## Lab 2 - PageRank on the Wikipedia Corpus

**Assigned -** 8/16/07
**Due -** 23:59, 8/24/07
**Turn in -** Your code and your write up.
**Partner** - You may use one if you'd like

**The Goals:** Implement PageRank, turn Wikipedia into a giant graph, run PageRank on said graph, run it several more times (ideally until the values converge), return (in a humanly parsable sort of way) the PageRank of all the articles.

**The Algorithm:** http://infolab.stanford.edu/~backrub/google.html (section 2.1.1)

**The Corpus:**

- Wikipedia offers a compressed XML file for download that contains every article on the site, including redirect and disambiguation pages.
- We've reformatted the data set into 390 30MB files. Each 30MB file contains several thousand Wikipedia articles, formatted one per line.
- The characters '<' and '>' have been replaced by '&lt' and '&gt' because the input format we recommend strips XML tags.
- That input format is org.apache.hadoop.mapred.TextInputFormat.class
- The title of each article is marked by "<title>Name of the article</title>" in which the brackets have been reformatted (as mentioned above) so the title is now marked by "&lttitle&gtName of the article&lt/title&gt".
- Links to other wikipedia articles are of the form "[[Name of other article]]".
- Find the data files on dfs at /user/root/lab2

**A Possible Outline** (each of these is a MapReduce)**:**

1. graphBuilder - Takes the Wikipedia data and constructs a link graph in a SequenceFileFormat file.
2. pageRankIter - Iteratively updates the page rank value of a graph in sequenceFileFormat
3. pageRankViewer - Converts sequenceFileFormat to text format and sorts nodes by their page rank score. It's helpful here to negate the pagerank scores so that sites with the highest values appear first.

**Complications:**

This isn't super efficient (sorting all of the Wikipedia articles in the reduce step of PageRank is slow!) so three things:

- Don't save this for last minute.

• Set the number of reduce tasks (five is good, more is probably better).

• Get extra use out of your PageRank reduce method by also setting it as your combiner class.

**Some super sweet and totally-recommended-although-not-explicitly-required extension possibilities:**

• We used the TextInputFormat class because we didn't have the time to figure out how to MapReduce over an XML file. Figure it out, write it up, let us know. It's probably trivial.

• Re work your graphBuilder so that it works with the Wikipedia HTML input and then run pagerank on some other languages. Make a couple sweeping conclusions about cultural differences evidenced by differences in highly ranked pages and include these in your write up.

• Link Hack Wikipedia. Use your data to subtly edit important Wikipedia pages to increase the pagerank of a pet cause. Tell us what you did and why it should work.

• Implement variably weighted links. Explain your heuristic and convince us that its working.

• Run PageRank over the Nutch crawl or your InvertedIndexer over Wikipedia and then combine both data sets into a rudimentary search engine.

**Write-up:**

1. How long do you think this project will take you to finish?
2. How much time did you actually spend on the project?
3. Acknowledge any assistance you received from anyone except assigned course readings and the course staff.
4. Explain why it's ok to use a combiner class with the PageRank algorithm.
5. What are the ten pages with highest rank in the provided Wikipedia corpus?
6. Describe any extensions you've implemented.