ReadMe and Report

To run this application you need to have these libraries, files and directories:
Task_1:
Libraries: BeautifulSoup, nltk, matplotlib, urllib, string, re, json and also lxml
Files: BFS.txt, FOCUSED.txt
Directory: scrap

Task_2:
Libraries: Urllib, BeautifulSoup, collections.
Files: g1.txt, g2.txt (generated from task1).

Task_3: this task is for question 4a and 3d
Libraries : numpy, collections
Files: g1.txt, g2.txt and ungrade.txt(included in this submission)

Task_4: this task is for question 4-b.
Libraries : numpy, collections
Files: g1.txt, g2.txt

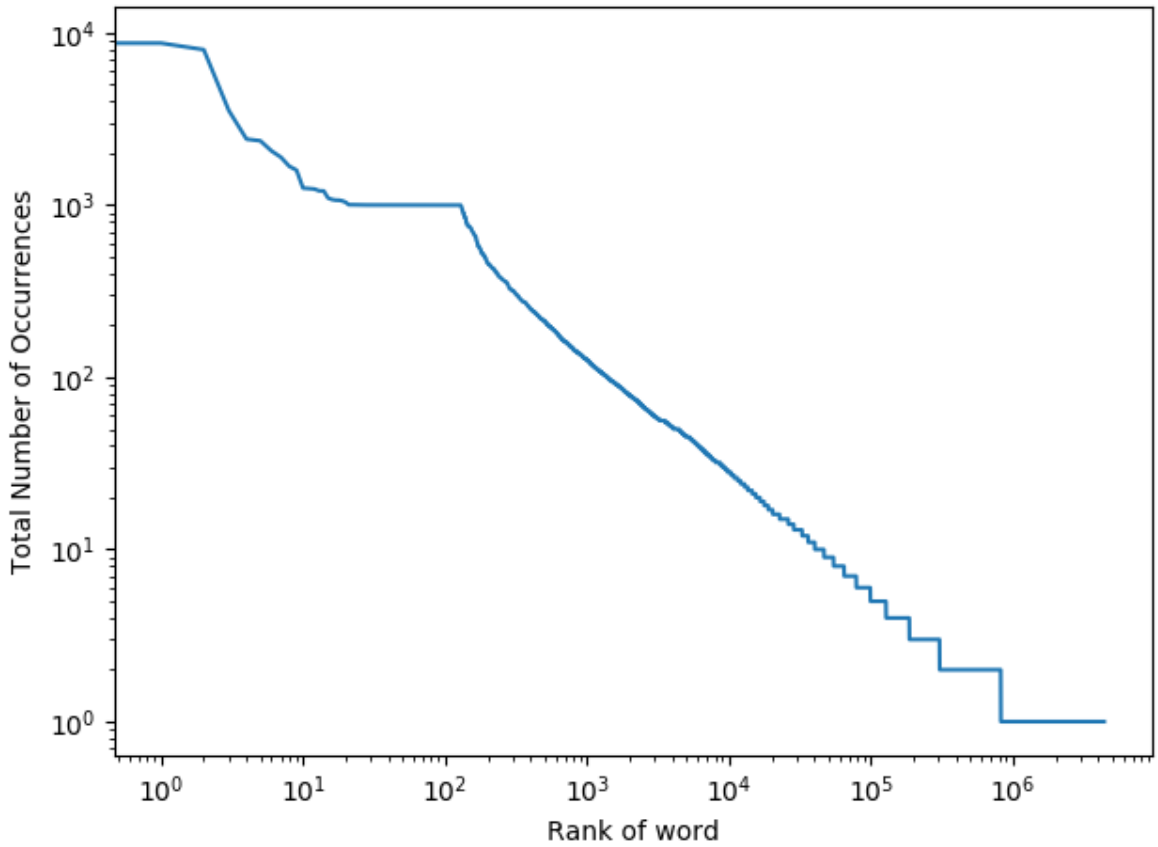
The code is in python 3. Make sure you have scrap named directory and BFS.txt , FOCUSED.txt where you try to run your task.

To run each task run python Task_X.py where X is the number of task. E.g.
  python Task_1.py for task 1.

Task 1:

The file for trigrams and corpus are:
All the corpus are in scrap file directory with required names and all other requirements.
File listing all trigrams in sorted order is “abc.txt” and with rank data with corresponding constants are in file “constant_rank.txt”

Log-log graph of frequency vs rank is provided below. This is generated after each run in SciView in intelliJ pyCharm. We can see that it follows the zipf’s law except the low and high values of ranks.



Task 2:

For task 2 graph files for BFS and Focused are made in g1.txt and g2.txt respectively.

1. The number of pages with no in-links (sources)

   G1: 0, G2: 0

2. The number of pages with no out-links (sinks)

   G1: 1 (ceres), G2: 228

3. Maximum in-degree

   G1: 344, G2: 464

4. Maximum Out-degree

   G1: 236, G2: 236

In this we can see that there are no pages which has zero in links. In raw data we can observe every page has some incoming link. For sinks there is just one for G1 and more than 200 for G2. Similarly statistics are provided for maximum in degree and out degree for both G1 and G2.

Task 3:
File listing l2 norm values and sum of page ranks until it converges is in g1_norm.txt and g2_norm.txt respectively for G1 and G2.

Sorted pages in G1 and G2 by page rank with document Id and rank are in top_50_g1.txt and top_50_g2.txt. After serial numbers from 1-50 the list contains ('documentId', PageRank) in this format. If you run Task_3.py the result in the console is the page rank in descending order.

You can run task 3-d by using existing code in Task_3.py but to run code in task 4-a you need to uncomment each line e.g list_sort = pr.produce(lamb=0.25) to required configuration of lambda. Smoke test of ungraded graph is also added in this. It runs as expected with F and C have same rank 0.15130589604515457 and order A>E>(C=F)>B>D.

Task 4:

Question: Re-run the PageRank algorithm using l = 0.25, l = 0.35 and l = 0.5. What do you observe in the resulting PageRank values relative to the baseline? Discuss the results.

Ans: Baseline takes 5 iterations to converge at l2 norm of 0.015514913951889602 and d = 0.2, has l2 norm= 0.01368962995754868, 0.35 has l2 norm= 0.011864345963209644 and 0.5 has l2 norm of 0.00912641997169912. All take 5 iterations to converge. The top 50 pages have similar ranking, the order is different. This is because at higher d values randomness becomes more contributive than links in the pages.

Question: Re-run the PageRank algorithm in Task3-d) for exactly 4 iterations. Discuss the results obtained with respect to the baseline.

Ans: The top 50 ranking page has very similar convergence l2 norm values. I think this is because in previous assignment we crawled max 3 depth in BFS and most the links were we crawled were enough to cover the path in 4 iterations. Therefore ranking becomes stable very fast.

Question: Sort the documents based on their raw in-link count. Compare the top 25 documents in this sorted list to those obtained in Task 3-d) sorted by PageRank. Discuss the pros and cons of using the in-link count as an alternative to PageRank (address at least 2 pros and 2 cons).

Ans: Pros:

1. This is just sorting therefore it runs really fast.
2. It also discovers popular pages based on links in the pages.

Cons:

1. Results tends to be biased towards the initial seed given. Pages which are near seed links gets higher rank.
2. In link is just an integer and a lot of other pages have same in links. This does not help in page ranking for certain topic.