I427 Final Project Report-Spbooth

ATTENTION: All burrow programs using python3, not python

A. Back-End programs explanation

I. Crawler

The crawler runs the same way it did in assignment three. It takes in a seed url, a page limit, a directory, and a search algorithm.
Sample input line:

python3 crawl.py https://www.reddit.com/ 10 pages/ bfs

II. Index

The indexer runs the same way it did in assignment for. It takes in a directory and a index file name.
Sample input line:

python3 index.py pages/ index.dat

III. PageRank

The page rank file takes in one input which is a graph in a dictionary form with the keys being a node (url) and the values being a set of nodes (urls) that it connects too. It outputs the PageRank dictionary as well as writes the dictionary to a dat file using pickle.
Sample input line:

##python3 index.py pages/ index.dat

IV. Final

The easiest way to run all the of the background programs is to run Final.py. This program takes in a seed url, a page limit, a directory, and a search algorithm. It imports the other programs and runs the crawler using in aforementioned inputs, then runs the index program using the given directory and the index.dat file. Then it runs the PageRank program using the web graph output during the crawl program.
Sample input line:

python3 Final.py https://www.reddit.com/ 10 pages/ bfs

V. Web retrieve

Web retrieve is not an application that is run on burrow. It is a modified version of the retrieve function that outputs the list of hits or a string that says an error message or a results message. It is the function that the cgi program calls.

B. https://cgi.soic.indiana.edu/~spbooth/i427/i427.html

C. To find the hit list I made a got each url that contained each query word and counted the number of times each url occurred. Then I got the minimum number of query words were needed by calling a function that took the mode and the number of query terms and returned the minimum based on the mode. Then if any of urls met at least the minimum number of query terms, they were added to a set. For each url in the set, the tfidf was calculated then that number was multiplied by the page rank after loading in the page rank dat file, to give the total score of the page. We decided on multiplying the scores because we thought it was the most effective way to get the importance score. The score, url, and title were all returned as a list of lists to be used by the cgi file.

D. Evaluation Reports

First person:

1. How long it takes for searching: 5-10 seconds
2. Are the results fit the expetation: Most of them. Even though not as good as Google.
3. What percentage of the time found a page interesting in top 10 hits: 85%
4. How many pages are interesting/valuful in top 15 pages: 9
5. How satisfy you feel during searching (score from 0-10, 10 is best): 8
6. Other comments/suggestion: It is good to have a large number of results (50+ urls show up), but list them in different pages, so each page show only 10-15 urls is good.

Second person:

1. How long it takes for searching: About 10 seconds
2. Are the results fit the expetation: Large percentatage of them.
3. What percentage of the time found a page interesting in top 10 hits: 70%
4. How many pages are interesting/valuful in top 15 pages: 7
5. How satisfy you feel during searching (score from 0-10, 10 is best): 7
6. Other comments/suggestion: The searching time is longer than I expected. Some url results are not that interesting.

Third person:

1. How long it takes for searching: 7-9 seconds
2. Are the results fit the expetation: Large percentatage of them.
3. What percentage of the time found a page interesting in top 10 hits: 80%

4. How many pages are interesting/valuful in top 15 pages: 11
5. How satisfy you feel during searching (score from 0-10, 10 is best): 9
6. Other comments/suggestion: Good work. Very comprehensive results. Maybe choose a better server or improve algorithm so that the result can show up faster.

E. The extra credit we did was implements bigram search and retrieval
F. Azadeh, Pik Mai, Thomas