**Research Notes**

**Web Crawler:**

* Run separately from the many program so that a crawl list can be created with URLs, names and keywords relating to the webpage
* Crawl list needs to be ordered in some way so that a binary search can be performed on it when the user searches using the search engine
* Need to find a way to ensure that only relevant webpages are put into the crawl list so that it isn’t unnecessarily long – ie a page may be stored there even though it doesn’t have any relation to the search – maybe a way to stop this is by having the program check if the page contains any of the keywords in the search or can determine the relevance of the page in some way and if it doesn’t meet a certain threshold then it’s not added to the list.
* It’s good to make use of some ranking algorithm to determine how relevant a webpage is and where it should be placed in the list of webpages that is returned to the user – lots of algorithms already out there that I can try. (Chung, 2014)
* Where a page is ranked often depends on the number of links to and from that page there are – shows relevance
* Search of crawl list could be tough because only one keyword could be in the title of the website so a binary search might not work as this will only check the title/first keyword – need to think of the best way that a search can be done – ie what do you search for and how do you store and represent the data in the file
* Probably best to run the web crawler fairly recently so that it’s updated with new sites/changes to previously stored sites and can give a new ranking for each of them

**Search Engine:**

* Will search the crawl list for webpages that the crawler has found and ranked
* This is a lot faster than using a web crawler to get results for each search
* UI is pretty simple – only need text box to enter search, button to confirm entry – then when search results are returned and sent to the user, an area for this will appear
* Need to add a mechanism that lets the user go back to the previous page – best way is to store the previous webpages in a queue so that they can then be popped when the user goes back once
* Could maybe have a forward button as well?
* Webpages should be shown to the user in a clear and easy to understand layout showing the user exactly what each page is about
* Maybe don’t load all of the pages at once in order to save space and time in terms of loading. Instead, the next x pages can be retrieved if the user has looked at the first x pages that have been shown to them.

**Other Areas:**

* Could be useful to design my own search method for the crawl list instead of using some built in function so that its more tailored to what I’m trying to do.
* How/where to store different files?

**Note:**

I think it would make sense to create a new txt file for each new user of the system. The txt file will store data about them like what websites they’ve interacted with and their recent search history.

Then, a python file can interpret this data and create a user profile of trends and interests that I can hopefully use along with the crawl list to select the best matches for the current user.

Chung, F. (2014) ‘A Brief Survey of PageRank Algorithms’, *IEEE Transactions on Network Science and Engineering*, 1(1), pp. 38–42. Available at: https://doi.org/10.1109/TNSE.2014.2380315.