Big Data Management
2st programming assignment
Counting common Facebook friends using MapReduce

Name : Spyratos Angelos
Email : dsc17021@uop.gr
Reg.Num. : 2022201704021

Name : Kotsiras Dimitris
Email : dsc17009@uop.gr
Reg.Num : 2022201704009

1. Instructions

   The task needs hadoop standalone version to be installed and configured to work with a compatible version of Java. At first, we need to move our java files in the hadoop installation directory and create 2 folders for the input files (we place the input files in separate folders as we need to do separate runs for each file to produce separate output files as our code was not optimized to do it automatically). In order to create the class files from the java files we need to run in bash the following commands:
   bin/hadoop com.sun.tools.javac.Main FriendsNum.java
   bin/hadoop com.sun.tools.javac.Main FriendsCommon.java

   After doing so we now have the class files and can compile the jars by running:
   jar cf friendsnum.jar FriendsNum.class*
   jar cf friendscommon.jar FriendsCommon.class*

   Now we are ready to run our Jar executables to get the outputs. So we run the following:

   bin/hadoop jar friendsnum.jar FriendsNum input20/ output20/

   bin/hadoop jar friendsnum.jar FriendsNum input500/ output500/

   bin/hadoop jar friendscommon.jar FriendsCommon input20/ output20-common/

   bin/hadoop jar friendscommon.jar FriendsCommon input500/ output500-common/

   the input20 and input500 folders hold the respective input files, the output20 and output500 folders contain the resulting files for the first question, while the folders output20-common and output500-common contain the results for the second question.

2. Calculate the number of friends

   Here we created a Map method that uses the first word of each line (first username) as a Key and emits this key with 1 as value for each other word in the same line (each “friend”).
   Then we created a Reducer method that is used both as a combiner and as a

reducer (as they have the exact same functionality) where we simply sum these values of 1 on a per key basis. So, the combiner sends the partially summed values and the reducer simply sums these values to produce the results.

3. Common friends between all pairs of persons

   For the problem of finding common friends between all pairs of persons we created a Map function that takes the first person of each line of the file, creates some tuples containing the first person and each other person respectively over a loop (we check the names order each time so that we avoid creating different keys for the same 2 persons) so that these tuples can be used as keys during the emit process and finally emits these keys with the friend list of the first person.

   This way, the reducer could take as input something like this :
   key=(personA,PersonB), values =["friends of personA","friends of personB"]
   Then, the reducer simply compares the friends contained in the first string with the ones in the second to find which are contained in both, thus creating the final result.

4. Outputs.

   Except from the fact that we didn't manage to fix both our codes to have the ability to distinguish the 2 input files in order to create automatically the 2 output files we had no problems. Both input files run smoothly and quite fast and produced the desired results!