PolyU COMP4434 Assignment 1

# Introduction

This assignment will help you understand Hadoop MapReduce better through a simple exercise. Our goal is to build an equal-width histogram for a dataset, based on the parameters such as required ranges and the number of partitions.

# Background

As we often study the distribution of values in a dataset, building histogram is basic and it is one of simple methods to help us understand dataset features.

For example, given a dataset of students scores, if we are required to analyze their performance under the range and 8 partitions, below is the histogram (in chart format) [[1]](http://www.cqeacademy.com/cqe-body-of-knowledge/continuous-improvement/quality-control-tools/histograms/). The range is divided into 8 small partitions, and the program should return the number of students in each score range.

# To Begin with

* In this “building histogram” program, you do NOT need to make charts like the picture above. You ONLY need to make the program output the count in each partition.
* Apart from data input and results output, we set **3 parameters** in this program:
  + : Integer. The lower bound of the range when we do statistics on the datasets. In the example of Background section, ‘20’ is set as ‘***’*** parameter, because the students scores are studied under the range.
  + : Integer. The upper bound of the range when we do statistics on the datasets. In the example of Background section, ‘100’ is set as ‘’ parameter, because the students scores are studied under range.
  + : Integer. The number of partitions when we build histogram. Because we make equal-width histogram, which means the width of each partition is the same, we suppose . In the example of Background section, ‘8’ is set as ‘’ parameter, also it satisfies .

# You need to do some simple coding work

After unzip our materials, you will see two folders:

1. Codes

We provide two java files and you need to complete relevant codes:

* PartitionCounting.java:
  + In this java file, you need to complete Map and Reduce methods, which are intended to Map the values to appropriate partition, and count the number of values in each partition via Reduce.
* BuildHistogram.java:
  + In this java file, you need to complete some configurations in main method to make sure that the program works well, in other words, the exported jar works well on Hadoop.
* Please remember NOT to modify any **given** parts of the java files, otherwise your program might not work under our assessment.

1. Data

We provide 3 data files, named DataSet1 DataSet2 DataSet3. These three datasets share same format that each of them includes 100 integer number and one integer number per line.

You could use this bunch of datasets to test your results.

1. Output

We provide a sample output file (get from HDFS to local disk) in this folder for you to understand the output format better.

# Input and output format

* The input files consist of integers that **one integer per line**, as you can observe from the files in Data folder.
* The output files consist of the number of values in each required partition, which is a pair of range (key) and its relevant count (value), of which the range of the partition(key) is presented as the lower bound and upper bound that **separated by comma**. Output key-value is **separated by tab**.
  + The example chart in Background section could be represented in our output format as:

20,30 1

30,40 2

40,50 1

50,60 2

60,70 6

70,80 9

80,90 13

90,100 5

# Run the program

As you have learned from our lab tutorials, export ‘jar’ file (e.g., HistogramCalculating.jar) of your program then run it with parameters. We give the command format:

hadoop jar HistogramCalculating.jar min\_value max\_value num\_partitions input\_path output\_path

Besides the first 3 parameters that are introduced in previous section:

* : input datasets path.
* : output results path.

In the example of Background section, we could set the command like this:

hadoop jar HistogramCalculating.jar 20 100 8 students/input/scores students/output

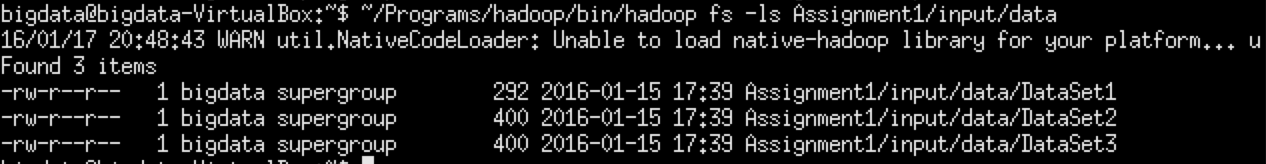
where the students score datasets are in the directory students/input/scores on HDFS, and the results will be in students/output.

# Test your program

You can test your program by watching the output, as three datasets we provide in Data folder are tricky.

DataSet1 contains 1 2 … 99 100, DataSet2 contains 101 102 … 199 200, DataSet3 contains 201 202 … 299 300. For example,

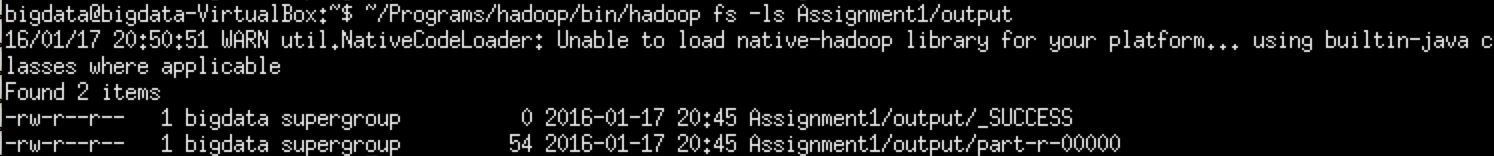
* We put the datasets on HDFS on the path below:

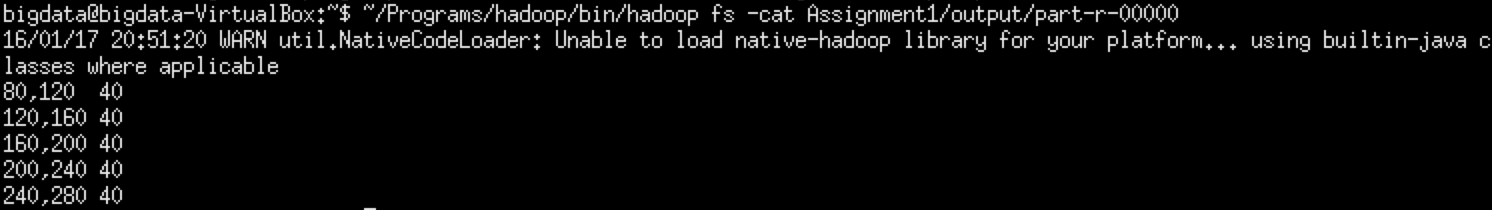


* Run the program:



* Check the output:





This ‘part-r-00000’ is exact the sample results file that we provide in Output folder.

# What to submit

Make a zip file called Assignment1.zip which included two java files (PartitionCounting.java BuildHistogram.java), submit it through blackboard.

# Grading Policy:

We will export a jar file from your source file as jar file and execute it according to “Run the program” section.

We will use several different input files (such as DataSet4 … DataSet9) and set different parameters to test your program, say we use 5 test cases, and if your program only passed 4 of the 5, you get 80.

# DEADLINE: Feb 15 11:59am

# LATE PENALTY

Late X day: your score = raw score \* (100 – 20\* X)%

# Plagiarism

Detecting the similarity of your codes is easy, and cases will be strictly handled according to the University’s regulation, so please don’t risk doing that.

# Questions?

Feel free to ask TA during labs. Or contact [csrbai@comp.polyu.edu.hk](mailto:csrbai@comp.polyu.edu.hk) with email subject “COMP4434 Assignment1 Questions”.