Instituto Superior Técnico | Universidade de Lisboa
Network Science 2021/22
Problem Set 4

1. What is the relationship between percolation theory and analysis of network robustness?

2. Briefly explain why scale-free networks are robust against random failures.

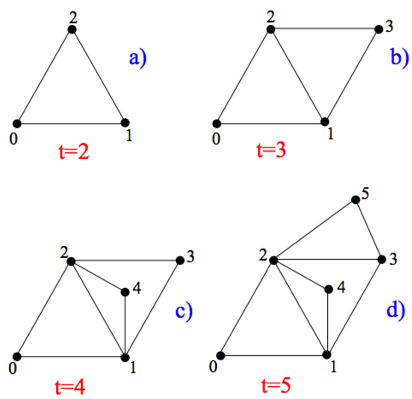
3. The analysis of robustness of networks show that each complex system has its own *Achilles' Heel*. What is the *Achilles' Heel* of scale-free networks?

4. [Conspiracy theory[1] (involves some programming)] In a Big Brother society, the thought police want to follow a "divide and conquer" strategy by fragmenting the social network into isolated components. You belong to the resistance and want to foil their plans. There are rumors that the police want to detain individuals that have many friends and individuals whose friends tend to know each other. The resistance puts you in charge to decide which individuals to protect: those whose friendship circle is highly interconnected or those with many friends. To decide you simulate two different attacks on your network, by removing

   - the nodes that have the highest clustering coefficient
   - the nodes that have the largest degree.

   Study the size of the giant component in function of the fraction of removed nodes for the two attacks on the following networks:

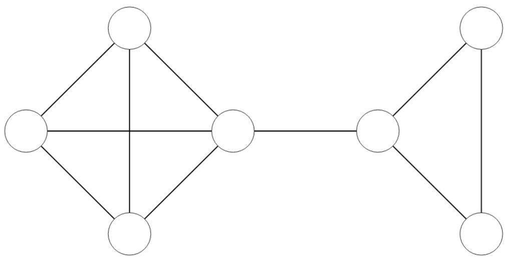
   a) A network with $N = 10^4$ nodes and $<k>=4$ generated with the Minimal model (*aka* link selection model, see figure).
   b) A network with $N = 10^4$ nodes and $<k>=4$ generated with the Barabási-Albert model.



   Which is the most sensitive topological information, clustering coefficient or degree, which, if protected, limits the damage best? Would it be better if all individuals' information (clustering coefficient, degree, etc.) could be kept secret? Why?

5. (Community structure) Briefly explain the Newman-Girvan (NG) algorithm for community finding.

6. Let us resort to the Newman-Girvan algorithm with edge betweenness to find the communities of the following graph. Compute the *modularity* obtained before and after the first iteration of the Newman-Girvan algorithm.



7. (Community structure) Gephi implements the Louvain method. Briefly explain the idea of the method and follow a quick introduction (~10min) (URL: https://www.youtube.com/watch?v=7LMnpM0p4cM) on how to deal with communities in Gephi. Following this video, use Gephi to analyze the community structure of your favorite dataset, exploring different levels of community resolution.

[1] Adapted from A. L. Barabási, Network Science, 2016.

Solutions:

. 1. Percolation theory is a highly developed subfield of mathematics and physics. A typical problem addressed by it is to consider a graph, where we place pebbles with probability *p* at each node. Neighboring pebbles are considered connected, forming clusters. Given that the position of each pebble is decided by chance one may ask what is the expected size of the largest cluster (giant component) or what is the average cluster size. Typically for small $p<p_{critical}$ clusters are small and isolated. For $p>p_{critical}$ pebbles become connected in a single cluster and a large giant component emerges. The question is often to determine the value $p_{critical}$ for a given network. Similarly, **network breakdown can be seen as an Inverse Percolation problem.** Instead of randomly occupy vertices with a probability *p*, we randomly simulate the failure of each node with a probability f=1-p, and all results of percolation theory can still be used.

. 2. In scale-free networks, by randomly removing nodes we are likely hitting low-degree nodes with low centrality. This an intuitive explanation for the robustness of scale-free networks. Being more precise, the critical fraction f above which the giant component disappears is given by the Molloy-Reed criterion. It states that the giant component exists if the fraction f of removed nodes is $f < f_c = 1 - \frac{1}{\frac{\langle k^2 \rangle}{\langle k \rangle} - 1}$. For scale-free networks with a degree distribution $P(k) \sim k^{-\gamma}$ with $\gamma \leq 3$, the second moment diverges ($\langle k^2 \rangle \to \infty$), such that the f_c = 1. In other words, **to fragment a scale-free network we must remove all its nodes**!! Please note that $\langle k^2 \rangle \to \infty$ when $N \to \infty$. Thus, for finite networks we always have f_c<1, yet close to 1.

. 3. The important role hubs play in holding together a scale-free network, provides its Achilles' heel. If we create a targeted attack, first removing the highest degree node, followed by the node with the next highest degree and so on, we will rapidly fragment our network. Moreover, one can show that the observed f_c is remarkably low, indicating that we need to remove only a tiny fraction of the hubs to cripple the network.

. 4. Try to implement this simulation using your favorite network framework.

. 5. The NG algorithm is a divisive procedure where we systematically remove the links connecting nodes that belong to different communities, eventually breaking a network into isolated communities. At each time-step 1) we remove the link with highest centrality (for instance, link betweeness). 2) recalculate the centrality of each link for the altered network. Repeat these 2 steps until all links are removed. Choose the partition of communities giving the highest modularity, *M*.

. 6. The modularity of a network with *E* edges is given by $M = \sum_{r=1}^{n} \left[\frac{E_r}{E} - \left(\frac{k_r}{2E} \right)^2 \right]$, where E_r is the total number of links within the community *r* and k_r is the total degree of the nodes in this community. In the beginning all nodes belong to the same community. Thus, $M_0 = 1 - \left(\frac{4 \times 3 + 2 \times 2 + 4}{2 \times 10} \right)^2 = 0$. In the first iteration clearly the link with highest betweeness centrality will be the link connecting the right-hand and left-hand clicks. If one removes this link we get $M_1 = \frac{6}{10} - \left(\frac{12}{20} \right)^2 + \frac{3}{10} - \left(\frac{6}{20} \right)^2 = 0.45 > M_0$.

. 7. The Louvain algorithm consists in a modularity optimization algorithm (as the greedy algorithm), yet with a better scalability. It consists of two steps that are repeated iteratively. In the first step, modularity is optimized by local changes. We choose a node and calculate the change in modularity, if the node joins the community of its immediate neighbors. This process is repeated for each node. In the second step, nodes belonging to the same community are merged into a single node, building a new network of communities. Once the second step is completed, we repeat the same sequence of steps calling their combination a *pass*. The number of communities decreases with each pass. Passes are repeated until there are no more changes and maximum modularity is attained.

.

.

.