THE UNIVERSITY OF SYDNEY

School of Information Technologies

1st Homework
Thursday March, 9, 2017
MapReduce Simulator

Homework Task

Your first homework task until Thursday Week 2 (16 March) is to finish a small MapReduce simulation program that scans a given input file (data.csv) and calls two functions to filter and aggregate the data found in there.

The following instructions assume that you have logged in to a **Unix shell**, for example at one of the servers of the School of IT, though it also works on a local Linux or OSX machine, and that you have either **Python 3.x** or **Java 7** or higher installed. We provide two alternative versions of our skeleton code and leave the choice of which programming language you prefer to work in (Java or Python3) to you. For a local installation of the Java version of our code, you also will need the **ant** program for the building process.

1.1 Downloading and compiling the example code

Download and extract the source code and data as follows:

1.1.1 Python Version

```
wget http://web.it.usyd.edu.au/~comp5349/code/mrsim_python.tar.gz
tar xzvf mrsim_python.tar.gz
```

You can view/modify the source code using a text editor in Linux, but note that you only need to work on two files: mapper.py and reducer.py. If you find it more convenient to do the editing/packaging in an IDE such as IDLE or PyCharm, this is possible too.

1.1.2 Java Version

```
wget http://web.it.usyd.edu.au/~comp5349/code/mrsim_java.tar.gz
tar xzvf mrsim_java.tar.gz
ant
```

You can view/modify the source code using a text editor in Linux, but note that you only need to work on two files: RatingFilter.java and RatingReducer.java. Running the command **`ant`** again will generate a new jar. If you find it more convenient to do the editing/packaging in an IDE such as Eclipse, this is possible too.

1.2 Format of the input data

In the downloaded archive, you also find an example data set (**`data.csv`**) in CSV (comma separated values) format that contains 100,000 user ratings for films. Users and films are given as numeric IDs. The file format is:

```
user_id \t film_id \t rating \t timestamp
```

Note: Be careful parsing this file – there are a few lines which are incomplete, and also make sure that user and film ids are integer values, and ratings should be integers in the range 1 to 5.

1.3 Python Implement the missing RatingFilter and RatingReducer classes

Your homework task is to implement two missing classes: **RatingFilter** and **RatingReducer**, so that the final program determines the average film rating (as float value rounded to 1 decimal place) for those films whose ID is in a given range. You will find the skeleton source code of those classes in the within the respective python files.

RatingFilter

The RatingFilter class is a specialization of the generic Filter class. You have to implement an initialiser that allows to specify the search range of film ids (min and max film_id), as well as a filter() method.
The filter method is called for each individual line of the input file. Each string, which you process in this method, corresponds to the format specified in Section 1.2. The filter method takes a String as input and either returns a tuple in the form of (key, value) as result, if the given input string is about a film id in the search range, or *None* otherwise. The tuple should consist of the film id as key, and the rating for the film in the current input line as value. The structure and methods of the class are defined in the same file.

RatingReducer

The RatingReducer class is a specialization of the generic Reducer class. You have to implement a reduce method which computes the average rating (as Float) of all the given input ratings of the same film. The reduce() method is called by our map_reduce_simulator with a String key and a List of Integer values. For our example data set, this will be the filtered film_id (key) and the list of integer ratings given by various users to this film. The reduce() method shall compute the average rating of all those input ratings and return as result the average rating as a Float value rounded to one decimal place (eg. 3.5).

Once you implemented the content of these two classes, you can then run the Python program as described below.

1.3.1 Python: Running the Python program and example output

Before running the program, ensure that the example data set (data.csv) is in the same directory as the main python file (map_reduce_simulator.py). The program is then executed as follows:

```
python3 map_reduce_simulator.py
```

Optionally, you can provide an input file name and different film ids as command line parameters. The program takes three (optional) parameters: an input file path and the range (start and end) of film ids to search for:

```
python3 map_reduce_simulator.py  data.csv  1  2
```

If you code is correct, it should produce the following output for films 1 to 2:

```
1: 3.9
2: 3.2
MapReduce Simulator using mapper:  <mapper.RatingFilter object at … >
MapReduce Simulator using reducer: <reducer.RatingReducer object at … >
Lines in File:    100000 records (should be: 100000)
Filtered records: 582 (should be: 582)
```

1.4 Java Implement the missing RatingFilter and RatingReducer classes

Your homework task is to implement two missing classes: **RatingFilter** and **RatingReducer**, so that the final program determines the average film rating (as float value rounded to 1 decimal place) for those films whose ID is in a given range. You will find the skeleton source code of those classes in the in the src/mrsim/ subdirectory.

RatingFilter

The RatingFilter class is a specialization of the generic Filter class (cf. Filter.java). You have to implement an constructor that allows to specify the search range of film ids (min and max film_id), as well as a filter() method.
The filter method is called for each individual line of the input file. Each string, which you process in this method, corresponds to the format specified in Section 1.2. The filter method takes a String as input and either returns a *Record* as result, if the given input string is about a film id in the search range, or *null* otherwise. The Record should consist of the film id as key, and the rating for the film in the current input line as value. The structure and methods of the Record class are defined in Record.java.

RatingReducer

The RatingReducer class is a specialization of the generic Reducer class (cf. Reducer.java). You have to implement a reduce method which computes the average rating (as Float) of all the given input ratings of the same film. The reduce() method is called by our MapReduceSimulator with a String key and a List of Integer values. For our example data set, this will be the filtered film_id (key) and the list of integer ratings given by various users to this film. The reduce() method shall compute the average rating of all those input ratings and return the result as a Float object rounded to one decimal place (eg. 3.5).

Once you implemented the content of these two classes, you can compile the source code with the **`ant`** command:

```
ant
```

1.4.1 Java: Running the Java program and example output

After successfully compiling the example code, a jar file (**`mrsim.jar`**) should have been created in the current directory. Make sure, that the example data set (**`data.csv`**) is in the same directory than this jar file. The program is then executed as follows:

```
java -jar mrsim.jar
```

Optionally, you can provide an input file name and different film ids as command line parameters. The program takes three (optional) parameters: an input file path and the range (start and end) of film ids to search for:

```
java -jar mrsim.jar  data.csv  1 2
```

If you code is correct, it should produce the following output for films 1 to 2:

```
1: 3.9
2: 3.2
MapReduce Simulator using mapper: comp5349.mrsim.RatingFilter@...
MapReduce Simulator using mapper: comp5349.mrsim.RatingReducer@...
Lines in File:    100000 records (should be: 100000)
Filtered records: 582 (should be: 582)
```