Data Challenge

Language: *Scala (2.10)*
Framework: *Apache Spark (1.6.0)*
Build: *Maven*
IDE: *Scala IDE (Eclipse)*

How to run

- Import this project in Scala IDE (or eclipse) as maven project and give 2 (or 3) input arguments:
    - input file path
    - N
    - (optional) output file path (preferably hdfs path not local path)
- or you can just build and run the project using following
    - mvn clean install (don't forget to check fat-jar)
    - run this command (as i'm loading and saving files using spark context so it would be great if you can give the hdfs path of both files)
        - spark-submit --class i_degree_friends.Driver --master local[*] i-degree-friends-project-0.0.1-SNAPSHOT-jar-with-dependencies.jar input-file-path N outputFilePath(optional)
- or you can just run spark-submit command on attached fat-jar. (I'm sending you fat-jar also)

What I did

I've used Spark to solve this problem which is a large scale distributed computing framework for big data. RDDs are being used to exploit the parallelism and distributed abstraction of spark. Data is ingested in RDDs so automatically it gets distributed across spark executors.

Time and space complexity concepts become exciting in the case of spark because of its inherent distributed and large scale nature. Let suppose if some work was being done in O(n) then if we do it on spark then normally that would take O(n/k) where k is the number of machines/cores in spark cluster and same is the case for memory purpose in space complexity. I'll roughly list down the time and space complexities of the steps that I've done in my program.

There's one more thing also. I've created a dictionary of person corresponding to its friends list. This is for now in memory structure. I think as our data will grow we can do different things here. Like we can use Redis (due to it's in memory functionality) for efficient querying or we can use some other key-value NoSQL DBs also.

Space Complexity

1. RDD is created from a file of tuples: O(n/K) where K is the number of machines in spark cluster
2. Pair RDD is created (so that we can have all kind of combinations for later purposes): O(2n/K)

3. FriendsRDD is created using groupByKey: O((F+ D)/K) where F is number of persons and D is the list of friends that a person have.
4. Friends collection is created and broadcasted so that we can use this as dictionary: O(F+D)
5. FindIDegreeFriends Function
    a. it will run Q (Q = degree) times for each person. for each person it also creates three temporary collections of size O(N): O((Q*3N)/K)
6. I-Degree-Friends are collected and returned: O(F)

Time Compelxity

1. PairRDD is created by traversing a complete RDD: O(n/K)
2. friendsRDD is created using groupBYKey: Now argument can be made that we can use one of reduceByKey, foldByKey or combineByKey rather than groupByKey due to groupBY's shuffling but thing is that if we do some of former operations then we'll have to create new collection in each step to add the previous collection. So that would be really inefficient. So that's why I went for groupBYKey here
3. findIDegreeFriends Function
    a. it will run Q (where Q = degree) times for each person and will find i degree friends using that collection that we stored and broadcasted: O((F*Q)/K)
4. I-Degree-Friends are collected and returned: O(F)

Correctness

I've also written test case for N=2 and verified that. Apart from that I've done and tested for N=1 to 5 for the given sample data set.

Now regarding the correctness. I found list of friends of every person and stored that in dictionary. This dictionary has 1-Degree friends of every person. For N>1 what I'm doing is that, I'm doing breadth first kind of search and finding all further degree friends. I'm also taking into account the fact that doing breadth first can also give us redundant friends as we can run in to a cyclic structure so in each step I'm removing those friends from list which have already been explored in breadth. I think this solution should work for large and diverse datasets also.

Misc

For test purposes I'm also sending you the sample input file and output file that I got after running my program on the sample input file.

Sample input file: inputFile.txt

Sample output file (N = 2): outputFile

Following is the screenshot (eclipse and linux console) for N=3 on the given sample data set.

```
17/05/29 01:35:55 INFO Executor: Finished task 1.0 in stage 8.0 (TID 11). 1299 bytes result sent to driver
17/05/29 01:35:55 INFO Executor: Finished task 0.0 in stage 8.0 (TID 10). 1334 bytes result sent to driver
17/05/29 01:35:55 INFO TaskSetManager: Finished task 0.0 in stage 8.0 (TID 10) in 28 ms on localhost (1/2)
17/05/29 01:35:55 INFO TaskSetManager: Finished task 1.0 in stage 8.0 (TID 11) in 29 ms on localhost (2/2)
17/05/29 01:35:55 INFO TaskSchedulerImpl: Removed TaskSet 8.0, whose tasks have all completed, from pool
17/05/29 01:35:55 INFO DAGScheduler: ResultStage 8 (collect at Driver.scala:55) finished in 0.030 s
17/05/29 01:35:55 INFO DAGScheduler: Job 3 finished: collect at Driver.scala:55, took 0.080387 s
brendan davidbowie      kim     omid    torsten
davidbowie      brendan kim     mick    omid    torsten ziggy
kim     brendan davidbowie      mick    omid    torsten ziggy
mick    davidbowie      kim     omid    ziggy
omid    brendan davidbowie      kim     mick    torsten ziggy
torsten brendan davidbowie      kim     omid    ziggy
ziggy   davidbowie      kim     mick    omid    torsten
17/05/29 01:35:55 INFO SparkContext: Invoking stop() from shutdown hook
```

```
17-05-29 01:34:45 INFO org.apache.spark.scheduler.TaskSetManager: Finished task 0.0 in stage 8.0 (TID 10) in 42 ms on localhost (2/2
)
17-05-29 01:34:45 INFO org.apache.spark.scheduler.DAGScheduler: ResultStage 8 (collect at Driver.scala:55) finished in 0.042 s
17-05-29 01:34:45 INFO org.apache.spark.scheduler.TaskSchedulerImpl: Removed TaskSet 8.0, whose tasks have all completed, from pool
17-05-29 01:34:45 INFO org.apache.spark.scheduler.DAGScheduler: Job 3 finished: collect at Driver.scala:55, took 0.101261 s
brendan davidbowie      kim     omid    torsten
davidbowie      brendan kim     mick    omid    torsten ziggy
kim     brendan davidbowie      mick    omid    torsten ziggy
mick    davidbowie      kim     omid    ziggy
omid    brendan davidbowie      kim     mick    torsten ziggy
torsten brendan davidbowie      kim     omid    ziggy
ziggy   davidbowie      kim     mick    omid    torsten
17-05-29 01:34:45 INFO org.apache.spark.SparkContext: Invoking stop() from shutdown hook
```