Assignment 1: Basics and Map-Reduce

Formative, Weight(10%), Learning objectives (1,2,3),
Abstraction (4), Design (4), Communication (4), Data (5), Programming (5)

Due date: 11:59pm, 21 August 2016, Weight: 10.0 % of the course

1 Overview

Assignments should be done in groups consisting of two students. If you have problems finding a group partner use the forum to search for group partners or contact the lecturer.

2 Assignment

Exercise 1 *Suspected Pairs (10 points)*

Using the information from the first lecture (or Section 1.2.3 in the textbook), what would be the number of suspected pairs if the following changes were made to the data (please calculate the number for each change respectively).

1. The number of days of observation was raised to 3000.
2. The number of people observed was raised to 3 billion (and there were therefore 300,000 hotels).
3. We only reported a pair as suspect if they were at the same hotel at the same time on four different days.

Exercise 2 *Summary of 2.4 and 2.5 (10 +10 points)*

For this exercise you have to read Section 2.3.9-2.3.11, 2.4, and 2.5 in Leskovec, Rajaraman, Ullman (second edition, 2014).

- Summarize the content of 2.4 in your own words (0.75-1 page).
- Summarize the content of 2.5 in your own words (0.75-1 page).

Exercise 3 *Hadoop (15+15 points)*

For this exercise, you have to set up and configure your system to use Hadoop. Follow the instructions in Stanford document at `http://web.stanford.edu/class/cs246/homeworks/tutorial.pdf` and set up the virtual machine as described in Section 1. Run the example program of Section 2 and carry out the different steps given in that section.

- Write your own Hadoop Map-Reduce job that outputs the number of words that start with each letter (see Section 3 of the Stanford document).
- Run your job on the file http://www.gutenberg.org/files/100/100.txt in *standalone mode* and *pseudo-distributed mode* and record the output.

Exercise 4 *Friend Recommendation System (Stanford 2014) (40 points)*

Write a MapReduce program in Hadoop that implements a simple People You Might Know social network friendship recommendation algorithm. The key idea is that if two people have a lot of mutual friends, then the system should recommend that they connect with each other. You have to run the program on the system set up previously in Exercise 3 in order to receive points for this exercise.

Input: Download the input file from the link: http://snap.stanford.edu/class/cs246-data/hw1q1.zip. The input file contains the adjacency list and has multiple lines in the following format:

```
<User><TAB><Friends>
```

Here, `<User>` is a unique integer ID corresponding to a unique user and `<Friends>` is a comma separated list of unique IDs corresponding to the friends of the user with the unique ID `<User>`. Note that the friendships are mutual (i.e., edges are undirected): if A is friend with B then B is also friend with A. Algorithm: Let us use a simple algorithm such that, for each user U, the algorithm recommends N = 10 users who are not already friends with U, but have the most number of mutual friends in common with U.

Output: The output should contain one line per user in the following format:

```
<User><TAB><Recommendations>
```

where `<User>` is a unique ID corresponding to a user and `<Recommendations>` is a comma separated list of unique IDs corresponding to the algorithms recommendation of people that `<User>` might know, ordered in decreasing number of mutual friends. Even if a user has less than 10 second-degree friends, output all of them in decreasing order of the number of mutual friends. If there are recommended users with the same number of mutual friends, then output those user IDs in numerically ascending order. Also, please provide a description of how you are going to use MapReduce jobs to solve this problem. Do not write more than 3 to 4 sentences for this: we only want a very high-level description of your strategy to tackle this problem. Note: It is possible to solve this question with a single MapReduce job. But if your solution requires multiple map reduce jobs, then that is fine too.

For your submission

- Include your source code
- Include in your writeup a short paragraph describing your algorithm to tackle this problem.
- Include in your writeup the recommendations for the users with following user IDs: 924, 8941, 8942, 9019, 9020, 9021, 9022, 9990, 9992, 9993.

3 Procedure for handing in the assignment

Work should be handed in using the course website. The submission should include:

- pdf file of your solutions for theoretical assignments
- all source files
- descriptions as required in the statement of the exercises
- Hadoop outputs for the exercises
- a README.txt file containing instructions to run the code, the names, student numbers, and email addresses of the group members

In addition, there will be a discussion session where you will have to explain your solutions:

1. Individuals (not groups) will be chosen randomly (by a program) to explain their solutions to parts of exercises in the assignment.
2. Each member of a group shall be able to explain their solution. An answer like "my group partner did this hence I can't explain this solution" will be counted as "no explanation" ($\rightarrow$ 0 marks for the particular part).
3. Absence without a formal excuse[1] for the session will be counted as "no explanation" if the person is chosen to explain his/her solution.

[1] Markus Wagner and Junhua Wu need to be informed via email *before* the discussion session.