Project Report
By: Sahil Gupta

1. If it is not blatantly obvious, please indicate where in your source code the PageRank algorithm exists.

Inside the src folder there is the file: Main.java. The code for PageRank algorithm is inside this file.

2. Description of system, design tradeoffs, etc. Focus on how you handled the large file, or any issues that came up

There is one class Main class. File contents from a .gz extension file are read through

- The load() function reads a .gz file which is passed through it. Subsequently the graph containing the page and its outline and an inLink HashMap is made inside the function.
- The pageRank() function computes the page rank for all the pages in the graph.
- The sorting function which utilised creating linked list from the hashmap sort the inLink map and revisedPageRank. The top 75 results from these are stored in text files.
- The writeFile() function takes in parameter a file path and the content to write in it.
- Since the file was large, I utilised StringBuilder for memory management purpose while writing content into files.
- For traversing through hashMap, I utilised the foreach methods and entrySet() methods as well.
- Choosing a method for checking convergence was a design choice. For checking whether the page rank is converged I chose to calculate the square root of sum of squares of difference between the initial and revised pagerank values. I then compared if it was less than a value ‘tau’ which is provided as a parameter. If so then the page rank has converged else iterate again and check.
- For reading such large amounts of data, I utilised GZIPInputStream for the reading the links.srt.gz file. It reads compressed data in the GZIP file format. It reads data into memory which is more efficient than file input output operations.

3. List the software libraries you used, and for what purpose. Implementation must be your own work. It is OK to use a regular expression library for this task as well. If you are thinking of using any other libraries for this assignment, please send us an email to check first.

- For understanding how writing into a file works I referred to https://www.geeksforgeeks.org/java-program-to-write-into-a-file/.

- In order to write top 75 pages I wanted to sort the HashMap. I referred to geeksforgeeks page https://www.geeksforgeeks.org/sorting-a-hashmap-according-to-values/ for understanding sorting a HashMap.

- import java.util.Map: building the graph and initial and revised configurations of the page rank for all pages
- import java.util.HashMap: building the graph and initial and revised configurations of the page rank for all pages
- import java.util.HashSet: Storing all outlines from a page
- import java.util.Set: Storing all outlines from a page
- import java.io.BufferedReader: reading content from a file
- import java.util.zip.GZIPInputStream: reading content from a file
- import java.lang.Math: calculating norm for the initial and revised page rank to check for convergence
- import java.nio.file.Files: for writing content into a file
- import java.nio.file.Path: for writing content into a file
- import java.io.IOException: for throwing and catching expiation if encountered during file reading
- import java.util.List: creating a linked list for sorting inLink HashMap and pageRank HashMap
- import java.util.LinkedList: creating a linked list for sorting inLink HashMap and pageRank HashMap
- java.util.Collections for sorting the elements of HashMap.

4. Discuss the differences between the pages with highest inlink count and those with highest PageRank. What might cause this difference?

The difference is that there are some pages which have better page rank with less unlinks and there are some pages with lesser page rank even with higher number of inlinks. The reason for this might be that even though there are more inlinks for certain pages, but the page rank of those links wouldn't be high. Meaning that those pages wouldn't be of high quality which is why these pages have a low page rank score. On the other hand for those pages that appear in top list in pagerank.txt and not in inlink.txt although have fewer pages pointing to them, but those pages themselves would have higher page rank score. This is the cause for the difference

5. **What would happen if you initialized the PageRank scores to random values instead of even scores? What about zeroes? What if you remove the random-surfer component? Will it converge faster? Slower? Not at all?**

- If the PageRank is initialised to 0 then it converges at a slower rate but will definitely converge since the random surfer is present.
- If the PageRank is initialised to random values then it will converge but the rate varies. If the random values are closer to the true values then it will convene faster

else if they are very different from the true values then it will converge at a slower rate.

- If the random surfer component is removed by setting alpha to 1 then it will not converge as (1- lambda) become 0 so it will never go to a random page if there are no outlines from a page.