Programming Assignment 1

Profiling Ego Networks in Social Media using Scalable Graph Algorithms

Due: September 23, 2021 5:00PM
Submission: via Canvas, individual submission

Objectives
The goal of this programming assignment is to enable you to gain experience in:

- Profiling a large-scale Social Network Graph using Hadoop MapReduce
- Extracting a feature graph using Hadoop MapReduce

1. Introduction

Social networks allow users to interact with their friends and acquaintances via posts. Current social networking sites organize their networks and posts to categorize their friends into social circles. For example, 'circles' on Google+ and 'lists' on Facebook and Tweeter have similar functionality. To capture "initiative" actions between users, a social network is often represented as an Ego network. "Ego" is an individual "focal" node. A network has as many egos as it has nodes. Egos can be persons, groups, organizations, or whole societies.

Circles are user-specific as each user organizes their personal network or friends independently of all other users. In Figure 1, we are given a single user u and we form a network encompassing the set $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ of that user's friends. We refer to the user u as the *ego* and the set of nodes V_i as *alters*. Ego networks consist of a focal node ("ego") and the nodes to whom ego is directly connected to (these are called "alters") plus the ties, if any, among the alters.

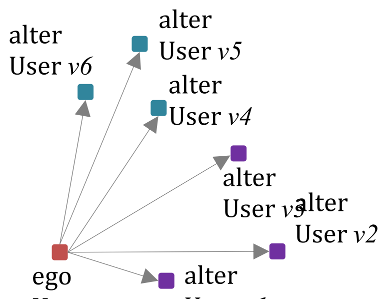

Figure 1: An ego network with a single ego

This assignment uses Google+ ego-networks. The Google+ dataset consists of 'circles' and was collected from users who had manually shared their circles using the 'share circle' feature. The Google+ circles are quite distinctive compared to Facebook or Twitter. Their creators of circles had chosen who they would release their posts to. As a result, the Google+ ego network is a directed network. As an interesting example, one circle contains candidates from the 2012 primaries, who presumably do not follow their followers, nor each other.

The objectives of this assignment include:

- Calculating the global statistics of the graph such as the number of edges using MapReduce
- Counting In-degree and out-degree of all vertices in the graph using MapReduce
- Extracting a "Friends" network using MapReduce

2. Programming Requirements

2.1 Programming Language

You are required to use Java (Version 1.8 or higher) for this assignment.

2.2 Installing and configuring Hadoop

For this assignment, you will be working on your own Hadoop Cluster, which should have finished this as a part of PA0. The walkthrough guidelines are available at: [Link]

2.3 Dataset

The dataset is organized as tuples of [User A, User B], where the user A invited the user B to see his/her posts. The file looks like:

```
    …
    293840028 948276384
    389479278 998983872
    393300021 100000029
    …
```

For example, the user 293840028 has invited 948276384 to her circle, and the user 389479278 has invited 998983872 to his circle. Therefore, there is an edge from 293840028 to 948276384. Similarly, there is an edge from 393300021 to 100000029.

The dataset is available at: [Link] Note that this is the preprocessed dataset with all the duplicate edges removed

3. Counting the number of edges

In this assignment, you should count the number of total edges included in this graph using MapReduce. Your software must return ***one total count*** for this problem.

4. In-degree and out-degree of distinct vertices

You should measure the in-degree and out-degree of each distinct vertex using MapReduce. Your software must return two sorted lists of distinct vertices with associated counts in descending order; one of the in-degree and another for the out-degree. For the submission, you must submit the first 100 vertices in this list. For vertices that have the same degree, your list of vertices must be sorted in alphabetical order (ascending).

5. Extracting the "Friends" network

In this assignment, we define a "Friends" relationship as a pair of vertices that have edges in both directions. Since this dataset represents only directional relations, if there is a pair of nodes A and B and two different edges (A, B) and (B, A) exist, we assert that there is a "*Friend*" relationship between nodes A and B. In this context, if two users invited each other to their circles, these users will be considered as "Friends". In Figure 2, user 7 has 6 alters, but has only one friend, User 1. Similarly, user 4 and user 3 are friends in this network.

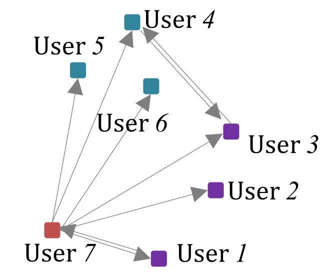

Figure 2. "Friends" network

You should create a network with only "Friends" from the original Google+ network. If there are nodes that do not have any "Friend", please do not include those nodes in your results. If there are nodes with multiple "Friends", include all the friends in your results. Your graph must be represented as a list of "Friend" edges. The edges should include two vertex ids, V1 and V2 and these vertex ids must be in an alphabetical order (ascending order). Finally, your complete edge list must be sorted in the alphabetical order (ascending order) of the first vertex. For the submission, you must submit the first 100 edges in this list.

For example, for the network depicted in Figure 2, your output should be,

```
User1 User7
User3 User4
```

6. Submission

This is an individual assignment. The result of your computation should be stored as file(s). You should generate the results using the Hadoop's MapReduce programing framework and **not a standalone program** that executes only on one machine [standalone programs will result in an automatic zero on the assignment].

Please submit a tar ball of your source files and output data (From the full dataset) via Canvas. The source files should include your java code of the Mapper/Reducer functions and any script file that you will use for demo. You will download your source files/script from Canvas for the demo. Do not miss any files. For your demo, you are not allowed to use any file outside of your Canvas submission.

7. Grading

Each of the submissions will be graded based on the demonstration of your software to GTA. During the demonstration, you should present capabilities to accomplish the following requirements:

1. Counting the number of edges and distinct vertices (2 points)
2. In-degree and out-degree of distinct vertices (2 points)
3. Extracting the "Friends" network (3 points)

Note: You should generate the results using the Hadoop's MapReduce programing framework and not a standalone program that executes only on one machine [standalone programs will result in an automatic zero on the assignment].
You should generate the results using the Hadoop's MapReduce programing framework with at least 5 nodes in the cluster.
If your software does not use MapReduce, your score will be zero on the assignment.

The demo includes short interviews of your software design and implementation details. Your responses and demonstration of capabilities will count towards your score. This assignment will account for 7% of your final grade. The grading will be done on a 7 point scale.

You are required to work alone on this assignment.

4. Late policy

Please check the late policy posted on the course web page