Social Media Analysis (SS2016) Dr. Krishna P. Gummadi

Exercise Sheet 5 – Community Detection I

Max Planck Institute for Software Systems / Saarland University

Instructions

1. You have to submit this assignment (either digitally or on paper) by 14:00 on 28th June 2016.
2. Irrespective of your submission preferences (digital or paper), you have to email the code of the coding assignments to the instructors.
3. To submit your assignments digitally, email your solutions to sma-ss16-instructors@mpi-sws.org, with the subject as "[SMA] HW 5 : <last name 1> <matriculation number 1>, <last name 2> <matriculation number 2> solutions".
4. Add your full name(s) and the matriculation number(s) in the beginning of all types of assignment solution submissions (in digital copy, on paper, inside the code as comment etc).
5. Instructions for coding assignment:
   (a) You are free to use any programming language you are comfortable with. A reference for an efficient way to implement graph classes and methods is the github code base for snap at `https://github.com/snap-stanford/snap`.
   (b) You should be able to run your code in a computer with a reasonable configuration (for instance 2GB or more RAM).
   (c) Please include the results outputted from your code into your solution sheet as described in the respective tasks. Additionally, you should also submit your code and instructions on how to run your code in a text file via email to the instructors. The name of the main code file for Task 1 should be: **agglomerative-approach.py** (or some other programming language), and the one for Task 2 should be **divisive-approach.py** (or some other programming language). The name of the running instruction file should be: **compute.community-detection.howtorun.txt**. If there are any additional resources (eg. library) that are required to run your code, please mention those in the how to run file.
   (d) We should be able to run the submitted code on our machine. If it fails to run on our machine, you will not get any marks for the coding part of the assignment.
   (e) If any part of your code takes a long time to run (e.g., more than 10 minutes) report that in the instruction file with an estimate of time required.
6. For graph visualization in Task 1 and 2, you are free to use any tool for visualization that you are comfortable with. A reference tool is Gephi, which is easy to install and use. Gephi can be downloaded from `https://gephi.org/`, where you can also find documentation about how to use it.

Problem 1 (10 marks)

Motivation and dataset description

As explained in the class, communities can be detected in networks using two kinds of hierarchical clustering approaches: agglomerative and divisive clustering. These two approaches are briefly summarized below:

I. **Agglomerative (bottom-up) Clustering:**

1. Start with all nodes in their own cluster.
2. In each iteration, greedily merge the two most similar clusters, stopping when there is only one cluster left. The pairwise similarity between two nodes i and j – $S(i, j)$ – is given by the number of common friends that nodes i and j have. Similarity between two clusters A and B can be defined as:

$$S(A, B) = max_{x_A \in A, x_B \in B} S(x_A, x_B) \quad (1)$$

II. **Divisive (top-down) Clustering:**

1. Start with all nodes in one large cluster.
2. In each iteration, repeatedly remove the edge with the highest betweenness centrality, until each nodes is in its own cluster. You can use the GetBetweennessCentr function from SNAP, which also gives the edge betweenness values.

The goal of this exercise is to detect communities using the above two hierarchical clustering approaches in the Rice University students' Facebook network. The network can be downloaded from the following page: `http://courses.mpi-sws.org/sma-ss16/assignmentData/community-detection/rice-univ-facebook-links.elist.txt.gz`. The graph is undirected and unweighted. Each line in the file contains two anonymized user IDs, which indicates that a friendship link exists between the two users.

Task 1 (5 marks)

Write a code to compute the communities in the aforementioned dataset, using the agglomerative approach described above. Your code should take as input the edge list file. In each iteration step, when the number of edges are equal to $k\%$ of total number of edges in the whole graph (for $k \in [5, 10, 25, 50, 75]$), your code should output the edges formed till that iteration step. Store and submit these edges in a file named agglomerative-approach-step-$< k >$.elist.

Also visualize the resulting graph at the iteration step for every k, using a visualization tool of your choice and submit as a pdf file named agglomerative-approach-communities.pdf, also indicating the value k for each figure.

Task 2 (5 marks)

Write a code to compute the communities in the aforementioned dataset, using the divisive approach described above. Your code should take as input the edge list file. In each iteration step, when the number of edges are equal to $k\%$ of total number of edges in the whole graph (for $k \in [5, 10, 25, 50, 75]$), your code should output the edges remaining till that iteration step. Store and submit these edges in a file named divisive-approach-step-$< k >$.elist.

Visualize the resulting graph at each iteration step for every k, using a visualization tool of your choice and submit as a pdf file named divisive-approach-communities.pdf, also indicating the value k for each figure.

Take a look at the graph clusters you have generated at each iteration step in the previous two tasks and compare the clusters obtained by the two approaches. Briefly describe your observations about the difference between two approaches.