Community detection in networks

Santo Fortunato

Aalto University

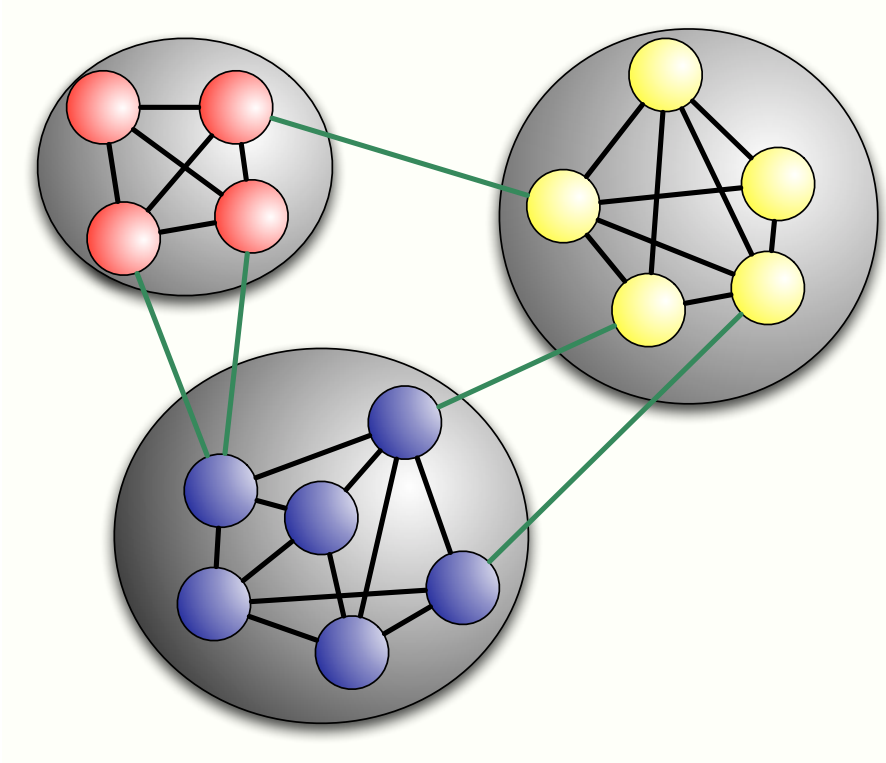
Outline

- Intro
- Elements of community detection
- Genesis of community structure
- Modularity optimization
- Infomap
- Overlapping communities
- Testing methods
- Outlook

Community structure

Communities: sets of tightly connected nodes

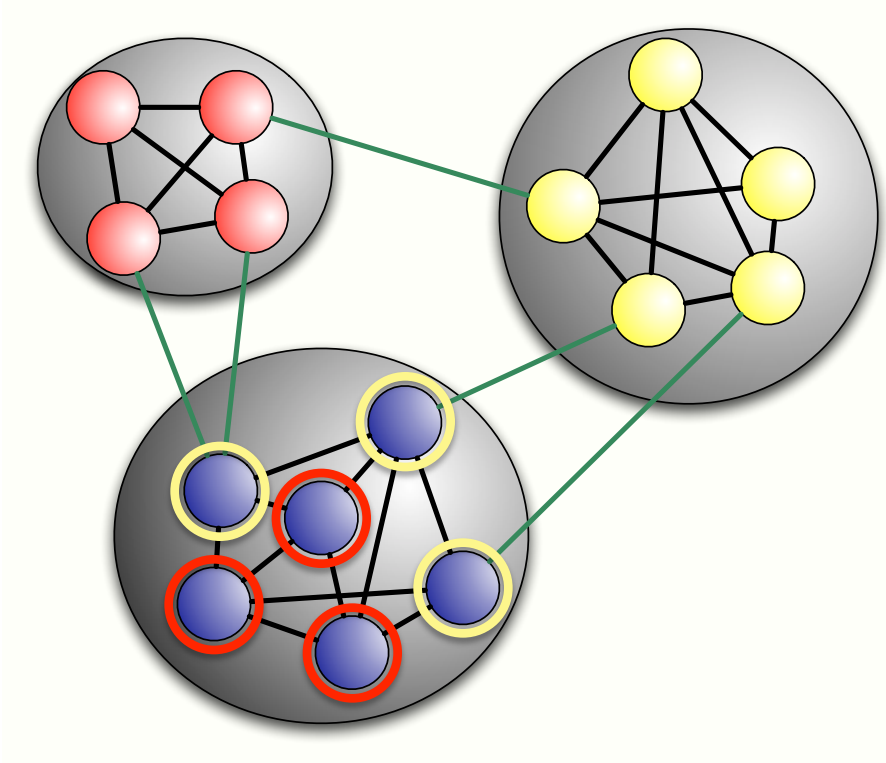
- People with common interests
- Scholars working on the same field
- Proteins with equal/similar functions
- Papers on the same/related topics
- …



Community detection

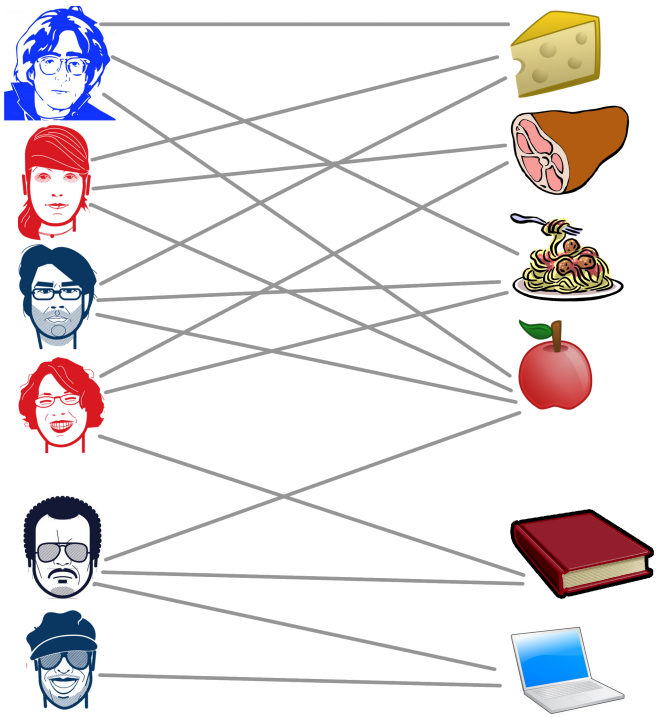
Theoretical reasons

- Organization
- Node features
- Node classification
- Missing links



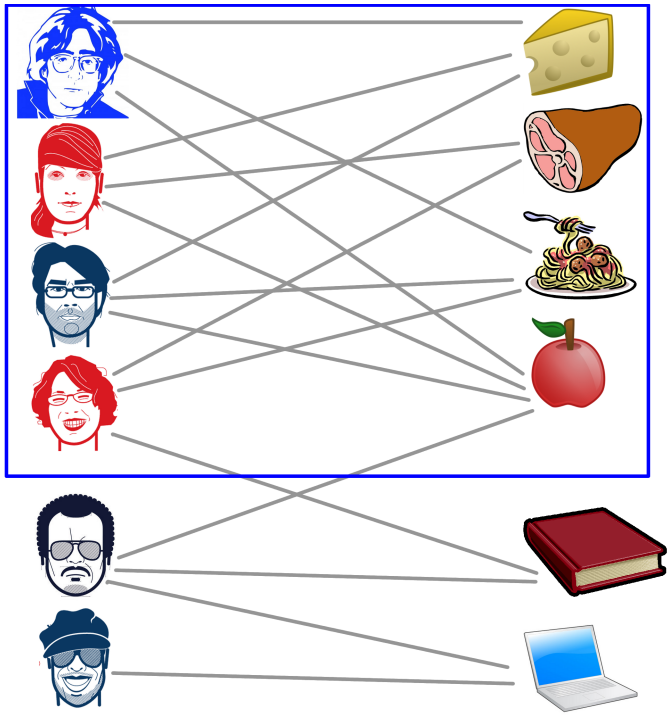
Community detection

Practical reasons: recommendation systems



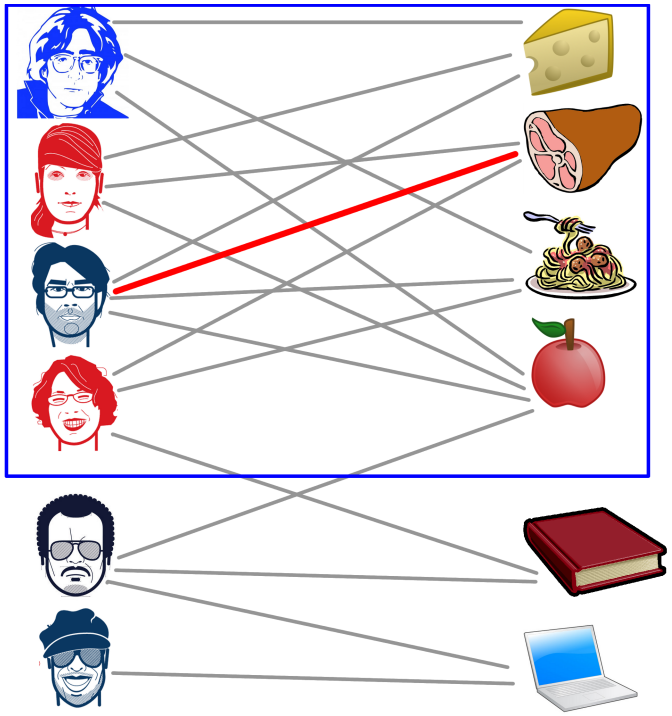
Community detection

Practical reasons: recommendation systems



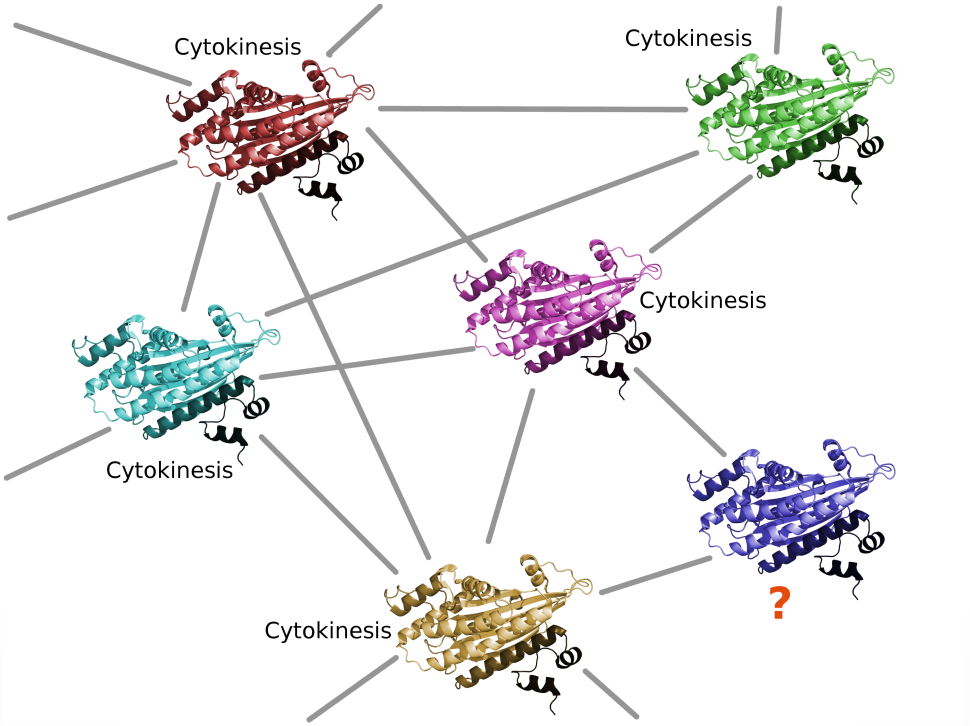
Community detection

Practical reasons: recommendation systems



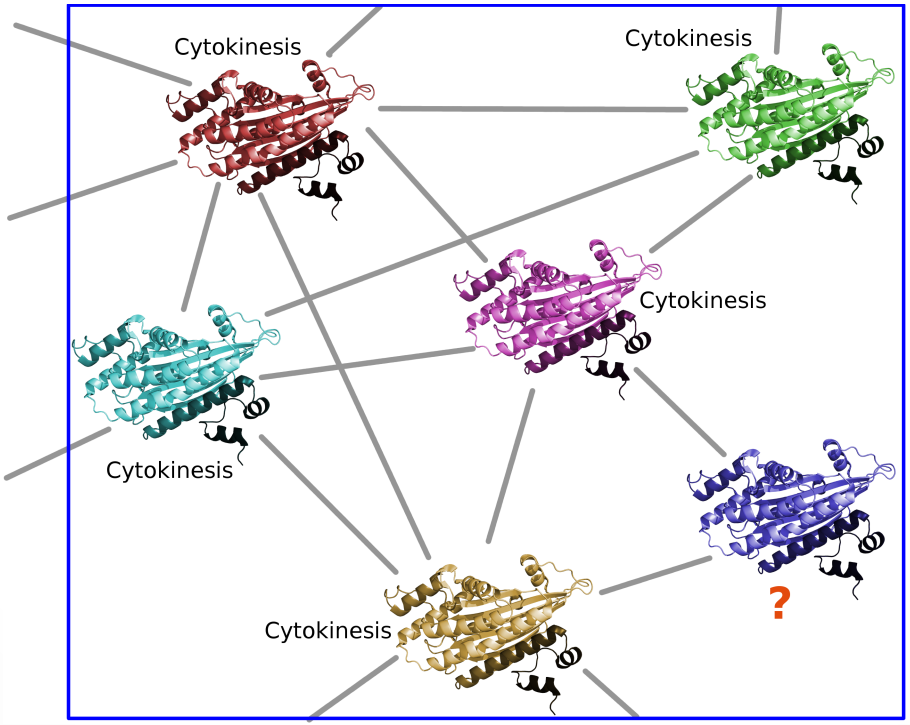
Community detection

Practical reasons: unknown protein functions



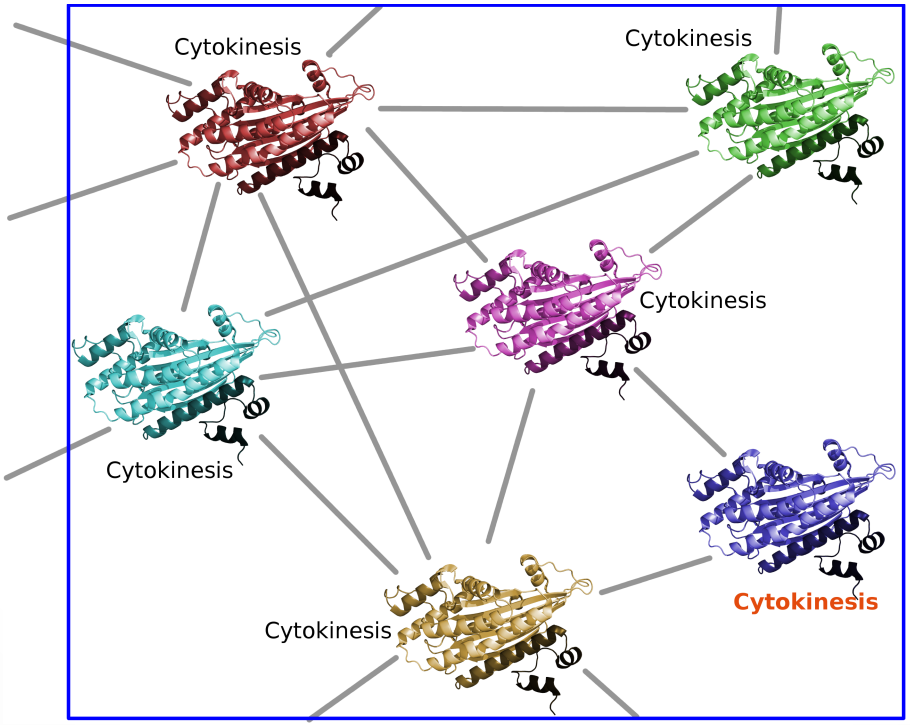
Community detection

Practical reasons: unknown protein functions

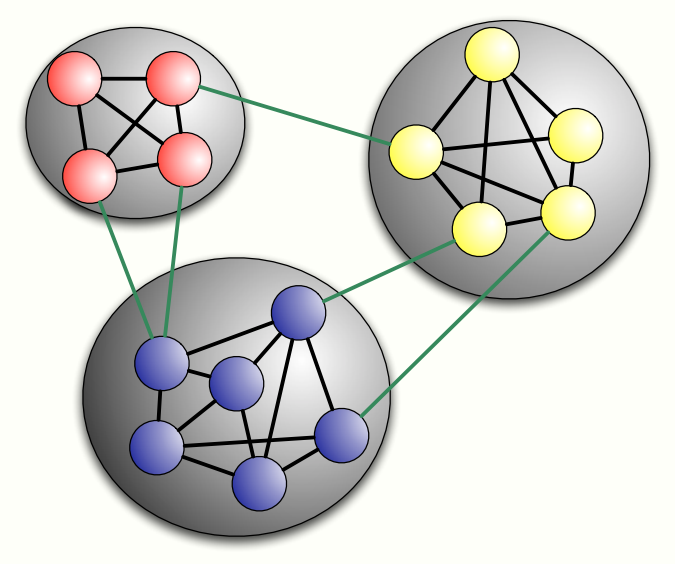


Community detection

Practical reasons: unknown protein functions



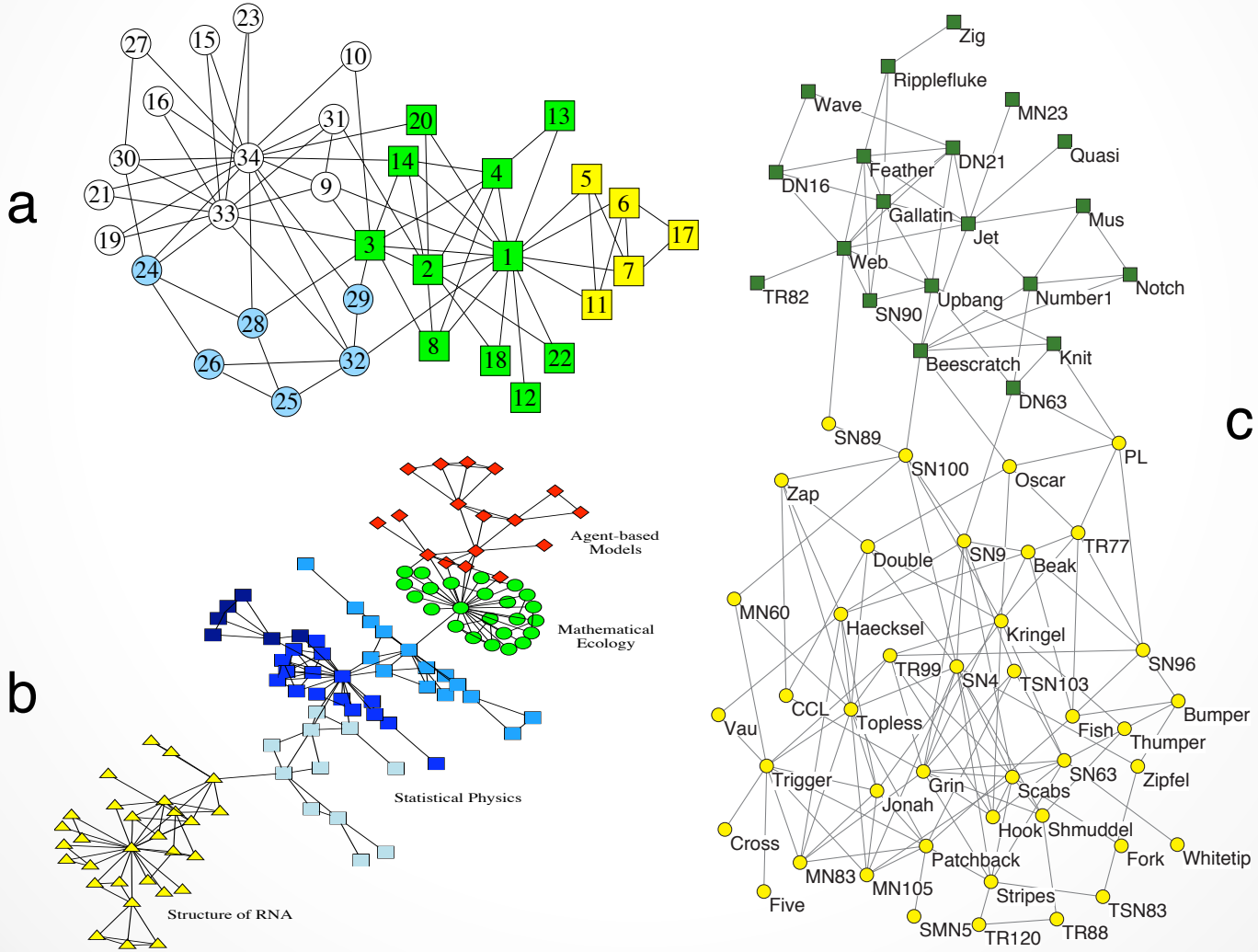
Difficult problem!



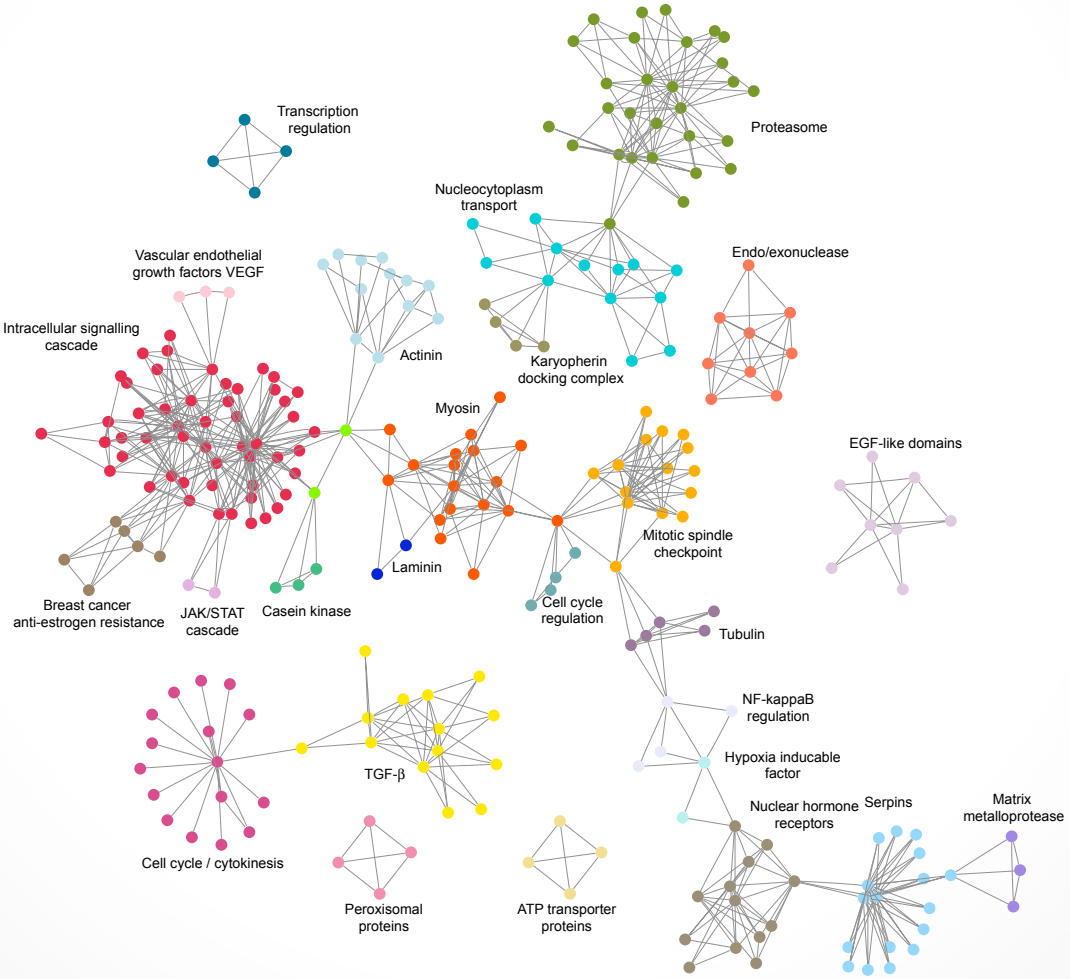
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1		1							1	1					
2	1				1				1	1					
3				1				1			1				1
4			1				1	1		1					1
5		1							1		1	1	1		
6								1			1		1	1	
7				1				1							1
8			1	1		1	1								1
9	1	1			1					1					
10	1	1		1					1						
11			1		1	1						1		1	
12					1						1		1	1	
13					1	1						1		1	
14						1					1	1	1		
15			1	1			1	1							

	10	1	2	9	5	13	12	14	6	11	4	3	8	7	15
10		1	1	1							1				
1	1		1	1											
2	1	1		1	1										
9	1	1	1		1										
5			1	1		1	1			1					
13					1		1	1	1						
12					1	1		1		1					
14						1	1		1	1					
6						1		1		1			1		
11					1		1	1	1			1			
4	1										1	1	1	1	1
3										1	1		1		1
8									1		1	1		1	1
7											1		1		1
15											1	1	1	1	

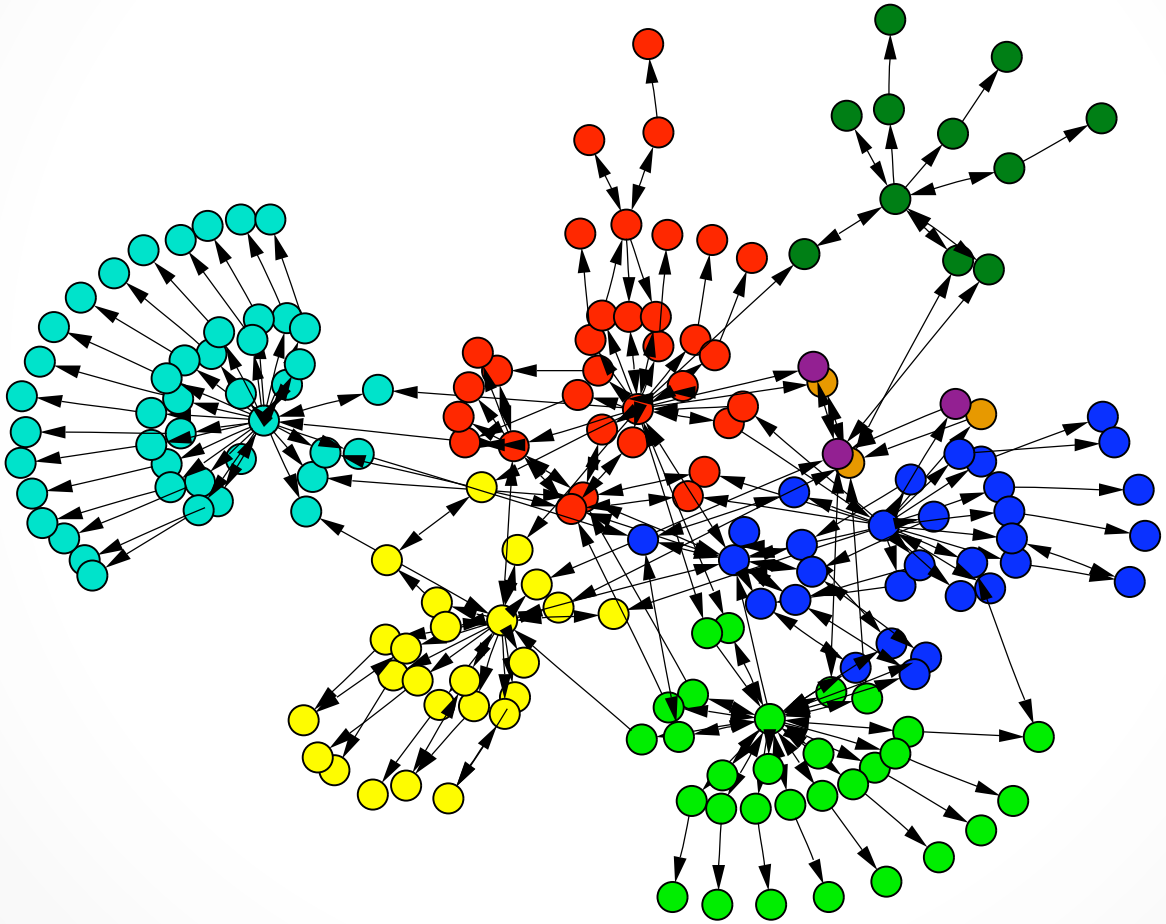
Communities in social networks



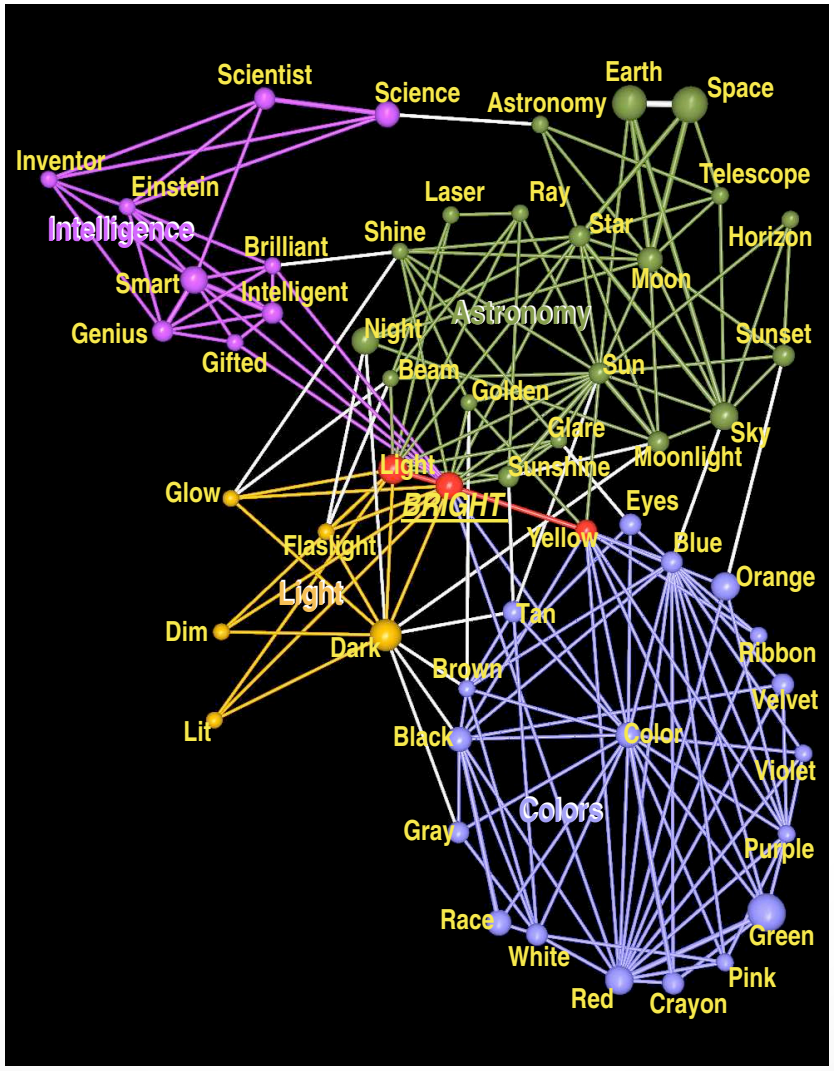
Communities in biological networks



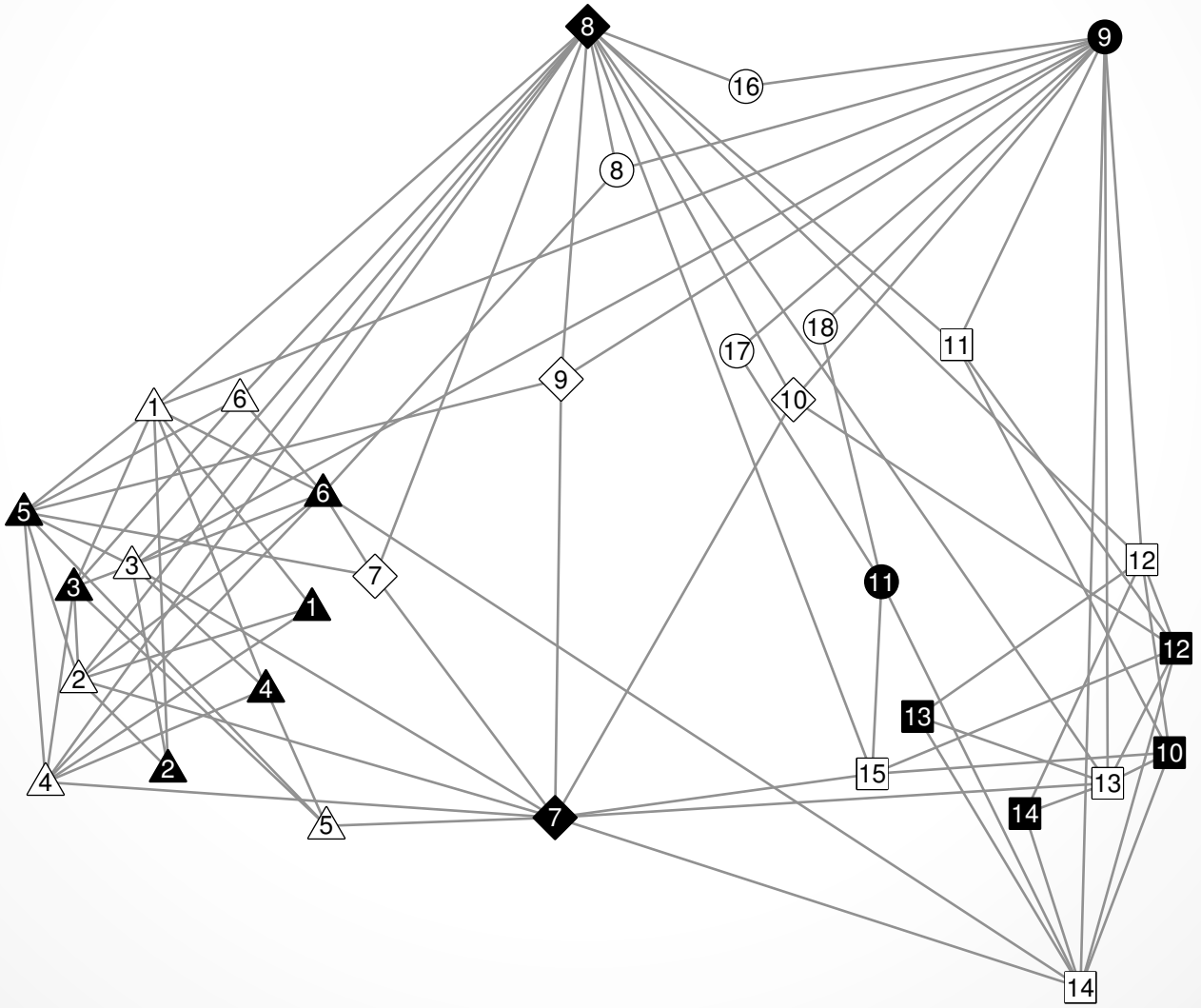
Communities in information networks



Overlapping communities



Communities in bipartite networks



Elements of community detection

Warnings:

1) Null hypothesis: communities are inferred just from structural information, relationship with actual groups is unclear!
2) The number m of edges of the graph is of the order of the number of vertices n, $m \sim n$

Topics:

1) Computational complexity
2) Communities
3) Partitions

Computational complexity

Definition: the computational complexity of an algorithm is the amount of resources required by the algorithm to perform a task

Resources: number of computation steps and memory units

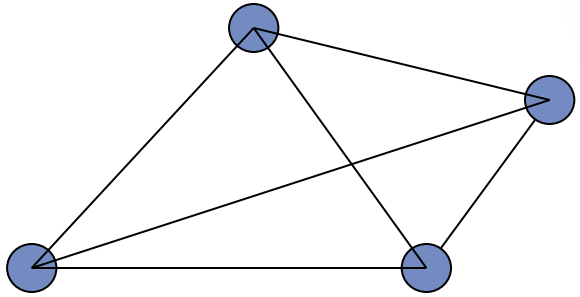
Notation: $O(n^{\alpha}m^{\beta})$, polynomial complexity (class P)

Exact complexity often unknown: worst-case complexity!

Communities: local definitions

Principle: look at the subgraph, forget the rest of the graph

Clique or complete graph



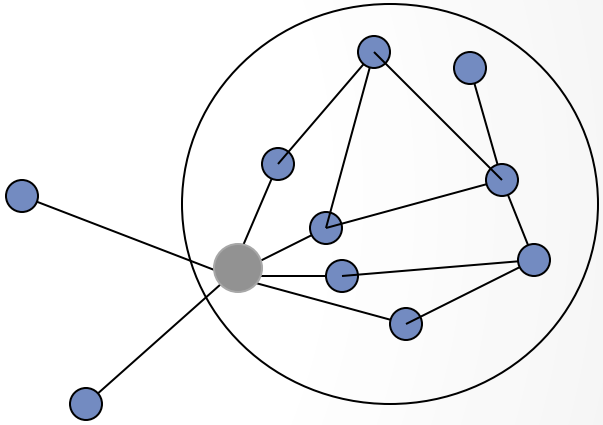
Problems

1) Condition is too strict!
2) All vertices are symmetric, whereas in real communities they usually have different roles
3) Cliques are hard to find: NP-complete problem. Bron-Kerbosch method has a complexity growing exponentially with the graph size (Bron & Kerbosch, 1973)

Communities: local definitions

Principle: comparison between internal and external cohesion of a subgraph

LS-set or strong community: subgraph such that the internal degree of each vertex is greater than its external degree (Luccio & Sami, 1969)



Problem: condition too strong, unrealistic in practical cases!

Weak community: subgraph such that the internal degree of the subgraph is greater than its external degree (Radicchi et al., 2004)

Communities: local definitions

Variant of concepts of strong and weak community (Hu et al., 2008)

Strong community: subgraph such that the internal degree of each vertex is greater than its internal degree in any of the other communities of the partition

Weak community: subgraph such that the internal degree of the subgraph is greater than its external degree in each of the other communities

Link with planted ℓ-partition model (Condon & Karp, 1999)

Warning

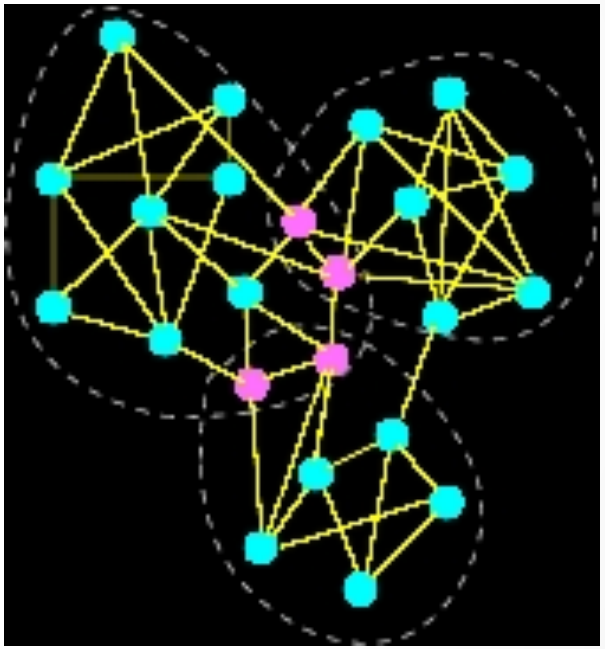
Communities are usually implicitly defined by the specific algorithm adopted, without an explicit definition!

The practical definition may depend on the specific system/application

Partitions: basics

A partition is a division of a graph into clusters, such that each vertex is assigned to one and only one cluster!

If vertices can belong to two or more clusters simultaneously, one speaks of covers



G. Palla, I. Derényi, I. Farkas, T. Vicsek, Nature 435, 814, 2005

Partitions: basics

The number of possible partitions in k clusters of a graph with n vertices is the Stirling number of the second kind S(n, k)

Total number of possible partitions: Bell number

$$B_n = \sum_{k=1}^{n} S(n,k)$$

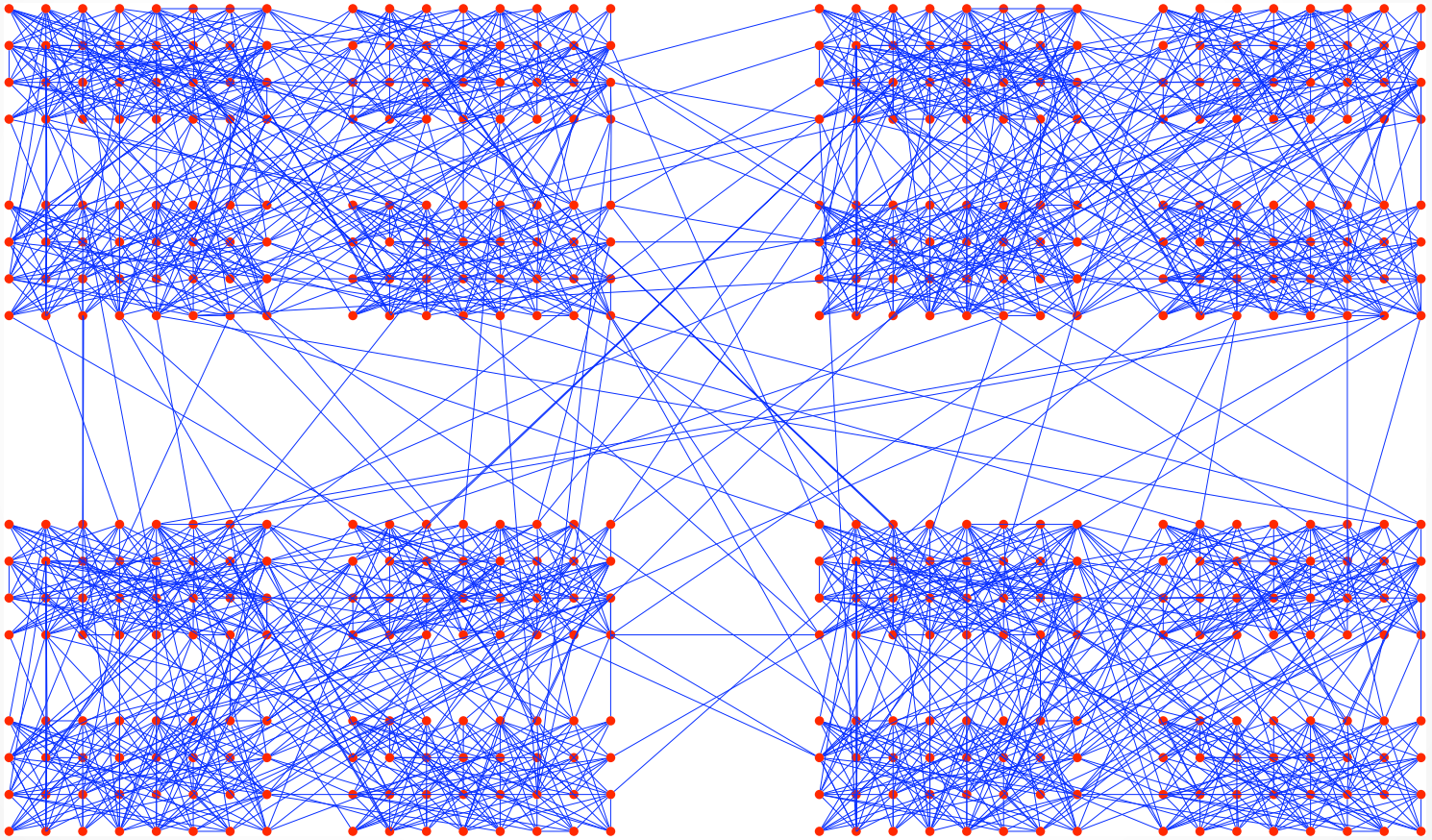
Large-n limit

$$B_n \sim \frac{1}{\sqrt{n}}[\lambda(n)]^{n+1/2} e^{\lambda(n)-n-1}$$

$$\lambda(n) = e^{W(n)} = n/W(n),\ W(n) \text{ Lambert function}$$

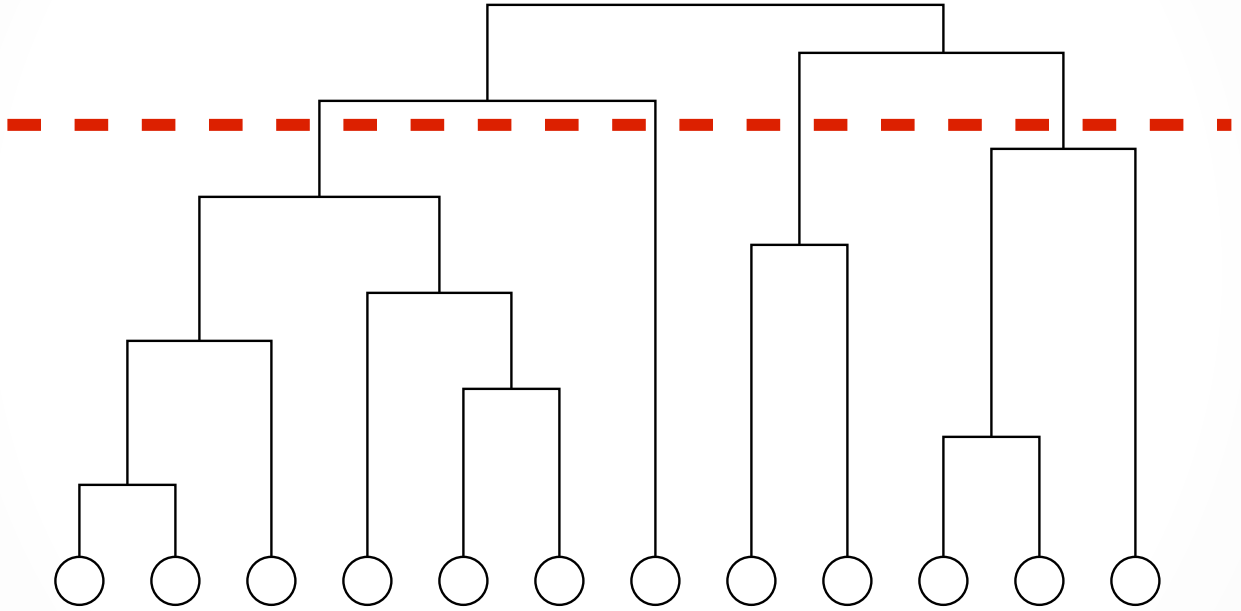
Hierarchy

Clusters may be included in other clusters, etc. (hierarchical order!)



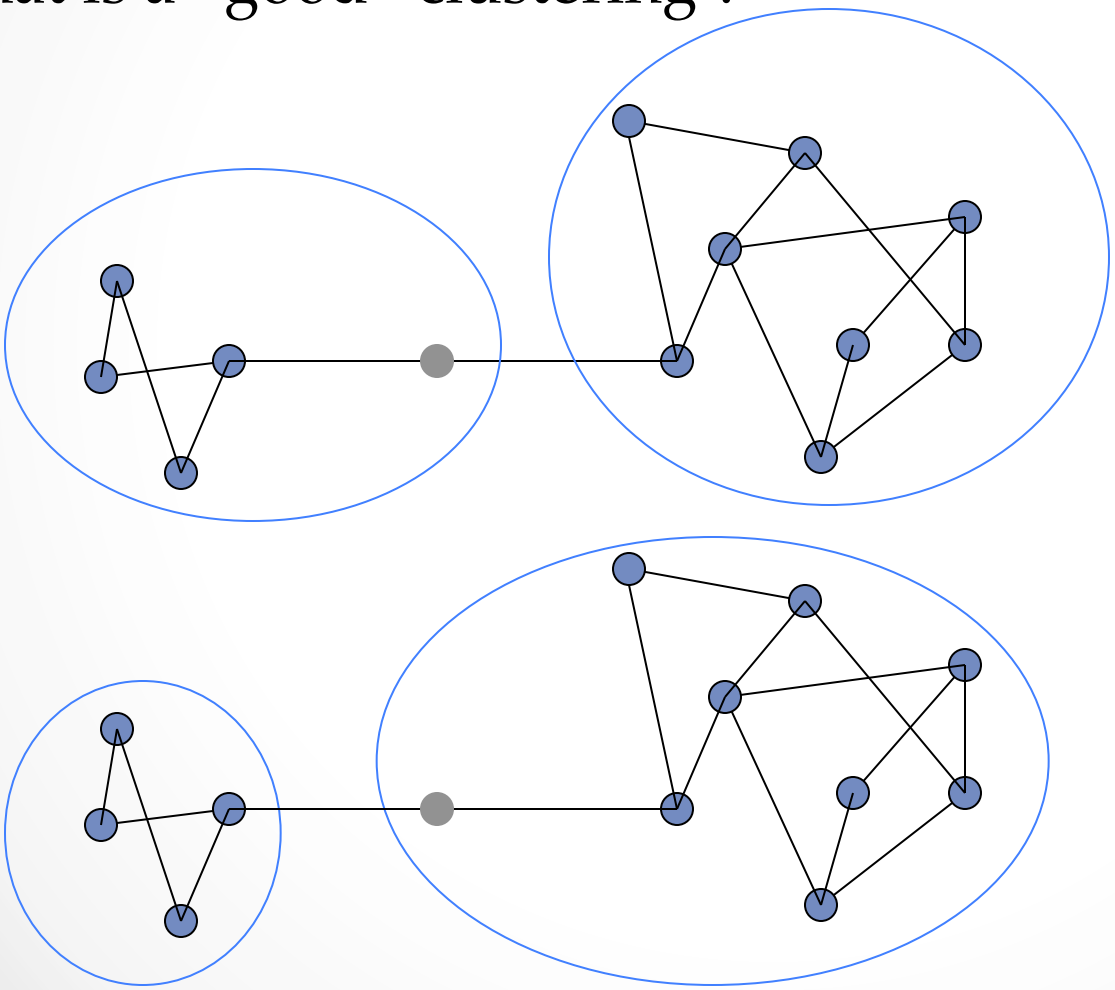
Clauset, Moore & Newman, LNCS 4503, 1, 2007

Dendrograms



Partitions: quality functions

What is a “good” clustering ?



?

Partitions: quality functions

A quality function assigns a score to each partition/cover of a graph

High-score partitions are "good", low-score partitions are "bad"

Additivity $$Q(\mathcal{P}) = \sum_{\mathcal{C} \in \mathcal{P}} q(\mathcal{C})$$

Examples of quality functions

1) Performance:

$$P(\mathcal{P}) = \frac{|\{(i,j) \in E, C_i = C_j\}| + |\{(i,j) \notin E, C_i \neq C_j\}|}{n(n-1)/2}$$

2) Coverage:

$$C(\mathcal{P}) = \frac{|\{(i,j) \in E, C_i = C_j\}|}{m}$$

Partitions: modularity

Newman & Girvan, 2004

Principle: random graphs have no community structure!

Method: comparing the edge density in each cluster with the edge density of the cluster in a randomized version of the graph

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - P_{ij})\, \delta(C_i, C_j)$$

Null model in principle arbitrary

Ex. Bernoulli random graph

$$P_{ij} = p = 2m/[n(n-1)]$$

Partitions: modularity

Problem with Bernoulli random graph: degree distribution is binomial/Poissonian

Modularity's null model: random graph with identical expected degree sequence of original graph

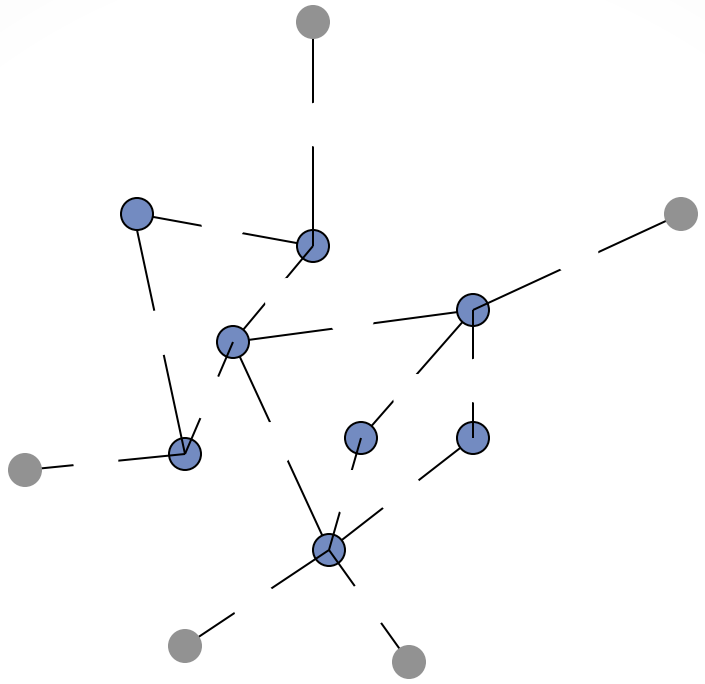
$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j)$$

$$Q = \frac{1}{m} \sum_{c=1}^{n_c} \left[l_c - \frac{d_c^2}{4m} \right]$$

$l_c =$ Number of edges inside cluster c

$d_c =$ Total degree of cluster c

$n_c =$ Number of clusters



$\frac{d_c}{2m}$ = probability that a stub, randomly selected, ends in module c

$\frac{d_c}{2m}$ $\frac{d_c}{2m}$

$$\frac{d_c}{2m} \cdot \frac{d_c}{2m} = \frac{d_c^2}{4m^2}$$

probability that the link is internal to module c

$$\frac{d_c^2}{4m^2} \cdot m = \frac{d_c^2}{4m}$$

expected number of links in module c

Partitions: modularity

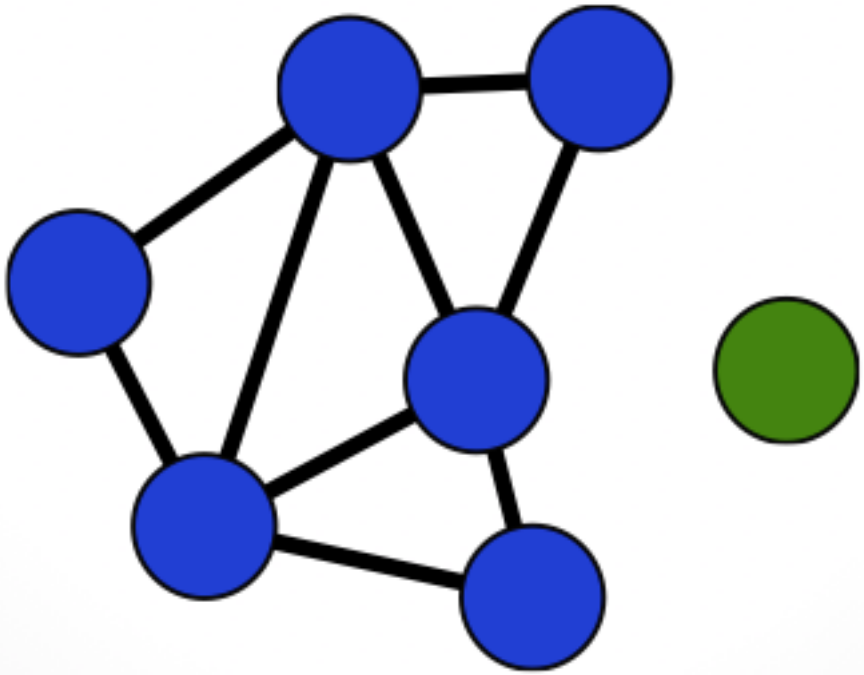
$$Q = \frac{1}{m} \sum_{c=1}^{n_c} \left[l_c - \frac{d_c^2}{4m} \right]$$

Some features:

1) $Q < 1$
2) $Q = 0$ for the partition in which the whole graph is one cluster
3) Modularity can be negative (multipartite structure)
4) Modularity in general depends on graph size: partitions of different graphs cannot be compared to each other based on their modularity values
5) Large values of modularity not necessarily indicate good partitions: random graphs may have high-modularity partitions

How is community structure generated?

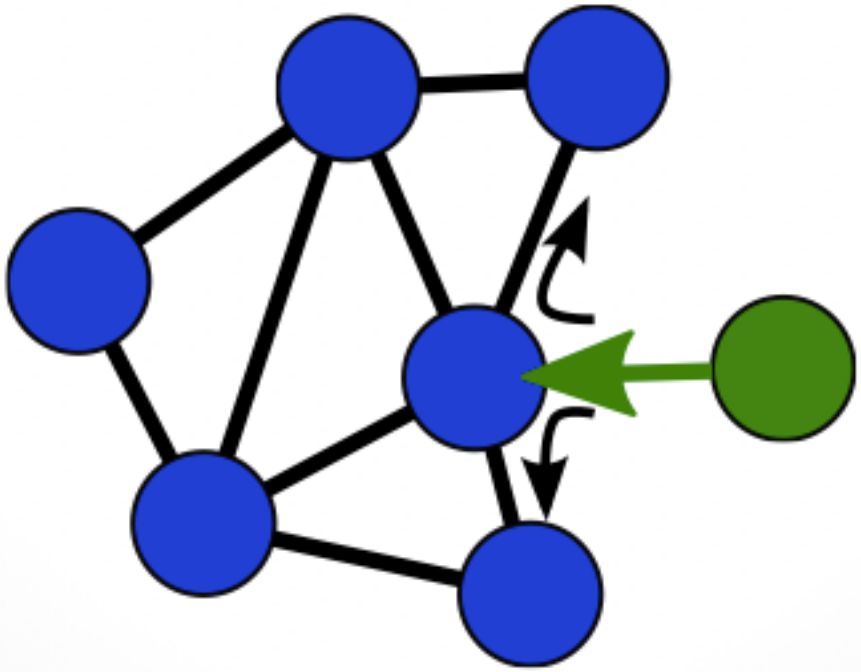
Triadic closure



G. Bianconi, R. K. Darst, J. Iacovacci, S. F., Phys. Rev. E 90, 042806 (2014)

How is community structure generated?

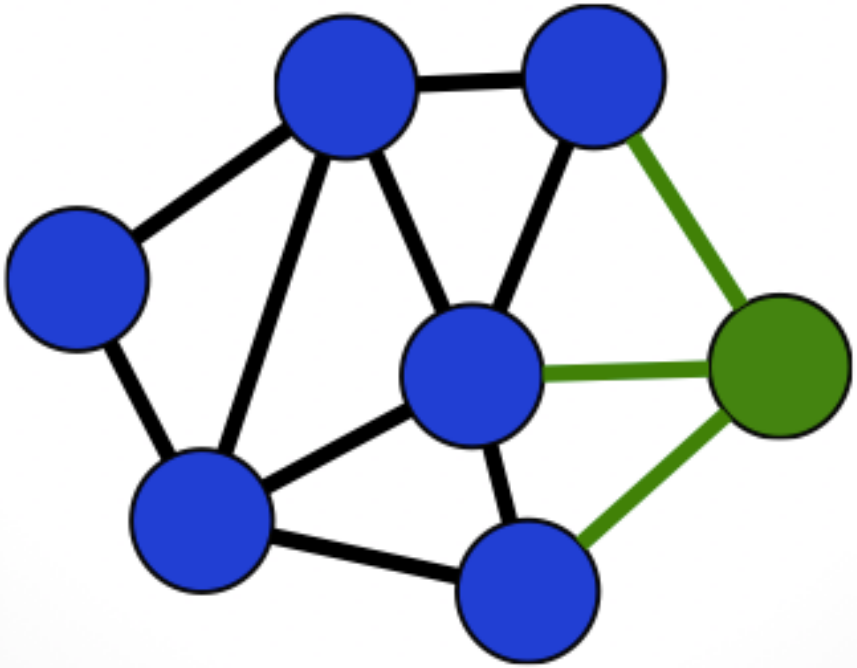
Triadic closure



G. Bianconi, R. K. Darst, J. Iacovacci, S. F., Phys. Rev. E 90, 042806 (2014)

How is community structure generated?

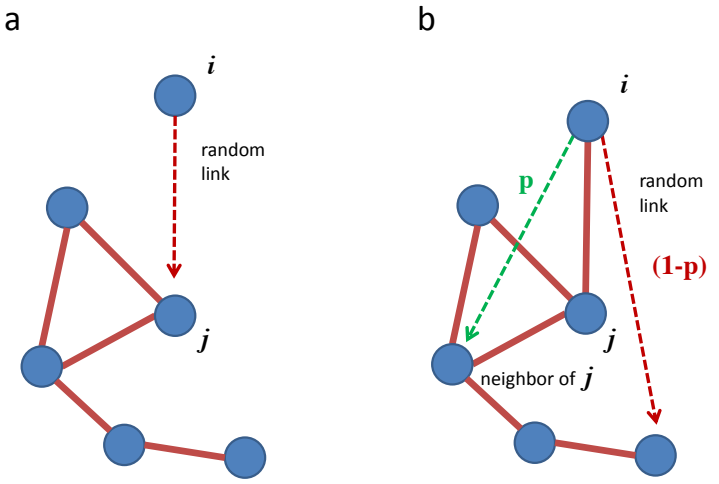
Triadic closure



G. Bianconi, R. K. Darst, J. Iacovacci, S. F., Phys. Rev. E 90, 042806 (2014)

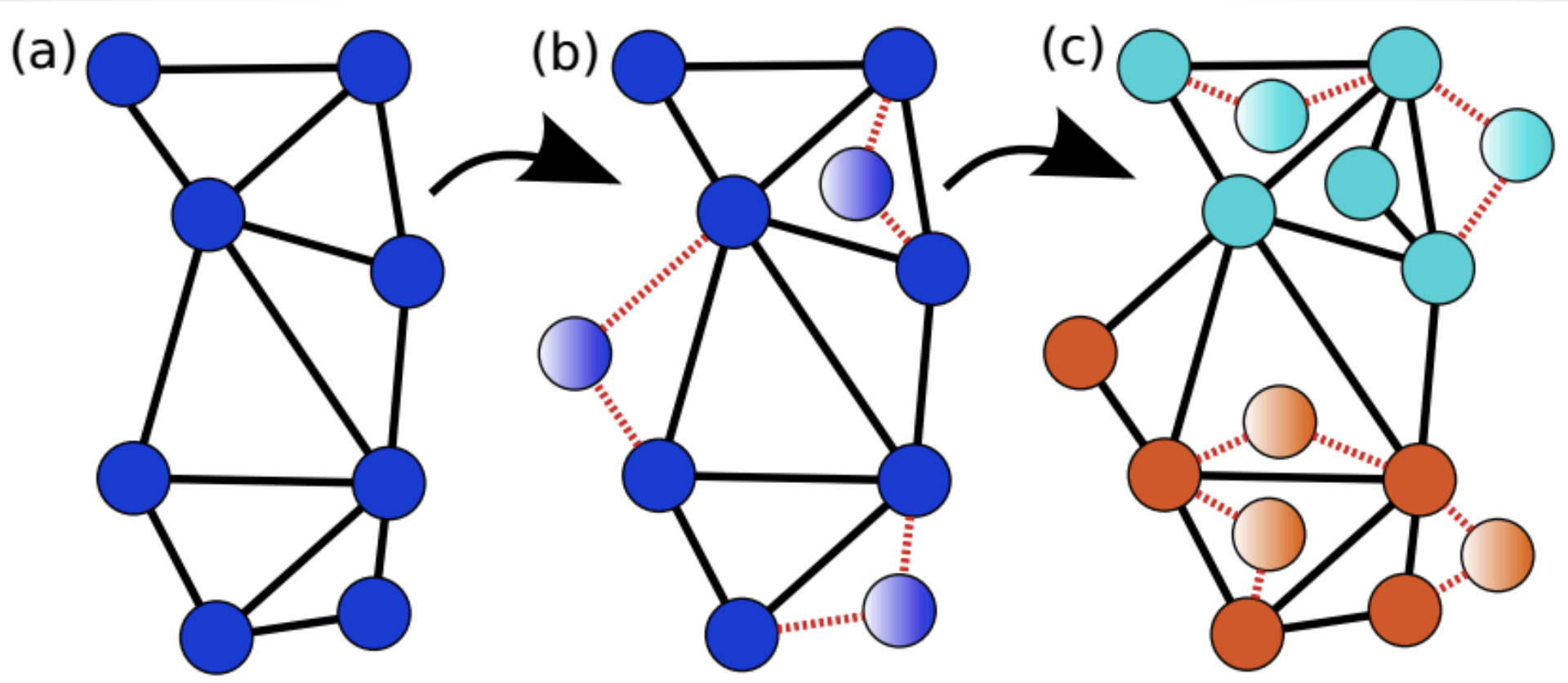
How is community structure generated?

Question: can models based on triadic closure explain the emergence of communities?



G. Bianconi, R. K. Darst, J. Iacovacci, S. F., Phys. Rev. E 90, 042806 (2014)

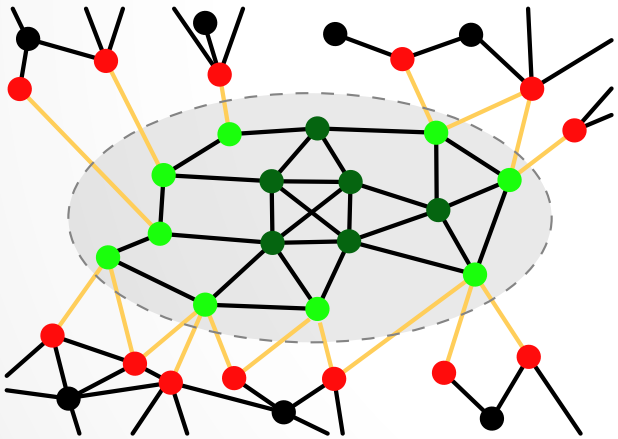
How is community structure generated?

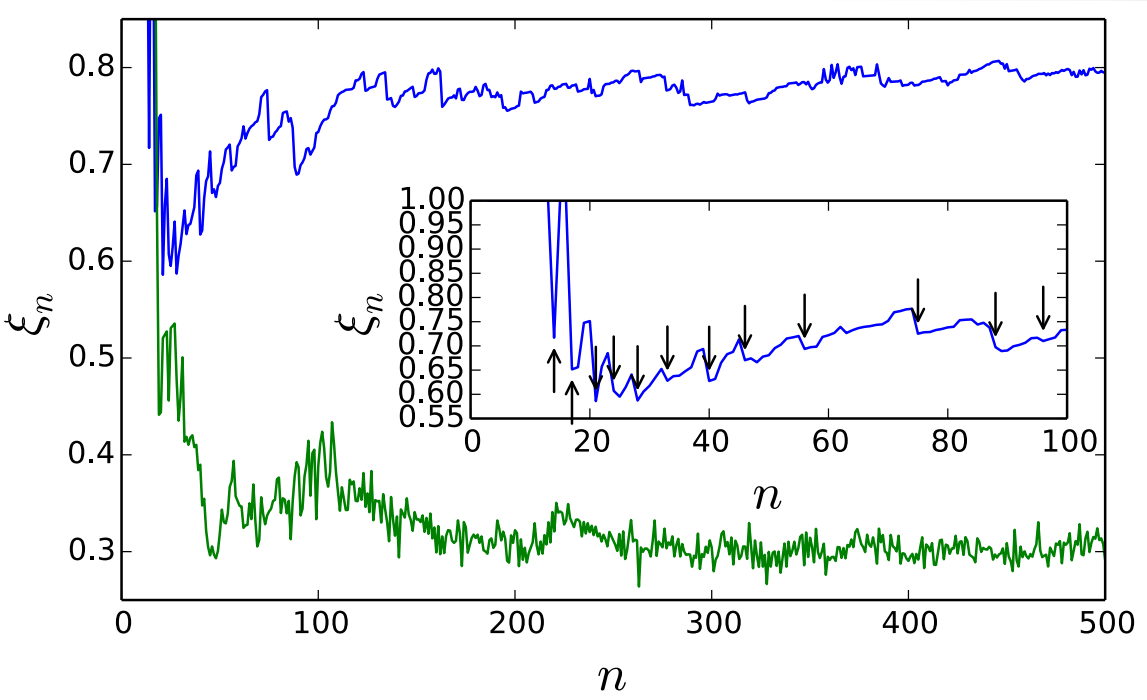


G. Bianconi, R. K. Darst, J. Iacovacci, S. F., Phys. Rev. E 90, 042806 (2014)

How is community structure generated?

$$\text{Cluster embeddedness} \sim \frac{\#\text{ internal neighbours of cluster nodes}}{\#\text{ total neighbours}}$$





G. Bianconi, R. K. Darst, J. Iacovacci, S. F., Phys. Rev. E 90, 042806 (2014)

Modularity optimization

$$Q = \frac{1}{m} \sum_{c=1}^{n_c} \left[l_c - \frac{d_c^2}{4m} \right]$$

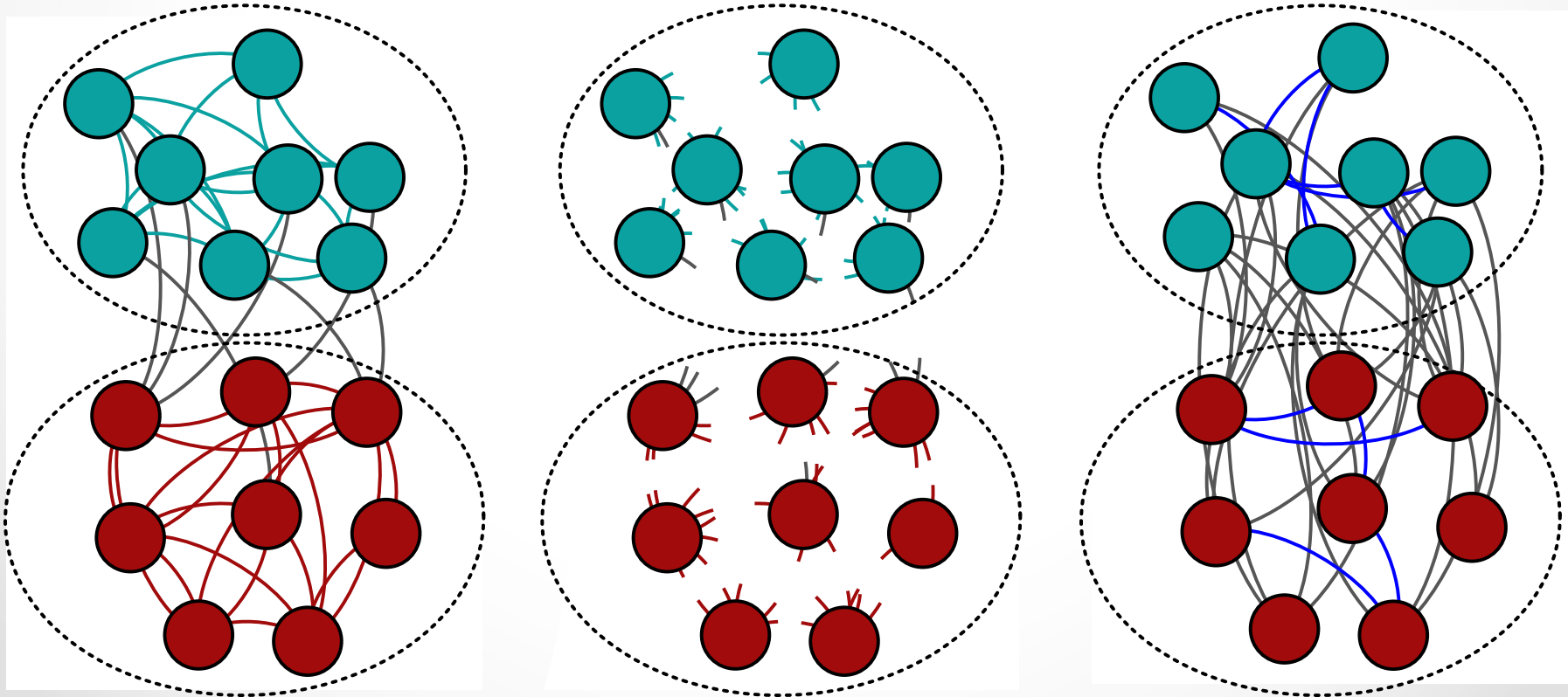
Goal: find the maximum of Q over all possible network partitions

Problem: NP-complete (Brandes et al., 2007)!

1) Greedy algorithms
2) Simulated annealing
3) Extremal optimization
4) Spectral optimization
5) Other strategies

Modularity optimization

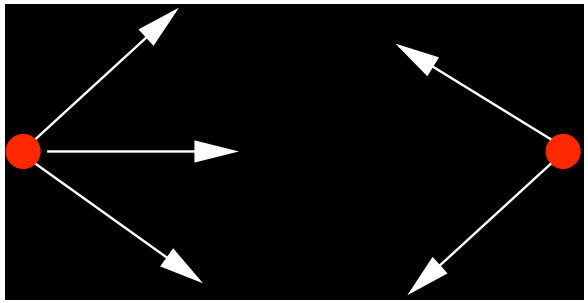
$$Q = \frac{1}{m} \sum_{c=1}^{n_c} \left(l_c - \frac{d_c^2}{4m} \right)$$



Modifications of modularity

Directed modularity (Arenas et al., 2007)

$$Q_d = \frac{1}{m} \sum_{ij} \left(A_{ij} - \frac{k_i^{out} k_j^{in}}{m} \right) \delta(C_i, C_j)$$



Directed weighted modularity

$$Q_{gen} = \frac{1}{W} \sum_{ij} \left(W_{ij} - \frac{s_i^{out} s_j^{in}}{W} \right) \delta(C_i, C_j)$$

Greedy algorithms

Newman's method (Phys. Rev. E 69, 066133, 2004)

- Start: partition with one vertex in each community
- Merge groups of vertices so to obtain the highest increase of Q
- Continue until all vertices are in the same community
- Pick the partition with largest modularity

CPU time $O(mn)$ [$O(n^2)$ on a sparse graph]

Greedy algorithms

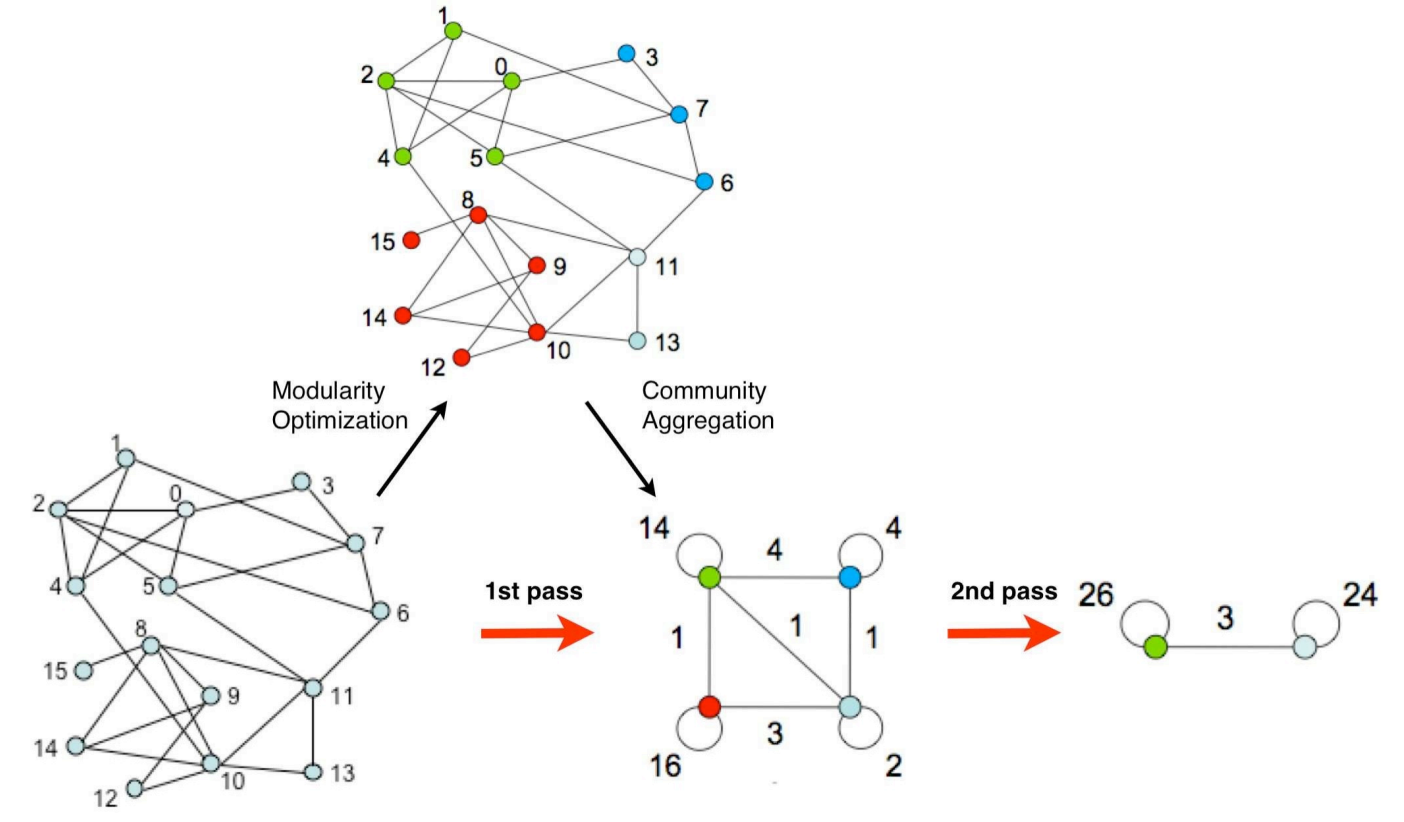
Louvain method (Blondel et al., JSTAT P10008, 2008)

Steps:

1) Loop over the vertices: each vertex is put in the community of their neighbors that yields the largest increase of modularity
2) Communities are replaced by supervertices, edges between supervertices are weighted by the number of simple edges between them
3) Repeat from 1 for the current weighted graph
4) Modularity is always computed with respect to the original graph, when it cannot increase any more the process stops

Complexity: $O(m)$

Louvain method



Problems

1) Order of sequence over vertices influence final results
2) Unbalanced partitions on large graphs

Limits of modularity

Question: does high modularity imply that a partition is good?

Answer: no! Random graphs may have large values of the modularity maximum, due to fluctuations!

Reason: modularity's null model term is expected (average) value, it does not consider fluctuations

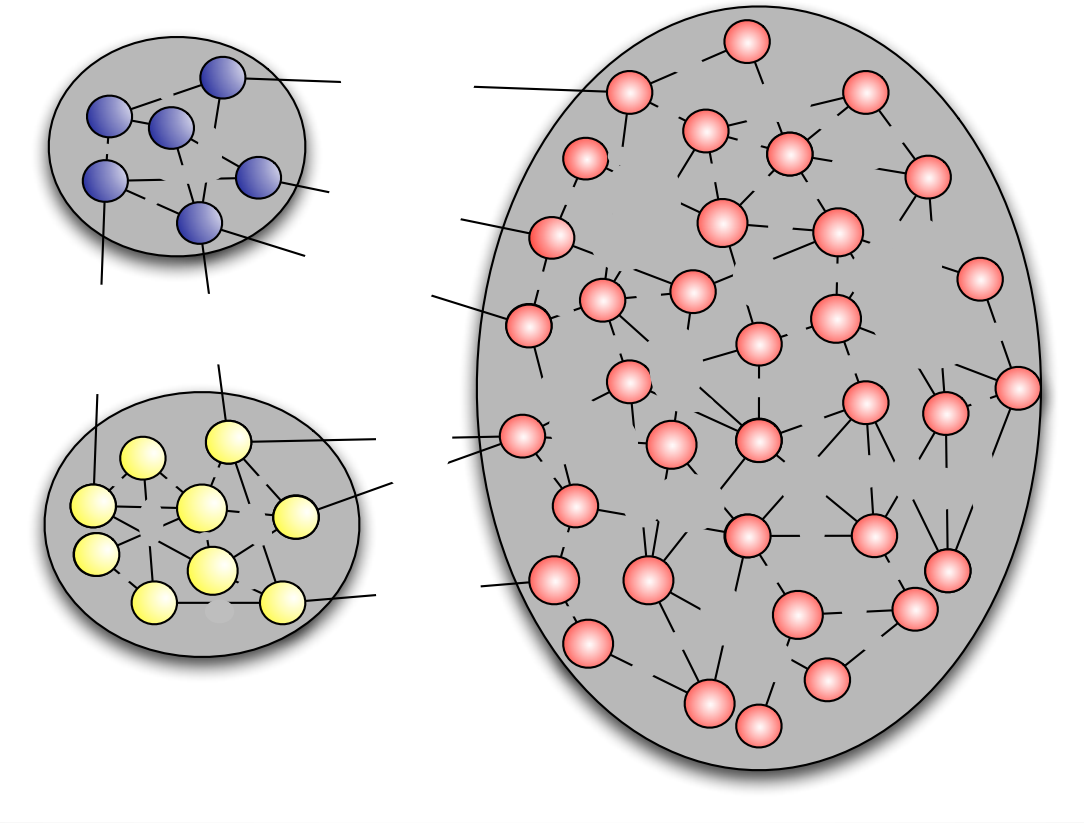
Possible recipe: computing both the average maximum modularity $\langle Q \rangle_{NM}$ and the standard deviation σ_Q^{NM} out of a large number of null model graphs

Z-score:

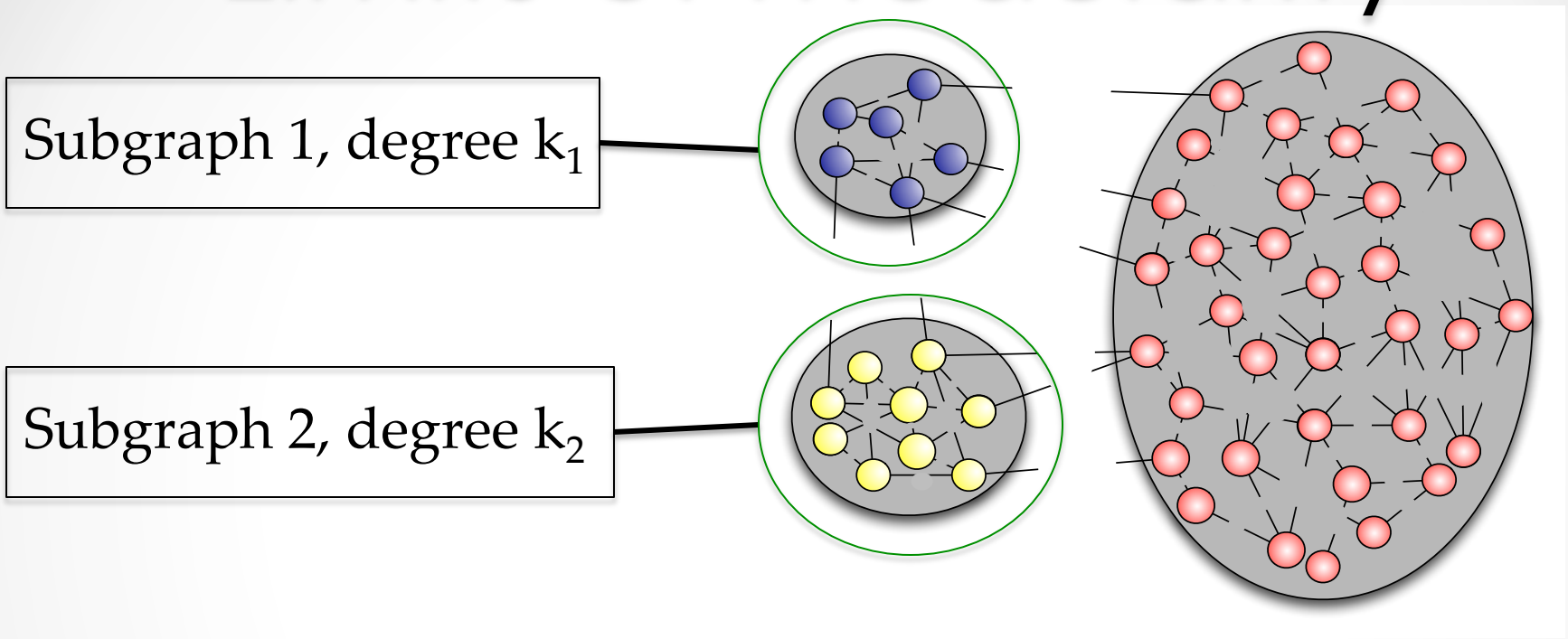
$$z = \frac{Q_{max} - \langle Q \rangle_{NM}}{\sigma_Q^{NM}}$$

Limits of modularity

Resolution limit (Fortunato & Barthélemy, 2007)



Limits of modularity



Expected number of edges between the two subgraphs in modularity's null model:

$$m\left(2\cdot\frac{k_1}{2m}\cdot\frac{k_2}{2m}\right)=\frac{k_1k_2}{2m}$$

$$\text{if } k_1=k_2=d_c \rightarrow \frac{d_c^2}{2m}$$

Limits of modularity

if $\frac{k_1 k_2}{2m} < 1 \rightarrow$ Q is higher when 1 & 2 are in the same cluster!

Ex. $k_1 \sim k_2 < \sqrt{2m}$

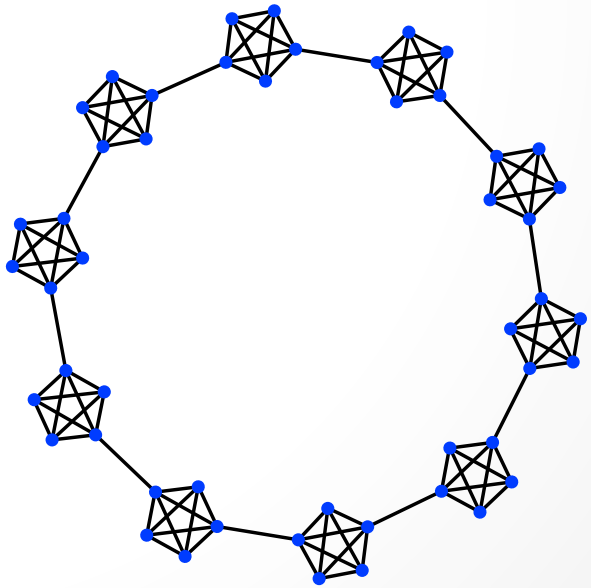
Resolution limit of modularity

Resolution limit

$$Q = \frac{1}{m} \sum_{c=1}^{n_c} \left[l_c - \frac{1}{4} \left(\frac{d_c}{\sqrt{m}} \right)^2 \right]$$

modularity's scale

Consequences

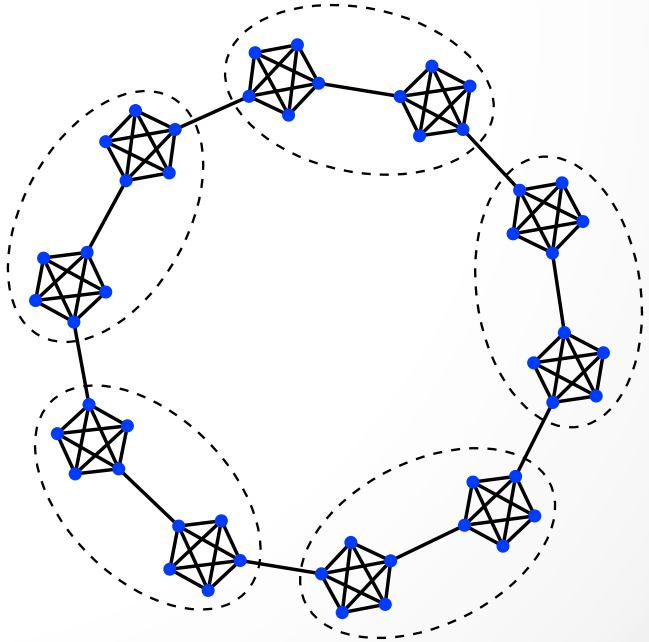


Resolution limit

$$Q = \frac{1}{m} \sum_{c=1}^{n_c} \left[l_c - \frac{1}{4} \left(\frac{d_c}{\sqrt{m}} \right)^2 \right]$$

modularity's scale

Consequences

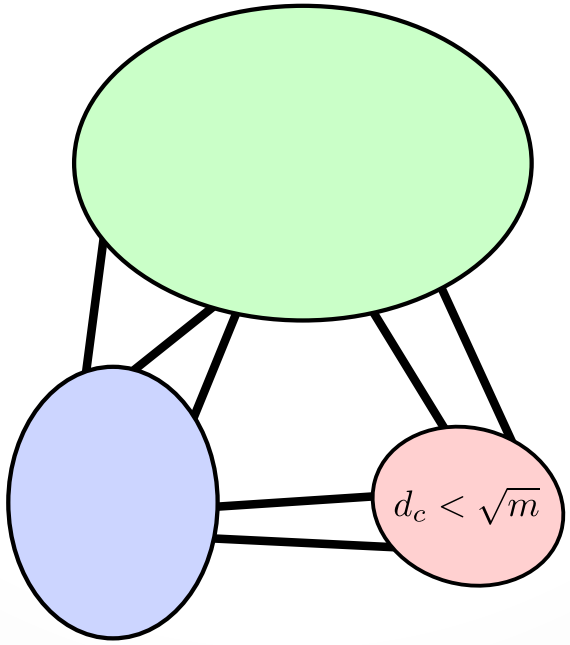


Resolution limit

$$Q = \frac{1}{m}\sum_{c=1}^{n_c}\left[l_c - \frac{1}{4}\left(\frac{d_c}{\sqrt{m}}\right)^2\right]$$

modularity's scale

Consequences

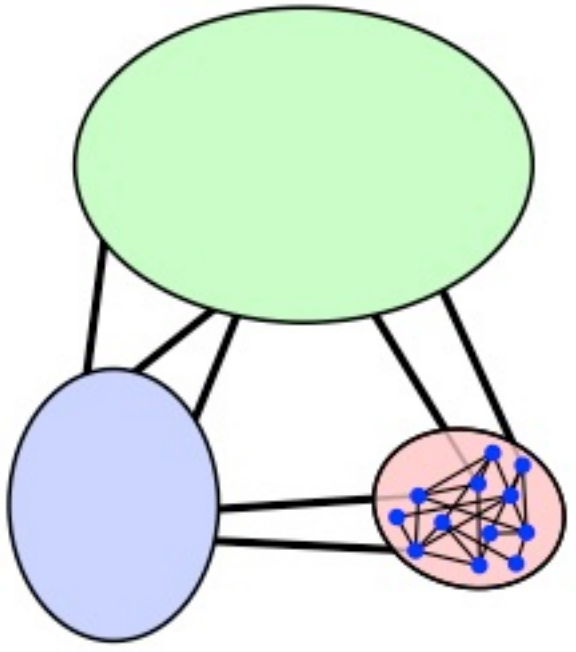


Resolution limit

$$Q = \frac{1}{m} \sum_{c=1}^{n_c} \left[l_c - \frac{1}{4} \left(\frac{d_c}{\sqrt{m}} \right)^2 \right]$$

modularity's scale

Consequences

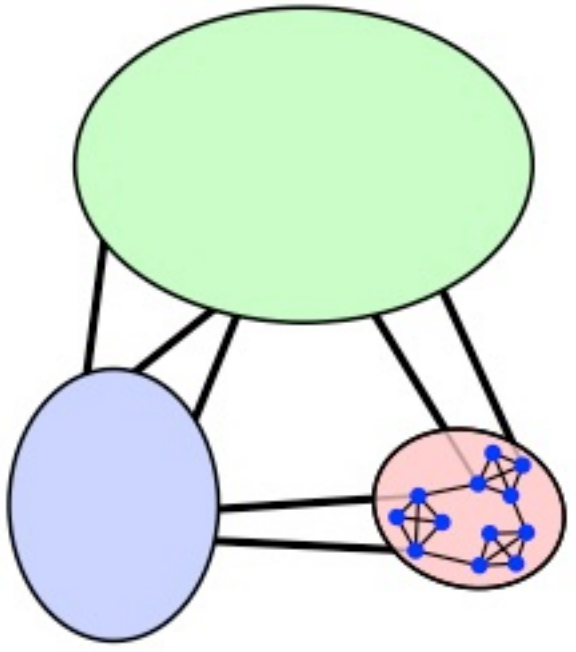


Resolution limit

$$Q = \frac{1}{m} \sum_{c=1}^{n_c} \left[l_c - \frac{1}{4} \left(\frac{d_c}{\sqrt{m}} \right)^2 \right]$$

modularity's scale

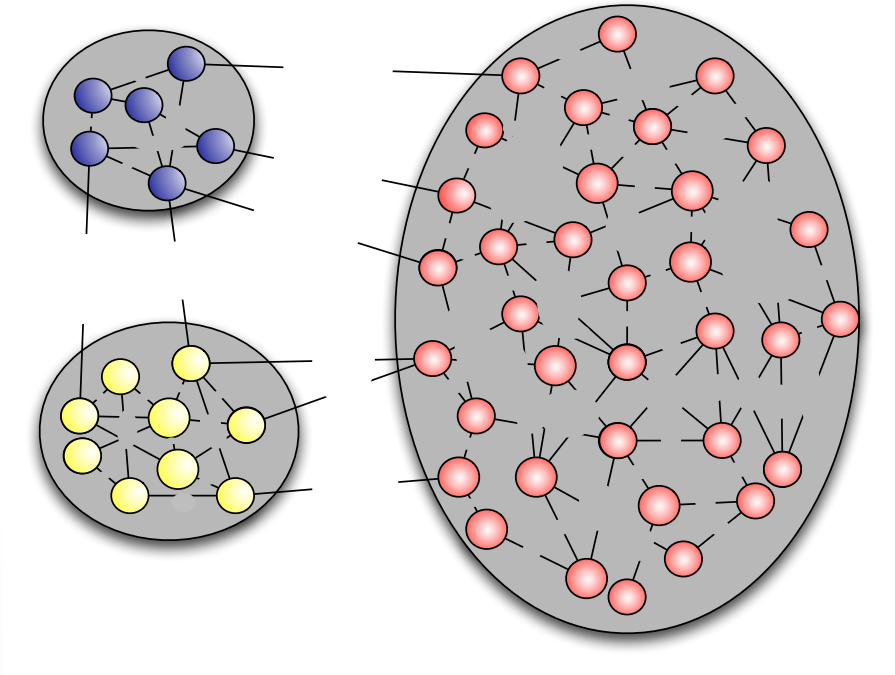
Consequences



Limits of modularity

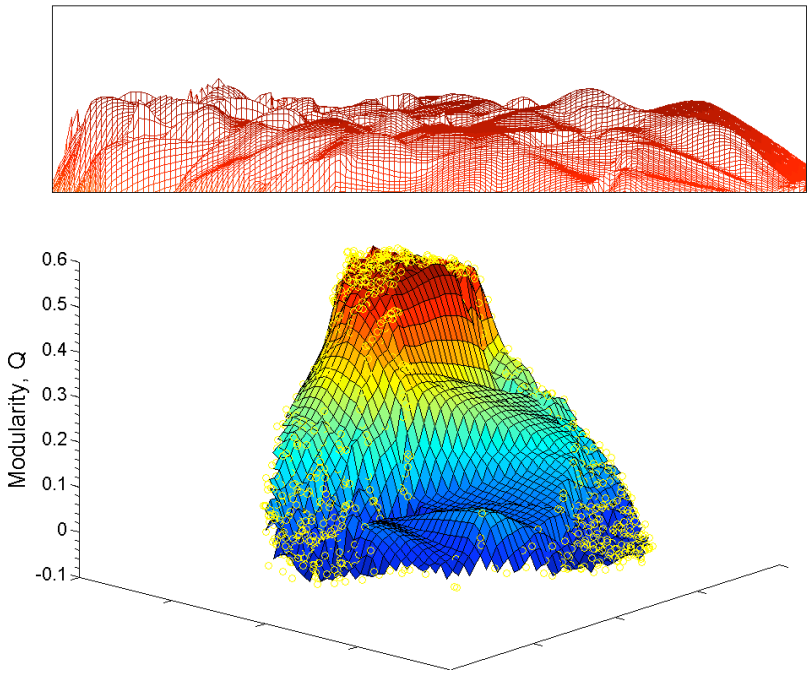
Question: What is the origin of the resolution limit?

Answer: global null model is unrealistic!



Limits of modularity

High degeneracy of large modularity partitions (Good et al., 2009)



1) Problem particularly severe on graphs with hierarchical structure
2) It explains why many heuristics give good estimates for the modularity maximum

Spectral methods

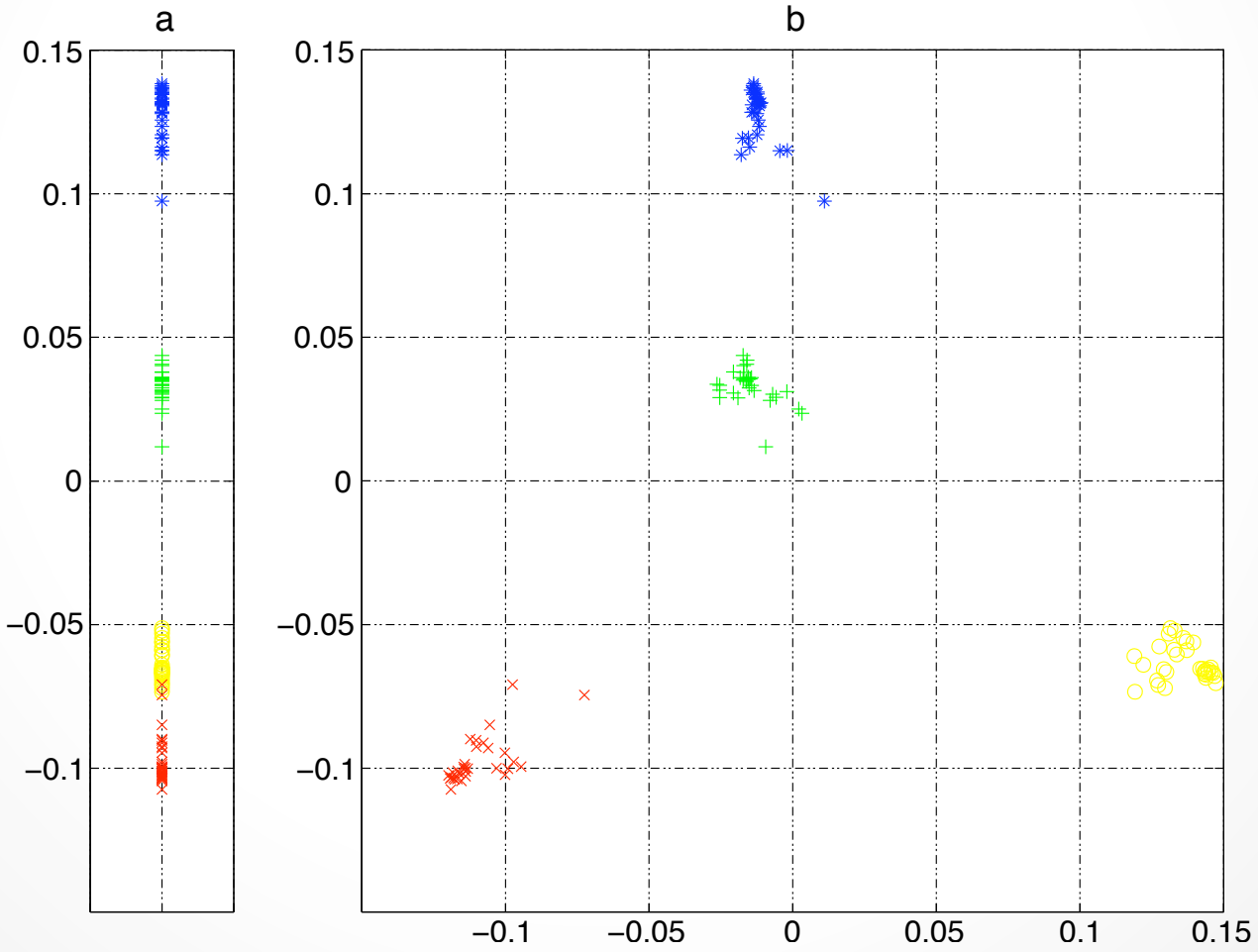
Finding communities from spectral properties of graph matrices: adjacency matrix, Laplacian matrix, etc.

Ex. Algorithm by Donetti & Muñoz (JSTAT, P10012, 2004)

Steps:

1) First few eigenvectors of Laplacian are computed (say k)
2) Eigenvector components used to represent vertices as points in k-dimensional Euclidean space
3) Hierarchical clustering used to group points
4) Modularity is used to pick best partition of resulting dendrogram

Spectral methods: Donetti & Muñoz



Dynamic methods

- Spin models
- Synchronization
- Random walks

Dynamic methods: spin models

Potts-model method (Reichardt & Bornholdt, 2004)

$$\mathcal{H} = -J \sum_{i,j} A_{ij} \delta(\sigma_i, \sigma_j) + \gamma \sum_{s=1}^{q} \frac{n_s(n_s - 1)}{2}$$

Ferromagnetic term: favors spins alignment (all vertices in the same cluster)

Antiferromagnetic term: favors (many) clusters of the same size

γ/J resolution parameter (usually set to edge density in applications)

Dynamic methods: random walks

Principle: in a graph with strong community structure, a random walker would spend a lot of time in a cluster before leaving it

Similarity measures can be defined through random walks, and hierarchical clustering can then be used

Examples of similarity measures:

1) Average number of edges that a random walker has to cross to reach j from i (Zhou, 2003)
2) Probability that the walker reaches j from i in a fixed number of steps (Latapy & Pons, 2005)
3) Commute-time, average first passage-time, escape probability, etc.

Dynamic methods: random walks

Markov Cluster Algorithm (MCL) (Van Dongen, PhD thesis, 2000)

Basic idea: diffusion flow on a network

$$W_{ij} \rightarrow S_{ij}$$

Transfer matrix

$$S_{ij} = W_{ij} / s_j$$

Dynamic methods: random walks

Three parameters: p, α, k

Steps:

1. (Diffusion) Raise the stochastic matrix to the power p (e.g. p=2)
2. (Inflation) Raise each resulting matrix element to the power α
3. Normalize the elements of the resulting matrix (by row)
4. Keep only the k largest elements per column
5. Repeat from 1.

Dynamic methods: random walks

After a sufficient number of iterations the matrix converges to a matrix with 0s and 1s, with disconnected components!

Problem: the final configuration depends on the parameters p, k and (mostly!) α

Complexity: $O(nk^2)$

http://www.micans.org/mcl/

Dynamic methods: synchronization

Principle: in a graph with strong community structure, oscillators placed at the vertices synchronize first within the clusters

Kuramoto oscillators

$$\frac{d\theta_i}{dt} = \omega_i + \sum_j K \sin(\theta_j - \theta_i)$$

On a graph, coupling between neighboring oscillators

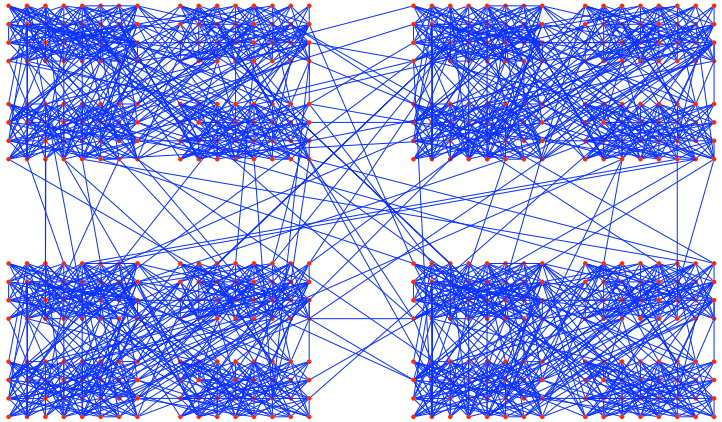
Steps:

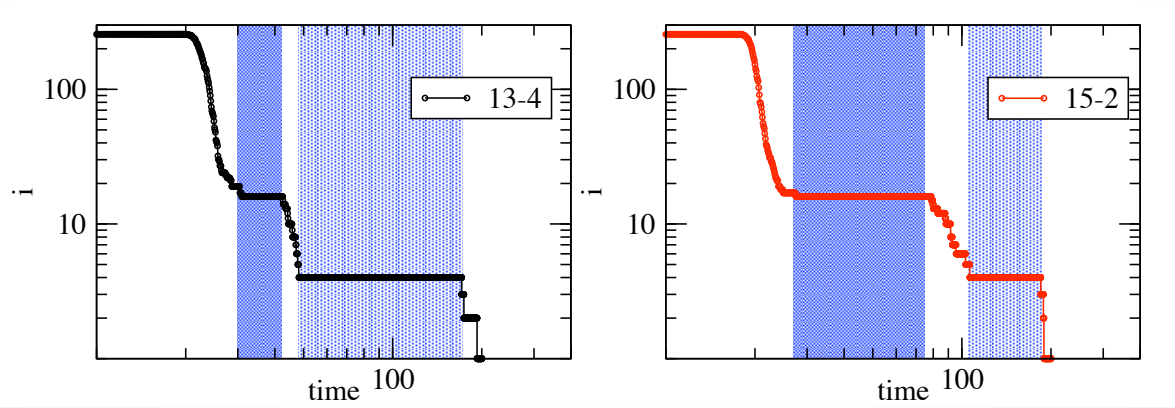
1) Initial configuration with random phases

2) For K larger than a threshold depending on the width of the distribution of the natural frequencies ω_i oscillators partially synchronize

Dynamic methods: synchronization

By following the dynamics over time, long-lived synchronized clusters emerge (Arenas et al., 2006)





Methods based on statistical inference

Bayesian inference

Two ingredients:

1) The evidence: information D that one has on the system (adjacency matrix)

2) A statistical model with parameters {θ}

Maximization of posterior probability

$$P(\{\theta\}|D) = \frac{1}{Z} P(D|\{\theta\}) P(\{\theta\})$$

$$Z = \int P(D|\{\theta\}) P(\{\theta\}) d\theta$$

Problems:

1) Computing Z is a challenge

2) Choice of prior distribution $P(\{\theta\})$

Methods based on statistical inference

Method by Newman & Leicht (2006)

g_i = group of vertex i

π_r = fraction of vertices in group r

θ_{ri} = probability of directed edge from vertices of group r and vertex i

Best classification corresponds to the maximum of the average likelihood that the model, with its parameters and $\{\pi_i\}$ fits $\{\theta_{ri}\}$ the adjacency matrix of the graph

Methods based on statistical inference

Equations of method by Newman & Leicht

$$q_{ir} = Pr(g_i = r|A, \pi, \theta) = \frac{\pi_r \prod_j \theta_{rj}^{A_{ij}}}{\sum_s \pi_s \prod_j \theta_{sj}^{A_{ij}}}$$

$$\pi_r = \frac{1}{n} \sum_i q_{ir}, \qquad \theta_{rj} = \frac{\sum_i A_{ij} q_{ir}}{\sum_i k_i q_{ir}}$$

Equations are self-consistent and can be solved by iteration starting from suitable initial conditions

Methods based on statistical inference

Complexity: parameter-dependent, but low (graphs with up to 10^6 vertices can be studied)

Plus: no need to specify group structure to search, the method recognizes if there is community structure, multipartite structure or combinations of both

Minus:

1) The number of groups to find must be given as input, the method is not able to find it on its own
2) Results strongly depend on initial conditions

Methods based on model selection

Model selection: finding a model which is simple and good enough for the system (ex. Curve fitting!)

No clear-cut recipe, several heuristics: Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), Minimum Description Length (MDL), etc.

MDL: minimizing length of description of system/clustering for a given coding scheme

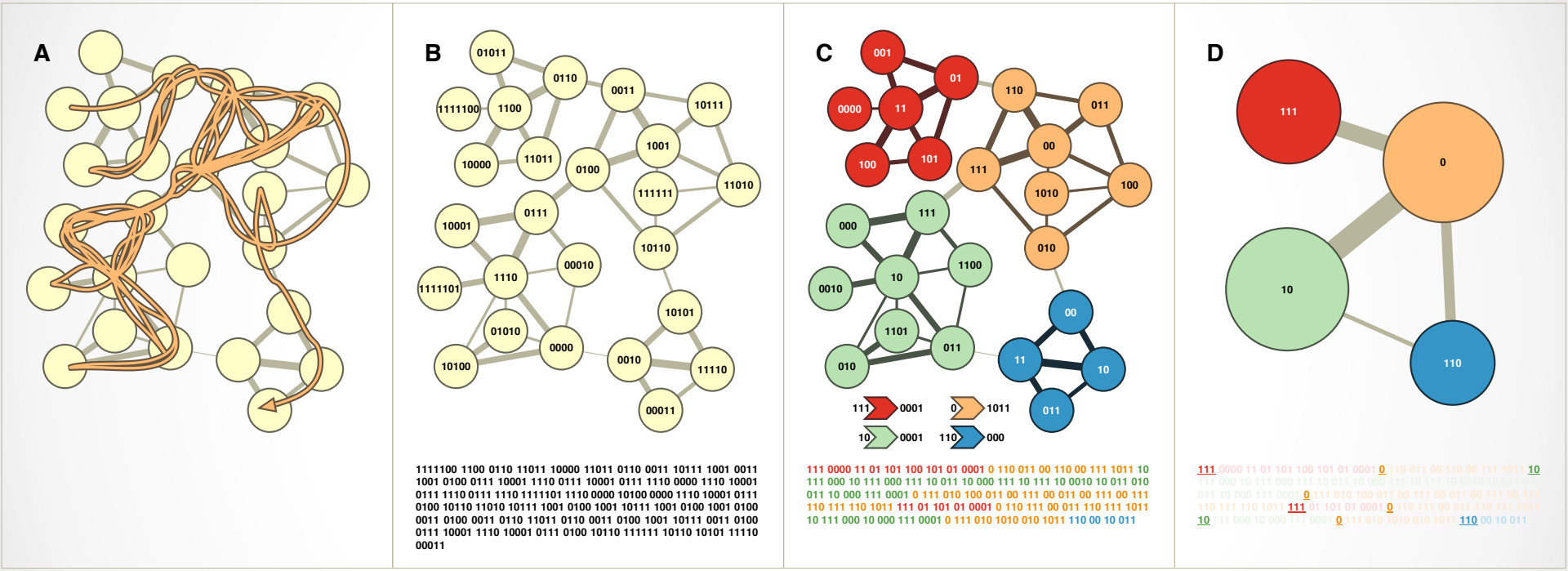
Infomap

Rosvall & Bergstrom, 2008

Idea: finding a compressed description of a random walk taking place on the graph

Procedure:
1)Each vertex is given a coded name (Huffman code)
2)Each cluster receives a coded name
3)Names of vertices can be recycled, as long as they are not repeated in the same cluster (just like in geographic maps)
4)The recycling procedure enables to spare the space required by assigning a different name to each vertex
5)When a vertex passes from one cluster to another one must indicate the name of the new cluster
6)If the graph has a strong community structure, recycling the vertex names is convenient

Infomap



Best partition ➔ minimum description length, optimization can be carried out with simulated annealing, greedy methods, etc.

Methods to find overlapping communities

Clique Percolation Method (CPM) (Palla et al., 2005)

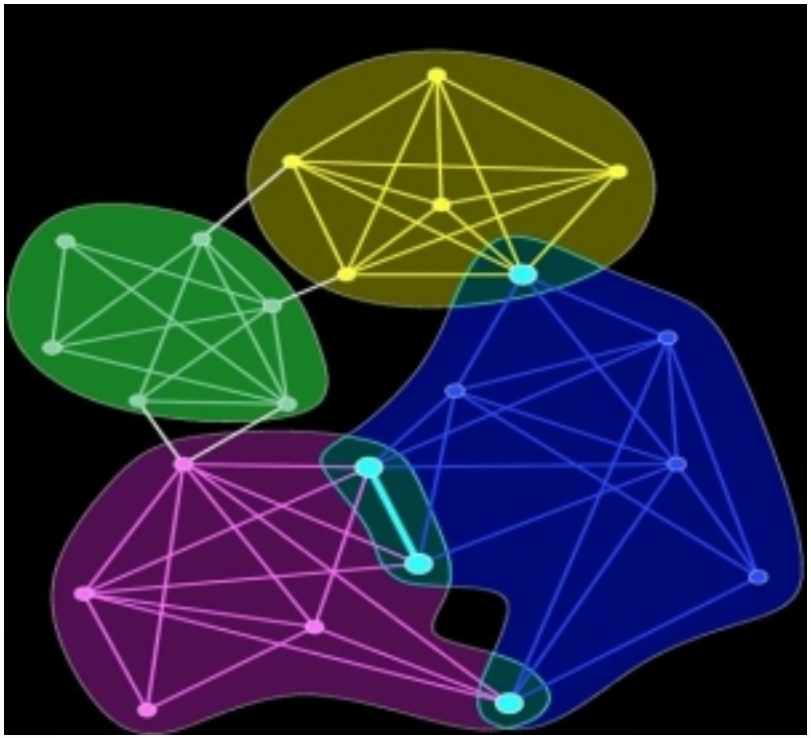
Principle: in a graph with community structure there are many cliques within the clusters

Cliques can be used as probes to explore the graph:
1) Two k-cliques are neighbors if they share a (k-1)-clique
2) One can travel along paths of neighboring cliques

Cliques may be trapped within clusters, which can then be identified

Methods to find overlapping communities

Clique Percolation Method



Methods to find overlapping communities

Complexity: finding all k-cliques on sparse graphs can be done quickly

Problems:

1) Results strongly depend on the density of cliques, and may be trivial if there are too many or too few
2) Vertices with less than k-1 neighbors cannot be reached by k-cliques and remain unclassified
3) Which value of k?

Methods to find overlapping communities

Local Fitness Method (LFM) (Lancichinetti et al., 2009)

Principle: finding local communities about individual vertices

A local community is built by maximizing a fitness function

$$f_i = \frac{k_{in}^i}{(k_{in}^i + k_{out}^i)^\alpha}$$

Fitness of vertex A with respect to cluster i

$$f_i^A = f_{i \cup A} - f_{i - A}$$

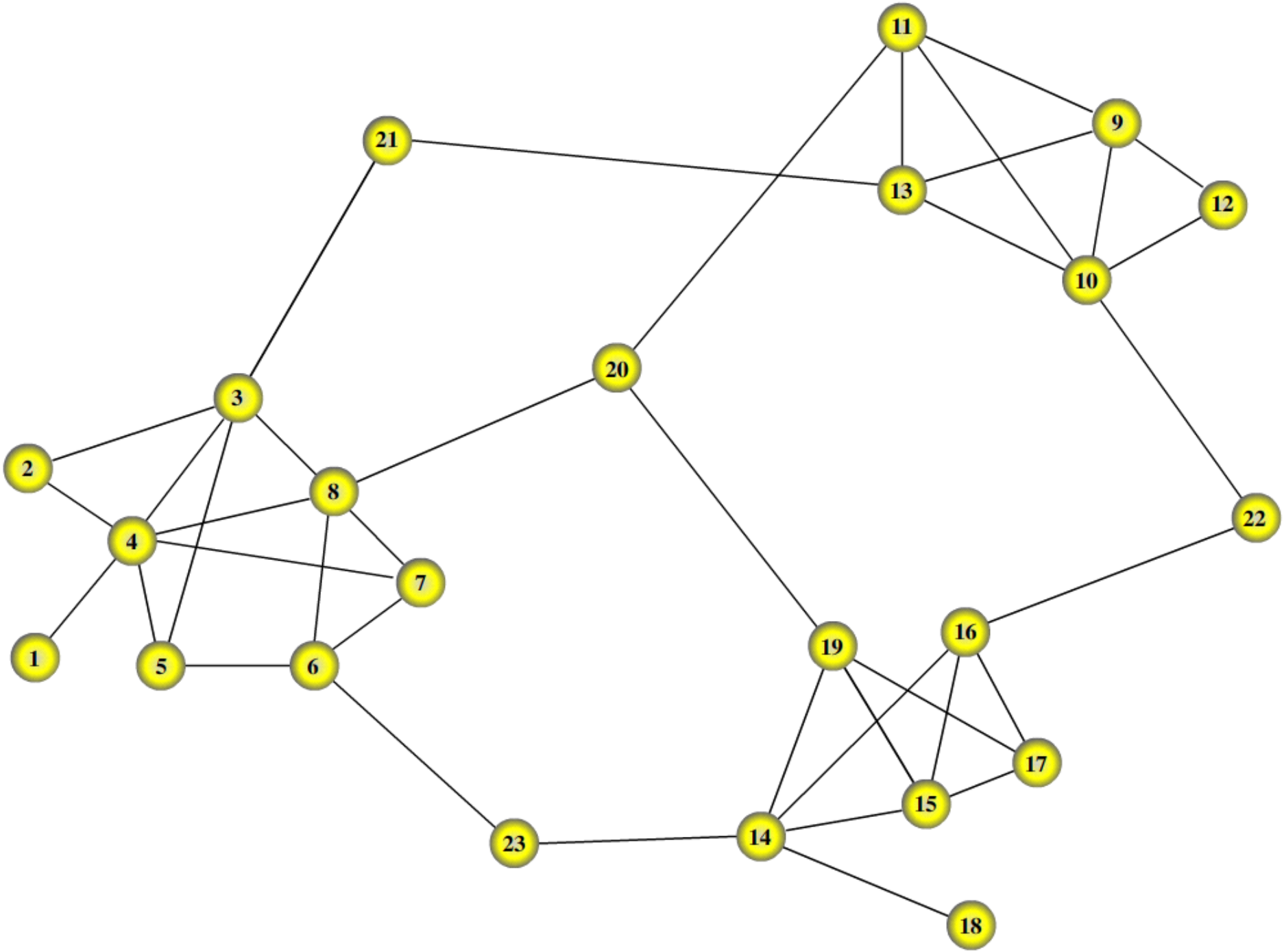
Methods to find overlapping communities

Steps:

1) Take a vertex A at random
2) Look for community of A
3) Pick a vertex B at random not yet assigned to a community
4) Find community of B, it may overlap with the other communities
5) Go on until all vertices have been assigned to at least a community

How to build a cluster:

1) Start: cluster with s vertices
2) The neighboring vertex with largest positive fitness is included in the cluster; fitness of all vertices is recalculated
3) Vertices with negative fitness are removed
4) Process goes on until all vertices of the group have positive fitness and all their neighbors negative fitness


11
9
21
13
12
10
20
3
2
8
22
4
7
16
19
1
5
6
17
15
23
14
18

OSLOM

Basics:

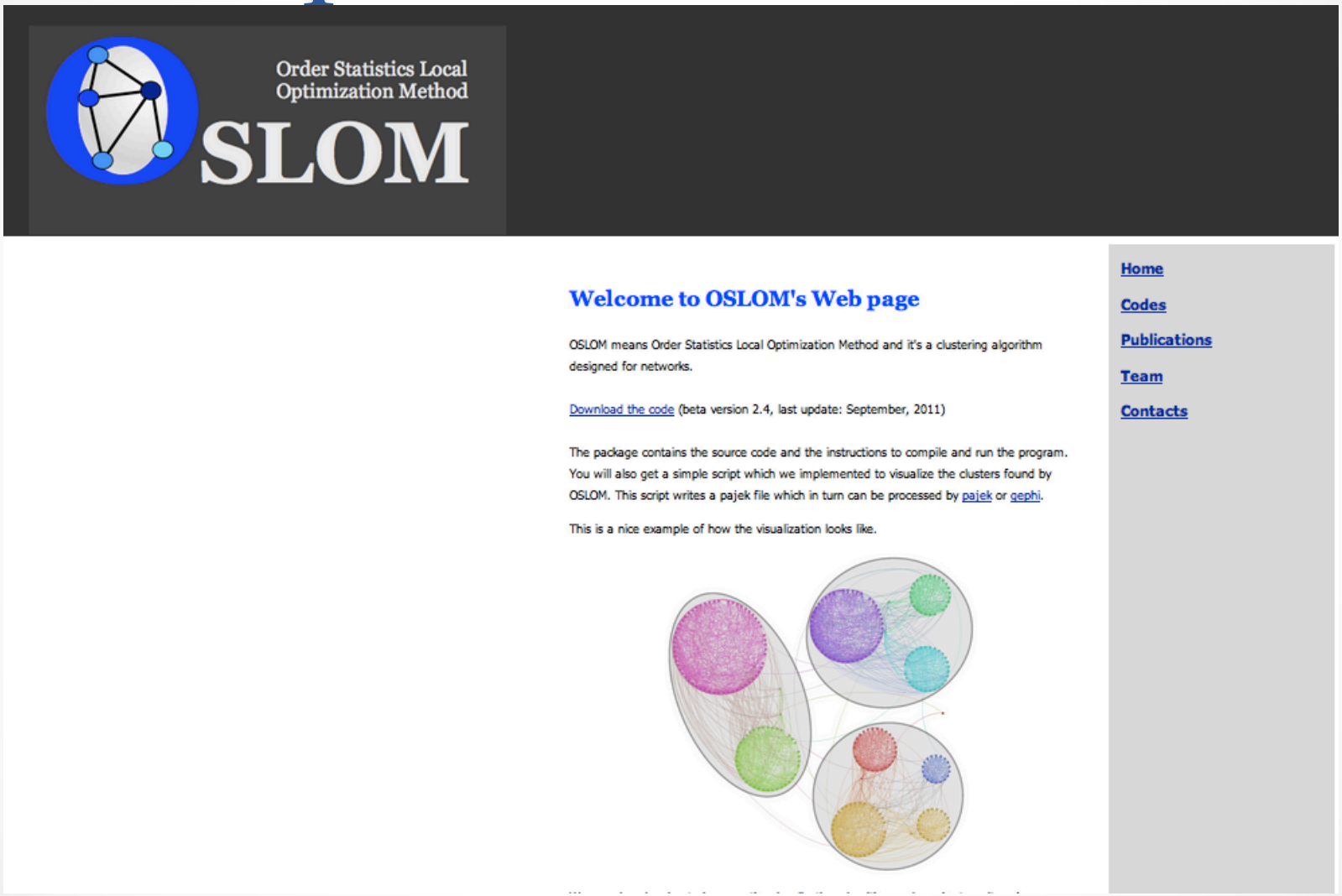
- LFM with fitness expressing the statistical significance of a cluster with respect to random fluctuations
- Statistical significance evaluated with Order Statistics

First multifunctional method:

- Link direction
- Link weight
- Overlapping clusters
- Hierarchy

A. Lancichinetti, F. Radicchi, J. J. Ramasco, S. F., PLoS One 6, e18961 (2011)

Local optimization: OSLOM



http://www.oslom.org/

Multiresolution methods and cluster hierarchy

Question: most real networks have hierarchical structure, but most methods find just one partition, what to do?

(Possible) Answer: introducing a tunable parameter so that the method yields partitions at different scales

Examples:

1) Spin glass modularity by Reichardt & Bornholdt (2006)
2) Multiresolution modularity by Arenas et al. (2008)
3) LFM method by Lancichinetti et al.

Multiresolution methods and cluster hierarchy

Method by Ronhovde & Nussinov (2009)

Potts model: rewarding edges within clusters and non-edges between clusters

$$\mathcal{H}(\{\sigma\}) = -\frac{1}{2}\sum_{i \neq j}[A_{ij} - \gamma(1 - A_{ij})]\delta(\sigma_i, \sigma_j)$$

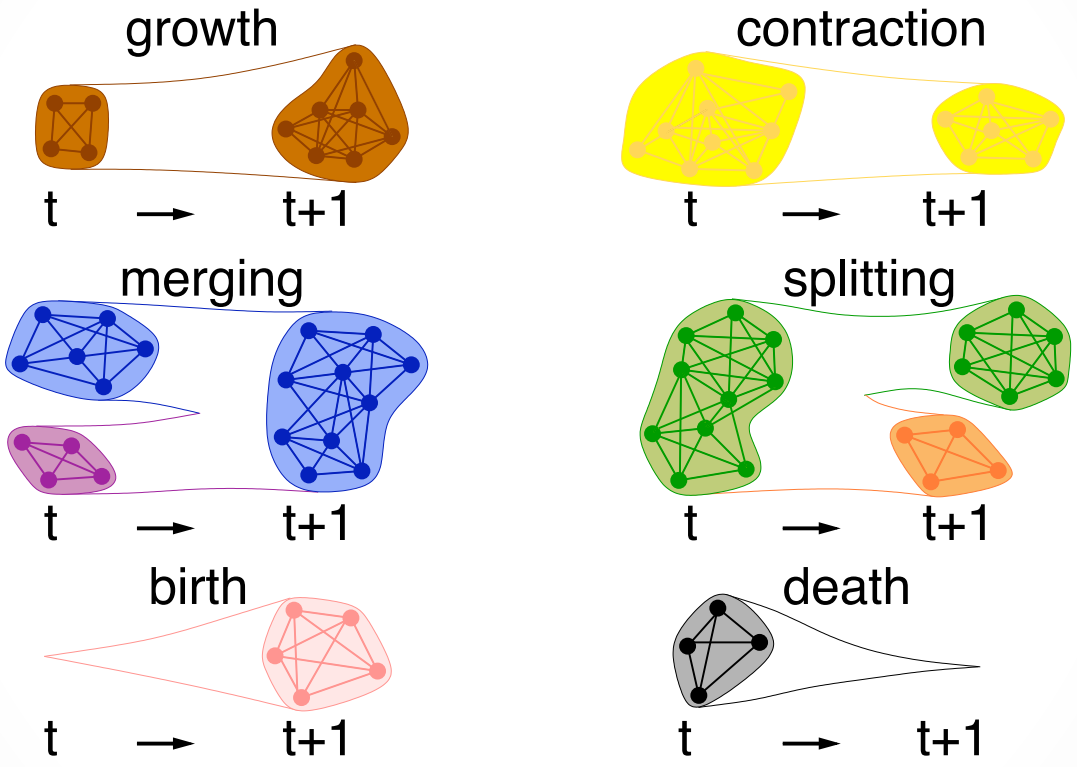
No null model!

Energy is minimized by shifting single vertices to the clusters that yield the largest decrease of $\mathcal{H}(\{\sigma\})$

Complexity: $O(m^{\beta}),\ \beta \sim 1$

Meaningful partitions: values of γ yielding most "stable" partitions

Detection of dynamic communities



Problem: how to track the images of a community at various times?

$$\mathcal{C}(t) \rightarrow \mathcal{C}(t+1) \ ?$$

Detection of dynamic communities

Analysis by Palla et al. (2007)

Datasets: mobile communication network and scientific collaboration network

Method: CPM

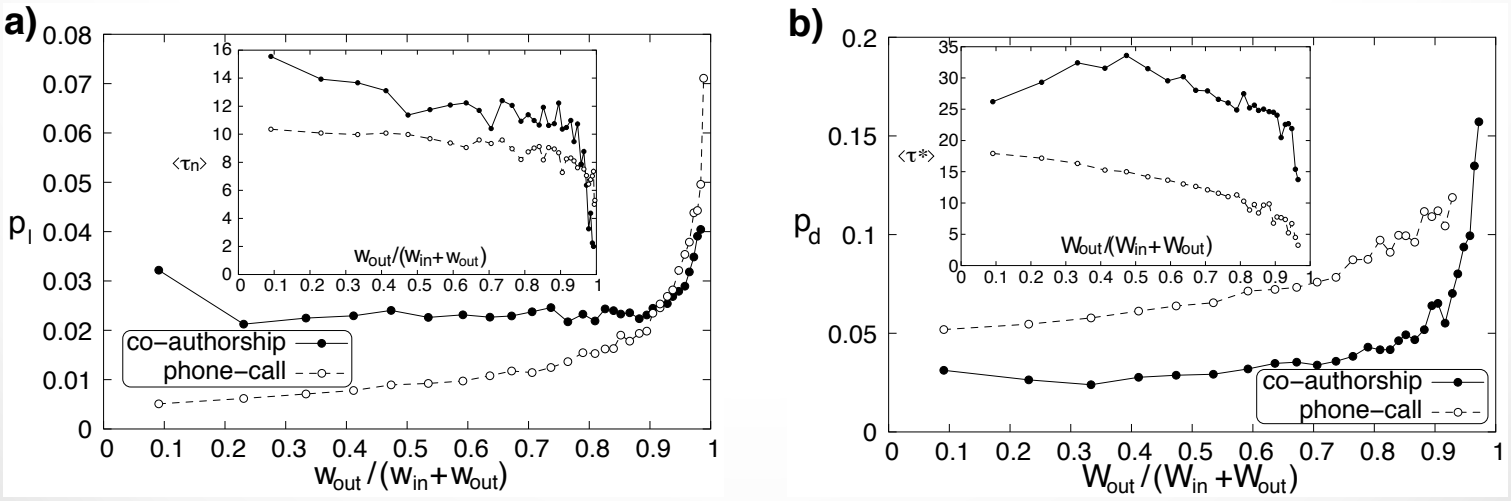
Community tracking:

1) Take the graph $\mathcal{G}(t, t+1)$, made by the union of the graphs $\mathcal{G}(t)$ and $\mathcal{G}(t+1)$ at times t and t+1, find the communities in it: they include communities of both snapshots at time t and t+1
2) Given a community $\mathcal{C}(t)$ of $\mathcal{G}(t)$ its image in $\mathcal{G}(t+1)$ is the community of $\mathcal{G}(t+1)$ having the largest overlap with $\mathcal{C}(t)$ among those included in the community of $\mathcal{G}(t, t+1)$ including

Detection of dynamic communities

Results:

1) Large communities are more variable, small communities essentially static
2) Vertices which are weakly connected to their community have a sizeable chance to leave it
3) Vertices which are tightly connected to an external community have a high chance to join it
4) Results 3 and 4 hold at the community level as well



Detection of dynamic communities

Most methods are two-stage, like that by Palla et al.

Alternative approach: evolutionary clustering (Chakrabarti et al., 2006)

Unified framework: partition derived both from information at time t and from information at previous times

Ingredients:

1) Snapshot quality of a partition: goodness of the partition with respect to the graph structure at a given time t
2) History cost: measure of distance of partition at time t from partition at time t-1

Principle: a good partition should have high snapshot quality and low history cost

Testing algorithms

Question: how to test clustering algorithms?

Answer: checking whether they are able to recover the known community structure of benchmark graphs

Warning: definition of community of benchmark and methods should be consistent!

Planted l-partition model (Condon & Karp, 1999)

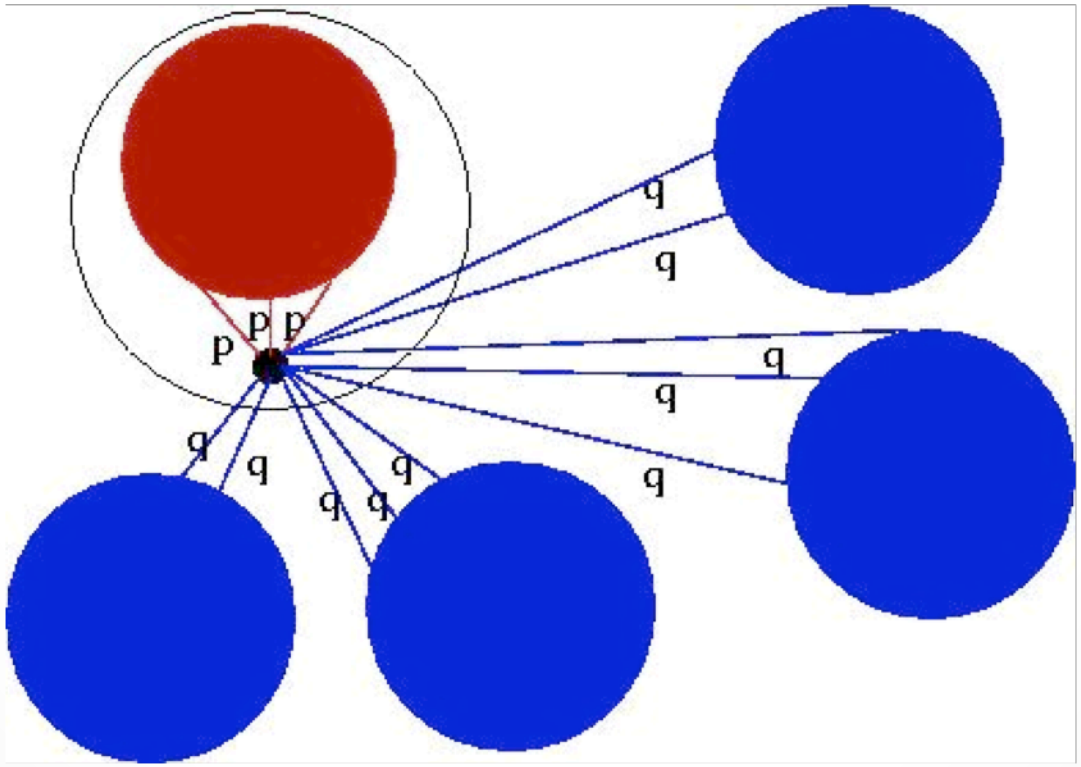
Ingredients:

1) n vertices, l equal-sized groups with g=n/l vertices each
2) p=probability that vertices of the same cluster are joined
3) q=probability that vertices of different clusters are joined

Idea: if p>q the groups are communities

Testing algorithms

Planted l-partition model



$$\langle k \rangle = p(g-1) + qg(l-1)$$

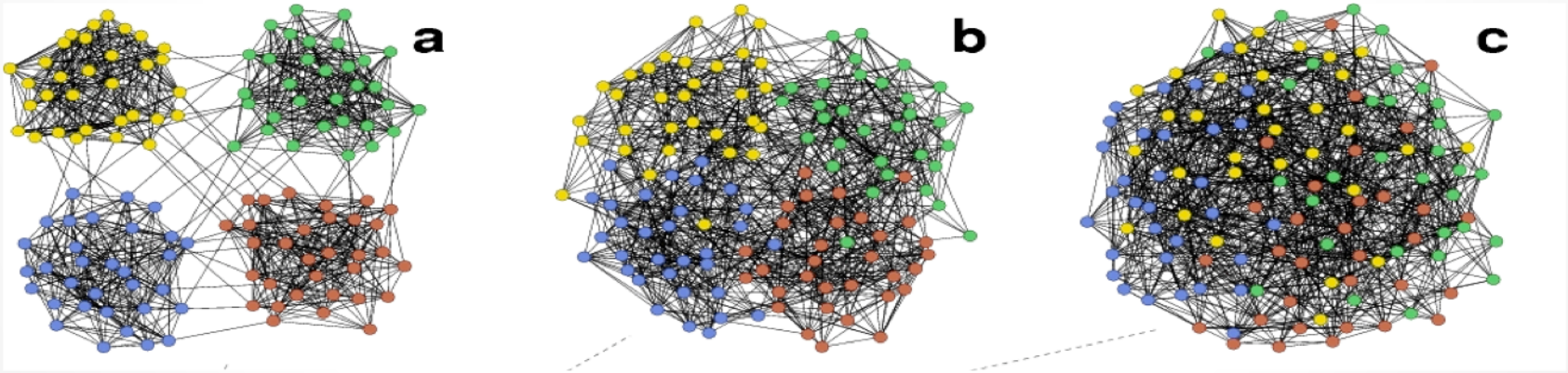
Testing algorithms

Special case: the benchmark by Girvan and Newman (2002)

n=128, l=4, g=32

$$k_{in} = p(g-1) \sim pg \quad k_{out} = qg(l-1) \quad k_{in} + k_{out} = 16$$

$$p \geq q \rightarrow k_{in} \geq 4, \; k_{out} \leq 12$$



Problems:

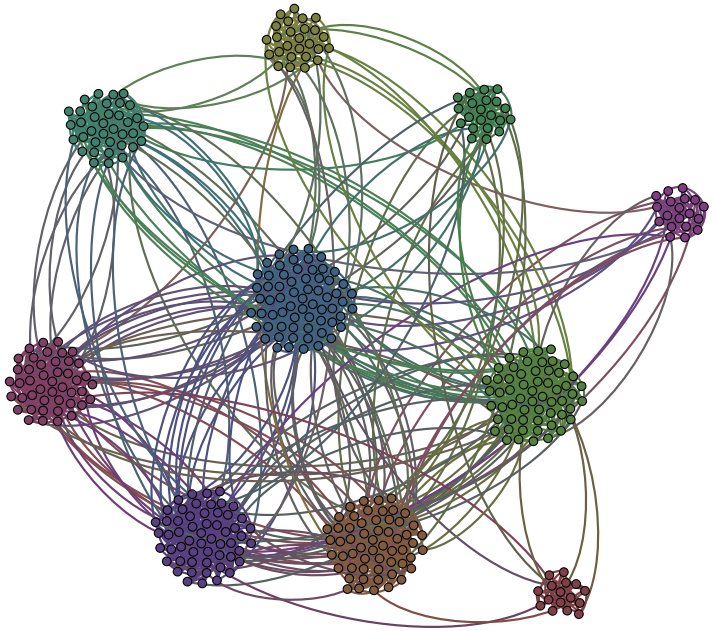
1) All vertices have the same degree
2) All communities have the same size

Testing algorithms

LFR benchmark (Lancichinetti et al., 2008)

Features:

1) Power law distribution of degree (exponent τ_1)
2) Power law distribution of community size (exponent τ_2)
3) A mixing parameter μ_t sets the ratio between external and total degree of each vertex



https://sites.google.com/site/andrealancichinetti/files/

Testing algorithms

Necessary ingredient: similarity measure between partitions

Ex. Normalized mutual information

x_i, y_i : community assignments

$P(X = x) = n_x/n, \;\; P(Y = y) = n_y/n$

$$H(X) = -\sum_x P(x) \log P(x)$$ Shannon entropy

$$H(X|Y) = -\sum_{xy} P(x,y) \log P(x|y)$$ Shannon conditional entropy

$$I(X,Y) = H(X) - H(X|Y)$$ Mutual information

Problem: mutual information identical for all Y subpartitions of X

$$I_{norm}(X,Y) = \frac{2I(X,Y)}{H(X) + H(Y)}$$ Normalized mutual information

Testing algorithms

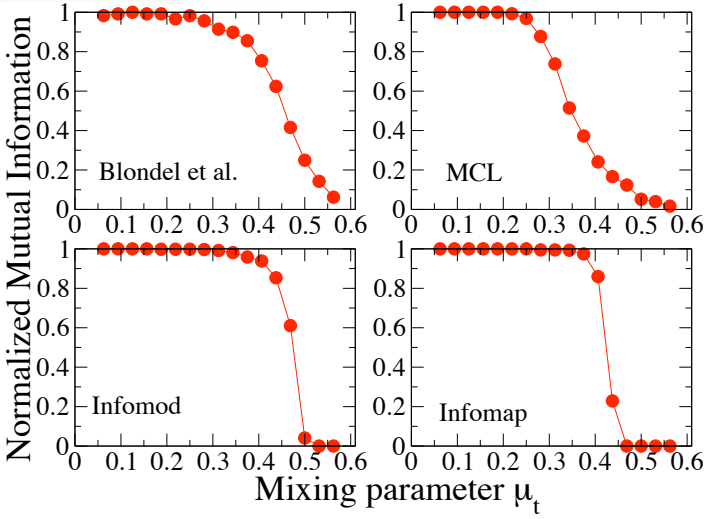
First analysis: Danon et al. (2005), using GN benchmark

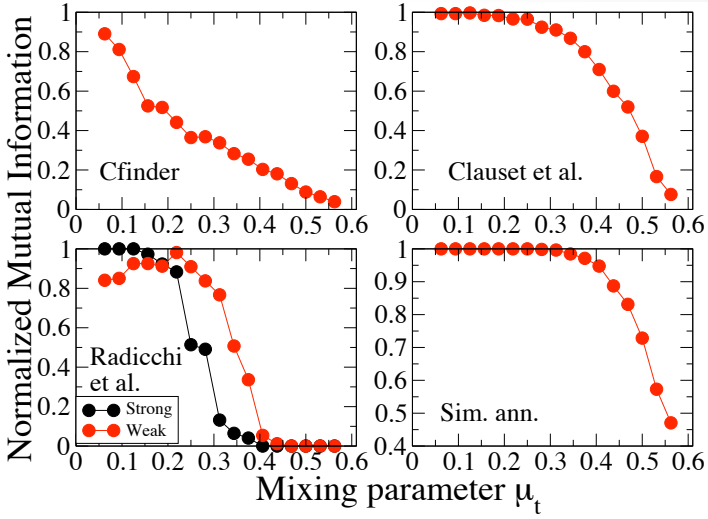
New analysis: Lancichinetti & Fortunato (2009), using LFR benchmark

Author	Label	Order
Girvan & Newman	GN	$O(nm^2)$
Clauset et al.	Clauset et al.	$O(n \log^2 n)$
Blondel et al.	Blondel et al.	$O(m)$
Guimerà et al.	Sim. Ann.	parameter dependent
Radicchi et al.	Radicchi et al.	$O(m^4/n^2)$
Palla et al.	Cfinder	$O(\exp(n))$
Van Dongen	MCL	$O(nk^2)$, $k < n$ parameter
Rosvall & Bergstrom	Infomod	parameter dependent
Rosvall & Bergstrom	Infomap	$O(m)$
Donetti & Muñoz	DM	$O(n^3)$
Newman & Leicht	EM	parameter dependent
Ronhovde & Nussinov	RN	$O(n^\beta)$, $\beta \sim 1$

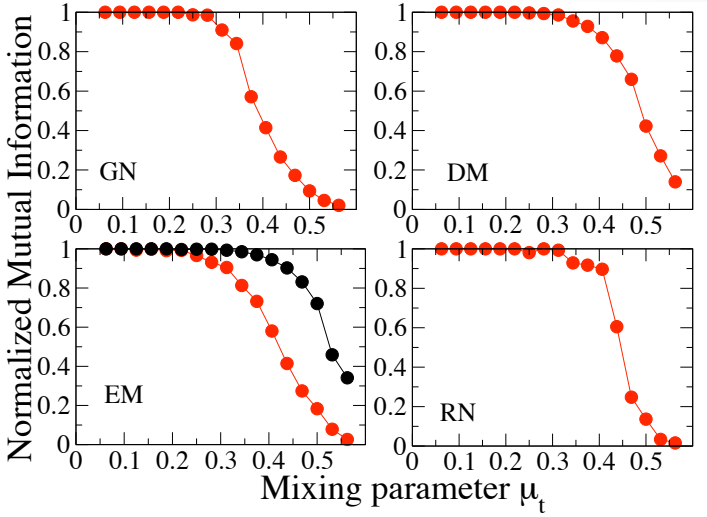
Testing algorithms

A comparative analysis



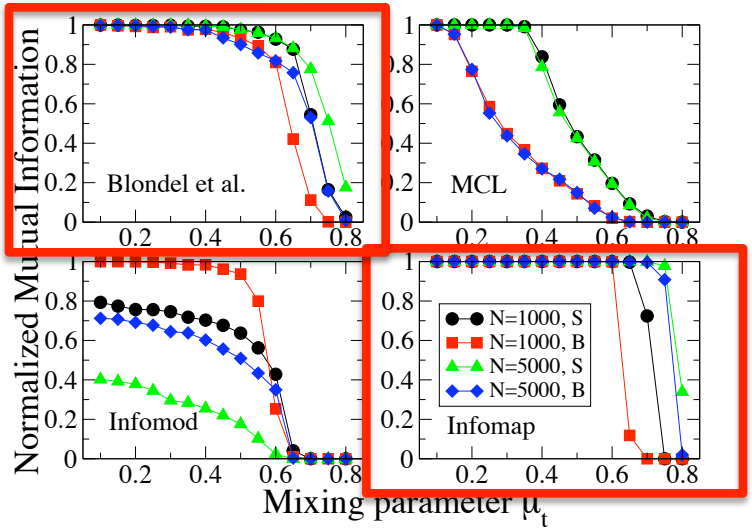


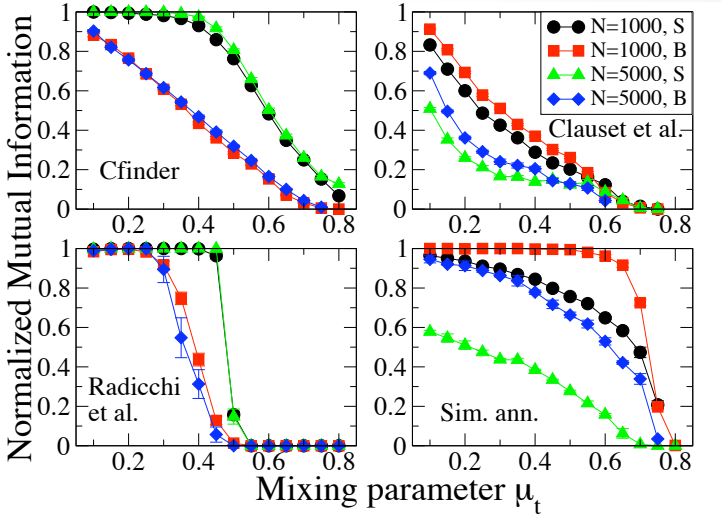
GN benchmark



Testing algorithms

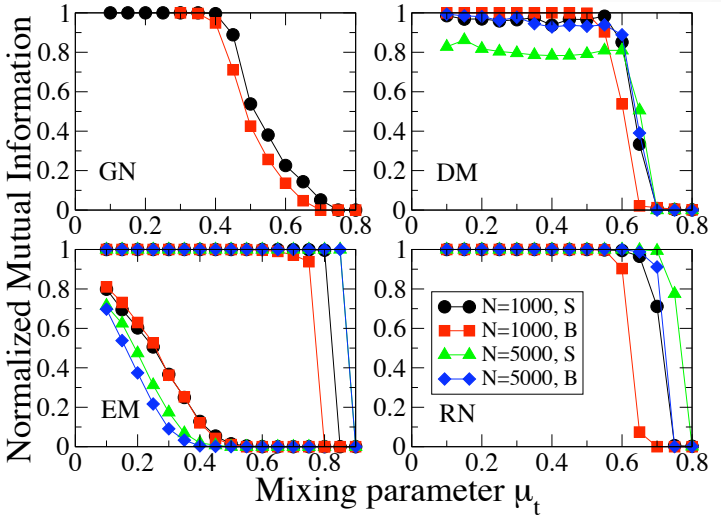
A comparative analysis





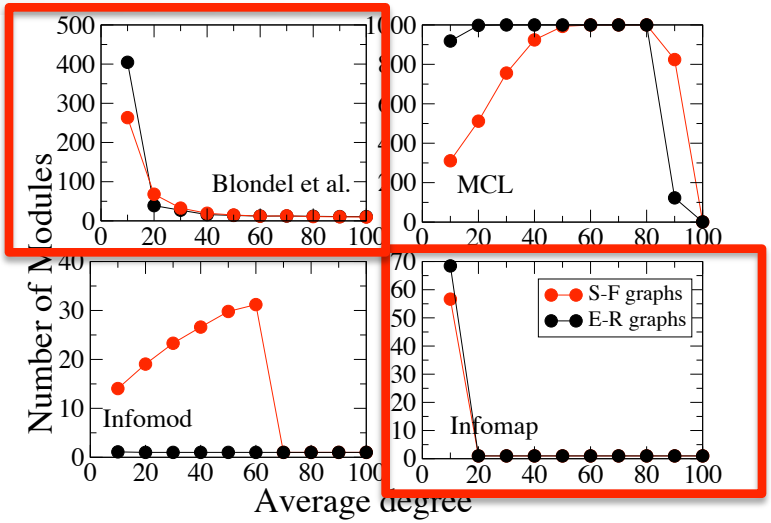
... and the winner is:

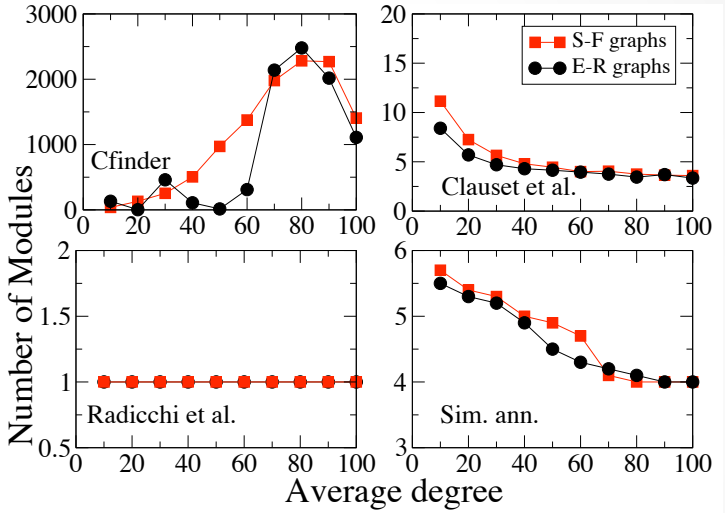
- Infomap
- Louvain method *



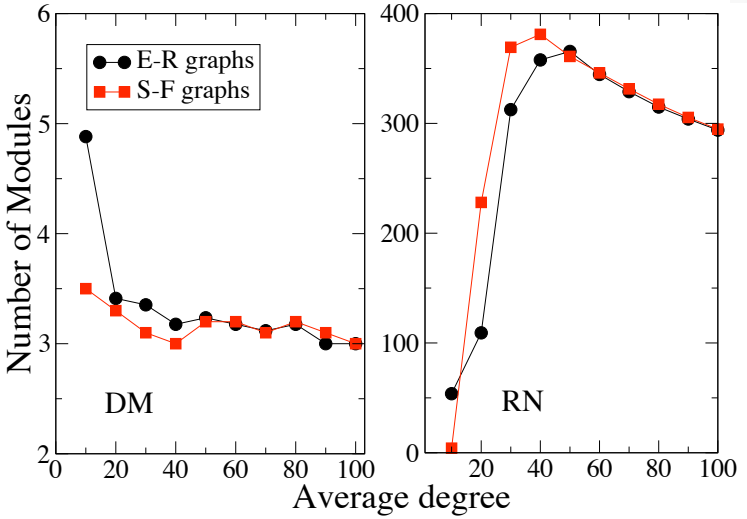
Testing algorithms

A comparative analysis





Random graphs: no clusters!

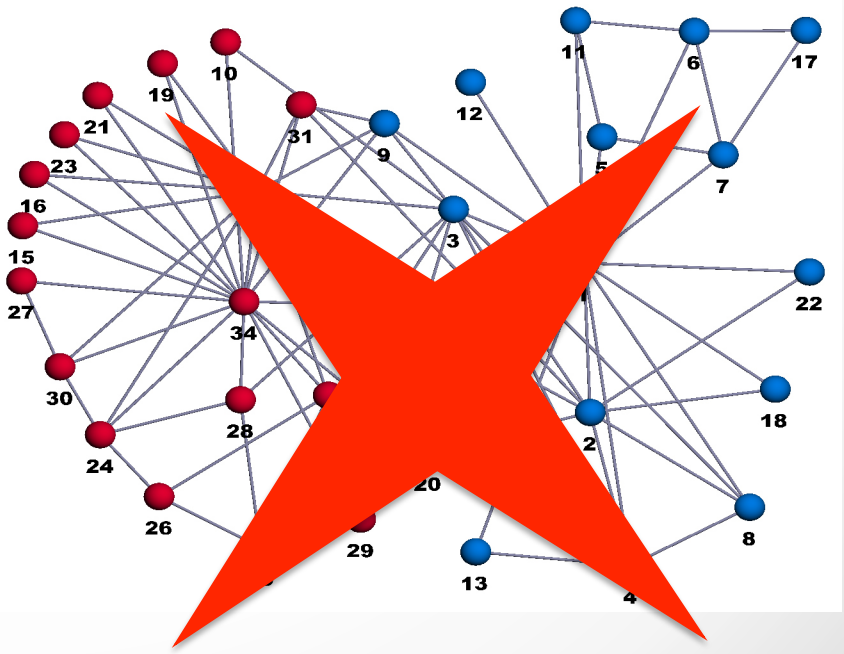


Topology vs metadata

Zero-th postulate of community detection: communities group nodes with similar properties!

Is that true? Never tested, lack of data

Zachary's karate club?

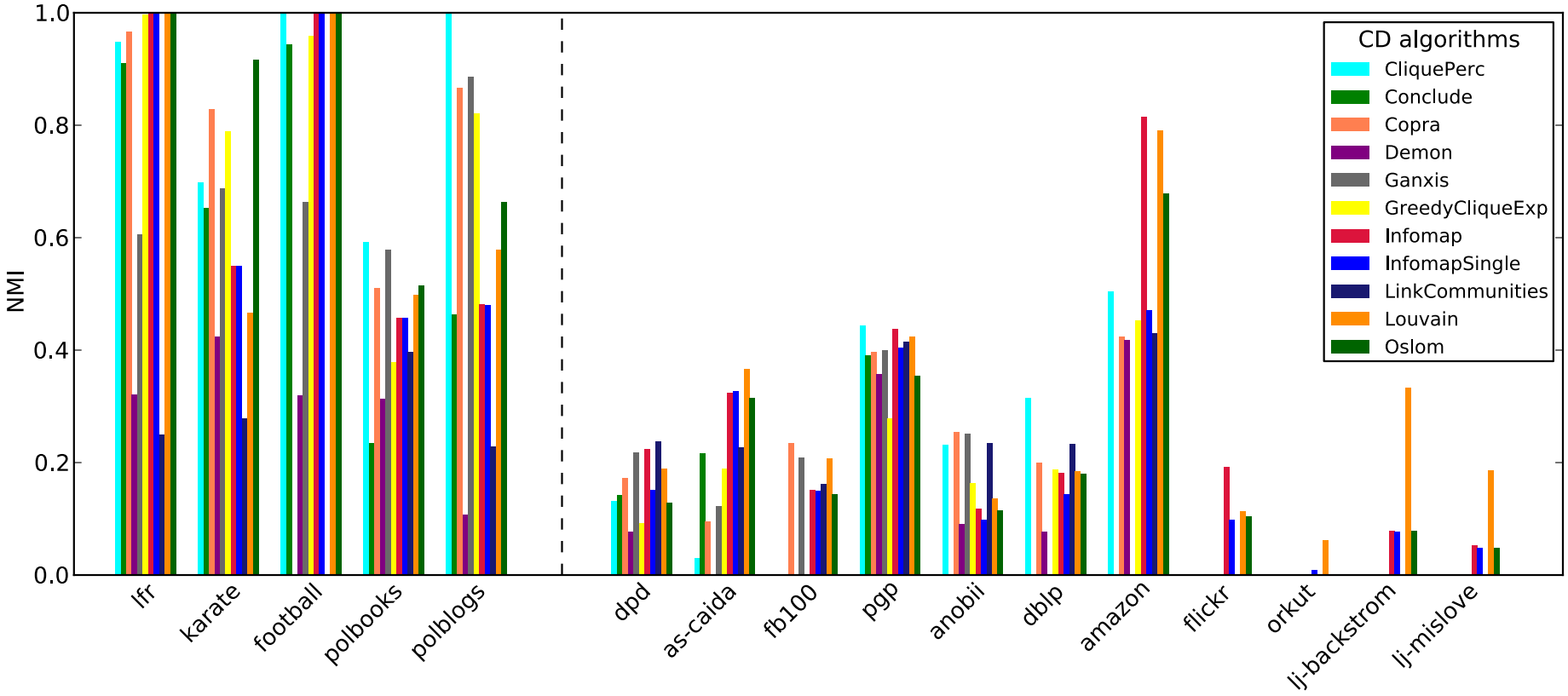


Topology vs metadata

Name	#Nodes	#Edges	#Communities	Description of community nature
lfr	1000	9839	40	artificial network (lfr, 1000S, $\mu = 0.5$)
karate	34	78	2	membership after the split
football	115	615	12	team scheduling groups
polbooks	105	441	2	political alignment
polblogs	1222	16782	3	political alignment
dpd	35029	161313	580	software package categories
as-caida	46676	262953	225	countries
fb100	762–41536	16651–1465654	2–2597	common students' traits
pgp	81036	190143	17824	email domains
anobii	136547	892377	25992	declared group membership
dblp	317080	1049866	13472	publication venues
amazon	366997	1231439	14–29432	product categories
flickr	1715255	22613981	101192	declared group membership
orkut	3072441	117185083	8730807	declared group membership
lj-backstrom	4843953	43362750	292222	declared group membership
lj-mislove	5189809	49151786	2183754	declared group membership

D. Hric, R. K. Darst, S. F., Phys. Rev. E 90, 062805 (2014)

Topology vs metadata



Mismatch between structural (detected) clusters and ground truth clusters -> new models/methods ?

D. Hric, R. K. Darst, S. F., Phys. Rev. E 90, 062805 (2014)

Consensus clustering

- Stochastic (non-deterministic) methods yield many result partitions: which one shall one choose?
- Optimization methods have an intrinsic criterion to pick the "best" partition (minimization/maximization of quality function) -> can one exploit the information of the discarded partitions as well?

Similar problem: clustering in dynamic networks

- Many methods use individual snapshots, ideally one should combine the information of different snapshots

Consensus clustering

Goal

- Searching for the partition which is most similar, on average, to the input partitions (*median* or *consensus partition*)

Similarity measurable with, e.g., Normalized Mutual Information (NMI)

Problem: difficult combinatorial optimization task

Greedy solution: consensus matrix

A. Lancichinetti, S. Fortunato, Sci. Rep. **2**, 336 (2012)

Consensus clustering

Definition

- Matrix **D** whose entry $\mathbf{D}_{ij}$ is the frequency that vertices i and j were in the same cluster in the input partitions

Idea

- Finding the clusters of **D** by applying the same method used to find the input partitions
- The resulting partitions are combined in a new consensus matrix **D′**
- The procedure is iterated until the consensus matrix is composed of disconnected cliques

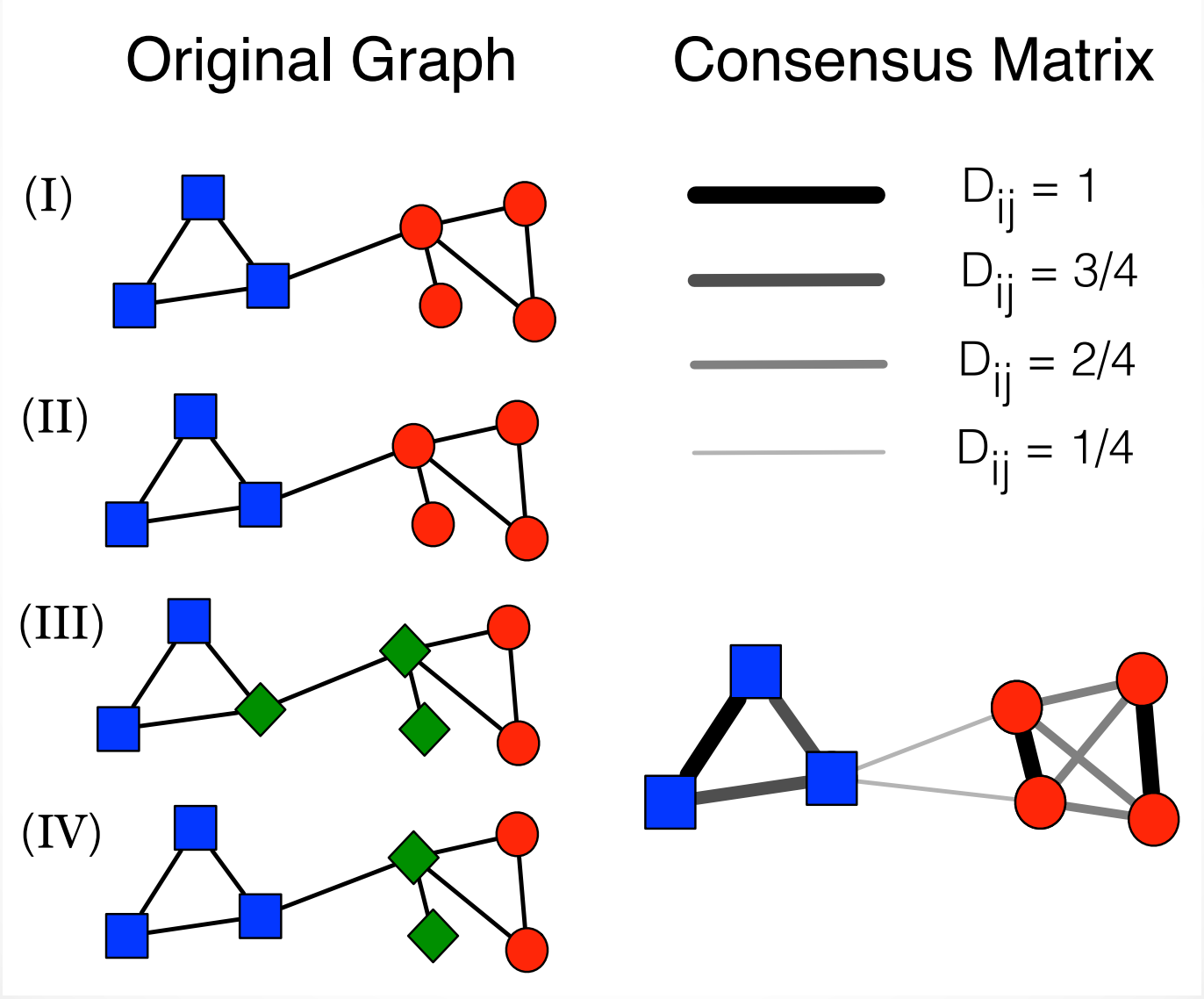
Consensus clustering

The algorithm

Starting point: network G with n vertices, clustering method A.

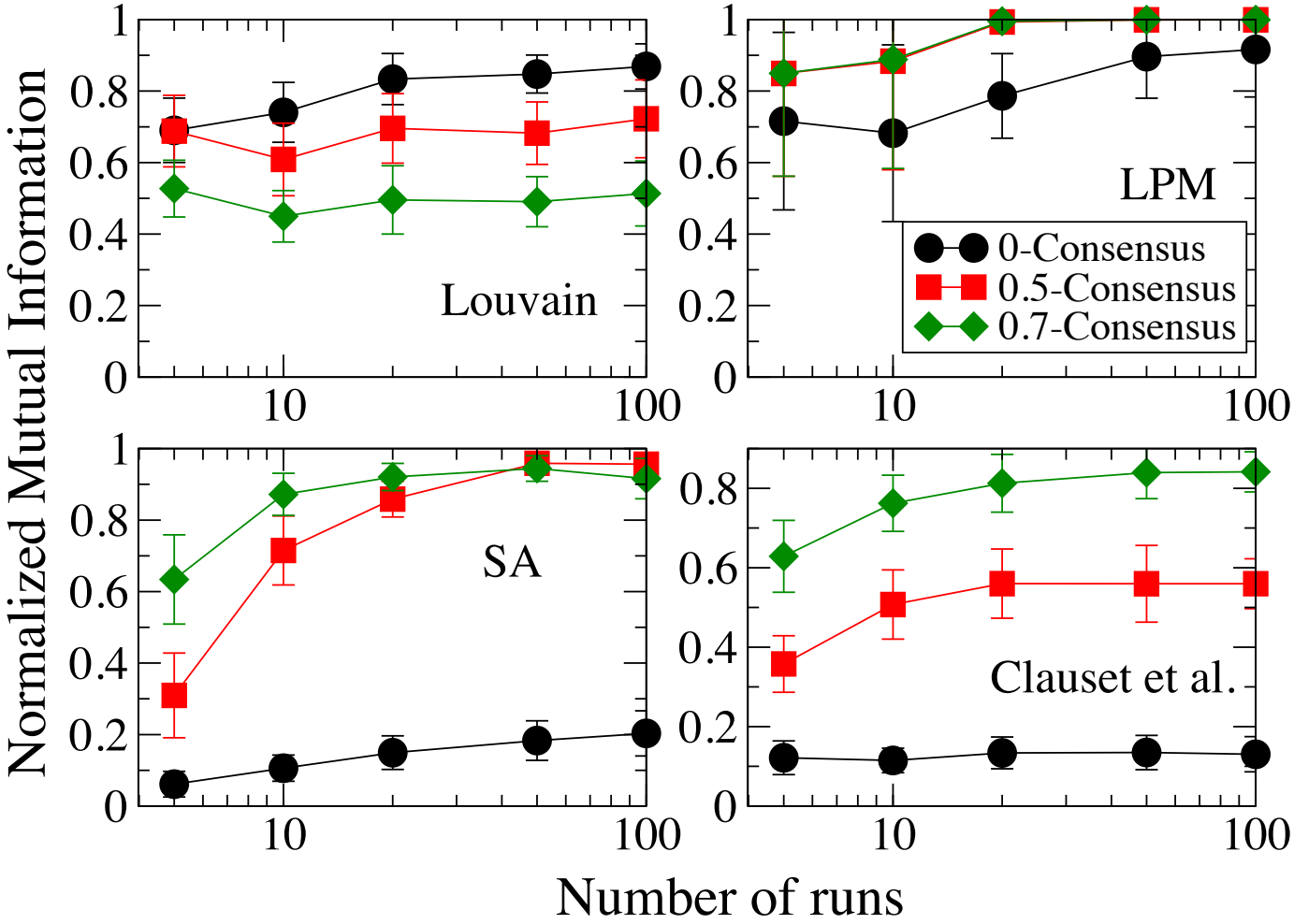
- Apply A on G n_P times -> n_P partitions
- Compute the consensus matrix **D**: D_{ij} is the number of partitions in which vertices i and j of G are assigned to the same cluster, divided by n_P
- All entries of **D** below a chosen threshold t are set to zero
- Apply A on **D** n_P times -> n_P partitions
- If the partitions are all equal, stop (the consensus matrix would be block-diagonal). Otherwise go back to 2.

Consensus clustering



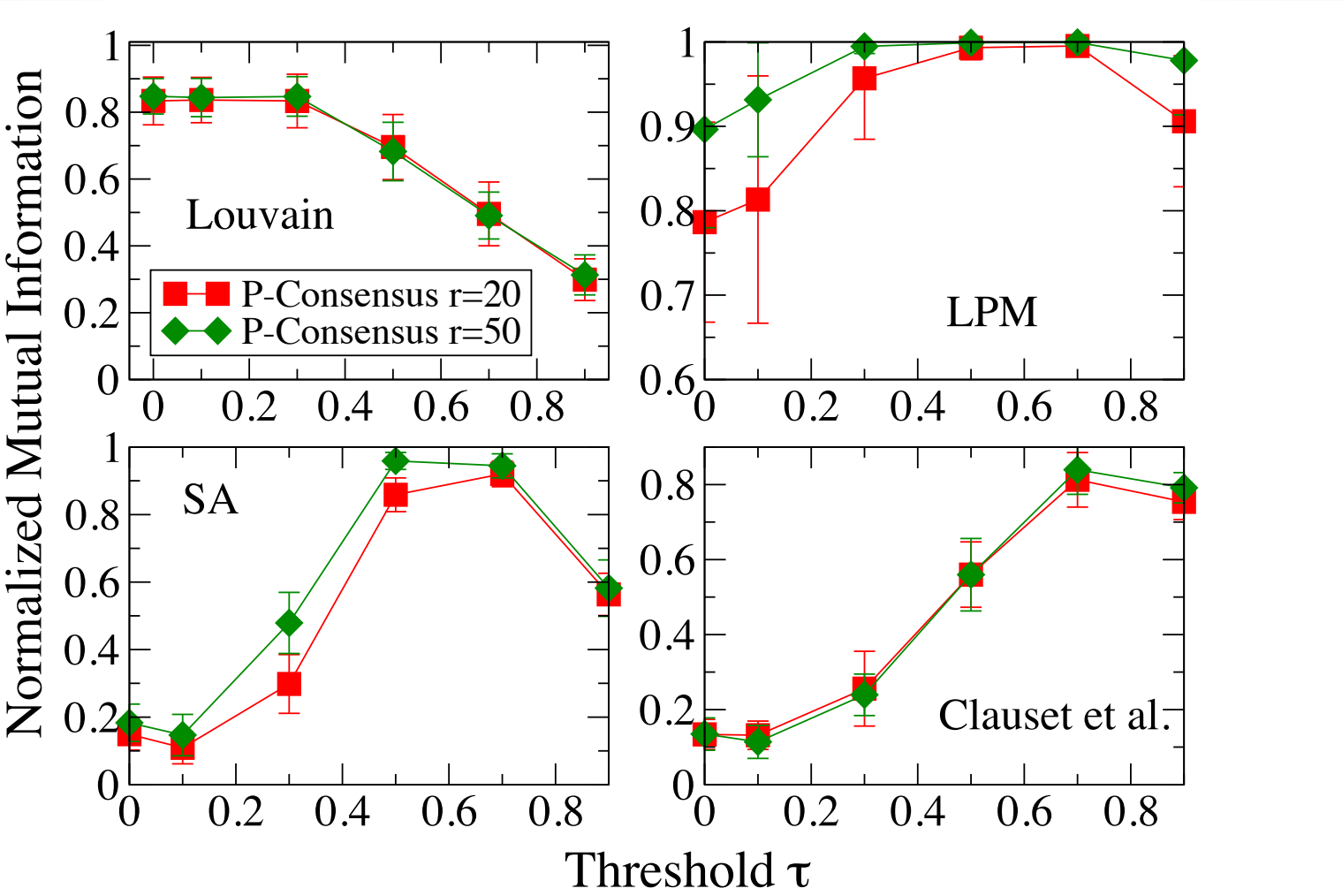
Consensus clustering

Setting the number of inputs



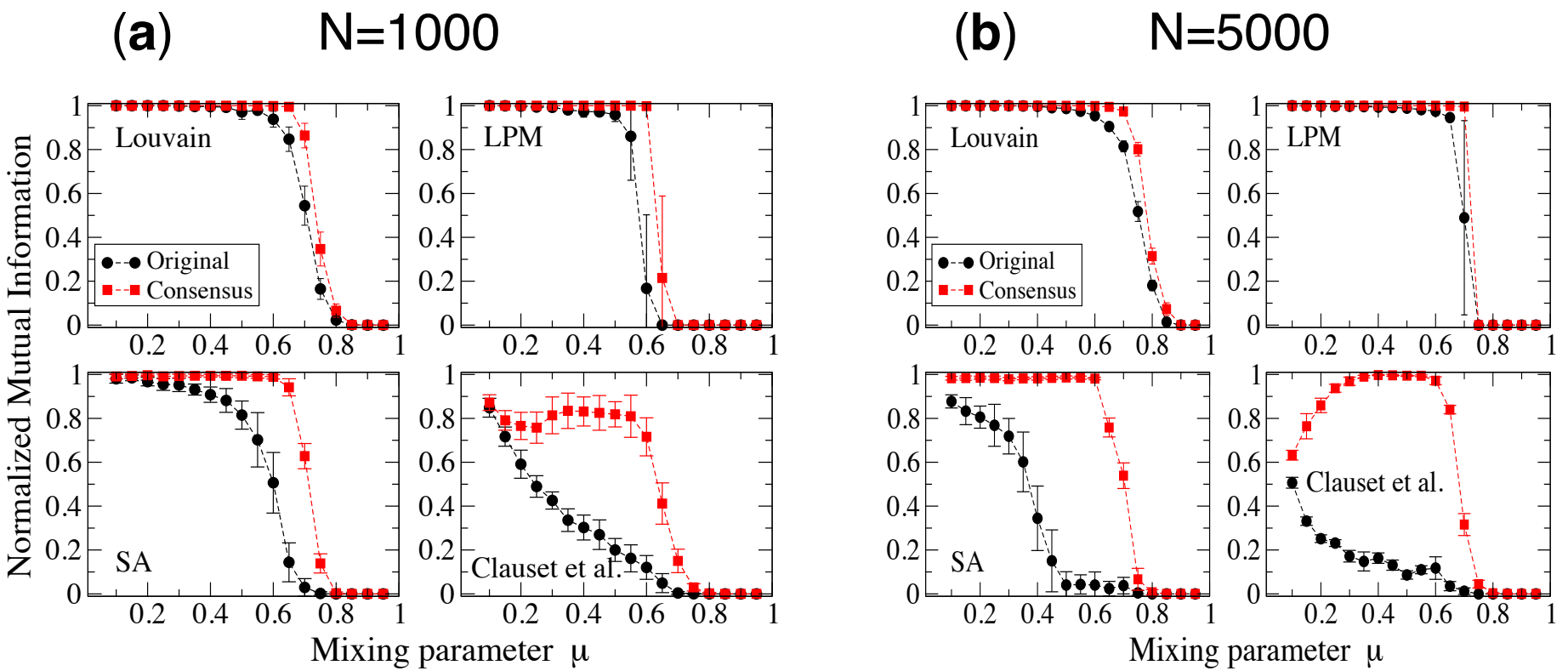
Consensus clustering

Setting the threshold



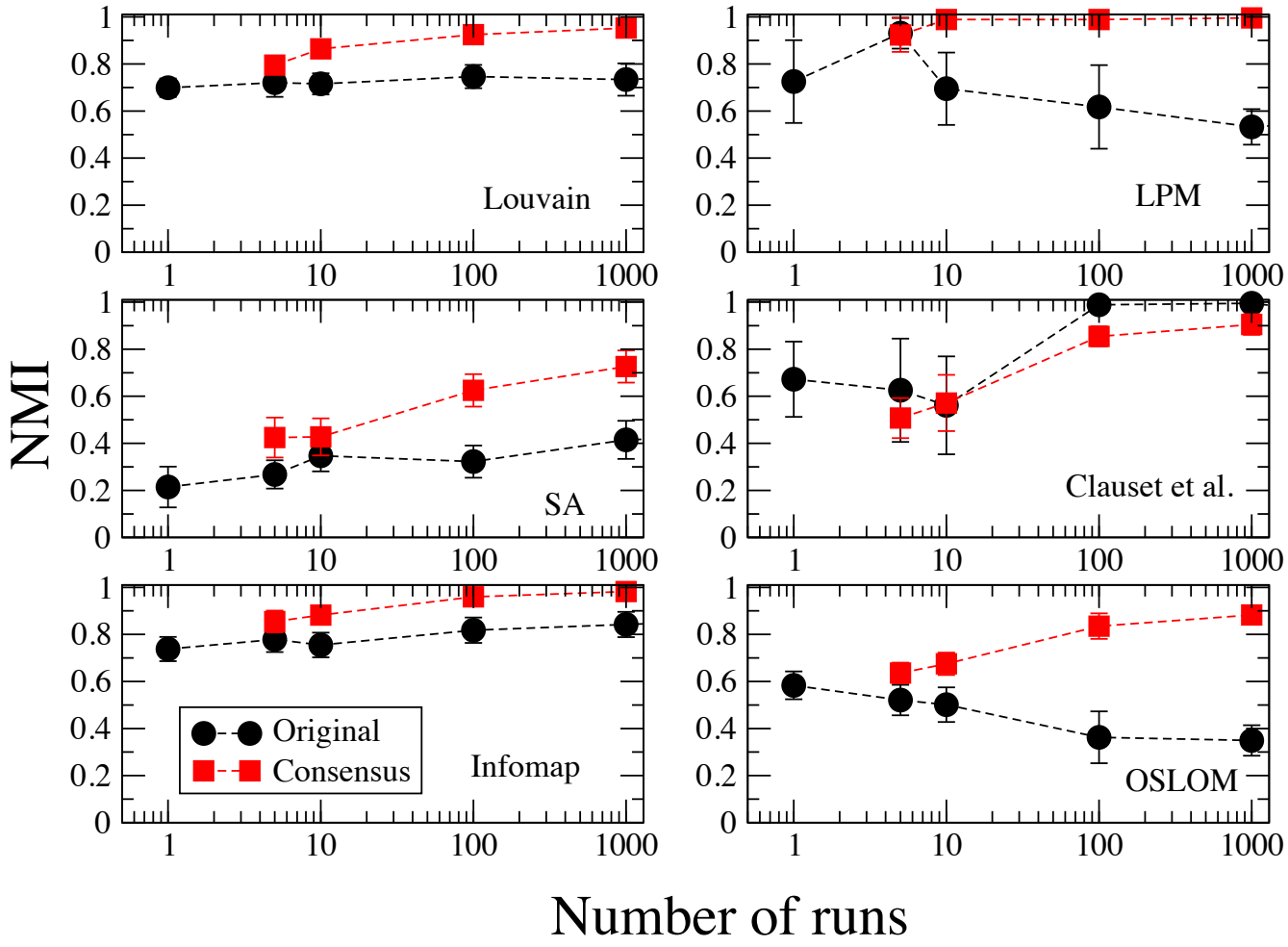
Consensus clustering

Results on the LFR benchmarks



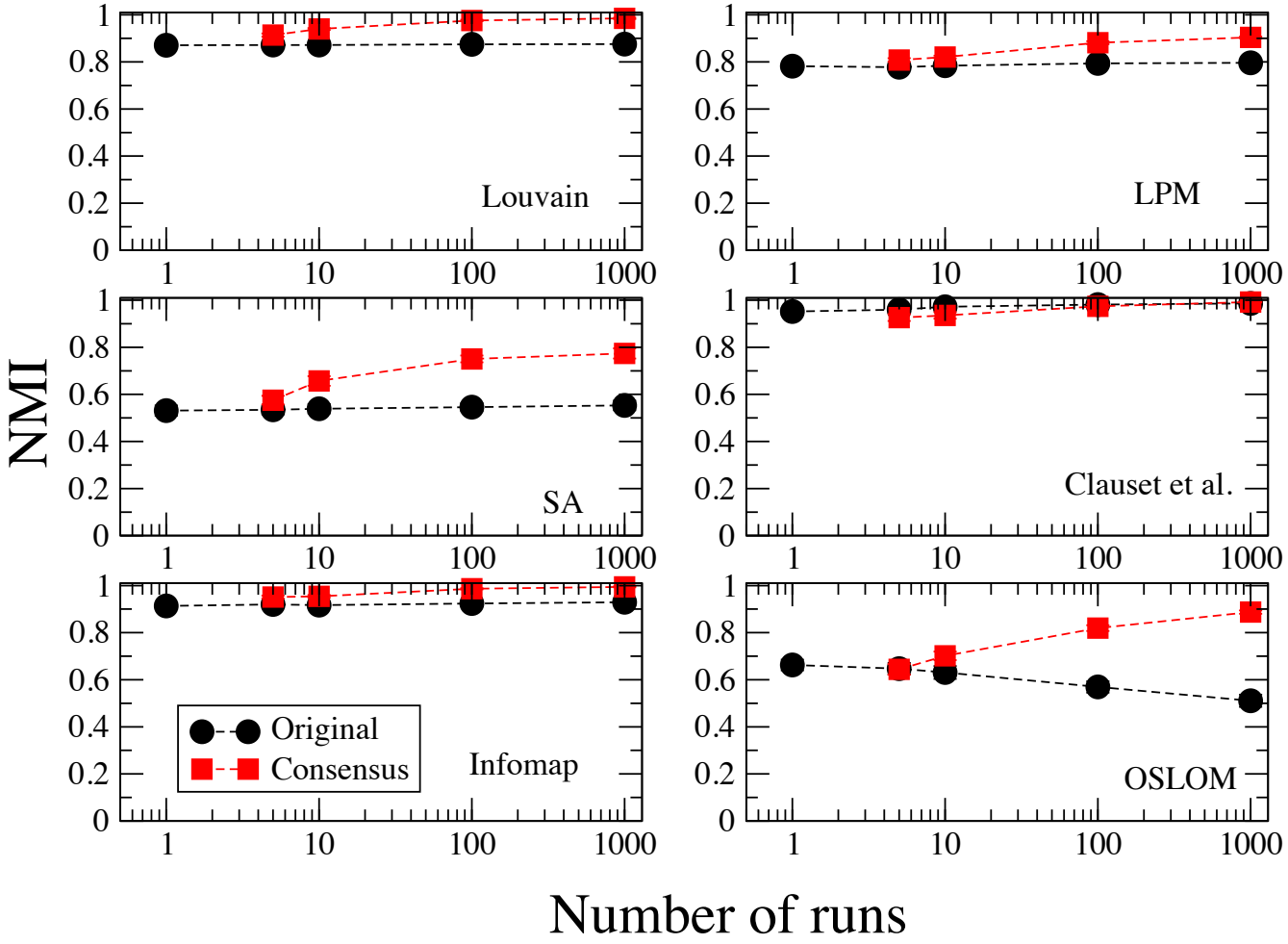
Consensus clustering

Stability: C. elegans



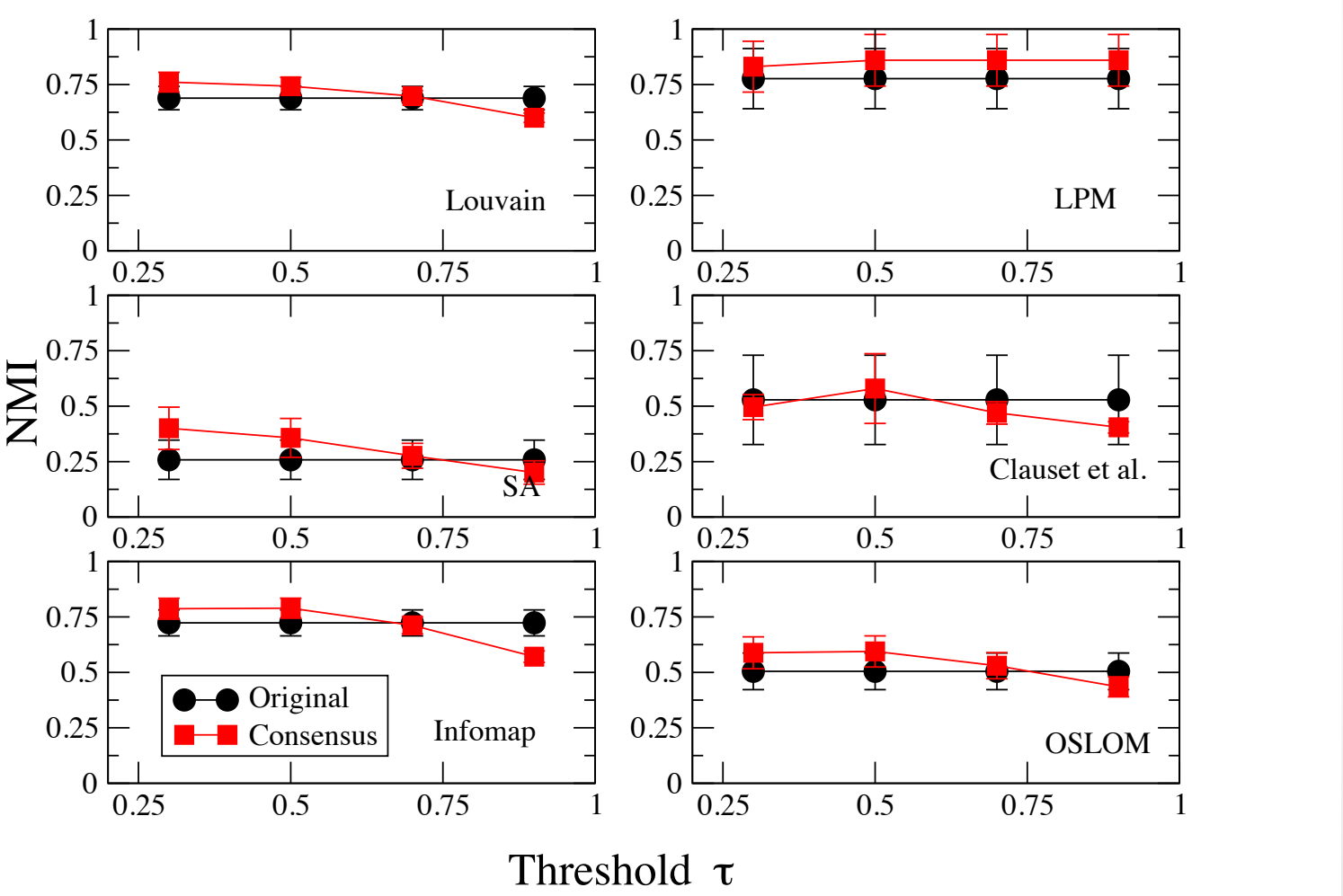
Consensus clustering

Stability: APS citation network



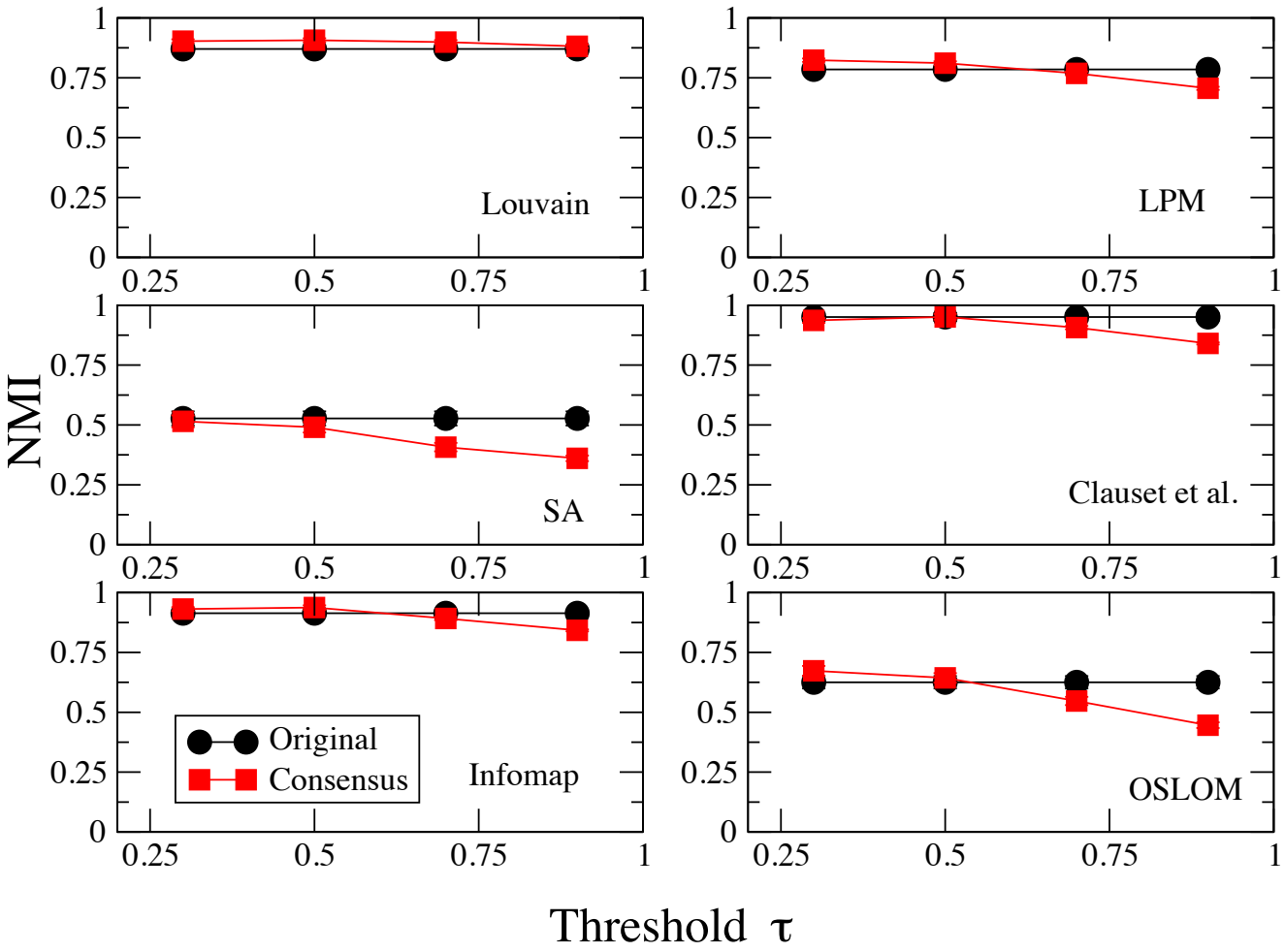
Consensus clustering

Fidelity: C. elegans



Consensus clustering

Fidelity: APS citation network



Consensus clustering

Dynamic networks

- Succession of snapshots, corresponding to m overlapping time windows of size Δt: $[t_0, t_0+\Delta t]$, $[t_0+1, t_0+1+\Delta t]$, $[t_m-\Delta t, t_m]$
- Consensus partition from subsets of r consecutive snapshots, with r suitably chosen.
- One starts by combining the first r snapshots, then those from 2 to r+1, and so on until the interval spanned by the last r snapshots.
- Our calculations: Δt=5 (years), r = 2.
- D_{ij} = number of times vertices i and j are clustered together, divided by the number of partitions corresponding to snapshots including both vertices

Consensus clustering

Dynamic networks

$$\mathcal{C}_t \longrightarrow \mathcal{C}_{t+1} \ ?$$

Problem: A cluster may fragment, and thus there would be many "children" clusters at time t + 1 for the same cluster at time t.

Our strategy: computing the Jaccard index of C_t with all clusters of partition at time t + 1, and pick the cluster with the highest value

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Consensus clustering

Dynamic networks

In the same way we find the "father" of cluster C_{t+1}

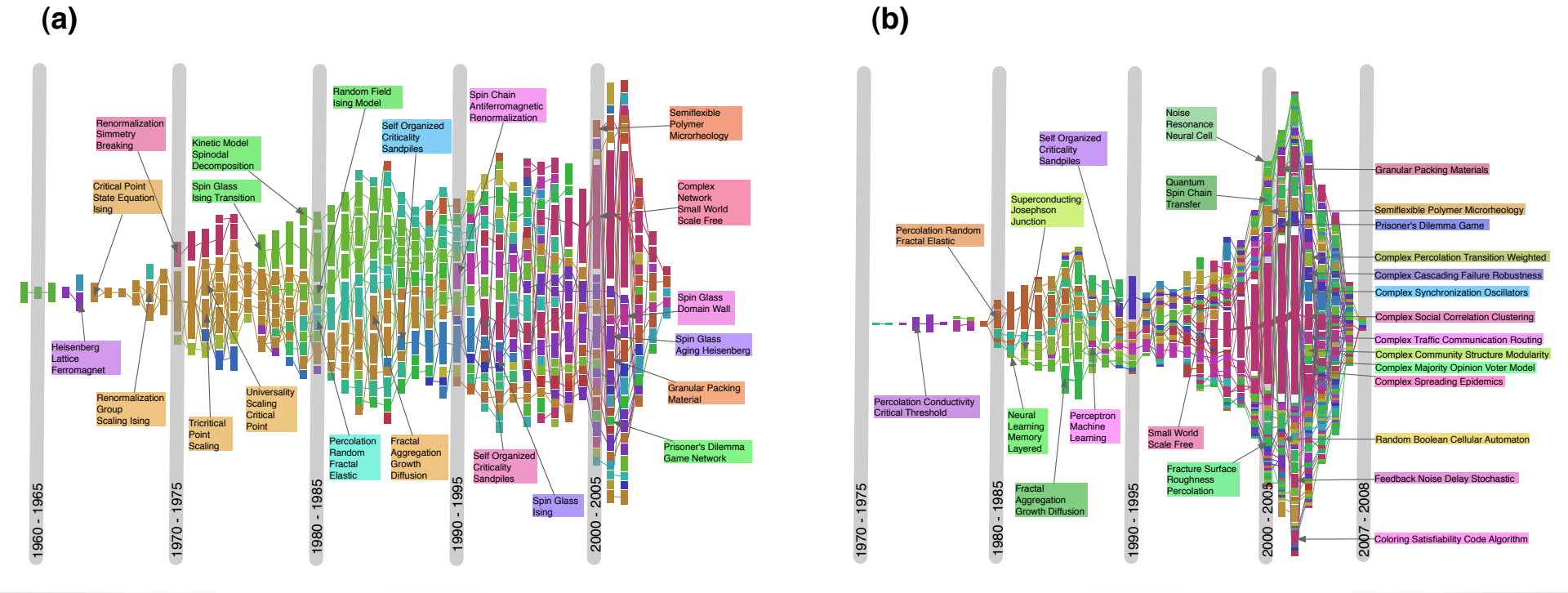
Important: if cluster A at time t is the best match of cluster B at time t + 1, the latter may not be the best match of A

Criterion:

- if clusters A and B are each other's best match, A "survives" to time t +1
- If clusters A and B are not each other's best match, A "dies" at time t and B is considered as a new cluster

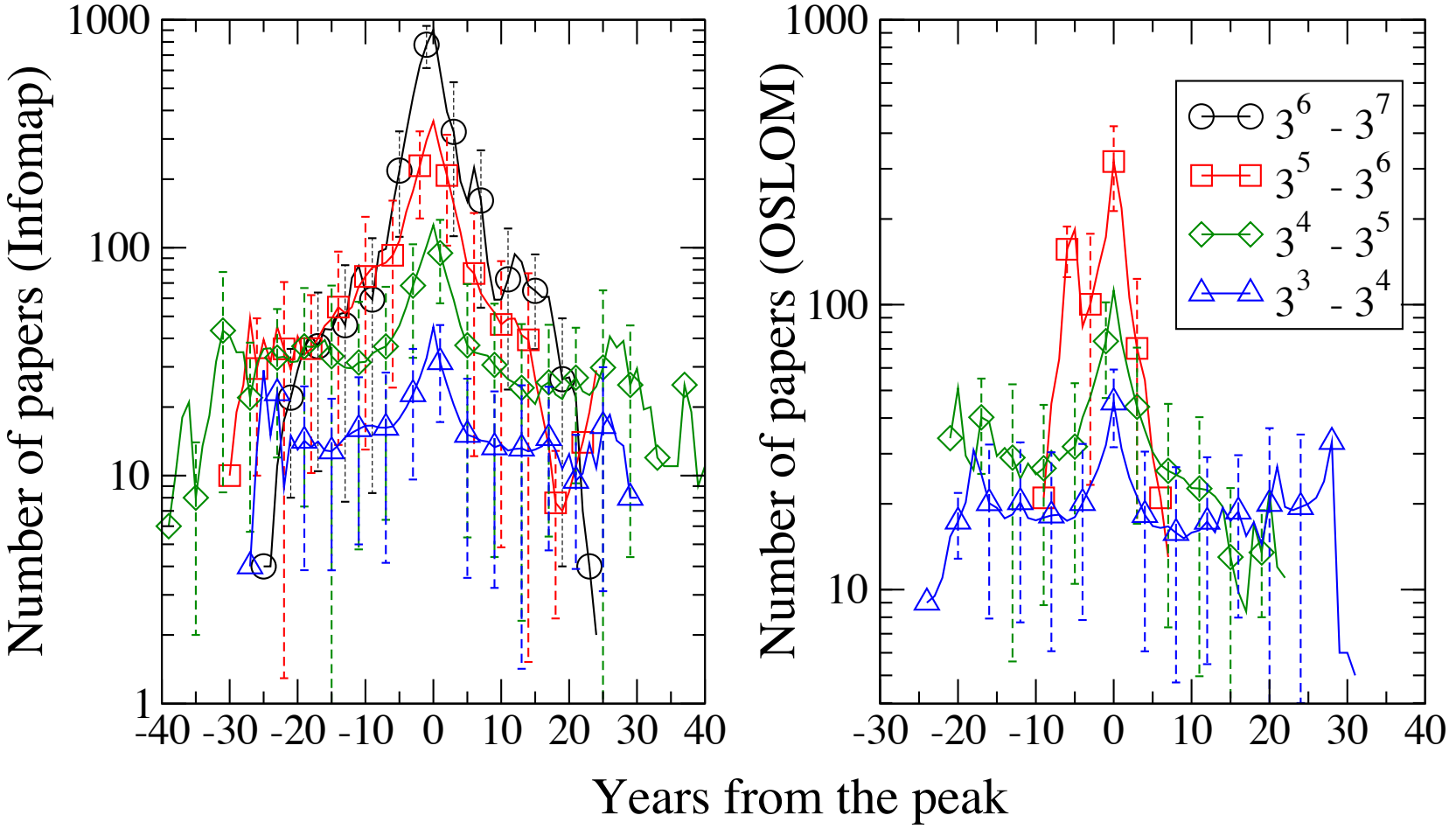
Consensus clustering

Tracking dynamic clusters: the APS citation network



Consensus clustering

Tracking dynamic clusters: the APS citation network



Summary

1) What is a community? **No unique answer**! **Definition is system- and problem-dependent**
2) Magic method? **No such thing**! **Domain dependent methods?**
3) **Triadic closure naturally yields community structure**
4) **Global optimization** methods have important limits: **local optimization** looks more natural and promising
5) Attention on **validation**

S. F., Phys. Rep. 486, 75-174 (2010)

Top 25 Hottest Articles
Physics and Astronomy

Physics Reports 486 (2010) 75–174

Contents lists available at ScienceDirect

Physics Reports

journal homepage: www.elsevier.com/locate/physrep

Community detection in graphs

Santo Fortunato *

Complex Networks and Systems Lagrange Laboratory, ISI Foundation, Viale S. Severo 65, 10133, Torino, I, Italy

ARTICLE INFO

Article history:
Accepted 5 November 2009
Available online 4 December 2009
editor: I. Procaccia

ABSTRACT

The modern science of networks has brought significant advances to our understanding of complex systems. One of the most relevant features of graphs representing real systems is community structure, or clustering, i.e. the organization of vertices in clusters, with many edges joining vertices of the same cluster and comparatively few edges joining vertices of different clusters. Such clusters, or communities, can be considered as fairly

Most Cited Physics Reports Articles

The most cited articles published since 2008, extracted from SciVerse Scopus.

Community detection in graphs

Volume 486, Issues 3-5, February 2010, Pages 75-174
Fortunato, S.

The modern science of networks has brought significant advances to our understanding of complex systems. One of the most relevant features of graphs representing real systems is community structure, or clustering, i.e. the organization of vertices in clusters, with many edges joining vertices of the same cluster and comparatively few edges joining vertices of different clusters. Such clusters, or communities, can be considered as fairly independent compartments of a graph, playing a similar role like, e.g., the tissues or the organs in the human body. Detecting communities is of great importance in sociology, biology and computer science, disciplines where systems are often represented as graphs. This problem is very hard and not yet satisfactorily solved, despite the huge effort of a large interdisciplinary community of scientists working on it over the past few years. We will attempt a thorough exposition of the topic, from the definition of the main elements of the problem, to the presentation of most methods developed, with a special focus on techniques designed by statistical physicists, from the discussion of crucial issues like the significance of clustering and how methods should be tested and compared against each other, to the description of applications to real networks.