INFSCI 2711 Homework3

Hao Wang(haw69) Dan Sun(das225)

GitHub Link: https//github.com/danielsun510/Infsci2711_Homework-3

Q1.

Write a Hadoop MapReduce program, which outputs the number of words that start with each letter. This means that for every letter we want to count the total number of words that start with that letter. In your implementation ignore the letter case, i.e., consider all words as lower case. You can ignore all non-alphabetic characters.

Goal:

Count the total number of words that start with specific alphabetic.

Input File:

pg100.txt

Output File:

output/part-r-00000

Result:

a 85667

b 45597

c 34694

d 29830

e 19116

f 36930

g 21021

h 60736

i 62305

j 3354

k 9491

l 29697

m 55800

n 26834

o 43590

p 27860

q 2378

r 14396

s 66069

t 126395

u 9185

v 5745

w59844

x 14

y 25888

z 79

How to Run:

1.Down Load Project from GitHub.

2.Import the Project into Eclipse

3.Choose the Project and Select **Run As → Run Configuration**

4.Input the Name field,Project name and Path in the Main class field.

5. Switch to the Arguments tab and input{ pg100.txt output} in the Program arguments field.

6.Click Run

Q2.

Write a MapReduce program in Hadoop that implements a simple "People You Might Know" social network friendship recommendation algorithm. The key idea is that if two people have a lot of mutual friends, then the system should recommend that they connect with each other.

Goal:

The best friendship recommendations often come from friends. The key idea is that if two people have a lot of mutual friends, but they are not friends, then the system should recommend them to be connected to each other.

Description of how to use MapReduce :

1.Map phase

1.Emit <fromUser, r=toUser; flag=-1> for all toUser.

If two people are already mutual friends , we can emitted key, and r, and set flag as -1.

If two people are not friends, we can record the ID of these two people and set flag 1 to present they are not friends so far.

2.Emit <toUser1, r=toUser2; flag=fromUser> for all the possible combination of toUser1, and toUser2 from toUser, and they have mutual friend, fromUser. It will emit n(n - 1) records.

3.Totally, the number of friends <USER> is n and there will be n^2 records after Map phase.

2.Reduce phase

1. Summing how many mutual friends they have between the key, and r in the value. If any of them has mutual friend set flag as -1.If two people are already friends, there is no recommendation friends between these two people.

2.Sort the result based on the total number of mutual friends.And the result will be descend order according to the number

Input File:

friends.txt(soc-LiveJournal1Adj.txt)

Output File:

outputfriend/part-r-00000

Result:

924: 2409,11860,15416,43748,45881

8941: 8943,8944,8940

8942: 8939,8940,8943,8944

9019: 9022,9023

9020: 9021,9016,9017,9022,9023

9021: 9020,9016,9017,9022,9023

9022: 9019,9020,9021,9016,9017,9023

9990: 13134,13478,13877,34299,34485,34642,37941

9992: 9989,35667,9991

9993: 9991,13134,13478,13877,34299,34485,34642,37941

How to Run:

1.Down Load Project from GitHub.

2.Import the Project into Eclipse

3.Choose the Project and Select **Run As → Run Configuration**

4.Input the Name field,Project name and Path in the Main class field.

5. Switch to the Arguments tab and input{ friends.txt outputfriend} in the Program arguments field.

6.Click Run