Creating Customized Metasearch Engines on Demand Using SE-LEGO
(Extended Abstract)

Zonghuan Wu[1], Vijay Raghavan[1], Weiyi Meng[2], Hai He[2], Clement Yu[3], and Chun Du[1]

[1]University of Louisiana at Lafayette, Center for Advan. Computer Studies
Lafayette, LA 70504, {zwu,raghavan}@cacs.louisiana.edu
[2]Department of Computer Science, State University of New York at Binghamton
Binghamton, NY 13902, USA, meng@cs.binghamton.edu
[3]Dept. of Computer Science, University of Illinois at Chicago
Chicago, IL 60607, yu@cs.uic.edu

1 Introduction

Frequently, the documents needed by a user are available only via multiple search engines. For example, research papers about a particular subject may be found from the search engines of related digital libraries and journals. It is inconvenient for the user to search these search engines separately. An effective way to address this problem is to employ a metasearch engine, which is a system that provides unified access to multiple existing search systems. When a metasearch engine receives a user query, it passes the query to its underlying search engines. The results returned by the search engines, are then combined by the metasearch engine to form a single ranked list for presentation to the user [2].

Building customized metasearch engines is important to many people and organizations. For example, a researcher may use a particular set of search engines for finding papers on a particular subject. A customized metasearch engine based on these search engines will provide convenience and efficiency for this researcher. As another example, a company may have several competitors and it keeps track of these competitors using their search engines. In this case, a customized metasearch engine on top of the competitors' search engines can be very useful to the company. In both examples, the set of the search engines one wishes to use as well as the characteristics of the search engines themselves may change (e.g., a competitor's search engine may need to be added and a search engine changes its result format). Therefore, the metasearch engines need to change accordingly. Currently, building and maintaining a metasearch engine are expensive and labor-intensive tasks that need diverse expertise. As a result, it is difficult for an ordinary Web user to create and maintain a metasearch engine based on the search engines of the user's choice. Some metasearch engine companies (e.g., ProFusion) allow users to build customized metasearch engines, but only search engines in a pre-compiled list can be used because the capability to connect to these search engines needs to be established in advance.

G. Dong et al. (Eds.): WAIM 2003, LNCS 2762, pp. 503–505, 2003.

In this demonstration, we present an automatic metasearch engine construction tool, SE-LEGO. When the URLs of the desired search engines are provided, SE-LEGO creates a customized metasearch engine based on these search engines on demand.

SE-LEGO is also useful for building large-scale metasearch engines connecting to numerous search engines. It is estimated that there are hundreds of thousands of search engines on the Web, including both the Surface Web and the Deep Web [1]. At present, the largest metasearch engines such as ProFusion (www.profusion.com) connect to about 1,000 search engines. This means that only a small fraction of the information sources on the Web are connected. The goal of our WebScales project is to create a metasearch engine that connects to all useful search engines on the Web. Clearly, in the context of WebScales, it will be too costly to manually produce the connection program for every search engine. Furthermore, changes/upgrades of the connection format of a search engine may affect the connection program, causing a maintenance nightmare when a huge number of search engines are involved. The automatic connection capability of SE-LEGO is necessary for our WebScales project.

2 Components of SE-LEGO

SE-LEGO consists of the following components:

1. *Automatic Search Engine Connection*: This component automatically analyzes the source (HTML) file of the interface of any given search engine and generates a program that can pass queries to the search engine. Based on the analysis, important attributes for search engine connection such as the URL of the search engine service agent program, the HTTP communication method and other parameters in the search engine form are extracted. The extracted information is then used to automatically generate the connection program for the search engine.
2. *Automatic Search Result Extraction*: For any given search engine, this component automatically generates a program to extract the results (e.g., URLs) related to the retrieved pages from the result pages of the search engine. Typically, a search engine result page contains not only the URLs of retrieved documents but also URLs of advertisement/internal organization pages. The issue is how to differentiate useful URLs (those of retrieved pages) from useless URLs automatically. This component also extracts the number of hits and the next page pattern for retrieving more result URLs than shown in the initial result page.
3. *Query Dispatching and Result Merging*: This component dispatches queries to appropriate search engines and merges the results extracted from the returned pages into a single ranked list for presentation to the user. Different result merging algorithms are implemented. The basic algorithm ranks the results based on their local ranks in local search engines. Another algorithm ranks the retrieved documents based on a global match of the contents of each page with the query.

3 Demonstration

During the demonstration, we will show how to use SE-LEGO to build a metasearch engine on demand and then conduct metasearching. People in the audience are welcome to provide the URLs of Web search engines for the demonstration.

Acknowledgements. This work is supported in part by the following grants from NSF (IIS-0208574, IIS-0208434, EIA-9911099), Army Research Office (ARO-2-5-30267), and the IT Initiative of the State of Louisiana to Lafayette.

References

1. M. Bergman. The Deep Web: Surfacing the Hidden Value. BrightPlanet White Paper (www.completeplanet.com/Tutorials/DeepWeb/index.asp), 2000.
2. W. Meng, C. Yu, K. Liu. Building Efficient and Effective Metasearch Engines**.** ACM Computing Surveys, 34(1), March 2002, pp.48–84.