PROJECTS DESCRIPTIONS FOR THE NETWORKS COURSE

General description of the projects

The term projects are aimed at providing a test ground for the skills and knowledge you picked up during the course. To fulfil a project you have to go through basically the following steps:

- understand the problem posed, (i.e., dig up the slides from the course related to the problem, look up in Wikipedia, read the relevant research papers, etc.),
- write/download and assemble the program(s) needed to carry out the project,
- calibrate and run your programs: in some cases parameters have to be tuned, in other cases the same routine has to be re-run several times with different parameters, etc.
- prepare figures showing the results,
- and finally write a short report about the project. In case you choose to work in ipython jupyter environment, a notebook with markdown fields describing the problem and the progress together with well prepared figures is a simple solution for that.

For most of the projects you are supposed to test/run on real networks, for that, input data can be downloaded e.g., from `https://icon.colorado.edu`. Although for the first tests of your code you may want to use small networks, for the final version of the project presentation please use networks having at least 1000 (but preferably 10000 or even more) nodes.

The problems

1. Write a program that can randomize a network by preserving the degree sequence and enhancing or decreasing the frequency of a chosen motif consisting of 3 nodes. (The user should be able to change the

motif and to decide whether its frequency should be increased or decreased).

Demonstrate how the program works on a couple of real networks: show a few figures of the networks before and after the randomization, and show figures of the changing of network parameters such as the average clustering coefficient, assortativity coefficient, average shortest path length, etc., during the randomization.

2. Compare random graphs generated with the Holme-Kim model and with the configuration model. Set the number of nodes and links to be the same, and also the degree distribution to be the same. Is there any difference between the behavior of the clustering coefficient, the behavior of the average nearest neighbors degree, and the closeness? Show pictures of the networks generated by the two different algorithm, and also present figures showing the examined statistics (e.g., the clustering coefficient as a function of degree, the $k_n n$ as a function of degree, etc.).

3. Study the percolation transition of the E-R graph numerically. How does the size of the largest connected component scale with the system size at the critical point? How does the size of the largest connected component grow with p at the critical point? Show pictures of the generated graphs both below and above the critical point, and also figures revealing the scaling of 2the largest connected component at p_c.

4. Study epidemics numerically in the SIS model on computer generated directed networks. What is the difference between spreading on Erdős-Rényi, and on scale-free networks? In case of scale-free networks, examine whether it makes a difference if only the in- (or out-) degree distribution is scale-free?

   Reproduce the main figures from the following paper: Romualdo Pastor-Satorras, Alessandro Vespignani, "Epidemic dynamics and endemic states in complex networks", *Phys. Rev. E* **63**, 066117 (2001). (free download from `https://arxiv.org/abs/cond-mat/0102028`)

5. Study the resilience against random breakdown of real networks! Apply both a random node removal- and a random link removal process separately, and compare the results. Analyse the problem of resilience on the corresponding random graphs as well. Use an ensemble of E-R

graphs and also randomized networks with preserved degree distribution.

Plot the relative size of the largest connected component as a function of the removed fraction of nodes (or links). Since we are interested in whether we see a phase transition like disassembly of the network or not, plot also the quantity corresponding to the susceptibility, $\chi = \sum_\alpha \frac{N_\alpha^2}{N^2}$, where α runs over the connected components except for the largest one, and N_α denotes the number of nodes in the given component.

6. Study the resilience against targeted attack of real networks! Apply removal processes where the removal order of the nodes depends on different centralities: the degree, the betweenness, PageRank, etc., and compare the results. Analyse the problem of resilience on the corresponding random graphs as well. Use an ensemble of E-R graphs and also randomized networks with preserved degree distribution.

   Plot the relative size of the largest connected component as a function of the removed fraction of nodes. Since we are interested in whether we see a phase transition like disassembly of the network or not, plot also the quantity corresponding to the susceptibility, $\chi = \sum_\alpha \frac{N_\alpha^2}{N^2}$, where α runs over the connected components except for the largest one, and N_α denotes the number of nodes in the given component.

7. Study random walks on Watts-Strogatz networks. Generate large W-S graphs with fixed N and varying rewiring probability β, and record the number of visited nodes s for a random walker as a function of the time-steps. On a linear chain we would expect $s(t) \sim \sqrt{t}$, whereas in an E-R random-graph $s(t) \sim t$, (at least as long as $s << N$). Assume that in case of a W-S graph we have $s(t) = \sqrt{t} f(t\beta^\alpha)$, where $f(x)$ is some universal function, and α is a yet unknown exponent. Plot $s(t)/\sqrt{t}$ as a function of $t\beta^\alpha$, and find the α exponent for which the datapoints collapse onto the universal $f(x)$ function.

8. Compare different community finding algorithms on the "classical" computer generated test bed and on a few real networks.

   Details:

   - The computer generated test bed is defined as follows: consider 4 groups of nodes with 32 members each, these correspond to the communities the algorithms should find. The links are drawn

between the nodes at random, however the average number of links going from one node to other nodes in its own community is given by z_{in}, whereas the average number of links going to the rest of the network is given by z_{out}, and z_{in} and z_{out} always add up to $z_{\text{in}} + z_{\text{out}} = \langle k \rangle = 16$. When z_{out} is small and z_{in} is close to 16, the algorithms have an easy job, the communities can be spotted even by eye, since they the links are very dense inside and occur very rarely in between the communities. However, as z_{out} is increased the communities become less and less well defined, and around $z_{\text{out}} = 8$ the graph becomes homogeneous.

- The evaluation of the community finding results is based on a matching between the predefined 4 groups and the found communities. Choose the matching which gives the highest number of correctly classified nodes. (E.g., index the original groups by A, B, C, D, then index the found communities also with A, B, C, D, and search for the permutation of these indices maximizing the number of nodes on which the two community index is the same).
- Plot the ratio of correctly classified nodes over the total number of nodes as a function of z_{out}. Also show figures about the found communities in the test graph and in the real networks.

9. Study the extent of hierarchy in random graphs. Hierarchical organization is a ubiquitous feature of a wide range of systems from cells through living organizations to animal flocks and the human society. There were several different hierarchy measures introduced in the scientific literature for quantifying the extent of hierarchy in the structure of directed networks. The task is to examine the behavior of a couple of these measures around the critical point of percolation in the directed Erdős–Rényi graph.

   The directed E-R graph is very similar to the undirected one, i.e., directed links are introduced independently with probability p between N nodes, (where we consider both directions for every pair). If p is very low, the obtained directed random graph is dispersed and is consisting of many disconnected components, whereas if p is large enough, a giant (weakly connected) component is emerging, which will eventually "swallow" all smaller component if p is increased further.

   The task is to monitor the value of three hierarchy measures as functions of the average degree around the critical point of the percolation using numerical simulations. Since the structure of the giant com-

ponent is tree-like in the undirected E-R graph at the critical point, and trees are highly hierarchical objects, it is an interesting question whether we observe any peaks in the hierarchy measures for the directed E-R graph at the percolation transition? The hierarchy measures to be studied are the following:

- the fraction of links not participating in any directed cycles,
- the Global Reaching Centrality introduced in E. Mones, L. Vicsek and T. Vicsek, Hierarchy measure for complex networks. *PLoS ONE* **7**, e33799 (2012), available from `https://doi.org/10.1371/journal.pone.0033799`

10. The same as the previous problem, but instead of the Global Reaching Centrality, use the Random Walk Hierarchy introduced in D. Czégel and G. Palla, Random walk hierarchy measure: What is more hierarchical, a chain, a tree or a star? *Scientific Reports* **5**, 17994 (2015), available from `https://www.nature.com/articles/srep17994.pdf`.

11. There are several indications that hidden metric spaces might exist be- hind the often observed general properties of real networks. A remark- able network model exploiting this idea is the Popularity-Similarity- Optimization (PSO) model, where nodes are placed one by one on the Poincaré disk representation of the 2D hyperbolic plane with a logarithmically increasing radial coordinate and a random angular co- ordinate, and links are drawn with probabilities determined by the hy- perbolic distance between the node pairs. In vague terms, the degree of nodes is determined by their radial coordinate (lower distance from the origin corresponds to larger degree), and the angular proximity of the nodes can be interpreted as a sort of similarity, where more simi- lar nodes have a higher probability to be connected. Random graphs generated this way usualy exhibit a scale-free degree distribution, high clustering coefficient and the small world property at the same time. The goal of the project is to implement the PSO model on the com- puter based on the original paper by F. Papadopoulos et al, `https://doi.org/10.1038/nature11459` (A free version is available on the arXiv: `https://arxiv.org/abs/1106.0286`) Generate random graphs according to this approach (up to a size with at least 1000 nodes) and study their properties, and prepare plots proving that they are indeed small world, highly clustered and scale- free. Provide layouts of the obtained graphs both on the Poincaré disk and using Cytoscape.

12. Networks (apart from transportation systems such as road networks) usually do not represent and are not part of any Euclidean space, making tasks like node clustering or path finding significantly harder compared to e.g., lattices. Graph embedding algorithms aim to find a suitable representation (d-dimensional vectors) for the nodes of a complex network that preserves best the original relations (e.g., neighborhoods), and basically map the given network onto a set of points (vectors) in a d-dimensional Euclidean space. One of these algorithms is node2vec (`https://doi.org/10.1145/2939672.2939754`), which explores the graph via guided random walks, and optimizes the embedding based on the groups of nodes that were seen close together. It also allows changing the random walk strategy (between Breadth First Search and Depth First Search) via two parameters. One implementation can be found here: `https://github.com/aditya-grover/node2vec`. Explore the algorithm on small graphs (you can do the embedding in 2 dimensions, or use t-SNE or UMAP, to visualize the embedded vectors for higher dimensional embeddings).

    The main task of the project is to implement clustering based on the embedding. The basic idea is to group nodes together based on the Euclidean distance between the points (vectors) they are mapped onto. However, any other distance metric isalso possible, e.g. cosine distance. You can use K-Means (like the authors), or any other clustering algorithms, e.g. DBSCAN or hierarchical clustering (implementations can be found in the scikit-learn package). Reproduce the figure with the two types of clustering done on "Les Miserables co-appearances" described in the original paper (figure 3.), and summarize the results. Analyze how node2vec performs on the "classical" computer-generated test-bed for community finding (described in project 8.). Which node2vec parameters worked best for community finding? What type of random walk do they correspond to?

13. The same as the previous project, except that the main task is link prediction. Usually, real-world networks are not fully observed (links can be missing), and the prediction of these unobserved connections is an important task from the point of view of practical applications. In addition, networks can also evolve over time, and a similar task is to predict where the new links are most likely to appear in the future. In both cases, providing a list of node pairs that should be (or that is likely to become) connected is the central goal. A notable approach to link prediction is to use graph embeddings, with a fitted logistic

regression.

As a first step in the project, create three test networks of 100-200 nodes, with different assortativity: one highly assortative, one highly disassortative, and one with no degree correlations. Remove a portion of the links (10-20%, taking care that the network remains connected), and use node2vec to obtain the embeddings for the networks with missing links. Fit a logistic regression which is able to decide which embedding vector pairs were actually linked in the original version of the test network, and which were not. The train set should consist of equal amounts of node pairs corresponding to edges and non-edges. Try to rank all node-pairs based on which one should have edge with highest probability. The goal is to predict most of the removed links this way. Good performance metrics are AUROC (area under receiver operating characteristic curve) and AP (average precision) scores. Find the node2vec parameters which work best for each test graph, and try to explain why. You can also test link prediction on real-world networks in a similar fashion, by first removing a small portion of the links, and then trying to predict them back based on the embedding.

14. The same as the previous 2 projects, but the main task is graph reconstruction. It is important for an embedding to accurately encode the original network structure using as few embedding dimensions as possible. Choose a not too big (¡1000 nodes) real network, e.g "Word adjacencies of David Copperfield" and obtain the embedding using the whole graph. The reconstruction is usually done via logistic regression. Assemble a train and a test set of node pairs: the train set should consist of 80-90% of the edges, and equal amounts of not connected node pairs; the test set should contain the remaining 10-20% edges, and a realistic amount of negative samples (i.e. the proportion of edges to non-edges should match the original graph). Fit a logistic regression on the train set, and measure the AUROC (area under receiver operating characteristic curve) and AP (average precision) scores on the test set. Do this in a cross-validated fashion to obtain results on the whole graph. Analyze how increasing/decreasing the number of embedding dimensions effect the reconstruction performance. Is it also sensitive to other node2vec parameters? You can also create E-R random graphs with same number of nodes and edges, and compare the reconstruction performances. Which graph was easier to reconstruct?

15. The Granovetter hypothesis assumes that the function of strong and

weak ties are quite different in social networks: strong ties usually occur within groups or communities and their role is to hold the given cluster together, whereas weak ties usually appear in between communities, and their role is to hold the different groups (and thereby, the whole system) together. An interesting related empirical study on a large phone call network is presented in the following paper: `https://www.pnas.org/content/pnas/104/18/7332.full.pdf` , where it is shown that the critical point of the percolation is at different value of the fraction of the removed links if we remove the links in descending or in increasing order of the link weights. Furthermore, a quantity O measuring the overlap between the neighbourhoods of the link endpoints is also introduced, and again, the disassembly of the giant component into small fragmented components is different if we remove the links according to descending or increasing order given by O. The main aim of this project is to repeat the link removal experiments shown in the paper on the weighted co-authorship network between scientists that can be downloaded from `http://www-personal.umich.edu/~mejn/netdata/netscience.zip` Do you observe a difference in the percolation transition under link removal if you remove links according to increasing/decreasing order given by the link weights? What happens if you remove according to increasing/decreasing order given by the O value specified in the paper by J.P. Onnela et al.? Prepare the usual plots showing the relative size of the largest component as a function of the fraction of the removed links, and plot the susceptibility as well.

16. An interesting generalisation of the configuration model is given by the edge-coloured configuration model, where links are allowed to take colours, and instead of the degree, the "state" of a node is described by a list of integers (a vector), giving the number of the links of the different colours connected to the node (in the same fashion as the degree gives the number of links connected to the node in the original configuration model). Networks generated in this framework can show very interesting percolation behaviour as described in `https://www.nature.com/articles/s41467-018-08009-9` Since this paper is relying on quite advanced mathematical tools that are beyond the scope of the present course, full understanding of the theoretical derivations is not required. The main goal of the project is to implement the edge-coloured configuration model as described in the paper, allowing the generation of edge-coloured random graphs. Demonstrate the abilities of your framework by reproducing the nu-

merical results shown in Fig.3 in the paper for networks specified by Eq.(33). In addition, make illustrations similar to Fig.4a, and try to numerically reproduce the results of the percolation analysis shown in Figs.4e-4f.

17. Wikipedia is an open source knowledge collection, where anyone from the world can contribute to any topic by writing or improving an article, which explains a word or phrase. Wikipedia tracks each contribution by logging the name of the editor (if logged in) or the IP address from where the author worked besides the exact time of the contribution. Typically, an article results as a cooperative work of several authors. Each wikipedia article is labelled with one or more categories. Categories are keyword like tags that are organised into a taxonomy (hierarchical relations from specific to more general words). This project aims to discover the groups of editors in Wikipedia, who are related to each other by contributing to articles. So two editors are called to be in a coauthorship relation, if they contributed to at least one common wikipedia article. By creating a network of authors and by running a community finding algorithm on this network you will find several groups. If you restrict the coauthorship relation to a time interval (e.g. you consider coauthors only if they edited the same article in a given year), and create the coauthorship network for all available years, then you will get a series of networks. In each of these networks you can find groups. By finding common members between the groups, you can track the evolution of these groups. The groups can be characterised by the categories of the articles that were coauthored of the members of the groups. This way you can find the evolution of groups of editors and see how they have moved from one topic to another one, and you will see some editors who have changed their group membership during the time, while other ones remained in the same group and kept working on the same field.

    You can find further hints in the manuscript: `https://arxiv.org/abs/1605.00509`

    Create a plot for the size distribution of the groups. Select some interesting groups, which are either stable (the members do not change or the change is relatively small. Find some examples for the opposite. Try different clustering algorithms, and interpret the differences between the results.

18. The European Commission supports research and innovation through

specific programmes. These programmes target some actual problems and try to connect researchers and research centres in Europe. The main descriptive data of these programmes are available for exploration through the EU Open Data Portal. Here, data are available in different formats. An interesting representation of these data is the RDF form, where related pairs of entities are linked with a descriptive connection. In this project you should explore the Data Portal of the EU Commission and build networks with SPARQL queries. You can extend the information with other formats as well, but the main skeleton should be achieved with SPARQL. After composing the appropriate queries, and building the networks, you should be able to answer the following questions:

- Which Universities were the most successful cooperators in the last 10 years?
- Which research fields created the largest and most dense cooperation networks?
- By identifying the most supported researchers (principal investigators) and exploring the coauthorship networks of the publications from the resulting articles of the EU programmes, is it possible to predict the members of consortia of the current supporting scheme?

You can start your data collection from the Cordis website: `https://data.europa.eu/euodp/en/linked-data` `https://data.europa.eu/euodp/data/dataset/cordisfp7projects`

Create a plot of the collected networks, try to embed them into a geographical map. In another figure visualise the cooperation networks for some selected research fields. As a third option, try to build a taxonomy (hierarchy) of the research fields. Finally, visualise the predicted new project leaders!

19. Imagine, that a group of travelling salesman wants to visit all public transport stations in Budapest. They want to accomplish this task as quickly as possible. So they hook up the map of Budapest, locate the stations and start to plan their visits. They can use only public transport to reach the stations except in the first step, where they can start at any station. In the first planning round they build the network of the stations by connecting that ones where at least one public transport line connects the two stations. On this networks they

try to find the optimal starting nodes (stations) from which they can reach all nodes with the minimal number of steps. The steps are counted by the number of stations that they enter or travel through as they use the public transport lines. Depending on the size of the group (number of salesmen in the group) try to make a station-visiting plan, which results the minimal number of steps, which is needed to visit all the stations for the group! Try to minimise to sum of the steps taken by all members of the group, and try to minimise the number of steps, which is taken by the member who makes to most steps in the group!

Compare your result with a reshuffled network!

The time table and the public transport database for Budapest is available from the BKK website: `https://bkk.hu/apps/gtfs/` `https://bkk.hu/en/developers/`

20. The same as the previous project, but here try to minimise the time, which is needed for the group! Here you should take into account the timetable as well for the lines! A salesman visits a station by getting off the vehicle and staying at least 5 seconds on the station. Note, that in this case you should play with the optimal starting time as well!

    Compare your results with a randomised version of timetable, where you randomly redistribute the starting time on the nodes but keeping the time difference statistics intact.

    In this reference work you find a good description of the time randomisation: `https://arxiv.org/pdf/1006.2125.pdf`

    The time table and the public transport database for Budapest is available from the BKK website: `https://bkk.hu/apps/gtfs/` `https://bkk.hu/en/developers/`