AMS 598 - Project2 Report

Weiwei Tao

October 13, 2021

1 Problem Description

In this project, we were given a csv file with size of 8.2 GB, which contains links information of about 10 million webpages. Each row in the file has two webpage numbers, indicating that we can direct to the second webpage through the first webpage. The goal of this problem is to find out the top 10 webpages with highest pagerank values.

2 Page Rank Algorithm

PageRank is a recursive algorithm which assign a score to each webpage where the higher the score is, the more important the webpage is. The algorithm is motivated by the ideas that users of webs tend to place links to pages they think are good or useful pages and users are likely to visit useful pages than useless pages. Thus, the importance of a page depends on the number of pages pointing to it and link's weight is proportional to the importance of its source page.

PageRank algorithm can be implement as following:

1. initialize each pages' importance as $r_i = 1/N$ where N is the total number of webpages.
2. Successively update each page's rank according to $r_j = \beta \sum_{i \to j} \frac{r_i}{d_i} + (1-\beta)/N$, where d_i is the number of pages that i links to and $\beta = 0.9$ (taxation method) in our case to deal with the issue of dead end and spider trap.
3. Repeat the second step until the page ranks stabilize.

To implement this method using MapReduce, we need to implement two stages:

1. Stage1 - Map: takes (URL_1, URL_2) pairs and maps them to $(URL_1, (PR_{init}, [URLs]))$.
2. Stage1 - Combine list of URLs with same URL_1.
3. Stage 2- Map: takes $(URL_1, (PR_{init}, [URLs]))$ and emits $(URL_2, \frac{PR(URL_1)}{length([URLs])})$ for each URL_2 in the list of URLs.
4. Stage 2- Reduce: update PR by summing up $\frac{PR(URL_1)}{length([URLs])}$ by URL_2.

Repeat stage 2 until the PR converges.

3 Implementation in Python

My code contains 3 parts:

1. Ppliting the original file into 20 files.
2. 20 mappers to maps URL pairs to URL-list pairs (i.e. analysis-xaa.py).

3. 1 Reducer to aggregate URL list by URLs (reduce1.py).

4. 1 Mapper Reducer to update Page Ranks for 5 loops (mapper2.py).

The results shown that the top 10 most important URLs are listed in Figure 1 as:

Index	pr
6598584	1.70377e-07
8459203	1.68105e-07
3682269	1.35769e-07
8899394	1.15267e-08
6744001	1.1513e-08
8377336	1.14934e-08
6761964	1.14901e-08
4430670	1.14892e-08
2425577	1.14735e-08
1686277	1.14716e-08

Figure 1: Top 10 pages and their page ranks