Community detection and k-cores decomposition

You are given three small **social networks with known sociological partitioning** of nodes.

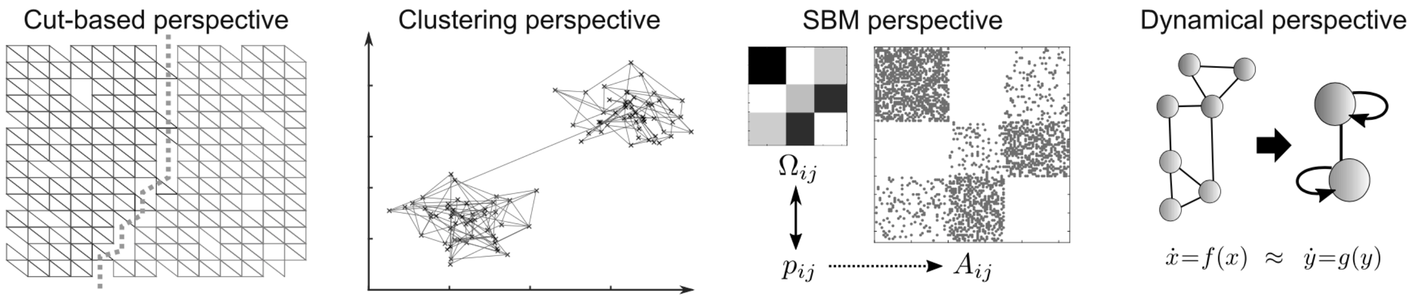
- Zachary's karate club network (2 clusters)
- Davis's southern women network (2 to 4 clusters)
- Lusseau's bottlenose dolphins network (2 clusters)

Furthermore, you will be studying the following larger **real networks with node labels**.

- Game of Thrones character appearance network (characters)
- Human diseasome network by common symptoms (diseases)
- Conflicts and alliances between world nations (countries)
- US airplane traffic transport network (airports)
- Java software class dependency network (classes)
- IMDb actors collaboration network (actors)
- WikiLeaks cable reference network (cables)

All networks are available in Pajek, edge list and LNA formats.

Browse CDlib programming library for implementations of **network community detection and graph partitioning algorithms**. Select an algorithm which you will be using for the exercises below. For instance, select one of the most popular algorithms like optimization of modularity known as Louvain algorithm, map equation algorithm called Infomap, simple label propagation algorithm, hierarchical clustering based on edge betweenness, stochastic block models minimizing description length etc.



I. Social networks with known sociological partitioning

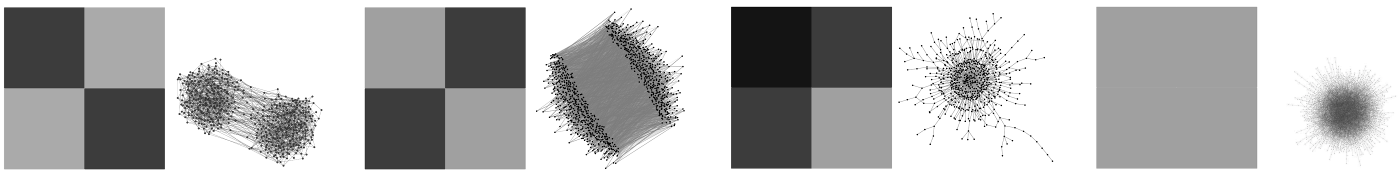
Apply selected algorithm to **small social networks** above and test whether the **revealed communities coincide with known sociological partitioning** of these networks. Ideally, you should apply the algorithm to each network multiple times and compare partitions using some standard measure like normalized mutual information or adjusted Rand index. Since these networks are very small, you can also visualize the revealed communities.

II. Larger networks with labels associated with nodes

Apply selected algorithm to **larger real networks** above and test whether the **revealed communities provide a reasonable decomposition or abstraction** of the networks. Ideally, you should apply the algorithm to each network multiple times and print out the labels of the nodes in the revealed communities.

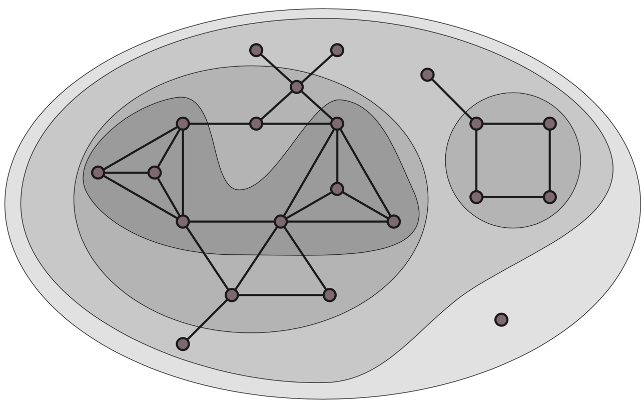
III. Random graphs with no mesoscopic structure

(tentative) Apply selected algorithm also to **Erdös-Rényi random graphs** that should have **no community or other structure** and test whether selected algorithm is able to detect this. You should apply the algorithm to random graphs with increasing average degree $\langle k \rangle$ and test for which values of $\langle k \rangle$ the algorithm doesn't reveal any structure.



IV. k-cores decomposition of real networks

1. *(tentative)* Consider the following **algorithm for computing network k-cores** for a given k. Starting with the original network, iteratively remove nodes with degree less than k. When no such node remains, connected components of the resulting network are the k-cores.



2. *(tentative)* Try to implement the algorithm and compute all **k-cores of the larger real networks** above. For instance, print out the number of k-cores and the size of the largest one for different values of k. Remember that $k+1$-cores are always a subset of k-cores. What is the maximum value of k denoted k_{max} for which there exists at least one k-core?

3. *(tentative)* For each network, print out the **labels of the nodes in k_{max}-core** and interpret the results.