Exercise set #4 (26 pts)

- The deadline for handing in your solutions is Dec 5th 2016 23:55.
- 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 tree 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 Python template (`weight_topology_correlations.py`) to get started. `scipy.stats.binned_statistic` function is especially useful throughout this exercise.

a) (3 pts) To gain some idea how the network is like, plot the complementary cumulative distribution (1-CDF) for node degree k, node strength s and link weight w.

   - **Show** all three distributions **in one plot** using loglog-scale.
   - Briefly **describe** the distributions: are they Gaussian, power laws or something else?
   - Based on the plots, roughly **estimate** the 90th percentiles of the degree, strength, and weight distributions.

   Hints:

   - See the binning tutorial for help on computing the 1-CDFs.
   - For reading in the network, use `net = nx.read_weighted_edgelist`
   - For getting node strengths: `strengths = nx.degree(net, weight="weight")`

b) (4 pts) Let us now study the correlations between node strength s and the node degree k. More specifically, we are interested in the average link weight per node $\langle w \rangle = \frac{s}{k}$ and how it behaves as a function of the node degree k.

   Thus, **compute** s, k, and $\langle w \rangle = \frac{s}{k}$ for each node. **Make a scatter plot** that shows $\langle w \rangle$ as a function of k. Do this using both linear and logarithmic x-axes.

   The resulting plots can be a bit messy, so **create** also **a bin-averaged versions** of the plots, *i.e.* calculate the average $\langle w \rangle$ for node groups whose degree lie within some range.

   Hints:

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

- For the linear scale use constant width bins, and for the logarithmic scale use logarithmic bins.
- Use the number of bins you find most reasonable; typically, it is better to use too many than too few bins. In the end, you should be able to spot a trend in the data.

c) (2 pts) Based on the plots created in b), **answer** the following questions questions:

- Which of the two approaches (linear or logarithmic x-axes) suits this for presenting how $\langle w \rangle$ scales as a function of k? Why?
- In social networks, $\langle w \rangle$ typically decreases as a function of the degree due to time constraints required for taking care of social contacts. Are your results in accordance with this conception? If not, how would you explain your finding?

Hint: Remember your results from a): the low number of observations for high values of node degree and link weight may obscure results. Also note that you are dealing with real data that may be noisy. So, interpretation of results may be confusing at first - do not worry!

d) (3 pts) The *link neighborhood overlap* O_{ij} of a link is defined as the fraction of common neighbors of i and j out of all their neighbors:

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

In social networks, this quantity relates to the Granovetter hypothesis, which states that the overlap is an increasing function of link weight. Your task is now to find out whether this is the case also for this data set.

To this end, **calculate the link overlap** for each link. **Create a scatter plot** showing the overlaps as a function of link weight. As in b), **produce** also **a bin-averaged version** of the plot. Use a binning strategy that is most suitable for this case. In the end, you should be able to spot a subtle trend in the data. Based on your plot, **answer** the following questions:

- Is this trend in accordance with the Granovetter hypothesis? If not, how would you explain your findings?

2. Error and attack tolerance of networks (8 pts)

Error and attack tolerance of networks are often characterized using percolation analysis, where links are removed from the network according to different rules. Typically this kind of analyses are performed on infrastructure networks, such as power-grids or road networks. In this exercise, we will apply this idea to the Facebook-like social network used in Ex. 1, and focus on the role of strong and weak links in the network.

Your task is now to remove links (one by one) from the network in the order of

(i) descending link weight (i.e. remove strong links first),

(ii) ascending link weight (i.e. remove weak links first),

(iii) random order

(iv) descending order of edge betweenness centrality (computed for the full network at the beginning).

When doing so, monitor also the *size of the largest component* S as a function of the fraction of removed links $f \in [0, 1]$.

a) (4 pts) **Show** S as a function of f in all four cases **in one plot**. There should be clear differences between all four curves.

Hints:

- In the exercise, `networkx.connected_components(G)` may turn out handy.
- The overall running time of this simulation can take up to a couple of minutes but not orders of magnitudes more.

Based on the plots, **answer** following questions:

b) (2 pt) For which of the four approaches is the network most and least vulnerable? In other words, in which case does the giant component shrink fastest / slowest? Or is this even simple to define?

c) (1 pt) When comparing the removal of links in ascending and descending order strong and weak links first, which ones are more important for the integrity of the network? Why do you think this would be the case?

d) (1 pt) How would you explain the difference between the random removal strategy and the removal in descending order of edge betweenness strategy?

3. 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 a network describing the US Air Traffic between 14th and 23rd December 2008 [1]. 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 data and some code for visualizing the network is provided at the course web-page. 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 `air_traffic_network_base.py` contains a function for visualizing the air transport network, and an example how to use it. You can extend your own work to the same file, or import the file as a Python module. In this exercise, you may also freely use all available `networkx` functions.

a) (1 pt) When facing a new network, it is always good to to first get some idea, how the network is like. Thus, **compute** and list the following basic network properties:

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

     Hint: For the clustering coefficient, consider the undirected and unweighted version of the network, where two airports are linked if there is a flight between them in either direction.

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

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

   - If the connections of Hawai'i are considered, how would you explain the differences between the minimal and maximal spanning trees?
   - If you would like to understand the overall organization of the air traffic in US, would you use the minimal or maximal spanning tree? Why?

d) (2 pts) **Threshold and visualize** the network by taking only the strongest M links into account, where M is the number of links in the MST. Then, **answer** following questions.

   - How many links does the thresholded network share with the maximal spanning tree?
   - Given this number and the visualizations, does simple thresholding yield a similar network as the maximum spanning tree?

Hint: For computing minimum spanning trees, use `nx.minimum_spanning_tree`. Note that you can obtain the maximal spanning tree by computing the minimal spanning tree with negated weights.

Feedback (1 pt)

To earn one bonus point, give feedback on this exercise set and the corresponding lecture latest two day after the report's submission deadline.
Link to the feedback form: `https://goo.gl/forms/2gsQ9HgOga16m8pi1`.

References

[1] [Online]. Available: http://www.rita.dot.gov/bts/