Assignment 6

Map-Reduce

2nd March 2022

This assignment is on map-reduce, which is a distributed and scalable way of extracting/mining required information from multiple datasets stored on multiple servers. Follow the tutorial to understand how you can design mapper and reducer for specific queries/operations.

Tutorial on map-reduce

You can start with a simple word count problem. Say, we have a text file and we want to count the frequency of occurrence of each word. The tutorial below explains how to solve this problem using a map-reduce algorithm.

Tutorial References :
http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/

Next you can also look into the following tutorial for slightly harder query (tf-idf scores)

https://www.tutorialspoint.com/map_reduce/map_reduce_tutorial.pdf

Tasks

1. Study HDFS and MapReduce basics.
2. Write the necessary mapper and reducer routines in python to implement the queries mentioned in later sections.
3. Ensure that the reducer routine prints results of queries to a text file (as asked in the later sections).

Dataset

We will be using a dataset of an email network of a R&D company. Please download the data from moodle.

There are 2 files in the input data- (i) network.txt (ii) dept_labels.txt

In **(i) network.txt**, each line represents a **directed edge** between two persons (i.e., employees) indicated by two Node_IDs separated by space. There is an edge (u, v) in the network if person u sent person v at least one email.

==Sample data from the file=
0 1
2 3
2 4
5 6
5 7
8 9
10 11
12 13
12 14
15 16
17 18
12 19
20 21
20 22
23 24
==========

The **(ii) dept_labels.txt** file contains department membership assignment of the employees, i.e., nodes. Each individual belongs to exactly one of 42 departments at the company. Each line in the file represents the following-
Node_ID Department_ID

- Node_ID: id of the node (an employee of the company)
- Department_ID: id of the employee's department (number in 0, 1, ..., 41)

==Sample data===
0 1
1 1
2 21
3 21
4 21
5 25
6 25
7 14

8 14
9 14
10 9
11 14
12 14
13 26
=========

Queries

The queries are to be implemented in the mapper and reducer phases. Some of them may give empty results. You need to implement the following queries in this assignment. The queries have an imaginary backstory to help you find a real world perspective in this assignment.

Assume you are the HR manager of the company. You want to conduct a quantitative analysis of the communication between the employees of your company using the email network. The network is generated using email communication data of your company. The information about all incoming and outgoing email between employees of the company is anonymized. By incoming email, you can assume the emails that are sent to a typical employee and vice-versa as outgoing email. The email links in the network represent communication between company employees only, and the dataset does not contain incoming messages from or outgoing messages to the rest of the world.

1. For each employee, you want to compute the number of distinct employees to whom emails have been sent by the respective person. Write mapper and reducer routines for this.
   Note: The result will have lines such as: 10 30 which means that employee id 10 has sent emails to 30 different employees. The output file should contain one such count in each line. [20]

2. You want to find out the top-10 employees who have the email communication with the highest number of employees. By email communication, you are considering the outgoing emails only. Write mapper and reducer routines for this.
   Note: The result will contain the employee-ids with the highest number of outgoing emails, ordering within these top 10 node-ids does not matter. The output file should contain one node-id in each line. You are allowed to use the output of query 1. [20]

3. Being a responsible HR, you want to find out the number of employees who have sent one or more emails to these top-10 employees (inclusive of these top 10 employees). Write mapper and reducer routines for this.
   Note: The result will have lines such as: 7 15 which means that there are 15 employees each who have sent emails to the employee having id 7. The output file should contain such 10 lines, ordering across the first employee id doesn't matter. [25]

4. Besides the analysis at individual employee level, you want to extend this experiment at department level also. You have decided to reward the most influential department that has set up maximum email communication (i.e., outgoing emails) to other departments. The emails that are sent to employees outside of a specific department can be assumed as outgoing emails. You are aware of the department allocation of the employees. The sum of outgoing emails of individual employees assigned to a department will be assumed as the number of outgoing emails sent by the department collectively. Write mapper and reducer routines to find out the most influential department of your company.
   Note: The output file should contain the department id and the number of outgoing emails sent by the department printed on a line and separated by space. [25]

How to run and test your code:

Since we do not have access to a Hadoop cluster, we will be testing our codes on a Linux system as follows:

```
cat network.txt | python mapper.py | sort | python reducer.py  > result.txt
```

Or just `python mapper.py | sort | python reducer.py > result.txt`

Explanation on the above commands:

1. "cat" is a linux command to print the contents of a file on the console.
2. The pipe operator (|) directs the previous command's output to the next command.
3. "sort" is a linux command to sort the input lexicographically.
4. ">" can be used to save the standard output in a file.

Deliverables: 4 sets of python codes (mapper and reducer), 4 Makefiles, 4 result.txt, 1 readme

Evaluation Scheme: Results: 90 marks, Coding Style: 10 marks

Important Instructions

1. **The mapper routine can pass over the network.txt / dept_lables.txt file only once. You can not use any intermediate file structure or data structure to save data. Python dictionaries should not be used in the mapper routine. However, you can use one dimensional arrays in the mapper or reducer, and dictionaries in the reducer only. Note that these must be limited to one dimension only. The reducer can't access the data file. So, the mapper should just go through the data and generate a series of key-value pairs which will then be passed to the suitable sort command. Finally the reducer should just go through the stream of tokens once to give the required output. Any violation will attract a penalty of upto 100% of the marks assigned.**
2. **Submission Rule:** Python code for above functionalities must be compressed as **.tar.gz** (gzip) and named "**A6_<RollNo>.tar.gz"**. For each query, make a directory named "**Query<no.>"**. Your files must be inside the respective directories. Strictly adhere to this naming convention. Submissions not following the above guidelines will attract penalties.
3. **Makefile:** Each query folder **must** have it's own Makefile to execute the routines. You can find a relevant tutorial here (https://opensource.com/article/18/8/what-how-makefile).
   Note: "sort" behaves differently in different environments. Ensure that you use sort in a way which works properly on a linux machine.
4. **Code error:** If your code doesn't run or gives error while running, you will be awarded with zero mark. Your code must run correctly on **a linux machine**.