Exercise set #1 (19 pts)

- The deadline for handing in your solutions is September 18th 2023 23:59.
- There are multiple ways for submitting your solutions. Check each question for details.
- Return also one `.zip` file containing all your Python code of the round in MyCourses.
- Check also the course practicalities file in MyCourses for more details on submitting your solutions.

1. Basic network properties (8 pt)

Answer the following questions by taking a quiz in MyCourses.

a) (7 pt) Define the following quantities in words, and calculate them for graph $G = (V, E)$ in Figure 1. Optionally, you may add the definition in mathematical formulas using LaTeX math syntax. In that case, please define/explain all the variables that appear in the formulas.

   - The degree k_i of each node $i \in V$
   - The mean degree $\langle k \rangle$
   - The edge density ρ
   - The diameter d
   - The clustering coefficient c_i for each node $i \in V$ that has degree $k_i > 1$
   - The size of the largest component S
   - The adjacency matrix A

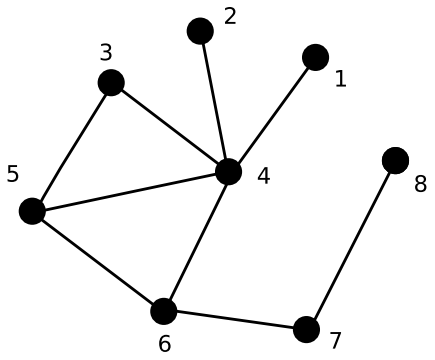


Figure 1: The graph for the exercise.

b) (1 pt) Which of the graphs presented in figure 2 is the subgraph of G induced by nodes 1, 2, 3, and 4?

Aalto University
CS-E5740 Complex Networks, Autumn 2023
Hiraoka, Kivelä, Korhonen
Exercise set #1

2. Computing network properties with NetworkX (6 pts)

In this exercise, you will get hands-on experience with NetworkX by calculating basic network properties. The dataset we use here is the coappearance network of characters in the famous novel *Les Miserables*. The dataset edge list file (`les_miserables_edge_file.edg`) can be found on the course MyCourses page and in the coursedata directory on JupyterHub.

To get you started, you may use the Jupyter Notebook code template available at JupyterHub. If using the template, you only need to fill in the required functions. Some of the functions do not need modifications.

Figures can be saved to the same directory as this Notebook (or any directory in your environment) using Matplotlib's `savefig` function. (No need for taking screenshots!)

Submit your solutions by taking a quiz in MyCourses. Further, return the code you used. Compress all the scripts/notebook files for the exercise round in a single zip file.

Check also the NetworkX online tutorial and index:
`https://networkx.github.io/documentation/stable/tutorial.html`
`https://networkx.github.io/documentation/stable/reference/index.html`

a) (1 pt) Load the edge list and visualize the network.

b) (1 pt) Calculate the edge density of the Les Miserables network. First, write your own code without using `nx.density` and then compare your result to the output of `nx.density` (the corresponding NetworkX function). Report up to three digits.

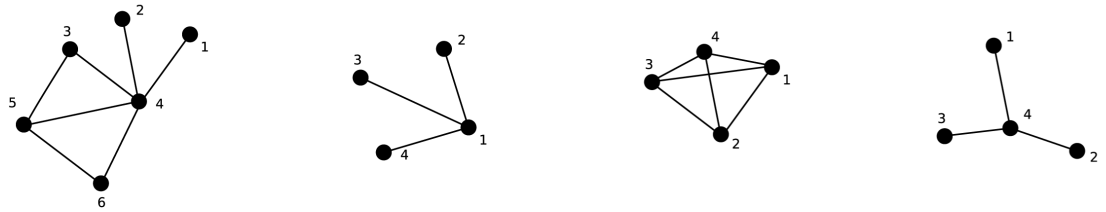
c) (1 pt) Calculate the average shortest path length $\langle l \rangle$ of the network using the relevant ready-made NetworkX function. Report up to three digits.

d) (1 pt) Calculate the average clustering coefficient of the network using the relevant ready-made NetworkX function. Report up to three digits.

e) (2 pt) Calculate the degree distribution $P(k)$ and the complementary cumulative degree distribution CCDF(k) of the network. Plot the distributions using Matplotlib.

Hint: CCDF(k) is defined as the probability that a randomly picked node has a degree larger than or equal to k.

Figure 2: Answer options for problem 1.1 b)



(a) (b) (c) (d)

3. Challenge exercise: Counting number of walks using the adjacency matrix (5 pts) (pen and paper)

Submit your solution as a pdf file in MyCourses.

Many network properties can be computed from the adjacency matrix. In this exercise, we investigate the relationship between the powers of the adjacency matrix and the number of walks between pairs of nodes.

a) Draw the *induced subgraph* G^* that is induced by vertices $V^* = \{1 \dots 4\}$ of network visualized in Figure 1. Calculate by hand the number of walks of length two between all node pairs (i, j), $i, j \in \{1, \dots, 4\}$ in G^*. The length of a walk is defined as the number of links traveled to get from i to j; a link can be traveled in both directions and the walk can visit a node multiple times. Remember to consider also walks, where $i = j$.

   Then, compute the matrix A^2 (you may do this also using a computer), where A is the adjacency matrix of the network G^*. Compare your results; what do you notice?

b) Compute the number of walks of length three from node 3 to node 4 in G^*. Then, starting from matrices A^2 and A, compute by hand the value of $(A^3)_{3,4}$ showing also the intermediate steps for computing the matrix element.

c) Now, let's consider a general network with adjacency matrix A. Show that the element $(A^m)_{i,j}$, $m \in \mathbb{N}$ corresponds to the number of walks of length m between nodes i and j.

   Hint: Make use of mathematical induction: Show first that the statement holds for $m = 1$ by analyzing the elements of the matrix A^1. Next, assume that the statement holds for a general m and prove that it holds also for $m + 1$. To do that, consider the element $a_{i,j}^{(m+1)} (= (A^{m+1})_{i,j})$ assuming that $a_{i,j}^{(m)}$ gives the number of walks of length m.

Feedback (1 pt)

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

References