BIG DATA – AX II

Andrei-Mihai Nicolae
2147392n

INSTRUCTIONS

In order to run the code, please follow the following instructions (also outlined in the instructions text file):

```
$ cd task1
$ export HADOOP_CLASSPATH=ax2_task1.jar
$ javac-run.sh $(find . -name '*.java')
$ jar -cvf ax2_task1.jar $(find . -name '*.class')
$ java-run.sh MainTaskOne {input_file} {output_file} {start_date} {end_date}
```

```
$ cd task2
$ export HADOOP_CLASSPATH=ax2_task2.jar
$ javac-run.sh $(find . -name '*.java')
$ jar -cvf ax2_task2.jar $(find . -name '*.class')
$ java-run.sh MainTaskTwo {input_file} {output_file} {timestamp}
```

DESIGN

As the assignment was split into 2 tasks, I divided the work into 2 different packages. Both of them have a Main class that sets everything up, among which:

- they both take the input and output files from the command line as first arguments
- they set up the reducer classes, the job name, output format etc.
- they initialize the table mapper job along with the mapper used, the class of the key and value as well as the job

Afterwards, in Task 1 we have a custom mapper and a custom reducer. The reducer is rather simply implemented, adding all revisions into an ArrayList from which, after it's sorted, we create a string containing all revision IDs in a sorted fashion. In the end, we write the key, the value's size and the actual value pair where the key is the articleID, the size is the number of different revisions we've found and the value is the list above-mentioned.

The Task 1 mapper first gets the 2 timestamps from the command line as the last 2 arguments. Then, it checks if the current timestamp is between the 2 provided by the user and, if so, we write the key value pair.

In Task 2, the Reducer has been developed such that it performs as optimally as possible. As it loops through the revisions we have, it checks