

**BUCKNELL UNIVERSITY
UNDERGRADUATE RESEARCH PROPOSAL
SUMMER 2010**

**Designing, Implementing, and Testing of a Intelligent
Crawler to Support Vertical Search**

**DENIZ GORGUN
CLASS OF 2011
BOX: C3168
BUID: 10799021
dg024@bucknell.edu**

**XIANNONG MENG (FACULTY MENTOR)
PROFESSOR OF COMPUTER SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
xmeng@bucknell.edu**

Student signature:

Date:

Faculty signature:

Date:

PART A

Project Description

In this age of information technologies, it is crucial for us to retrieve any information we demand, whether it be on a specific topic of interest or research subject, or personal information through social networking. It is also important for the prospective audience that this information available online is relevant, now that the web is rapidly growing. Search engines are a very effective way of establishing this connection.

There are two types of search engines: *horizontal* and *vertical*. As in the traditional examples of Google, Bing and Yahoo, horizontal search engines are generally greater in search volume. These search engines search through the entire web to generate results, but do not classify the resulting sites in categories. A vertical search engine, on the other hand, searches within specified categories or sites which are relevant to the subject of the query, making the search volume considerably lower. Even though these search engines receive considerably less traffic compared to the horizontal search engines, a big advantage of vertical search over horizontal search is the capability of generating content specific results.

An example implementation of vertical search engine concept would be Google Scholar¹. This freely accessible search engine is indexing scholarly articles and peer reviewed journals published all around the world from a variety of disciplines. When a search is made through Google Scholar, the engine returns and ranks the matched results via an algorithm that takes into consideration the publish date and citation count of each article. This algorithm allows the user to find more relevant and valid articles more efficiently[1]. Another feature that gives Google Scholar greater coverage is its ability to crawl into the repositories of major private publishers such as JSTOR or Elsevier. There are other public search engines that focus on crawling academic and scientific documents as well. Search engines like CiteSeer², getCited³ or Scirus⁴, rank the articles using similar algorithms to that of Google Scholar.

Even though these free implementations of vertical search on scholarly documents exist, there is a lack of specialization over college coursework. According to the 2004 Digest of Education Statistics⁵, there are more than 4000 institutions offering post-secondary education, and we think that many of these colleges apply similar degree programs. Students from College A, majoring in Computer Science and Engineering have a good chance of having similar curricula to students from College B, C and D. In such a case, we think that any means of collaboration in between these students and colleges would provide benefits to the quality of education. Finding solutions through different approaches to different problems could only be possible through the means of communication that is available. What Professor Meng and I are planning is to create

¹ <http://scholar.google.com>

² <http://citeseer.ist.psu.edu>

³ <http://www.getcited.org>

⁴ <http://www.scirus.com>

⁵ http://nces.ed.gov/programs/digest/d04/tables/dt04_244.asp

such channel for communication that would not only be easily available, but also be easy to use, generating effective results.

We will create our vertical search engine to crawl into a specific set of sites and create indices from the related pages that are a part of these sites. We will then parse the resulting indices, and narrow down our gathered information, since there is a great chance of coming across irrelevant information that we do not need. Collecting keywords will give us credible information and discarding the irrelevant information from our repository will enable us to concentrate on specific areas, such as courses. With continuous cycles of crawling and parsing, we are aiming to selectively re-index our contents, to make the search engine more accurate.

Methods

A regular crawler retrieves Web pages, starting off with the URL for an initial page, and extracting any URLs in it. It later adds them to a queue to be scanned, parsed and indexed [2]. This process will result in building a massive repository of various data from the targeted sites. To make the process a more effective one, our design will try to match the raw extracted URLs with the specified, topic related keywords to re-index them. Consequently, by eliminating the irrelevant indices, we will save memory space in our server. Another benefit of this process will be the shortening of the response time of a searched string. Since there will be fewer indices to analyze, relevant information will be accessed quicker.

Outcomes

The project will build upon Prof. Meng's past work on search engine optimization, and will base on the software Search Engine Made Easier (SEME), a small scale search engine written by Professor Meng and his previous students. Our primary goal is to re-build SEME, as previously described, to optimize the crawling algorithm and data structures such that it can collect optimal amount of information on specific subjects, such as computer science curricula across departments in the country. In addition, we are planning to publish this new software to become a publicly available tool for teaching and undergraduate research purposes, by creating a channel for communication among colleges. Lastly, we are expecting to gain understanding over the development of the technology, and create possible opportunities for future research on the topic.

Milestones

We expect the research and analysis of various current models to take approximately two weeks, as this is one of the most crucial steps in the project. Upon completion of the analysis, the second step will be getting acquainted with the scaffold of the model, to set the design specifications of the crawler, so that it is compatible with the rest of the code. The next step will be the design and implementation of the crawler, which we expect to take the longest time, about four weeks. Lastly, testing and optimizing the entire model will conclude the project. Although we plan the testing part

to take about two weeks, this might not be enough, in which case I will spend the rest of my summer working on perfecting the implementation, clearing out the bugs. Here is a sample timeline:

- Weeks 1-2: Investigate literature on theory and technology behind the vertical search engine architectures.
- Weeks 3-4: Design and start implementing an intelligent crawler algorithm.
- Weeks 5-8: Finish implementation of the entire model and start execution.
- Weeks 9-10: Run the search engine through rigorous testing and make changes on required areas.

As we are finishing the project, we will try to make room for future developments that may be possible to add onto the ground work. I would like the project to have a system that is set up to enable updating the crawled information every two to four weeks, to keep the indexed data constantly validated. Another idea for future development is to have an open ended memory allocation for the repository, which will enable increasing the reach of the search engine.

PART B

Research Environment

Professor Meng and I will be in close contact for the duration of the project, communicating both personally and via internet media. Professor Meng has offered guidance as an important part of the project for whenever I need clarification on any topic.

We have decided to use the Java programming language for the development of the crawler, based on the compatibility with the scaffold that Professor Meng made available, as well as the increased string processing and networking capabilities that come with this programming language.

References

- [1] Brin, S. & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. In *Proceedings of the Seventh International World Wide Web Conference (April 14-18, 1998, Brisbane, Australia)* 107-117. Retrieved July 31, 2009 from: <http://infolab.stanford.edu/~backrub/google.html>
- [2] Cho, J., Garcia-Molina, H. & Page, L. (1998). Efficient crawling through URL ordering. In *Proceedings of the Seventh International World Wide Web Conference*, 161-172. Retrieved July 31, 2009 from <http://ilpubs.stanford.edu:8090/347/>

[3] Meng, X. (2008). Web search engine architectures and their performance analysis. In Munoz, C.C., Moraga, M.A. & Piattini, M. (Eds.) *Handbook of research on web information systems quality*, 491--509. Hershey, PA: Ideal Publishing.