

Machine Learning for Network Modeling

Argyris Kalogeratos

[name.surname@ens-paris-saclay.fr](mailto:argyris.kalogeratos@ens-paris-saclay.fr)

LIPS research group

Center Giovanni Borelli

ENS Paris-Saclay

Who's talking?

Argyris Kalogeratos

- **Researcher** at the **Center Giovanni Borelli***, ENS Paris-Saclay
- Background: Computer Science – Machine Learning
- Coordinating the **Machine Learning for Graphs** research theme within the **LIPS team**** of the lab

* ex. Center of Applied Mathematics – CMLA, ENS Cachan

** Learning and Information Processing Systems

How is this going to work?

Attend the **courses**

Do a short **project**

- Using the tools of the course for a problem of your main discipline or a thematic you'd like to pursue in the future
- The subject and perimeter of each project should be discussed
- Deliverables: report + codes (Python, R, Matlab...)

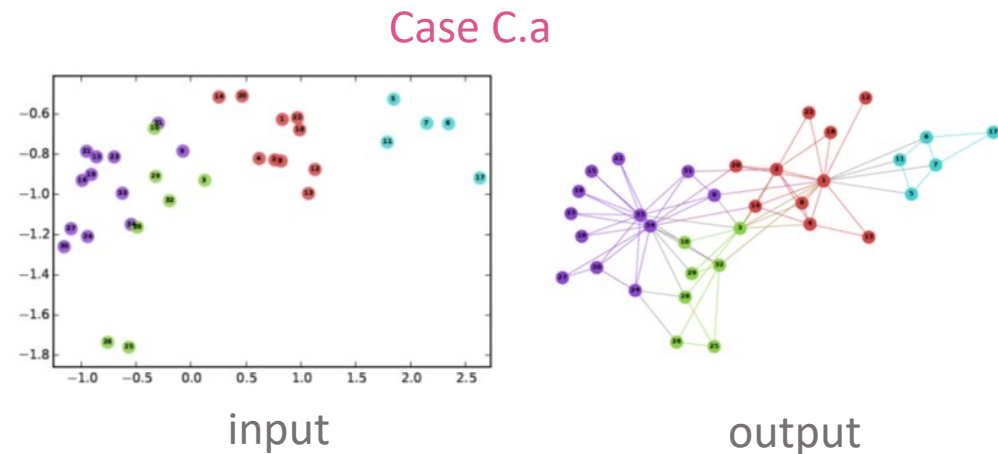
∴ In this lecture

1. How to define structure when studying networks
2. Why network 'structure' matters
3. Community detection
4. Possible projects

Structure in Network Science

Defining a graph

- **First thing first: How we can define a graph for a given problem?**
 - Case A: the graph is given (graph or a matrix for relational data)
 - Use it as is
 - `correct` / `improve` the input graph
 - Case B: for native network problems, network modeling can define a problem-specific graph
 - Case C: infer it from the data
 - Case C.a: for a **point-cloud**, any sort of similarity matrix can serve the role of a graph
 - Usefull tools to make a sparse graph:
 - k-Nearest Neighbors
 - kernelized similarities (e.g. via a Gaussian kernel)
 - value thresholding (setting to zero the low similarity values)
 - Case C.b: infer it from more **complicated data**, such as signals



Studying structure

- Macro-level
 - Degree distribution
 - Small-world phenomena (i.e. short diameter)
 - # Connected components
 - Network model identification
- Meso-level
 - Motifs (e.g. triads, cliques, cores, ...)
 - Group behavior (for dynamic graphs)
 - Structural holes / weak ties
 - Community structure
- Micro-level
 - Node-user modeling
 - Static: reciprocity, node's relation to communities: hub, internal, interface nodes, ...
 - Dynamic: Actions (for dynamic graphs)



Distant cousins

Link analysis ... qualitative

Network analysis ... quantitative

Recall: Direct vertex connectivity



non-connected



simply connected



one-way connected



two-way connected
(*reciprocity*)

Meso: Graph motifs

Graph motifs are statistically significant sub-graphs or patterns existing in a graph

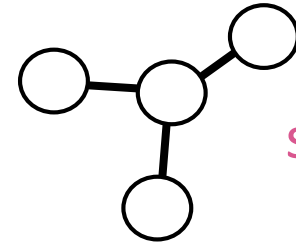
- Very complicated to work with large motifs
- Usual analysis goes up to motifs of 3 to 5 nodes

Clique is a complete (sub)graph of vertices that is totally connected (i.e. complete) – ... *restrictive!!*

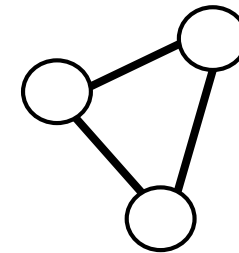
Clique relaxations

k -clique is a set of vertices among any pair of which the distance (shortest path) $< k$

k -club (or **k -plex**) is a set of vertices that has diameter $< k$



star motif



clique



Clique relaxation: the paths can pass from anywhere in the graph, not just the induced subgraph

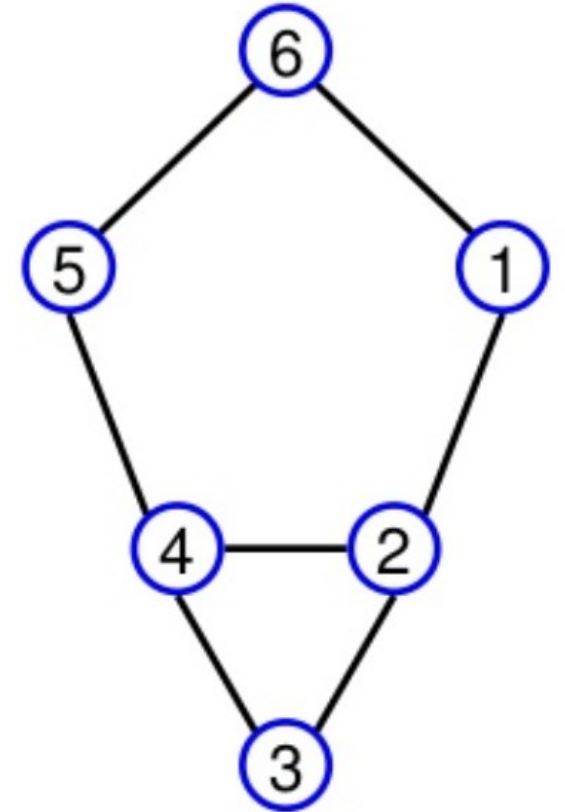
Meso: Graph motifs

Examples

- $\{2,3,4\}$ is a 1-club (also regular clique)
- $\{1,2,4,5,6\}$ is a 2-club
- $\{1,2,3,4,5\}$ is a 2-clique but not a 2-club

(Note: In this case we consider the 5 vertices as part of the full graph and therefore 1 is 2 hops away from 5)

Clique relaxation:
the paths can pass from anywhere in the graph, not just the induced subgraph



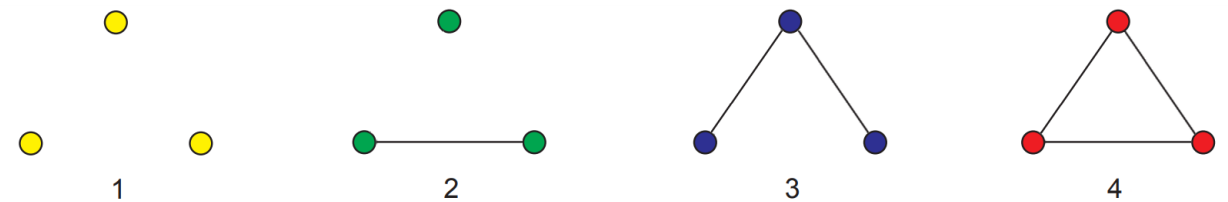
Meso: Graph motifs

Triads are motifs that can be formed between three vertices

Labeling undirected triads can be done using the numbering 1...4

- $label - 1 = \# \text{ edges in triad}$

The 4 possible undirected triads



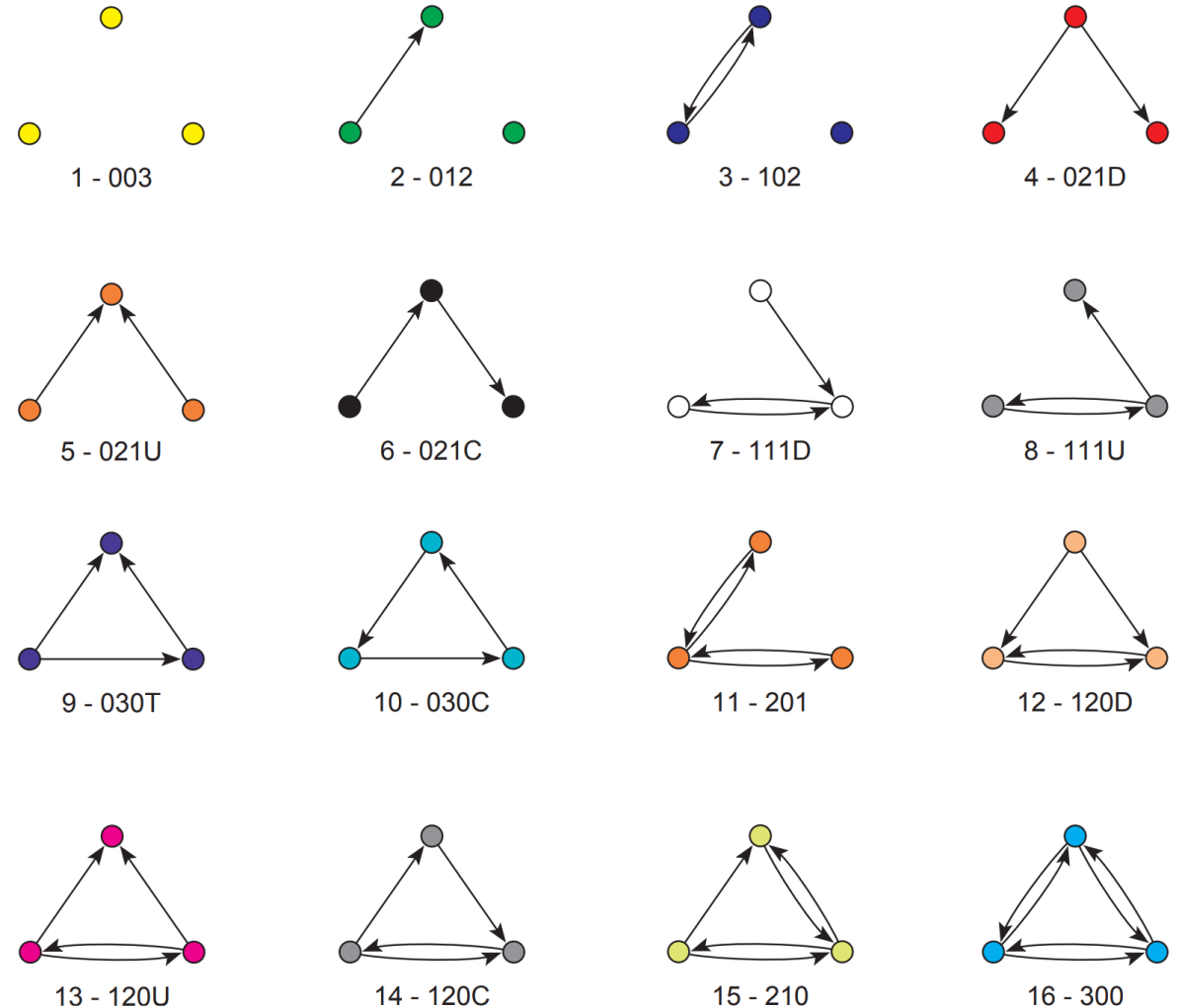
Also a clique

Meso: Graph motifs

The 16 possible directed triads

Labeling undirected triads can be done

- either by the numbering 1...16
- or by using a format *xyz*
 - *x* – # pairs of vertices connected with bidirected edges
 - *y* – # pairs of vertices connected with one-direction edges
 - *z* – # non-connected pairs of vertices
 - When unclear, use also one letter:
Down, Up, Cyclic, Transitive



Meso: Graph motifs

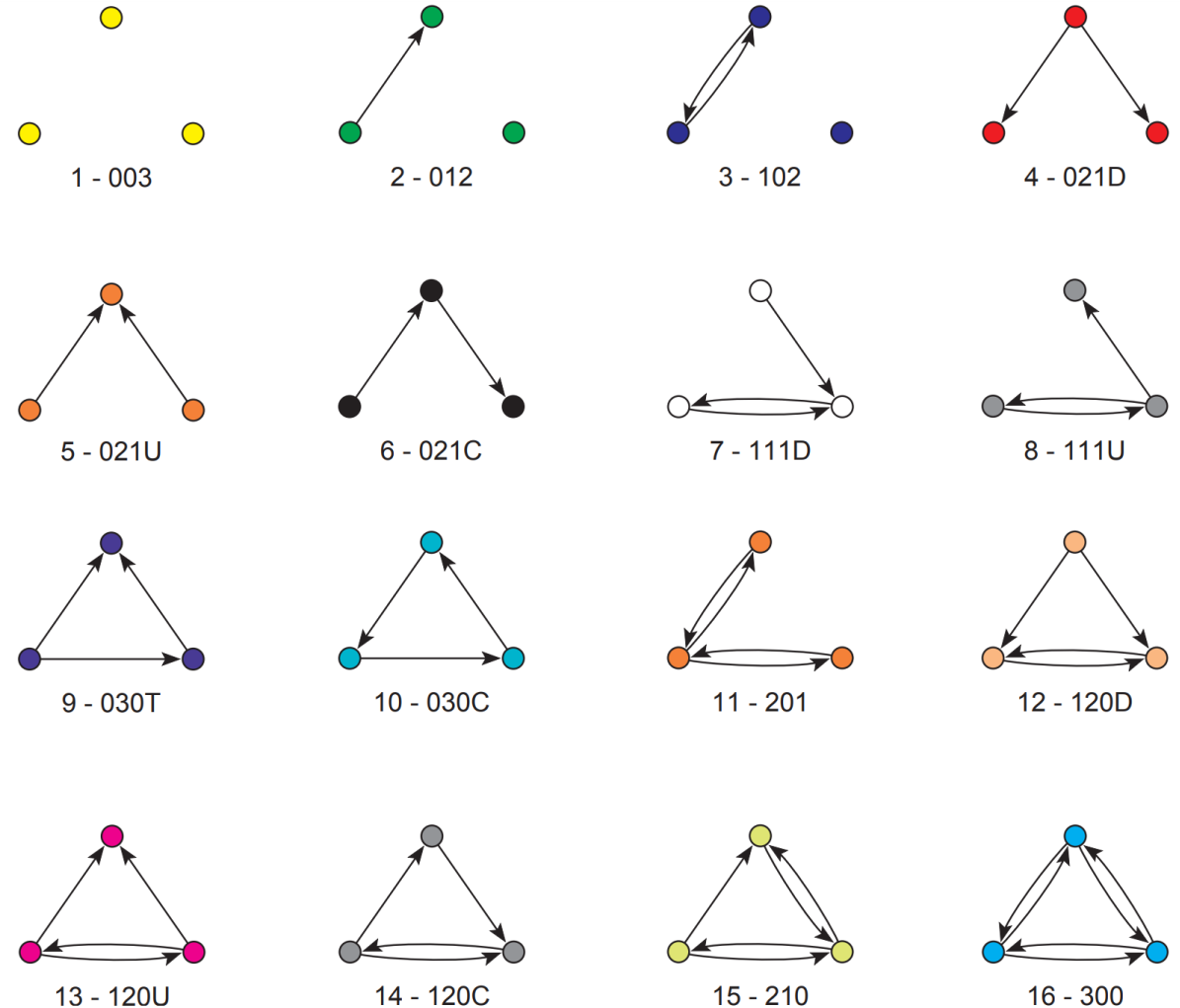
The 16 possible directed triads

Transitivity in triads

If there are edges $u \rightarrow v$ and $v \rightarrow w$, then there exists also the edge $u \rightarrow w$

Labeling undirected triads can be done

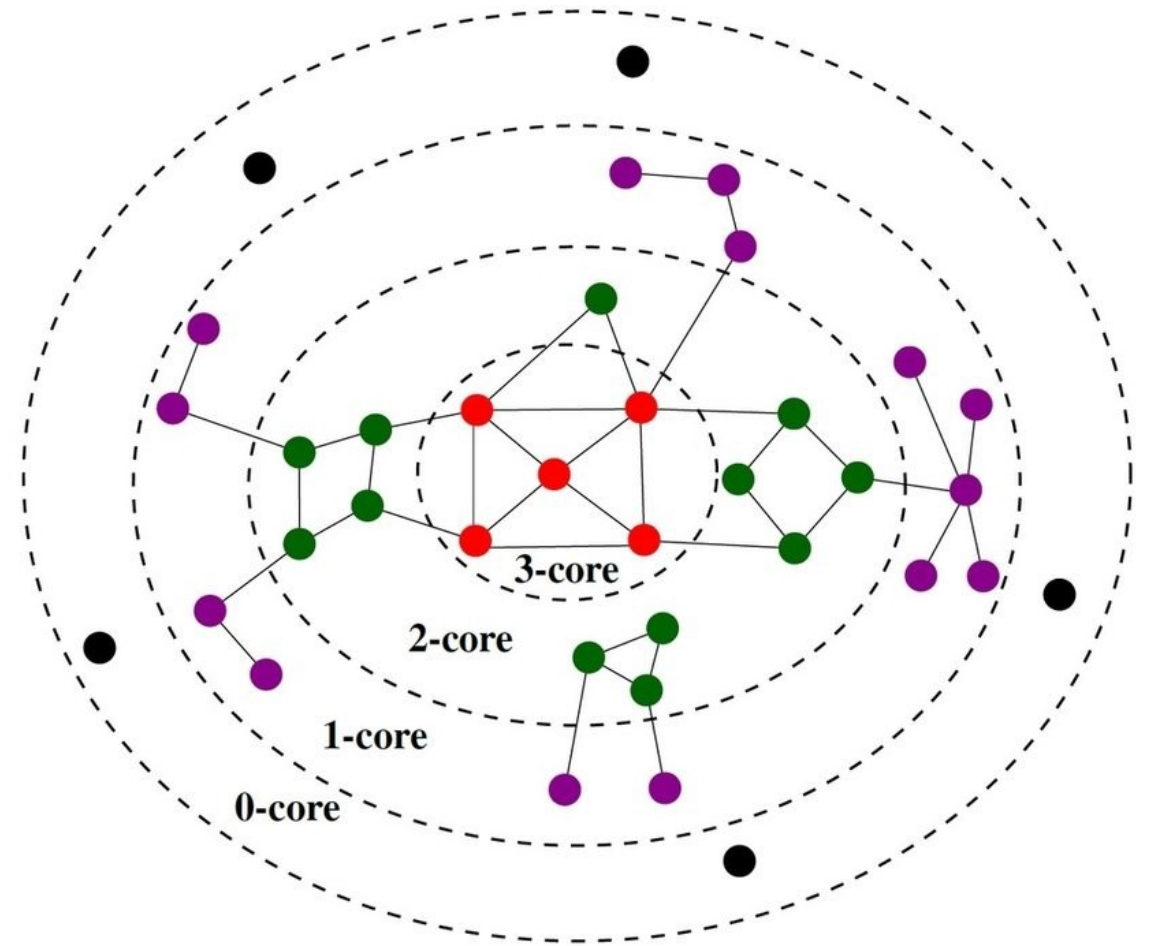
- Triads 9, 12, 13, 16 are transitive
- Triads 6, 7, 8, 10, 11, 14, 15 are intransitive (... to different levels)
- Triads 1, 2, 3, 4, 5 are vacuously transitive (i.e. cannot be categorized)



Meso: Graph cores

A k -core is a maximal connected (sub)graph, whose vertices have at least degree k (to vertices that belong or not to the core)

- Cores do not respect density!!

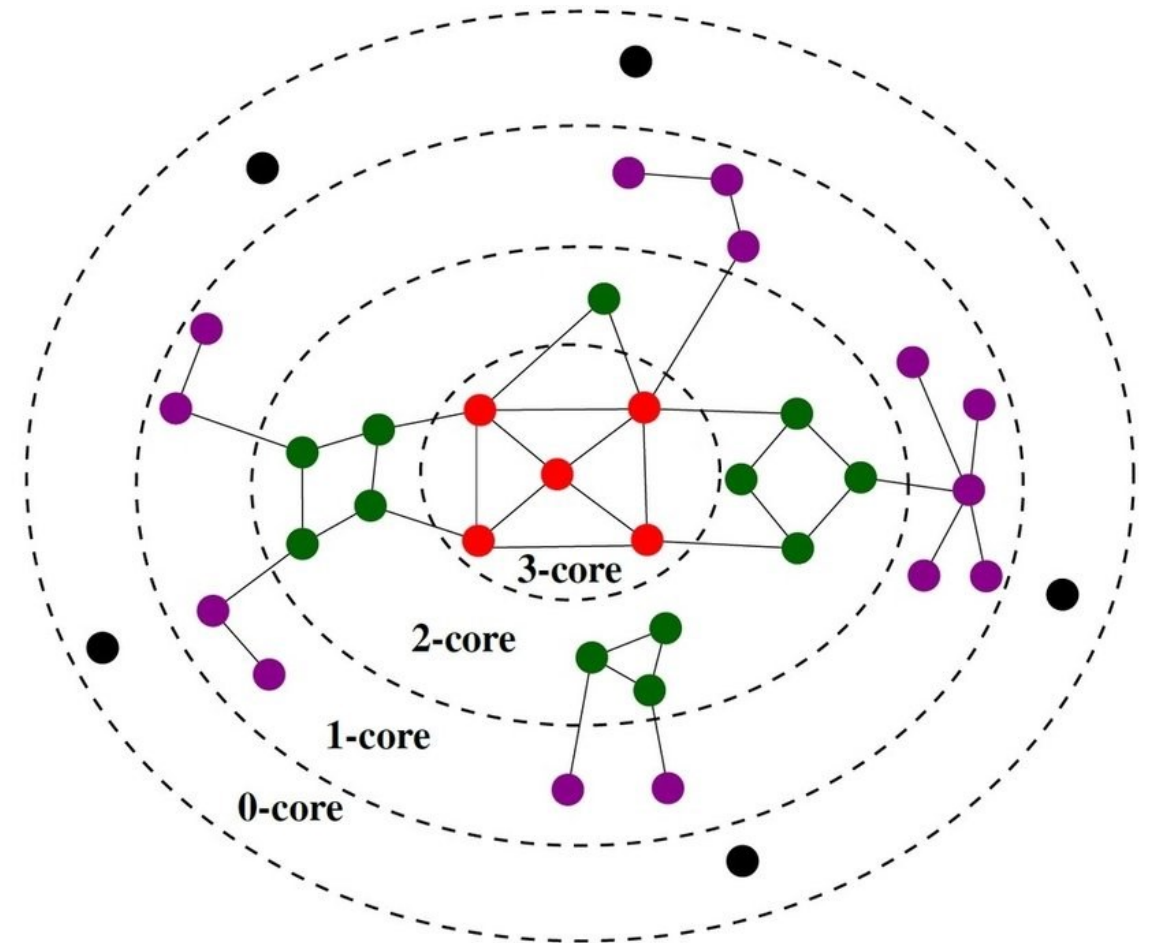
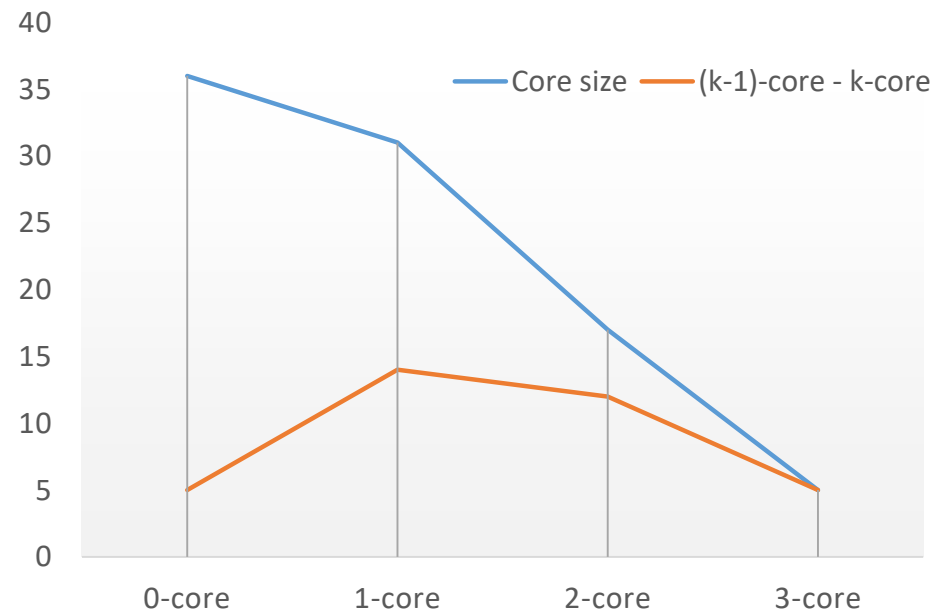


Meso: Graph cores

A k -core is a maximal connected (sub)graph, whose vertices have at least degree k (to vertices that belong or not to the core)

- Cores do not respect density!!

Graph degeneracy analysis

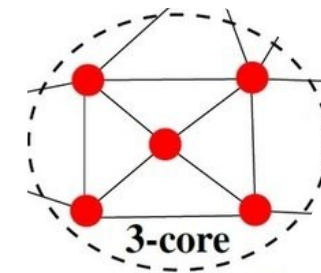
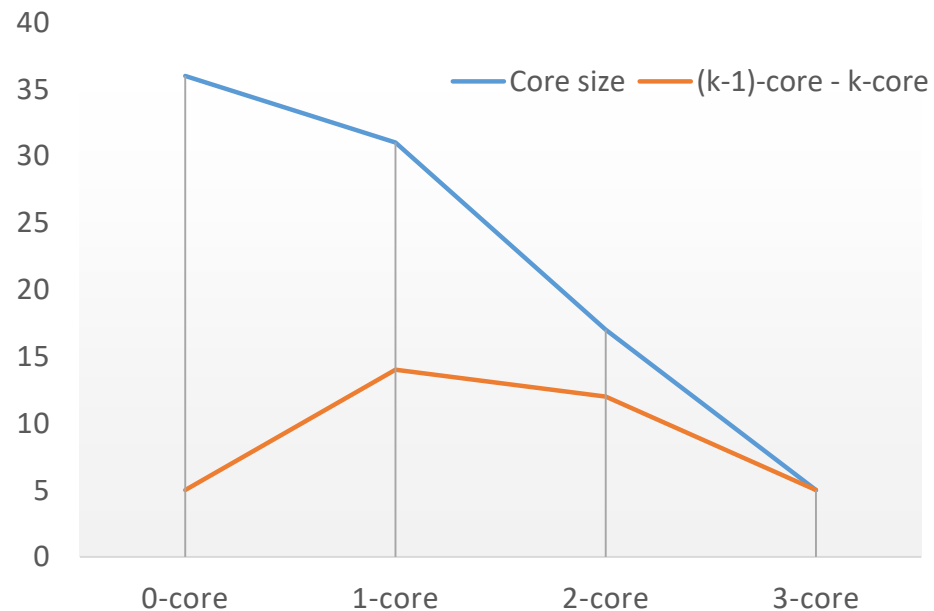


Meso: Graph cores

A k -core is a maximal connected (sub)graph, whose vertices have at least degree k (to vertices that belong or not to the core)

- Cores do not respect density!!

Graph degeneracy analysis



Also a 2-clique
and a 2-club

Meso: Density/clusters/communities

Clusterability (Statistical ML principle)

Intra-cluster density / inter-cluster density $\gg 1$

Network closure

Homogeneity is higher inside a cluster than across different clusters. New edges are more likely to appear inside clusters.

Triadic closure

Emergent edges will most likely 'close' some triangle

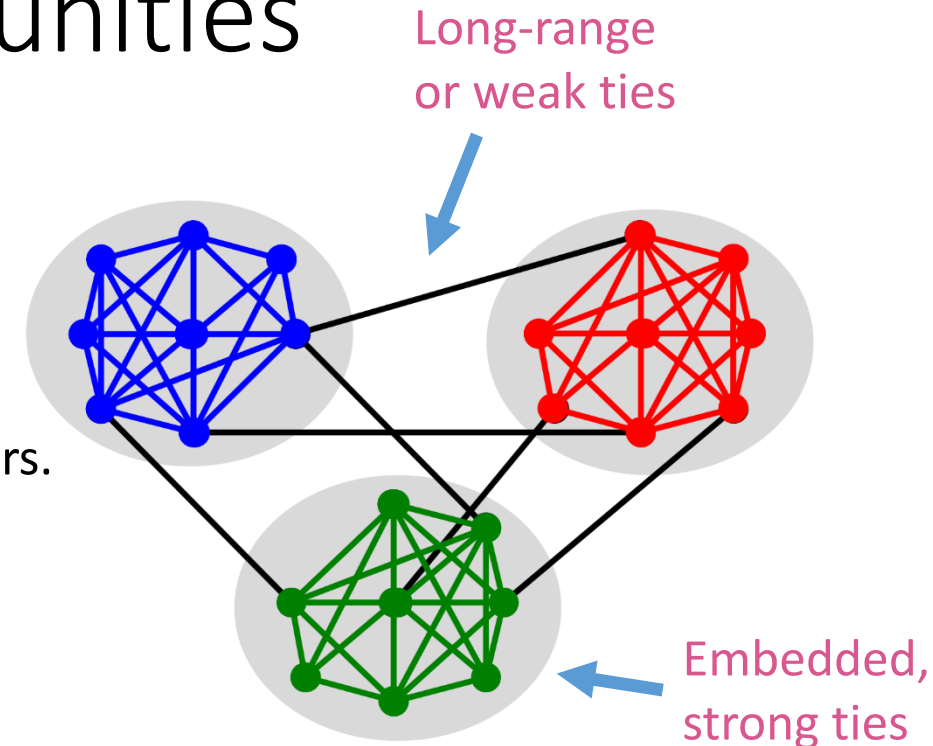
Weak-ties property

The strongest the connection (tie) between two individuals, the more likely they share contacts. Weak-ties are responsible for most communication among clusters.

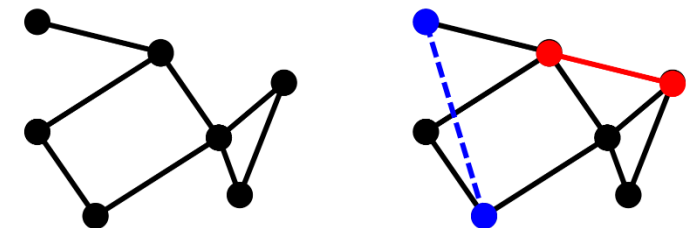
Structural hole

(Similar to the above but with the opposite direction of causality)

Is the gap between two individuals who have complementary sources to information.



Triadic closure



Network cohesion

- The concept of **network cohesion** is central for answering many questions:
- Definitions depend on the context
 - Scale from **local** (e.g. triads) to **global** (e.g. giant component)
 - **Explicit** (e.g. cliques) or **implicit** (e.g. clusters) definition
- Desirable in-group properties of any **cohesive group**:
 - *Familiarity*... high degree
 - *Reachability*... small distance
 - *Robustness*... connectivity
 - *Density*... edge density
- **Cliques** indeed maximize these properties, but... are restrictive!
 - Large cliques are super rare
 - They are sensitive: one edge destroys the property
 - Usually very costly to identify, or find maximal cliques in a graph (NP-complete)

Network cohesion - Density

- **Global network density**: measures how close to being a clique

$$\text{density}(G) = \frac{N_e}{N_v(N_v - 1)/2} \in [0,1]$$



This is for the whole undirected graph, but can be applied also to subgraphs

- Alternatively, it can be seen as a rescaling of the average degree

$$\bar{d}(G) = \frac{1}{N_v} \sum_{v \in V} d_v = \frac{1}{N_v} 2N_e \xrightarrow{\text{repl. } N_e} \text{density}(G) = \frac{\bar{d}(G)}{N_v - 1}$$

- **Local density at node v** : we can use v 's **egonet**, i.e. the subgraph induced by its neighbors

Network cohesion – Clustering coefficient

- **Clustering coefficient of node v** : measures the fraction of v 's neighbors that are connected (here E_v is the #edges in the v 's egonet)

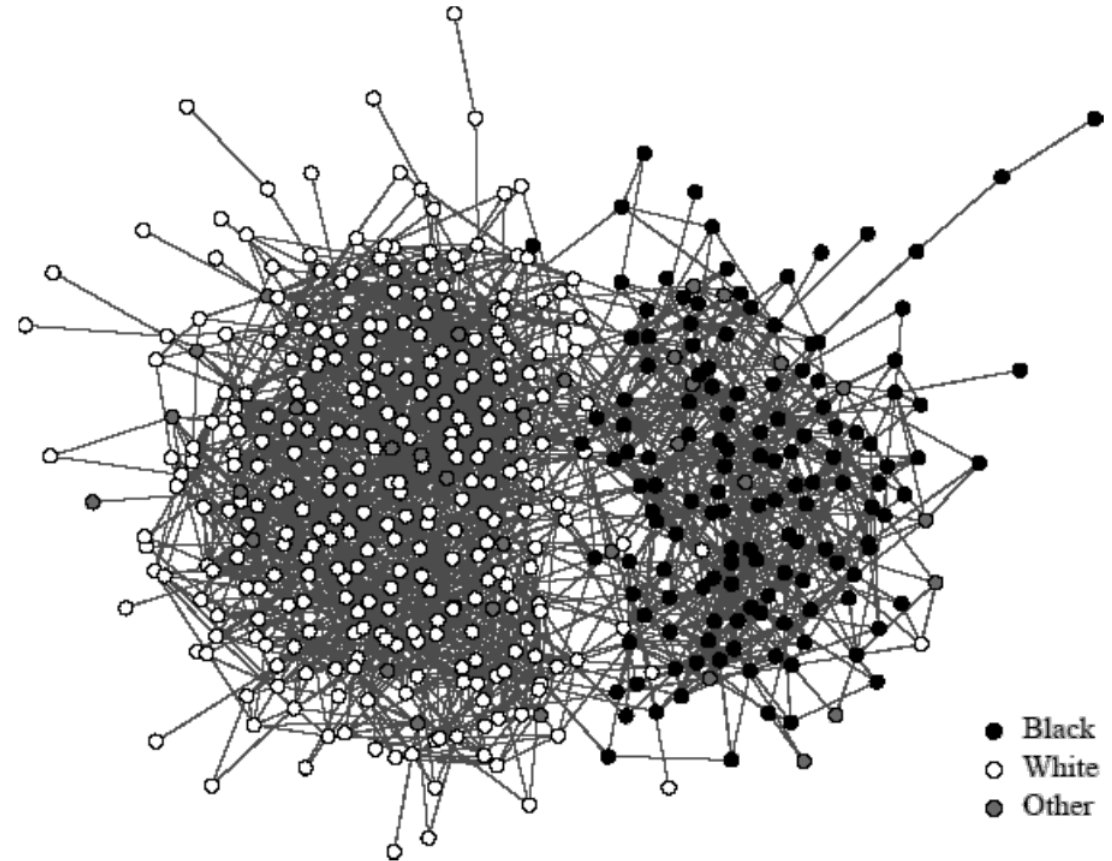
$$cl(v) = \frac{2E_v}{d_v(d_v - 1)} \in [0,1]$$

- **Global (average) clustering coefficient**

$$cl(G) = \frac{1}{N} \sum_{v \in V} cl(v)$$

Assortative mixing

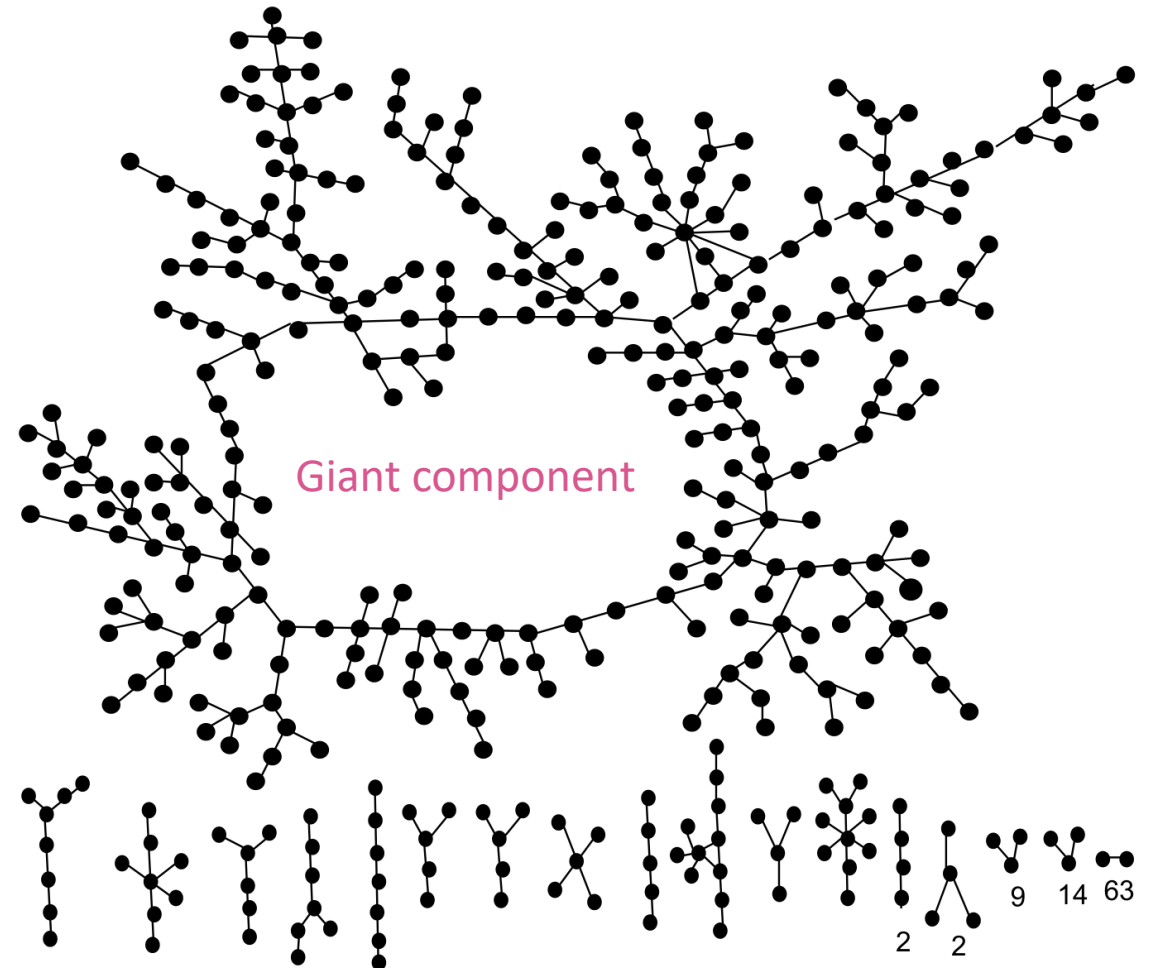
- **Homophily or assortative mixing**
Individuals tend to get connected/interact with equal/similar others
E.g. high-school students by race, bloggers by political party, ...
- **Assortativity coefficient**
Measures this property [Newman '03]
- **Dissortative mixing** exists too...
E.g. romantic relationships between males and females
→ *heterophilous graphs*



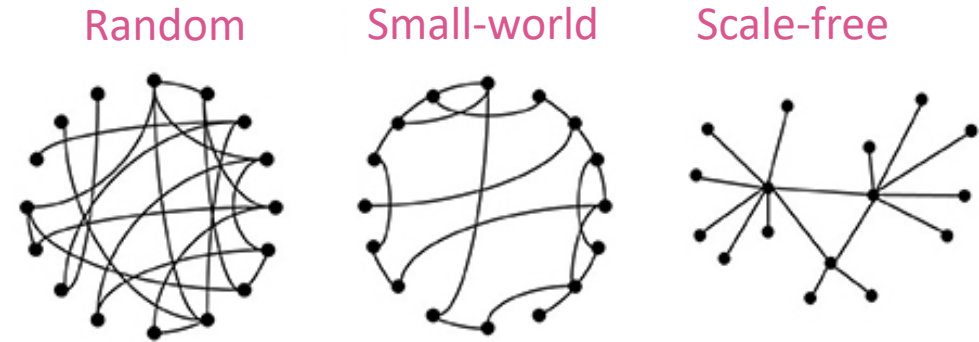
Network cohesion – Giant component

- Large real-world networks typically exhibit one giant component
- **Example:** romantic relationships in a US high school [Bearman et al. '04]

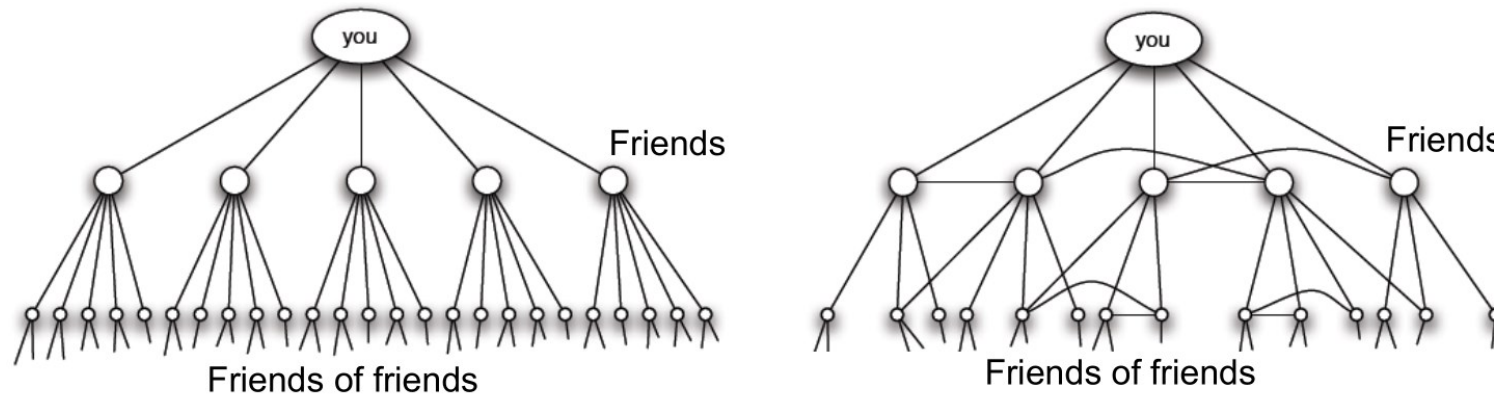
Small disconnected components



Small world phenomenon



- It refers to small average (shortest) path length $\approx O(\log N \nu)$
- Intuitively... long hops reduce drastically the length of paths

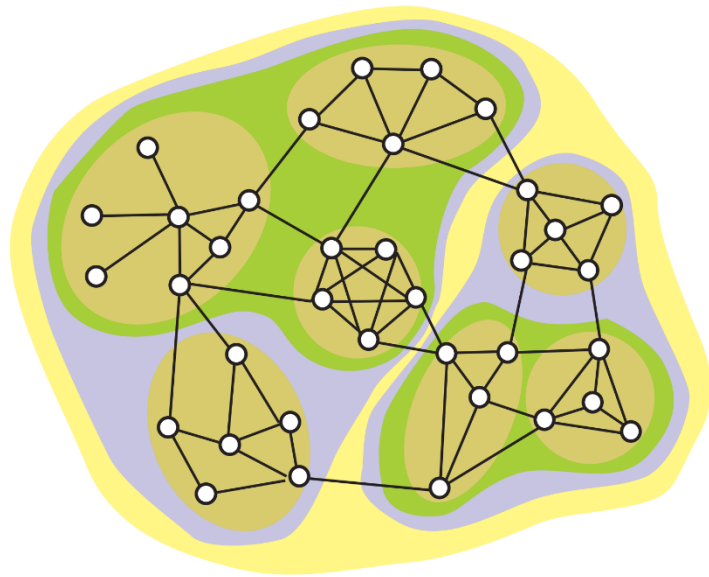


- This property facilitates the spread of information, diseases, etc...
- **Put in perspective:** Spread speeds-up in one cluster, yet a different cluster may be the reason to get 'blocked'

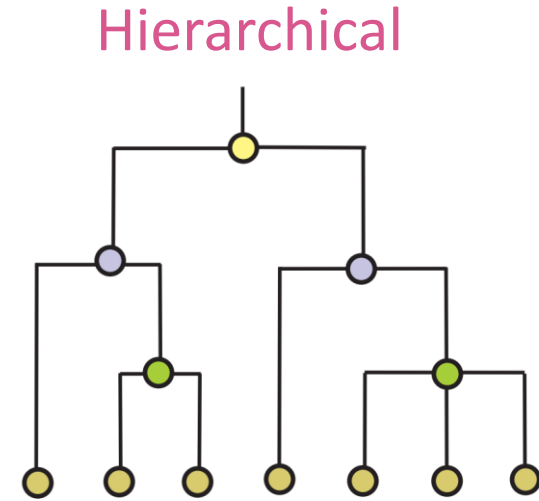
Network cohesion – Group centrality

- Recall **node centrality measures**
 - ***Closeness centrality***: the node has small average shortest path to other nodes
 - ***Betweenness centrality***: the node is frequently part of the shortest path between pairs of other nodes
- These can be extended to measure **group centrality**

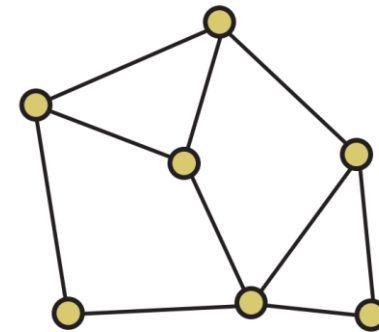
Global vs local graph views



Global views

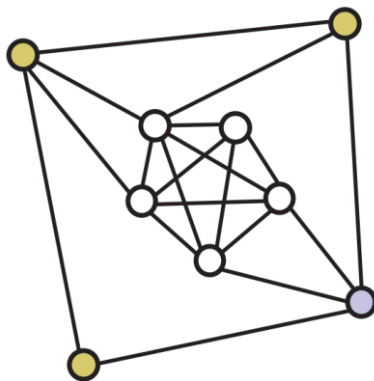


Reduction

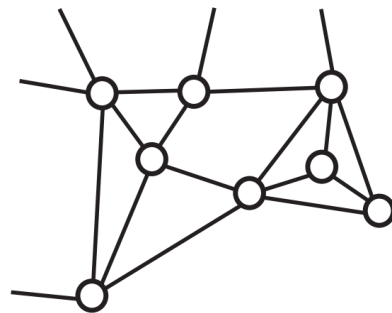


Local views

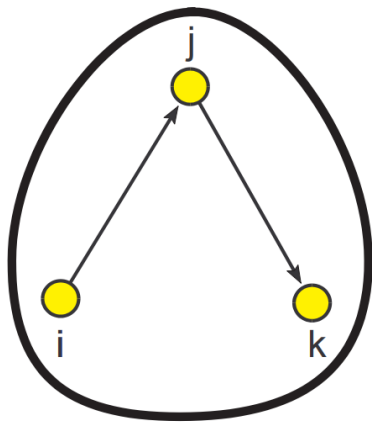
Context



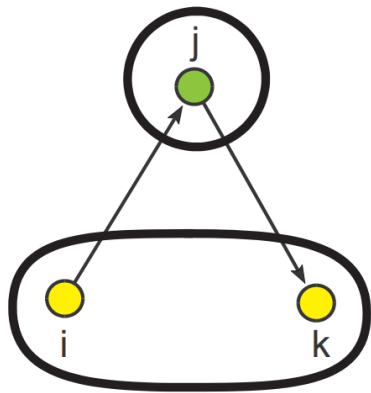
Cluster cut-out



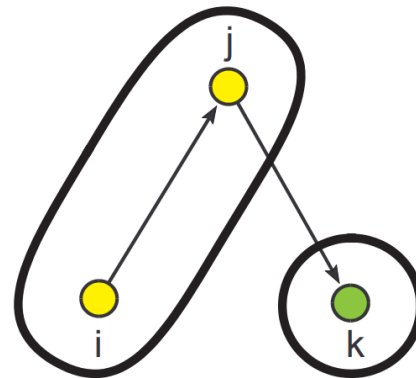
Micro: Node 'roles'



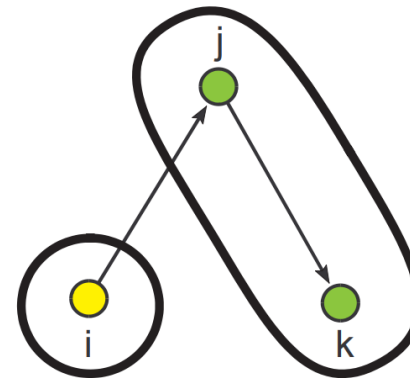
coordinator



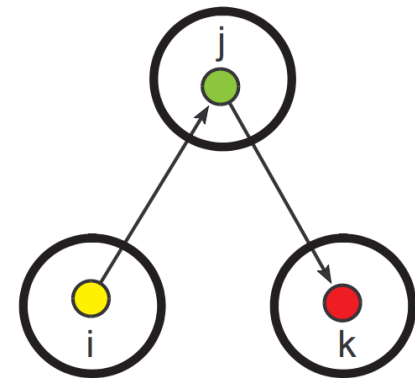
itinerant broker



representative



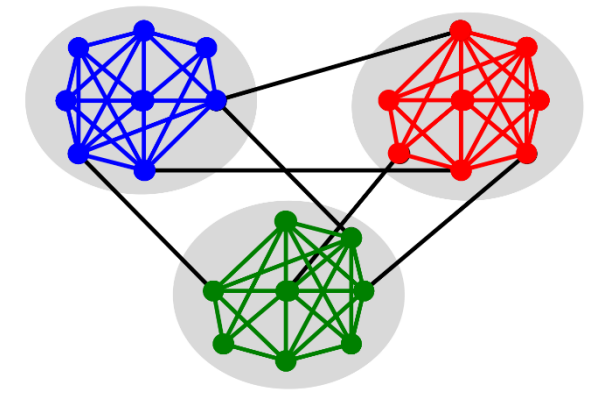
gatekeeper



liaison

Community detection

Community detection

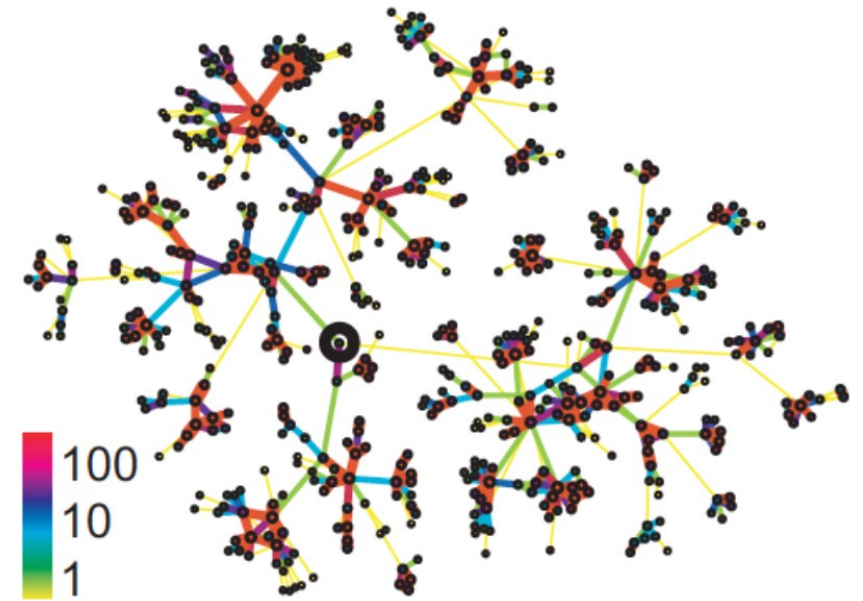


Split V into a given number k (non-overlapping) groups

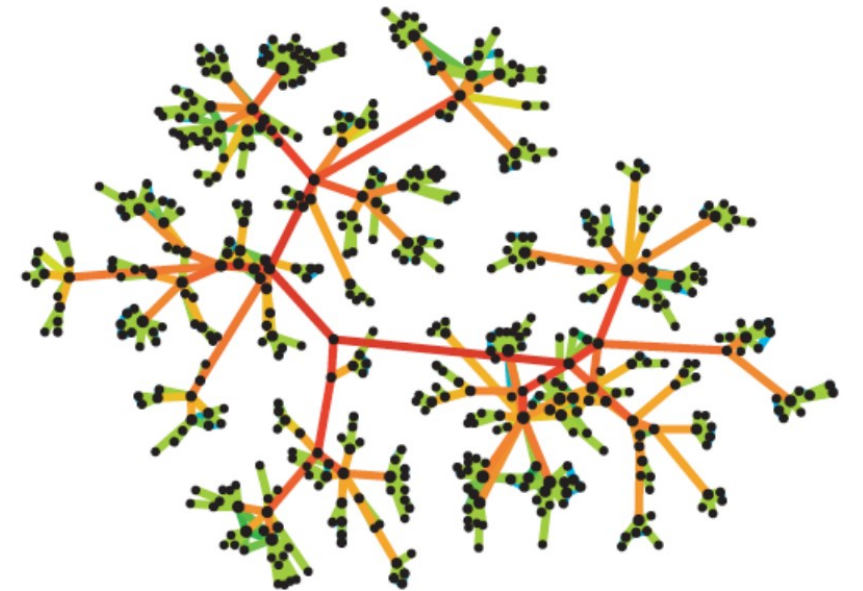
- Finding communities in networks is a challenging **clustering problem**
 - No consensus on the definition of « *what is a cluster* »
 - NP-hard problem: i.e. there are combinatorics behind separating optimally subsets of vertices and the problem cannot be solved in polynomial time.
 - Lack of ground truth to validate results
- Result of great interest for a plethora of reasons
 - Understanding data
 - Visualization
 - Compression
 - ...

Graph partitioning

- Idea: try removing local bridges (weak ties) to decompose the graph
 - Combinatorial problem again: these may be many and with different importance
- Idea: target edges with large edge betweenness centrality (eBC)
 - These edges are part of many shortest paths among vertices
 - => stand at the interface between clusters



Edge strength



Edge betweenness

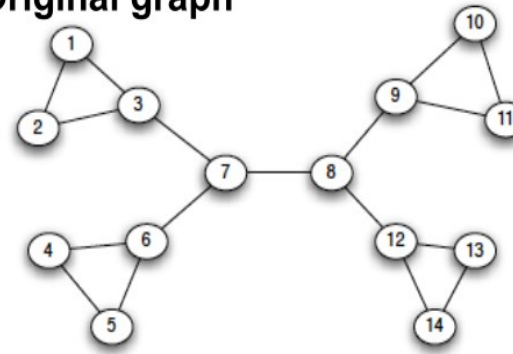
Girvan-Newman's method

- Idea: Find and remove **spanning links** between cohesive subgroups
- **Algorithm:** Repeat until there are no edges left
 - Calculate the eBC of all remaining edges
 - Remove the edges(s) with the highest eBC
- What we get
 - Different connected components as communities
 - Nested partitioning (top-down), returns a dendrogram
 - Requires recomputing all centralities at each step $O(NvNe)$

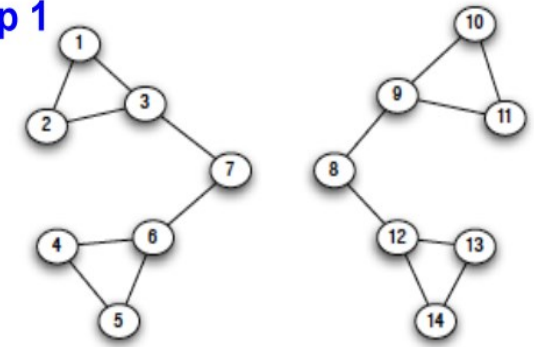
Girvan-Newman's method

In action

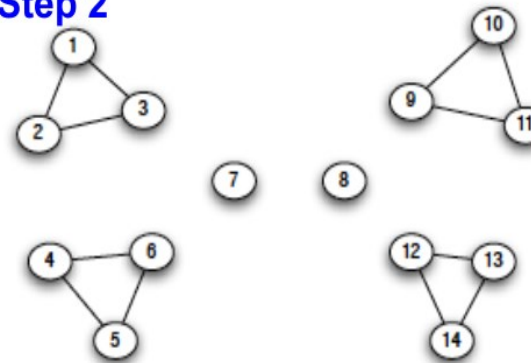
Original graph



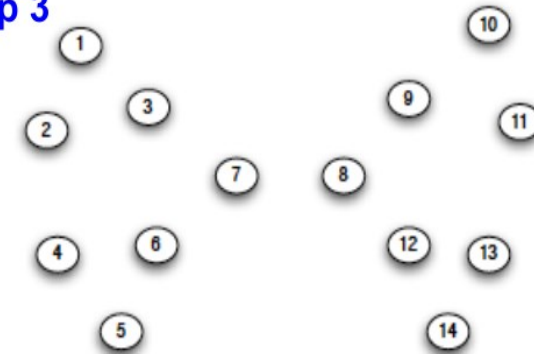
Step 1



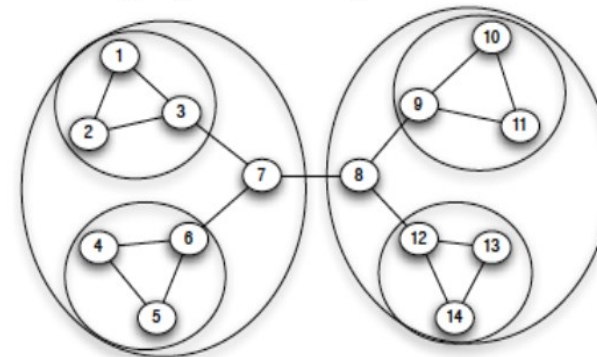
Step 2



Step 3



Nested graph decomposition



Hierarchical clustering

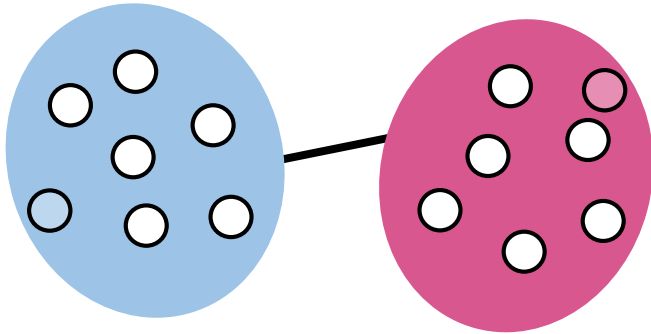
- A greedy approach that successively changes the solution
 - **Agglomerative**: is merging groups (bottom up)
 - **Divisive**: is splitting groups
 - Returns a *dendrogram* with all the hierarchical structure
 - Cutting the hierarchy at any level γ (from the top) gives $\gamma + 1$ clusters
- At each step the change should minimize a **cost function**
 - This measures the dissimilarity between the vertices in each group
 - Many options; a simple one is just the Euclidean or Manhattan

Agglomerative clustering

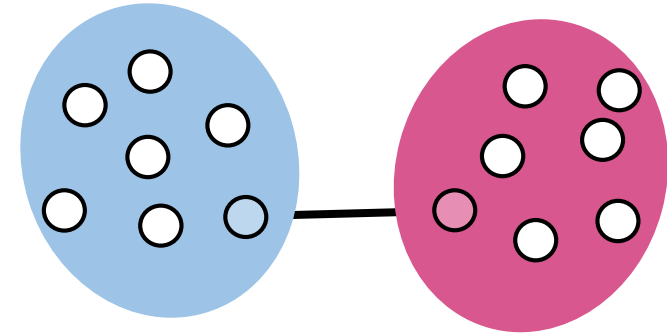
- Hierarchical Agglomerative clustering (bottom up)
 - 1) Choose dissimilarity metric between groups (!!!)
 - 2) Assign each vertex to its own singleton group (1 vertex per group)
 - 3) Merge the two groups with the smallest dissimilarity
 - 4) Compute the dissimilarity of the new group with the preexisting ones
 - 5) If the current number of groups > 1 ... **goto** (3)
- Most critical part is to define **group similarity**

Agglomerative clustering

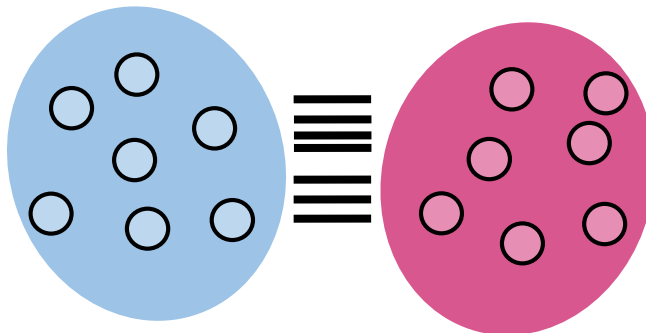
Complete linkage



Single linkage



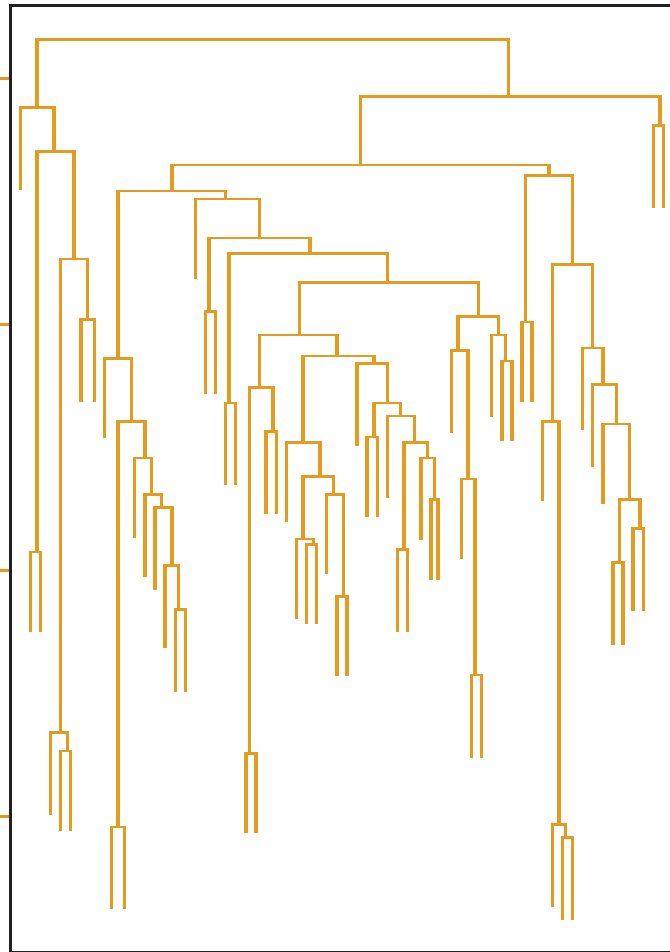
Average linkage



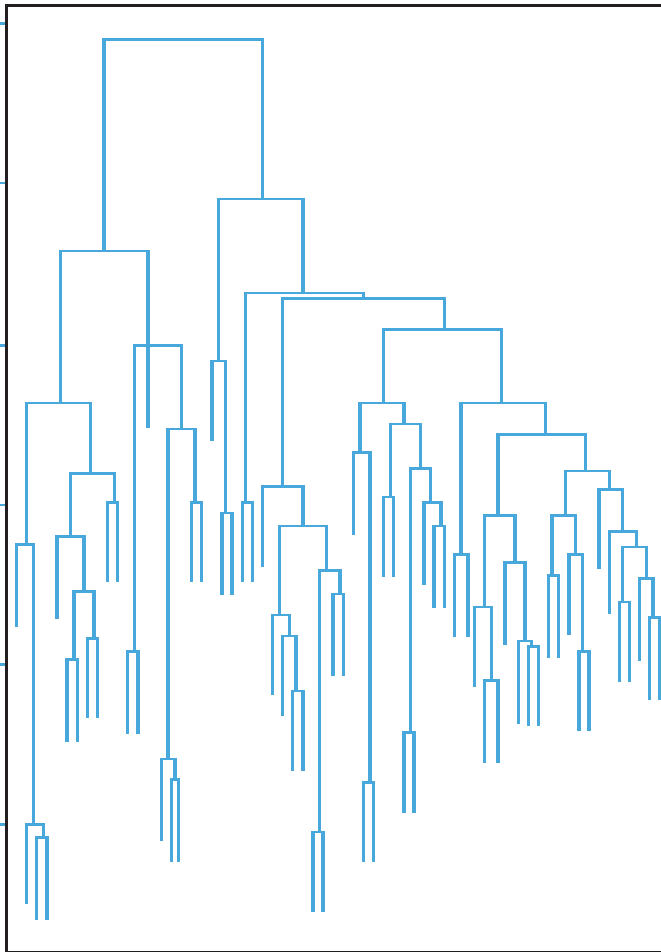
Agglomerative clustering

In action – dendrograms

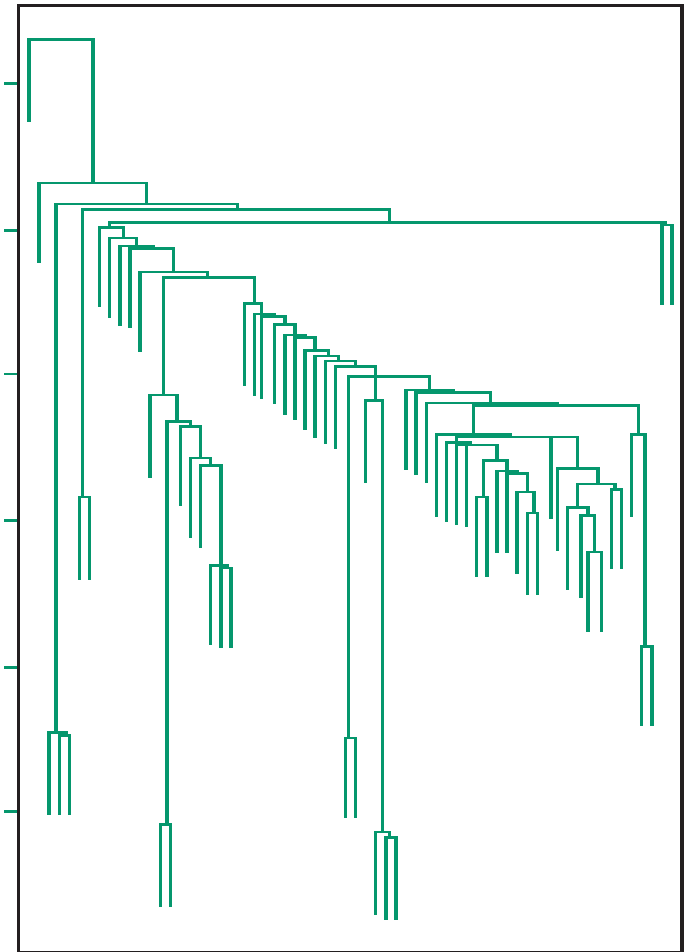
Average linkage



Complete linkage



Single linkage



Modularity-based community detection

Modularity

- Consider a graph G and a partition into groups $s \in S$

$$Q(G, S) \propto \sum_{s \in S} [(\# \text{ of edges within group } s) - \mathbb{E} [\# \text{ of such edges}]]$$

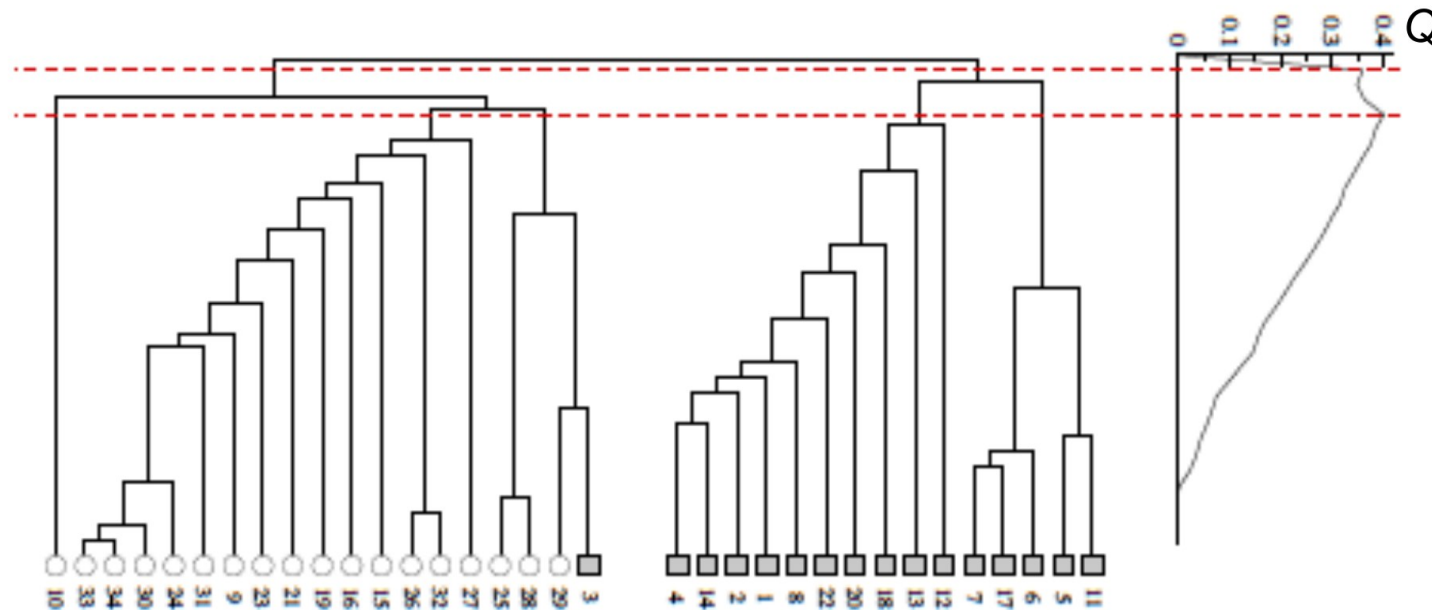
- After normalization such that $Q(G, S) \in [-1, 1]$

$$Q(G, S) = \frac{1}{2N_e} \sum_{s \in S} \sum_{i, j \in s} \left[A_{ij} - \frac{d_i d_j}{2N_e} \right]$$

- *Null model*: one with random edges that preserves the degree distribution!

Modularity-based community detection

- We can evaluate modularity at the levels of a hierarchical dendrogram
- Keep the solution of the level that gives the ‘best’ community structure w.r.t Q



- Why not to optimize the partitioning directly w.r.t Q

Modularity-based community detection

Optimizing modularity

- Define a modularity matrix B with entries $B_{ij} = A_{ij} - \frac{d_i d_j}{2N_e}$
- Any 2-partition S can be defined by a $\{+1, -1\}$ binary vector $s = [s_1, \dots, s_{N_v}]^T$
- Modularity gets a quadratic form

$$Q(G, S) = \frac{1}{4N_e} \sum_{i,j \in V} B_{ij} s_i s_j = \frac{1}{4N_e} \mathbf{s}^T \mathbf{B} \mathbf{s}$$

- For **graph bisection** (2-split) the modularity-based criterion is formulated as

$$\hat{\mathbf{s}} = \arg \max_{\mathbf{s} \in \{\pm 1\}^{N_v}} \mathbf{s}^T \mathbf{B} \mathbf{s}$$

- Due to the ‘nasty’ binary constraints of s , this optimization is NP-hard!

Modularity-based community detection

The relaxation of constraints yields a feasible optimization

- By letting constraints $s \in \mathbb{R}^{N_v}$, $\|s\|_2 = 1$ we get

$$\hat{\mathbf{s}} = \arg \max_{\mathbf{s}} \mathbf{s}^\top \mathbf{B} \mathbf{s}, \quad \text{s. to } \mathbf{s}^\top \mathbf{s} = 1$$

- Associate a Lagrange multiplier λ to the constraint $\mathbf{s}^\top \mathbf{s} = 1$
- Optimality conditions yields

$$\nabla_{\mathbf{s}} [\mathbf{s}^\top \mathbf{B} \mathbf{s} + \lambda(1 - \mathbf{s}^\top \mathbf{s})] = \mathbf{0} \Rightarrow \mathbf{B} \mathbf{s} = \lambda \mathbf{s}$$

- Conclusion: s is an eigenvector of B with eigenvalue λ
- At the optimum $Bs = \lambda s$ the objective becomes

$$\mathbf{s}^\top \mathbf{B} \mathbf{s} = \lambda \mathbf{s}^\top \mathbf{s} = \lambda$$



To maximize modularity,
pick the dominant
eigenvector of B

Modularity-based community detection

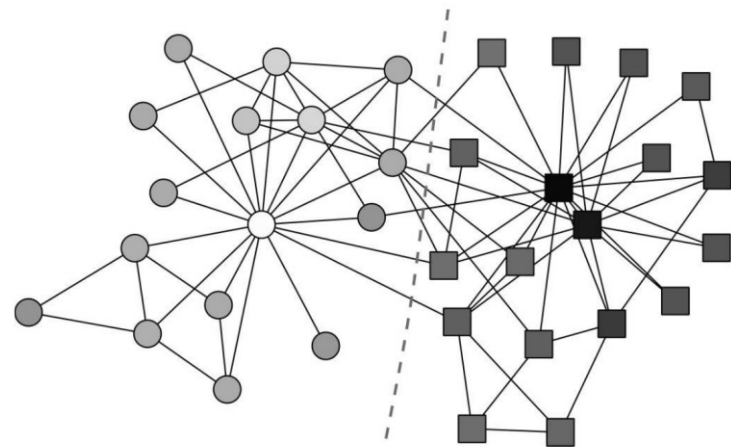
- ▶ Spectral modularity maximization algorithm

S1: Compute modularity matrix \mathbf{B} with entries $B_{ij} = A_{ij} - \frac{d_i d_j}{2N_e}$

S2: Find dominant eigenvector \mathbf{u}_1 of \mathbf{B} (e.g., power method)

S3: Cluster membership of vertex i is $s_i = \text{sign}([\mathbf{u}_1]_i)$

- ▶ Multiple (> 2) communities through e.g., repeated graph bisection



Zachary's karate club

Spectral clustering

One moment... lets recall some spectral properties of the matrix representations of a graph

Graph Laplacian

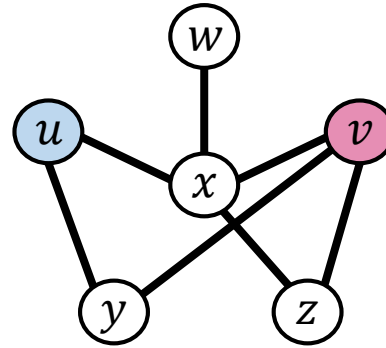
Although there are many definitions for **Laplacian matrix**, the most common is the following $N_v \times N_v$

$$L = D - A$$

where D is a matrix with the degrees of the graph vertices in its diagonal, and zero values everywhere else

Graph Laplacian

Graph



Laplacian matrix L

	u	v	w	x	y	z
u	2			-1	-1	
v		3		-1	-1	-1
w			1	-1		
x	-1	-1	-1	4		-1
y	-1	-1			2	
z		-1		-1		2

=

Degree matrix D

	u	v	w	x	y	z
u	2					
v		3				
w			1			
x				4		
y					2	
z						2

-

Adjacency matrix A

	u	v	w	x	y	z
u				1	1	
v				1	1	1
w				1		
x	1	1	1			1
y	1	1				
z		1		1		

Graph Laplacian

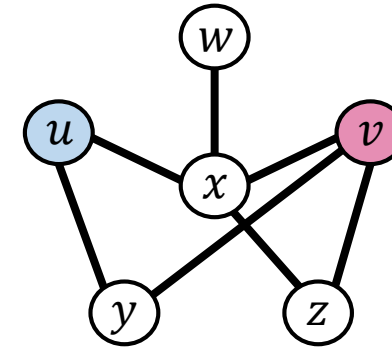
Basic properties

- Zero-sum rows and columns \implies zero-sum matrix L
- All negative values except in the diagonal
- Same off-diagonal zeros as A has information only for the directly connected pairs of vertices
- The input graph G cannot be a multigraph, edge weights are ignored
- Like in multivariate calculus, for (a problem-specific) $x \in \mathbb{R}^{N_v}$ that comes from some function $f(G, \dots)$,

$$x^T L x = \sum_{(i,j) \in E} (x_i - x_j)^2 = \sum_{i,j \in [1, \dots, N_v]} A_{ij} (x_i - x_j)^2$$

The closest $x^T L x$ is to zero, the more **smooth** the x is with respect to the graph, and so for the f

Graph



Laplacian matrix L

	u	v	w	x	y	z
u	2			-1	-1	
v		3		-1	-1	-1
w			1	-1		
x	-1	-1	-1	4		-1
y	-1	-1			2	
z		-1		-1		2

Eigen-analysis of graph Laplacian

Generally, L 's eigenvalues and eigenvectors yield a lot of interesting information for a graph regarding

- G 's connectivity
 - The smallest eigenvalue is 0 with eigenvector $\mathbf{1}$
 - If the second smallest eigenvalue is 0 then the graph is disconnected. The multiplicity of 0's gives the # of components
 - The larger the non-trivial eigenvalues, the more connected a graph is
 - A connected graph of diameter δ has at least $\delta + 1$ distinct eigenvalues
- G 's conductance (how fast does a random walk converge)
- the potential growth of a diffusion on G
- ...

Graph Laplacian

Alternative definition: the **normalized Laplacian matrix**

$$\begin{aligned}\tilde{L} &= D^{-\frac{1}{2}} L D^{-\frac{1}{2}} \\ &= D^{-\frac{1}{2}} (D - A) D^{-\frac{1}{2}} = (D^{-\frac{1}{2}} D - D^{-\frac{1}{2}} A) D^{-\frac{1}{2}} \\ &= D^{-\frac{1}{2}} D D^{-\frac{1}{2}} - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}\end{aligned}$$

Properties

- Symmetric, positive semi-definite with n non-negative eigenvalues
- The smallest eigenvalue is 0 with eigenvector $D^{-1/2}\mathbf{1}$
- More appropriate when there is degree inhomogeneity
- Constrains the eigenvalues in $[0, 2]$

Spectral clustering

Principle

- ① Use the spectral property of \mathbf{L} to perform clustering in the eigen space
- ② If the network have K connected components, the first K eigenvectors are $\mathbf{1}$ span the eigenspace associated with eigenvalue 0
- ③ Applying a simple clustering algorithm to the rows of the K first eigenvectors separate the components

↪ This principle generalizes to a graph with a single component: spectral clustering tends to separates groups of nodes which are highly connected together

Spectral clustering

Algorithm

Input: Adjacency matrix and number of classes Q

Compute the normalized graph Laplacian \mathbf{L}

Compute the eigen vectors of \mathbf{L} associated with the Q **smallest eigenvalues**

Define \mathbf{U} , the $p \times Q$ matrix that encompasses these Q vectors

Define $\tilde{\mathbf{U}}$, the row-wise normalized version of \mathbf{U} : $\tilde{u}_{ij} = \frac{u_{ij}}{\|\mathbf{U}_i\|_2}$

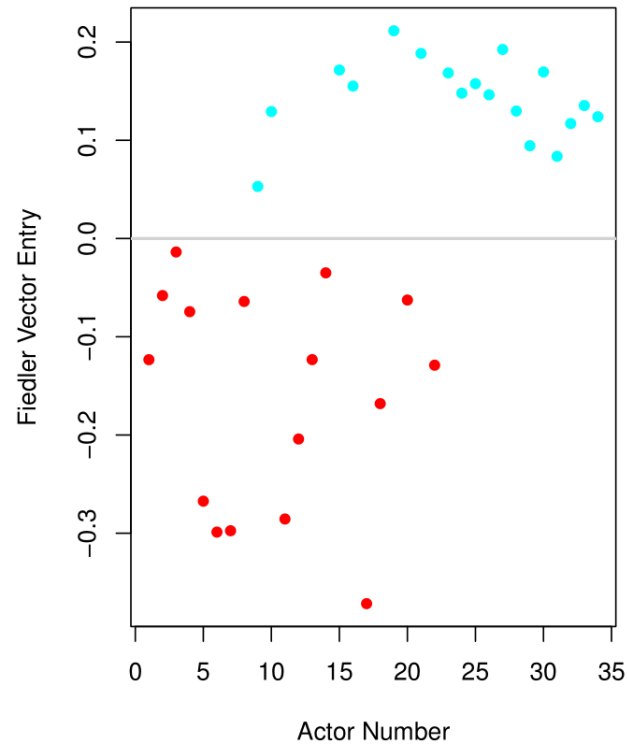
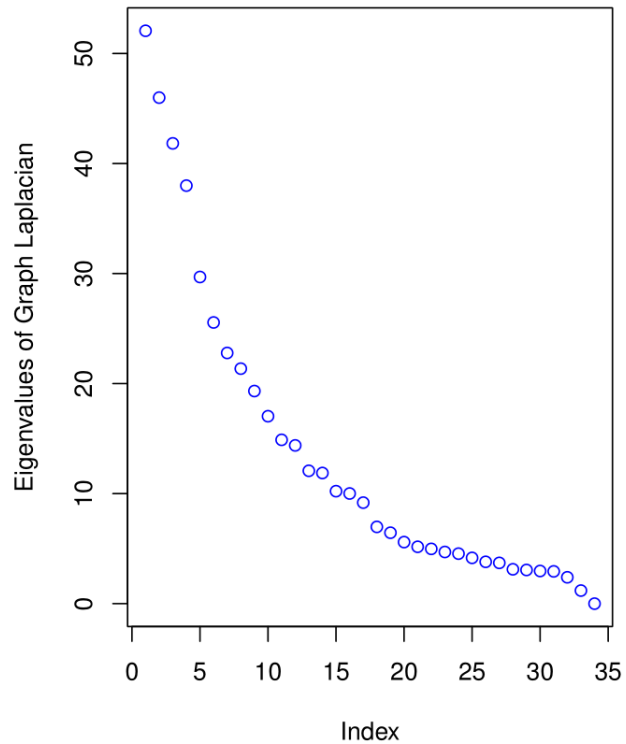
Apply k-means to $(\tilde{\mathbf{U}}_i)_{i=1,\dots,p}$

Output: vector of classes $\mathbf{C} \in \mathcal{Q}^p$, such as $C_i = q$ if $i \in q$

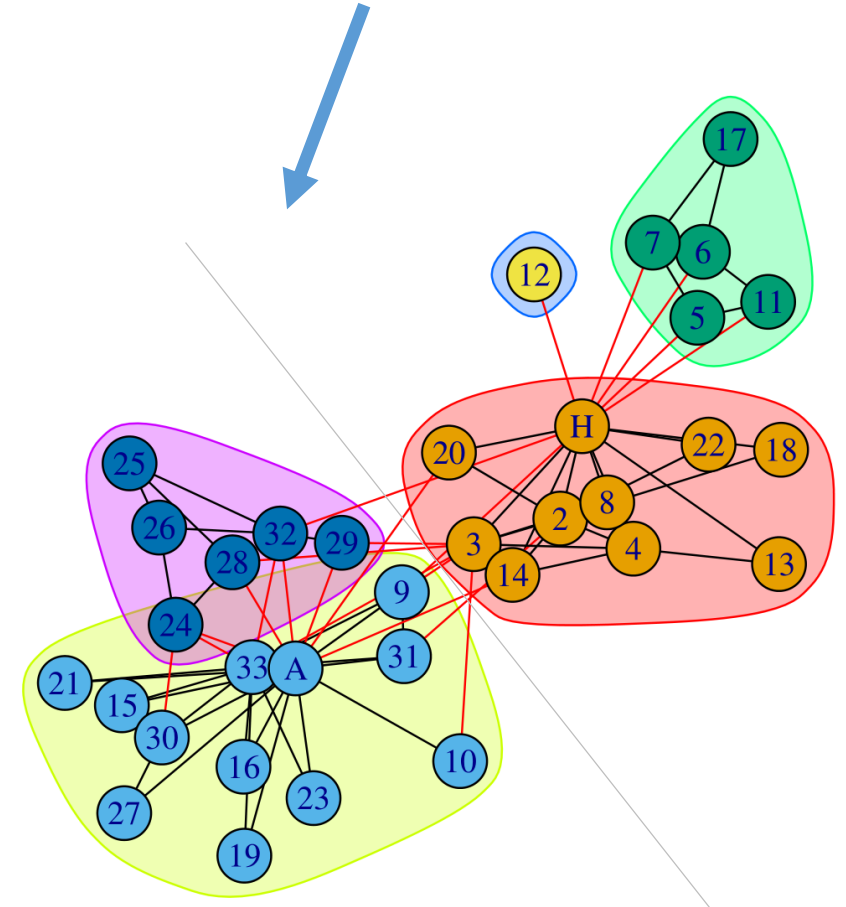
Algorithm 2: Spectral Clustering by Ng, Jordan and Weiss (2002)

Spectral clustering

Example on Zachary's karate club



Split decided according to the **Fiedler vector**: the eigenvector corresp. to the **2nd smallest eigenvalue**



Possible projects

Possible projects @ Centre Borelli

- Graph degeneration
- Online stream k-diversification
- Community detection
- Outlier detection for graph signals
- Graph representation learning
- Graph-theoretic pooling for Graph Neural Networks
- Electricity load forecasting

Discussion

Q & A

