AMS 598 Project 1 Report

Jiecheng Song
October 2021

1 Introduction

The file "/gpfs/projects/AMS598/Projects/project2/project2_data.csv" contains the link information for 10M webpages.Each row in the file has two webpage numbers, indicating that the first webpage contains a link to the second webpage.

Use the taxation method (beta = 0.9) to calculate and report the top 10 webpages with the largest pagerank values. You need to perform the map-reduce steps on SeaWulf but only need to run five (5) iterations.

2 Methods

To conduct the taxation method, we separate the tasks in to two steps (phase 1 and phase 2). And due to the data set is large (about 8.5G), we need separate data into a few small files and use map reduce method.

2.1 Separate data

Use following latex commands separate the project2_data.csv file into a few file, each small file will contain 6M records, and there will be total 10 files named by project2_data_part_(00-09)

```
split -d -l 60000000 project2_data.csv project2_data_part_
```

2.2 Phase 1

In this step, we used the project data to generate the key-value pair (url, (pr_init, url_list)) the pr_init is the initial PageRank, and url_list is the url_list linked by url. To make it easier, I used two files to store the information. One is a .csv file generate by 'equal_weight.R' (I used equal weight as the initial PageRank), which denoted the initial PageRank of each url. Another is the .json file, which denoted the information of pair (url, url_list).

Because R doesn't support hast table well, so I chose to use list data structure to record the information, and the index of the list, will be the key, for example the first element in the list is the url_list contained by url_1. Also, here I used a map-reduce method, fist, use mappers to generate the pairs in each small file, then use reducers to combine the same indexes url_list together. (Each mapper 'phase1_(0-9).R' take one small files (10 mappers) and each reducer 'url_list_(0-9).R' take 1M url indexes (10 reducers)).

2.3 Phase 2

In this step, first we used mapper to take the (url, (PR, url_list)) pairs and for each u in the url_list generate the new pairs (u, PR/length(url_list)), here I still used list to contain the information (indexes of list is the keys) and used 10 mappers, one mapper for each file generated by parsing 1. ('mappers_(0-9).R')

Then used 10 reducer to calculated the new PageRank, each reducer will calculate the new PageRank for 1M url with following formula:

$$PR_{new} = (1-\beta) * 1/1e7 + \beta * \sum vals_i$$

Here, β is 0.9 and vals is $PR_i/length(url\_list_i)$ generated by mappers. ('reducer_(0-9).R')

And update the PR into the 'current_weight.csv' file by 'update_weight.R'. Then repeat the Phase2 5 times to do 5 iterations and get the final results, then select 10 url with highest PageRank by 'top10.R'.

3 Results

The top 10 urls with highest PageRank are as follow:

url	PageRank	url	PageRank
1278445	3.218338e-05	7716978	2.820331e-05
5043239	3.105838e-05	4751585	2.703763e-05
69917	2.986444e-05	3891296	2.361961e-05
8937112	2.95462e-05	2088650	2.225116e-05
6971730	2.91754e-05	9933087	2.042497e-05

Note: For each part of code, I submit separately one example file (eqaul_weight.R, phase1_0.R, url_list_0.R, mappers_0.R, reducer_0.R,update_weight.R, top10.R and corresponding slurm files) on blackboard and all the other code was in a .zip file.