, (i.e. online documents written in RDF or OWL), is essentially a web universe parallel to the web of HTML documents. Semantic Web doc- uments (SWDs) are characterized by semantic annotation and meaningful references to other SWDs [5]. Since conventional search engines do not take advantage of these features, a search engine customized for SWDs, especially for ontologies, is needed by human users as well as by software agents and services. At this stage, human users are expected to be semantic web researchers and developers who are interested in accessing, exploring and querying RDF and OWL documents found on the web.

2.1 EXISTING SYSTEM

Nevertheless, because of their general-purpose approach, it is always less uncommon that obtained result

sets provide a burden of useless pages. It is not uncommon that even the most renowned search engines return result sets including many pages that are definitely useless for the user this is mainly due to the fact that the very basic relevance criterions underlying their information retrieval strategies rely on the presence of query keywords within the returned pages.

2.1.1 Disadvantage

(i) Text based searching example (Google, yahoo, msn, Wikipedia).
(ii) Without semantic relationship to give exact result.
(iii) Query only focus single search engine.
(iv) Most existing search engines however, provide poor support to accessing the web results.
(v) No analysis of stopping keywords from the user query.
(vi) It will not give relevant or exact result.
(vii) Number of iterations is high.

2.1.2 Prototype of a Relation-Based Search Engine

To evaluate the feasibility of the proposed approach, we first constructed a controlled Semantic Web environment. To do this, we selected the well-known travel.owl ontology written in the OWL language, and we modified it by adding new relations in order to make it more suitable for demonstrating system functionality. We then created a knowledge base by either downloading or automatically generating a set of web pages in the field of tourism, and we embedded into them RDF semantic annotations based on the ontology above. Finally, we designed the remaining modules of the architecture, including a Webpage database, a crawler application, a knowledge database, an OWL parser a query interface, and the true search engine module embedding the proposed ranking logic.

In Fig1.the crawler application collects annotated Web pages from the Semantic Web (in this case, represented by the controlled environment and its Web page collection) including RDF metadata and originating OWL ontology. RDF metadata are interpreted by the OWL parser and stored in the knowledge database. A graphics user interface allows for the definition of a query, which is passed on to the relation-based search logic. The ordered result set generated by this latter module is finally presented to the user. The details of the system workflow will be provided in the following sections, starting with the query definition process, since it was through the analysis of its dynamics that we came to the identification of our ranking strategy.

Architecture Diagram

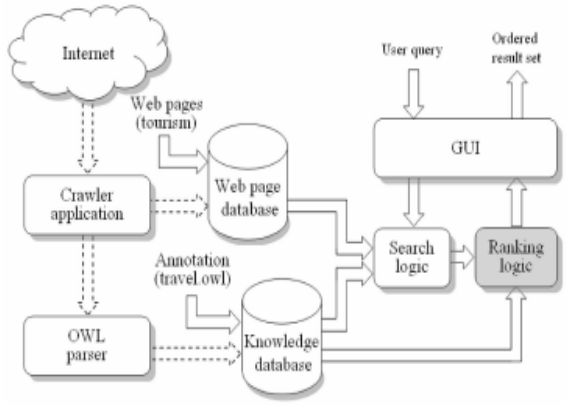


FIGURE 1: Semantic Web infrastructure (prototype).

3.1 PROPOSED SYSTEM

The Semantic Web will offer the way for solving this problem at the architecture level. In fact, in the Semantic Web, each page possesses semantic metadata that record additional details concerning the Web page itself. It will be proved that relations among concepts embedded into semantic annotations can be effectively exploited to define a ranking strategy for Semantic Web search engines. A similarity score measuring the distance between the systematic descriptions of both query and retrieved resources is defined. They first explode an initial set of relations (properties) by adding hidden relations, which can be inferred from the query. Similarity is then computed as the ratio between relation instances linking concepts specified in the user query and actual multiplicities of relation instances in the semantic knowledge base. This method is applied on each property individually and requires exploring all the Semantic Web instances. Moreover, the user is requested to specify all the relations of interest. Thus, since it is predictable that the number of relations will largely exceed the number of concepts, its Applicability in real contexts is severely compromised. A similar approach, aimed at measuring the relevance of a semantic association (that is, a path traversing several concepts linked by semantic relations)[3]. We provide an interesting definition of relevance as the reciprocal of the ambiguity of the association itself. Ontology-based lexical relations like synonyms, antonyms, and homonyms between keywords (but not concepts) have been used to “expand” query results which automatically associate related concepts, and exploit the semantic knowledge base to automatically formulate formal queries.

3.1.1 ADVANTAGE OF PROPOSED SYSTEM

- The Semantic Web will offer the way for solving this problem at the architecture level. In fact, in the Semantic Web, each page possesses semantic metadata that record additional details concerning the Web page itself.

- This method is applied on each property individually and requires exploring all the Semantic Web instances.

We will prove that relations among concepts embedded into semantic annotations can be effectively exploited to define a ranking strategy for Semantic Web search engines

3. 2. IMPLEMENTATION

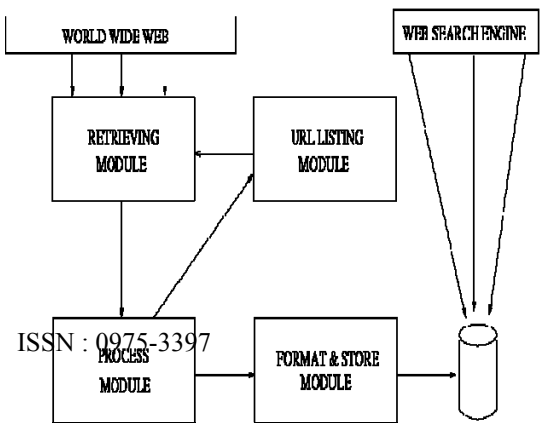
Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

3.2.1 Web Search Engine Design

The term "search engine" is often used generically to describe both crawler-based search engines and human-powered directories. These two types of search engines gather their listings in radically different ways. Crawler-based search engines, such as Google, create their listings automatically. They "crawl" or "spider" the web, then people search through what they have found. A human-powered directory, such as the Open Directory, depends on humans for its listings. When we submit a short description to the directory for your entire site or editors write one for sites they review. A search looks for matches only in the descriptions submitted.

The typical crawler-based search engine has several major elements. First is the **spider**, also called the crawler. The spider visits a Web page, reads it, and then follows links to other pages within the site. This is what it means when someone refers to a site being "spidered" or "crawled." The spider returns to the site on a regular basis, such as every month or two, to look for changes. Everything the spider finds goes into the second part of the search engine, the **index**. The index, sometimes called the catalogue, is like a giant book containing a copy of every Web page that the spider finds. If a Web page changes, then this book is updated with new information. **Search engine software** is the third part of a search engine. This is the program that shifts through the millions of pages recorded in the index to find matches to a search and rank them in order of what it believes is most relevant.

3.2.2 Web Crawler



A search engine cannot work without a proper index where possible searched pages are stored, usually in a compressed format. This index is created by specialized robots, which crawl the Web for new/modified pages (the actual crawlers, or spiders). Typical crawler architecture is depicted in the figure4 below.

FIGURE 2: TYPICALCRAWLER ARCHITECTURE

"Smart" crawling technology is used to crawl 'valuable' sites more frequently or more deeply. The measure of 'value' is, of course, itself an important research topic. Smart crawling is also used to estimate the rate of change of web pages and adjust the crawling algorithm to maximize the freshness of the pages in the repository.

3.2.3 Multiple Search engine Design

The most known general search engines are Google and Yahoo! but one of the oldest search engines is AltaVista. All existing search engines have weaknesses, even Google (link searches must be exact, it does not support full Boolean, it only indexes a part of the web pages or PDF files, etc.). This part represents a real reason for building more search engine. A scalable distributed repository is used to store the crawled collection of Web pages. Strategies for physical organization of pages on the storage devices, distribution of pages across machines, and mechanisms to integrate freshly crawled pages, are important issues in the design of this repository. The repository supports both random and stream-based access modes. Random access allows individual pages to be retrieved based on an internal page identifier. Stream-based access allows all or a significant subset of pages to be retrieved as a stream. Query-based access to the pages and the computed features (from the feature repository) is provided via the Web Base query engine[8].Unlike the traditional keyword-based queries supported by existing search engines, queries to the Web Base query engine can involve predicates on both the content and link structure of the Web pages. In selection of search engines twenty five search engines were selected to conduct our experiment. They are AlltheWeb, AltaVista, google, yahoo, clusty, you tube, file tube, citeceer etc., to name a few. At first, the search engines were selected and the user query is submitted to all search engines under consideration. The queries covered a broad range of topics. The topics are as follows: Computer science, education, Internet, literature, music, plants, sports, travel etc. The precision of content of these pages is compared to give the result.

3.2.4 RDF

RDF is a general framework for describing a Web site's metadata, or the information about the information on the site. It is a short form for resource description

framework[13]. It provides interoperability between applications that exchange machine-understandable information on the Web. RDF details information such as a site's sitemap, the dates of when updates were made, keywords that search engines look for and the Web page's Resource Description Framework is a framework for processing metadata and it describes relationships among resources with properties and values. It is built on the following rules[12]:

a. **Resource**: Everything described by RDF expressions is called a resource. Every resource has a URI and it may be an entire web page or a part of a web page

b. **Property**: "A property is a specific aspect, characteristic, attribute, or relation used to describe a resource" – W3C, Resource Description Framework (RDF) Model and Syntax Specification .Note that a property is also a resource since it can have its own properties.

c. **Statements**: A statement combines a resource, a property and a value. These three individual parts are known as the "subject", "predicate" and "object".

3.2.5 Design of Ontology Search

As mentioned in the last section, finding ontologies to satisfy user requirements is a very important issue, in both KB reuse and Semantic Web areas. There is no existing tool to solve this problem. Google does have the power, but does not seem to be specific enough to give good results. After some experiments, we noticed that the problem arises because Google does not offer a good visualization function for the ontology files (in different formalisms, such as RDFs, etc.)[14], as the user cannot view the ontology in an intuitive graphic format; they have to look thro ugh the ontologies as structured text files. This process takes a lot of time and cannot guarantee a good result, as the plain text of the ontology cannot show the internal structure of the ontology clearly.

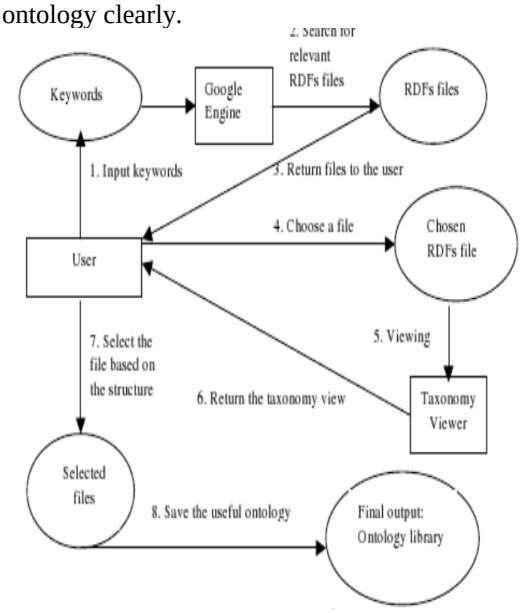


FIG 3: Ontology searching steps

3.2.4 Clustering the web result's

Clustering is the act of grouping similar object into sets. In the web search context: organizing web pages (search results) into groups, so that different groups correspond to different user needs. Existing search engines such as Google and Yahoo return ranked lists of Web pages in response to a user's query request. Web users have to shift through the list to locate pages of interest [9]. This is a time-consuming task when multiple sub-topics of the given query are mixed together. A possible solution to this problem is to cluster search results into different groups and to enable users to identify their required group at a glance.

4. EXPERIMENTAL RESULTS:

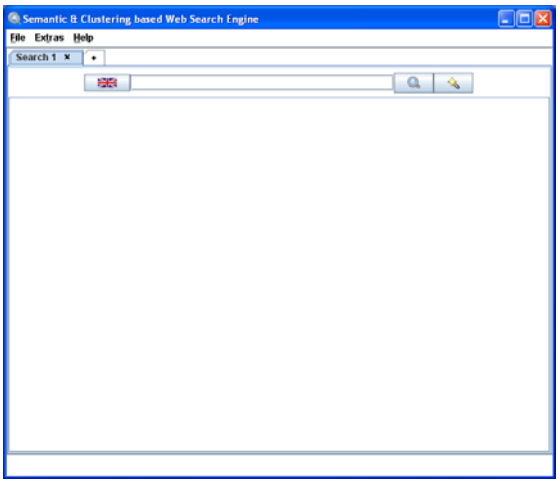


FIGURE 4 : MAIN GUI

The figure 4 shows the main Graphical User Interface (GUI). Java swing is used to design the GUI.It has three menus; the first menu is file menu. It has two submenus one is export and the other is exit menu. The export menu has two submenus, text and XML. The two submenus are used to save the user query result as text format and XML format for future reference.

The second menu is extra menu. It has two submenus quick preference and preference menu. (i) Quick preference is used to enable proxy connection and (ii) Preference submenu shows the sub window which has four windows .The general sub window has three languages from which we can select the design language for example English, French and German.

The design submenu is used to show the window as the day mode and night mode.

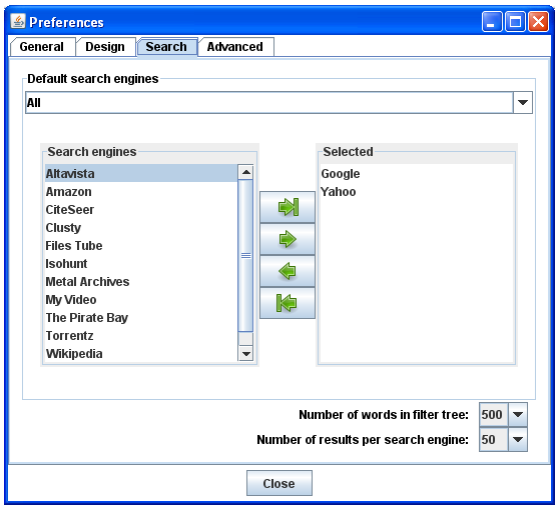


FIGURE 5 : SEARCH MENU

The figure 5 shows the search menu from which we can select the search engine based on user query and also give the filter tree value. The filter tree value is set up to 500 and another one is number of results for search engine to give the user query result. The advanced menu has three submenus, the properties, window position and proxy menu.

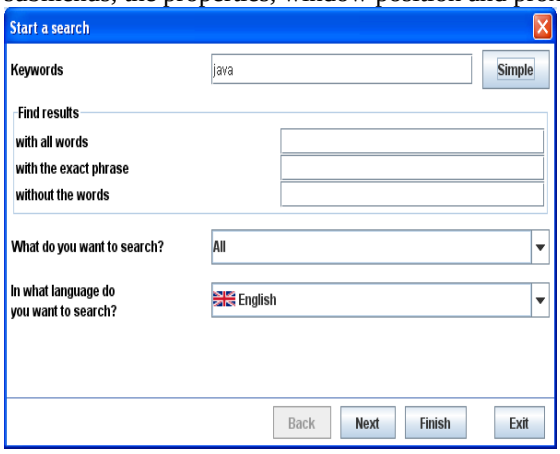


FIGURE6 : ADVANCED QUERY PROCESSING TECHNICQE

This is advanced window search processing.It can find the result with all words, exact phrase and without word's.

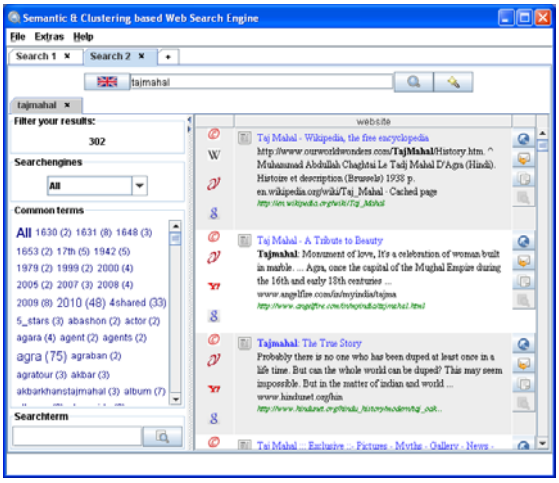


FIGURE 7: QUERY FOR TAJMAHAL

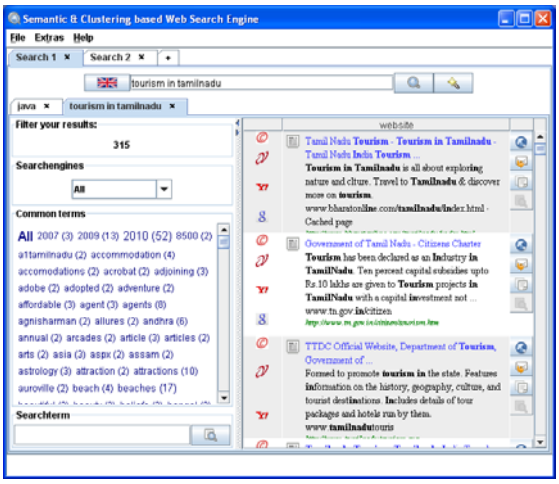


FIGURE 8: RESULT WINDOW FOR THE QUERY TOURISM

The lexical parser and syntax parser supports for forming the filter value of the user query. The common terms are clustered based on the user query given. The clustering concept used here is k means clustering algorithm. The user query takes the relation of some common terms and gives the result as shown in figure 7 and 8. For a single user query what is java? the result is got from multiple search engines; such as Google, yahoo, AltaVista, clusty, excite, all the web, file tube, you tube, amazon, cite seer, wikipedia, isohunt etc., here the answer is got from 25 search engines, searches take place simultaneously from different search engines, the main advantage is that searches do not have to wait for each search engine, the results are computed simultaneously from multiple search engines and they are displayed on the screen. The result is therefore much faster.

TABLE 1: PERFORMANCE COMPARISON

Web search engine URL	User query	Normal query result	Web cache optimization result	Semantic based web result
www.google.com	what is java swing	50	80	95
www.yahoo.com	what is java swing	30	60	79
www.altavista.com	what is java swing	60	75.5	90.1
www.youtube.com	what is java swing	10	60	89.5
www.clusty.com	what is java swing	60	80	98.9
www.filetube.com	what is java swing	59	75	88
www.torrentz.com	what is java swing	70	50	80
www.citeseer.com	what is java swing	70	78	95

The comparison table shows search engine performance for the user query what is java swing? That user query gets the result from different search engines. I can measure the user query based on normal query result, without any semantic analysis and web cache optimization analysis. Then the same query get different result from different search engine using web cache optimization. Then the same user query get the different search engine result based on semantic web result. Then I can form the bar chart using three column values. In a similar way a line chart is formed for the same query with three column values to analyze the performance. In a similar way for any user query the performance table, bar chart and line chart is formed as shown in figures 9 and 10.

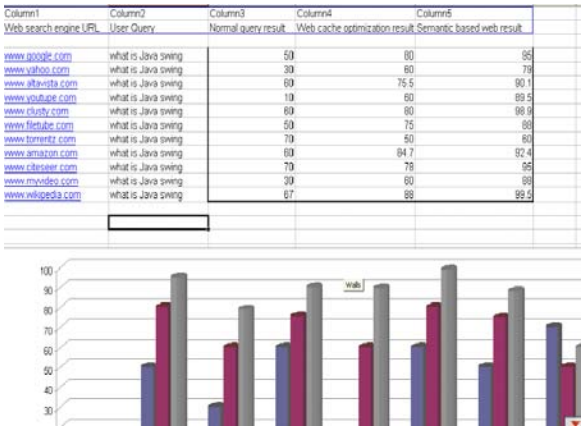


FIGURE 9: BAR CHART FOR COMPARISON TABLE

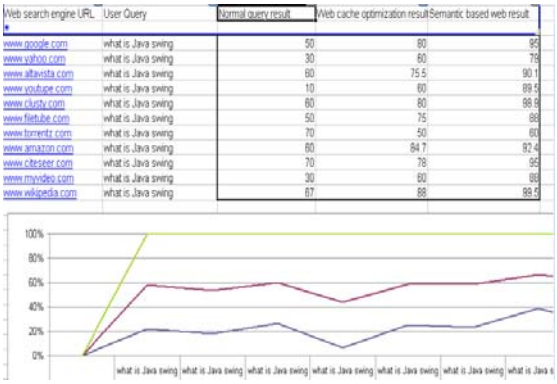


FIGURE 10: LINE CHART FOR COMPARISON TABLE

5. CONCLUSION AND FUTUREWORK

The next-generation Web architecture represented by the Semantic Web will provide adequate instruments for improving search strategies and enhance the probability of seeing the user query satisfied without requiring tiresome manual refinement. Nevertheless, they mainly use page relevance criteria based on information that has to be derived from the whole knowledge base, making their application often unfeasible in huge semantic environments. By neglecting the contribution of the remaining annotated resources, a reduction in the cost of the query answering phase could be expected. Despite the promising results in terms of both time complexity and accuracy, further efforts will be requested to foster scalability into future Semantic Web repositories based on multiple ontology, characterized by billions of pages, and possibly altered through next generation “semantic” spam techniques.

It has been designed and partially implemented to capture more metadata on classes and properties and to support millions of documents. We have also built an ontology dictionary based on the ontologies discovered by our research, which we continue to refine. We have described a prototype crawler-based indexing and retrieval system for Semantic Web documents.

6. REFERENCES

[1]B. Aleman-Meza, C. Halaschek, I. Arpinar, and A. Sheth, “A Context-Aware Semantic Association Ranking,” Proc. First Int’l Workshop Semantic Web and Databases (SWDB ’03), pp. 33-50, 2003.

[2] K. Anyanwu, A. Maduko, and A. Sheth, “SemRank: Ranking Complex Relation Search Results on the Semantic Web,” Proc. 14th Int’l Conf. World Wide Web (WWW ’05), pp. 117-127, 2005.

[3] R. Baeza-Yates, L. Caldero´n-Benavides, and C. Gonza´lez-Caro, “The Intention behind Web Queries,” Proc. 13th Int’l Conf. String Processing and Information Retrieval (SPIRE ’06), pp. 98-109, 2006.

[4] T. Berners-Lee and M. Fischetti, Weaving the Web. Harper Audio, 1999.

[5] T. Berners-Lee, J. Hendler, and O. Lassila, “The Semantic Web,” scientific Am., 2001.

[6] S. Brin and L. Page, “The Anatomy of a Large-Scale Hypertextual Web Search Engine,” Proc. Seventh Int’l Conf. World Wide Web (WWW ’98), pp. 107-117, 1998.

[7] S. Cohen, J. Mamou, Y. Kanza, and Y. Sagiv, "XSEarch: A Semantic Search Engine for XML," Proc. 29th Int'l Conf. Very Large Data Bases, pp. 45-56, 2003.
[8] L. Ding, T. Finin, A. Joshi, R. Pan, R.S. Cost, Y. Peng, P. Reddivari, V. Doshi, and J. Sachs, "Swoogle: A Search and Metadata Engine for the Semantic Web," Proc. 13th ACM Int'l Conf. Information and Knowledge Management (CIKM '04), pp. 652-659, 2004.
[9] L. Ding, T. Finin, A. Joshi, Y. Peng, R. Pan, and P. Reddivari, "Search on the Semantic Web," Computer, vol. 38, no. 10, pp. 62-69, Oct. 2005.
[10] L. Ding, P. Kolari, Z. Ding, and S. Avancha, "Using Ontologies in the Semantic Web: A Survey," Ontologies, pp. 79-113, Springer, 2007.
[11] R. Guha, R. McCool, and E. Miller, "Semantic Search," Proc. 12th Int'l Conf. World Wide Web (WWW '03), pp. 700-709, 2003.
[12] Z. Gyongyi and H. Garcia-Molina, "Spam: It's Not Just for Inboxes Anymore," Computer, vol. 38, no. 10, pp. 28-34, Oct. 2005.
[13] C. Junghoo, H. Garcia-Molina, and L. Page, "Efficient Crawling through URL Ordering," Computer Networks and ISDN Systems, vol. 30, no. 1, pp. 161-172, 1998.
[14] S. Kapoor and H. Ramesh, "Algorithms for Enumerating All Spanning Trees of Undirected and Weighted Graphs," SIAM J.Computing, vol. 24, pp. 247-265, 1995.
[15] Y. Lei, V. Uren, and E. Motta, "SemSearch: A Search Engine for the Semantic Web," Proc. 15th Int'l Conf. Managing Knowledge in a World of Networks (EKAW '06), pp. 238-245, 2006.