M2 – NSD (Project)

Community detection in graphs

Lionel Tabourier, Fabien Tarissan

You will work in pairs for this project. You are asked to make an oral presentation of a community detection method in a graph and its experimental results.

Three choices are available, listed by decreasing difficulty :

1. Propose your own community detection method. (/20 + Bonus points)
2. Make your own implementation of an existing method. (/20)
3. Use an existing implementation of an existing method. (/16)

Instructions for the presentation :

Presentations will be made on **November 22nd** session.

— Presentations last 15 minutes each (please respect the schedule), either in English or in French, each member of the pair present a part of the work.

— You will present your work in a slideshow, you are asked to send it before the 21st of November in pdf format, along with the codes that you have used for the experiments, and the instructions to execute the codes.

— It is expected that you present
  - a description of the algorithm (most important part) and an evaluation of its complexity,
  - experimental results : datasets used (and their features), optimization score (e.g. modularity), features of the communities obtained (e.g. size distribution), computation times, comparison to other methods (e.g. Louvain)
  - you can conclude on a discussion of your results : performances, variations or upgrades or any thought that you consider relevant.

For the pairs which choose options 2 or 3, a list of recommended methods is available on the back, along with an article describing it and the link to an existing implementation. Implementations are there for information purposes only, some methods have others and they have not all been tested. You can choose an algorithm out of this list (excepted Louvain which has been described in the course). You are asked to choose the method that you will present for October 25th and each pair will present a different method, accorindg to the rule “first come, first serve”.

Method list :

— Divisive edge-betweenness algorithm :
Community structure in social and biological networks, Girvan and Newman, 2002.
Implementation in python : `github.com/kjahan/community`

— Random walk based algorithm :
Computing Communities in Large Networks Using Random Walks, Pons and Latapy, 2005.
Implementation in C++ : `www-complexnetworks.lip6.fr/~latapy/PP/walktrap.html`

— Leading eigenvector algorithm :
Finding community structure in networks using the eigenvectors of matrices, Newman, 2006.
Implementation in R : `igraph.org/r/doc/cluster_leading_eigen.html`

— Simulated annealing algorithm :
Functional cartography of complex metabolic networks, Guimera and Amaral, 2005.
Implementation in C : `seeslab.info/downloads/network-c-libraries-rgraph`

— Label propagation algorithm :
Near linear time algorithm to detect community structures in large-scale networks, Ragahavan et al., 2007.
Implementation in R : `igraph.org/r/doc/cluster_label_prop.html`

— K-cliques based algorithm (overlapping communities) :
Uncovering the overlapping community structure of complex networks in nature and society, Palla et al., 2005.
Implementation in R : `igraph.wikidot.com/community-detection-in-r`

— Physics inspired algorithm (overlapping communities) :
Detecting fuzzy community structures in complex networks with a Potts model, Reichardt and Bornholdt, 2004.
Implementation in R : `igraph.org/r/doc/cluster_spinglass.html`

— Map equation algorithm :
Maps of random walks on complex networks reveal community structure, Rosvall and Bergstrom, 2007.
Implementation in R : `igraph.org/r/doc/cluster_infomap.html`