Eli Sorey

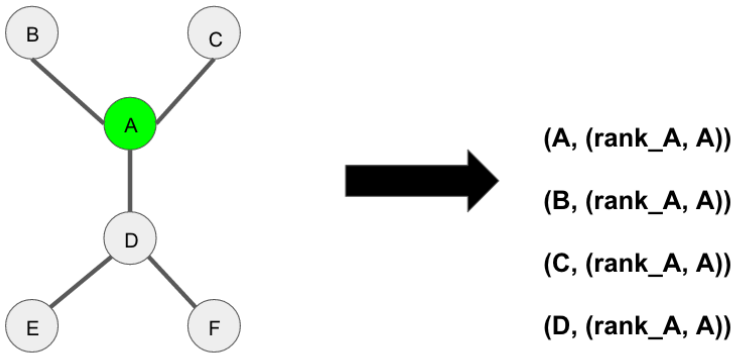
CS 11 Hadoop

5/11/2016

Homework Four

My goal with this project was to implement the topic organization algorithm as outlined in the assignment description. However, due to unforeseen issues and work from other classes piling up, I was unable to get a working implementation. Nevertheless, I feel that my design for this project could be useful for someone looking to implement such an algorithm within the Hadoop framework.

The general workflow of my design is as follows. I wrote a Hadoop job called RemoveTopPages that flags all pages with ranks higher than all of their neighbors' ranks. The idea behind this job's mapper is summarized in the graphic below:



In words, each node emits a key-value pair for each of its neighbors, including its name and its page rank. Thus, the reducer has access to all of the ranks of a page's neighbors, along with the rank of the page. It then determines if the page has the

highest rank amongst all of its neighbors. If so, the reducer marks this with a control character (@ in my implementation) and the current page removal round, and does not pass on the rest of the information for that node (its links, etc). This effectively removes the node from the graph for future iterations of the page rank calculation job.

I was unsure how to determine when only leaf pages remained in the graph, so I opted to just run the rank calculation -> top page removal cycle a fixed number of times. This would result in a partial information tree with only the most highly ranked pages included. The final results after this process would be a file with entries beginning with the @ control character for pages that had been removed, and other entries for pages that weren't removed during the process. From here, I planned to isolate the removed pages and use a scripting language such as Python to build an adjacency list for the information tree. This would consist of matching each node in a given removal round (eg, all nodes removed during the third RemoveTopPages job) to the node in the previous round with the closest page rank. Finally, Gephi (https://gephi.org/) could be used to visualize the resulting information tree.