Alexandria University
Faculty of Engineering
Computer and Systems Engineering Dept.
Fourth Year
Spring 2016

CS432: Distributed Systems and Net-Centric Computing
Assigned: Sunday April 10th, 2016
Due: Sunday April 17th, 2016

Introduction to Apache Hadoop.

Calculating Movie Ratings.

Objectives

This assignment aims to make you practice the MapReduce framework we studied in theory and see how it distributes work among the mappers and reducers.

Introduction

The objective of this lab is to gain familiarity with the MapReduce distributed programming framework. For this lab, you will be responsible for the creation of a Hadoop job that is able to aggregate and summarize movie ratings from a movie rating database of approximately 10 million ratings. There are two distinct phases for this lab, two MapReduce iterations and a reporting phase. Phase 1 averages ratings by movie. Phase 2 will bin average movie ratings from Phase 1 to determine the distribution of ratings in the dataset.

Phase 1: Rating Aggregation

The goal of Phase 1 will be to aggregate movie reviews for each individual movie. For example, if a movie in the ratings database contains ten ratings, the result of this MapReduce iteration will be a single averaged rating for this movie. This job will read from a rating file ***rating.dat***, and the output of this phase is to be used in the next phase.
The file will be split by lines and sent to the Map method, which will then parse the lines and emit <movieID, rating> pairs. The userID and timestamp fields can be discarded. The Reduce function will take these pairs and compute an average rating for each movieID. Results from Phase 1 (i.e.<movieID, average-rating> pairs) should be stored in a file called ***average-ratings.dat***. The file should contain one <movieID, average-rating> pair per line.

Phase 2: Rating Summarization

The goal of Phase 2 is to bin the average movie ratings that were output from Phase 1. A bin is just a range of values with a count that represents the total number of movies in the bin. A movie belongs to a bin if its average rating from Phase 1 falls into the bin’s range. Because the bin ranges are mutually exclusive, each movie will belong to exactly one bin. There will be ten bins total, and there ranges will be as follows:

- bin 1: [0, .5)
- bin 2: [.5, 1)
- bin 3: [1, 1.5)
- bin 4: [1.5, 2)
- bin 5: [2, 2.5)
- bin 6: [2.5, 3)
- bin 7: [3, 3.5)
- bin 8: [3.5, 4)
- bin 9: [4, 4.5)
- bin 10: [4.5, 5]

Phase 2 will input the text file ***average-ratings.dat*** that was the result of the calculations from Phase1. The Map <movieID, average-rating> method will parse these pairs and determine the appropriate bin ID based on the average movie rating. The final result of Phase 2 will be a count for each bin representing the number of movies mapped to that bin. Write these results to a file called ***ratings-summary.dat***. The format should be <binID, count of movies>.

Prerequisites

You need to install Apache Hadoop on your machine. A good tutorial to help you to set it up, can be find here,
http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-single-node-cluster/

Dataset

We need to use the 10M movie rating dataset from UMNgrouplens group (http://grouplens.org/datasets/movielens/). All ratings are contained in the file ***ratings.dat***. Each line of this file represents one rating of one movie by one user, and has the following format:

UserID::MovieID::Rating::Timestamp

Where userID and movieID fields are integer values that uniquely identify user and movies respectively. Ratings are made on a 5-star scale, with half-star increments. Timestamps represent seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970.

Hadoop Specific

For this project, you should use the org.apache.hadoop.mapreduce package, not the older once deprecated org.apache.hadoop.mapred package. Because we are dealing with text input for both Map methods, you will use the org.apache.hadoop.mapreduce.lib.input.TextInputFormat class as the input type. This will parse input file by line rather than by chunk size. The keys will be the byte offset of the start of the line, and the value will be the line itself. The byte offset key will not be needed by either of the Map methods.

Notes

- This assignment is based on assignment from Fall 2013 CIS 570 course in Brown University.
- Again and Again, performance evaluation is a must, the more figures and execution times the better, try to find a pattern, a perfect configuration, etc. .
- ***BONUS:*** Using multiple machines as a cluster: for extra help use the following link http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-multi-node-cluster/ . Contact the TA for Lab hours.
- ***BONUS:*** Using MS Azure account to run the map reduce job, ask for your id and pass; you have to make sure it works on your machine first since they come with only limited time and storage (what is worth around a 100$ a month) and you will need it next assignment.
- The Assignment is straightforward; hence, the reports are expected to mirror your experience with the Hadoop file system and mapReduce framework.
- You should deliver a report, please follow the template.

Grading Policies

- No Late submission is allowed.
- You should work in groups of 3 students.
- Plagiarizing is not acceptable. Sharing code fragments between groups is prohibited and all the groups that are engaged in this action will be severely penalized. Not delivering the assignment will be much better than committing this offence.