Aalto University, School of Science
CS-E5740 Complex Networks, Fall 2021
Saramäki, Korhonen
Exercise set #7

# Exercise set #7 (18 pts)

- The deadline for handing in your solutions is November 15th 2021 23:59.
- Return your solutions (one `.pdf` file and one `.zip` file containing Python code) in MyCourses (Assignments tab). Additionally, submit your pdf file also to the Turnitin plagiarism checker in MyCourses.
- Check also the course practicalities page in MyCourses for more details on writing your report.

## 1. Weight–topology correlations in social networks (12 pts)

In this exercise, we will do some weighted network analysis using a social network data set describing private messaging in a Facebook-like web-page[1]. In the network, each node corresponds to a user of the website and link weights describe the total number of messages exchanged between users.

In the file `OClinks_w_undir.edg`, the three entries of each row describe one link:
`(node_i node_j w_ij)`,
where the last entry `w_ij` is the weight of the link between nodes `node_i` and `node_j`.

You can use the accompanying Jupyter notebook template to get started. `scipy.stats.binned_statistic` function is especially useful throughout this exercise.

a) (3 pts) Before performing a more sophisticated analysis, let us begin with taking a look at some basic network statistics to get a rough idea of what the network is like. To this end, plot the complementary cumulative distribution function (CCDF) for node degree $k$, node strength $s$ and link weight $w$.

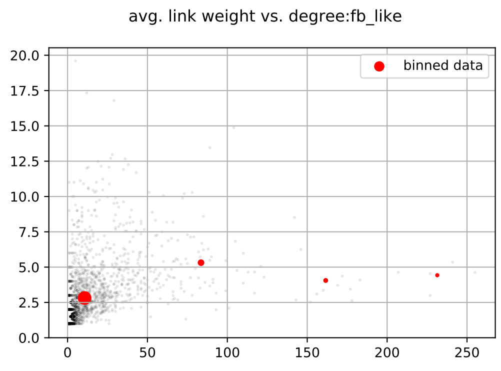
  - Show all three distributions in one plot using loglog-scale.
  - The resulting distributions have *heavy tails*, meaning that the right part of the distributions decays slower than exponentially. To see this, **plot** the CCDFs of exponential distributions having the same mean as each empirical distribution.
  - Based visually on the empirical CCDF plots, roughly **estimate** the 90th percentiles of the degree, strength, and weight distributions. **Explain** also how they can be read off from the plots.

*Hints:*

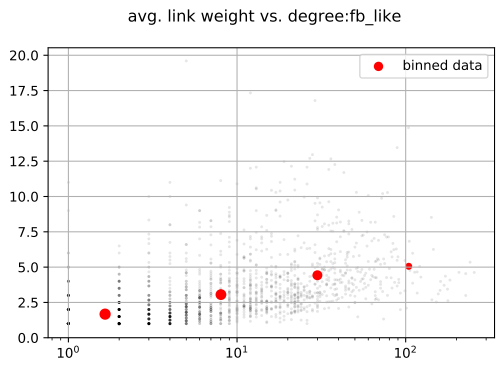
  - See the binning tutorial for help on computing the CCDFs.
  - As a check, calculate the maximum degree $k_{\max}$ and maximum strength $s_{\max}$. If everything is running okay so far, you should be getting $k_{\max} = 255$ and $s_{\max} = 1546$.

b) (5 pts) Next, we will study how the average link weight per node $v = \frac{s}{k}$ behaves as a function of the node degree $k$.

[1]Data originally from `http://toreopsahl.com/datasets/`



(a)



(b)

Figure 1: Example of $v$ as a function of $k$ using another Facebook-like social network data. 1a: linear axes. 1b: logarithmic axes.

– Compute $s$, $k$, and $v$ for each node.
– Make scatter plots of all the data points of $v$ as a function of $k$. Create two versions of the plots: one with linear and one with logarithmic horizontal axes.
– The large variance of the data can make the scatter plots a bit messy. To make the relationship between $v$ and $k$ more visible, **create** bin-averaged versions of the plots, i.e., divide nodes into bins based on their degree and calculate the average $v$ in each bin. Plot the bin-averaged versions on top of the scatter plots.
– Based on the plots, which of the two approaches (linear or logarithmic horizontal axes) suits better for presenting $v$ as a function of $k$? Why?

*Hints:*

– For the bin-averaged plots, use bins that are consistent with the scale of the horizontal axis: bins with constant width for the linear scale and logarithmic bins for the logarithmic scale. If in trouble, see the binning tutorial for help.
– An example of how the scatter and bin-averaged plots may look like is shown in Fig. 1. Obviously, the number of bins are too few in these plots. Typically, it is better to use too many than too few bins. A good choice for the number of bins in this case would be 20.
– Check that each degree bin includes roughly the same number of nodes. Unbalanced distribution of observations may obscure the results.

c) (4 pts) Let's consider a link between nodes $i$ and $j$. For this link, *link neighborhood overlap* $O_{ij}$ is defined as the fraction of common neighbors of $i$ and $j$ out of all their neighbors:

$$O_{ij} = \frac{n_{ij}}{(k_i - 1) + (k_j - 1) - n_{ij}}, \tag{1}$$

where $n_{ij}$ denotes the number of common neighbors.

According to the Granovetter hypothesis, link neighborhood overlap is an increasing function of link weight in social networks. Your task is to find out whether this is the case also for the present data set by visualizing it in an appropriate way. Use the binning strategy (linear or logarithmic) that is most suitable for this case.

- Calculate the link neighborhood overlap for each link.
- Create a scatter plot showing the overlap for every link as a function of link weight.
- As in b), produce also a bin-averaged version of the plot on top of the scatter plot. In doing this, choose a reasonable number of bins, which, in this case, will be between 10 and 30. **Specify** how many bins you used.
- In the end, you should be able to spot a subtle trend in the data. Based on your plot, **show** if the trend is in accordance with the Granovetter hypothesis.

*Hint:* Naive implementation of Equation (1) will result in ZeroDivisionError for some links. Examine for what kind of links this happens. What would be the appropriate way to define $O_{ij}$ for those links?

## 2. Network thresholding and spanning trees: the case of US air traffic (6 pts)

In this exercise, we will get familiar with different approaches to thresholding networks, and also learn how they can be used for efficiently visualizing networks. Now, you are given an undirected network describing the US Air Traffic between 14th and 23rd December 2008[2]. In the network, each node corresponds to an airport and link weights describe the number of flights between the airports during the time period.

The network is given in the file `aggregated_US_air_traffic_network_undir.edg`, and `us_airport_id_info.csv` contains information about names and locations of the airports. The file `US_air_bg.png` contains a map of the US; **you can use the functions provided in the Jupyter notebook template to visualize the network on the map**. In this exercise, you may also freely use all available `networkx` functions.

a) (1 pt) Let us first check the basic network properties to get some ideas on what the network is like. **Compute** the following quantities:

   - Number of nodes $N$, number of links $L$, and density $D$
   - Network diameter $d$
   - Average clustering coefficient $C$

   *Hints:*

   - For the clustering coefficient, consider the unweighted version of the network, where two airports are linked if there is a flight between them.
   - There is no need to report long decimal values with insignificant digits that the code returns. Adequately leave only the significant figures and drop the others.

b) (1 pt) Visualize the full network with all links on top of the map of the US. The resulting figure is somewhat messy due to the large number of visible links.

[2]Data from Bureau of Transportation Statistics

c) (2 pts) In order to reduce the number of plotted links, compute both the *maximal* and *minimal spanning trees* of the network and visualize them. Then, answer the following question:

   – If you would like to understand the overall organization of the air traffic in the US, would you use the minimal or maximal spanning tree? Why?

   *Hint:* For computing minimum spanning trees, use `nx.minimum_spanning_tree`. For computing maximum spanning trees, use `nx.maximum_spanning_tree`.

d) (2 pts) Threshold and visualize the network by taking only the strongest $M$ links into account, where $M = N - 1$ is the number of links in the maximal spanning tree. Then, answer following questions.

   – How many of the links in the thresholded network are the same as those in the maximal spanning tree?
   – Given this number and the visualizations, does simple thresholding yield a similar network as the maximal spanning tree?

   *Hint:* Note that the network is undirected, which means that edge $(i, j)$ is the same as edge $(j, i)$.

## Feedback (1 pt)

To earn one bonus point, give feedback on this exercise set and the corresponding lecture latest two days after the report's submission deadline. You can find the feedback form at the Assignments tab in MyCourses.

## References