Class Project Proposal

Project 5: Web Search Engine

Students (listed alphabetically by last name then first):

Sanner Barnes :: sannerbarnes@gmail.com
Leith Brandeland :: leith98b@gmail.com
Josh Guthrie :: josh.x.guthrie@gmail.com
Joseph Johnson :: joejunior1984@gmail.com
Revarr Johnson :: rmj092020@utdallas.edu
Revanth Segu :: Revanth.Segu@utdallas.edu

The Problem

We've decided as a group to build a search engine to cater to programmers that can be used to lookup solutions to common programming problems, such as programming errors encountered. Per the Project 5 description we may also have to open this up to more topics from DMOZ, and if so they will likely center around Programming-related paradigms.

The Methodology

The search engine that we will design will be comprised of a few main logical features as described in the Project 5 description:

- *Crawling Module*:
  This will be the module that collects pages from our chosen sites in DMOZ up to the min/max document requirements. We are investigating Crawler options, whether it's building our own or using an existing, open-source one.

- *Indexing Module*:
  This is the module that will store our documents/postings as an inverted index. We do plan to use Lucene for this.

- *Link Analysis Module*:
  This module will help us implement page ranking and will contribute to our retrieval module. We are still considering our ranking options, but the idea is to either use Stanford/Google's PageRank algorithm or to use the Topic-Sensitive version of PageRank, as time allows. The topic sensitive pagerank may be useful to narrow pages down by technology stack (e.g. Java vs C++) if we can train a good enough classifier, but we will see what time permits..

- *Retrieval Module*:
  This module will be used to compute our ranking for the documents and determine relevance. As we are using PageRank, we will likely use a probabilistic model, but we may also experiment with a vector model and perhaps use a methodology such as FastCosineScore to efficiently compute the document/query vector scores. We really hope to achieve a better understanding of ranking/scoring after completing this module.

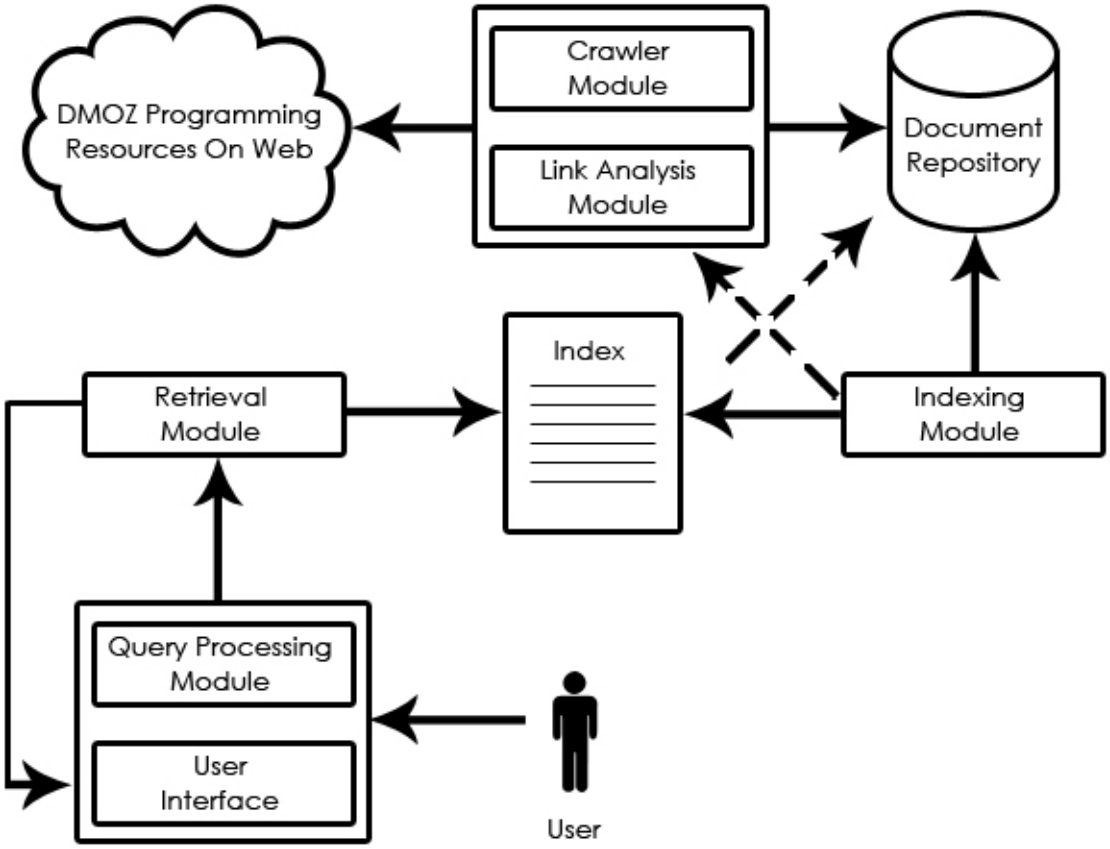
- *Query Processing Module:*
  This will be the module that retrieves the user's query, transforms it into something useful that can be interpreted by our retrieval module, and shows the results side by side with Google and Bing. As this is the user interface, it is also a major piece in itself (what the user will see/interact with) and we hope to give it a good treatment, perhaps something unique related to our programming-solutions paradigm (e.g. show the results in a horizontal scroll format instead of a vertical one, with a frame below it, is one idea we tossed around as a group, as users searching for programming-solutions tend to perform a lot of very quick "click-through and scan" operations, jumping from page to page somewhat randomly until the solution jumps out; the frame and vertical format would allow the results to remain in view of the user so that they could keep from having to hit "back" a lot, which is a problem today in Google if you perform these kinds of searches—wastes a lot of time for the user in the form of too many "go back, rescan the results, click next" operations, which consume a non-trivial percentage of the user's time).

The Text Data

To make the problem tractable, the data we would focus on might be for a particular language or technology stack (e.g. JAVA, or PHP, or .NET) and focus on those sites that have high authority in the DMOZ (such as Stack Overflow), or on particular types of errors (e.g. syntax errors, for instance, as opposed to runtime errors). This is something we will have to experiment with based on how much useful data we get back from crawling within the bounds of the project's crawling requirements.

Alternatively, we could consider filtering around hello world tutorials in our crawling (such that many more languages can be covered, but the scope of the problem we would solve would be severely limited—e.g. queries pertaining to hello worlds or initial environment setup, but over a broader spectrum of languages). We are limited here by the scope of the task and the amount of data we have so we will try to be clever in our target and see how much useful data can be returned in our crawling and what heuristics we can inject into our crawler to prefer the kinds of pages we are looking for.

Block Diagram of Project



Implementation Responsibilities

The tasks are divided like so (this is our project plan).

Crawl and collect relevant documents	Rivarre, Leath	Need to gather and target good sites to build data from and figure out a method for crawling; need to talk to Sanda about crawling and review upcoming lecture for this? Should we include link analysis in your guys' team? Maybe you can work with Joe/Raventh on this.	Nov 2 - Nov 16
Index construction	Sanner, Josh	Can use Lucene or build or own.	Nov 2 - Nov 16
Interface / UI, Query Processing	Leath, Josh	PHP, C#? This might also include stemming the query/performing lemmatization, spelling correction, or doing query expansion if we have time to do that.	Nov 2 - Nov 16
Retrieval Method and Link Analysis	Joe, Raventh	Code for performing document retrieval using the parsed query, link analysis, PageRank, etc.	Nov 2 - Nov 16
Ranking Method	Joe, Raventh	Figure out relevance/ranking that the retrieval model will use. Vector model, FastCosineScore, probabilistic with PageRank; topic-sensitive-PageRank?	Nov 2 - Nov 16

Assemble components	ALL	Once we design all of the individual pieces, we'll need to get together and make them all talk to each other, work out bugs, do a shakedown, revisit bad assumptions, and just make the whole search work end-to-end for the first time	Nov 16 - Nov 23
Presentation	ALL	Let's all come up with a few slides that describe our respective components and then assemble them into one presentation, if a formal presentation is required. We will also update the proposal to list our formal experimental results and conclusions.	Nov 16 - Nov 23
Final testing and presentation rehearsal	ALL	Do another round of testing and rehearse for presentation	Nov 23 - Nov 30

Experimental Results

This will come at the conclusion of our successful project.

Discussion

This will come at the conclusion of our successful project.

Conclusion

This will come at the conclusion of our successful project.