Assignment 1. MapReduce

Abstract

The objective in this assignment is to develop practical skills in parallel data processing for computational and data science. The focus is on scaling data-intensive computations using functional parallel programming over distributed architectures. The goal of the exercises is to practice some of the most frequent MapReduce design patterns, which can make application design and development easier allowing problems to be solved in a reusable and general way.

Guidelines

- The **datasets** to do the exercises are available for download from Google Classroom.
- Both the mapper and the reducer should be Python executable scripts that read the input from *stdin* (line by line) and write the output to *stdout*. The output of the mapper must be lines with a key and a value, separated by a tab character.
- It is very important that you follow an efficient MapReduce design pattern.
- Upload the files specified in each exercise to the assignment on **Google Classroom**.
- Submit each file individually (**DO NOT submit a compressed file**).
- There are **no late days**.
- The grade on this assignment is **10% of the final grade**.

1. Distributed Grep (2 points)

Develop a distributed version of the `grep` tool to search for words in very large documents. Use the design patterns explained in class. The output should be the lines that contain a given *word*.

Use as input the text used in the word count example described in class (eBook of Moby Dick). Your scripts will be tested using the following command:

```
$ ./P1_mapper.py word < input.txt | sort -k 1,1 -t $'\t' | ./P1_reducer.py
```

2. Count URL Access Frequency (2 points)

Develop a MapReduce job to find the frequency of each URL in a web server log. Use the design patterns explained in class. The output should be the URLs and their frequency.

Use as input the sample Apache log file `access_log` (downloaded from http://www.monitorware.com/en/logsamples/apache.php). Your scripts will be tested using the following command:

```
$ ./P2_mapper.py < access_log | sort -k 1,1 -t $'\t' | ./P2_reducer.py
```

3. Stock Summary (2 points)

Write a MapReduce job to calculate the average daily stock price at close of Alphabet Inc. (GOOG) per year. Use the design patterns explained in class. The output should be the year and the average price.

Use as input the daily historical data GOOGLE.csv (downloaded from Yahoo Finance https://finance.yahoo.com/quote/GOOG/history?ltr=1). Preprocess the GOOGLE.csv file to remove the header. Your scripts will be tested using the following command:

```
$ ./P3_mapper.py < GOOGLE.csv | sort -k 1,1 -t $'\t' | ./P3_reducer.py
```

4. Movie Rating Data (2 points)

GroupLens Research has collected and made available rating data sets from the MovieLens web site (http://movielens.org). The data sets were collected over various periods of time, depending on the size of the set. Before using these data sets, please review their README files for the usage licenses and other details. A small version of the dataset, ml-latest-small.zip (downloaded from https://grouplens.org/datasets/movielens/), can be used.

Develop a MapReduce job to show movie ids with an average rating in the ranges:

Range 1: 1 or lower
Range 2: 2 or lower (but higher than 1)
Range 3: 3 or lower (but higher than 2)
Range 4: 4 or lower (but higher than 3)
Range 5: 5 or lower (but higher than 4)

Use the design patterns explained in class. The job should have two MapReduce phases. The output of the first phase should be the movie ids and their average rating. The output of the second phase should be the ranges and the movie ids.

Your scripts will be tested using the following command:

```
$ ./P4a_mapper.py < ratings.csv | sort -k 1,1 -t $'\t' | ./P4a_reducer.py |
./P4b_mapper.py | sort -k 1,1 -t $'\t' | ./P1b_reducer.py
```

5. Meteorite Landing (2 points)

NASA's Open Data Portal hosts a comprehensive data set from The Meteoritical Society that contains information on all of the known meteorite landings. Table Meteorite_Landings.csv (downloaded from https://data.nasa.gov/Space-Science/Meteorite-Landings/gh4g-9sfh) consists of 34,513 meteorites and includes fields like the type of meteorite, the mass and the year. Write a MapReduce job to calculate the average mass per type of meteorite. Use the design patterns explained in class.

Preprocess the Meteorite_Landings.csv file to remove the header and do some data cleansing work. Your scripts will be tested using the following command:

```
$ ./P5_mapper.py < Meteorite_Landings.csv | sort -k 1,1 -t $'\t' | ./P5_reducer.py
```

Submission

- P1_mapper.py and P1_reducer.py
- P2_mapper.py and P2_reducer.py
- P3_mapper.py and P3_reducer.py
- P4a_mapper.py, P4a_reducer.py, P4b_mapper.py and P4b_reducer.py
- P5_mapper.py and P5_reducer.py