Nicolas Escobar (escobarn)  
Harsh Reddy (hagandav)

Cloud Computing - Project 2 Report

Technical report that contains:

1. **The description of the main steps and data flow in your program.**

Hadoop Page Rank:

The ‘HadoopPageRank’ package is the main application/driver where all PageRank Hadoop jobs are configured and executed.

1. Create Graph Reduce:
   1. This is the Initial Map/Reduce job that converts elements in the adjacency matrix format into key/value pairs.
2. Page Rank Map:
   1. The ‘PageRankMap’ task is the second job to be executed in one or more iterations as part of the ‘HadoopPageRank’ application, where each Map task processes a line in the adjacency matrix file which is the output produced by the CreateGraphReduce task.
   2. For each line, the value (source URL, pagerank value and list of target URLs of a node/web page) is decomposed using the ‘RankRecord’ class.
   3. The application then checks the outdegree of the node (whether the target URL list is higher than zero or equal to zero).
   4. If the outdegree is zero, then the pagerank value of the web page is distributed among all other nodes
   5. If the outdegree is higher than zero, then we iterate over the list of target URLs and generate the key/value pair *<targetURL, rankValueTargetUrl>* for each targetURL where the ‘rankValueTargetUrl’ is the division of the source node rank value by its out degree
   6. Finally, a key/value pair *<sourceURL, #targetURLs>* is generated for each web page

Page Rank Reduce:

* 1. The Reduce task processes the output generated by the ‘PageRankMap’ task
  2. For each line, the value is split based on “#” characters and the result stored in an array.
  3. If the size of the array is zero, then the value contains the *<targetURL, rankValueTargetUrl>* key/value pair. In this case, we sum up all of the pagerank values for the corresponding key (web page).
  4. If the size of the array is larger than zero, it means the value contains the *<sourceURL, #targetURLs>* key/value pair. In this case, we concatenate all of the target URLs for the corresponding key (source web page).
  5. Finally, the total sum of pagerank values is calculated considering the damping factor formula. Thus, the key/value pair or *<URL, sumOfPageRankValues#targetUrls>* is generated as the output of the Reduce task

1. Cleaning Up Results:
   1. The ‘CleaningUpResults’ Map/Reduce job collects the final *<URL, PageRank>* from previous output
2. Sorting Rank Value:
   1. This final ‘SortRankVaue’ Map/Reduce job sorts the *<URL, PageRank>* by the PageRank value in descending order. The Map task that is part of this job submits all values with the same key to the Reduce task which performs the actual sorting by storing all key/values in a HashMap and then applying a Comparator on the values. The other option was to implement a custom Comparator in the MapReduce framework but the size of the data was not big enough to justify a more complex solution.
3. **The output file (username HadoopPageRank output.txt) which contains the first 10 urls along with their ranks, sorted by rank values in descending order**

**5000 nodes with 5 iterations**

PageRankDataGenerator/pagerank5000g50.input.0 **5000 5**

4 0.12090657661891116

34 0.11580741387490097

0 0.09562223453436994

20 0.07501633581378547

146 0.0430260594551136

3424 0.03834387157392686

2 0.037649178381750845

14 0.013800778832186871

12 0.012069210696238066

16 0.011724665184813084

**5000 nodes with 15 iterations**

PageRankDataGenerator/pagerank5000g50.input.0 **5000 15**

4 0.1213363763921075

34 0.10843721973366303

0 0.09661702370554635

20 0.07691467659648708

2 0.036829181451109604

146 0.03501791690238924

3424 0.03082143838128042

14 0.016391488480333977

16 0.011370058474575992

12 0.01092587220639276

The full output files for each run are also provided as deliverables.