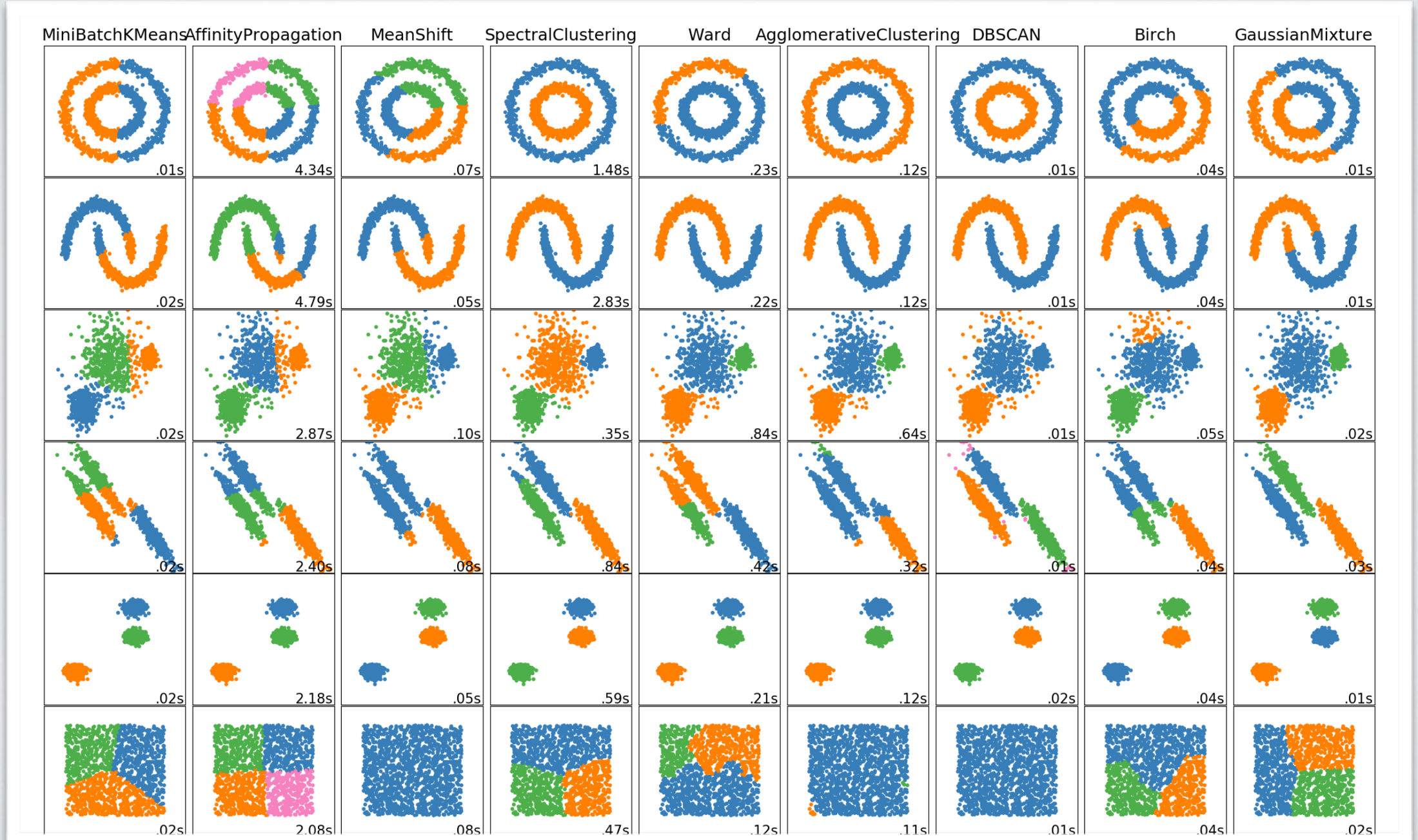
COMMUNITY DETECTION
&
GRAPH CLUSTERING

COMMUNITY DETECTION

- Community detection is equivalent to "clustering" in unstructured data

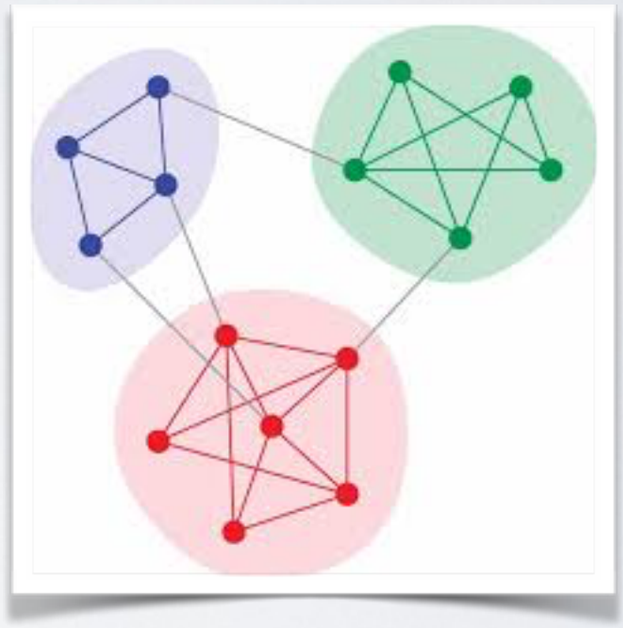
- Clustering: unsupervised machine learning
  - Find groups of nodes that are close to each other
  - Hundreds of methods published since 1950 (k-means)
  - Problem: what does "close to each other" means ?

COMMUNITY DETECTION



COMMUNITY DETECTION

- Community detection:
  - Find groups of nodes that are:
    - Closely connected to each other
    - Weakly connected to the rest of the network
  - No formal definition
  - Thousands of methods published since 2003



WHY COMMUNITY DETECTION ?

- Do you remember small world networks?
  - High clustering coefficient
  - (friends of my friends are my friends)
- Different from random networks. How to explain it ? Evenly distributed ?
- => In real networks, presence of dense groups: communities

SOME HISTORY

- The *graph partitioning problem* was a classic problem in graph theory

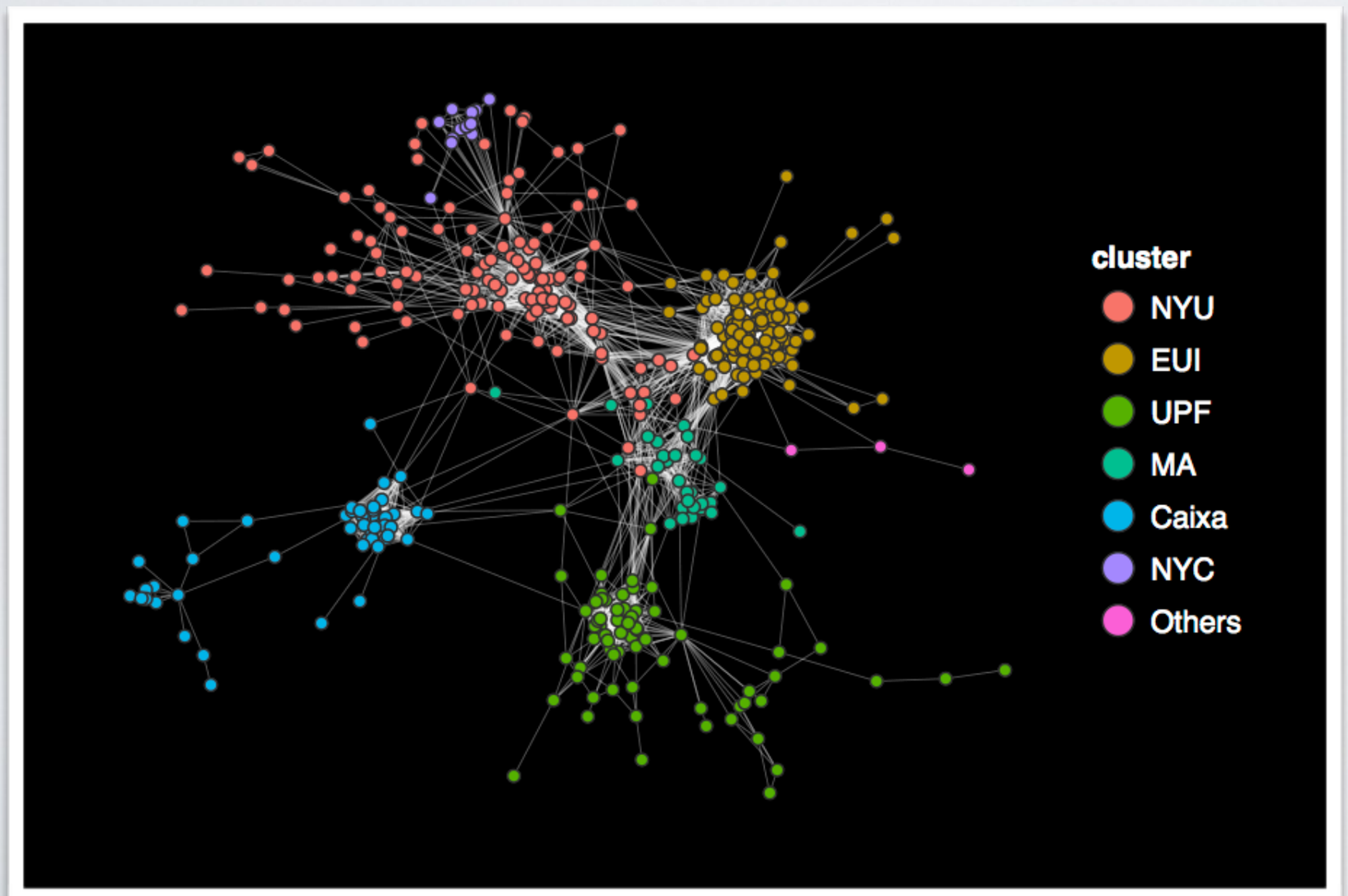
- It goes like this:
  - How to split a network in **k** equal parts such that there is a minimal number of edges between part.
  - It was one problem among many others
  - Variants were proposed:
    - What if partitions are not exactly same size ?
    - What if the number of parts is not exactly k ?
    - …

SOME HISTORY

- Then in 2002, [Girvan & Newman 2002], introduction of the problem of "community discovery":
  - Observation that social networks are very often composed of groups
  - The number and the size of these groups is not known in advance
  - Can we design an algorithm to discover automatically those groups ?

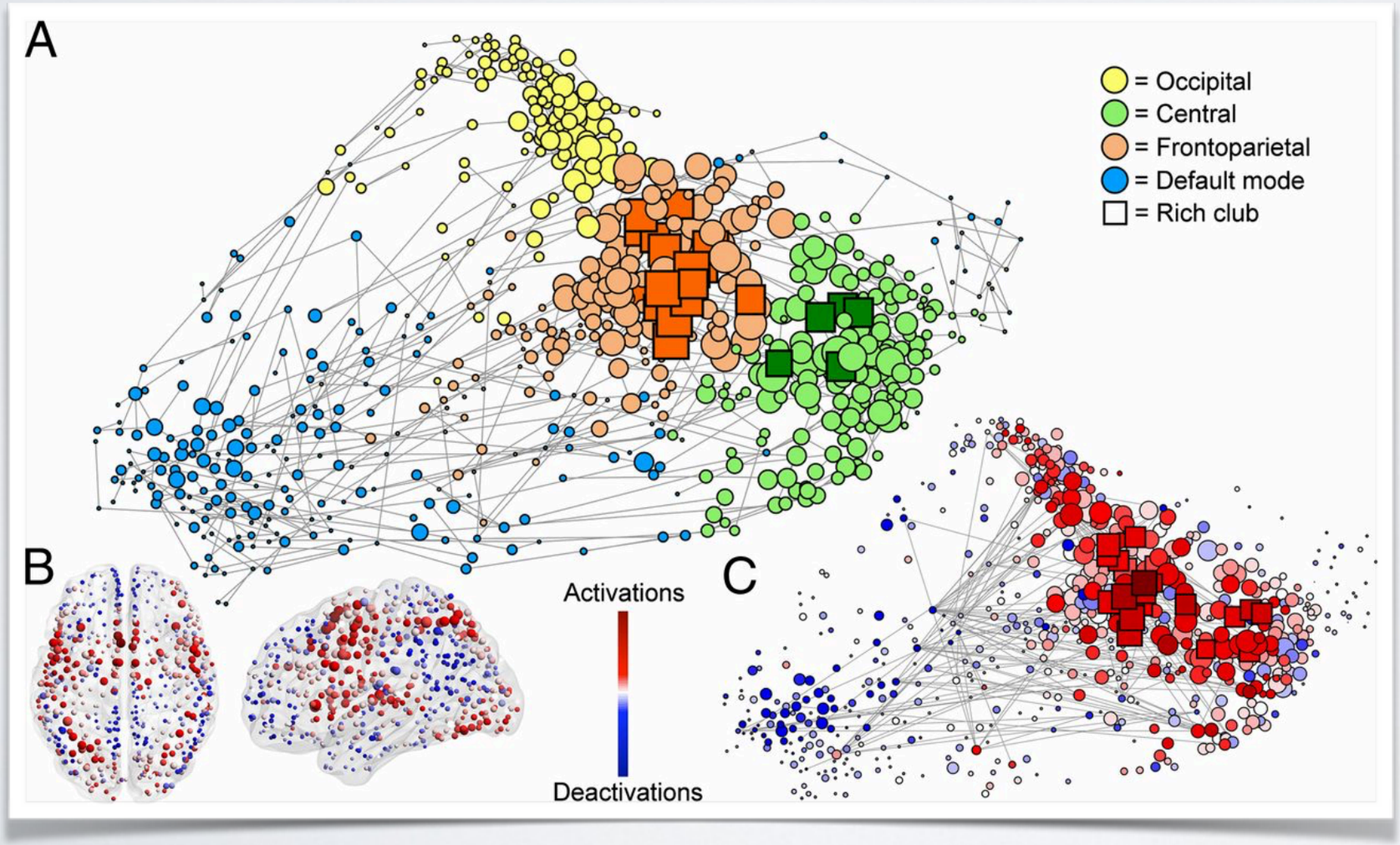
COMMUNITY STRUCTURE IN REAL GRAPHS

- If you plot the graph of your facebook friends, it looks like this



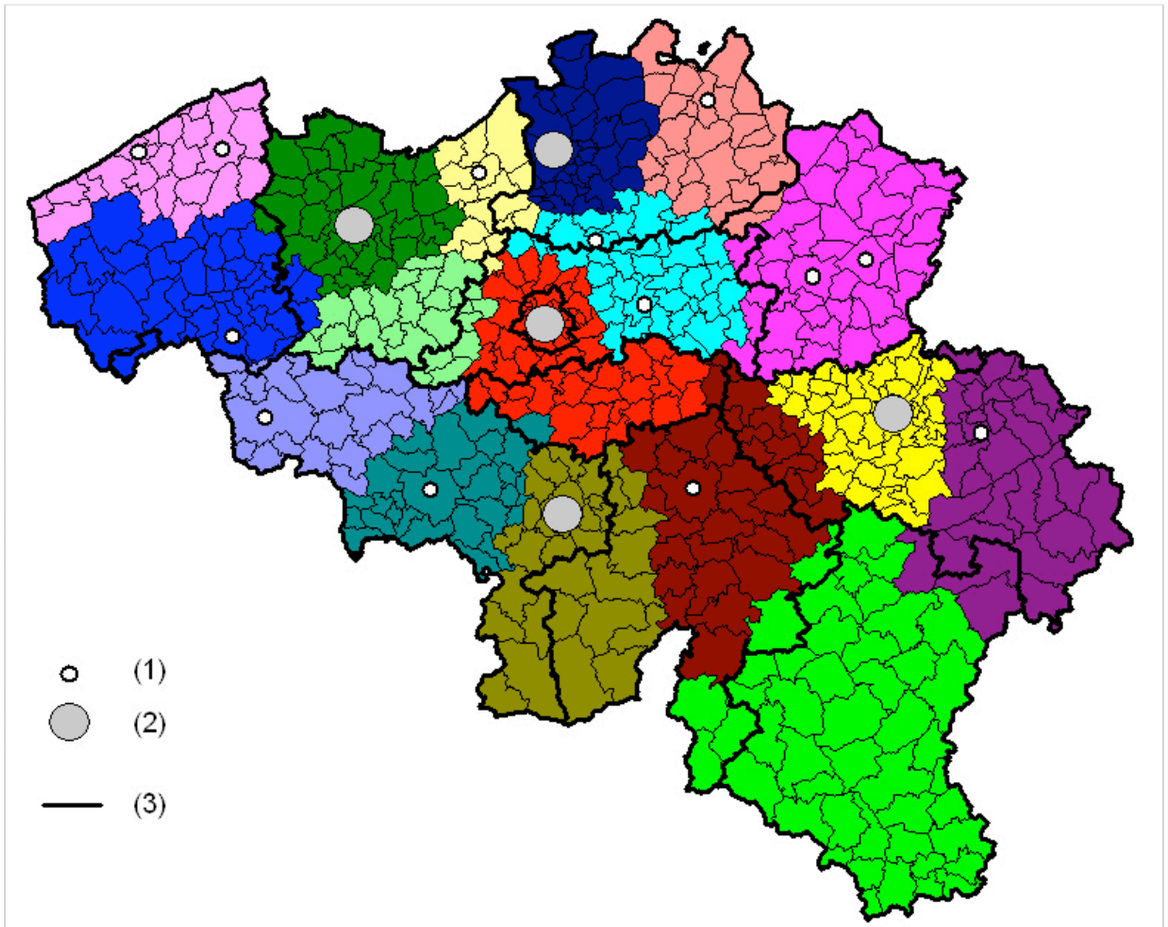
COMMUNITY STRUCTURE IN REAL GRAPHS

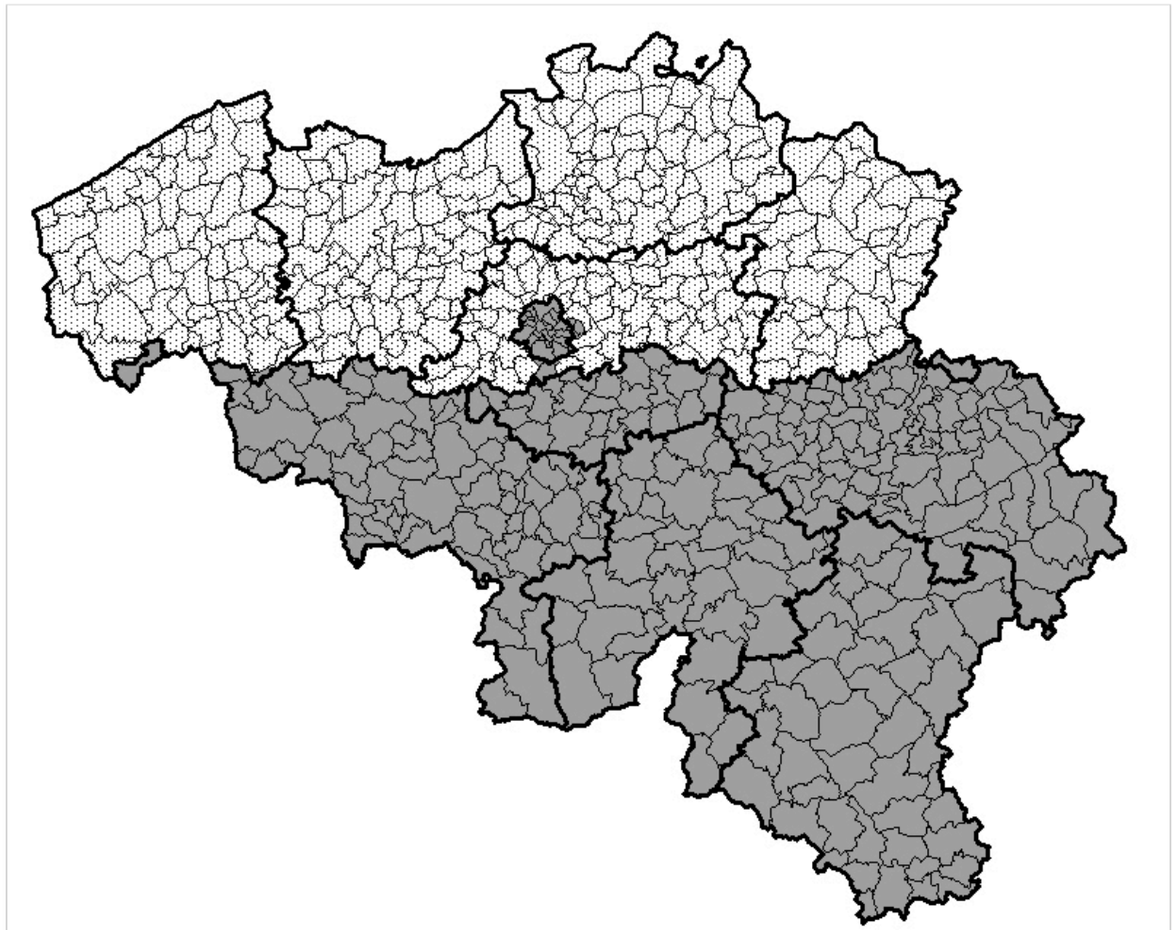
- Connections in the brain ?



COMMUNITY STRUCTURE IN REAL GRAPHS

- Phone call communications in Belgium ?

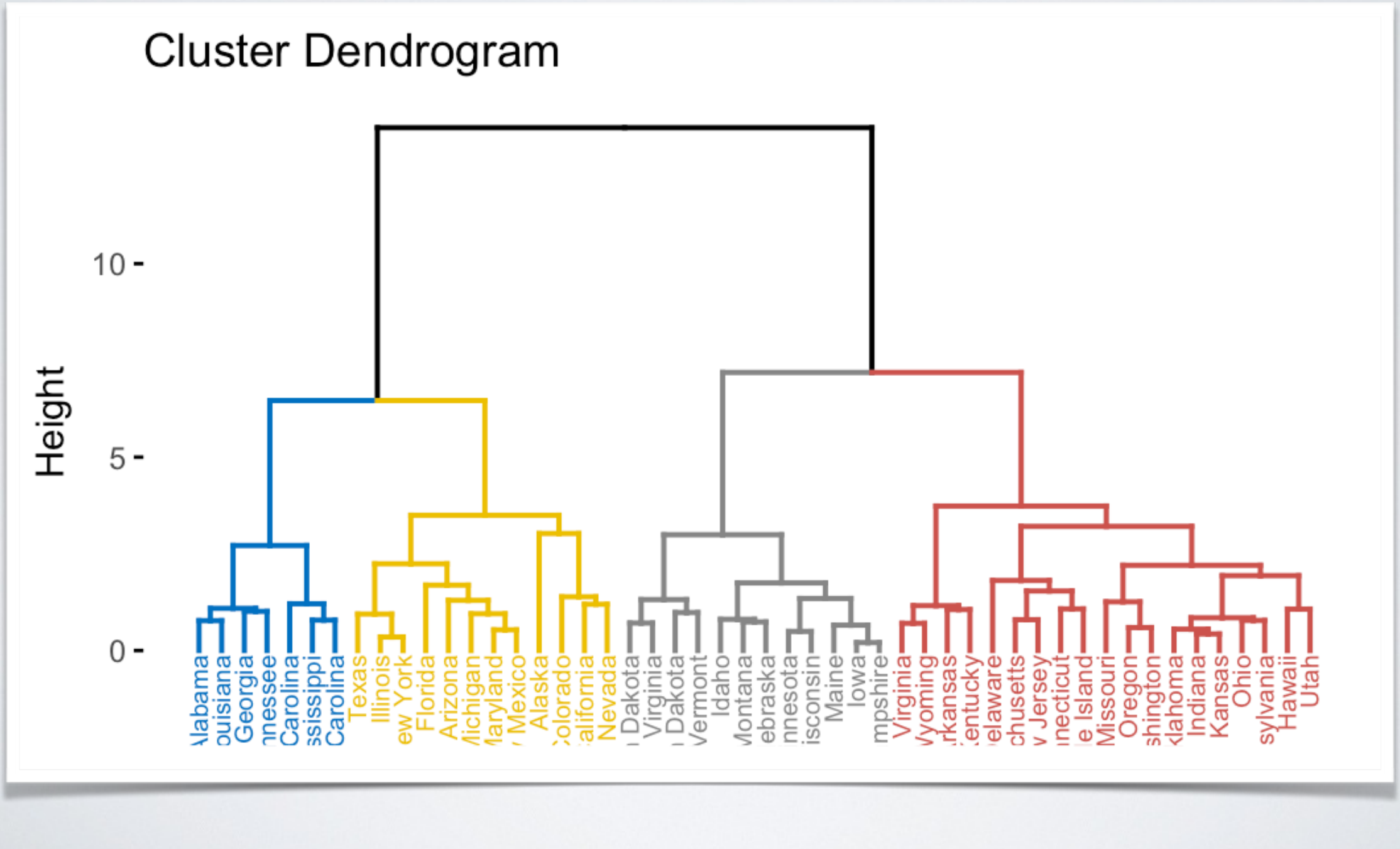




FIRST METHOD BY GIRVAN & NEWMAN

- 1)Compute the betweenness of all edges
- 2)Remove the edge of highest betweenness
- 3)Repeat until all edges have been removed
- => It is called a *divisive* method
- =>What you obtain is a dendrogram
- How to cut this dendrogram at the *best* level ?

FIRST METHOD BY GIRVAN & NEWMAN



FIRST METHOD BY GIRVAN & NEWMAN

- Introduction of the **Modularity**

- The modularity is computed for a partition of a graph
  - (each node belongs to one and only one community)

- It compares :
  - the **observed** *fraction of edges inside communities*
  - to the **expected** *fraction of edges inside communities* in a random network

MODULARITY

$$Q = \frac{1}{(2m)} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{(2m)} \right] \delta(c_v, c_w) = \sum_{i=1}^{c} (e_{ii} - a_i^2)$$

$$e_{ij} = \sum_{vw} \frac{A_{vw}}{2m} 1_{v \in c_i} 1_{w \in c_j}$$

$$a_i = \frac{k_i}{2m} = \sum_j e_{ij}$$

MODULARITY

$$Q = \frac{1}{(2m)} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{(2m)} \right] \delta(c_v, c_w) = \sum_{i=1}^{c} (e_{ii} - a_i^2)$$

Sum over all pairs of nodes

$$e_{ij} = \sum_{vw} \frac{A_{vw}}{2m} 1_{v \in c_i} 1_{w \in c_j}$$

$$a_i = \frac{k_i}{2m} = \sum_j e_{ij}$$

MODULARITY

$$Q = \frac{1}{(2m)} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{(2m)} \right] \delta(c_v, c_w) = \sum_{i=1}^{c} (e_{ii} - a_i^2)$$

1 if in same community

$$e_{ij} = \sum_{vw} \frac{A_{vw}}{2m} 1_{v \in c_i} 1_{w \in c_j}$$

$$a_i = \frac{k_i}{2m} = \sum_j e_{ij}$$

MODULARITY

$$Q = \frac{1}{(2m)} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{(2m)} \right] \delta(c_v, c_w) = \sum_{i=1}^{c} (e_{ii} - a_i^2)$$

1 if there is an edge between them

$$e_{ij} = \sum_{vw} \frac{A_{vw}}{2m} 1_{v \in c_i} 1_{w \in c_j}$$

$$a_i = \frac{k_i}{2m} = \sum_j e_{ij}$$

MODULARITY

$$Q = \frac{1}{(2m)} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{(2m)} \right] \delta(c_v, c_w) = \sum_{i=1}^{c} (e_{ii} - a_i^2)$$

Probably of an edge in
A random network

$$e_{ij} = \sum_{vw} \frac{A_{vw}}{2m} 1_{v \in c_i} 1_{w \in c_j}$$

$$a_i = \frac{k_i}{2m} = \sum_j e_{ij}$$

MODULARITY

- One point to note:
  - Number of edges in a **random** network: what type of random network ?

- Original (and still mostly used) modularity:
  - The **Configuration model**, or *degree preserving* random model
  - The degrees of nodes is conserved.
  - Multi-edges and loops are allowed (for practical reasons)
  - Probability/number of edges:

$$p_{ij} = \frac{k_i k_j}{2m-1} \simeq \frac{k_i k_j}{2m} ,$$

FIRST METHOD BY GIRVAN & NEWMAN

- Back to the method:
  - Create a dendrogram by removing edges
  - Cut the dendrogram at the best level using modularity

- =>In the end, your objective is… to optimize the Modularity, right ?

- Why not optimizing it directly !

MODULARITY OPTIMIZATION

- From 2004 to 2008: The golden age of Modularity
- Scores of methods proposed to optimize it
  - Graph spectral approaches
  - Meta-heuristics approches (simulated annealing, multi-agent…)
  - Local/Gloabal approaches…
- => 2008: the Louvain algorithm

LOUVAIN ALGORITHM

- Simple, greedy approach
  - Easy to implement
  - Extremely fast
- Yield a hierarchical community structure
- Beats state of the art on all aspects
  - Speed
  - Max modularity obtained
  - Do not fall in some traps (see later)

LOUVAIN ALGORITHM

- Each node start in its own community

- Repeat until convergence
  - FOR each node:
    - FOR each neighbor:
      if adding node to its community increase modularity, do it

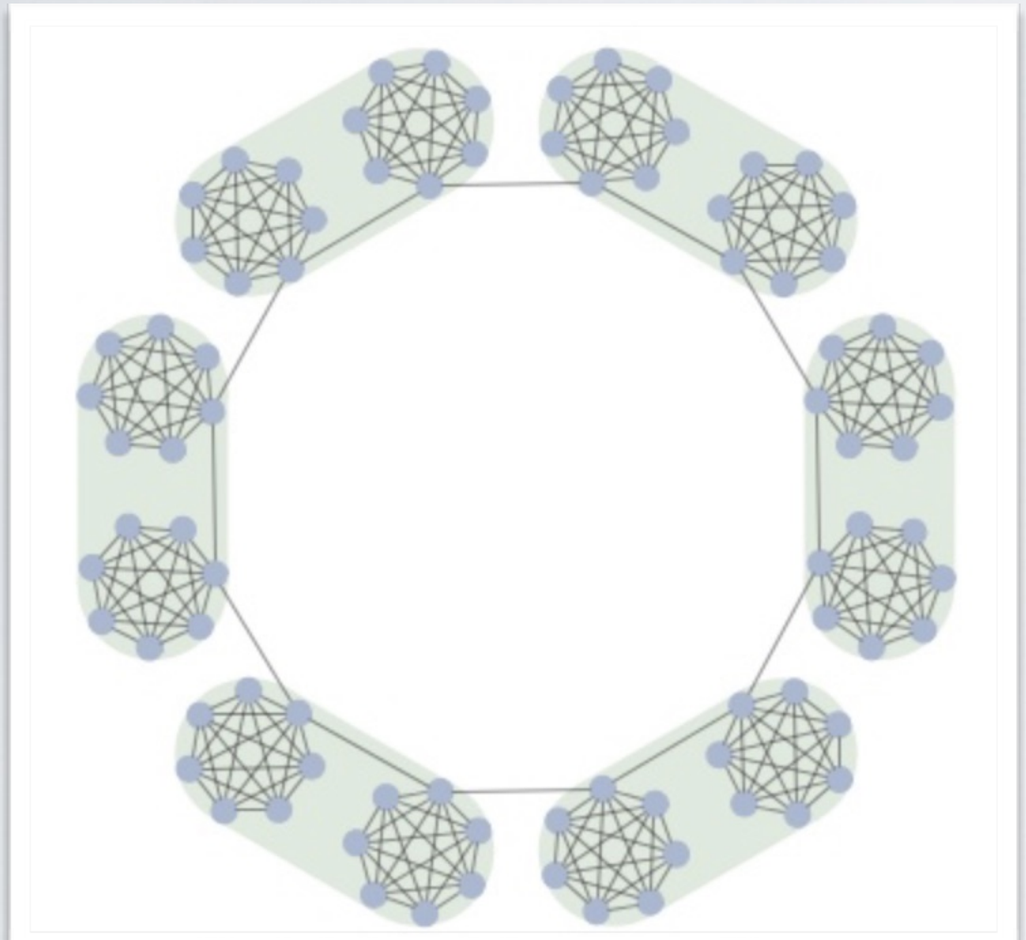
- When converged, create an *induced network*
  - Each community becomes a node
  - Edge weight is the sum of weights of edges between them

- Trick: Modularity is computed *by community*
  - Global Modularity = sum of modularities of each community

RESOLUTION LIMIT

- Modularity == Definition of good communities ?
- 2006-2008: series of articles [Fortunato-Lancicchinetti]
  - Resolution limit of Modularity
- => Modularity has intrinsic flaws, it is only *one* measure of the quality of communities
- Let's see examples

RESOLUTION LIMIT



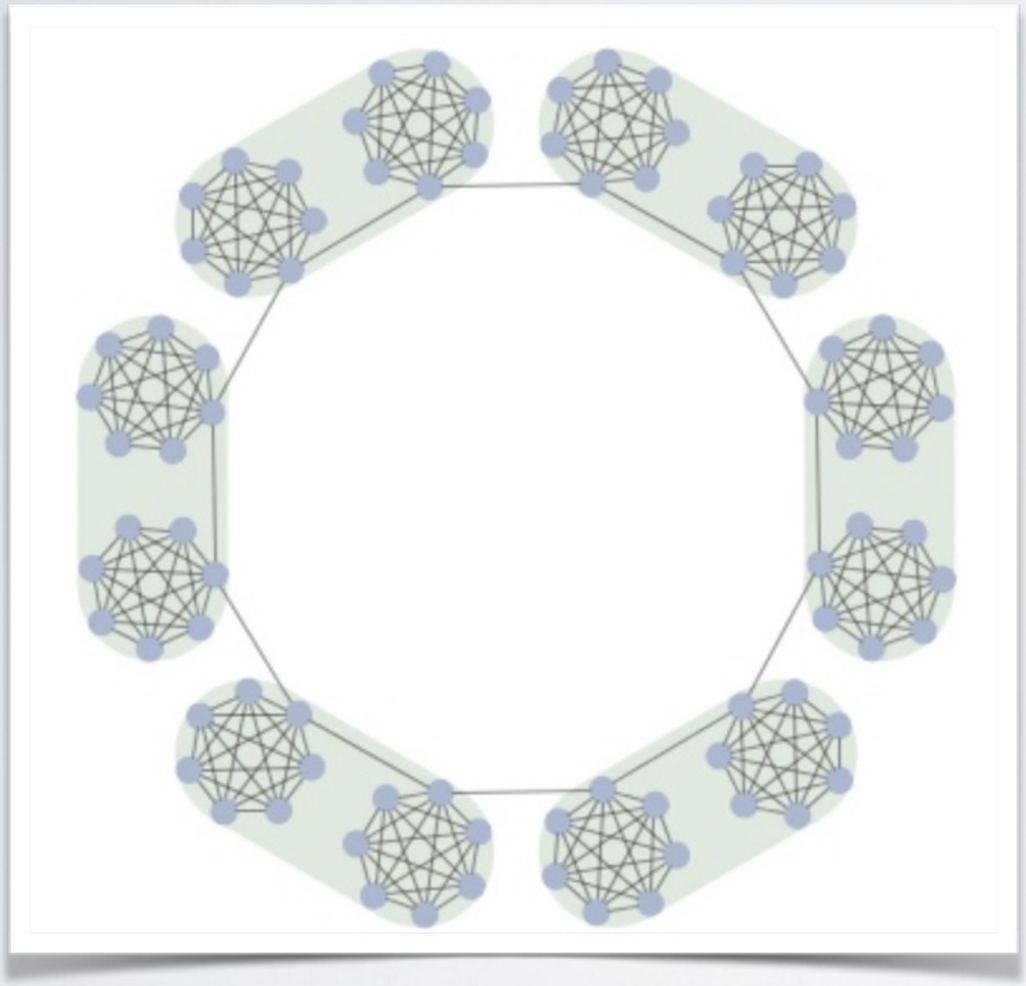
Let's consider a ring of cliques

Cliques are as dense as possible

Single edge between them:
=>As separated as possible

Any acceptable algorithm=>Each clique is a community

RESOLUTION LIMIT



But with modularity:

Small graphs=> OK

Large graphs=>
The max of modularity obtained by merging cliques

RESOLUTION LIMIT

- Discovery that Modularity has a "favorite scale":

- For a graph of given **density** and **size:**
  - Communities cannot be smaller than a fraction of nodes
  - Communities cannot be larger than a fraction of nodes

- Modularity optimisation will never discover
  - Small communities in large networks
  - Large communities in small networks

OTHER WEAKNESSES

- Modularity has other controversial/not-intuitive properties:
  - Global measure => a difference in one hand of the network can change communities at the other end (imagine a growing clique ring…)
  - Unable to find no community:
    - Network without community structure: Max modularity for random partitions
- To this day, Louvain and modularity still most used methods
  - Results are usually "good"/useful

ALTERNATIVES

- 1000+ Algorithms published, 2+ more every month (not an exaggeration)

- Most of them are mostly uninteresting:
  - They define their own definition of communities
  - They show on a few network using a single validation method that their method is better than Louvain (10y.o. algorithm)

- Common saying: "no algorithm is better than other, it depends on the network" (I don't really agree)

ALTERNATIVES

- Most serious alternatives (in my opinion)
  - Infomap (based on information theory —compression)
  - Stochastic block models (generative models)

- These methods have a clear definition of what are good communities. Theoretically grounded

- Most other methods are ad hoc=>They define a process, without a clear definition

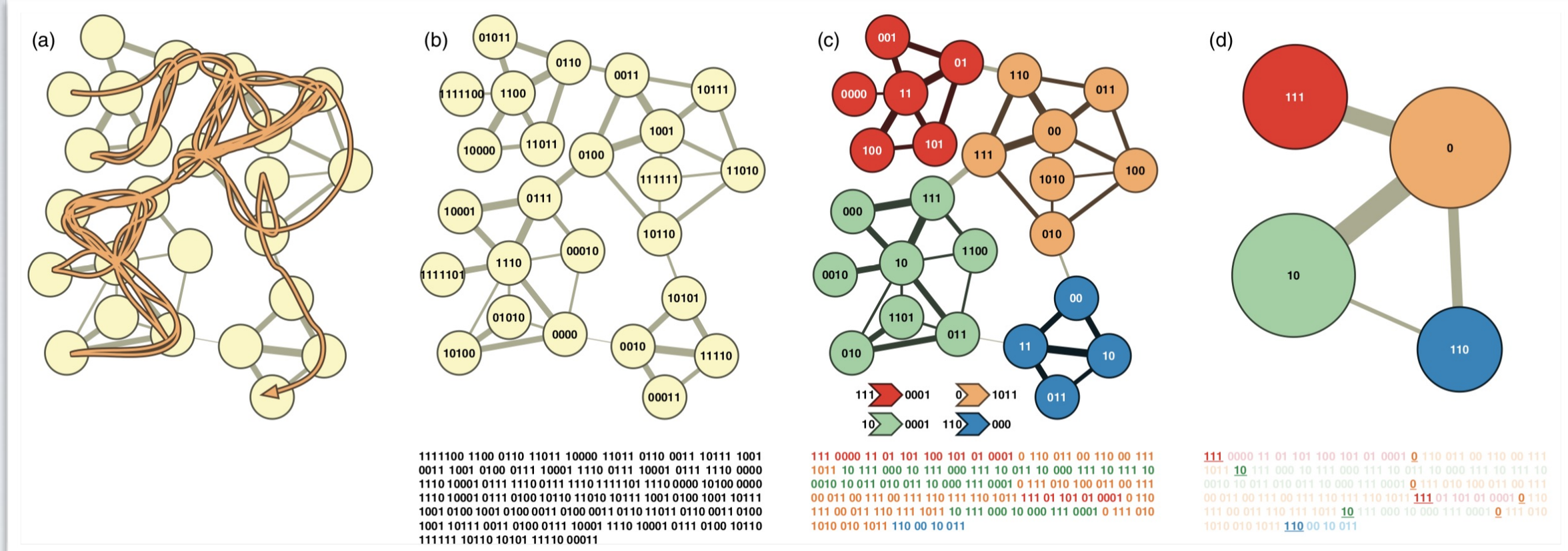
INFOMAP

- [Rosval 2009]

- Find the partition minimizing the *description* of any random walk on the network

- We want to *compress* the description of random walks

INFOMAP



Random walk

Description Without Communities

With communities

INFOMAP

- In practice, no assignment of codes to nodes
- Information theory (Shannon coding, entropy)
- Given
  - The size of communities
  - The number of out-going links
  - The number of intra-community links
  - =>We know how many bits we need for the optimal code

$$L(\mathsf{M}) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{m} p^i_{\circlearrowright} H(\mathcal{P}^i)$$

INFOMAP

$$L(\mathsf{M}) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{m} p^i_{\circlearrowright} H(\mathcal{P}^i)$$

$$H(\mathcal{Q}) = -\sum_{i=1}^{m} \frac{q_{i\curvearrowright}}{\sum_{j=1}^{m} q_{j\curvearrowright}} \log \left(\frac{q_{i\curvearrowright}}{\sum_{j=1}^{m} q_{j\curvearrowright}} \right)$$

$$H(\mathcal{P}^i) = -\frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta \in i} p_\beta} \log \left(\frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta \in i} p_\beta} \right) - \sum_{\alpha \in i} \frac{p_\alpha}{q_{i\curvearrowright} + \sum_{\beta \in i} p_\beta} \log \left(\frac{p_\alpha}{q_{i\curvearrowright} + \sum_{\beta \in i} p_\beta} \right)$$

INFOMAP

$$L(\mathsf{M}) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{m} p^i_{\circlearrowright} H(\mathcal{P}^i)$$

Probability to exit community i

$$H(\mathcal{Q}) = -\sum_{i=1}^{m} \frac{q_{i\curvearrowright}}{\sum_{j=1}^{m} q_{j\curvearrowright}} \log\left(\frac{q_{i\curvearrowright}}{\sum_{j=1}^{m} q_{j\curvearrowright}}\right)$$

$$H(\mathcal{P}^i) = -\frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta} \log\left(\frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta}\right) - \sum_{\alpha\in i} \frac{p_\alpha}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta} \log\left(\frac{p_\alpha}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta}\right)$$

INFOMAP

$$L(\mathsf{M}) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{m} p^i_{\circlearrowright} H(\mathcal{P}^i)$$

Probability of a transition intern to community i

$$H(\mathcal{Q}) = -\sum_{i=1}^{m} \frac{q_{i\curvearrowright}}{\sum_{j=1}^{m} q_{j\curvearrowright}} \log\left(\frac{q_{i\curvearrowright}}{\sum_{j=1}^{m} q_{j\curvearrowright}}\right)$$

$$H(\mathcal{P}^i) = -\frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta} \log\left(\frac{q_{i\curvearrowright}}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta}\right) - \sum_{\alpha\in i} \frac{p_\alpha}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta} \log\left(\frac{p_\alpha}{q_{i\curvearrowright} + \sum_{\beta\in i} p_\beta}\right)$$

INFOMAP

- To sum up:
  - Infomap defines a *quality function* for a partition different than modularity
  - Any algorithm can be used to optimize it (like Modularity)
  - The most recent version uses the same algorithm as Louvain
- Advantage:
  - Infomap can recognize random networks (no communities)
  - It is nearly as fast as Louvain
- Drawback:
  - It seems to suffer from a sort of resolution limit

STOCHASTIC BLOCK MODELS

- Stochastic Block Models (SBM) are based on generative models of networks

- They are in fact more general than normal communities.

- The model is:
  - Each node belongs to 1 and only 1 community
  - To each pair of communities, there is an associated density (probability of each edge to exist)

STOCHASTIC BLOCK MODELS

- SBM can represent different things:
  - Normal communities: density inside nodes of a same communities >> density of pairs belonging to different communities.
  - SBM allow to represent heterophily:
    - In a "sentimental" network, 2 clusters men/women, high density between, low density inside.
- This is very powerful and potentially relevant
- Problem: Often hard to interpret in real situations.
  - SBM can be "constrained": we impose that intra d.>inter d.

STOCHASTIC BLOCK MODELS

- General idea of SBM community detection:
  - Specify the desired number of cluster
  - Find parameters that minimize the "error" of the model, i.e. difference between observed network and average network generated by the SBM

- Underlying idea: Communities are "random sub-networks"

- Again, question is: what type of random networks ?
  - Erdos Renyi ?
  - Degree corrected ? <=gives better results on real networks

STOCHASTIC BLOCK MODELS

- Main weakness of SBM:
  - Number of clusters must be specified (avoid trivial solution)
- Usual approach to solve it
  - Similar to k-means in clustering: try different k and measure improvement (elbow-method)
  - Not satisfying
- [2016 Peixoto]
  - Non-parametric SBM
  - Use the principle of Minimum Description Length (MDL) (Occam's razor)
  - Principle of information theory and compression, combine
    - The cost of the error
    - The complexity of the model

STOCHASTIC BLOCK MODELS

- To sum up:
  - SBM have a convincing definition of communities
  - In practice, slower than louvain/infomap
  - But more powerful
  - Can also say if there is no community
  - And also suffer from a form or resolution limit

- Less often used, but regain popularity since works by Peixoto et al.

EVALUATION OF COMMUNITY STRUCTURE

EVALUATION

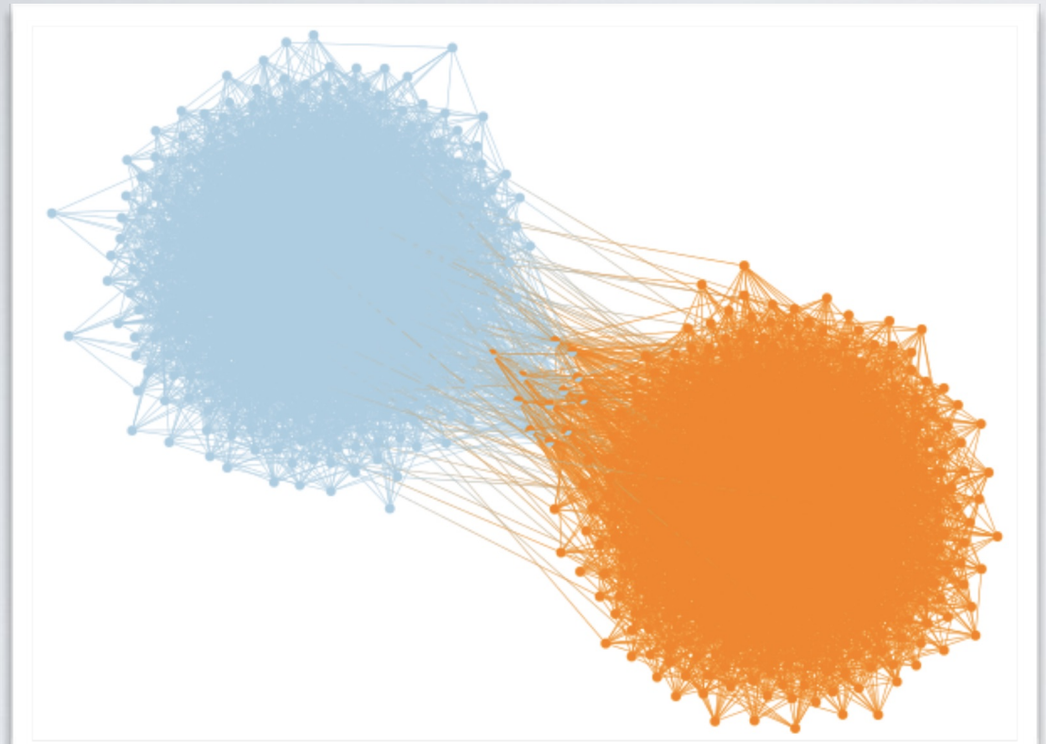
- Two main approaches, both Pros and Cons
  - Synthetic networks with community structure
  - Real networks with Ground Truth

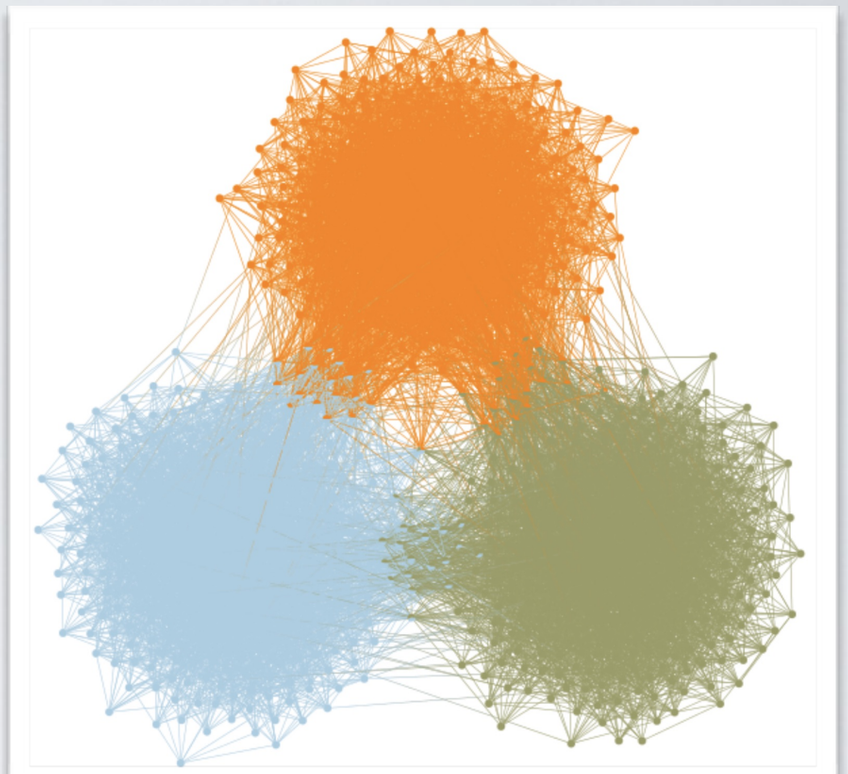
- Same idea: compare what we know we should find with what each algorithm finds

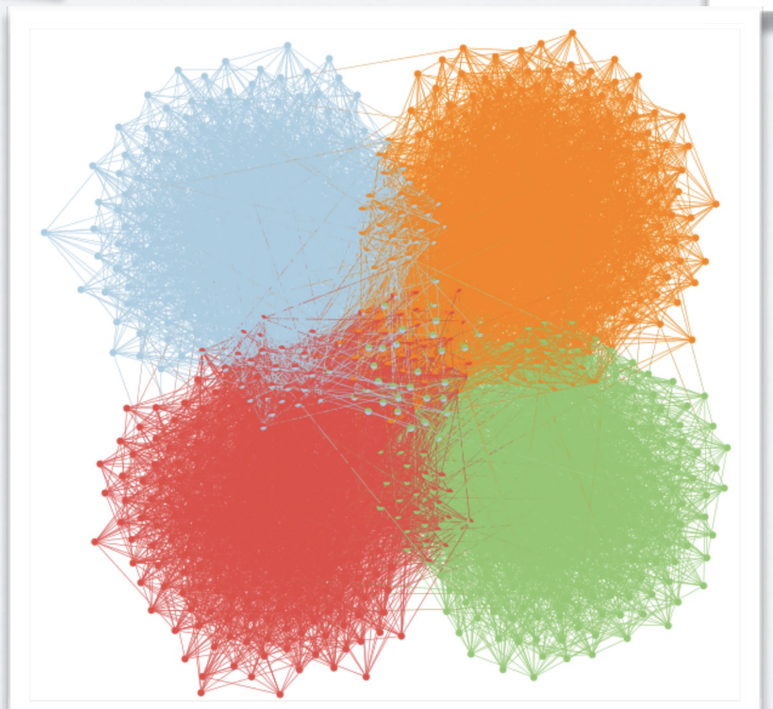
SYNTHETIC NETWORKS

- Planted Partition models:
  - Another name for SBM with manually chosen parameters
    - Assign degrees to nodes
    - Assign nodes to communities
    - Assign density to pairs of communities
    - Attribute randomly edges
  - Problem: how to choose parameters?
    - Either oversimplifying (all nodes same degrees, all communities same #nodes, all intern densities equals…)
    - Or too complex

SYNTHETIC NETWORKS





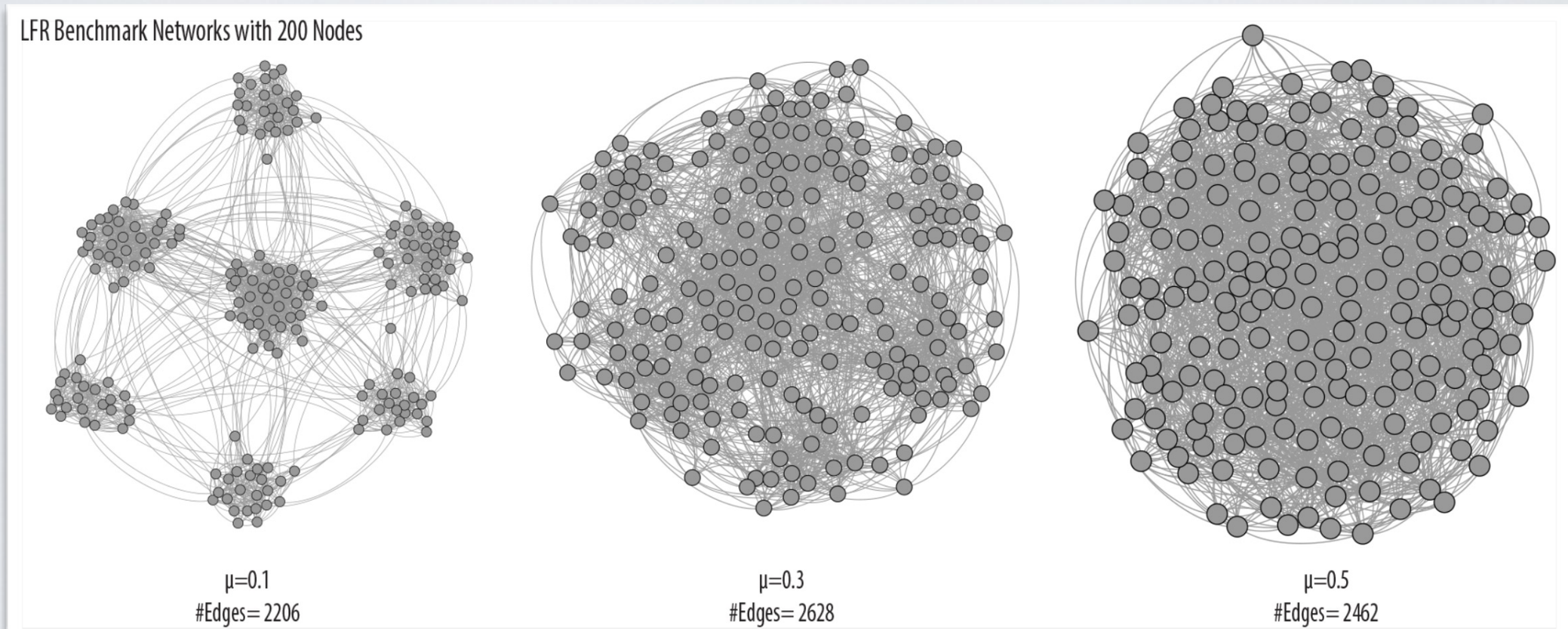


SYNTHETIC NETWORKS

- LFR Benchmark [Lancichinetti 2008]
  - High level parameters:
    - Slope of the power law distribution of degrees/community sizes
    - Avg Degree, Avg community size
    - Mixing parameter: fraction of intern edges of each node
  - Varying the mixing parameter makes community more or less well defined

- Currently the most used (by people not doing SBM)

SYNTHETIC NETWORKS



SYNTHETIC NETWORKS

- Pros of synthetic generators:
  - We know for sure the communities we should find
  - We can control finely the parameters to check robustness of methods
    - For instance, resolution limit…

- Cons:
  - Generated networks are not realist: more simple than real networks
  - Generated communities might not be realist (We don't really know what real communities look like…)

REAL NETWORKS WITH GT

- In some networks, Ground Truth communities are known:
  - Social networks, people belong to groups (Facebook, Friendsters, Orkut, students in classes…)
  - Products, belonging to categories (Amazon, music…)
  - Other resources with defined groups (Wikipedia articles, Political groups for vote data…)

- Some websites have collected such datasets, e.g.
  - http://snap.stanford.edu/data/index.html

REAL NETWORKS WITH GT

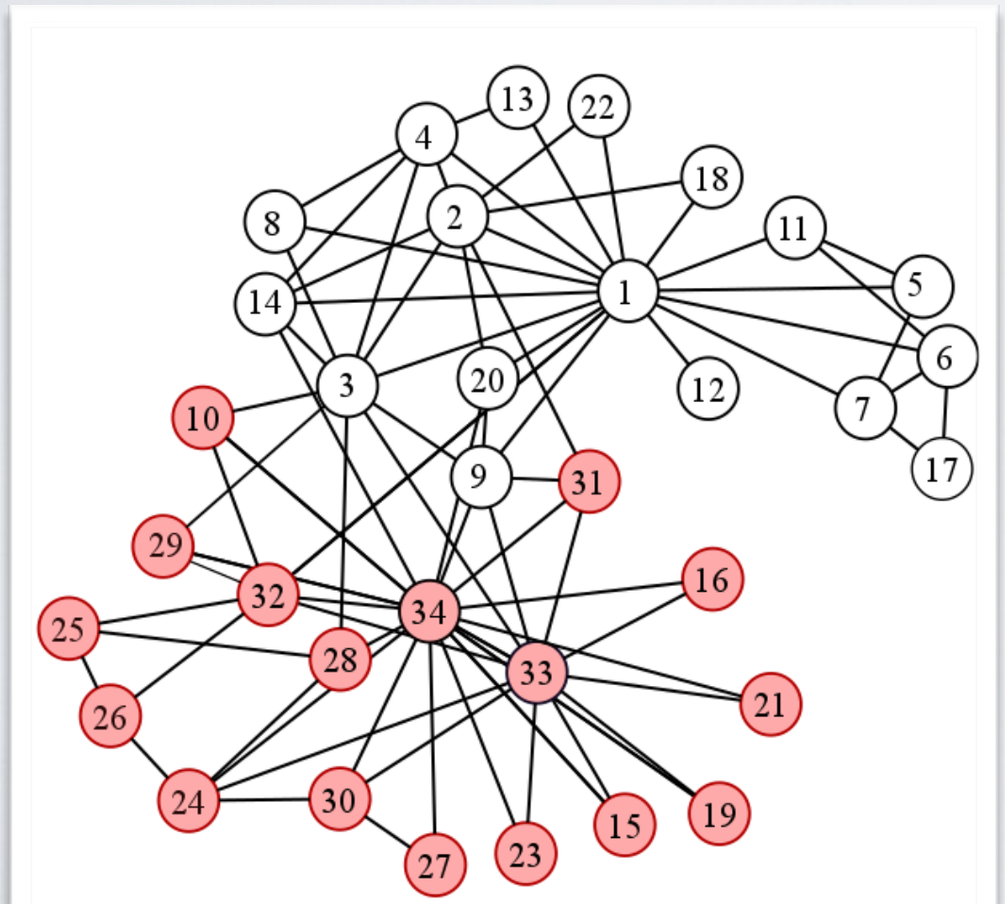
- Pros of GT communities:
  - Retain the full complexity of networks and communities

- Cons:
  - No guarantee that communities are **topological** communities.
  - In fact, they are not: some communities are not even a single connected component…

- Currently, controversial topic
  - Some authors say it is non-sense to use them for validation
  - Some others consider it necessary

REAL NETWORKS WITH GT

- Example: the most famous of all networks: Zackary Karate Club

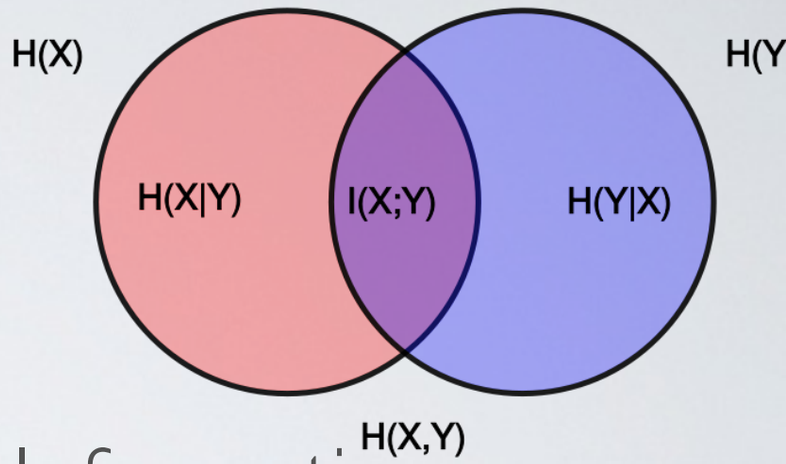


If your algorithm find the right communities,
Then it is wrong…

MEASURING PARTITION SIMILARITIES

- Synthetic or GT, we get:
  - Reference communities
  - Communities found by algorithms
- How to measure their similarity ?
  - NMI
  - aNMI
  - F1-score

MEASURING PARTITION SIMILARITIES



- NMI: Normalized Mutual Information
- Classic notion of Information Theory: Mutual Information
  - How much knowing one variable reduces uncertainty about the other
  - Or how much in common between the two variables

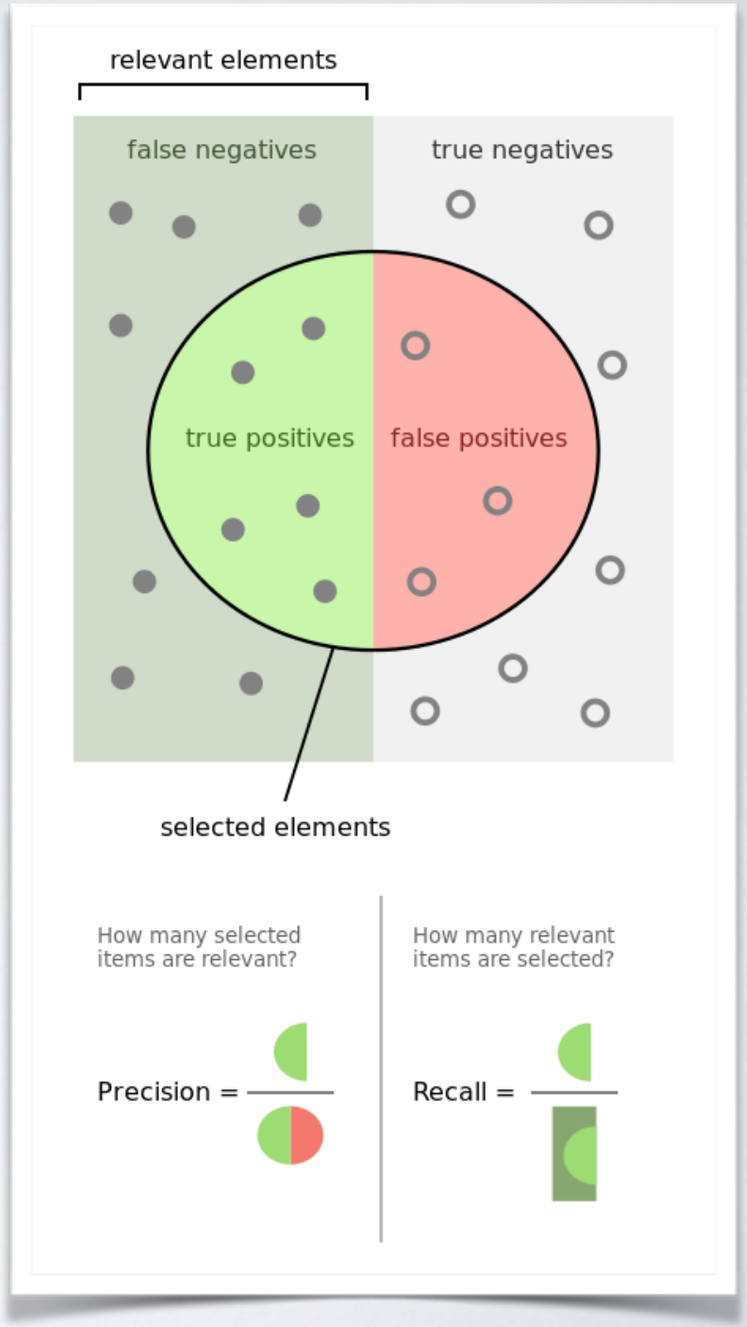
$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p(x)\,p(y)} \right)$$

- Normalized version: NMI
- Adjusted for chance: aNMI

MEASURING PARTITION SIMILARITIES

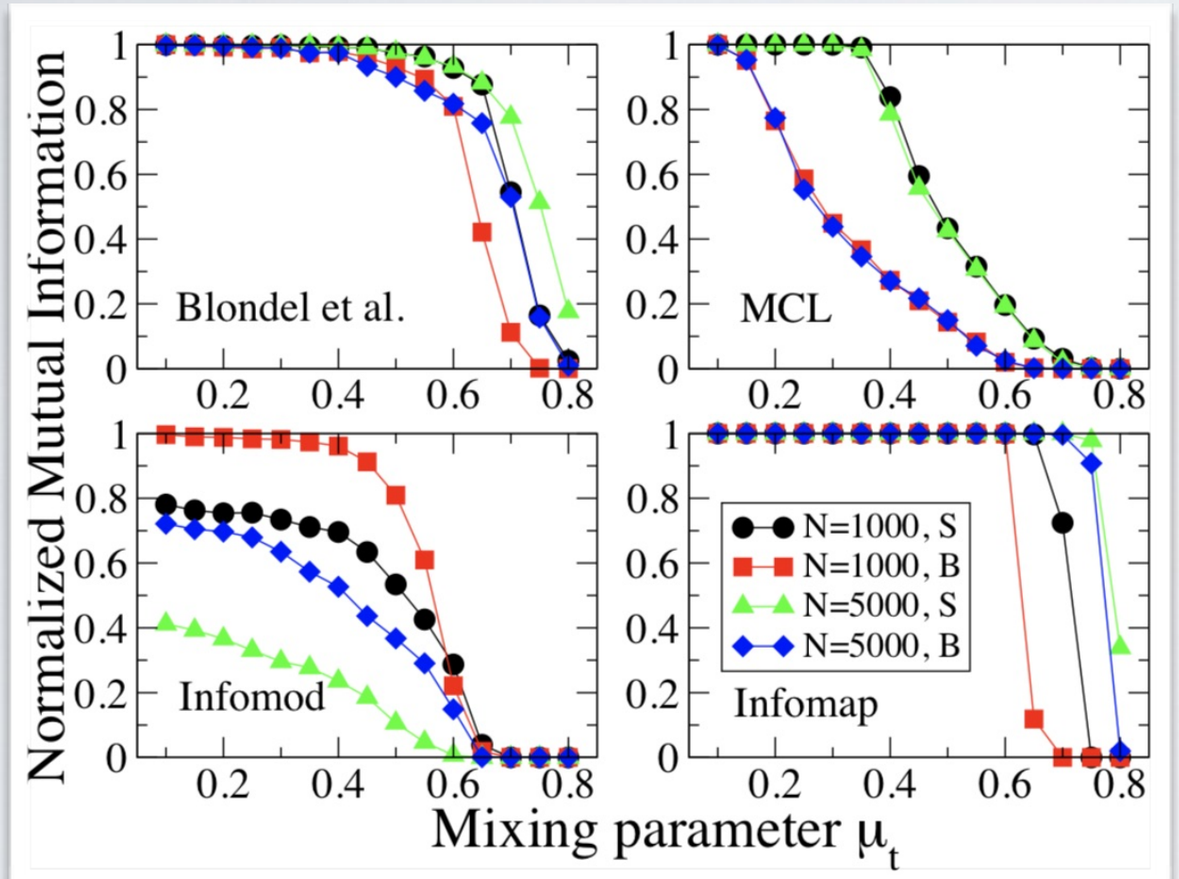
- F1-score: Borrowed from machine learning
  - Harmonic mean of Precision & Recall

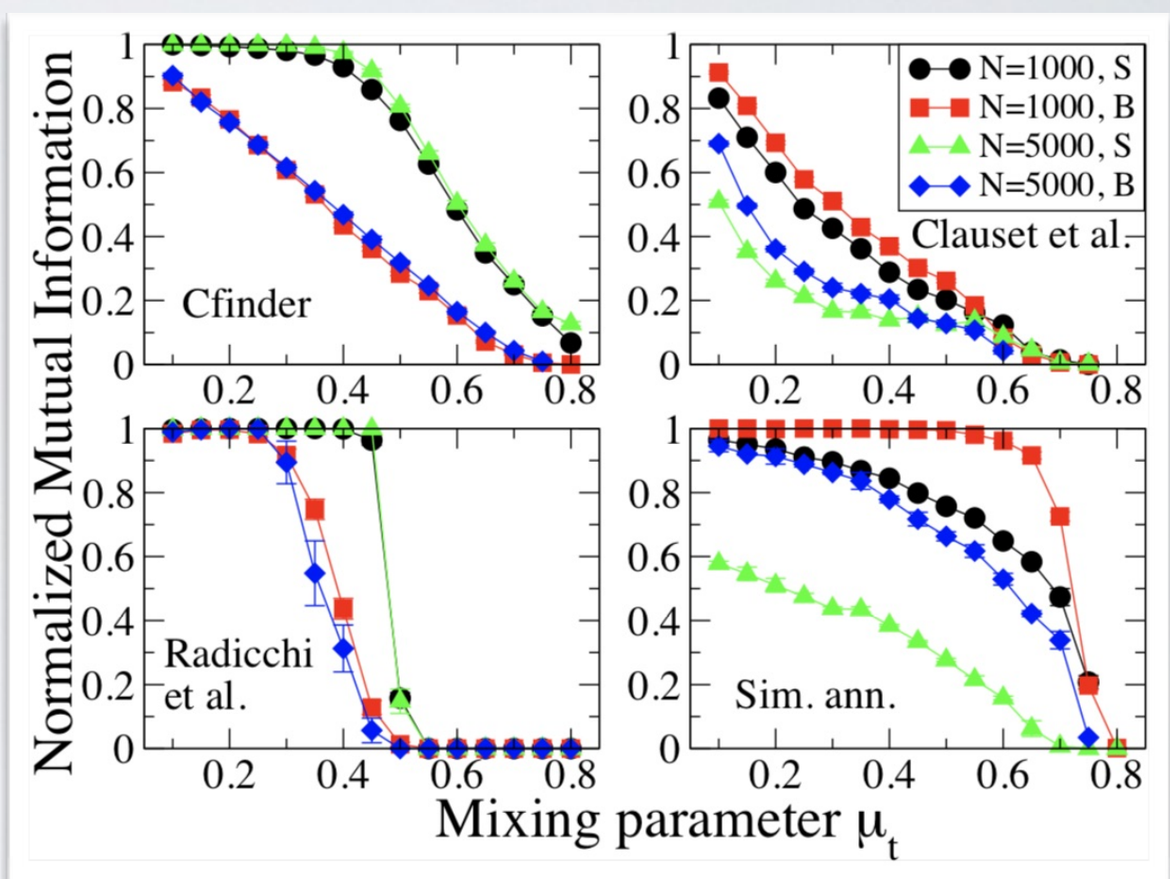
$$F_1 = \frac{2}{\frac{1}{\mathrm{recall}} + \frac{1}{\mathrm{precision}}} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$



Precision Recall for Clustering:
(Pairs of nodes in the same clusters)

ALGORITHMS COMPARATIVE ANALYSIS





[Fortunato 2008]

OTHER MESO-SCALE ORGANIZATIONS

MESO-SCALE

- Course 1: MACRO properties of networks

- Course 2: MICRO properties of networks

- MESO-scale: what is in-between
  - Community structure
  - Overlapping Community Structure
  - Core-Periphery
  - Spatial Organization

OVERLAPPING COMMUNITIES

- In real networks, communities are often overlapping
  - Some of your High-School friends might be also University Friends
  - A colleague might be a member of your family
  - …
- Overlapping community detection is considered much harder
  - And is not well defined
- Border between "attributes" and overlapping communities ?
  - Community of Women, Community of 17-19yo, Community of fans of X…

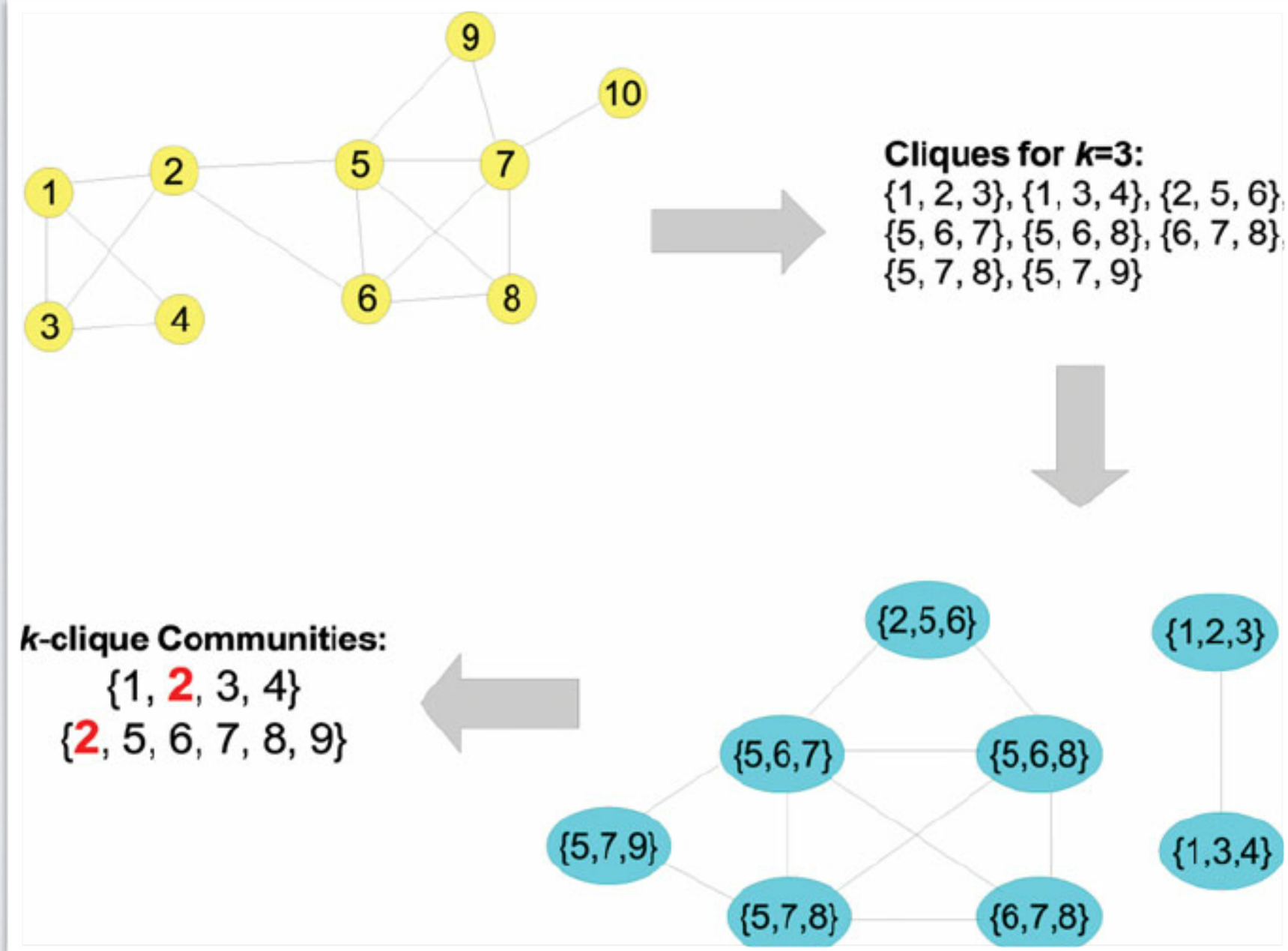
OVERLAPPING COMMUNITIES

- Nevertheless, many algorithms (50+)
- I present only the most famous although it is obsolete
  - Because no agreement on another reference…

K-CLIQUE PERCOLATION

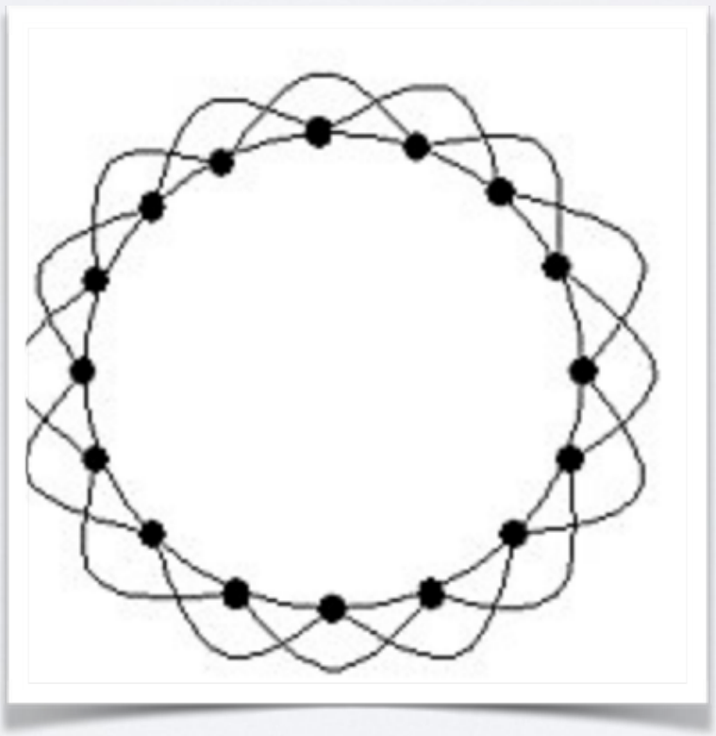
- (Other name: CPM, C-finder)
- Parameter: size k of cliques
- 1)Find all cliques of size k
- 2)merge iteratively all cliques having k-1 nodes in common

K-CLIQUE PERCOLATION



K-CLIQUE PERCOLATION

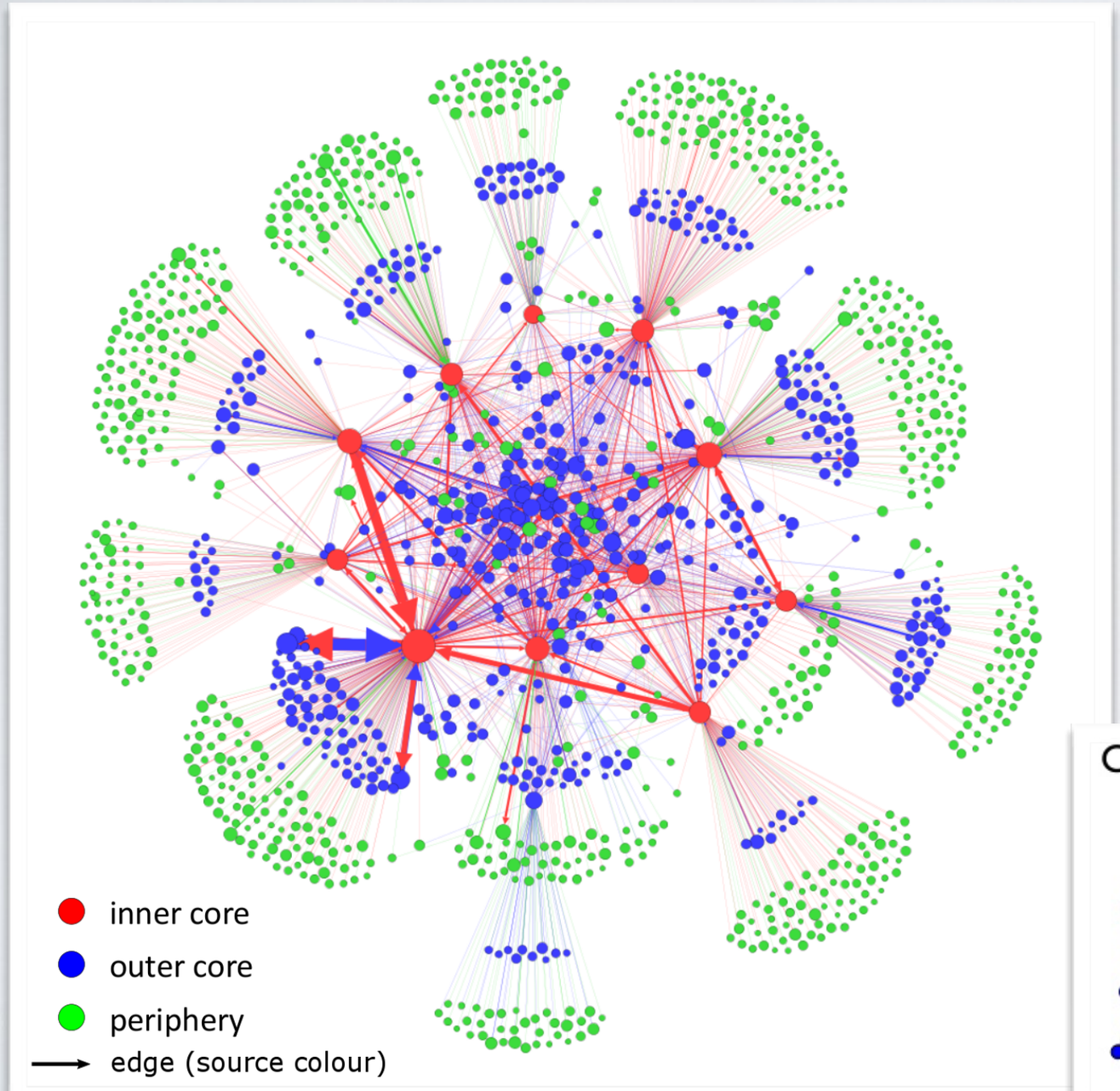
- Obvious weakness: communities can be very very far from random networks

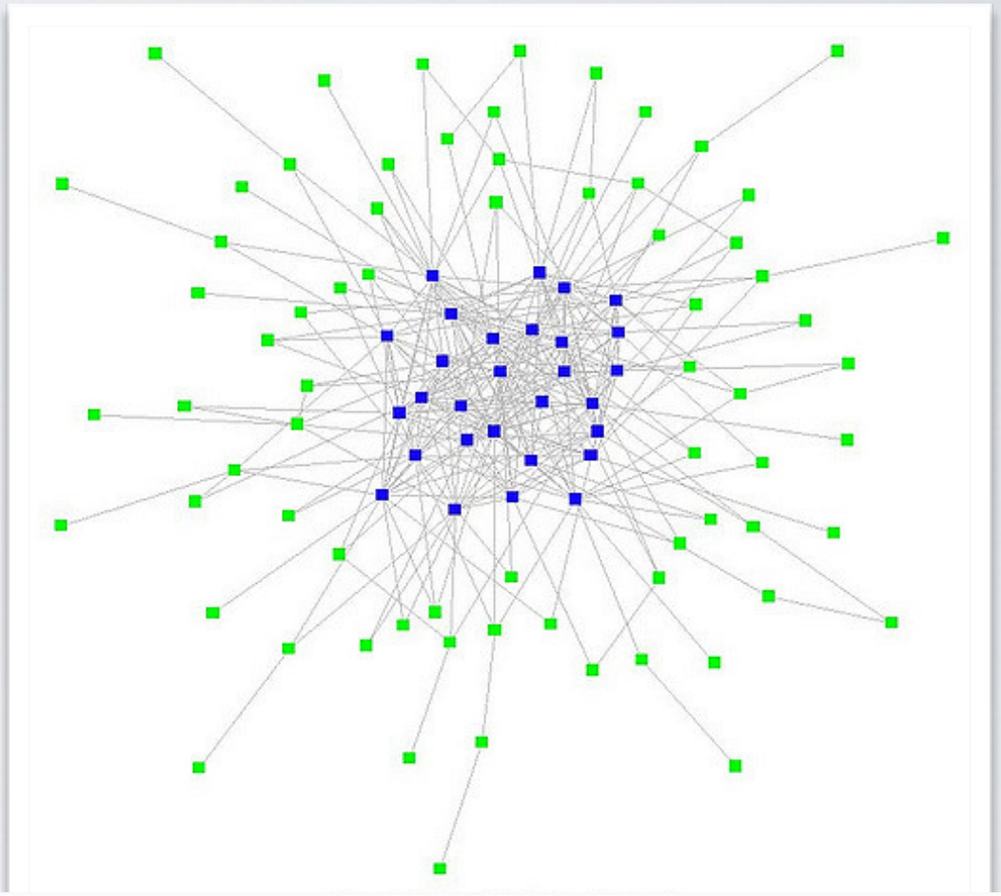


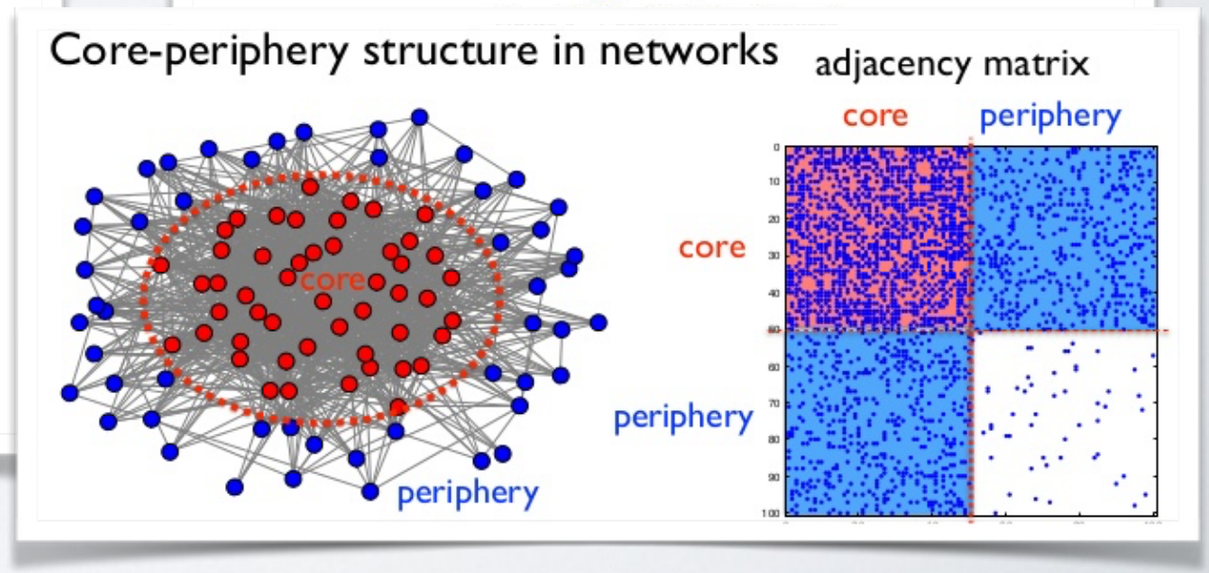
OVERLAPPING COMMUNITIES

- Another approach I like (many algorithms)
- Each community is defined intrinsically.
  - Must verify a property
  - Try to add and remove randomly nodes
  - Until the property is maximized.
  - Natural overlap, low complexity
  - Problem: which property ?

CORE-PERIPHERY





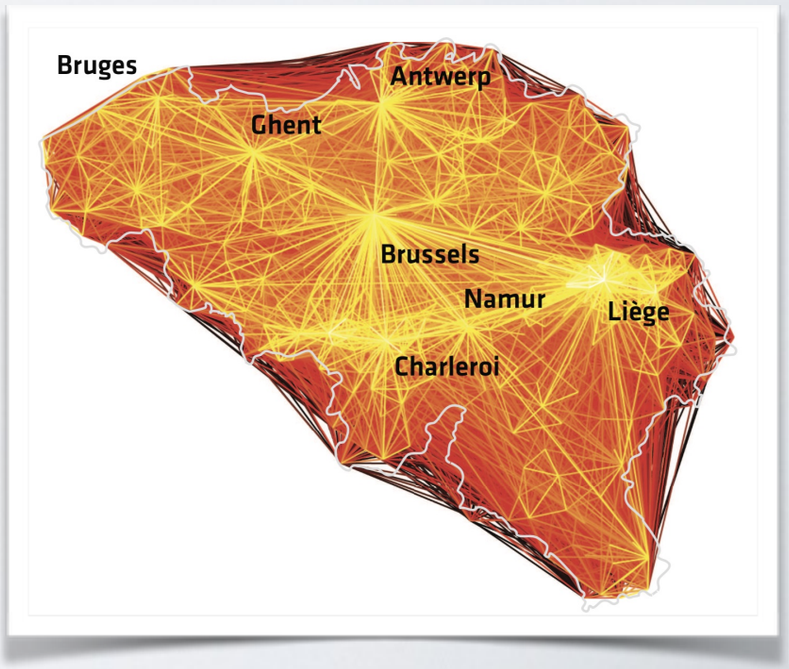


CORE-PERIPHERY

- Problem similar to communities:
- Concept easy to grasp
- Observed empirically in networks
- But how to define it formally?
- Main ideas:
  - Notion of decreasing density
    - Fuzzy
    - K-shells
  - Flow: need to go through core to communicate between periphery

SPATIAL NETWORKS

- Consider the network of telecommunications between cities
- The number of communications can be modeled as:
  - Population C1 * Population C2 / d^2
- This effect is very common on spatial networks
- It means a strong meso-scale organization, without need of community structures or core-periphery.

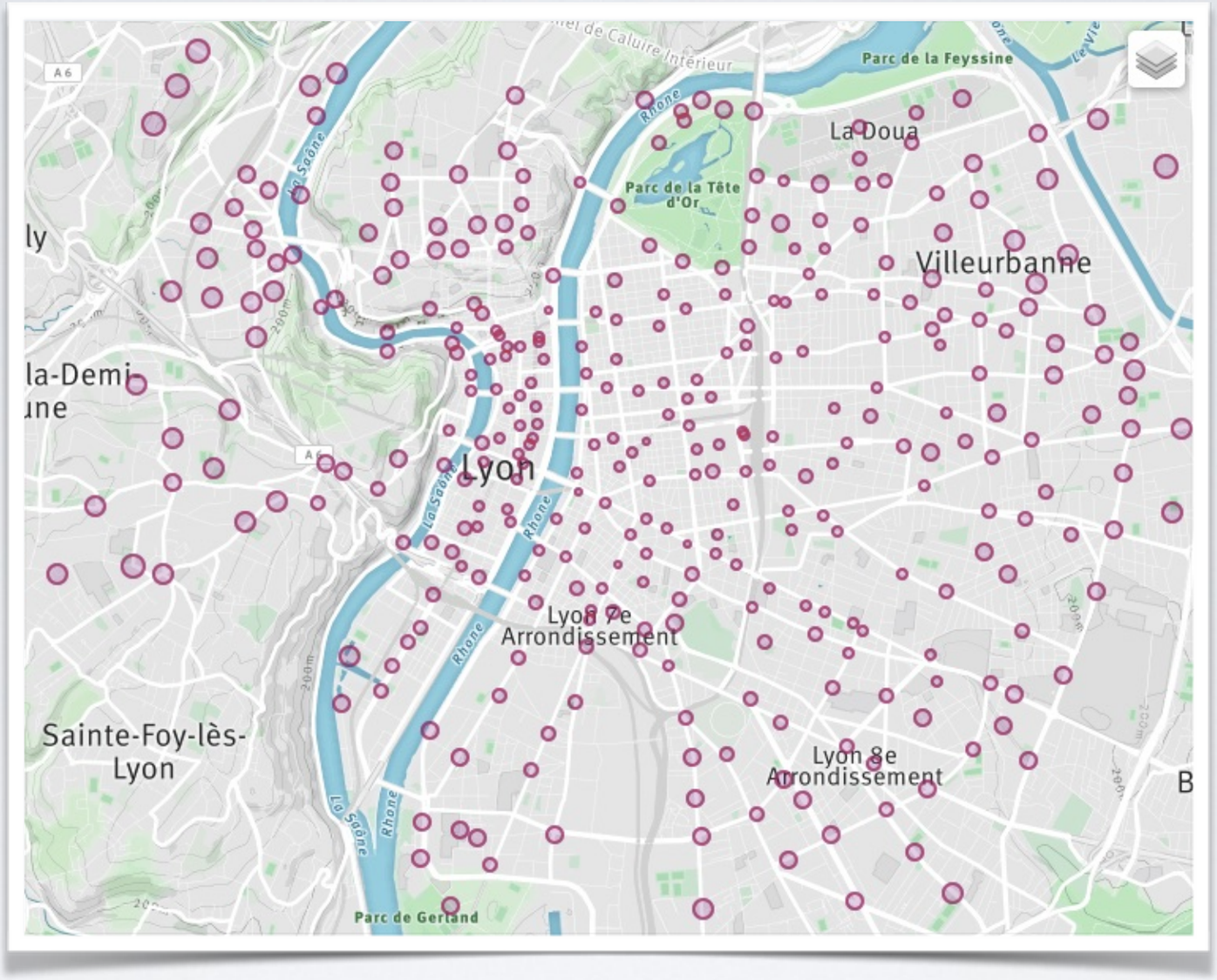


SPATIAL NETWORKS

Bicycle Sharing System (BSS) in Lyon

Dataset: trips (5y) + sociodemographic around stations

SPATIAL NETWORKS



Nodes: station (2D position)
Edges: number of trips over a period

SPATIAL NETWORKS

- Gravity with custom deterrence function

- #trips between any pair of station depends on their "popularity" and their distance.

- Distance influence learnt from data

$$p_{ij}^{Grav2} = W k_i k_j f(d_{ij})$$

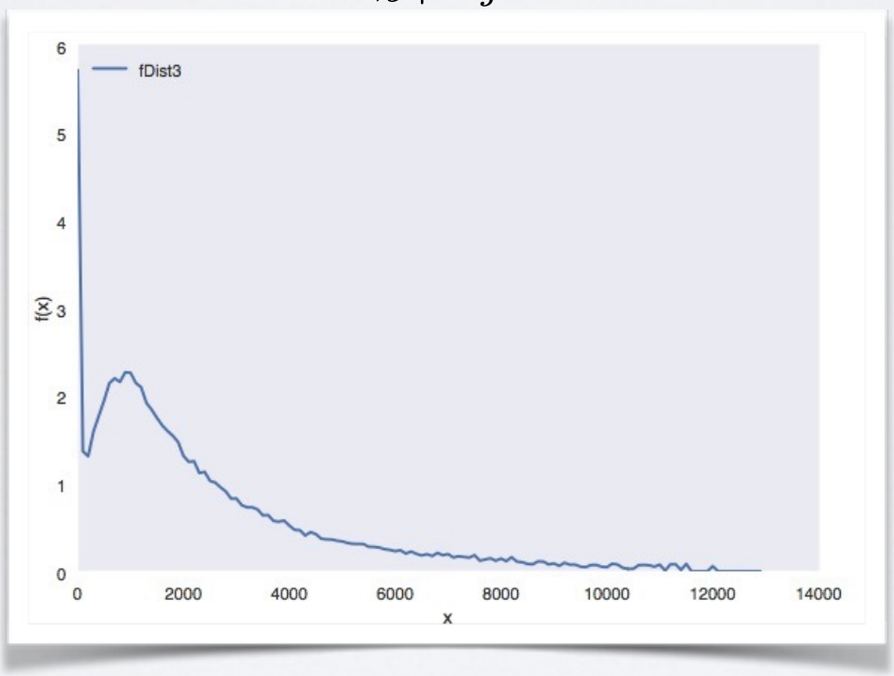
SPATIAL NETWORKS

Computation of a deterrence function:
Impact of distance on edge probability

(Comparing observation with Configuration Model)

$$f(d) = \frac{\sum_{i,j|d_{ij}=d} A_{ij}}{\sum_{i,j|d_{ij}=d} N_i N_j}$$

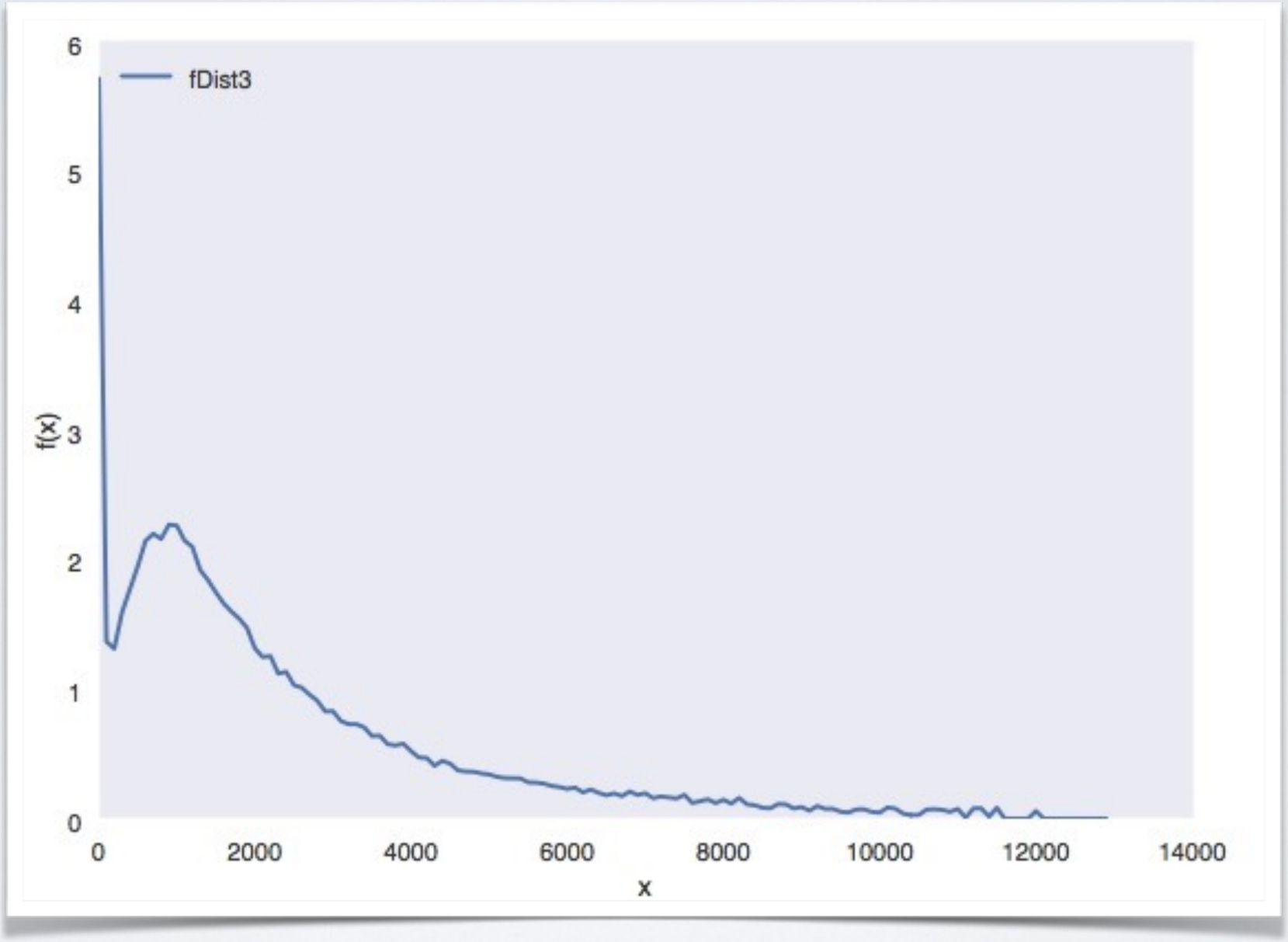
f(d)



Distance *d (meters)*

SPATIAL NETWORKS

f(d)

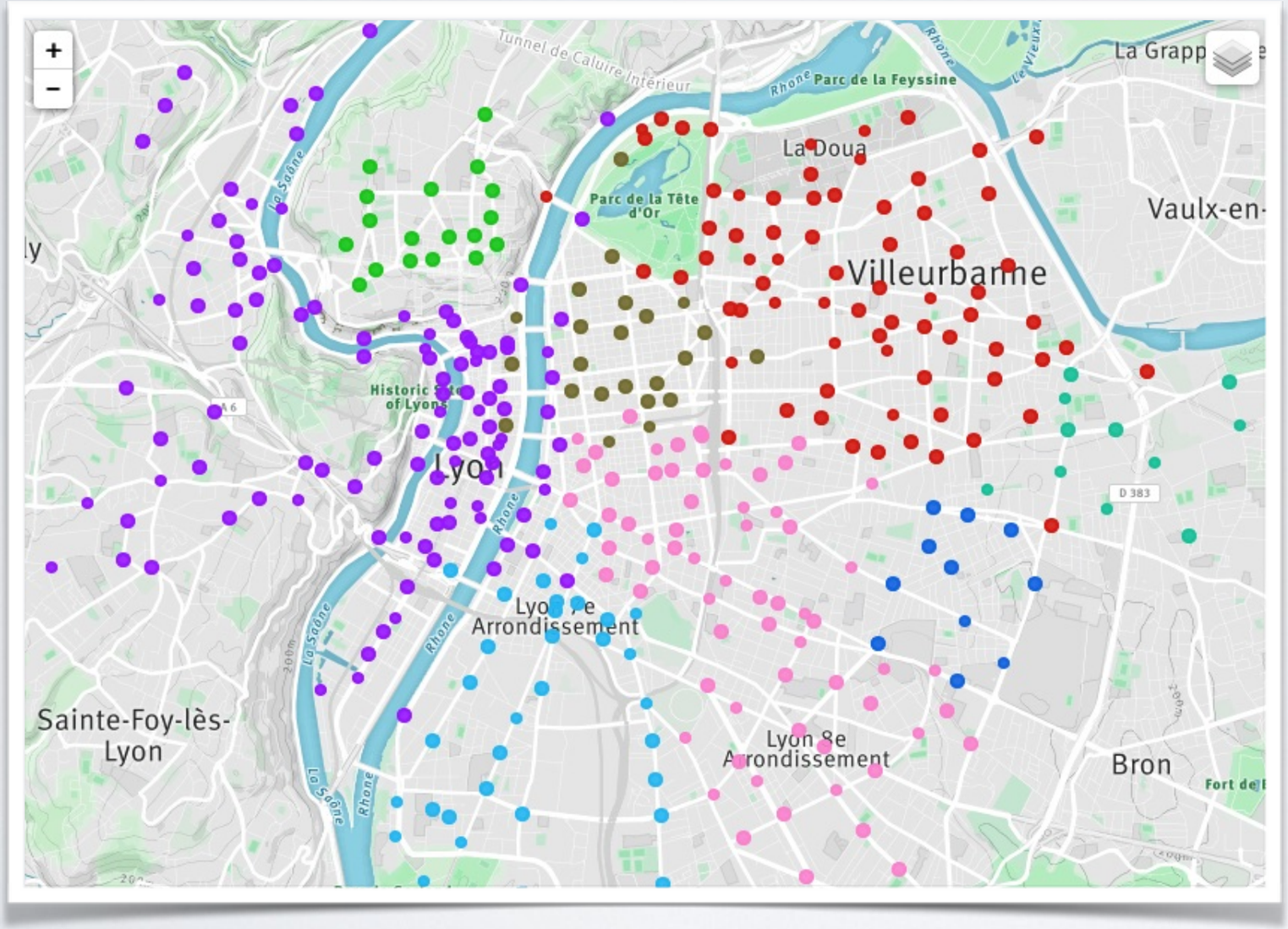


Distance *d* *(meters)*

COMBINING MESO-SCALE

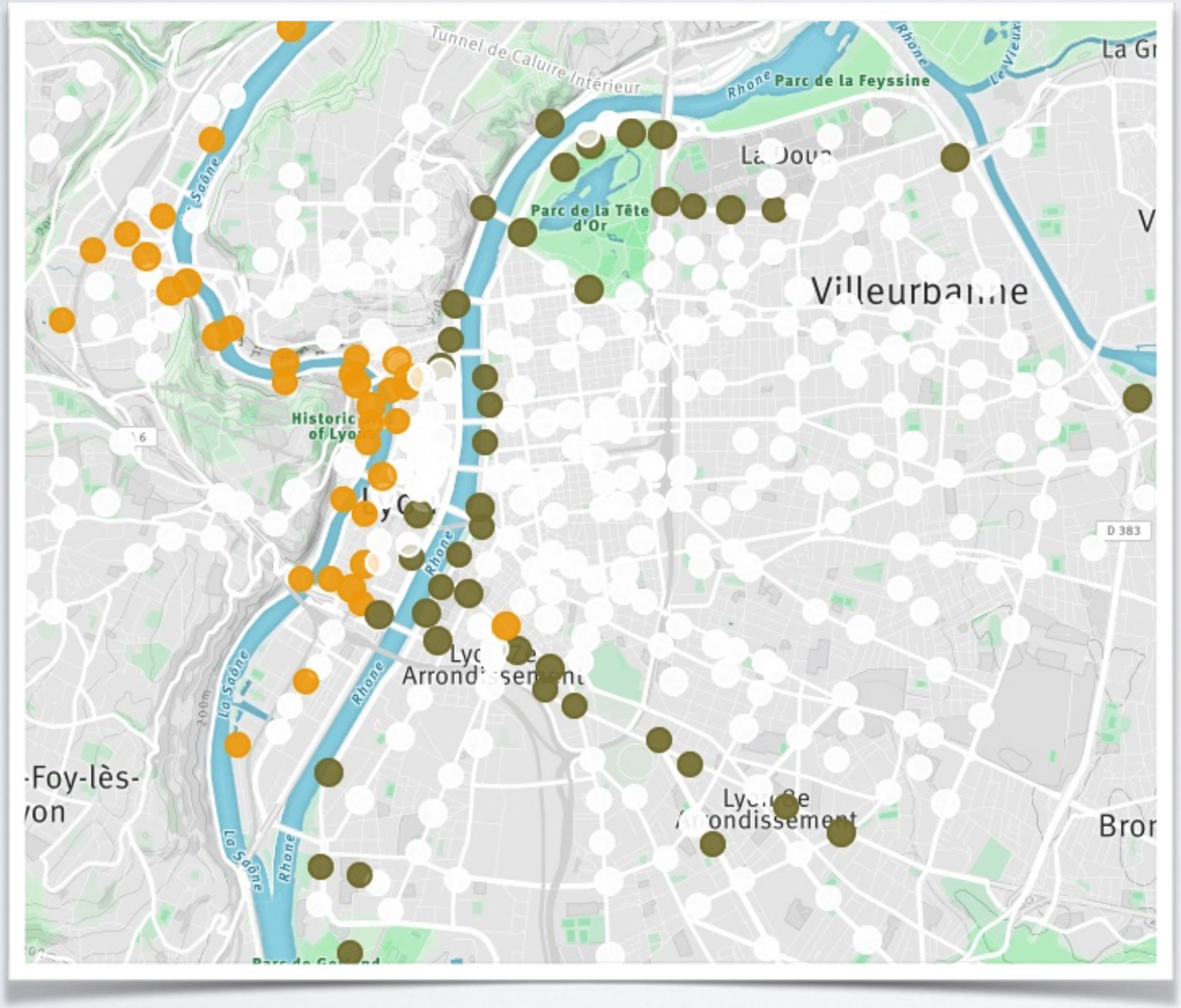
- 1)Compute the spatial model
- 2)Remove the effect of space
- 3)Use community detection to discover communities that were previously hidden by spatial effects

COMMUNITY STRUCTURE



Community Structure of the Original Network

COMMUNITY STRUCTURE



Community
Structure
Of trips
Unexplained
By Spatial Model

PRACTICALS

PRACTICALS

- 1)Synthetic networks
  - Using networkx, generate synthetic networks (planted partitions or LFR) of increasingly well defined community structure
  - Use several algorithms on them
  - Compare partitions to the ground truth and between them
    - (method “adjusted_mutual_info_score" from library “sklearn"

PRACTICALS

- 2)On your favorite network, detect communities
  - Compare communities found by several algorithms
  - Number and size ?
  - Search for stable parts ?
  - Study the relation of nodes with communities. Are there some nodes that have strong relations with several ? Do you think that overlaps are relevant for your network ?

PRACTICALS

- 3)On the airport dataset, find communities, and create the *induced* network (each community becomes a node, weight of edges=number of original links
  - Give automatically a name to communities, by combining properties of nodes of higher degree inside the community
  - Visualize this network
  - Is it different with another algorithm ?
  - What happens if you run an algorithm on the induced graph?
    - And if you run an algorithm on a single community seen as a graph?

PRACTICALS

- Networkx has few community detection algorithms.
- You can find algorithms outside:
  - Louvain: https://github.com/taynaud/python-louvain
  - Infomap: https://github.com/mapequation/infomap/blob/master/examples/python/infomap-examples.ipynb
  - SBM: https://graph-tool.skewed.de