M

Metasearch Engines

Weiyi Meng
Department of Computer Science, State University of New York at Binghamton, Binghamton, NY 13902, USA

Synonyms

Federated search engine

Definition

Metasearch is to utilize multiple other search systems (called *component search systems*) to perform simultaneous search. A metasearch engine is a search system that enables metasearch. To perform a basic metasearch, a user query is sent to multiple existing search engines by the metasearch engine; when the search results returned from the search engines are received by the metasearch engine, they are merged into a single ranked list and the merged list is presented to the user. Key issues include how to pass user queries to other search engines, how to identify correct search results from the result pages returned from search engines, and how to merge the results from different search sources. More sophisticated metasearch engines also perform *search engine selection* (also referred to as *database selection*), i.e., identify the search engines that are most appropriate for a query and send the query to only these search engines. To identify appropriate search engines to use for a query requires to estimate the usefulness of each search engine with respect to the query based on some usefulness measure.

Historical Background

The earliest Web-based metasearch engine is probably the MetaCrawler system [1] that became operational since June 1995. (The MetaCrawler's website (www.metacrawler.com) says the system was first developed in 1994.) Motivations for metasearch include (i) increased search coverage because a metasearch engine effectively combines the coverage of all component search engines, (ii) improved convenience for users because a metasearch engine allows users to get information from multiple sources with one query submission and the metasearch engine hides the differences in query formats of different search engines from the users, and (iii) better retrieval effectiveness because the result merging component can naturally incorporate the voting mechanism, i.e., results that are highly ranked by multiple search engines are more likely to be relevant than those that are returned by only one of them. Over the last twelve years, many metasearch engines have been developed and deployed on the Web. Most of them are built on top of a small number of popular general-purpose search engines but there are also metasearch engines that are connected to more specialized search engines (e.g., medical/health search engines) and some are connected to over one thousand search engines.

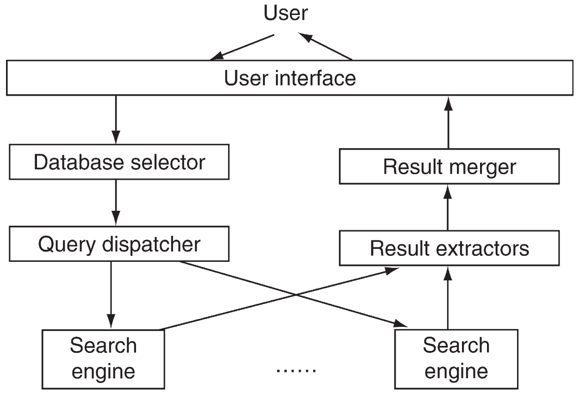
Even the earliest metasearch engines tackled the issues of search result extraction and result merging. Result merging is one of the most fundamental components in metasearch, and as a result, it has received a lot of attention in the metasearch and distributed information retrieval (DIR) communities and a wide range of solutions has been proposed to achieve effective result merging. Since different search engines may index a different set of web pages and some search engines are better than others for queries in different subject areas, it is important to identify the appropriate search engines for each user query. The importance of search engine selection was realized early in metasearch research and many approaches have been proposed to address this issue. A survey on some earlier result merging and search engine selection techniques can be found in [2].

Most metasearch engines are built on top of other search engines without explicit collaboration from these search engines. As a result, creating these metasearch engines requires a connection program and an extraction program (wrapper) for each component search engine. The former is needed to pass the query from the metasearch engine to the search engine and receive search results returned from the search engine, and the latter is used to extract the search result records from the result pages returned from the search engine. While the programs may not be difficult to produce by an experienced programmer, maintaining their validity can be a serious problem as they can become obsolete when the used search engines change their connection parameters and/or result display format. In addition, for applications that need to connect to hundreds or thousands of search engines, it can very expensive and time-consuming to produce and maintain these programs. As a result, in recent years, automatic wrapper generation techniques have received much attention. Figure 1 shows a basic architecture of a typical metasearch engine.

Scientific Fundamentals

Result Merging

Result merging is to combine the search results returned from multiple search engines into a single ranked list. Early search engines often associated a numerical matching score (similarity score) to each retrieved search result and the result merging algorithms at that time were designed to "normalize" the scores returned from different search engines into values within a common range with the goal to make them more comparable. Normalized scores will then be used to re-rank all the search results. When matching scores are not available, the ranks of the search results from component search engines can be aggregated using voting-based techniques (e.g., Borda Count [3]). Score normalization and rank aggregation may also take into consideration the estimated usefulness of each selected search engine with respect to the query, which is obtained during the search engine selection step. For example, the normalized score of a result can be weighted by the usefulness score of the search engine that returned the result. This increases the chance for the results from more useful search engines to be ranked higher.



Metasearch Engines. Figure 1. Metasearch engine component architecture.

Another result merging technique is to download all returned documents from their local servers and compute their matching scores using a common similarity function employed by the metasearch engine. The results will then be ranked based on these scores. For example, the Inquirus metasearch engine employs this approach [4]. The advantage of this approach is that it provides a uniform way to compute ranking scores so the resulted ranking makes more sense. Its main drawback is the longer response time due to the delay caused by downloading the documents and analyzing them on the fly. Most modern search engines display the title of each retrieved result together with a short summary called *snippet*. The title and snippet of a result can often provide good clues on whether or not the result is relevant to a query. As a result, result merging algorithms that rely on titles and snippets have been proposed recently (e.g., [5]). When titles and snippets are used to perform the merging, a matching score of each result with the query can be computed based on several factors such as the number of unique query terms that appear in the title/snippet and the proximity of the query terms in the title/snippet.

It is possible that the same result is retrieved from multiple search engines. Such results are more likely to be relevant to the query based on the observation that different ranking algorithms tend to retrieve the same set of relevant results but different sets of irrelevant results [6]. To help rank these results higher in the merged list, the ranking scores of these results from different search engines can be added up to produce the final score for the result. The search results are then ranked in descending order of the final scores.

Search Engine Selection

To enable search engine selection, some information that can represent the contents of the documents of each component search engine needs to be collected first. Such information for a search engine is called the *representative* of the search engine. The representatives of all search engines used by the metasearch engine are collected in advance and are stored with the metasearch engine. During search engine selection for a given query, search engines are ranked based on how well their representatives match with the query. Different search engine selection techniques have been proposed and they often use different types of representatives. A simple representative of a search engine may contain only a few selected key words or a short description. This type of representative is usually produced manually by someone familiar with the contents of the search engine but it can also be automatically generated. As this type of representatives provides only a general description of the contents of search engines, the accuracy of using such representatives for search engine selection is usually low. More elaborate representatives consist of detailed statistical information for each term in each search engine. In [7], the *document frequency* of each term in each search engine is used to compute the *cue validity variance* of each query term, which measures the skew of the distribution of the query term across all component search engines, to help rank search engines for each query. In [8], the *document frequency* and *collection frequency* of each term (the latter is the number of component search engines that contain the term) are used to represent each search engine. In [9], the *adjusted maximum normalized weight* of each term across all documents in a search engine is used to represent a search engine. For a given term t and a search engine S, its adjusted maximum normalized weight is computed as follows: compute the normalized weight of t in every document (i.e., the term frequency weight of t divided by the length of the document) in S, find the maximum value among these weights, and multiply this maximum weight by the global *idf* weight of t across all component search engines. In [10], the notion of optimal search engine ranking is proposed based on the objective of retrieving the m most similar (relevant) documents with respect to a given query Q from across all component search engines: n search engines are said to be optimally ranked with order $[S_1, S_2, \ldots, S_n]$ if for any integer m, an integer k can be found such that the m most similar documents are contained in $[S_1, \ldots, S_k]$ and each of these k search engines contain at least one of the m most similar documents. It is shown in [10] that a necessary and sufficient condition for the component search engines to be optimally ranked is to order the search engines in descending order of the similarity of the most similar document with respect to Q in each search engine. Different techniques have been proposed to estimate the similarity of the most similar document with respect to a given query and a given search engine [9,10]. Since it is impractical to find out all the terms that appear in some pages in a search engine, an approximate vocabulary of terms for a search engine can be used. Such an approximate vocabulary can be obtained from pages retrieved from the search engine using probe queries [11].

There are also techniques that create search engine representatives by learning from the search results of past queries. Essentially such type of representatives is the knowledge indicating the past performance of a search engine with respect to different queries. In the Savvy Search metasearch engine [12], for each component search engine S, a weight is maintained for every term that has appeared in previous queries. After each query Q is evaluated, the weight of each term in the representative that appears in Q is increased or decreased depending on whether or not S returns useful results. Over time, if a term for S has a large positive (negative) weight, then S is considered to have responded well (poorly) to the term in the past. For a new query received by the metasearch engine, the weights of the query terms in the representatives of different search engines are aggregated to rank the search engines. In the ProFusion metasearch engine [13], training queries are used to find out how well each search engine responds to queries in different categories. The knowledge learned about each search engine from training queries is used to select search engines for each user query and the knowledge is continuously updated based on the user's reaction to the search result, i.e., whether or not a particular retrieved result is clicked by the user.

Automatic Search Engine Connection

The search interfaces of most search engines are implemented using the HTML *form* tag with a query textbox. In most cases, the form tag of a search engine contains all information needed to make the connection to the search engine, i.e., sending queries and receiving search results, via a program. Such information includes the name and the location of the program (i.e., the search engine server) that evaluates user queries, the network connection method (i.e., the

HTTP request method, usually GET or POST), and the name associated with the query textbox that is used to save the query string. The form tag of each search engine interface is usually pre-processed to extract the information needed for program connection and the extracted information is saved at the metasearch engine. The existence of Javascript in the form tag usually makes extracting the connection information more difficult. After the metasearch engine receives a query and a particular search engine, among possibly other search engines, is selected to evaluate this query, the query is assigned to the name of the query textbox of the search engine and sent to the server of the search engine using the HTTP request method supported by the search engine. After the query is evaluated by the search engine, one or more result pages containing the search results are returned to the metasearch engine for further processing.

Automatic Search Result Extraction

A result page returned by a search engine is a dynamically generated HTML page. In addition to the search result records for a query, a result page usually also contains some unwanted information/links such as advertisements and sponsored links. It is important to correctly extract the search result records on each result page. A typical search result record corresponds to a retrieved document and it usually contains the URL and the title of the page as well as a short summary (snippet) of the document. Since different search engines produce result pages in different format, a separate result extraction program (also called *extraction wrapper*) needs to be generated for each search engine. Automatic wrapper generation for search engines has received a lot of attention in recent years and different techniques have been proposed. Most of them analyze the source HTML files of the result pages as text strings or tag trees (DOM trees) to find the repeating patterns of the search record records. A survey that contains some of the earlier extraction techniques can be found in [14]. Some more recent works also utilize certain visual information on result pages to help identify result patterns (e.g., [15]).

Key Applications

The main application of metasearch is to support search. It can be an effective mechanism to search both surface web and deep web data sources. By providing a common search interface over multiple search engines, metasearch eliminates users' burden to search multiple sources separately. When a metasearch engine employs certain special component search engines, it can support interesting special applications. For example, for a large organization with many branches (e.g., a university system may have many campuses), if each branch has its own search engine, then a metasearch engine connecting to all branch search engines becomes an organization-wide search engine. As another example, if a metasearch engine is created over multiple e-commerce search engines selling the same type of product, then a comparison-shopping system can be created. Of course, for comparison-shopping applications, a different type of result merging is needed, such as listing different search results that correspond to the same product in non-descending order of the prices.

Future Directions

Component search engines employed by a metasearch engine may change their connection parameters and result display format anytime. These changes can make the affected search engines un-usable in the metasearch engine unless the corresponding connection programs and result extraction wrappers are changed accordingly. How to monitor the changes of search engines and make the corresponding changes in the metasearch engine automatically and timely is an area that needs urgent attention from metasearch engine researchers and developers.

Most of today's metasearch engines employ only a small number of general-purpose search engines. Building large-scale metasearch engines using numerous specialized search engines is another area that deserves more attention. The current largest metasearch engine is a news metasearch engine called AllInOneNews (www.allinonenews.com). This metasearch engine currently connects to about 1,800 news search engines. Challenges arising from building very large-scale metasearch engines include automatic generation and maintenance of high quality search engine representatives needed for efficient and effective search engine selection, and highly automated techniques to add search engines into metasearch engines and to adapt to changes of search engines.

Cross-references

► Information Retrieval, ► Web Information Retrieval Models, ► Inverse Document Frequency, ► Document

Length Normalization, ► Term Weighting, ► Information Extraction, ► Query Routing, ► Result Integration, ► Wrapper, ► Web Search and Search Engine, ► Deep Web, ► Hidden Web Search, ► Web Data Extraction, ► Web Data Extraction Systems, ► Wrapper Generation, ► Wrapper Maintenance, ► Wrapper Induction, ► Wrapper Stability, ► Snippet

Recommended Reading

1. Selberg E. and Etzioni O. The MetaCrawler architecture for resource aggregation on the web. IEEE Expert, 12 (1):11–14, 1997.
2. Meng W., Yu C., and Liu K. Building efficient and effective metasearch engines. ACM Comput. Surv., 34(1):48–89, 2002.
3. Aslam J. and Montague M. Models for metasearch. In Proceedings of the ACM SIGIR Conference, New Orleans, LA, 2001, pp. 276–284.
4. Lawrence S. and Lee Giles C. Inquirus, the NECi meta search engine. In Seventh International World Wide Web conference, Brisbane, Australia, 1998, pp. 95–105.
5. Lu Y., Meng W., Shu L., Yu C., and Liu K. Evaluation of result merging strategies for metasearch engines. WISE Conference, New York, NY, 2005, pp. 53–66.
6. Lee J-H. Combining multiple evidence from different properties of weighting schemes. In Proceedings of the ACM SIGIR Conference, Seattle, WA, 1995, pp. 180–188.
7. Yuwono B. and Lee D. Server ranking for distributed text resource systems on the internet. DASFAA Conference, Melbourne, Australia, 1997, pp. 391–400.
8. Callan J., Lu Z., and Croft W.B. Searching distributed collections with inference networks. In Proceedings of the ACM SIGIR Conference, Seattle, WA, 1995, pp. 21–28.
9. Meng W., Wu Z., Yu C., and Li Z. A highly scalable and effective method for metasearch. ACM TOIS, 19(3):310–335, 2001.
10. Yu C., Liu K., Meng W., Wu Z., and Rishe N. A methodology to retrieve text documents from multiple databases. IEEE TKDE, 14(6):1347–1361, 2002.
11. Callan J., Connell M., and Du A. Automatic discovery of language models for text databases. In Proceedings of the ACM SIGMOD Conference, Philadelphia, PA, 1999, pp. 479–490.
12. Dreilinger D. and Howe A. Experiences with selecting search engines using metasearch. ACM Trans. Inf. Syst., 15 (3):195–222, 1997.
13. Fan Y. and Gauch S. Adaptive agents for information gathering from multiple, distributed information sources. AAAI Symposium on Intelligent Agents in Cyberspace, Stanford University, 1999, pp. 40–46.
14. Laender A.A., Ribeiro-Neto B., da Silva A., and Teixeira J. A brief survey of web data extraction tools. ACM SIGMOD Rec., 31 (2):84–93, 2002.
15. Zhao H., Meng W., Wu Z., Raghavan V., and Yu C. Fully automatic wrapper generation for search engines. World Wide Web Conference, Chiba, Japan, 2005, pp. 66–75.