Project Progress Report

Michael Potts (mjpotts2)
Group Name: Solo

Which tasks have been completed?

So far, I have completed the general frame work for my webcrawler to utilize only pages from wikipedia. The general idea so far is that the webcrawler program will land on a page, copy the contents of the page, store the data in a text file with the first line being the URL, the second line being the title, the third line being the date it was written, and the remainder being the content of the document, with each line being a new sentence. Once complete, the text file is closed, the program will then jump to a new page, and repeat the process N times as specified by the user. Further to this, the program loop is limited by a sleep function to ensure wikipedia is not overloaded with requests. Each URL and date gets copied to a reference text file in alphabetical order which will be checked each time the program jumps to a new page. Before the program copies any page information, the URL and date will be checked to see if 1. the file exists already, and 2. to see if the file is up to date. The page rank algorithm has not been implemented yet, but the program is ready to incorporate it.

Which tasks are pending?

So far, I have not completed the page rank algorithm, and some further things I hope to implement include performing PLSA or LDA on the resultant file collections. For this step, the program will ignore the URL, and date, but will use the Title and contents.

As another idea, I also thought it might be interesting to create a graph system of the documents the program has scanned so far. This of course will be very time dependant though. This idea would treat every document as a node, and every URL in the document as an edge. This could be an interesting way to identify hubs and authorities on the wikipedia documents I gather.

Are you facing any challenges?

I have not run into any major challenges as of yet. Some things that I needed to update myself with was how to read webpage data to extract the information I was searching for specifically without the text markup. Further to this, I expect that I may run into some difficulty implementing the algorithms as the Page Rank algorithm may involve some decision making about where it jumps, if it runs out of unique URLs to jump to. One solution to preemptively solve this would be to reference the gathered URL list before jumping to ensure the algorithm does not get stuck in an infinite loop. As for where to jump, It could be beneficial to keep an entire list of URLs not yet visited from each page, and then allow the algorithm to randomly select one of the URLs. This would allow my crawler to also crawl over similarly clustered (or referenced) information at first. This may introduce slight bias for the second step of performing PLSA or LDA, but only with fewer documents. The more documents my program jumps to, the less resulting bias would remain.