VE572 —Methods and tools for big data

Assignment 4

Jing and Manuel — UM-JI (Summer 2018)

Reminders

- Write in a neat and legible handwriting or use LaTeX
- Clearly explain the reasoning process
- Write in a complete style (subject, verb, and object)
- Be critical on your results

Ex. 1 — *Processes and cgroups*

1. Write a short summary describing what `cgroups` are.
2. Explain the differences and similarities between `cgroups` and processes in Linux.

Ex. 2 — *MapReduce*

In this exercise we write a MapReduce program to solve the second exercise from lab 3.

1. Write a `Map` class which extends the MapReduce `Mapper` class, extracts, and outputs pairs composed of a student ID and a grade.
   Hint: read the file by line and tokenize each of them using `StringUtils`.
2. Write a `Reduce` class which extends the MapReduce `Reducer` class, outputs pairs composed of a student ID and its highest grade.
   Hint: use `Iterable<Text>` to iterate over all the values of a given key.
3. Write a driver function write set all the necessary properties to configure the MapReduce job.
   Hint: specify what classes are to be used by the Mapper and Reducer, as well as where the input and output files are located.
4. Run the MapReduce program and compare the running time to the streaming approach used in the lab. Draw a table showing the comparison for various file sizes.

Ex. 3 — *Avro*

1. Install Avro.
2. Define a schema to represent an entry in the grade file generated in the second exercise of lab 3.
3. Write two short programs to serialize and de-serialize the grade file.
4. Explain the three ways into which Avro can be used in MapReduce, and when to apply each of them. The three approaches are (i) mixed-mode, (ii) record-based, and (iii) key-value based.

Ex. 4 — *Bloom filters*

Sometimes it is appropriate to filter the data before running actions on it. For instance when referring to exercise 2 of lab 3 we might only want to retrieve the maximum grade of the students whose ID ends with a three. In that case one might want to use a preprocessing job to create a Bloom filter and use it to filter out records in the mapper.

1. Describe what a Bloom filter is and how it works.
2. Using the `BloomFilter` class write a mapper which creates a Bloom filter.
3. Using `Iterable<BloomFilter>` combine all the Bloom filters together in the reducer and output the result into a serialized Avro file.