CPA - Practical Work

Community detection

Maximilien Danisch

In this practical we consider algorithms for partitioning the nodes in the input graph into communities.

Exercise 1 — *Simple bechmark*

Implement an algorithm to generate the following random graph.

- The graph has 400 nodes partition into 4 clusters of size 100.
- Each pair of nodes in the same cluster is connected with a probability p
- Each pair of nodes in different clusters is connected with a probability $q \leqslant p$

Draw the obtained graphs for various values of p and q using a software of your choice. For instance: `https://networkx.github.io/documentation/stable/reference/drawing`

What is the effect of increasing or decreasing $\frac{p}{q}$ on the community structure?

Exercise 2 — *Label propagation*

Implement the label propagation algorithm.
Run your program on the benchmark graphs generated for Exercise 1. Draw the graph and color the nodes nodes using a different color for each community. Comment your results.
Make sure that your program scales to graphs containting millions of edges (for examples the ones considered in the first practical).

Exercise 3 — *New algorithm*

Three choices are available for this exercise, listed in decreasing order of difficulty:

1. Suggest your own community detection method and implement it.
2. Make your own implementation of an existing method.
3. Use an existing implementation of an existing method.

Consider an algorithm significantly different from Label Propagation and Louvain.
Explain your algorithm: the intuition behind it and the implementation issues.

Exercise 4 — *Experimental evaluation*

Compare (i) the Label Propagation you have implemented in exercise 2, (ii) the Louvain algorithm (implementation available here: `https://perso.uclouvain.be/vincent.blondel/research/louvain.html`) and (iii) the algorithm in exercise 3. For this you will need to design your own experiments:

- Compare the scalability of the algorithms/programs using graphs of different sizes and reporting the running time.
- Compare the accuracy of the algorithms using benchmarks (for instance the benchmark made in question 1 and the LFR benchmark) and some metrics to compare partitions.

Which algorithm(s) perform(s) best?