Social Network Analysis for Computer Scientists
Fall 2020 — Assignment 2

`http://liacs.leidenuniv.nl/~takesfw/SNACS`

Deadline: October 26, 2020

This document contains three exercises that each consist of various numbered questions that together form Assignment 2 of the Social Network Analysis for Computer Scientists course taught at Leiden University.

For each question, the number of points awarded for a 100% correct answer is listed between parentheses. In total, you can obtain 100 points and 10 bonus points. Your assignment grade is computed by dividing your number of points by 10. Please do not be late with handing in your work. You have to hand in the solutions to these exercises **individually**. Discussing the harder questions with fellow students is allowed, but writing down identical solutions is not. Hand in your solutions, typeset using LaTeX, as instructed on the course website.

Clearly and concisely describe how you obtained each answer. Write down any nontrivial assumptions that you make. For the exercises that require some programming, you can use any programming language, scripting language or toolkit. In any case, always clearly describe which toolkit or programming language you used and how you obtained your answer using these tools. Include relevant source code, for example, in an Appendix that you reference in the text where needed. When asked for an algorithm, use simple and consistent pseudo-code.

Questions or remarks? Ask your questions during one of the weekly lectures or lab sessions, on the Brightspace discussion board, or send an e-mail.
Good luck!

Exercise 1: Clustering and centrality (18p)

The *average clustering coefficient* $C(G)$ of a connected undirected network $G = (V, E)$ indicates the extent to which nodes cluster together, and is based on the *node clustering coefficient* $c(v)$ of the nodes $v \in V$. The two are defined as follows.

$$C(G) = \frac{1}{n} \cdot \sum_{v \in V} c(v)$$

$$c(v) = \frac{2 \cdot |\{(u, w) \in E : (u, v) \in E \wedge (v, w) \in E\}|}{deg(v) * (deg(v) - 1)}$$

Here, $deg(v)$ is the degree of node v. We use $n = |V|$ for the number of nodes and $m = |E|$ for the number of undirected edges.

(3p) Question 1 Name two types of networks that have an average clustering coefficient of 0 by definition and one type of network that has an average clustering coefficient of 1 by definition.

(6p) Question 2 The *ego network* of a given node consists of that node, its direct neighbors and all connections between these neighbors. What type of relation exists between the given node's clustering coefficient and the density $\frac{m}{\frac{1}{2} \cdot n \cdot (n-1)}$ of that node's ego network?

(5p) Question 3 Consider a *bipartite network*, with its two types of nodes, and suppose there are n_1 nodes of type one and n_2 nodes of type two. Show that the mean degrees k_1 and k_2 of the two node types are given by

$$k_2 = \frac{n_1}{n_2} \cdot k_1$$

(4p) Question 4 Centrality measures are commonly used to assess the importance of individual nodes, based on their structural position in the network. By sorting nodes in the network by their centrality value, we can create a ranking of nodes. Discuss two special types of networks that we could be dealing with if the node ranking produced by *degree centrality* is equal to that of *betweenness centrality*.

Exercise 2: Diameter computation (12p)

Apply the BoundingDiameters algorithm on paper to find the exact diameter (maximum distance, length of a longest shortest path) of the undirected graph in Figure 1. The algorithm is discussed during the lectures and explained in:
F.W. Takes and W.A. Kosters, Determining the Diameter of Small World Networks, in *Proceedings of the 20th ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 1191-1196, 2011.
doi: `http://dx.doi.org/10.1145/2063576.2063748` or see
`http://liacs.leidenuniv.nl/~takesfw/pdf/diameter.pdf`.

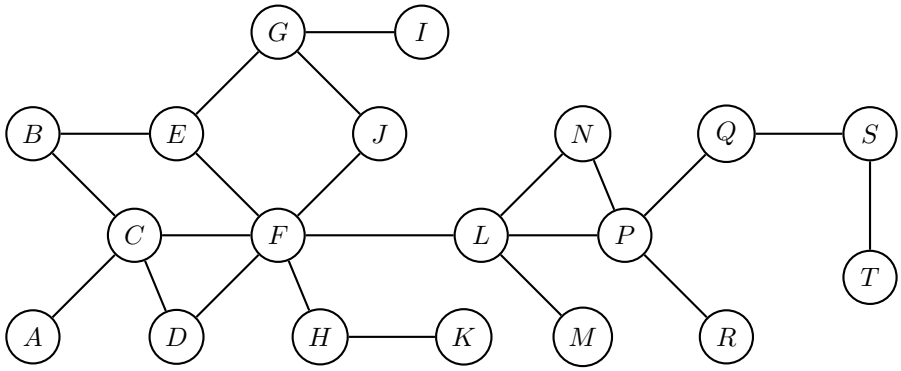

Figure 1: An undirected graph with 19 nodes.

You do not have to do the prepruning step discussed in the paper. Explain your steps in detail, and mention any nontrivial assumptions. As a selection strategy, alternate between choosing the node with the largest upper bound value and the node with the smallest lower bound value, breaking ties by taking the node with the highest degree. How many iterations did it take to compute the diameter, and how does this compare to the naive method for diameter computation?

Exercise 3: Twitter network analysis (70p+10p)

This is a practical exercise, for which you can use any toolkit or programming language. Two samples of Twitter datasets can be found at
`https://liacs.leidenuniv.nl/~takesfw/SNACS/twitter-small.tsv` and
`https://liacs.leidenuniv.nl/~takesfw/SNACS/twitter-larger.tsv`.
The full dataset (not required until Question 3.7) can be found at
`/vol/share/groups/liacs/scratch/SNACS/twitter.tsv` and
`/data/SNACS/twitter.tsv`.
The files are identical, where the first is in the university-provided remote Linux environment and the second in the LIACS data science lab environment.

Up until now, we have only looked at social network data which was already in a nicely formatted edge list. In practical network analysis research, this rarely ever happens. Therefore we will now work with real raw data from a Twitter crawl [2]. The dataset `twitter.tsv` contains over $450,000,000$ tweets, crawled from June 2009 to December 2009. The file `twitter-small.tsv` contains a small subset of these tweets that can be handled with Gephi, whereas `twitter-large.tsv` contains a bit larger subset that can be analyzed using for example NetworkX. Each line of these files contains one tweet, consisting of three tab-separated ('`\t`') fields denoting the timestamp, user who sent the tweet and the content of the tweet. For example:

```
2009-07-05 14:07:18    aeneas    Hi @achilles, how are you? #old
```

In the tweet content, a word starting with the `@` symbol (such as `@achilles`) means that user `achilles` is being mentioned by user `aeneas`, indicating that the tweet by `aeneas` was directed at or specifically about `achilles`. We refer to this as a *mention*. Mentions are the most direct sign of public communication on Twitter. Tweets can also be directed at more than one user.

The *mention graph* is a Twitter network represented by a directed graph $G = (V, E)$. In this graph, the set of nodes V consists of users (anyone sending out a tweet or being mentioned by someone else in a tweet). The set of links E consists of all user pairs (x, y) such that user x mentioned user y at least once. Optionally, this network can be a weighted directed graph where the number of times a user x mentions another user y is used for link weight. Also, the network could be analyzed over time by assigning a timestamp to each link, indicating when x first mentioned y. For the `twitter-small.tsv` dataset, answer Question 3.1–3.5.

(20p) Question 3.1 Extract the mention graph from the Twitter data. Relevant steps to do this could be:

- Parse the input file line by line (for example using Python or Perl)
- Generate the adjacency list: for each user (identified by its username), keep a list of the users that this user mentions, and possibly also count the number of mentions and keep track of the timestamp at which the user first mentioned the other user.
- Output the adjacency list as an edge list `csv`-file (perhaps with columns Source and Target) that you can import into Gephi or NetworkX, possibly also including columns for the Weight and Timestamp of each edge.

Discuss the steps that you took, and describe the issues that you ran into while parsing this "real-world" data, and how you solved them. For example, discuss possible text mining and parsing issues. From your answer, it should be possible to unambiguously reproduce your results.

(12p) Question 3.2 Present relevant statistics of your mention graph, including at least

- the number of nodes and edges,
- number and size of the strongly and weakly connected components,
- density,
- degree distributions,
- average node clustering coefficient and
- (undirected) (approximated) distance distribution of the giant component.

(8p) Question 3.3 Determine the top 20 users based on three different centrality measures (for example, betweenness, closeness and degree centrality). Mention how you deal with directionality. Discuss the results. Think of a way to compare the similarity of the rankings using some measure, and apply it.

(8p) Question 3.4 Apply a community detection algorithm to the giant component of your mention graph, and try to manually interpret and discuss the results. Briefly explain the relation between the result and the chosen algorithm and if applicable, its parameters.

(10p) Question 3.5 Visualize the giant component of the network (for example using Gephi), making the color of a node dependent on the community and the size of a node dependent on some sensible centrality measure. Optionally, you can try to come up with a way to incorporate the timestamp in the visualization. You could use node labels for the Twitter usernames and edge width to visualize the link weight.

(12p) Question 3.6 Run your code on the larger dataset given in the file `twitter-larger.tsv`, and answer Question 3.2 and 3.3.

(10p, bonus) Question 3.7 Run your code on the full dataset `twitter.tsv`, and answer Question 3.2 for the giant component. This is very challenging, and may require you to systematically filter certain users and links that are not part of the giant component, for example based on some threshold for the number of mentions. If you succeed on $x\%$ of the data, you can get up to $x\%$ of the 10 bonus points. Only do this after you have finished answering all the other questions.

[2] J. Yang and J. Leskovec, Temporal variation in online media, in Proceedings of WSDM, pp. 177–186, 2011.
Available at `dx.doi.org/10.1145/1935826.1935863`