Social Network Analysis for Computer Scientists
Fall 2014 Homework Assignment

http://www.liacs.nl/~ftakes/SNACS

Deadline: September 26, 2014, 11:15

This document contains 3 exercises that each consist of various numbered questions that together form the fall 2014 homework assignment 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 and sums to a total of 100 points plus 10 bonus points. Your grade (between 1 and 10, bounds included) 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 via e-mail (ftakes@liacs.nl) in a PDF-file, preferably generated using LATEX.

For each question, clearly describe how you obtained your 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. All practical exercises can be done on the student workstations. Most of the practical questions can be answered using Gephi (http://gephi.org). 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).

Questions or remarks? Contact the lecturer at ftakes@liacs.nl, walk by room 156a or ask your questions during one of the weekly lectures.

Exercise 1: Neighborhoods and Centrality (25p)

A directed graph $G = (V,E)$ consists of a set of nodes $V$ and a set of directed links $E$. For the number of nodes $|V|$ we use $n$, and the number of links $|E|$ will be denoted by $m$. The neighborhood $N(v)$ of a node $v \in V$ is defined as the set of nodes to which $v$ links:

$$N(v) = \{ w \in V : (v, w) \in E \}$$

Similarly, the reverse neighborhood $N'(v)$ can be defined as the set of nodes that links to node $v$:
\( N'(v) = \{ u \in V : (u, v) \in E \} \)

The notion of a neighborhood can be extended by defining the neighborhood of a set of nodes \( W \) as

\[ N(W) = \{ w \in V : v \in W \land (v, w) \in E \} \]

For convenience, for a node \( v \in V \) we say that \( N(v) = N(\{v\}) \). Next, the \( k \)-neighborhood is defined as follows for \( k > 0 \):

\[ N_k(W) = N(N_{k-1}(W)) \]

With \( N_0(W) = W \) for the case \( k = 0 \).

So, essentially, the \( k \)-neighborhood allows us to apply the neighborhood function to a set of nodes \( k \) times. Using this notion, it is possible to define various other measures. The distance \( d(u, v) \) is defined as the number of links that has to be traversed to get from node \( u \) to node \( v \) (and is equal to \( \infty \) if there is no such path). Using the notion of a neighborhood, the distance \( d(u, v) \) from node \( u \) to \( v \) can be defined as the smallest \( k \) such that \( v \in N_k(u) \) (of course, \( k \geq 0 \)).

**Question 1** What restriction on the graph has to apply so that the distance function \( d(u, v) \) always returns a finite value?

**Question 2** Define the indegree and outdegree of a node using the notion of a (reversed) neighborhood.

**Question 3** Give a definition of the reversed \( k \)-neighborhood \( N'_k(W) \), analogously to the definition of the \( k \)-neighborhood.

**Question 4** The eccentricity \( e(v) \) of a node \( v \) is the length of a longest shortest path starting at node \( v \) (so, the distance to a node furthest away from \( v \)), formally, \( e(v) = \max_{w \in V} d(v, w) \). Write down a definition of eccentricity without using the distance function, but by using the notion of a \( k \)-neighborhood.

**Question 5** The closeness centrality value \( c(v) \) of a node \( v \) indicates the average distance from node \( v \) to all other nodes, and is defined as \( \frac{1}{n-1} \sum_{w \in V} d(v, w) \). Give a definition of closeness centrality without using the distance function, but by using the notion of a \( k \)-neighborhood. Hint: you can use the eccentricity function \( e(v) \) from the previous question, which tells you what the distance to the node furthest from \( v \) is.
Exercise 2: Clustering Coefficient and Densest Subgraph Algorithm

The average graph clustering coefficient $C(G)$ of an undirected graph $G = (V, E)$ indicates the extent to which nodes cluster together, and can be defined as

$$C(G) = \frac{1}{|V|} \sum_{v \in V} \frac{2 \cdot |\{(u, w) \in E : (u, v) \in E \land (v, w) \in E\}|}{\text{deg}(v) \cdot (\text{deg}(v) - 1)}$$

Here, $\text{deg}(v)$ is the degree of a node $v \in V$, so the number of edges adjacent to node $v$.

(5p) Question 1 Consider all possible undirected graphs with $n = 8$ nodes and $m = 12$ edges. Draw such a graph (for example in \LaTeX using the \texttt{tikz} package) with a minimal average graph clustering coefficient.

(10p) Question 2 Give an exact algorithm, in words or in pseudo-code, for finding a graph with a minimal average graph clustering coefficient for any given $m$ and $n$. What can you say about the complexity of your algorithm?

\begin{figure}[h]
\centering
\begin{tikzpicture}
\node[fill,circle,inner sep=1] (A) at (0,0) {A};
\node[fill,circle,inner sep=1] (B) at (1,0) {B};
\node[fill,circle,inner sep=1] (C) at (2,0) {C};
\node[fill,circle,inner sep=1] (D) at (-1,-1) {D};
\node[fill,circle,inner sep=1] (E) at (1,-1) {E};
\node[fill,circle,inner sep=1] (F) at (3,-1) {F};
\node[fill,circle,inner sep=1] (H) at (3,1) {H};
\node[fill,circle,inner sep=1] (I) at (-1,1) {I};
\node[fill,circle,inner sep=1] (J) at (1,1) {J};
\node[fill,circle,inner sep=1] (K) at (2,1) {K};
\node[fill,circle,inner sep=1] (L) at (3,1) {L};
\draw (A) -- (B) -- (C);
\draw (D) -- (E) -- (F);
\draw (E) -- (H);
\draw (I) -- (J) -- (K) -- (L);
\end{tikzpicture}
\caption{An undirected graph with 11 nodes and 16 edges.}
\end{figure}

(10p) Question 3 Use the greedy algorithm by Charikar et al. (see Lecture 2 slides or [1]) to compute which nodes form the densest subgraph of the graph shown in Figure 1. Write down relevant state variables of the algorithm in each iteration as well as relevant branching steps.


Exercise 3: An Online Social Network

This is a practical exercise, for which you can use any toolkit or programming language. A dataset of an online social network is given in the shared UNIX file \texttt{/vol/share/groups/liacs/scratch/SNACS/network.in}. For convenience
and speed, you can copy the file located in that folder to the local hard disk (usually, that is /scratch on a student workstation).

The file contains a list of directed friendships of an online social network. Each line looks like `userA[whitespace]userB[newline]` and represents one directed link from a person identified by `userA` to a person identified by `userB`. You may assume that these identifiers are integers that fit in a 4-byte `signed int` in C. For each of the following questions, remember to write down how you obtained your answer and include any relevant source code.

(5p) **Question 1** How many links does this graph have?

(5p) **Question 2** How many users does this graph have? Hint: a node counts as a node if it is part of some link (either as source or target).

(10p) **Question 3** Visualize the social network so that it can be printed on A4 paper. Give the size and optionally the color of a node a sensible meaning based on some node-based measure. Which algorithm did you use?

(5p) **Question 4** How many nodes and links are in the giant component of this graph? To avoid confusion, define how you interpret “giant component” with respect to directed links.

(10p) **Question 5** Which node has the highest indegree? Give the indegree and outdegree distribution of this graph (so, for each degree value the number of times that it occurs). Present the results in a diagram, either generated using your graph analysis toolkit or by using the raw data and a simple tool such as `gnuplot`.

(5p) **Question 6** What is the average distance between two nodes of this graph? You may give an exact or an approximate value, as long as you explain how you obtained the value.

(10p) **Question 7** Give the answer to Question 1, 2, 4, 5 and 6 (2p for each question) for `/vol/share/groups/liacs/scratch/SNACS/larger.in`. This file is a larger version of the `network.in` graph. This may take some more computation time and memory, but should be possible on a student workstation with 16GB memory, perhaps even in Gephi.

(10p) **BONUS Question 8** Give the answer to Questions 1, 2, 4, 5 and 6 for `/vol/share/groups/liacs/scratch/SNACS/huge.in` (2p for each question), which is the full version of this online social network. This may take a considerable amount of computation time and memory, and will not be possible using standard toolkits such as Gephi, but should be possible on a student workstation with 16GB memory and some custom code. Hint: the number of nodes is greater than 3 million and the number of links is greater than 50 million.