Lecture V
Introduction to complex networks

Santo Fortunato

INSTITUTE
FOR SCIENTIFIC INTERCHANGE
FOUNDATION

Plan of the course

I. Networks: definitions, characteristics, basic concepts in graph theory
II. Real world networks: basic properties
III. Models
IV. Community structure I
V. **Community structure II**
VI. Dynamic processes in networks

Problems of traditional methods

- Graph partitioning, partitional clustering and spectral clustering: one needs to specify the number and the size of the clusters
- Hierarchical clustering: many partitions recovered, which one is the best?

One would like methods that can predict the number and the size of the partition and indicate a subset of "good" partitions

Girvan-Newman algorithm

M. Girvan & M.E.J Newman, PNAS 99, 7821-7826 (2002)

Divisive method: one removes the links that connect the clusters, until the latter are isolated

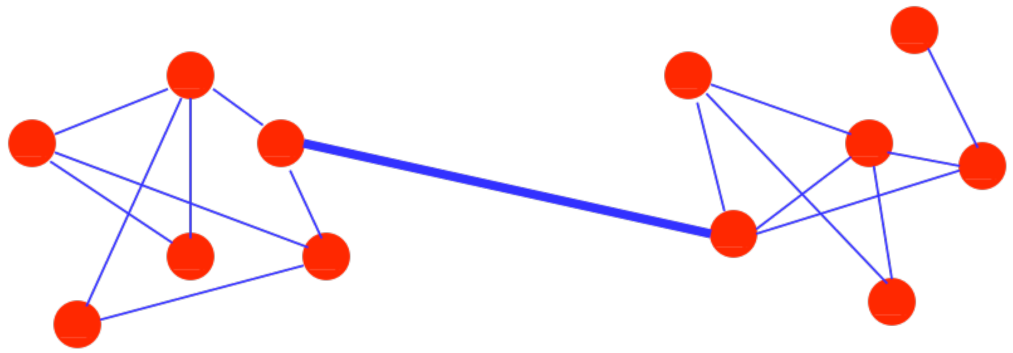
How to identify intercommunity links? Betweenness

Girvan-Newman algorithm

Edge betweenness: total number of shortest paths from any pair of vertices that cross the edge (Anthonisse, 1971)

If there are more geodesic paths between the same pair of vertices crossing the edge one divides the contribution of each path by their multiplicity

Computable with algorithms based on breadth-first-search, with complexity $O(mn)$ (Brandes, 2001)



Girvan-Newman algorithm

Steps

1. Calculate the betweenness of all edges
2. Remove the one with highest betweenness
3. Recalculate the betweenness of the remaining edges
4. Repeat from 2

The complexity of the algorithm is $O(m^2 n)$ [$O(n^3)$ on a sparse graph]

Recalculation step costly but important!

Girvan-Newman algorithm

The process delivers a hierarchy of partitions: which one is the best?

The best partition is the one corresponding to the highest modularity Q (Newman & Girvan, 2004)

The complexity may be lowered if one computes the edge betweenness of the edges only from randomly chosen pairs of vertices, instead of picking all of them (Tyler et al., 2003)

Variants of the method, where vertices can be split among clusters, enable to detect overlapping communities (Pinney & Westhead, 2006; Gregory, 2007)

New methods

- Divisive algorithms
- Modularity optimization
- Spectral methods
- Dynamic methods
- Methods based on statistical inference
- Methods based on model selection

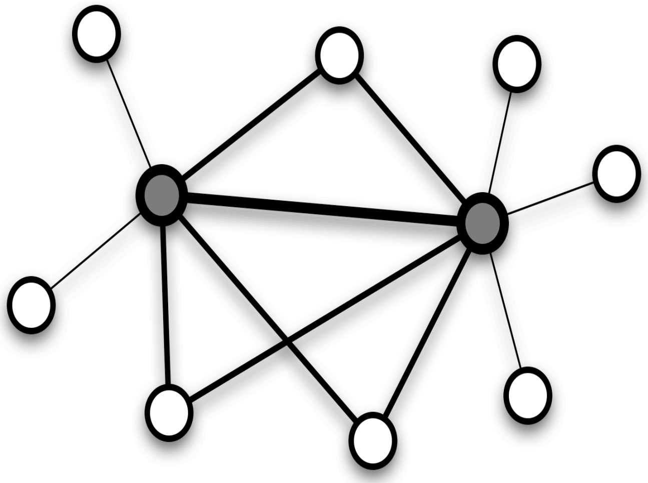
Divisive algorithms

Based on edge removal (like GN)

Ex. Algorithm by Radicchi et al., 2004

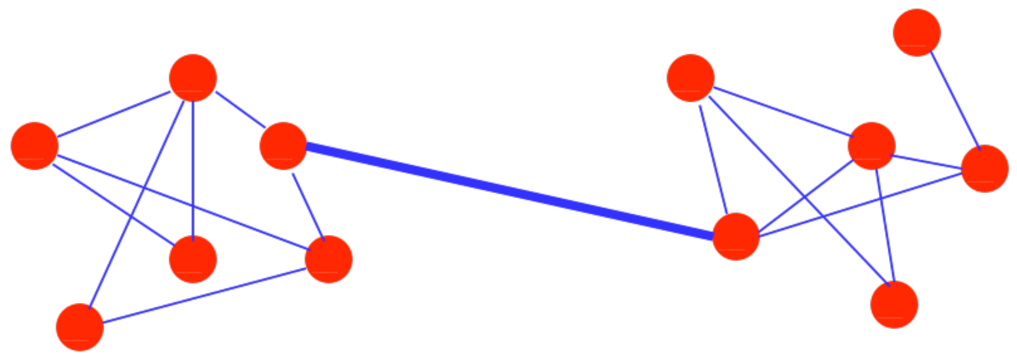
Edge clustering coefficient

$$\tilde{C}_{i,j}^{(g)} = \frac{z_{i,j}^{(g)} + 1}{s_{i,j}^{(g)}}$$



Divisive algorithms

Main idea: inter-community edges have low edge clustering coefficient



Steps

1. Calculate the edge clustering of all edges
2. Remove the one with lowest edge clustering
3. Recalculate the measure for the remaining edges
4. Repeat from 2 as long as clusters of partition are either LS-sets (“strong”) or “weak” communities

Divisive algorithms

Advantage over GN: fast!

The complexity is $O(m^4/n^2)$ [$O(n^2)$ on a sparse graph] if cycles are short (e.g. triangles)

For long cycles the edge clustering coefficient becomes a global measure and the complexity increases fast

Problem: the method gives poor results on graphs with few cycles

Modularity optimization

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

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

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

Fast version of Newman's method by Clauset et al. (Phys. Rev. E 70, 066111, 2004)

Use of special data structures (max-heaps) to speed up update of sparse matrices during computation

Complexity: $O(md\log n)$

d = depth of dendrogram

For graphs with strong hierarchical structure: $d \sim \log n$

Best complexity for sparse graphs: $O(n\log^2 n)$

Greedy algorithms

Problems of Newman's method:

1) Large communities are formed at the beginning at the expenses of the small ones, so partition is quite unbalanced!
2) Because of 1, the method often runs at its worst-case complexity, so the computation may be rather slow

Several modifications proposed:

1) Merging pairs of communities of similar size (Wakita & Tsurumi, 2007). Graphs with up to 10^7 vertices can be analyzed
2) Allowing for merging of more than one community pair at each iteration (Schuetz & Caflisch, 2008)
3) Starting the agglomeration from some reasonable intermediate partition (Pujol et al., 2006; Du et al., 2006; Xiang et al., 2009; Mei et al., 2009)

Refinement steps à la Kernighan-Lin improve final modularity

Greedy algorithms

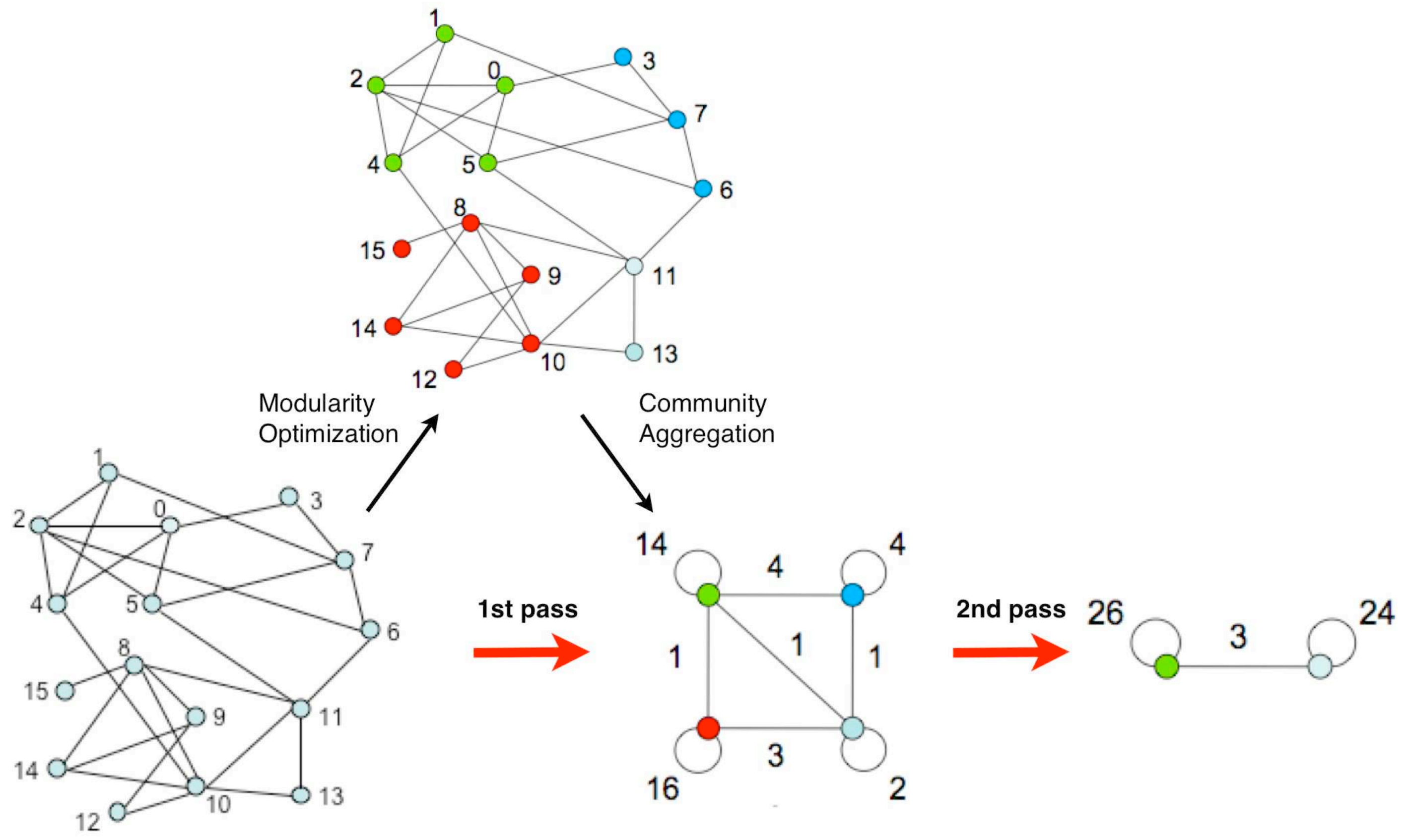
Louvain method (Blondel et al., JSTAT P10008, 2008)

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

Simulated annealing

Guimerà et al., 2004

Steps:

1) Start from a random partition
2) Local moves: single vertices are shifted from one cluster to another
3) Global moves: clusters are split and merged
4) If modularity increases, move is accepted
5) If modularity decreases, move is accepted with probability exp(βΔQ)

$$\exp(\beta \Delta Q)$$

Complexity: parameter-dependent, very high! Only graphs with up to about 10^4 vertices can be studied

Extremal optimization

Duch & Arenas, 2005

Crucial ingredient: modularity can be written as a sum over the vertices ➔ the fitness of a vertex is the ratio between the local modularity of the vertex and its degree

Steps:

1) Start from random partition in two clusters
2) The vertex with the lowest fitness is shifted to the other community
3) Fitness values are recalculated (as partition has changed)
4) Process stops when modularity cannot be improved any more
5) Process is repeated for each of the clusters, taken as an isolated graph

Complexity: $O(n^2 \log n)$

Spectral optimization

Newman, 2006

Modularity matrix: $B_{ij} = A_{ij} - \frac{k_i k_j}{2m}$

$$\begin{aligned} Q &= \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(C_i, C_j) \\ &= \frac{1}{4m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)(s_i s_j + 1) \\ &= \frac{1}{4m}\sum_{ij} B_{ij} s_i s_j = \frac{1}{4m}\mathbf{s}^T\mathbf{B}\mathbf{s} \end{aligned}$$

Graph partitioning?

Spectral optimization

Index vector: $\mathbf{s} \rightarrow s_i = \pm 1$

$$\mathbf{s} = \sum_i a_i \mathbf{u}_i, \quad \mathbf{u}_i$$

Eigenvectors of modularity matrix

$$a_i = \mathbf{u}_i^T \cdot \mathbf{s}$$

$$Q = \frac{1}{4m} \sum_i a_i \mathbf{u}_i^T \mathbf{B} \sum_j a_j \mathbf{u}_j = \frac{1}{4m} \sum_{i=1}^{n} (\mathbf{u}_i^T \cdot \mathbf{s})^2 \beta_i$$

Spectral optimization

$$Q = \frac{1}{4m}\sum_{i=1}^{n}(\mathbf{u}_i^T \cdot \mathbf{s})^2 \beta_i$$

Features of modularity matrix:

1) Eigenvalue 0 with eigenvector (1,1,1,1…,1)
2) Other eigenvalues both positive and negative
3) If eigenvalues are non-positive, the "best" partition corresponds to null modularity and to all vertices in the same cluster
4) If there are positive eigenvalues, maximum modularity is positive
5) If there are negative eigenvalues, minimum modularity is negative (bipartite structure!)

Spectral optimization

$$Q = \frac{1}{4m} \sum_{i=1}^{n} (\mathbf{u}_i^T \cdot \mathbf{s})^2 \beta_i$$

Maximum modularity: index vector parallel to eigenvector corresponding to largest eigenvalue of modularity matrix

Problem: modularity eigenvector components are not integer!

Practical recipe: separate vertices whose components of the largest modularity eigenvector are positive from the vertices with negative components

Advantage over graph partitioning: one needs not specify cluster sizes!

Spectral optimization

Refinement steps: shifting vertices from one cluster to the other, to have the highest increase (smallest decrease) of modularity

To find the modularity maximum over all partitions, one proceeds with iterative bisectioning, as long as modularity increases at each bisection

Complexity: $O(n^2 \log n)$

More modularity eigenvectors (with positive eigenvalues) can be used
➔ spectral clustering

Problem: iterative bisectioning not good!

Modifications of modularity

Weighted modularity (Newman, 2004)

$$Q_w = \frac{1}{2W} \sum_{ij} \left(W_{ij} - \frac{s_i s_j}{2W} \right) \delta(C_i, C_j)$$

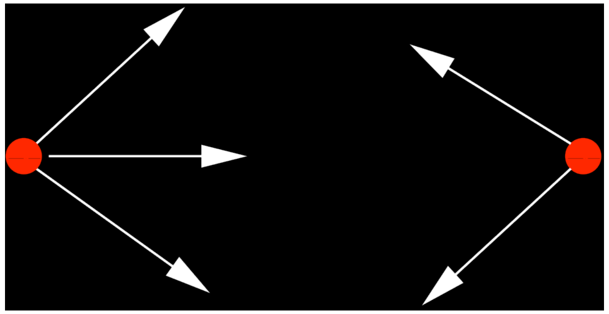
$$Q_w = \sum_{c=1}^{n_c} \left[\frac{W_c}{W} - \left(\frac{S_c}{2W} \right)^2 \right]$$

Null model term can be understood if one considers multiple edges between vertices!

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

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

Modifications of modularity

Modularity for covers? No unique definition!

Ex. Definition by Shen et al. (2009)

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

$O_i =$ Number of communities vertex i belongs to

Modifications of modularity

Localized modularity (Muff et al., 2005)

Idea: in modularity's null model each vertex can be attached to any other, in real systems clusters are linked to a few others

$$LQ = \sum_{c=1}^{n_c} \left[\frac{l_c}{L_{c_n}} - \frac{d_c^{in} d_c^{out}}{L_{c_n}^2} \right]$$

$L_{c_n} =$ Number of edges in subgraph made by cluster c and its neighbors

Modifications of modularity

Spin glass formulation (Reichardt & Bornholdt, 2006)

Idea: edges should be between vertices of the same cluster, non-edges should be between vertices of different clusters

$$\mathcal{H}(\{\sigma\}) = -\sum_{i<j} J_{ij}\delta(\sigma_i,\sigma_j) = -\sum_{i<j} J(A_{ij} - \gamma p_{ij})\delta(\sigma_i,\sigma_j)$$

$J =$ (irrelevant) constant, $\gamma > 0,\ p_{ij} =$ null model term

Ex. $\gamma = 1,\ p_{ij} = k_i k_j / 2m$ ➔ modularity!

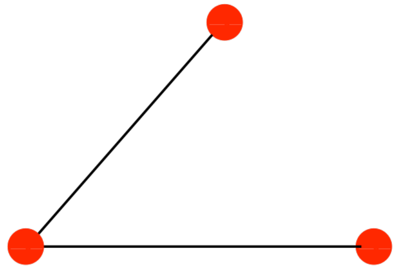
The parameter γ can be tuned and so the size of the clusters can be regulated (resolution parameter)

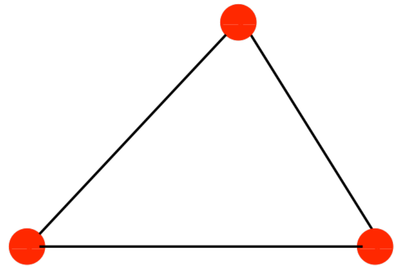
Modifications of modularity

Motif modularity (Arenas et al., 2008)

Idea: clusters do not only contain more edges than expected but also more motifs than expected

Motifs





$$Q_{\triangle}(\mathcal{C}) = \frac{\sum_{ijk} A_{ij}(\mathcal{C})A_{jk}(\mathcal{C})A_{ki}(\mathcal{C})}{\sum_{ijk} A_{ij}A_{jk}A_{ki}} - \frac{\sum_{ijk} n_{ij}(\mathcal{C})n_{jk}(\mathcal{C})n_{ki}(\mathcal{C})}{\sum_{ijk} n_{ij}n_{jk}n_{ki}}$$

Triangle modularity

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

Modularity in random graphs (Reichardt & Bornholdt, 2006; 2007)

Random graphs with arbitrary degree distribution P(k)

Largest modularity for an equipartition!

Relation between modularity and cut size

$$R_q = m[(q-1)/q - Q_q]$$

Bipartition:

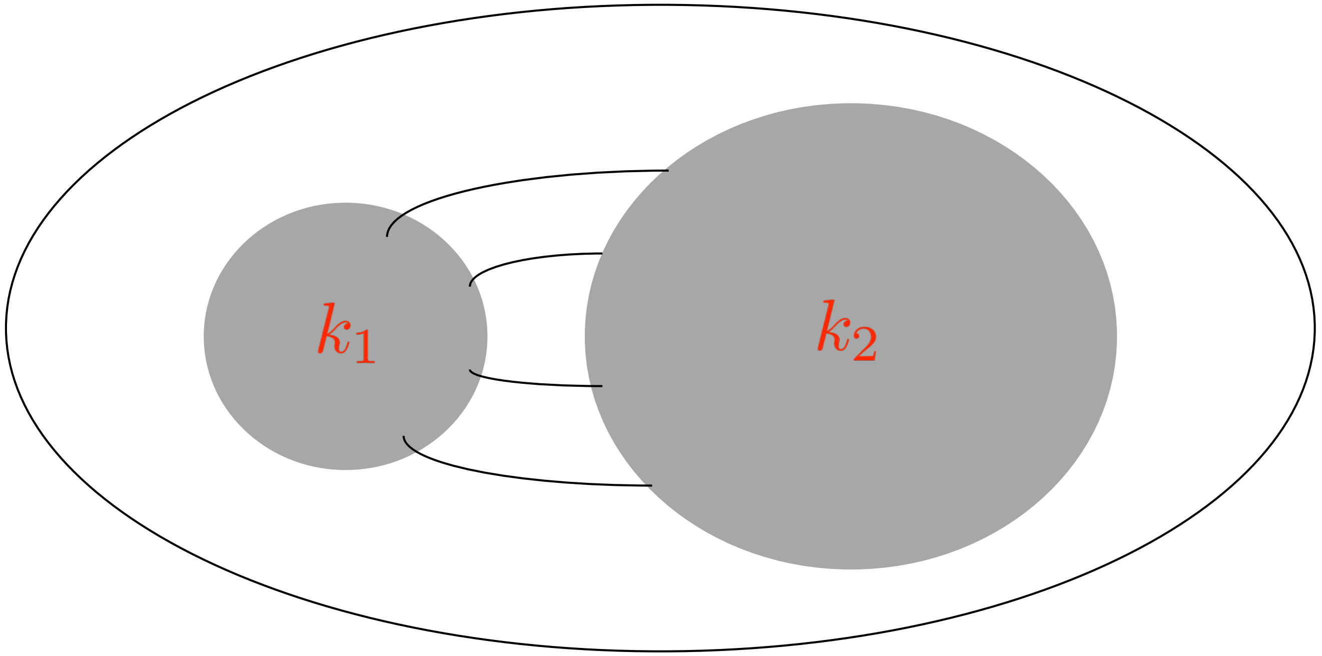
$$Q_2^{max} = U_0 J \frac{\langle k^{1/2} \rangle}{\langle k \rangle}$$

$$\langle k^\alpha \rangle = \int P(k) k^\alpha dk$$

Modularity maximum the larger, the sparser the graph: finding communities in sparse graphs is hard!

Limits of modularity

Resolution limit (Fortunato & Barthélemy, 2007)



Question: what is the expected number of edges between the two subgraphs in modularity's null model?

Answer:

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

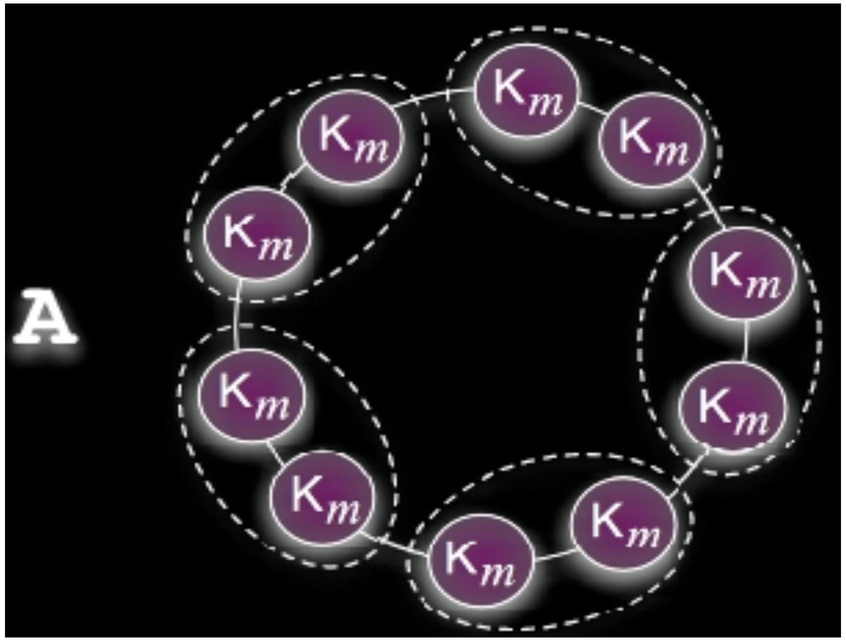
Limits of modularity

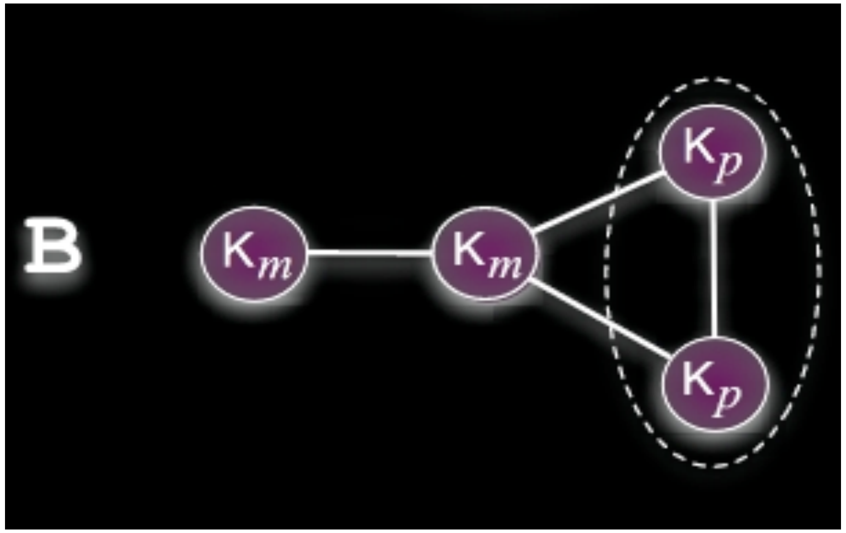
$\text{if } \frac{k_1 k_2}{2m} < 1 \rightarrow$

Modularity higher if the subgraphs are put in the same cluster

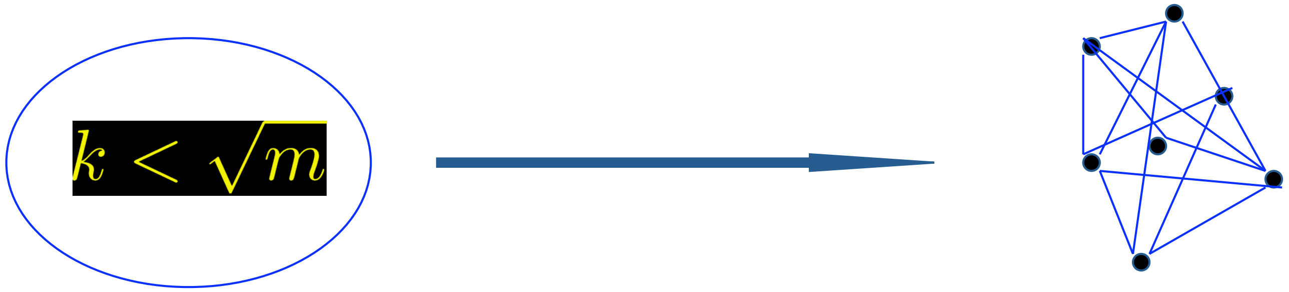
Ex. $k_1 \sim k_2 < \sqrt{2m}$

Resolution limit of modularity

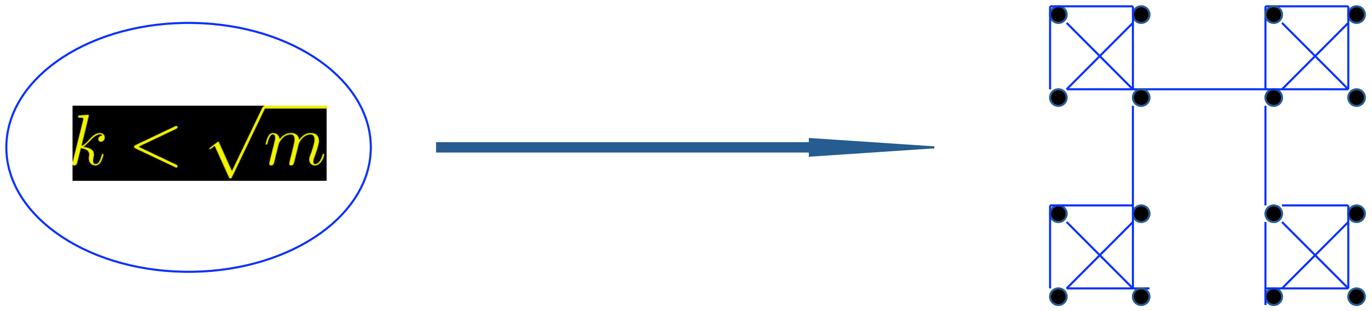




Limits of modularity



$$k < \sqrt{m}$$

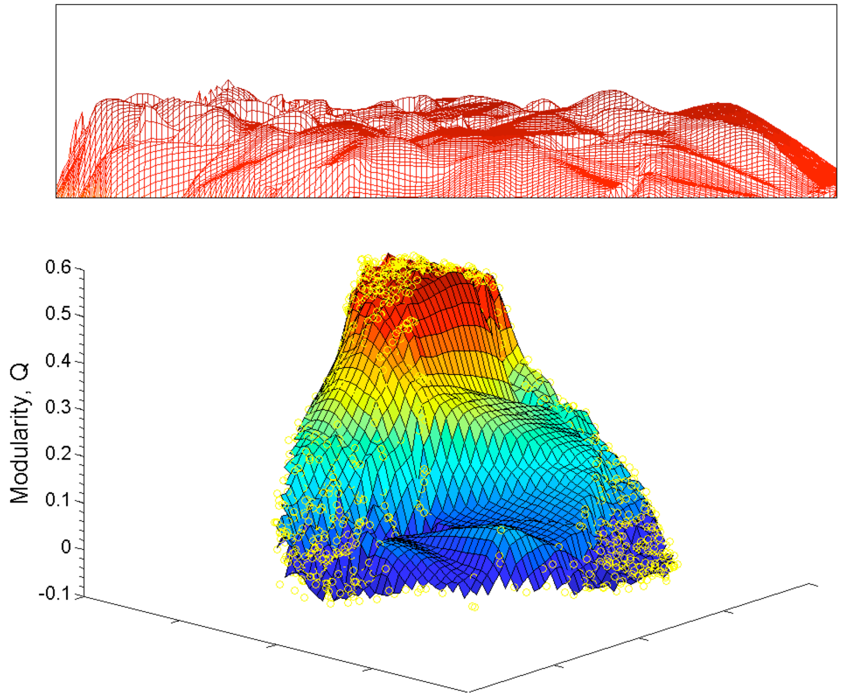


$$k < \sqrt{m}$$

Origin of resolution limit: modularity's null model is global, each vertex can in principle be attached to any other vertex of the graph (global information) ➔ 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

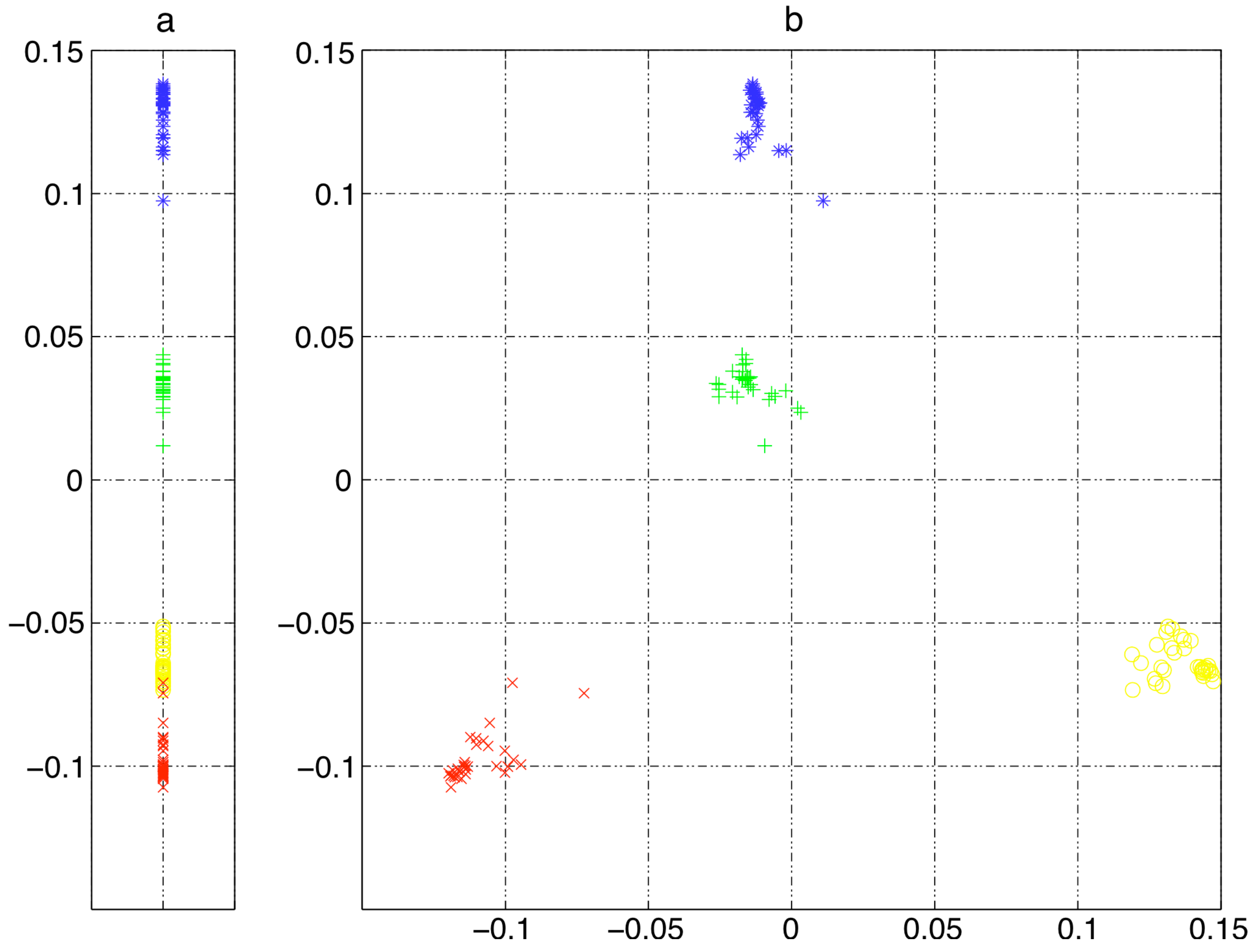
Finding communities from spectral properties of graph matrices: adjacency matrix, Laplacian matrix, etc.

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

$$W_{ij} \longrightarrow S_{ij}$$

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

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

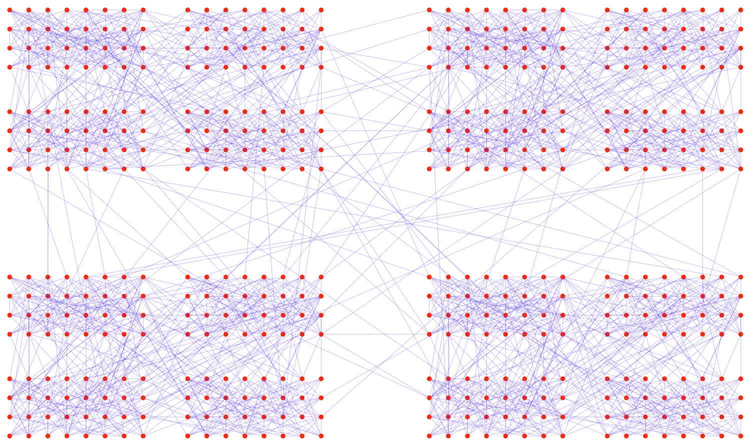
On a graph, coupling between neighboring oscillators

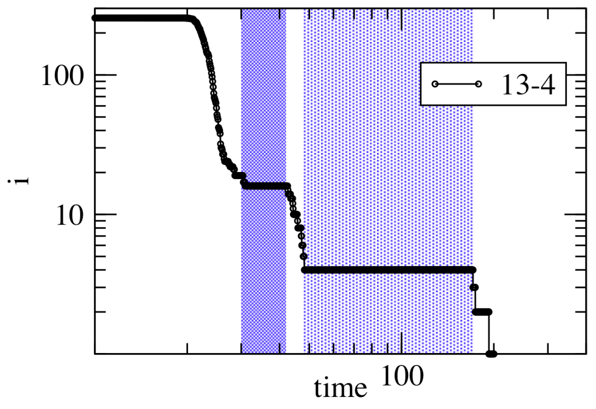
Steps:

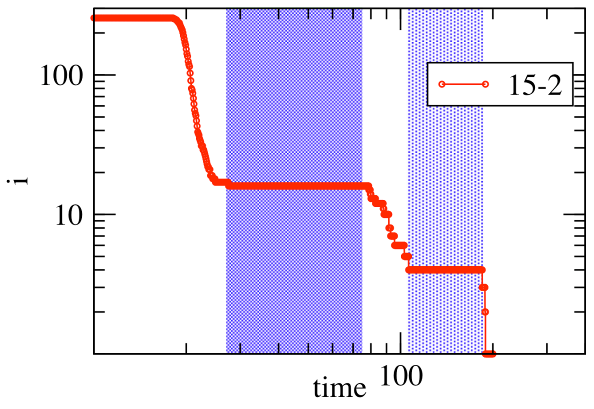
1) Initial configuration with random phases
2) For K larger than a threshold depending on the width of the distribution of the natural frequencies ω_i oscillators partially synchronize

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 $\{\pi_i\}$ and $\{\theta_{ri}\}$ fits 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

Methods based on model selection

Infomap (Rosvall & Bergstrom, 2008)

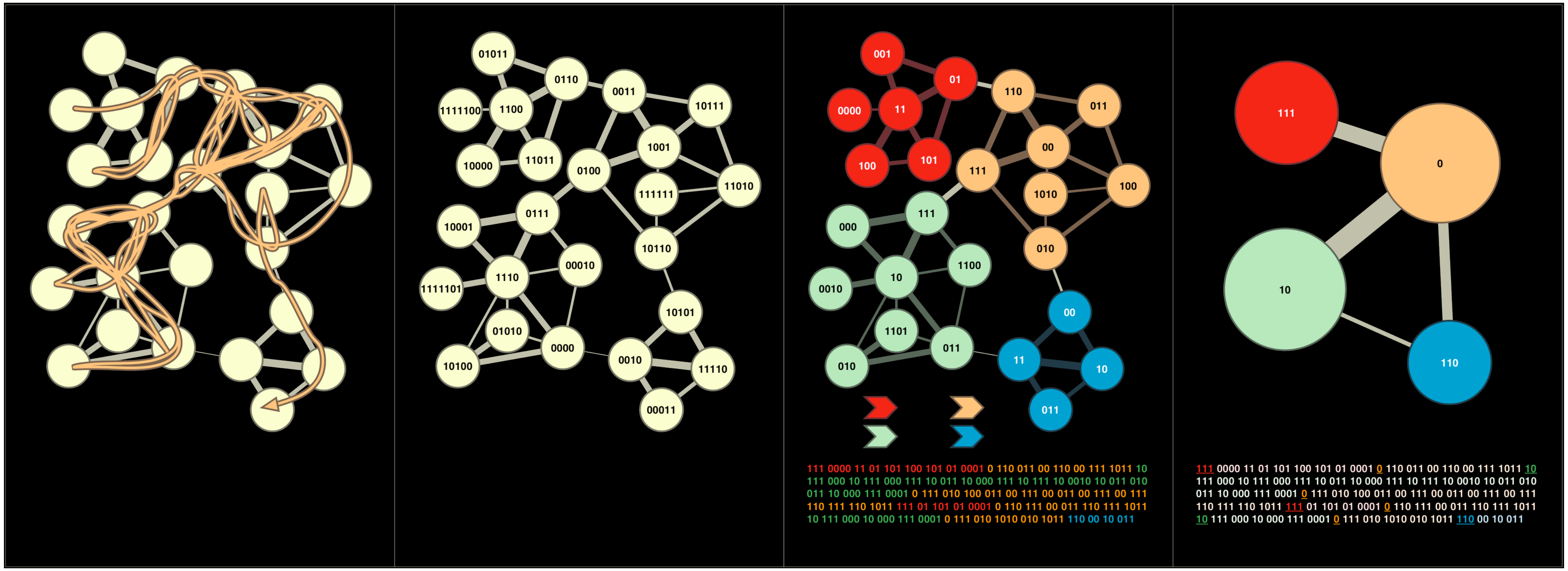
Idea: finding a compressed description of a random walk taking place on the graph

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

Methods based on model selection

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

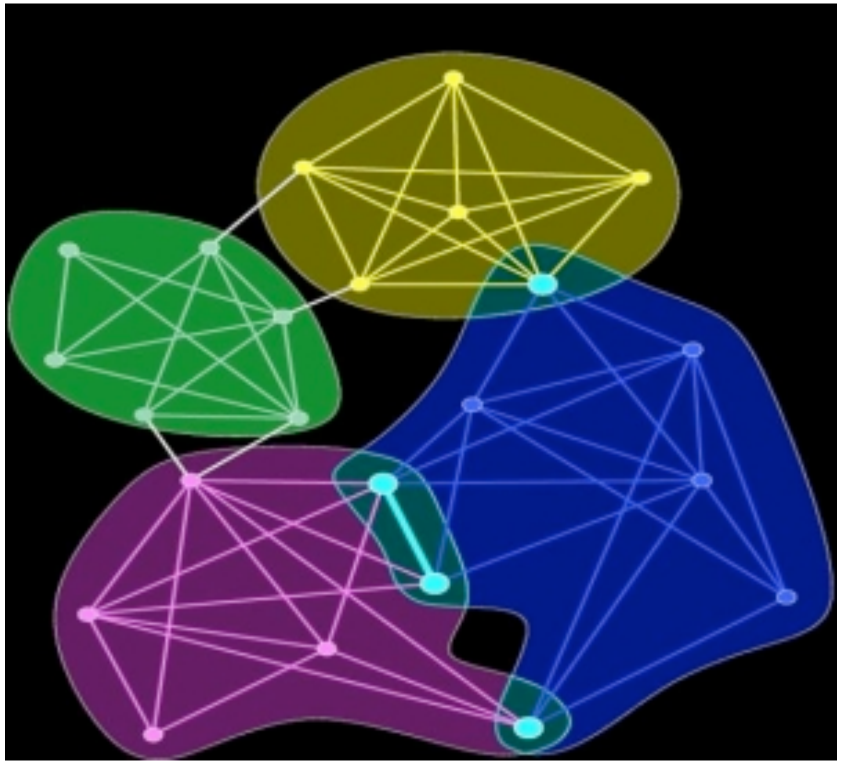
Cliques can be used as probes to explore the graph:

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 of a graph is an NP-complete problem, but on sparse graphs it can be done quickly (graphs with up to 10^5 vertices can be studied)

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

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

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

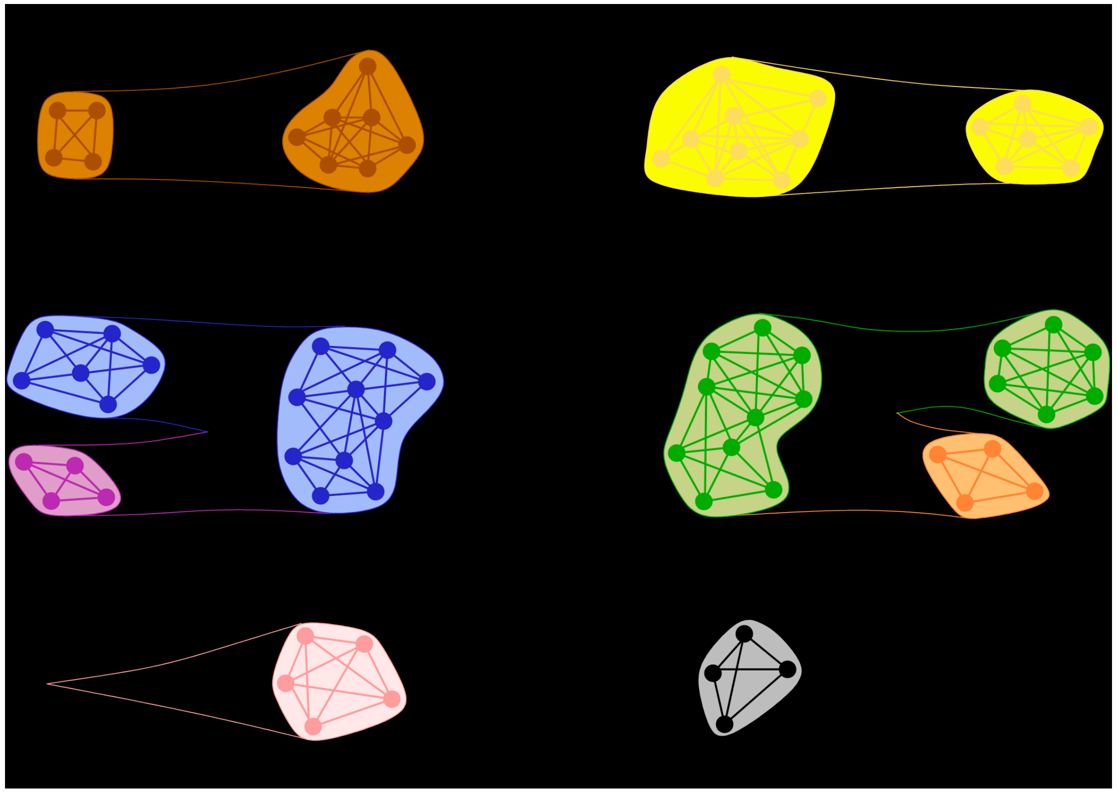
No null model!

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
1) 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 $\mathcal{C}(t)$

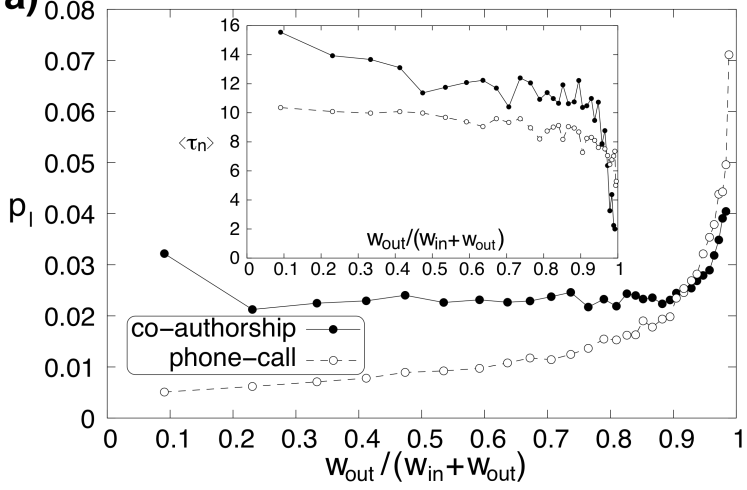
$\mathcal{G}(t,t+1)$ $\mathcal{C}(t)$

Detection of dynamic communities

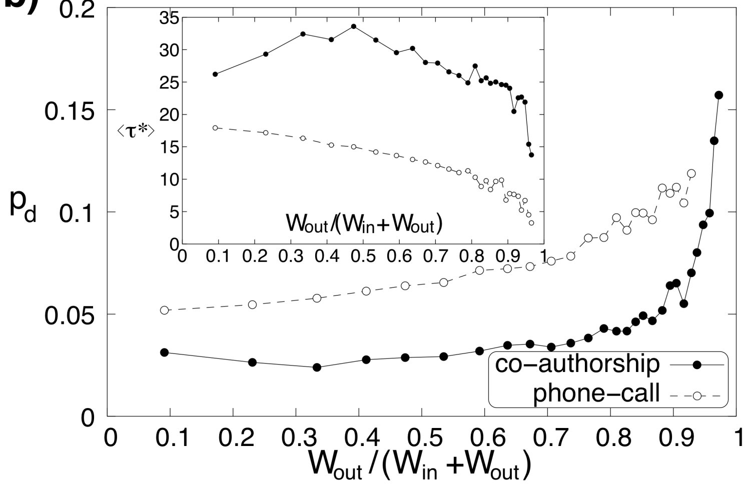
Results:

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

a)



b)



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)

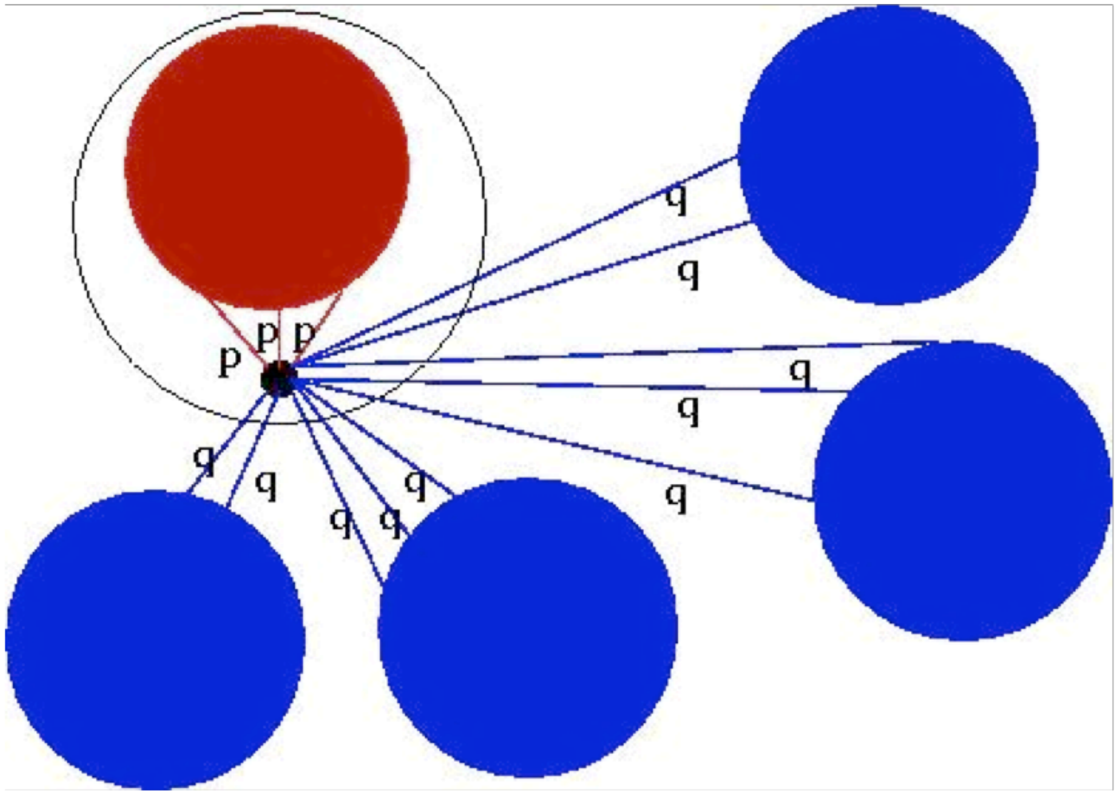
Ingredients:

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



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

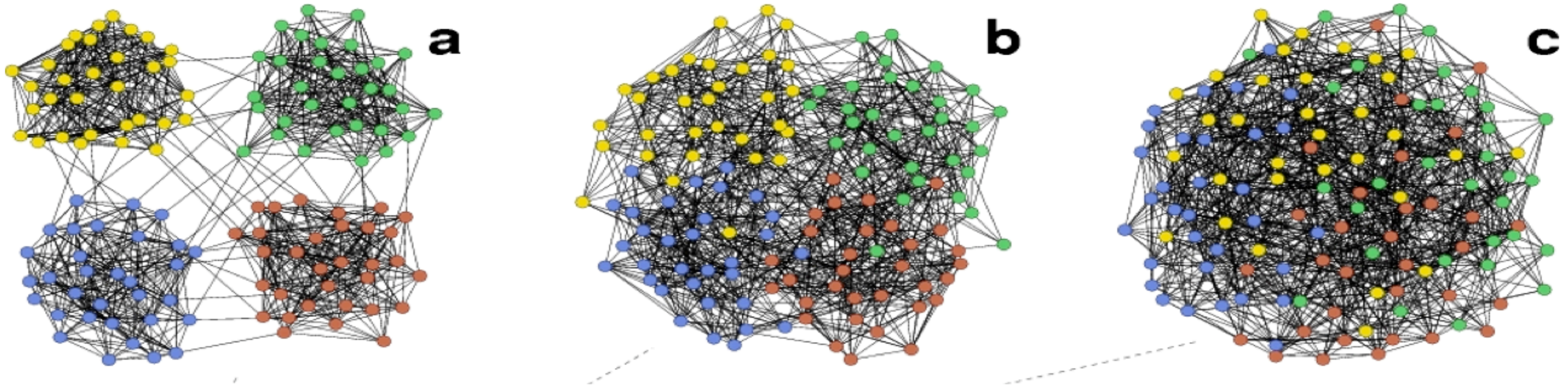
Testing algorithms

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:

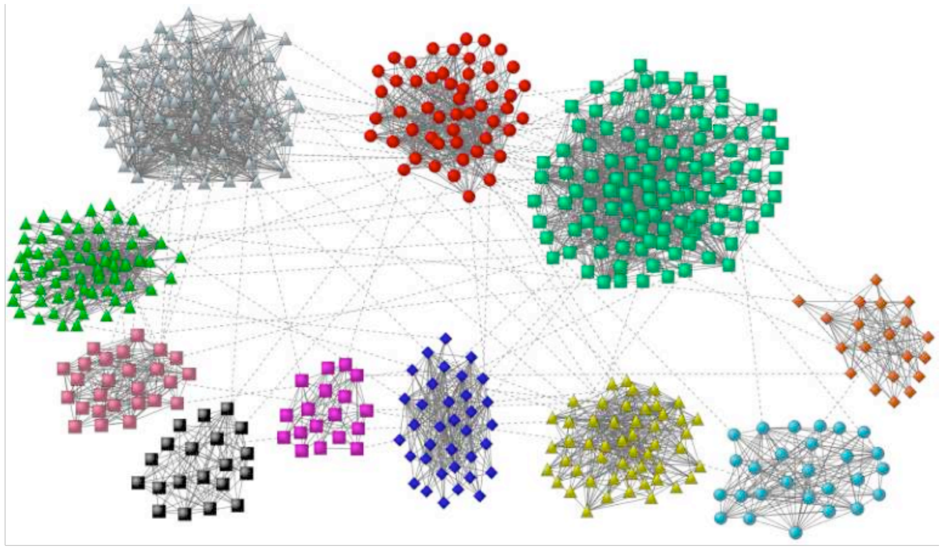
1) All vertices have the same degree
2) All communities have the same size

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



http://santo.fortunato.googlepages.com/inthepress2

Testing algorithms

Necessary ingredient: similarity measure between partitions

Ex. Normalized mutual information

x_i, y_i : community assignments

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

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

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

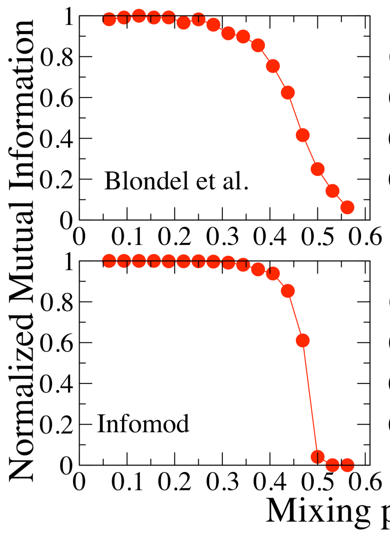
First analysis: Danon et al. (2005), using GN benchmark

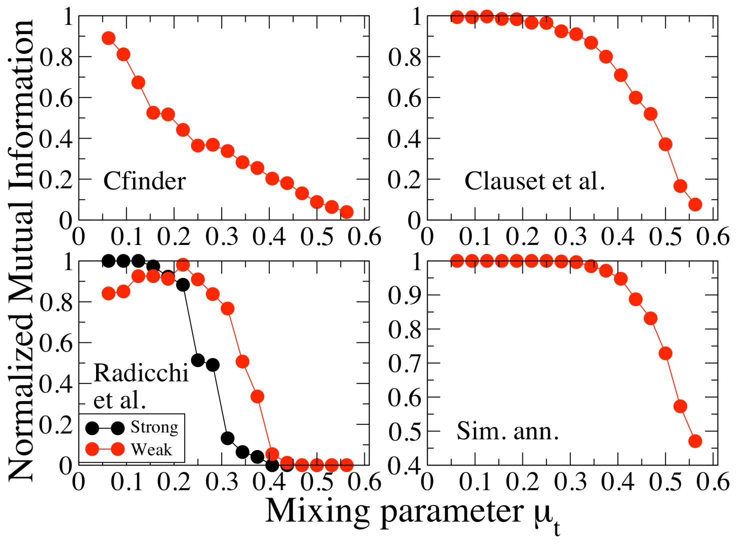
New analysis: Lancichinetti & Fortunato (2009), using LFR benchmark

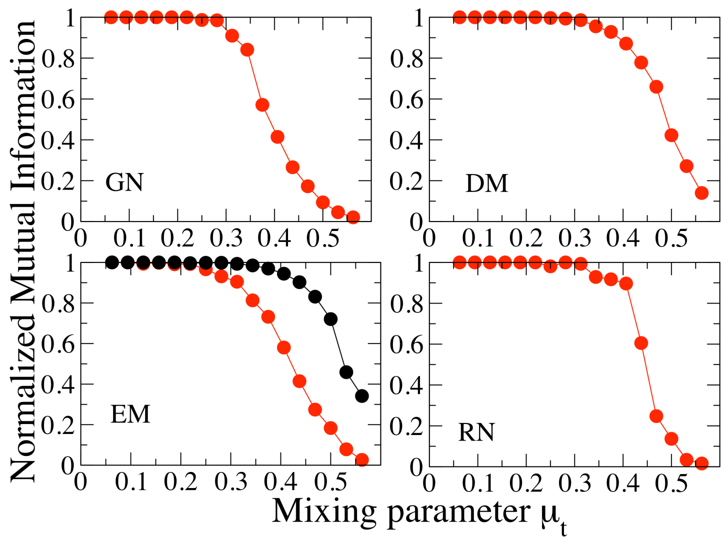
Author	Label	Order
Girvan & Newman	GN	$O(nm^2)$
Clauset et al.	Clauset et al.	$O(n \log^2 n)$
Blondel et al.	Blondel et al.	$O(m)$
Guimerà et al.	Sim. Ann.	parameter dependent
Radicchi et al.	Radicchi et al.	$O(m^4/n^2)$
Palla et al.	Cfinder	$O(\exp(n))$
Van Dongen	MCL	$O(nk^2)$, $k < n$ parameter
Rosvall & 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

GN benchmark

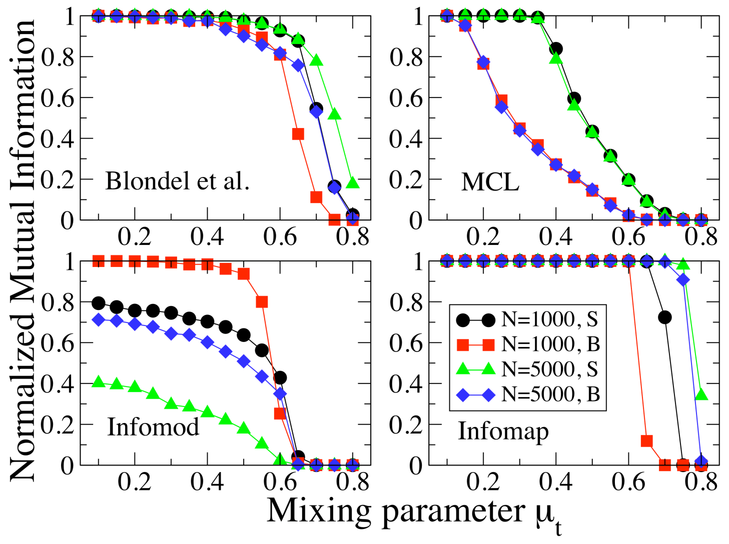


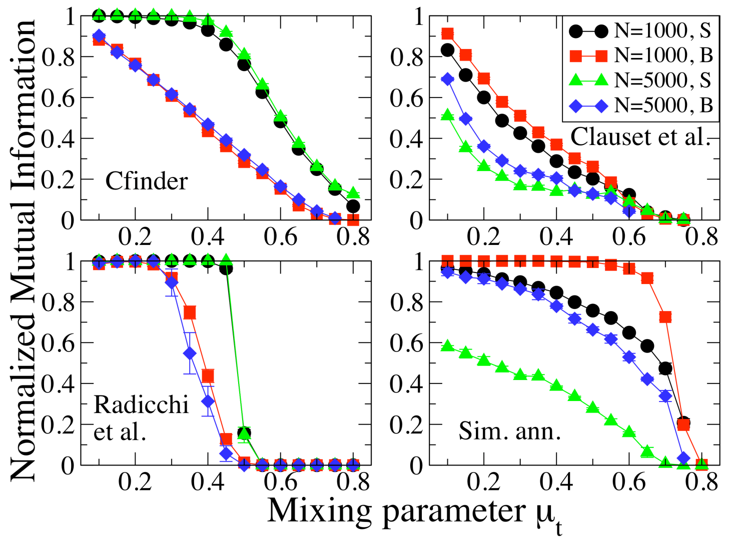


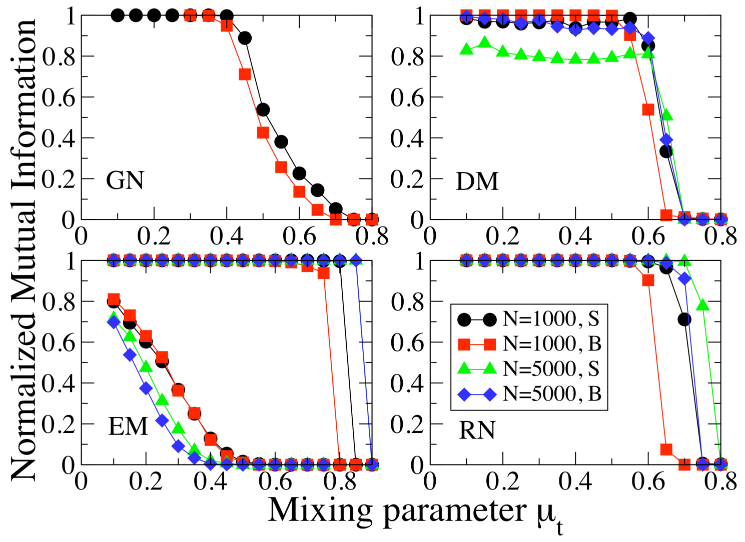


Testing algorithms

LFR benchmark

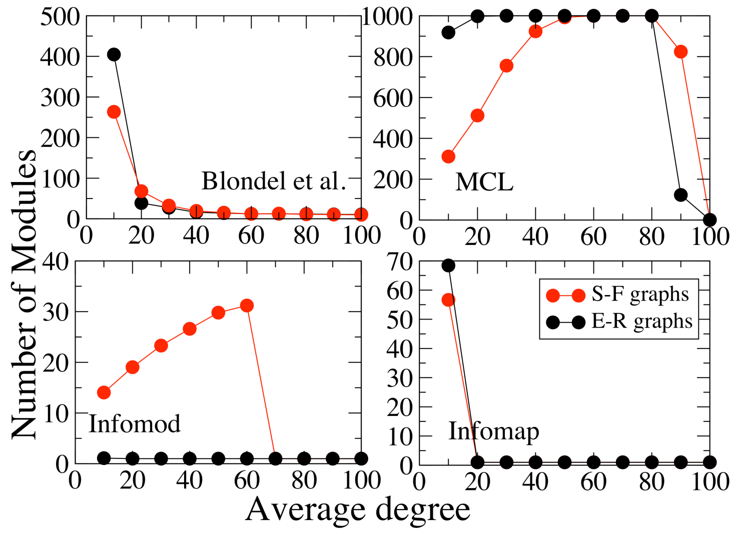


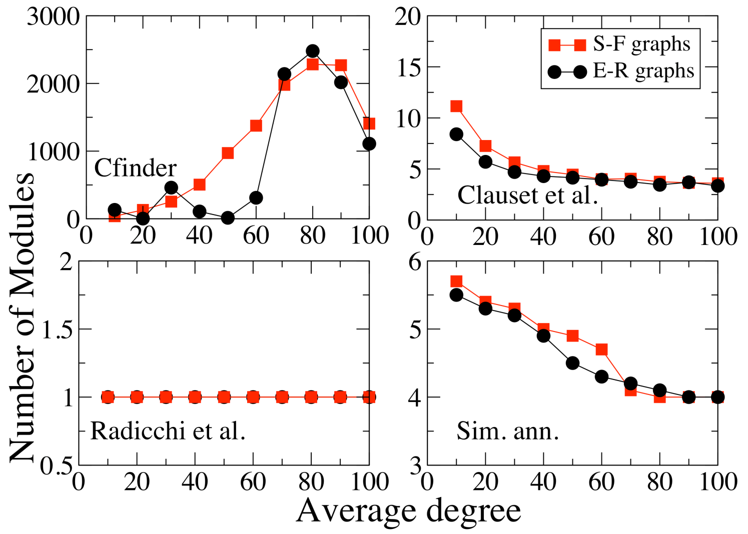


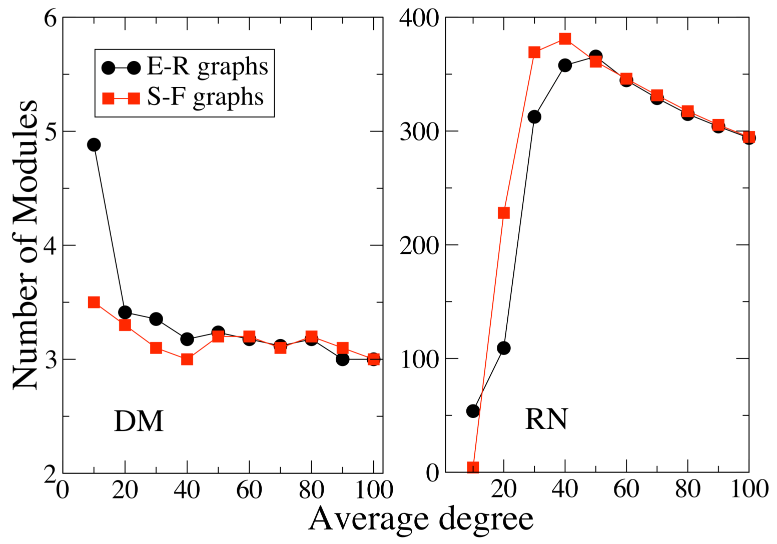


Testing algorithms

Random graphs







Outlook

A long way to go … more questions than answers from clustering

- Overlapping communities
- Hierarchies
- Testing
- Computational complexity
- Clustering in dense correlation matrices (i.e. neither sparse nor complete)

More rigorous definition of the problem!

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

Keywords:
Graphs
Clusters
Statistical physics

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 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.

Contents

* Tel.: +39 011 6603090; fax: +39 011 6600049.
E-mail address: fortunato@isi.it.

0370-1573/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.physrep.2009.11.002

S. F., arXiv: 0906.0612, Physics Reports 486, 75-174 (2010)

Plan of the course

I. Networks: definitions, characteristics, basic concepts in graph theory
II. Real world networks: basic properties
III. Models
IV. Community structure I
V. Community structure II
VI. Dynamic processes in networks