CS6675/ CS4675

Homework Assignment 3
(Programming Category)

Student Name:______________________

Student Session: cs6675 or CS4675 (circle one)

You are given the choice of 3 types of programming problems in the third homework assignment.

- The first problem is Hand-on Experience with Hadoop or Spark MapReduce
- The last two are Hand-on experience with Hadoop MapReduce or Spark Configuration.
    - Problem 2: Learning Optimal Configuration of HDFS / Hadoop MapReduce (2 options)
    - Problem 3: Learning Configuring SPARK Jobs (4 options)

You only need to choose one of the problems as your first homework. If a problem has multiple options, you only need to choose one option. Feel free to choose any of your favorite programming language Java, C, Perl, Python.

Due Date: Midnight on Friday of Feb 21 with graceful (no penalty) extension till 9am on Feb 22 (Sat). **No late submission will be accepted.**

Problem 1. Hand-on Experience with a Distributed Computation Algorithm

This problem is suitable for those who are familiar with some data mining and machine learning algorithms and wish to get a hand on experience with distributed version of such algorithms.

Example algorithms include PageRank, K-means, Logic Regression, SVM, Collaborative Filtering.

You are encouraged to use open source ML library, such as Scikit Learn, R library, Spark ML library, Hadoop Mahout.

You are required to choose your favorite algorithm in a distributed version and also create a distributed computation environment (multi-agent on a single node or multi-container/VM on a single host are acceptable). Then find a dataset that is larger than the allocated memory of a single executor. This will motivate you to run the distributed version of the algorithm on multiple executors concurrently.

Deliverable.

(a) Describe your distributed version of the algorithm and the URL if you downloaded from public domain.
(b) Describe your dataset and show some actual statistics about the dataset you choose to work with, such as size in GB and how do you provide the distributed partitions of this dataset and the partition statistics.
(c) Describe the design of the distributed computation algorithm, such as the design options and why the one design option in the algorithm you promote is good.
(d) Conduct measurement study of your execution for both single node, 2 nodes and 5 nodes to provide a comparison in performance.
(e) Show some screen shots of your distributed execution experience if any,
(f) Elaborate on your experience, experimental results and problems encountered. How you solve the problems if you did, including the analysis of your results and your learning experience.

Problem 2.
Learning Optimal Configuration of HDFS / Hadoop MapReduce.

This problem contains two options and suitable for students who have some experience with Hadoop MapReduce Platform already. For both options, you need to do the following 3 steps first:
(1) Install HDFS and Hadoop MapReduce on your laptop, and select two example map-reduce programs/applications provided with the package, such as word count, sort, grep.
(2) use the default configuration for both HDFS and Hadoop MapReduce and measure and report the runtime performance for each example program using two different sizes of datasets (you can triple the given dataset to generate a larger one). Note the package comes with both code and datasets.
(3) You may use excel file to generate your runtime statistics plot or organize the performance measurement data in a tabular format.

Option 1: Chunk Size optimization
Go through the configuration file to set the map input chunk sizes to small or larger than the default size such that you can have varying map input file size (say 3~5 different chunk sizes). Now measure the performance of the two MapReduce Program and explain the difference observed. Hint: you need to try to select chunk sizes that have larger differences such as 64KB, 128KB, 256KB, 512KB, 1GB, 2GB. You are asked to analyze your experimental comparison results and provide your intuition and discussion to elaborate what you observe and why.

Deliverable.
(a) URL to the MapReduce codes and the datasets used
(b) screen shots of your execution process.
(c) Runtime statistics in excel plots or tabular format.
(d) Your analysis.

Option 2: JVM Queue Size Optimization
Go through the configuration file and adjust JVM queue size for Map slot and Reduce slot and run the same two MapReduce programs using the same #mappers and the same #reducers. Compare with the default setting of JVM queue size.
You are asked to analyze your experimental comparison results and provide your intuition and discussion to elaborate what you observe and why.

Deliverable.
(a) URL to the MapReduce codes and the datasets used
(b) screen shots of your execution process.
(c) Runtime statistics in excel plots or tabular format.
(d) Your analysis.

Problem 3. Learning Configuring SPARK Jobs

This problem contains multiple options and suitable for students who have some experience with Spark Platform already. For all options, you need to do the following 3 steps first:

1. Download Spark on your laptop and run one example program of your choice on two example datasets provided in the Spark package.
2. Report the runtime performance for your chosen example program using two different sizes of datasets.
3. You may use excel file to generate your runtime statistics plot or organize the performance measurement data in a tabular format.

Option 1: Configuration Impact

Choose one type of application workloads, and run the application and report the runtime performance for your chosen application using two different configuration settings. Example applications can be a MapReduce program such as sort, or a machine learning package such as Logic Regression, kNN, or K-means clustering.

You may use excel file to generate your runtime statistics plot or organize the performance measurement data in a tabular format. You are expected to discuss and elaborate your comparison results.

Deliverable.
(a) Document your installation of Spark
(b) Screen shots of your execution process.
(c) Runtime statistics collected if any for the two configuration settings, you can use excel to plot figures, bar chart, and tables.
(d) Discuss your comparison results and experience.

Option 2: Chunk Size optimization
Go through the configuration file to vary the input data sizes from small to larger than the default size such that you can measure how different input size of data may impact on Spark performance. You can run MapReduce on Spark and use two MapReduce programs such as wordcount, grep as your test applications.
Then measure the performance of the two Spark Programs and explain the difference observed.
Hint: you need to try to select chunk sizes that have larger differences.

You are asked to analyze your experimental comparison results and provide your intuition and discussion to elaborate what you observe and why.

Deliverable.
(a) URL to the Spark code and the datasets used
(b) screen shots of your execution process.
(c) Runtime statistics in excel plots or tabular format.
(d) Your analysis.

Option 3: Configuration Optimization
Go through the configuration file of Spark and the program such as MapReduce you plan to run on Spark, and identify at least one configuration parameter that you want to reset by modifying the default value, such as JVM queue size.

Compare with the default setting with the new settings you have used. Ideally you should consider 3-5 different settings compared to the default to gain a better understanding on how to optimize Spark configuration.

You are asked to analyze your experimental comparison results and provide your intuition and discussion to elaborate what you observe and why.

Deliverable.
(a) URL to the Spark code and the datasets used
(b) Screen shots of your execution process.
(c) Runtime statistics in excel plots or tabular format.
(d) Your analysis.

Option 4. Learning Optimal Configuration
This option is designed for students who are familiar with both Hadoop MapReduce and Spark and interested in hand-on comparison of them through example programming problems or big datasets and / or through configuration tuning.

Compare Hadoop MapReduce and SPARK using a common analytic problem: a simple one like sort or word count, or a machine learning package such as logic regression or K-means Clustering or k nearest neighbor search. You may also write your own Spark program.

Deliverable.

(a) URL to the HDFS/Spark code, the MapReduce code and the datasets used
(b) screen shots of your execution process.
(c) Runtime statistics in excel plots or tabular format.
(d) Your analysis.