Homework #0

This homework is easy and is meant to get you started with network analysis. Although the homework will not be graded, it will still be taken into account for course participation. All questions and comments regarding the homework should be directed to Piazza.

Submission details

This homework is due on **February 27th** at 3:00pm, while late days expire on **March 2nd** at 10:00am. The homework must be submitted as a hard-copy in the submission box in front of R 2.49 and also as an electronic version to eUcilnica. It can be prepared in either English or Slovene and either written by hand or typed on a computer. The hard-copy should include (1) this cover sheet with filled out time of the submission and signed honor code, (2) short answers to the questions, which can also demand proofs, tables, plots, diagrams and other, and (3) a printout of all the code required to complete the exercises. The electronic submission should include only (1) answers to the questions in a single file and (2) all the code in a format of the specific programming language. Note that hard-copies will be graded, while electronic submissions will be used for plagiarism detection. The homework is considered submitted only when both versions have been submitted. Failing to include this honor code in the submission will result in **10% deduction**. Failing to submit all the developed code to eUcilnica will result in **50% deduction**.

Honor code

The students are strongly encouraged to discuss the homework with other classmates and form study groups. Yet, each student must then solve the homework by herself or himself without the help of others and should be able to redo the homework at a later time. In other words, the students are encouraged to collaborate, but should not copy from one another. Referring to any solutions obtained from classmates, course books, previous years, found online or other, is considered an honor code violation. Also, stating any part of the solutions in class or on Piazza is considered an honor code violation. Finally, failing to name the correct study group members, or filling out the wrong date or time of the submission, is also considered an honor code violation. Honor code violation will not be tolerated. Any student violating the honor code will be reported to **faculty disciplinary committee** and vice dean for education.

Name & SID: ____

Study group: ____

Date & time: ______________________

I acknowledge and accept the honor code.

Signature: ______________________

1 Network software (4 points)

Throughout the course you will be analyzing networks. This can be done using either some programming library or a software tool. As you will be required to analyze large networks and also implement some algorithms on your own, you should first select some network library. However, at least in the case of small networks, it is commonly beneficial to be able to get a quick insight into the network structure just by visualizing it. For this purpose, you should also select some network software. Below is a list of most popular network libraries and software. For most of them you will be able to find tutorials, documentation, code examples, discussion groups and other online. It is strongly recommended that you select a library for the programming language you are most comfortable with. Note also that your selection is not obligatory and you can switch to any other library or tool during the semester.

SNAP C/C++ library for network analysis

Snap.py Python interface for SNAP library

NetworkX Python library for network analysis and visualization

graph-tool Python library for network analysis and visualization

igraph R, Python and C/C++ library for network analysis and visualization

JGraphT Java library for network analysis and visualization

JUNG Java library for network analysis and visualization

LNA Java library for network analysis (available upon request)

Most popular network software is Gephi, Pajek, visone, Graphviz and UCINET.

Here is the Zachary karate club network in edge list, Pajek and LNA formats (Figure 1). This is a simple undirected network. Using your programming library, load the network and print out the number of nodes and links in the network. Using your software tool, try to neatly visualize the network and export or print-screen the figure.

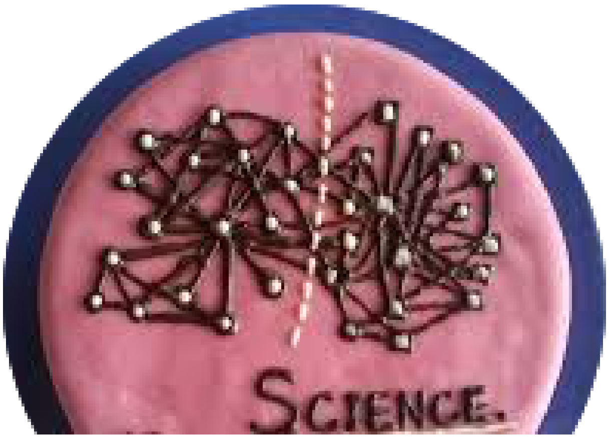

Figure 1: **Zachary karate club network cake**

What to submit?

Name selected network programming library and software tool. Very briefly explain your choice. State the number of nodes and links in the karate club network, and print out all code you have used (2 points). Include a visualization of the network (2 points).

2 Network collection (4 points)

Within the course you will be analyzing randomly generated synthetic graphs and real-world networks. Prior to the analysis of real-world networks, one must first collect the network and clean it. Contrary to what you might expect, often more time is spent on collecting the network than on analyzing it afterwards.

2.1 Your own network

Map out a network on your own. Try to be creative and collect a network that you have not seen in lectures or labs. For instance, see `https://podatki.gov.si`. The network can be anything, but since you will be analyzing it below, try to make it interesting. You might also use it for your course project. Compute the number of nodes and links, and the average degree in your network using either your network library or software tool.

2.2 Synthetic random graphs

Any network library or tool surely includes an option to generate the simplest possible synthetic graph called the Erdős-Rényi random graph [ER59]. Create a random graph with the same number of nodes as in your network and set any other required parameter thus to best fit your network. Compute the number of nodes and links, and the average degree in a generated random graph.

What to submit?

2.1 Briefly explain your network. What are the nodes and what are the links? State the number of nodes and links, and the average degree in your network (2 points). Print out any code you might have used.

2.2 How did you set the random graph parameters? State the number of nodes and links, and the average degree in a generated random graph (2 points). Print out any code you might have used.

3 Network analysis (4 points)

Commonly not all nodes in a network are equally important (Figure 2). PageRank [BP98] is probably the most well known measure of node importance and it should be included in any serious network library (as the Erdős-Rényi random graph above). PageRank is actually the algorithm that made Google famous.

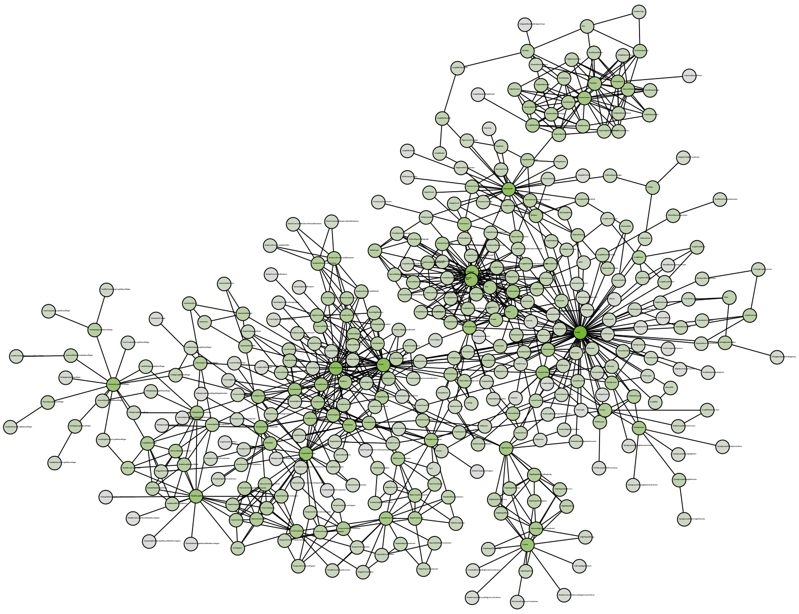

Figure 2: **JUNG software network with node colors corresponding to PageRank**

3.1 Google's PageRank algorithm

Using your library compute the PageRank score of all the nodes in your network and print out the labels and scores of ten nodes with the highest PageRank score. Which nodes are the most important according to PageRank and how would you interpret these results? Is the difference between the first and the tenth node substantial or are all ten nodes equally important?

3.2 Comparison with random graphs

Repeat the analysis for a generated Erdős-Rényi random graph with the same parameters as before and print out the scores of ten most important nodes according to PageRank. How do the PageRank scores compare to those obtained for your network? Is the difference between the first and the tenth node substantial or are all ten nodes somewhat equally important?

What to submit?

3.1 Print out the labels and scores of ten most important nodes in your network, and all the code you have used (1 point). Provide brief answers to both questions (1 point).

3.2 Print out the scores of ten most important nodes in a generated random graph, and all the code you have used (1 point). Provide brief answers to both questions (1 point).

References

[BP98] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Comput. Networks ISDN*, 30(1-7):107–117, 1998.

[ER59] P. Erdős and A. Rényi. On random graphs I. *Publ. Math. Debrecen*, 6:290–297, 1959.