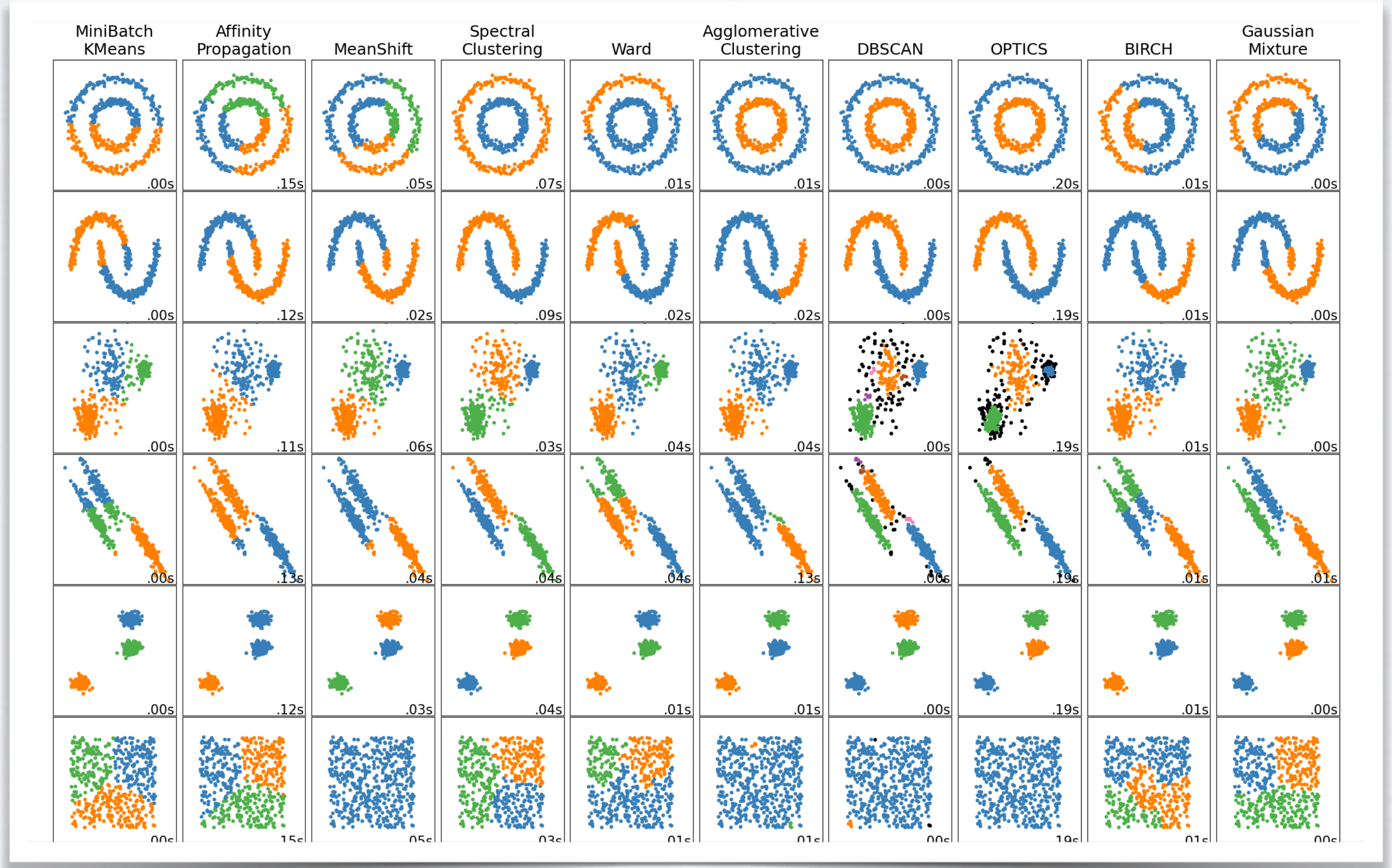
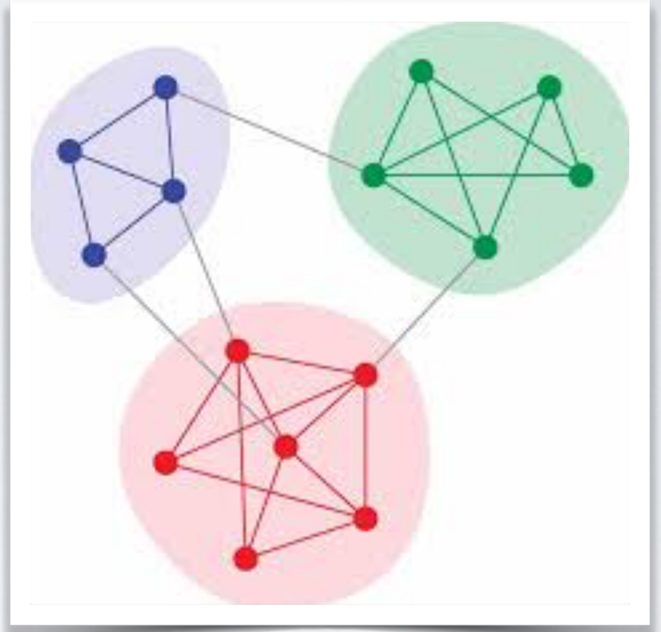
COMMUNITY DETECTION (GRAPH CLUSTERING)

COMMUNITY DETECTION

- Community detection is equivalent to "clustering" in unstructured data
- Similar problems: what is a good community ?



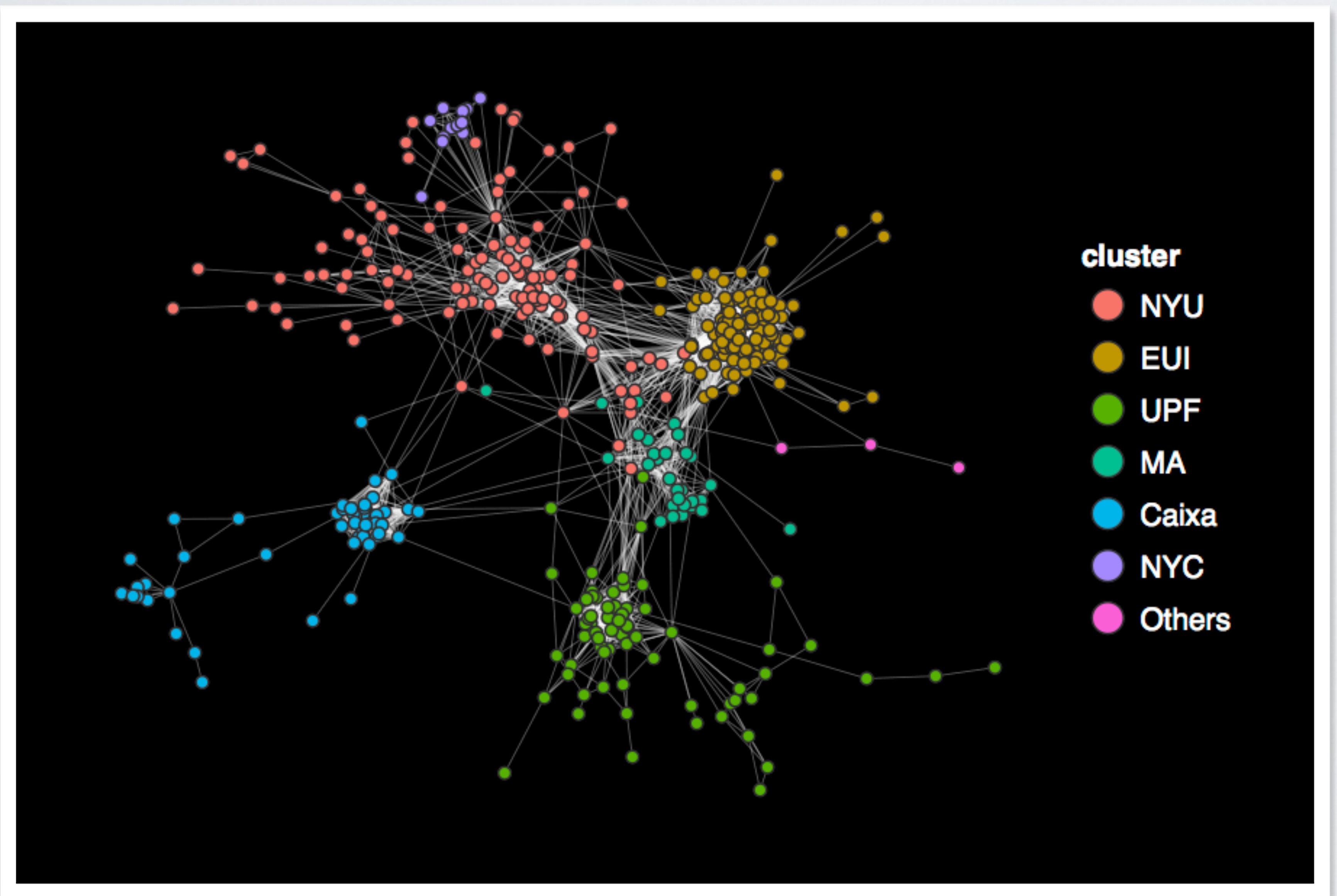
COMMUNITY DETECTION



- Community detection:
  - Find groups of nodes that are:
    - Strongly connected to each other
    - Weakly connected to the rest of the network
    - Ideal form: each community is 1)A clique, 2) A separate connected component
  - No formal definition
  - Hundreds of methods published since 2003

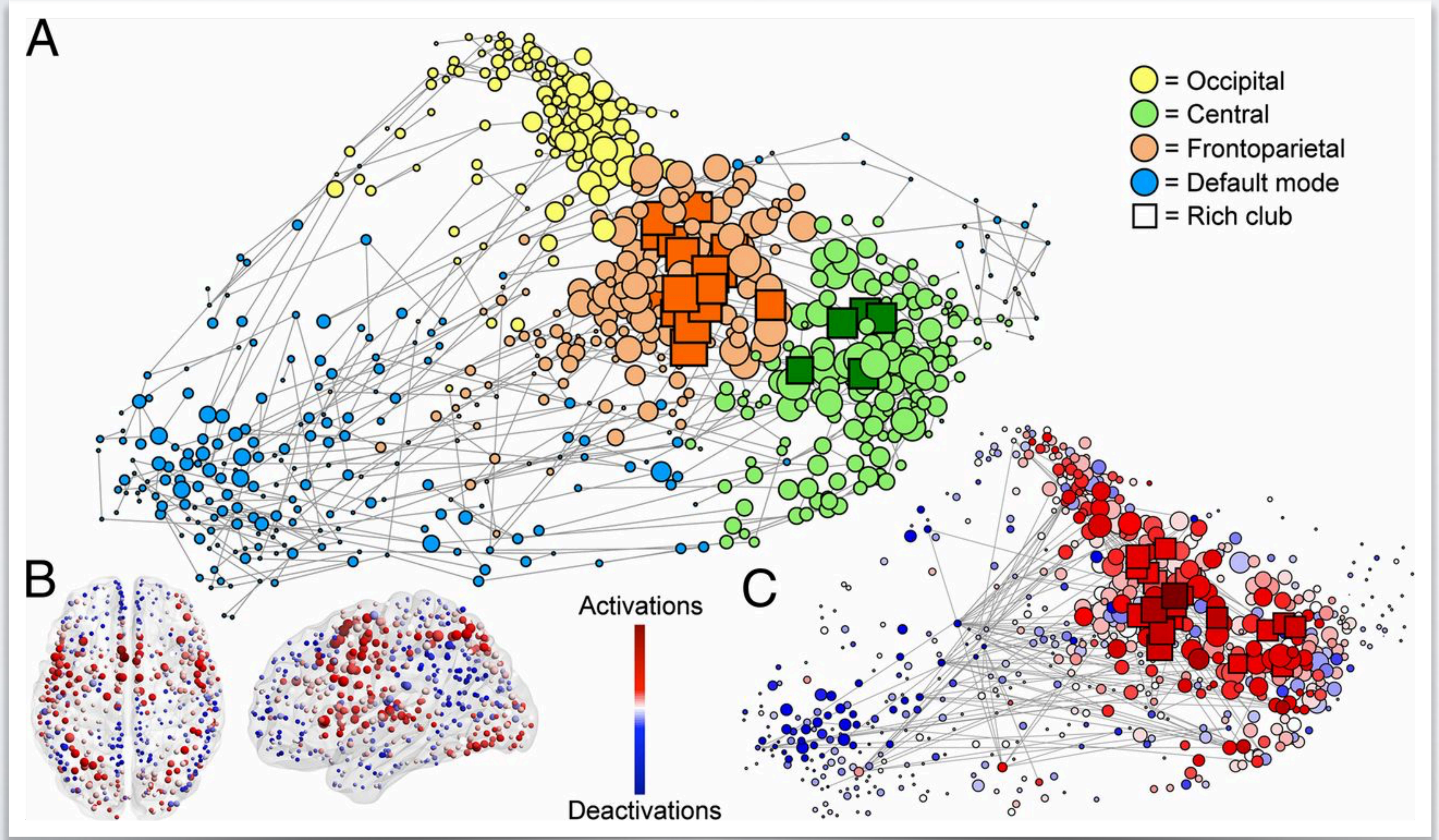
COMMUNITY STRUCTURE IN REAL GRAPHS

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



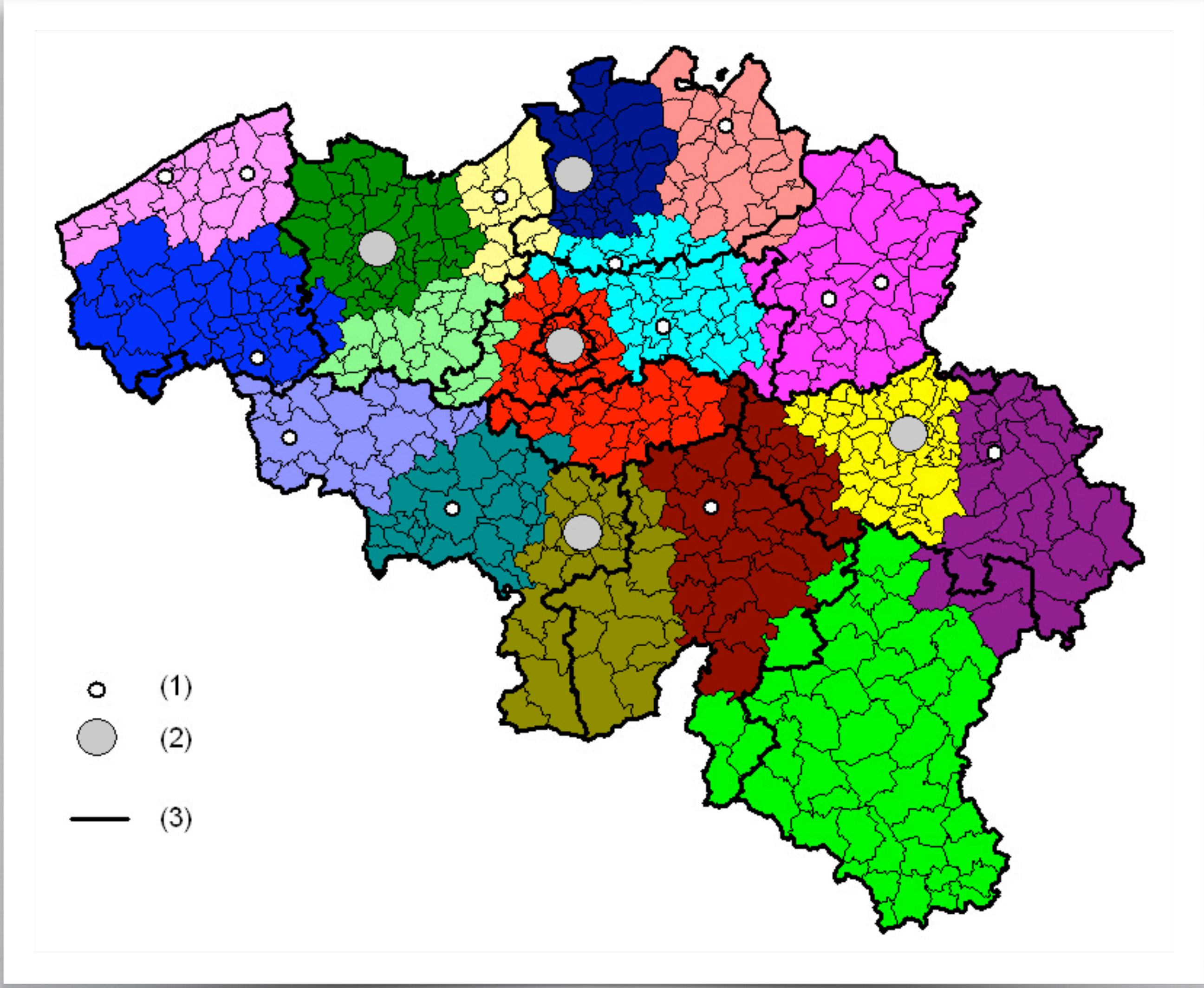
COMMUNITY STRUCTURE IN REAL GRAPHS

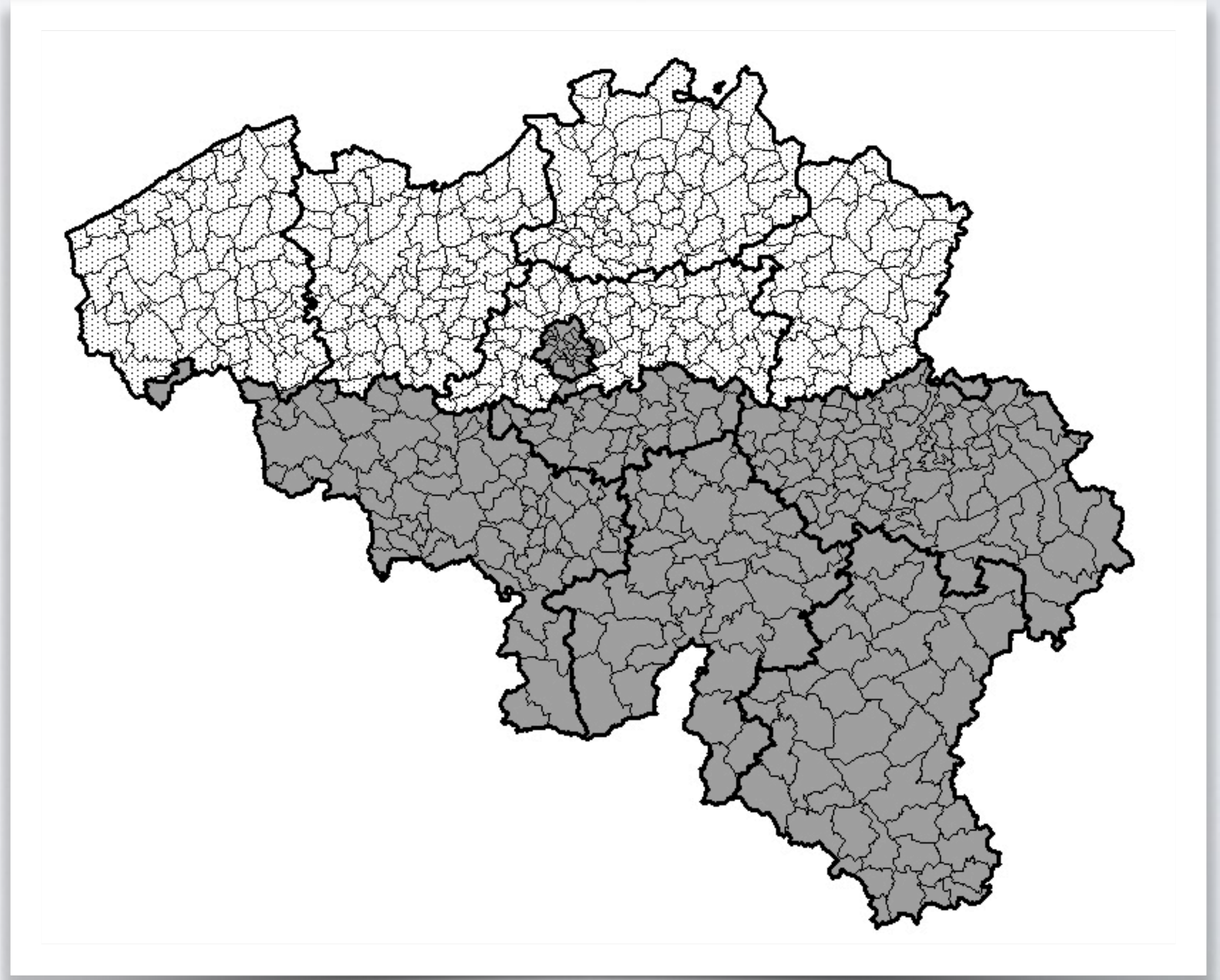
- Connections in the brain ?



COMMUNITY STRUCTURE IN REAL GRAPHS

- Phone call communications in Belgium ?

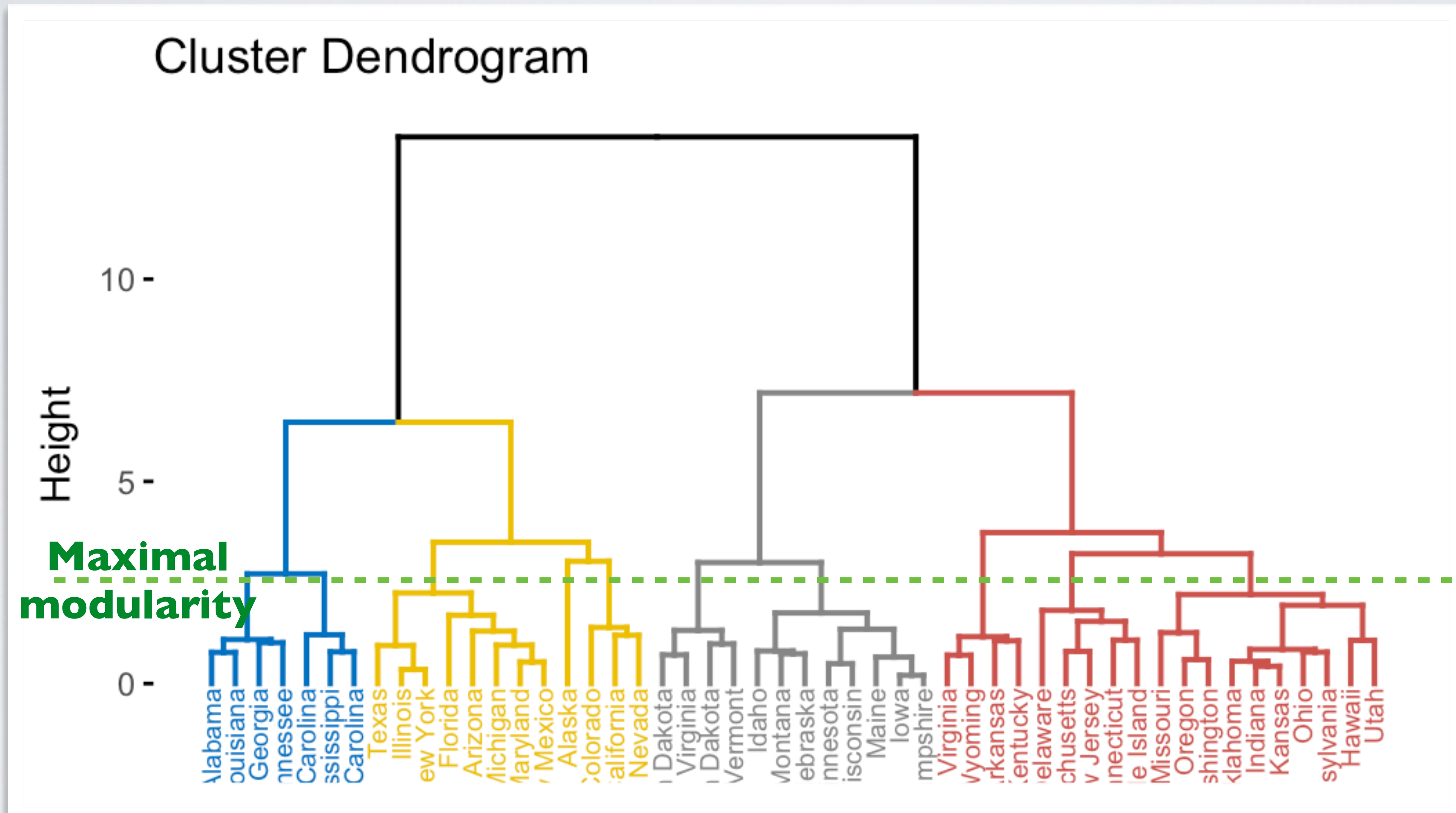




FIRST METHOD BY GIRVAN & NEWMAN

- 1)Compute the betweenness of all edges
- 2)Remove the edge of highest betweenness
- 3)Repeat until all edges have been removed
  - Connected components are communities
- => 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

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

Original formulation

MODULARITY

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

Sum over all pairs of nodes

MODULARITY

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

1 if in same community

MODULARITY

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

1 if there is an edge between them

MODULARITY

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

Probability of an edge in
a configuration model
(Edges at random, keeping degrees)

MODULARITY

Can also be defined
as a sum by community

$$Q = \frac{1}{L}\sum_{i=1}^{|C|}(L_i - \frac{1}{2}K_i^2)$$

with $L_i = L(H(c_i))$ the number of edges inside community i and $K_i = \sum_{u \in c_i} k_u$ the sum of degrees of nodes in community i.

MODULARITY

- Modularity compares the observed network to a **null model**
  - Usually the configuration model
    - Multi-edges and loops are allowed
  - Other models could be used, such as ER random graphs.

- Natural extension to weighted/multi-edge networks

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 MAXIMIZATION

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

LOUVAIN ALGORITHM

- Simple, greedy approach
  - Easy to implement
  - Fast
- Yields a hierarchical community structure
- Beat state of the art on all aspects (when introduced)
  - Speed
  - Max modularity obtained
  - Do not fall in some traps (see later)

LOUVAIN ALGORITHM

- Each node start in its own community

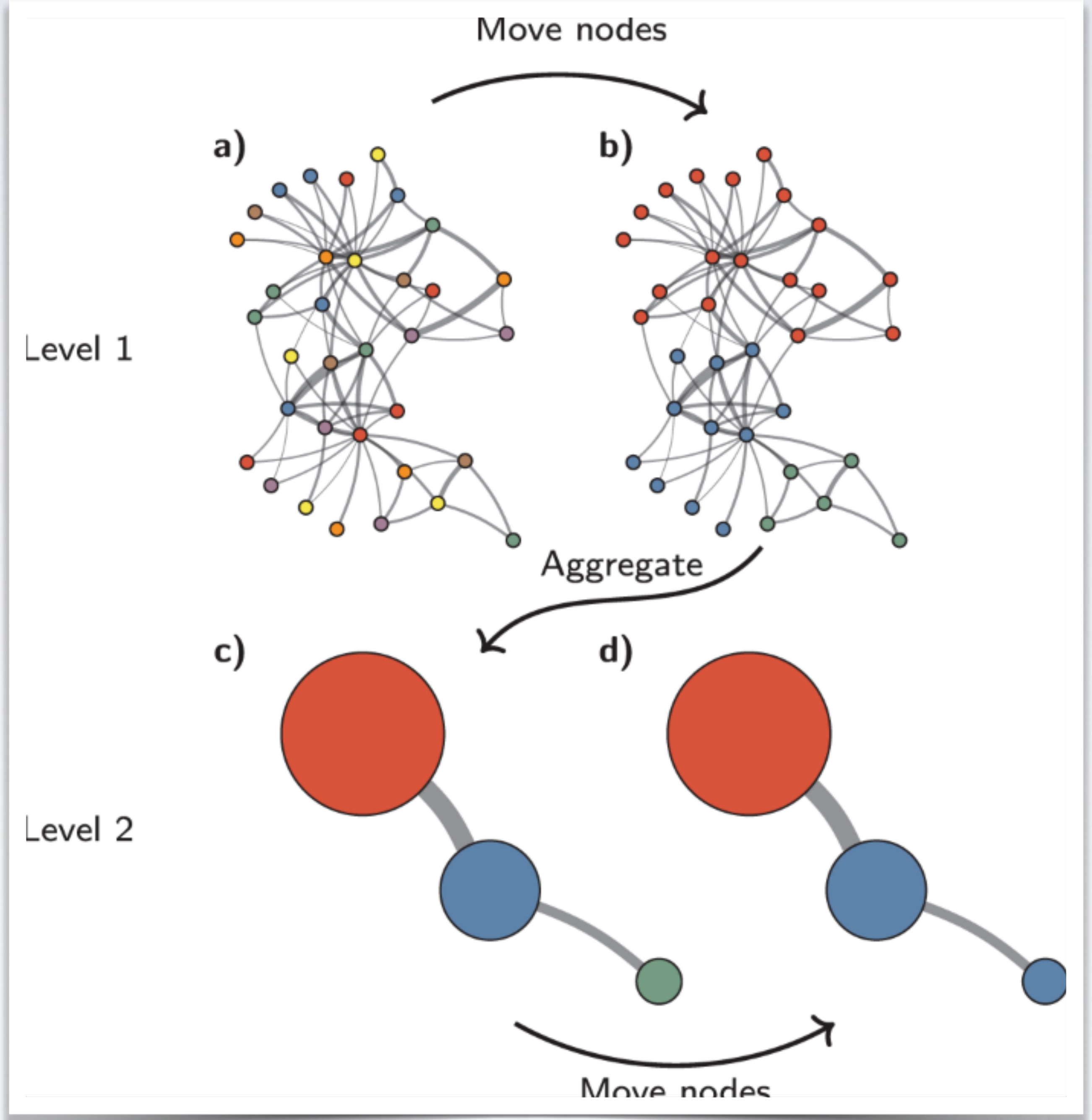
- Repeat until convergence
  - FOR each node:
    - FOR each neighbor:
      if adding node to its community increase modularity, do it

- 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

Blondel, Vincent D., et al. "Fast unfolding of communities in large networks." *Journal of statistical mechanics: theory and experiment* 2008.10 (2008): P10008.

LOUVAIN ALGORITHM



Blondel, Vincent D., et al. "Fast unfolding of communities in large networks." *Journal of statistical mechanics: theory and experiment* 2008.10 (2008): P10008.

ALTERNATIVES

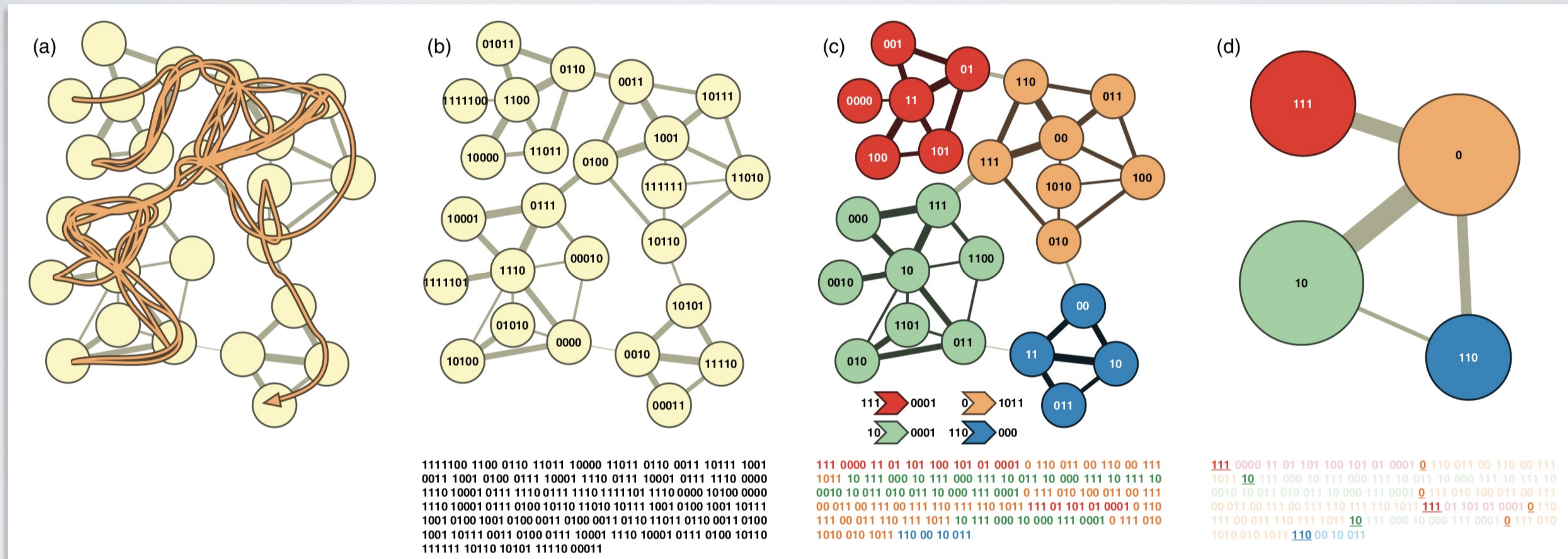
- Most serious alternatives
  - Infomap (based on information theory —compression)
  - Stochastic block models (bayesian inference)
- These methods have a clear definition of what are good communities. Theoretically grounded

INFOMAP

- [Rosvall & Bergstrom 2009]
- Find the partition minimizing the *description* of any *random walk* on the network
- We want to *compress* the description of random walks

Rosvall, Martin, and Carl T. Bergstrom. "Maps of random walks on complex networks reveal community structure." *Proceedings of the National Academy of Sciences* 105.4 (2008): 1118-1123.

INFOMAP



Random walk

Description Without Communities

With communities

Huffman coding: short codes for frequent items
Prefix free: no code is a prefix of another one (avoid fix length/separators)

The Infomap method

Finding the optimal partition M:

- Shannon's source coding theorem (Shannon's entropy)

for a probability distribution P = {p_i} such that $\Sigma_i p_i = 1$, the lower limit of the per-step code-length is

$$L(\mathcal{P}) = H(\mathcal{P}) \equiv -\sum_i p_i \log p_i$$

- Minimise the expected description length of the random walk

Sum of Shannon entropies of multiple codebooks weighted by the rate of usage

probability of between modules movements of a RW, i.e. the rate of usage of the index codebook

probability of within modules movements of a RW, i.e. the rate of usage of the module codebook

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

Expected decryption length of partition M

Entropy of movement between modules, i.e. the frequency weighted average length of codewords

Entropy of movement inside modules, i.e. the frequency weighted average length of codewords in the module codebook

Algorithm

1. Compute the fraction of time each node is visited by the random walker (Power-method on adjacency matrix)
2. Explore the space of possible partitions (deterministic greedy search algorithm - similar to Louvain but here we join nodes if they decrease the description length)
3. Refine the results with simulated annealing (heat-bath algorithm)

INFOMAP

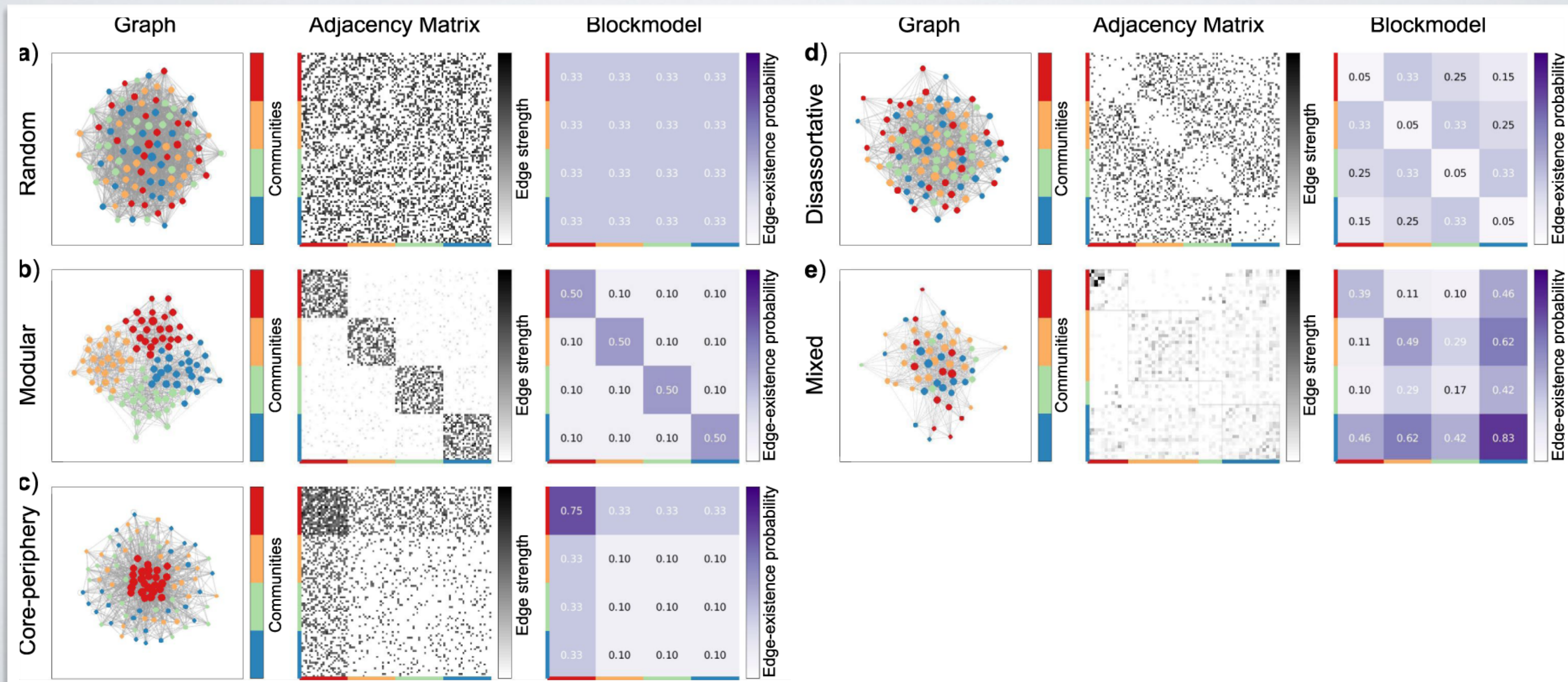
- To sum up:
  - Infomap defines a *quality function* for a partition different than modularity
  - Any algorithm can be used to optimize it (like Modularity)
- Advantage:
  - Infomap can recognize random networks (no communities)

STOCHASTIC BLOCK MODELS

- Stochastic Block Models (SBM) are based on statistical models of networks
- They are in fact more general than usual 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:
  - Associative SBM: density inside nodes of a same communities >> density of pairs belonging to different communities.



STOCHASTIC BLOCK MODELS

- General idea of SBM community detection:
  - Specify the desired number of cluster
  - Find parameters to optimize the maximum likelihood
    - Principle: The best parameters are those that allow to generate the observed network with the highest probability
- Main weakness of this approach
  - Number of clusters must be specified (avoid trivial solution)
- MDL (Minimum Description Lenght) approaches exist to automatically find the number of blocks

EVALUATION OF COMMUNITY STRUCTURE

EVALUATION

- Similar to clustering:
  - Intrinsic/Internal evaluation
    - Partition quality function
    - Individual Community quality function
  - Comparison of observed communities and expected communities
    - Synthetic networks with community structure
    - Real networks with Ground Truth

INTRINSIC EVALUATION

INTRINSIC EVALUATION

- Partition quality function
  - Already defined: **Modularity**, **graph compression**, etc.
- Quality function for individual community
  - Internal Clustering Coefficient
  - Conductance: $\frac{|E_{out}|}{|E_{out}|+|E_{in}|}$
    - Fraction of external edges

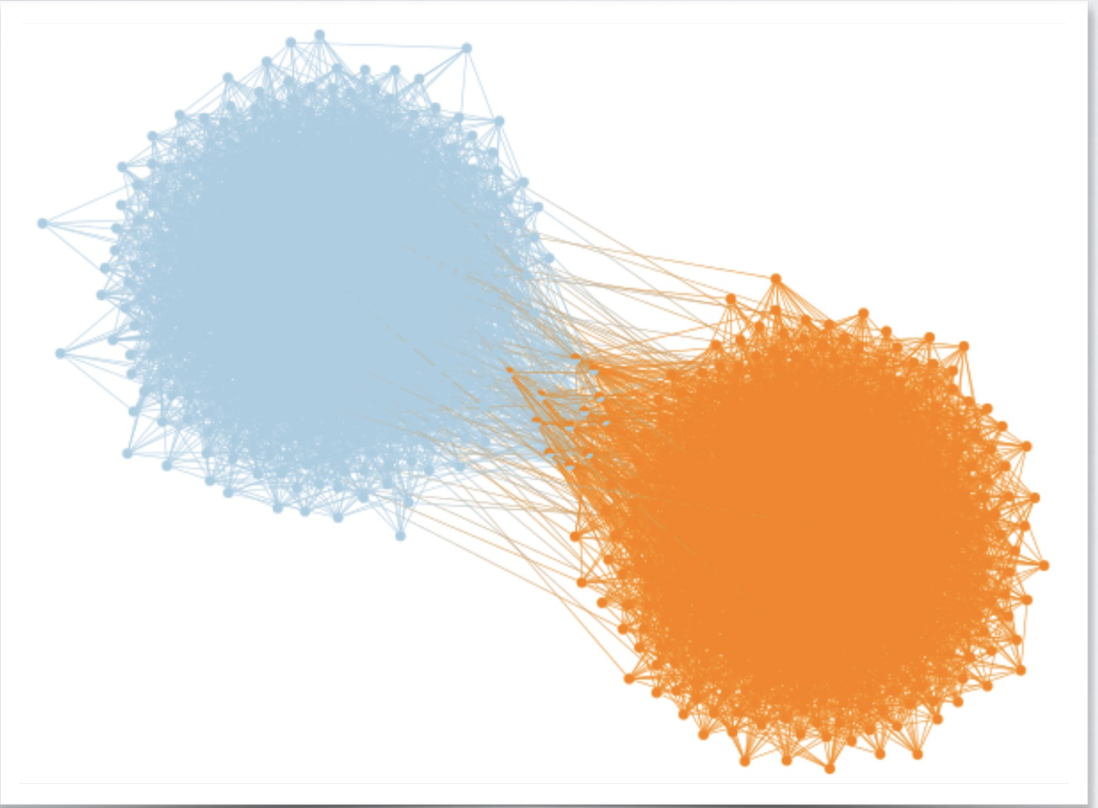
$|E_{in}|, |E_{out}|$:
# of links to nodes inside (respectively, outside) the community

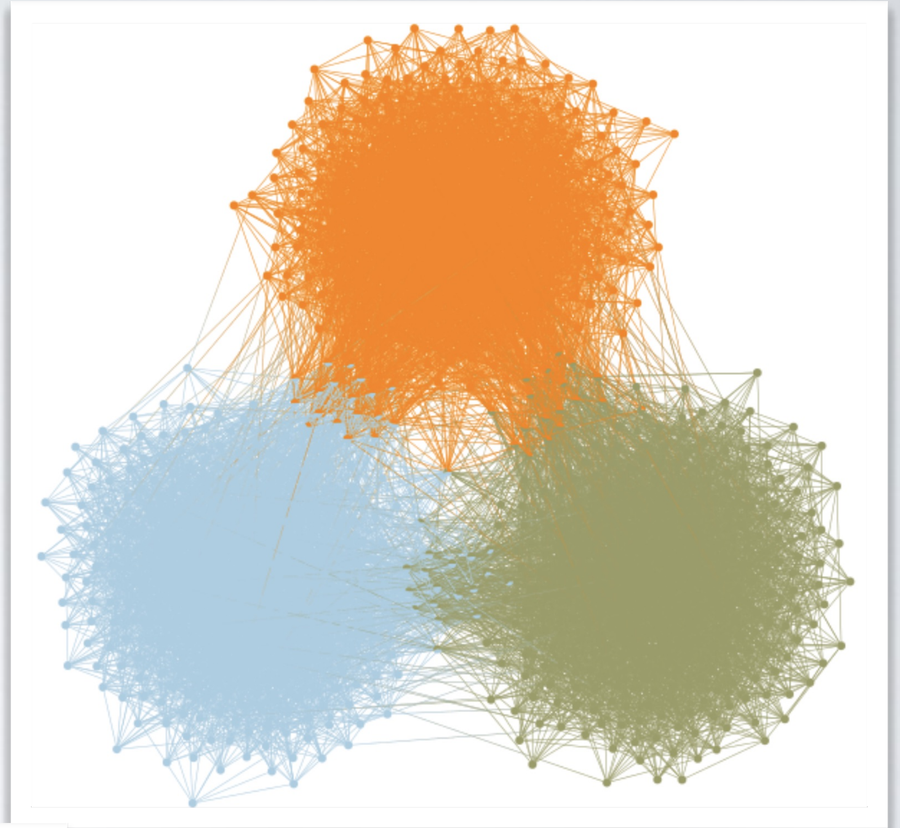
COMPARISON WITH GROUND TRUTH

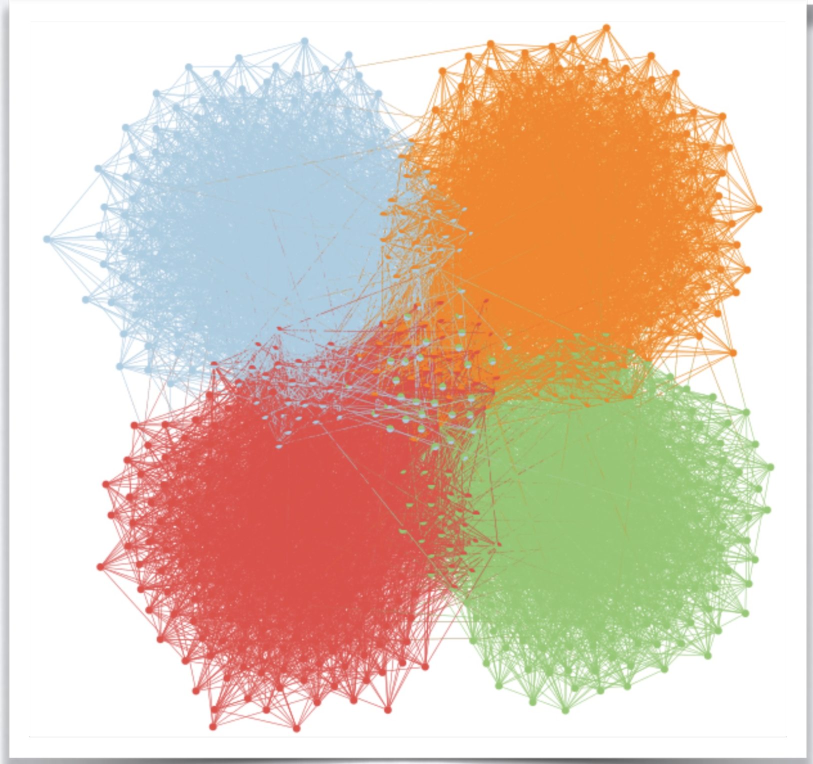
SYNTHETIC NETWORKS

- 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 ad-hoc process (sample values from distributions)

SYNTHETIC NETWORKS



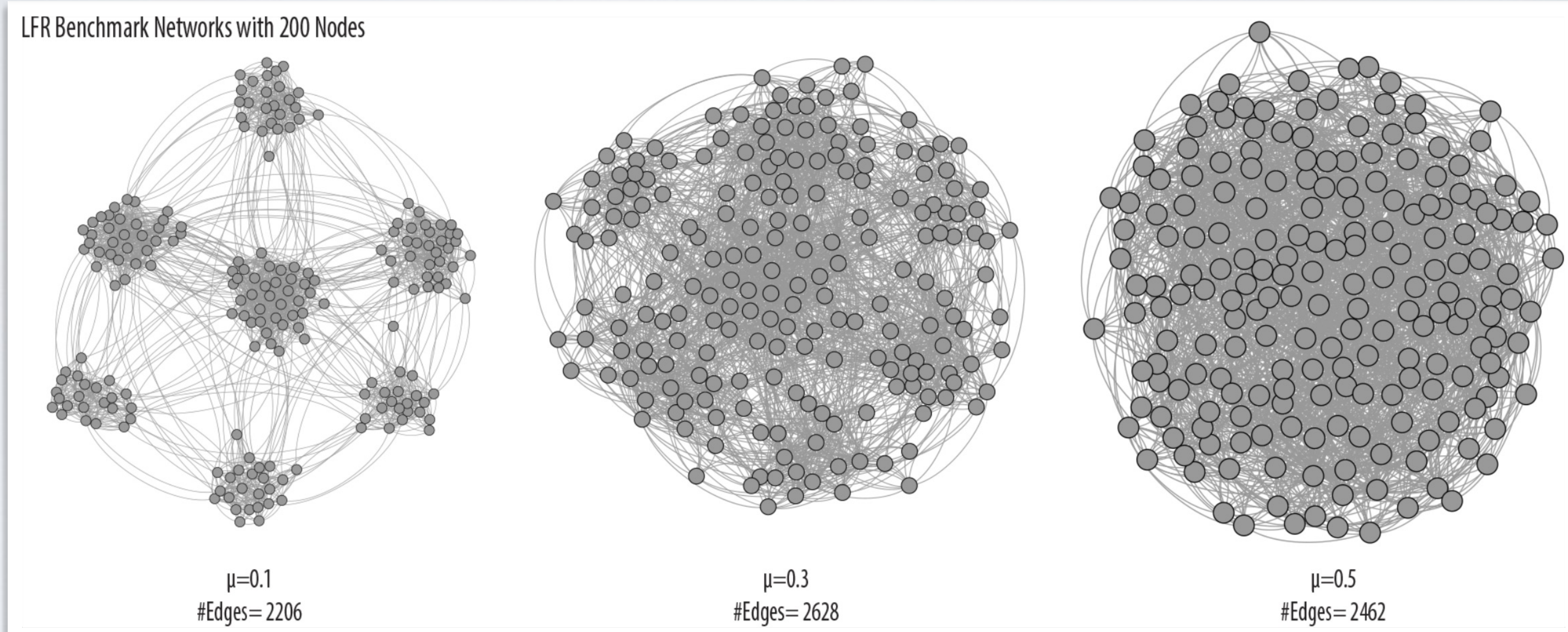




SYNTHETIC NETWORKS

- 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 external edges of each node
  - Varying the mixing parameter makes community more or less well defined
- Currently the most popular

SYNTHETIC NETWORKS



OTHER TYPES OF COMMUNITIES

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
- Difference between "attributes" and overlapping communities ?
  - Community of Women, Community of 17-19yo, Community of fans of…

HIERARCHICAL COMMUNITIES

Lancichinetti, Andrea, et al. "Finding statistically significant communities in networks." *PloS one* 6.4 (2011): e18961.

SUPERVISED MACHINE LEARNING 1:
LINK PREDICTION

LINK PREDICTION

- Do you know why Facebook "People you may know" is so accurate?
- How youtube/Spotify/amazon recommend you the right item?
- =>Link prediction
  - More generally, recommendation, but link prediction is a popular way to do it

LINK PREDICTION

- Observed network: current state
- Link prediction: What edge
  - Might appear in the future (*future link* prediction)
  - Might have been missed (*missing link* prediction)

LINK PREDICTION

- Overview:

- Link prediction based on network structure:
  - Local: High clustering (friends of my friends will become my friends)
  - Global: Two unrelated hubs more likely to have links that unrelated small nodes
  - Meso-scale organisation: different edge probability for nodes in different communities/blocks

- Link prediction can also be based on node properties
  - e.g., age, revenue, genre, etc.
  - Combining with usual machine learning, outside of the scope of this course

FIRST APPROACH TO LINK PREDICTION:

HEURISTIC BASED

(HEURISTICS, NOT SUPERVISED MACHINE LEARNING)

HEURISTICS

- Network science experts can design **heuristics** to predict where new edge might appear/be missing

- Principle: design a score based on network topology f(v1,v2) which, given two nodes, express their likeliness of being connected (if they aren't already)
  - Common neighbors
  - Jaccard coefficient
  - Hub promoted
  - Adamic Adar
  - Ressource allocation
  - Community based

Zhou, T., Lü, L., & Zhang, Y. C. (2009). Predicting missing links via local information. *The European Physical Journal B*, *71*(4), 623-630.

COMMON NEIGHBORS

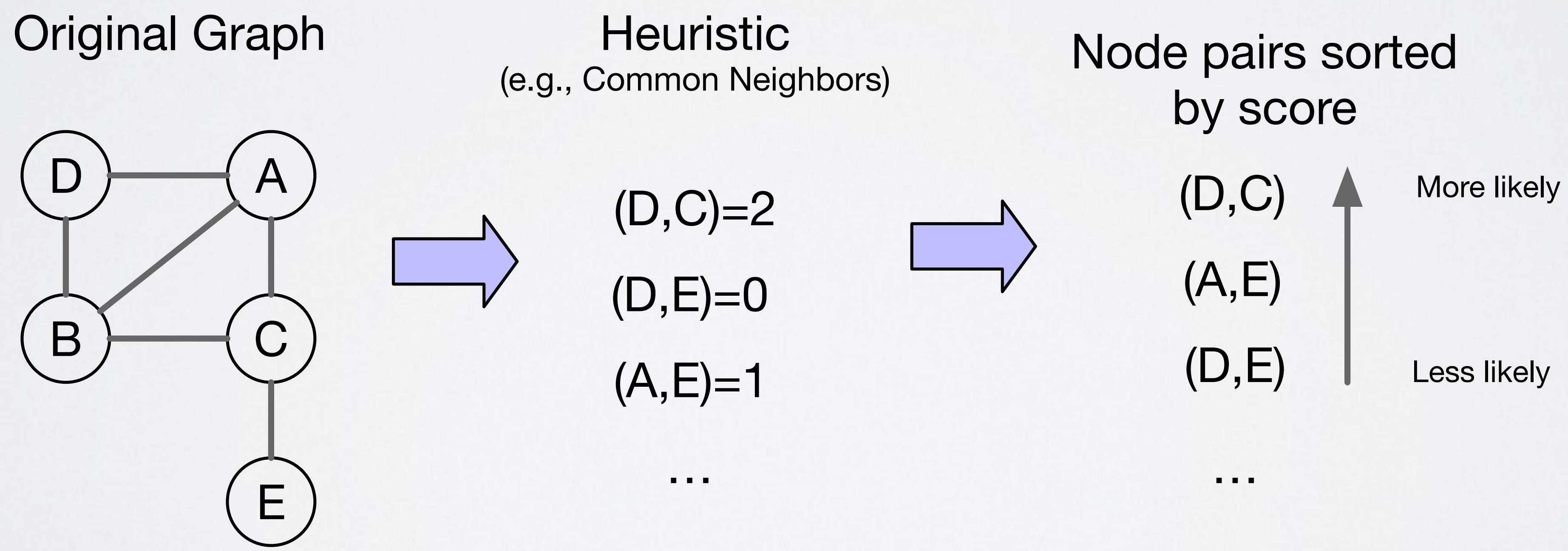
- "Friends of my friends are my friends"
- High clustering in most networks
- =>The more friends in common, the highest probability to become friends

$$\mathrm{CN}(x, y) = |\Gamma(x) \cap \Gamma(y)|$$

$\Gamma(x)$ = Neighbors of x

PREDICTION

- How to predict links based on Common Neighbors (CN)?



JACCARD COEFFICIENT

- Used in many applications:
  - Measure of similarity of sets of different sizes

$$\mathrm{JC}(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}$$

- Intuition:
  - Two people who know only the same 3 people
    - =>high probability
  - Two people who know 1000 people, only 3 in commons
    - =>Lower probability

HUB PROMOTED

- Intuition:
  - Normalized by total neighbors
  - But also the relation can be asymmetric

  - Two stars have 10 common followers or I have ten friends following a star

$$\mathrm{HP}(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{min(|\Gamma(x)|, |\Gamma(y)|)}$$

ADAMIC ADAR

- Intuition:
  - For previous scores: all common nodes are worth the same
  - For AA:
    - A common node with ONLY them in common is worth the most
    - A common node connected to everyone is worth the less
    - The higher the size of its neighborhood, the lesser its value

$$\mathrm{AA}(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log|\Gamma(z)|}$$

RESSOURCE ALLOCATION

- Similar to Adamic Adam, penalize more higher degrees

$$\mathrm{RA}(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{|\Gamma(z)|}$$

$$\mathrm{AA}(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log|\Gamma(z)|}$$

PREFERENTIAL ATTACHMENT

- Preferential attachment:
  - Every time a node join the network, it creates a link with nodes with probability proportional to their degrees
  - In fact, closer to the definition of the configuration model

- Score not based on common neighbors
  - =>Assign different scores to nodes at network distance >2

- Intuition: Two nodes with many neighbors more likely to have new ones than nodes with few neighbors

$$\mathrm{PA}(x, y) = |\Gamma(x)| \cdot |\Gamma(y)|$$

OTHER SCORES

Examples of other scores proposed

Sorenson Index

$$\mathrm{SI}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x)| + |\Gamma(y)|}$$

Salton Cosine Similarity

$$\mathrm{SC}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{\sqrt{|\Gamma(x)| \cdot |\Gamma(y)|}}$$

Hub Depressed

$$\mathrm{HD}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{max(|\Gamma(x)|, |\Gamma(y)|)}$$

Leicht-Holme-Nerman

$$\mathrm{LHN}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x)| \cdot |\Gamma(y)|}$$

COMMUNITY STRUCTURE

- General idea:
  - 1)Compute community structure on the whole graph
  - 2)Assign high score for 2 nodes in a same community, a low score otherwise
- How to choose the score?

COMMUNITY STRUCTURE

- For methods based on a quality function optimization (Modularity, Infomap's information compression, etc.)
  - Assign a score to each pair proportional to the change in quality function associated with adding an edge between them

- For instance, Louvain optimize Modularity.
  - Each edge added between communities:
    - Decrease in the Modularity
  - Edge added inside community:
    - Increase in Modularity, depends on properties of the community and nodes

Ghasemian, A., Hosseinmardi, H., & Clauset, A. (2019). Evaluating overfit and underfit in models of network community structure. *IEEE Transactions on Knowledge and Data Engineering*.

OTHER SCORES

- Distance based:
  - Length of the shortest path
  - Probability to reach a node from another on a random-walk of distance *k*
    - See next class on embeddings
  - Number of paths of length *l* between the nodes
- Problem: computational complexity

WHICH ONE IS BEST?

- All scores but PA are based on common neighbors
- =>No links between nodes at graph distance >2
- Inconsistent with observations
- =>We should combine PA and others

ML APPROACH TO LINK PREDICTION:

SIMILARITY SCORE, SUPERVISED

SUPERVISED MACHINE LEARNING

- Use Machine Learning algorithms to **learn** how to combine heuristics for optimizing predictions

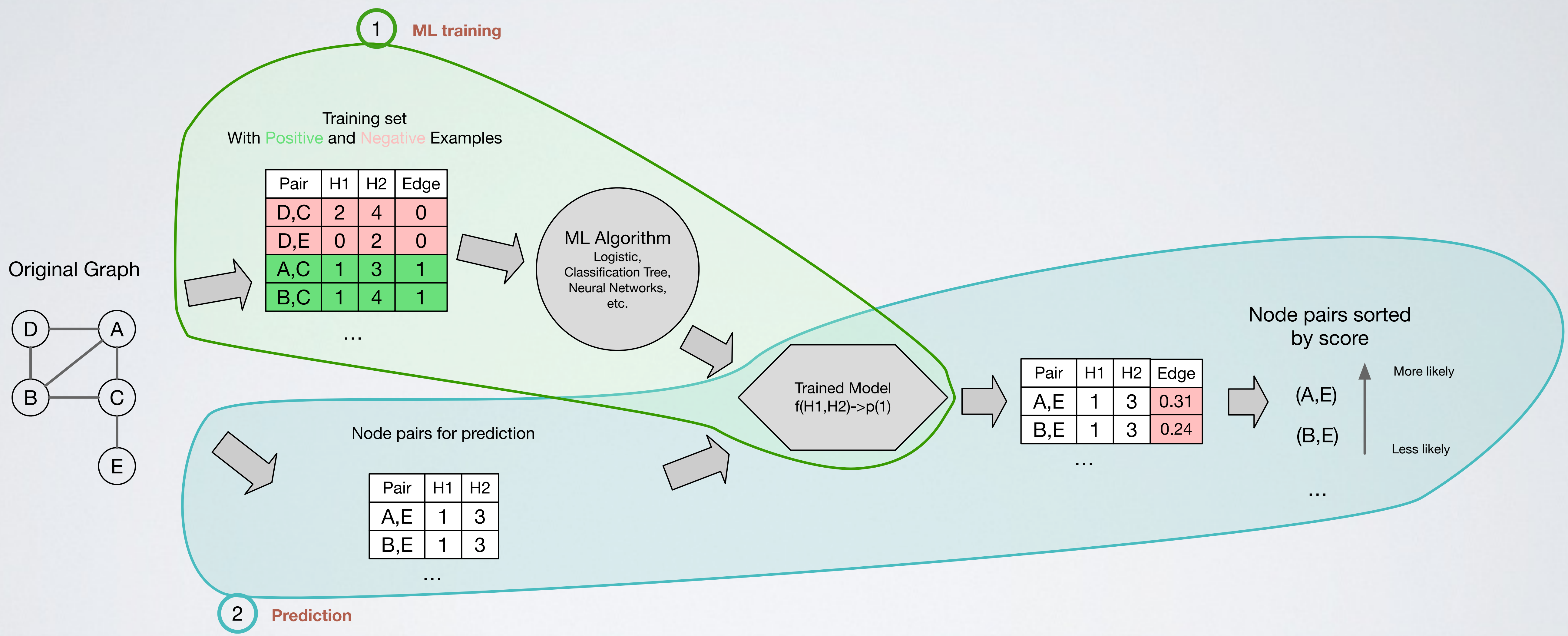
- Two steps:
  - Training: show features + value to predict
  - Using/Validating: try to predict value from features

SUPERVISED MACHINE LEARNING

- Our features: similarity indices (CN, AA, PA, …)
  - **One** (limited interest) or, obviously, **several**
  - Nodes attributes can be added of available (age, salary, etc.)

- Our label/value to predict: *Link(1)* or *No link(0)* (2 **classes**)

SUPERVISED MACHINE LEARNING



NODE CLASSIFICATION

Bhagat, S., Cormode, G., & Muthukrishnan, S. (2011). Node classification in social networks. In *Social network data analytics* (pp. 115-148). Springer, Boston, MA.

NODE CLASSIFICATION

- For the node classification task, we want to predict the class/category (or numerical value) of some nodes
  - Missing values in a dataset
  - Learn to predict, in a social network/platform(Netflix…) individuals':
    - Political position, opinion on a given topic, possible security threat, …
    - Interests, tastes, etc.
    - Age, genre, sexual orientation, language spoken, salary, etc.
    - Fake accounts, spammers, bots, malicious accounts, etc.
    - …
  - Wikipedia article category, types of road in an urban network, etc.

NODE CLASSIFICATION

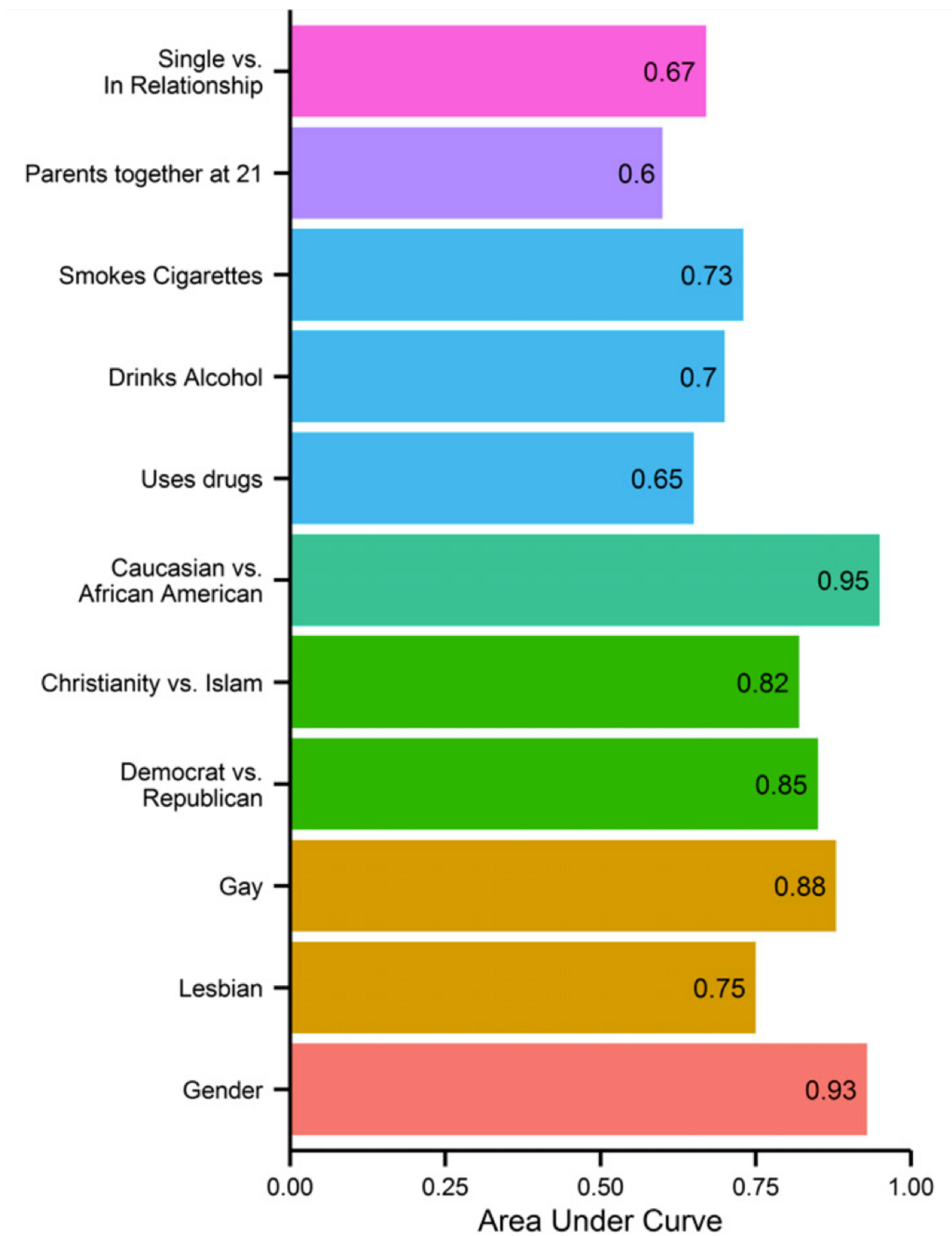


Fig. 2. Prediction accuracy of classification for dichotomous/dichotomized attributes expressed by the AUC.

Example of risks

Jernigan, C., & Mistree, B. F. (2009). Gaydar: Facebook friendships expose sexual orientation. *First Monday*, *14*(10).

Kosinski, M., Stillwell, D., & Graepel, T. (2013). Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, *110*(15), 5802-5805.

NODE FEATURES

- Non-network approach: Use a classification algorithm based on features of the node itself (age, salary, etc.)
- The network structure can be integrated using node centralities: Degree, clustering coefficient, betweenness, etc.
- But we can do much better:
  - "Tell me who your friends are, and I will tell you who you are"

NEIGHBORHOOD BASED CLASSIFICATION

- Classification based on the distribution of features in the neighborhood

- For each node, compute the distribution of labels in its neighborhood (vectors of length *m*, with *m* the set of all possible labels)
  - Pick the most frequent
    - e.g., political opinions
  - Train a classifier on this distribution
    - e.g., distribution of age, language in the neighborhoods to recognize bots (unexpectedly random)

MORE RECENT METHODS

- In the last 10 years, Deep Neural Networks have been introduced to perform ML tasks on networks
  - Considered state of the art for <u>supervised</u> tasks

- GCN: Graph Convolutional Neural Networks
  - Link prediction, Node classification, graph classification, etc.

- Variational Graph Autoencoder
  - Link prediction, graph embedding…

- GAT: Graph Attention Networks
  - Attention mechanism as in Transformers (ChatGP**T**) approach for graphs

- DCRNN (Diffusion, Convutionnal, Recurrent NN)
  - Dynamic data, e.g., traffic prediction…