CentraleSupélec

ESSEC BUSINESS SCHOOL

Network Science Analytics

Option Applied Math and M.Sc. in DSBA

Lecture 4

Graph Clustering and Community Detection

Fragkiskos Malliaros

Friday, February 9, 2018

Acknowledgements

- The lecture is partially based on material by
  - Jure Leskovec, Stanford University
  - Aaron Clauset, CU Boulder
  - Manos Papaggelis, York University
  - Gonzalo Mateos, University of Rochester
  - Albert-László Barabási, Northeastern University
  - Christos Faloutsos, CMU
  - Danai Koutra, University of Michigan
  - R. Zafarani, M. A. Abbasi, and H. Liu, Social Media Mining: An Introduction, Cambridge University Press, 2014. Free book and slides at http://socialmediamining.info/

Thank you!

Real graphs are not random

They have interesting properties that deviate from randomness

Properties That We've Seen So Far

- Degree distribution
- Small-world phenomenon
- High clustering coefficient
- Triangle power-law
- Eigenvalue power-law

Static graphs

- Densification power-law
- Shrinking diameter

Time-evolving graphs

and many more …

In this lecture:
- Community structure
- Modularity-based community detection algorithms
- Spectral clustering algorithm
- Observation about the community structure

Communities

- The notion of community structure captures the tendency of nodes to be organized into modules (communities, clusters, groups)
  - Members within a community are more similar among each other
- Typically, the communities in graphs correspond to densely connected entities (nodes)
- Set of nodes with more/better/stronger connections between its members, than to the rest of the network

- Why is this happening?
  - Individuals are typically organized into social groups (e.g., family, associations, profession)
  - Web pages can form groups according to their topic
  - …

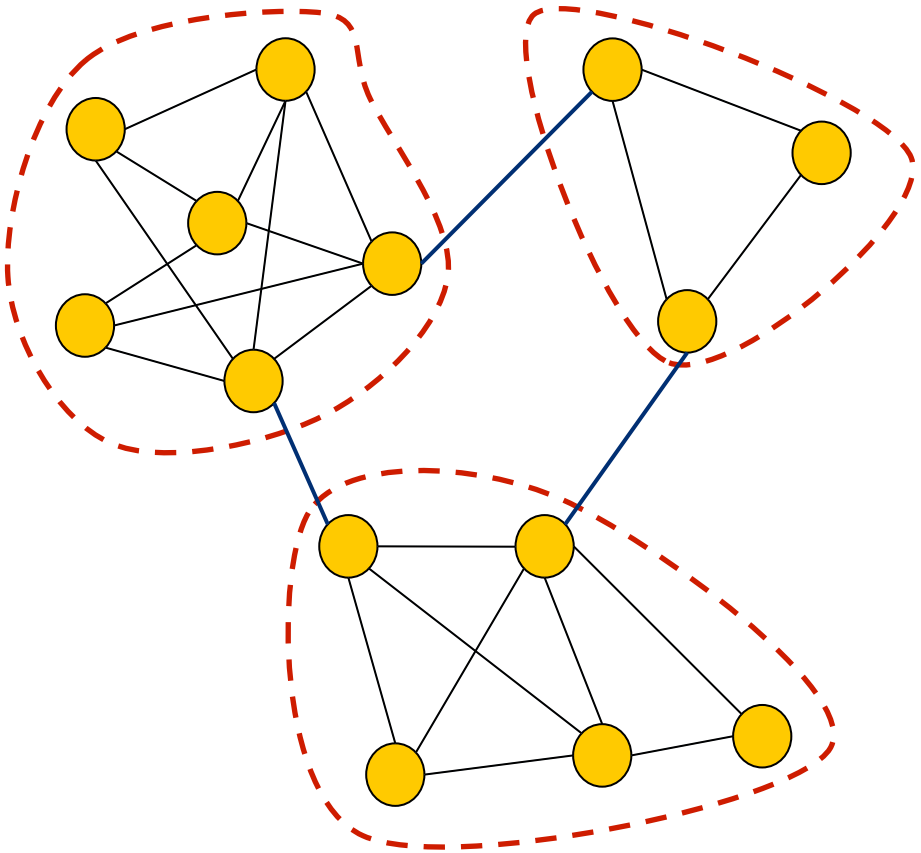
Definition of Communities

- How a community in graphs looks like?
- The property of community structure is difficult to be defined
  - There is no universal definition of the problem
  - It depends heavily on the application domain and the properties of the graph under consideration
- Most widely used notion/definition of communities is based on the number of edges within a group (density) compared to the number of edges between different groups

A community corresponds to a group of nodes with more intra-cluster edges than inter-clusters edges

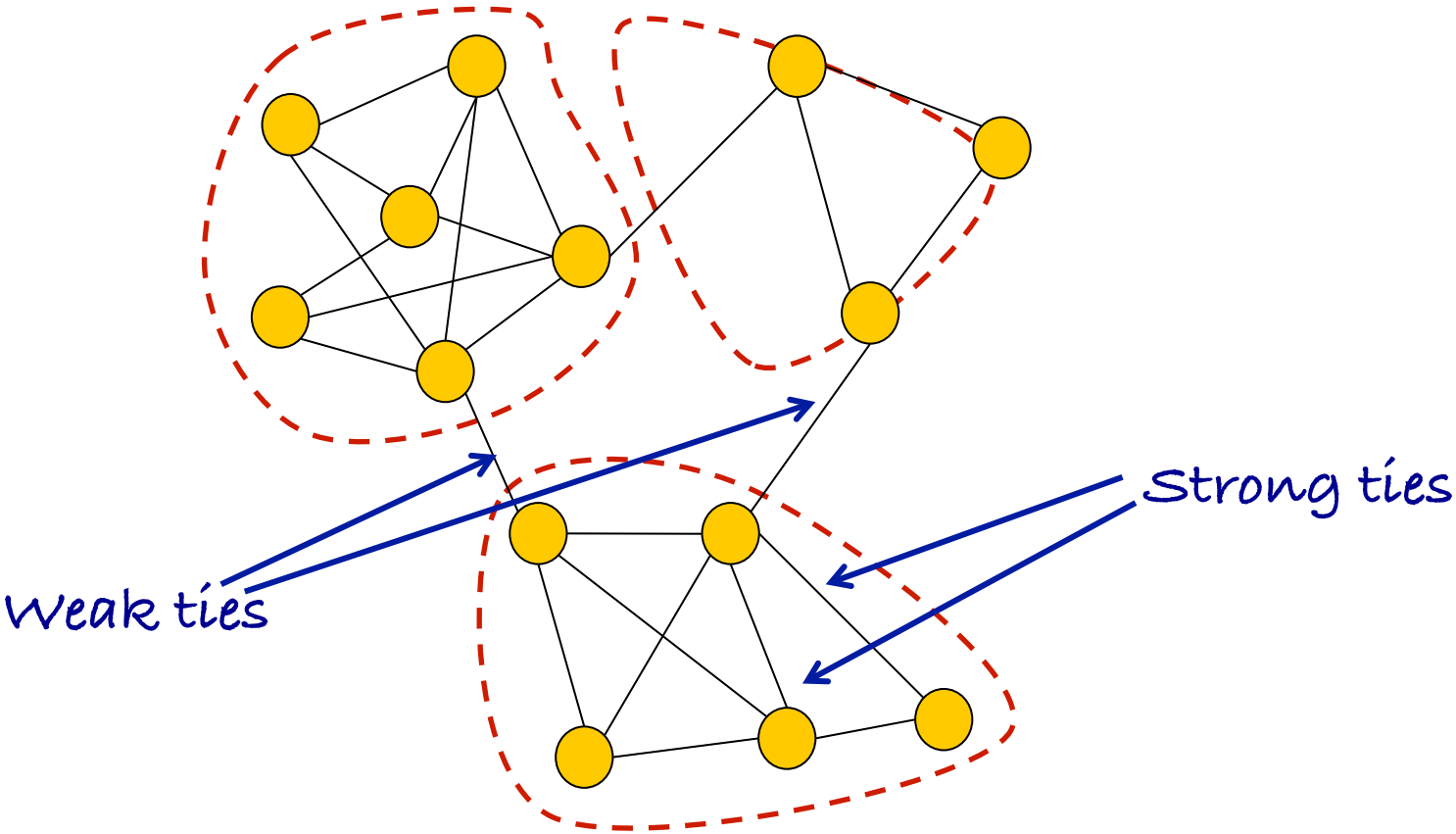
[Newman '03], [Newman and Girvan '04], [Schaeffer '07], [Fortunato '10], [Danon et al. '05], [Coscia et al. 11]

Community Structure



What leads to such a conceptual picture?

Community Structure

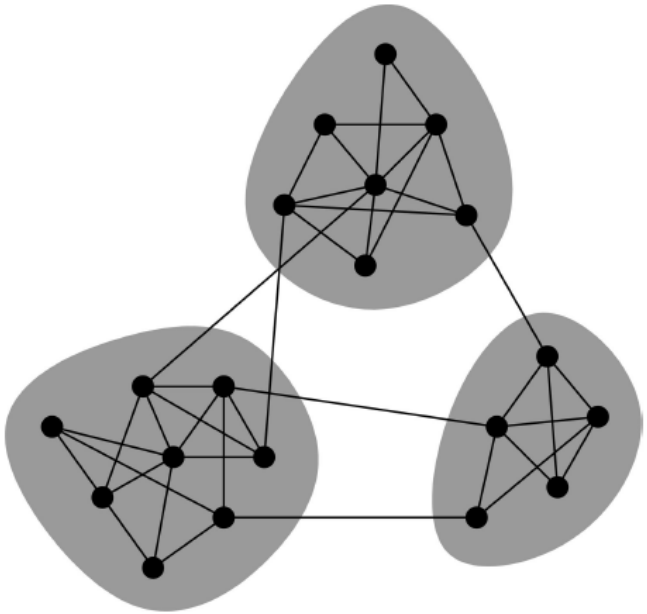


Network Communities

Network Communities

- Granovetter's theory suggests that networks are composed by tightly connected sets of nodes

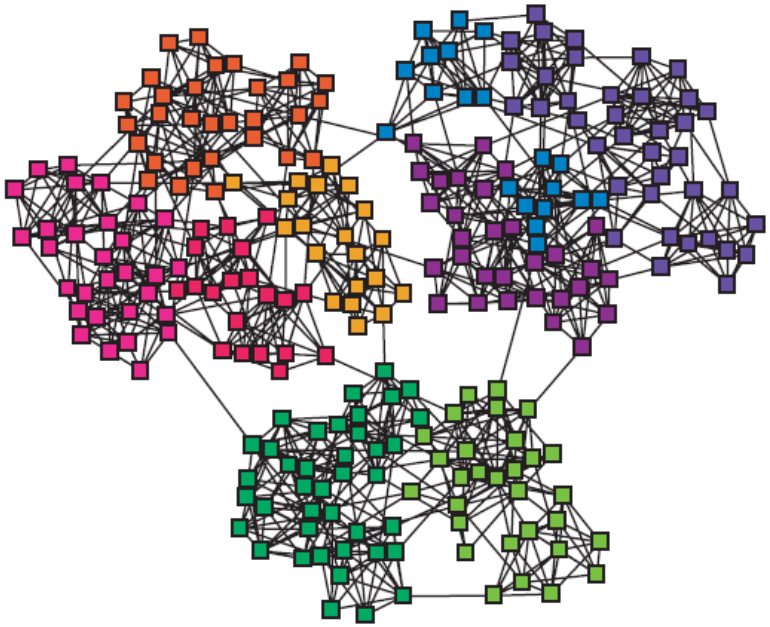
Communities, clusters, groups, modules



- Network communities
  - Sets of nodes with **lots** of connections **inside** and **few** to **outside** (the rest of the network)
  - Intra-community vs. inter-community density

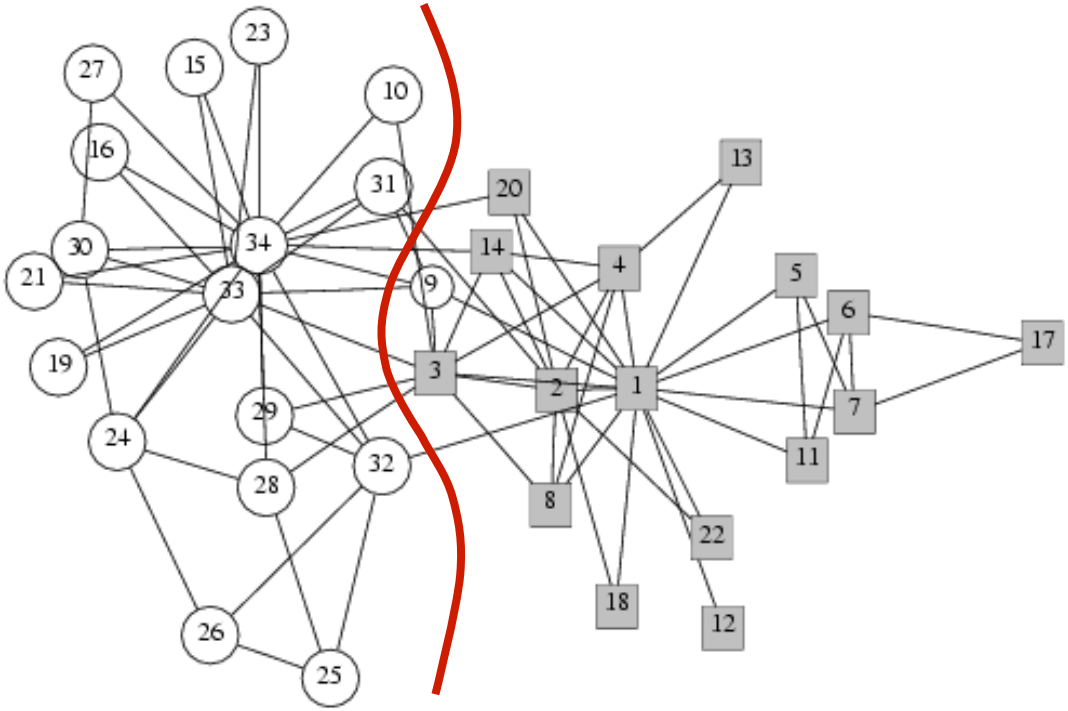
Finding Network Communities

- How to automatically find such densely connected groups of nodes?
- Ideally such automatically detected clusters would then correspond to real groups



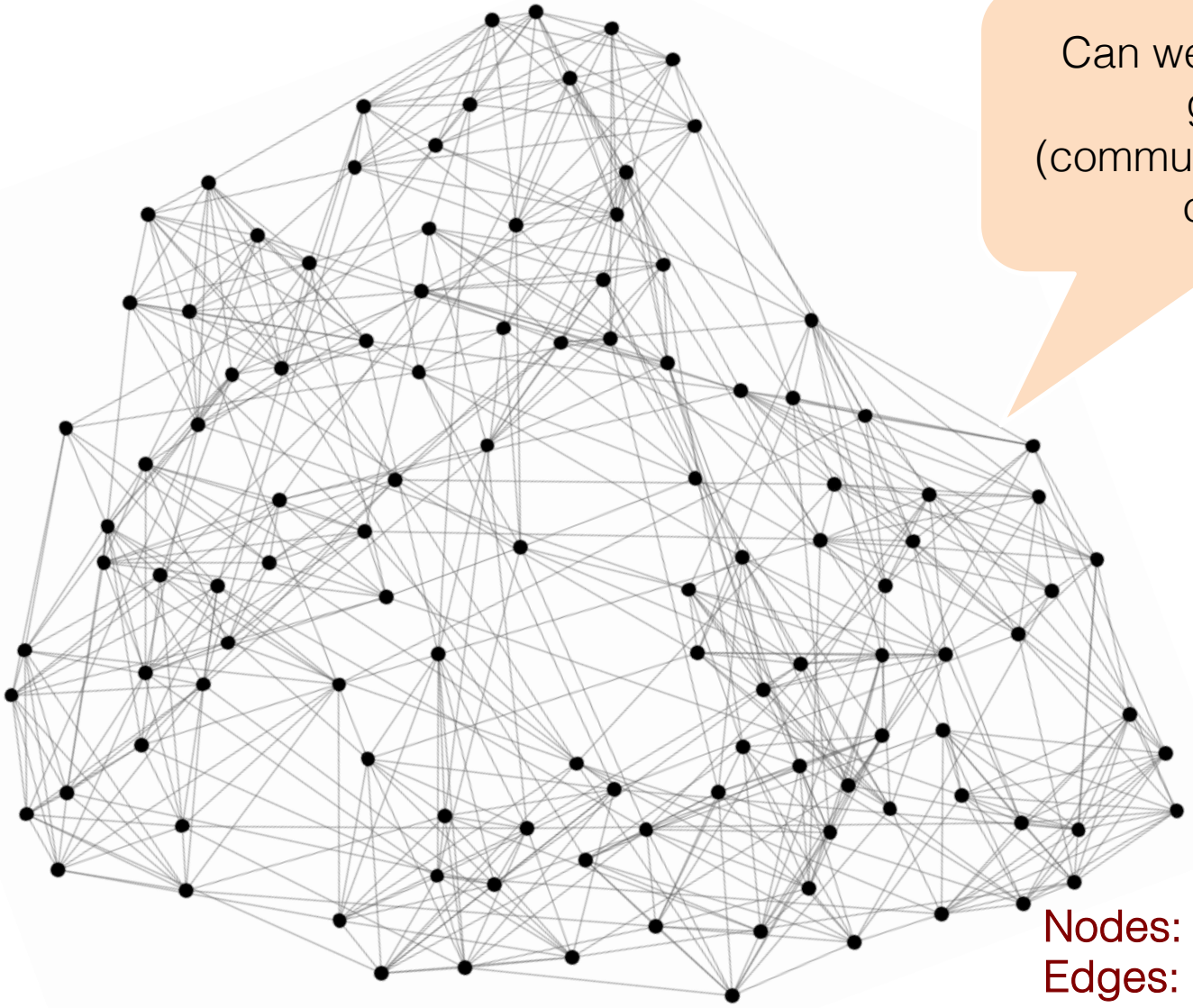
Communities, clusters, groups, modules

Social Network Data



- Zachary's Karate club network
  - Observe social ties and rivalries in a university karate club
  - During his observation, conflicts led the group to split
  - Split could be explained by a minimum cut in the network

NCAA Football Network (1/2)

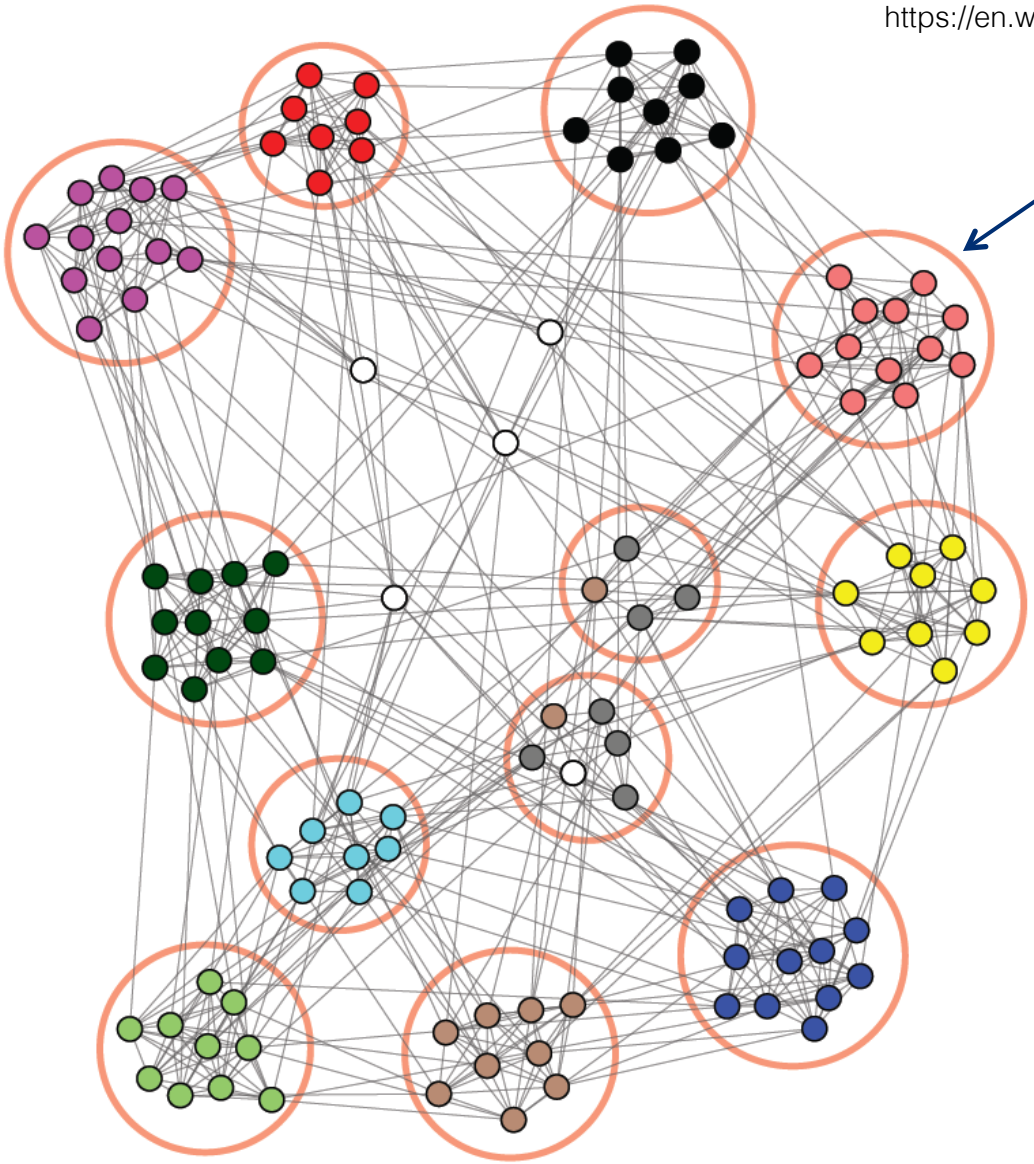


Can we identify node groups? (communities, modules, clusters)

Nodes: Teams
Edges: Games played

NCAA Football Network (2/2)

https://en.wikipedia.org/wiki/List_of_NCAA_conferences

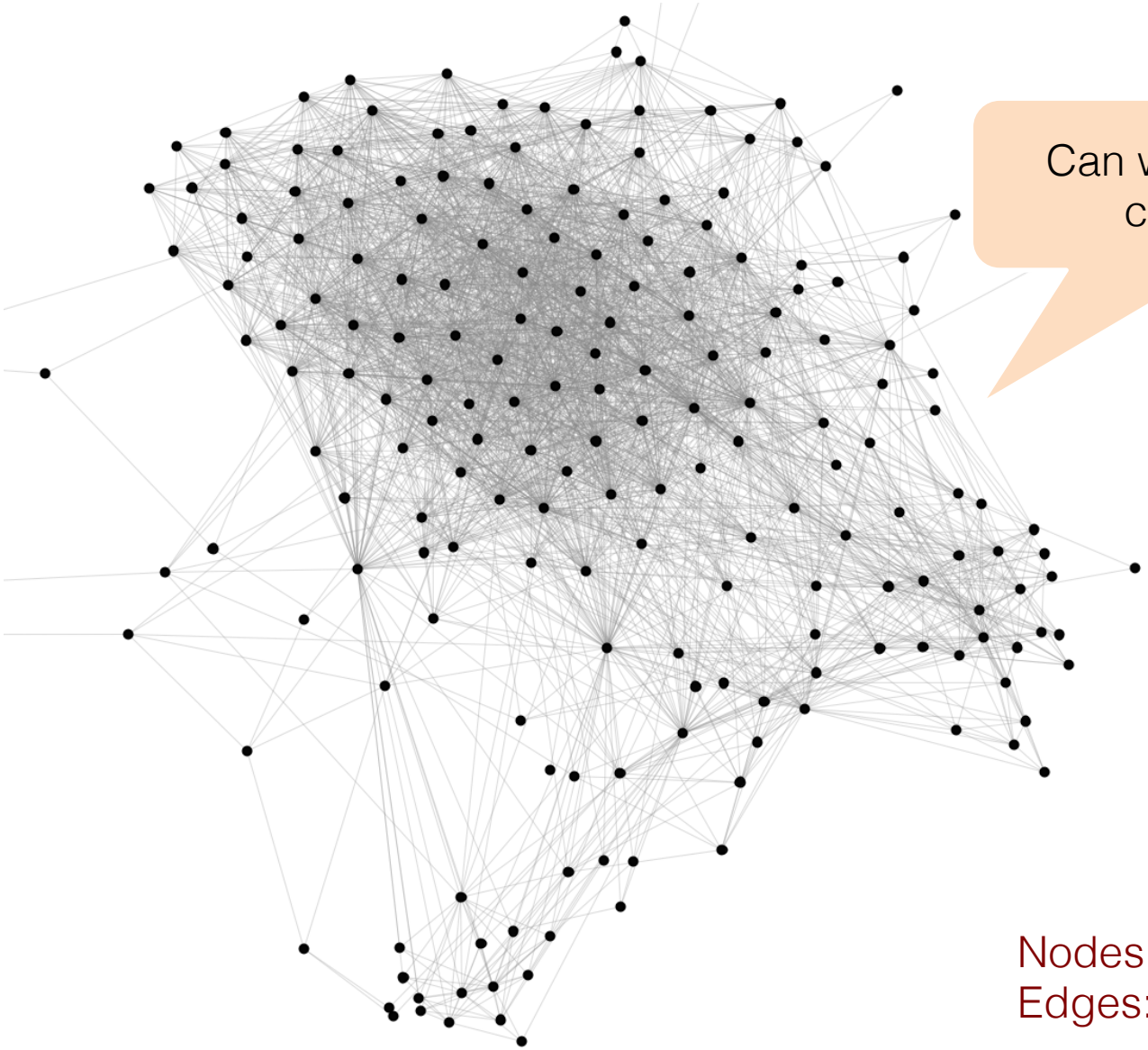


NCAA conferences

- Mid American
- Big East
- Atlantic Coast
- SEC
- Conference USA
- Big 12
- Western Athletic
- Pacific 10
- Mountain West
- Big 10
- Sun Belt
- Independents

Nodes: Teams
Edges: Games played

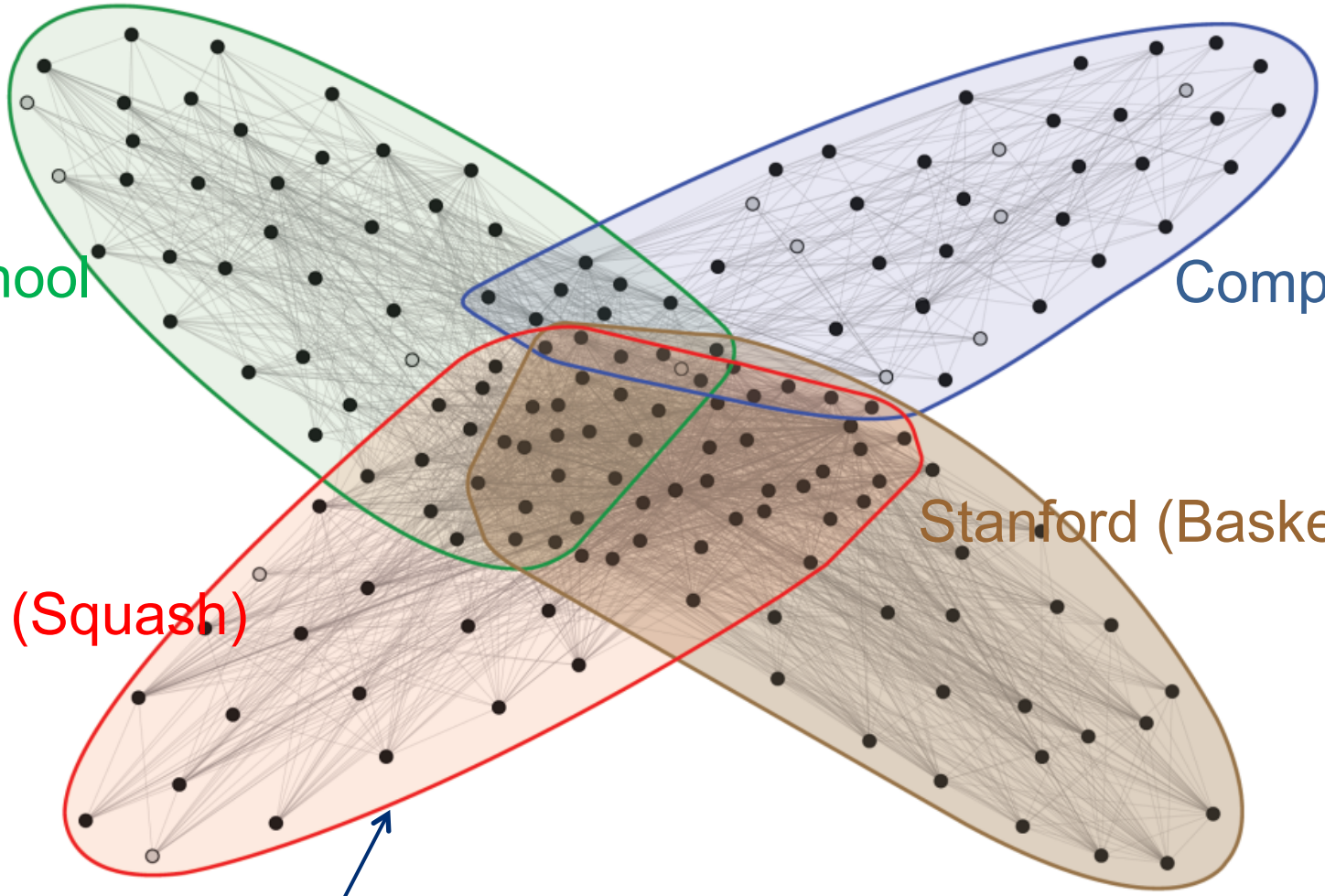
Facebook Ego-network



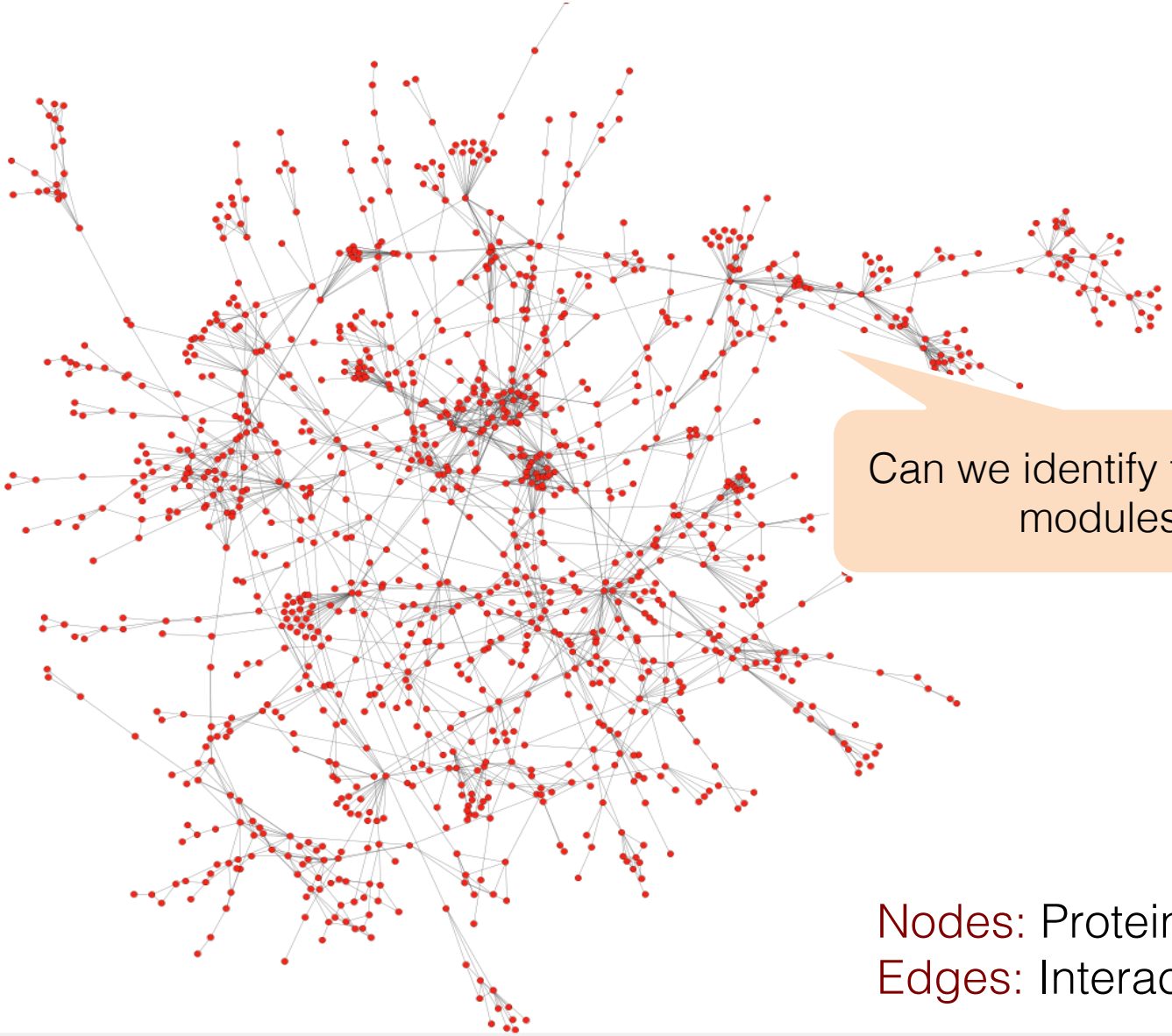
Can we identify social communities?

Nodes: Users
Edges: Friendships

Facebook Ego-network



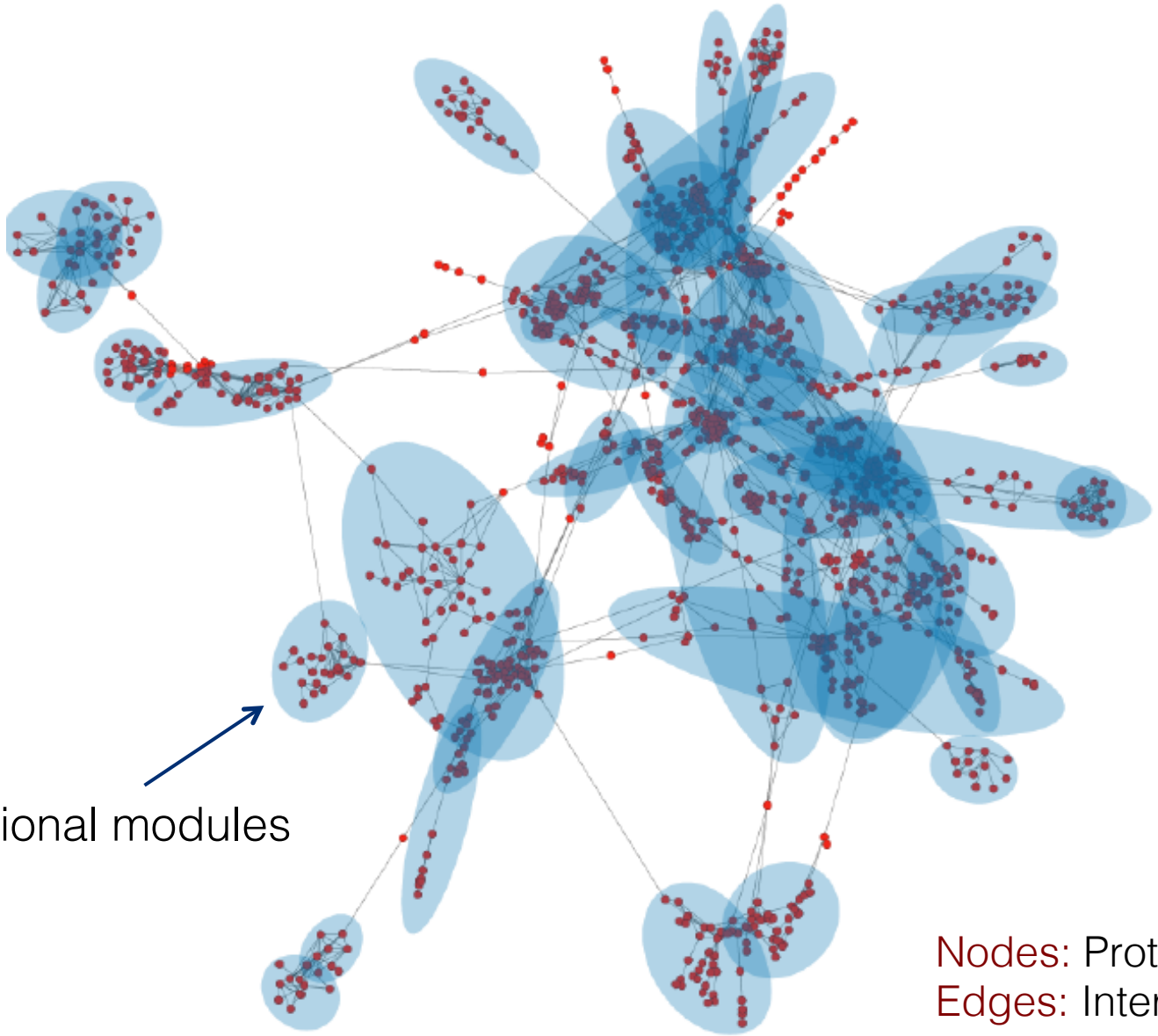
Protein-Protein Interactions



Can we identify functional modules?

Nodes: Proteins
Edges: Interactions

Protein-Protein Interactions



Functional modules

Nodes: Proteins
Edges: Interactions

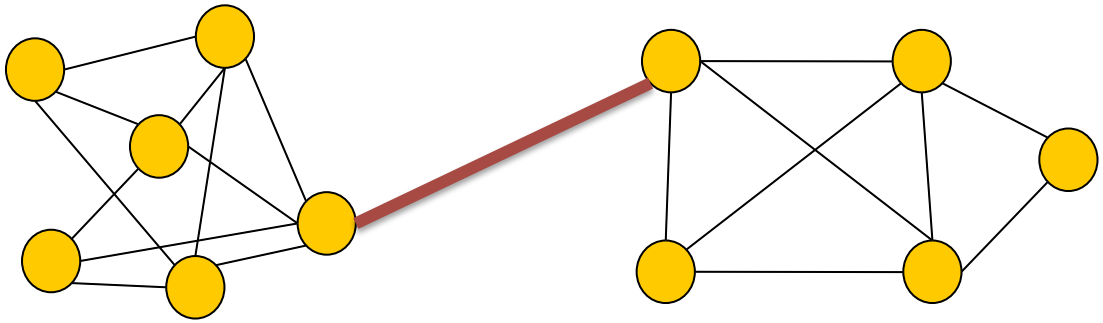
Community Detection

How to find communities?

Girvan-Newman's Method

- Divisive hierarchical clustering based on the notion of edge betweenness centrality
  - Number of shortest paths passing through the edge

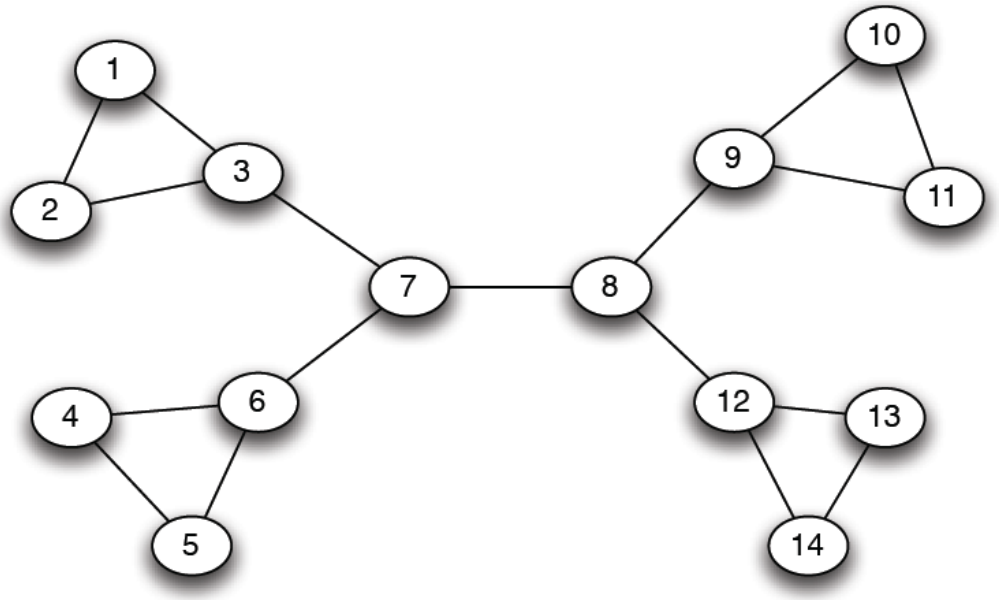
- Algorithm
  1. Calculate the betweenness centrality of all edges in the graph
  2. Remove the edge with the highest betweenness score
  3. Recalculate betweenness for all edges affected by the removal
  4. Repeat step 2 until no edges remain



Try to identify the edges of the graph that are most between other vertices
- Responsible for connecting many node pairs

[Girvan and Newman '02]

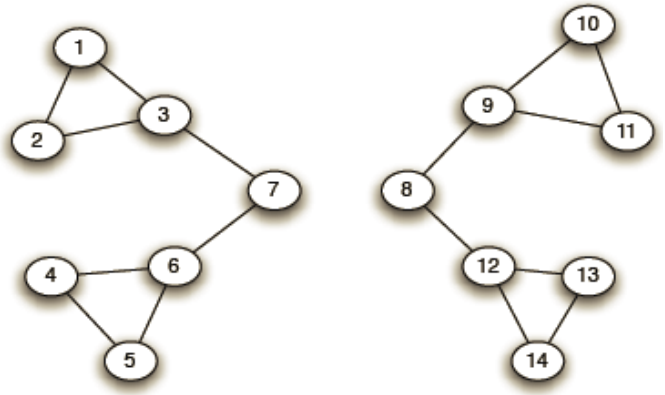
Girvan-Newman Algorithm - Example



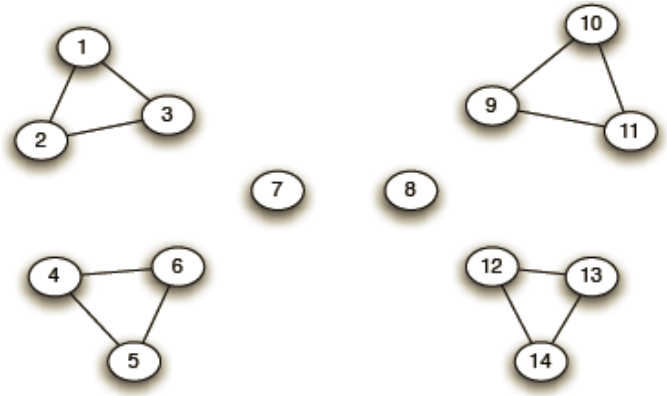
Edge with the highest betweenness centrality score?

Girvan-Newman Algorithm - Example

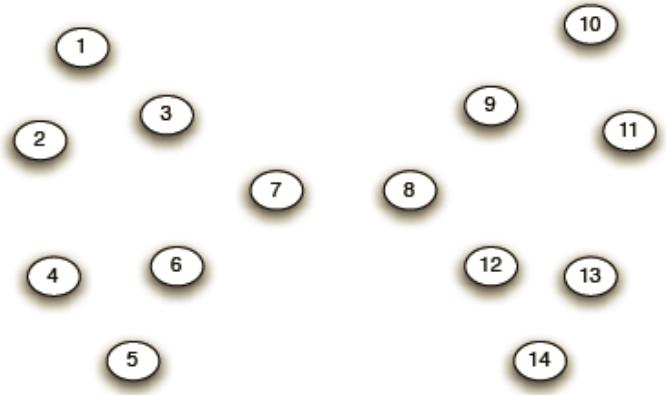
Step 1:



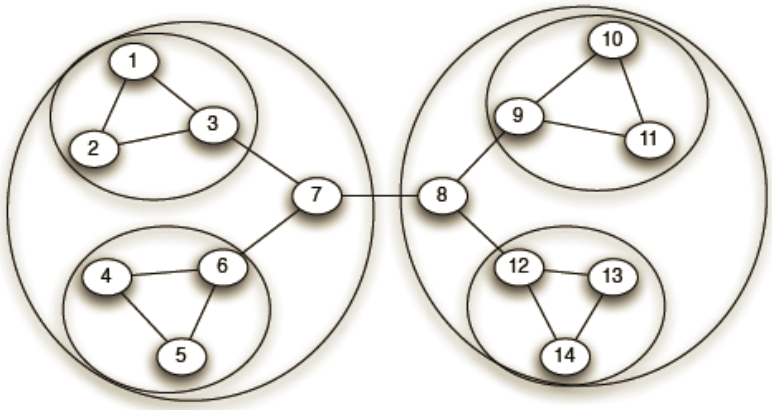
Step 2:



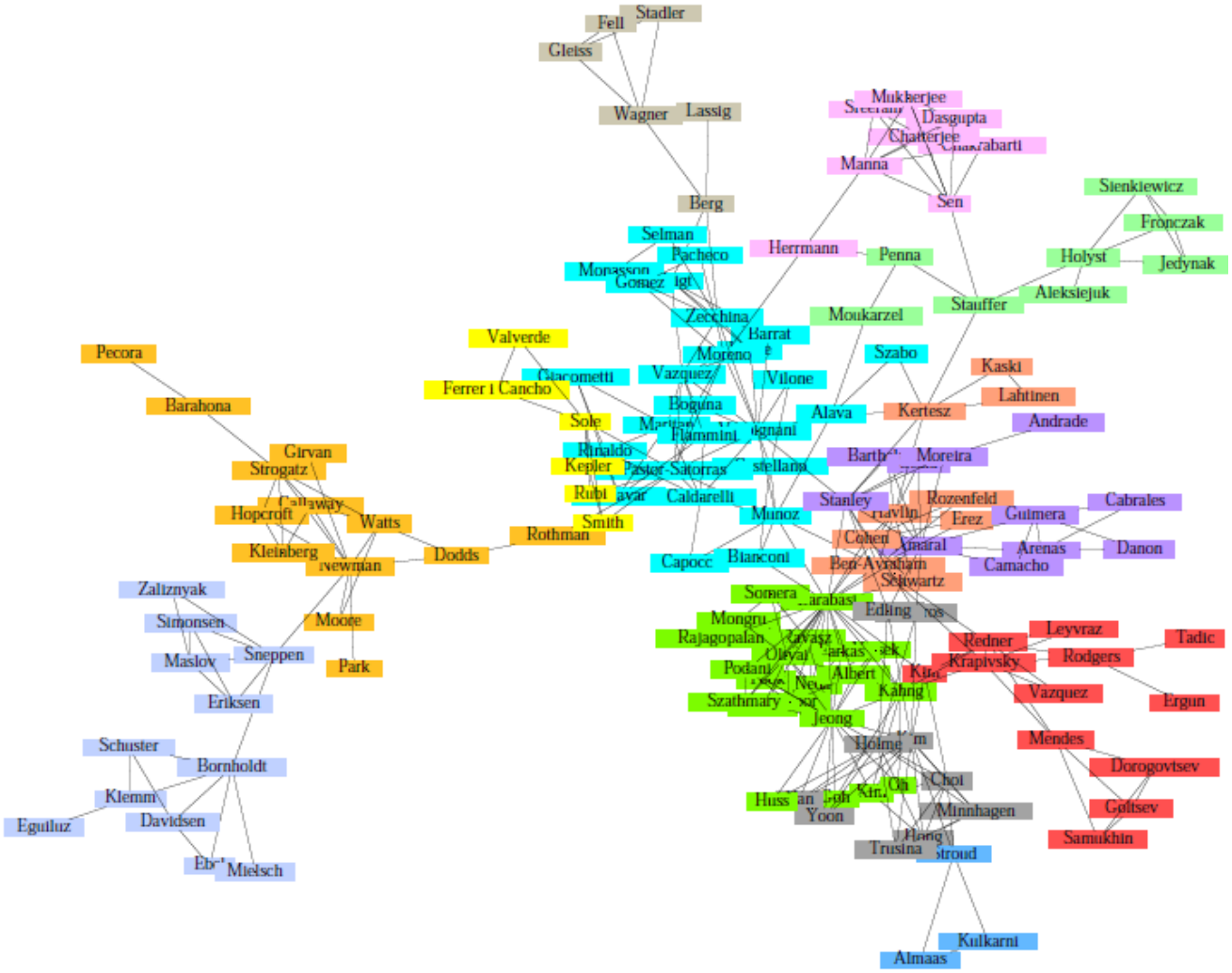
Step 3:



Hierarchical network decomposition:

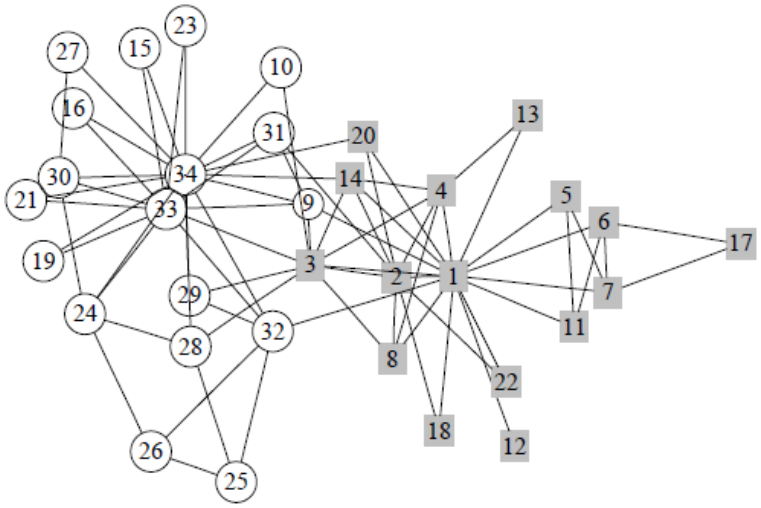


Scientific Collaboration Network

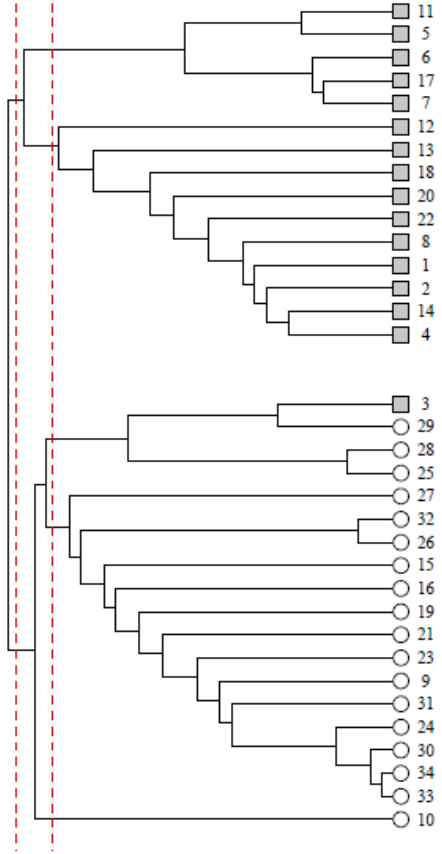


Communities in physics collaborations

Zachary's Karate Club



Zachary's karate club



Dendrogram

- Which of the divisions is the most useful (or optimal)?
  - Need to define metrics of quality of the community structure

How to Evaluate the Quality of Communities

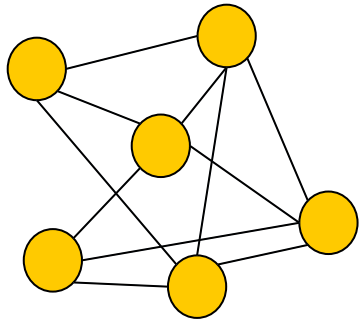
- Typically, two criteria
  - Internal connectivity (intra-community edges)
  - External connectivity (inter-community edges)

- Q: Is there any other way to distinguish groups of nodes with good community structure?

- Random graphs are not expected to present inherent community structure
- Idea: Compare the number of edges that lie within a community to the expected one in case of random graphs with the same degree distribution
  - Modularity measure

Modularity: Main Idea

- Modularity function Q
- Initially introduced as a measure for assessing the strength of communities
  - $Q = \sum_{c \in C}$ (number of edges within community c) – (expected number of edges within community c)
- What is the expected number of edges?
- Consider a configuration model
  - Random graph model with the same degree distribution
  - Let P_{ij} = probability of an edge between nodes i and j with degrees k_i and k_j respectively
  - Then $P_{ij} = k_i k_j / 2m$, where $m = |E| = \frac{1}{2} \sum_i k_i$



[Newman and Girvan '04], [Newman '06]

Formal Definition of Modularity

- Modularity Q

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

where

- A is the adjacency matrix
- k_i, k_j are the degrees of nodes i and j respectively
- m is the number of edges in the graph
- C_i is the community of node i
- δ(.) is the Kronecker function: 1 if both nodes i and j belong to the same community ($C_i = C_j$), 0 otherwise

[Newman and Girvan '04], [Newman '06]

Properties of Modularity

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

- Larger modularity Q indicates better communities (more than random intra-cluster density)
  - The community structure is better if the number of internal edges exceed the expected number
  - Q in the range of 0.3 - 0.7 means significant community structure
- Modularity value is always $-1 < Q < 1$
- It can also take negative values
  - E.g., if each node is a community itself
  - No partitions with positive modularity → No community structure
  - Partitions with large negative modularity → Existence of subgraphs with small internal number of edges and large number of inter-community edges

[Newman and Girvan '04], [Newman '06], [Fortunato '10]

Back to the Girvan-Newman Algorithm

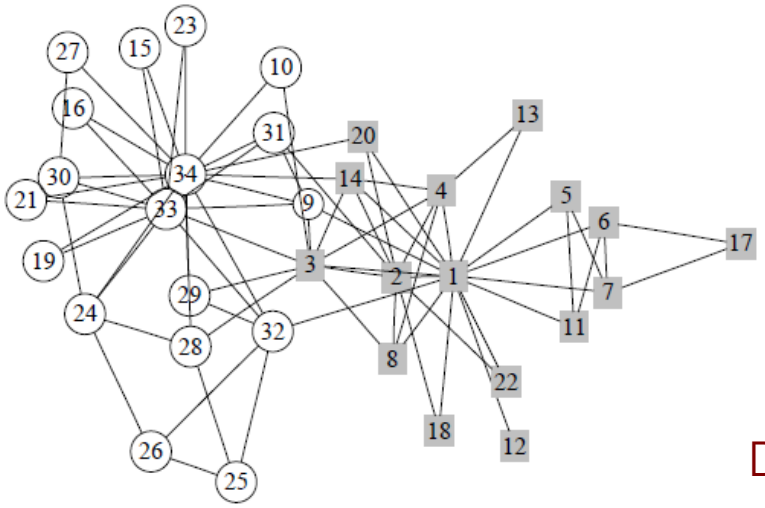
- Basic steps:
  1. Compute betweenness centrality for all edges in the graph
  2. Find and remove the edge with the highest score
  3. Recalculate betweenness centrality score for the remaining edges
  4. Go to step 2

- How do we know if the produced communities are of good quality in order to stop the algorithm?
  - The output of the algorithm is in the form of a dendrogram
  - Use modularity as a criterion to cut the dendrogram and terminate the algorithm (Q ~= 0.3-0.7 indicates good partitions)

- Complexity: $O(m^2n)$ (or $O(n^3)$ in sparse graphs)

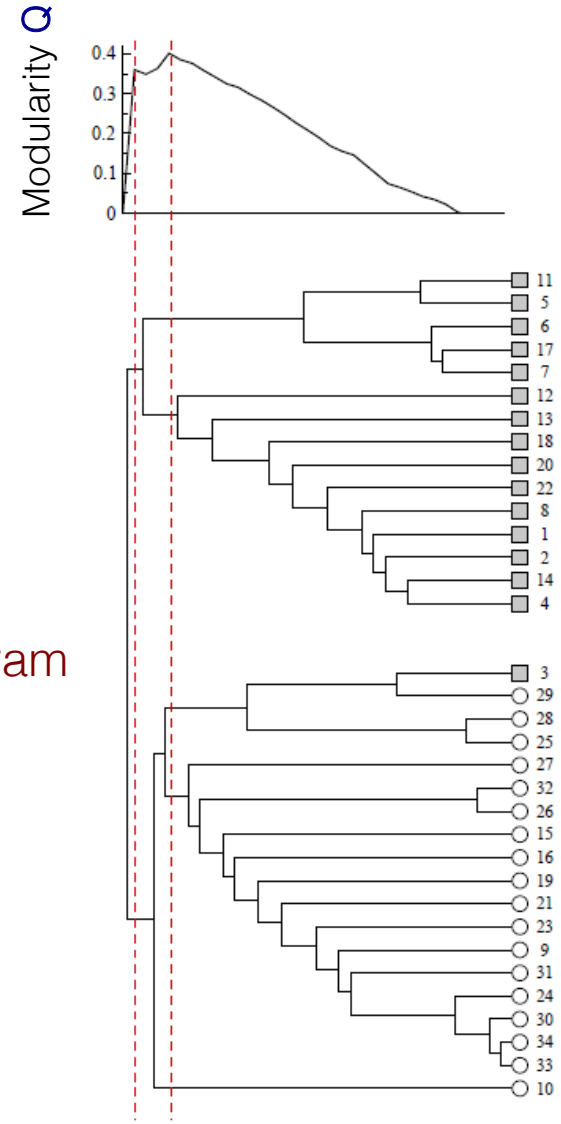
[Newman and Girvan '04], [Girvan and Newman '02]

Zachary's Karate Club – Modularity



Zachary's karate club

Dendrogram



Why not optimize modularity directly?

Applications of Modularity

- Modularity can be applied:
  - As quality function in clustering algorithms
  - As evaluation measure for comparison of different partitions or algorithms
  - As criterion for reducing the size of a graph
    - Size reduction preserving modularity [Arenas et al. '07]
  - As a community detection algorithm itself
    - Modularity optimization

Modularity Optimization

Modularity Optimization

- High values of modularity indicate good quality of partitions
- Goal: find the partition that corresponds to the maximum value of modularity
- Modularity maximization problem
  - Computational difficult problem [Brandes et al. '06]
  - Approximation techniques and heuristics
- Four main categories of techniques
  1. Greedy techniques
  2. Spectral optimization
  3. Simulated annealing
  4. Extremal optimization

[Fortunato '10]

Greedy Techniques (1/2)

- Newman's algorithm [Newman '04]
  - Agglomerative (bottom-up) hierarchical clustering algorithm
  - Idea: Repeatedly join pairs of communities that achieve the greatest increase of modularity (dendrogram representation)

1. Initially, each node of the graph belongs to its own cluster (n)
2. Repeatedly, join communities in pairs by adding edges
   a. At each step, choose the pairs that achieve the greatest increase (or minimum decrease) of modularity
   b. Consider only pairs of communities between which there exist edges (merging communities that do not share edges, it can never improve modularity)

- Complexity: $O((m+n)\,n)$ (or $O(n^2)$ in sparse graphs)

Greedy Techniques (2/2)

- Can we improve the complexity of Newman's algorithm?
  - Greedy optimization algorithm by Clauset, Newman and Moore
  - Key point: large graphs are sparse
  - Exploit sparsity by using appropriate data structures for sparse graphs (e.g., max-heaps)
    a. A sparse matrix for storing the variations of modularity $\Delta Q_{i,j}$ after joining two communities i, j
    b. A max-heap data structure for the largest element of each row of matrix $\Delta Q_{i,j}$ (fast update time and constant time for finndmax() operation)
  - Complexity: O(m d logn), d is the depth of the dendrogram describing the performed partitions (the community structure)
    - Sparse graphs: m ~ n. Graphs with hierarchical structure: d ~ logn. Therefore, the complexity is $O(n \log^2 n)$ for such graphs

[Clauset et al. '04]

Spectral Optimization (1/3)

- Idea: Spectral techniques for modularity optimization
  - Directly optimize modularity
- Goal: Assign the nodes into two communities, X and Y

- Let s_i, for all $i \in V$, be an indicator variable where $s_i = +1$ if i is assigned to X and $s_i = -1$ if i is assigned to Y

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

$$= \frac{1}{4m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)(s_i s_j + 1)$$

$$\sum_j B_{ij} = 0 \qquad = \frac{1}{4m}\sum_{ij} B_{ij} s_i s_j = \frac{1}{4m} s^T \mathrm{B} s$$

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

Modularity matrix

Find $s \in \{-1, 1\}^n$ that maximizes Q

[Newman '06], [Newman '06b]

Spectral Optimization (2/3)

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

- Vector s can be written as a linear combination of the eigenvectors u_i of the modularity matrix B

$$s = \sum_i a_i u_i \quad \text{where} \quad a_i = u_i^T s$$

Just perform eigenvalue decomposition of B

- Modularity can now expressed as

$$Q = \frac{1}{4m} \sum_i a_i u_i^T B \sum_j a_j u_j^T = \frac{1}{4m} \sum_{i=1}^{n} \left(u_i^T \bullet s \right)^2 \lambda_i$$

where λ_i is the eigenvalue of B corresponding to eigenvector u_i

s in one of the eigenvectors of B

- How to optimize?
  - Pick s that is parallel to u_1
  - Maximize $u_1 s$, i.e., the projection of s along vector u_1

[Newman '06], [Newman '06b]

Spectral Optimization (3/3)

- Spectral modularity optimization algorithm
  1. Consider the eigenvector u_1 of B corresponding to the largest eigenvalue
  2. Assign the nodes of the graph in one of the two communities X ($s_i = +1$) and Y ($s_i = -1$) based on the signs of the corresponding components of the eigenvector

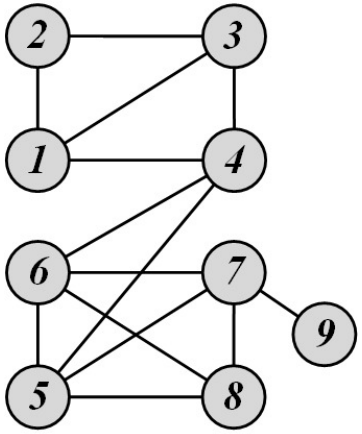
$$s_i = \begin{cases} 1 & if\ u_1(i) \geq 0 \\ -1 & if\ u_1(i) < 0 \end{cases}$$

More than two partitions?

1. Iteratively, divide the produced partitions into two parts
2. If at any step the split does not contribute to the modularity, leave the corresponding subgraph as is
3. End when the entire graph cannot be splitted into no further divisible subgraphs

– Complexity: $O(n^2 \log n)$ for sparse graphs

Modularity Maximization: Example



$$B = \begin{bmatrix} -0.32 & 0.79 & 0.68 & 0.57 & -0.43 & -0.43 & -0.43 & -0.32 & -0.11 \\ 0.79 & -0.14 & 0.79 & -0.29 & -0.29 & -0.29 & -0.29 & -0.21 & -0.07 \\ 0.68 & 0.79 & -0.32 & 0.57 & -0.43 & -0.43 & -0.43 & -0.32 & -0.11 \\ 0.57 & -0.29 & 0.57 & -0.57 & 0.43 & 0.43 & -0.57 & -0.43 & -0.14 \\ -0.43 & -0.29 & -0.43 & 0.43 & -0.57 & 0.43 & 0.43 & 0.57 & -0.14 \\ -0.43 & -0.29 & -0.43 & 0.43 & 0.43 & -0.57 & 0.43 & 0.57 & -0.14 \\ -0.43 & -0.29 & -0.43 & -0.57 & 0.43 & 0.43 & -0.57 & 0.57 & 0.86 \\ -0.32 & -0.21 & -0.32 & -0.43 & 0.57 & 0.57 & 0.57 & -0.32 & -0.11 \\ -0.11 & -0.07 & -0.11 & -0.14 & -0.14 & -0.14 & 0.86 & -0.11 & -0.04 \end{bmatrix}$$

Modularity Matrix

2 eigenvectors

$$\begin{bmatrix} 0.44 & -0.00 \\ 0.38 & 0.23 \\ 0.44 & -0.00 \\ 0.17 & -0.48 \\ -0.29 & -0.32 \\ -0.29 & -0.32 \\ -0.38 & 0.34 \\ -0.34 & -0.08 \\ -0.14 & 0.63 \end{bmatrix}$$

k-means

Two Communities:
{1, 2, 3, 4}
and
{5, 6, 7, 8, 9}

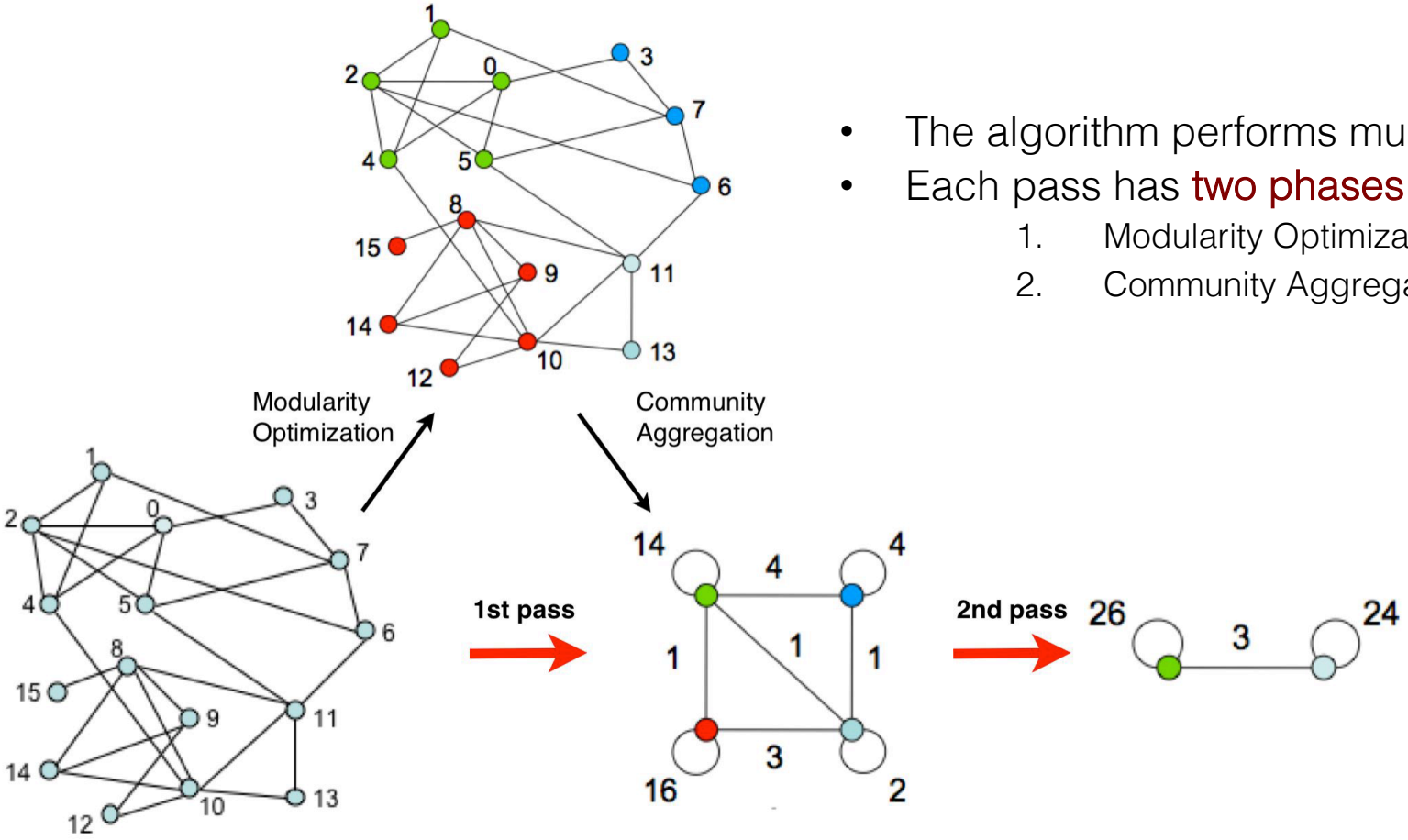
Faster Modularity Optimization?

- The spectral modularity optimization method is slow and does not scale to large networks
- We can perform greedy optimization of modularity to speed-up the process

- Louvain Method
  - A greedy modularity optimization method for community detection
    - Invented when all authors were affiliated with the Université Catholique de Louvain (UCL), Belgium

[Blondel et al. '08]

The Louvain Algorithm (1/3)



- The algorithm performs multiple passes
- Each pass has two phases
  1. Modularity Optimization
  2. Community Aggregation

The Louvain Algorithm (2/3)

Start with a weighted network where all nodes are in their own communities (i.e., n communities)

First phase:

- For each node v_i
  - For all neighbors v_j of v_i
    - Compute the modularity gain if v_i is removed from its community and placed in the community of v_j
  - Find the community with the maximum modularity gain
  - If the maximum gain is positive, remove v_i from its community, and place it in that community
  - If no positive gain, keep the same communities
- Repeat until no node changes its community

Local minima of modularity is achieved

The Louvain Algorithm (3/3)

Second phase:

- Build a new network
  - Nodes are the communities
  - Edges are the edges between nodes in the corresponding communities (weights are sum of the weights)
  - Self-loops represent edges within the community

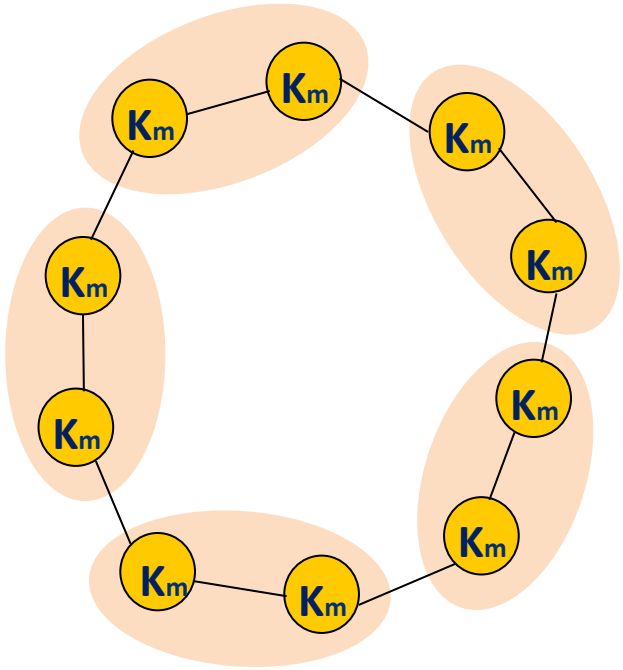
- The algorithm creates hierarchies of communities
- Typically, less than 10 passes are enough
- Complexity: O(nlogn)

Extensions of Modularity

- Modularity has been extended in several directions
  - Weighted graphs [Newman '04]
  - Bipartite graphs [Guimera et al '07]
  - Directed graphs [Arenas et al. '07], [Leicht and Newman '08]
  - Overlapping community detection [Nicosia et al. '09]
  - Modifications in the configuration model – local definition of modularity [Muff et al. '05]

Resolution Limit of Modularity

- Resolution Limit of modularity
- The method of modularity optimization may not detect communities with relatively small size, which depends on the total number of edges in the graph



- K_m are cliques with m edges ($m \leq \sqrt{|E|}$)
- K_m represent well-defined clusters
- However, the maximum modularity corresponds to clusters formed by two or more cliques
- It is difficult to know if the community returned by modularity optimization corresponds to a single community or a union of smaller communities

[Fortunato and Barthelemy '07]

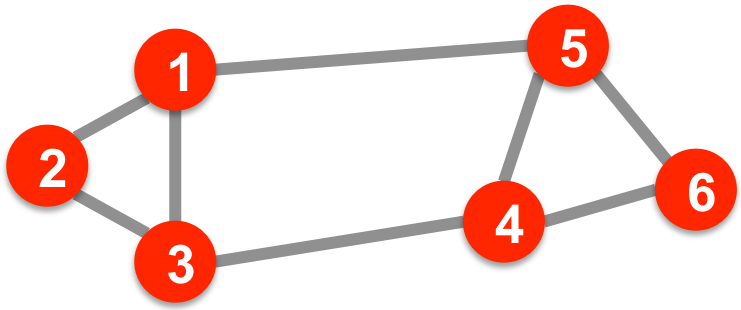
Next Part

- Graph partitioning
- Spectral clustering

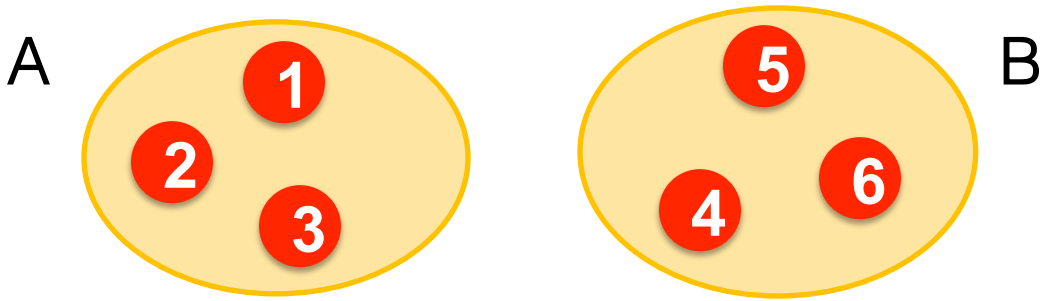
Graph cuts and graph partitioning

Graph Partitioning (1/2)

- Undirected graph *G=(V, E)*



- Bi-partitioning task:
  - Divide nodes into two disjoint groups A, B

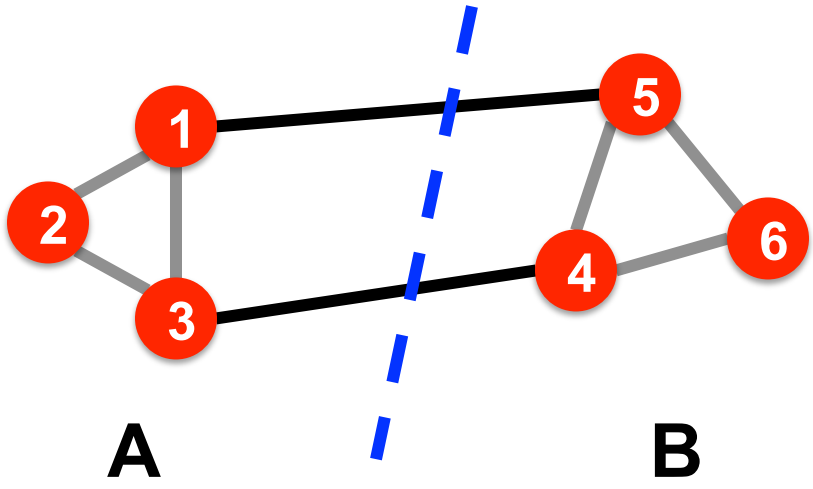


Questions:

- How can we define a good partition of *G* ?
- How can we efficiently identify such a partition?

Graph Partitioning (2/2)

- What makes a good partition?
  - Maximize the number of within-group connections
  - Minimize the number of between-group connections

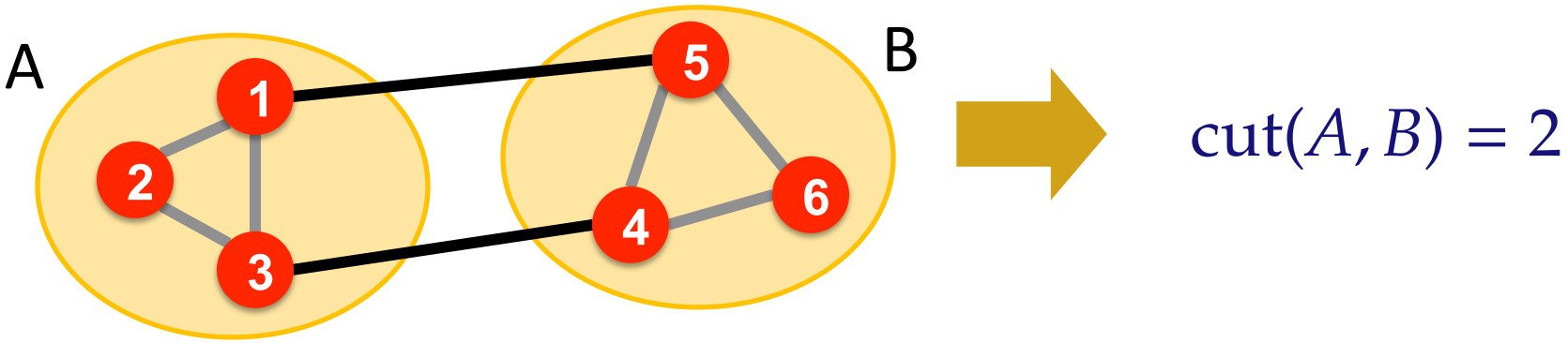


Graph Cuts

- Express partitioning objectives as a function of the edge cut of the partition
- Cut: Set of edges across two groups:

$$\text{cut}(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

Two partitions, A and B



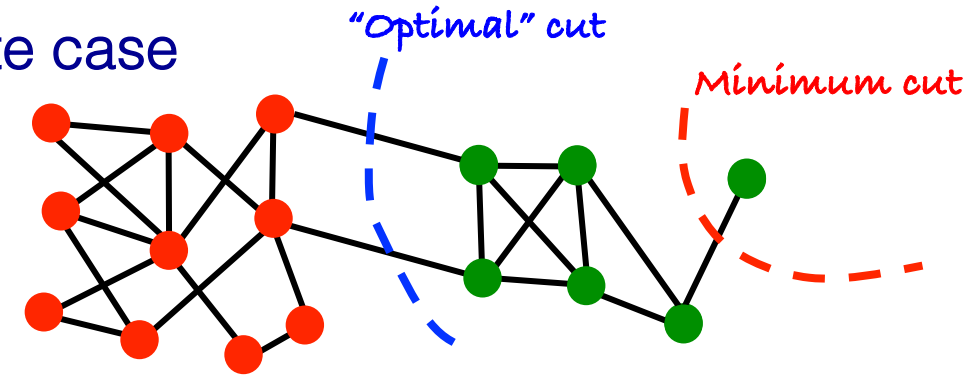
Graph Cut Criterion

- Criterion: Minimum-cut
  - Minimize the weight of connections between groups

$$\arg\min_{A,B} \ \text{cut}(A, B)$$

Can be solved in polynomial time
- Max flow – Min cut

- Degenerate case



Problem
- Not satisfactory partition – often isolated nodes
- Does not consider internal cluster connectivity

Graph Bisection

- Since the minimum cut does not always yield good results we need extra constraints to make the problem meaningful

- Graph Bisection refers to the problem of partitioning the nodes of the graph into two sets of equal size

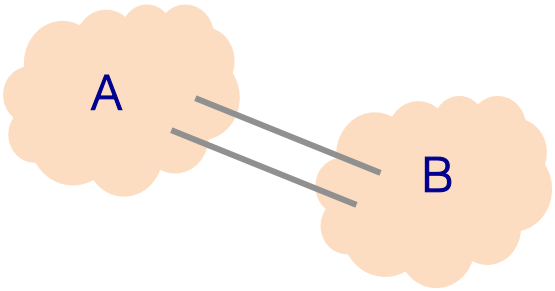
- Kernighan-Lin algorithm
  - Start with random equal partitions
  - Swap nodes to improve some quality metric (e.g., cut, modularity, etc.)
  - Repeat the process; each node can only be moved once

Ratio Cut

Normalize cut by the size of the groups

$$\text{ratio-cut}(A, B) = \frac{\text{cut}(A, B)}{|A|} + \frac{\text{cut}(A, B)}{|B|}$$

Size of A and B



Internal group connectivity is not taken into account

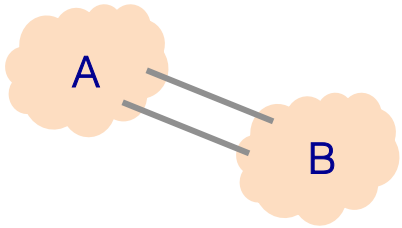
Normalized Cut

- Criterion: Normalized cut
  - Connectivity between groups relative to the density of each group

$$\text{normalized-cut}(A,B) = \frac{\text{cut}(A,B)}{\text{vol (A)}} + \frac{\text{cut}(A,B)}{\text{vol (B)}}$$

  - Vol(A): total weighted degree of the nodes in A, i.e., $\sum_{i \in A} k_i$

- Why use this criterion?
  - It produces more balanced partitions

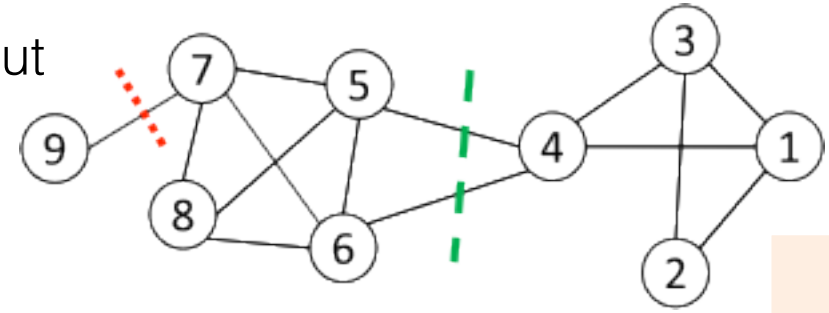


- How do we efficiently find a good partition?
  - Computing the optimal cut is NP-hard

[Shi and Malik '97]

Ratio Cut vs. Normalized Cut (1/2)

Red is Min-Cut



$$\text{ratio-cut}(A,B) = \frac{\text{cut}(A,B)}{|A|} + \frac{\text{cut}(A,B)}{|B|}$$

$$\text{normalized-cut}(A,B) = \frac{\text{cut}(A,B)}{\text{vol}(A)} + \frac{\text{cut}(A,B)}{\text{vol}(B)}$$

Ratio-Cut(Red) = 1/1 + 1/8 = 1.125

Ratio-Cut(Green) = 2/5 + 2/4 = 0.9

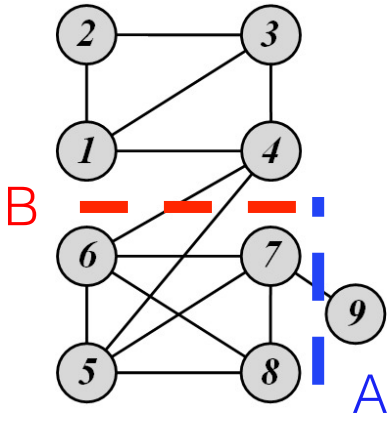
Lower value is better

Normalized-Cut(Red) = 1/1 + 1/26 = 1.03

Normalized-Cut(Green) = 2/12 + 2/16 = 0.29

Normalized is even better for Green due to density

Ratio Cut vs. Normalized Cut (2/2)



$$\text{ratio-cut}(A,B) = \frac{\text{cut}(A,B)}{|A|} + \frac{\text{cut}(A,B)}{|B|}$$

$$\text{normalized-cut}(A,B) = \frac{\text{cut}(A,B)}{\text{vol (A)}} + \frac{\text{cut}(A,B)}{\text{vol (B)}}$$

For Cut A

$$\text{Ratio Cut}(\{1,2,3,4,5,6,7,8\},\{9\}) = \tfrac{1}{2}(\tfrac{1}{1}+\tfrac{1}{8}) = 9/16 = 0.56$$

$$\text{Normalized Cut}(\{1,2,3,4,5,6,7,8\},\{9\}) = \tfrac{1}{2}(\tfrac{1}{1}+\tfrac{1}{27}) = 14/27 = 0.52$$

For Cut B

$$\text{Ratio Cut}(\{1,2,3,4\},\{5,6,7,8,9\}) = \tfrac{1}{2}(\tfrac{2}{4}+\tfrac{2}{5}) = 9/20 = 0.45 < 0.56$$

$$\text{Normalized Cut}(\{1,2,3,4\},\{5,6,7,8,9\}) = \tfrac{1}{2}(\tfrac{2}{12}+\tfrac{2}{16}) = 7/48 = 0.15 < 0.52$$

Both ratio cut and normalized cut prefer a balanced partition

Graph Cuts

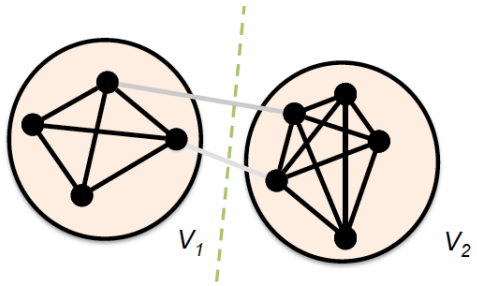
Ratio and normalized cuts can be reformulated using matrices and can be solved using spectral techniques

Spectral clustering
for
graph partitioning

From Graph Cuts to Spectral Partitioning

- For simplicity, let's consider the measure of the cut criterion
  - Recall that a cut C is defined as the number of edges between groups V_1 and V_2

$$C = \text{cut}(V_1, V_2) = \sum_{i \in V_1, j \in V_2} A_{ij}$$



  - The goal is to minimize the cut

- Let's define the binary community membership variables per node

$$s_i = \begin{cases} +1, & \text{if node } i \text{ belongs to } V_1 \\ -1, & \text{if node } i \text{ belongs to } V_2 \end{cases}$$

- Then, we have the following

$$\frac{1}{2}(1 - s_i s_j) = \begin{cases} 1, & \text{if nodes } i \text{ and } j \text{ are in different groups} \\ 0, & \text{if nodes } i \text{ and } j \text{ are in the same group} \end{cases}$$

From Graph Cuts to Spectral Partitioning

- Then, we can express the cut size in terms of s_i and s_j

$$C = \text{cut}(V_1, V_2) = \sum_{i \in V_1, j \in V_2} A_{ij} = \frac{1}{2} \sum_{i,j \in V} A_{i,j}(1 - s_i s_j)$$

Sum over all values of i and j

- The first term in the sum is the following:

$$\sum_{i,j \in V} A_{i,j} = \sum_{i \in V} k_i = \sum_{i \in V} k_i s_i^2 = \sum_{i,j \in V} k_i s_i s_j \delta_{ij}$$

$\sum_j A_{ij} = k_i$

$s_i^2 = 1$

δ: if i=j then 1

- Then, C can be written as

$$C = \frac{1}{2} \sum_{i,j \in V} (k_i \delta_{ij} - A_{ij})\, s_i s_j = \frac{1}{2} \sum_{i,j \in V} L_{ij}\, s_i s_j$$

$\mathbf{L} = \mathbf{D} - \mathbf{A}$

- The Cut can be expressed in terms of the Laplacian matrix L

$$C = \frac{1}{2} \mathbf{s}^T \mathbf{L} \mathbf{s}, \quad \mathbf{s} = (s_1, \ldots, s_n)^T$$

modularity

$Q \propto \mathbf{s}^T \mathbf{B} \mathbf{s}$

Graph Cut Minimization

- Since $|V_1| = n_1$ and $|V_2| = n_2$, then we have the following constraint

$\mathbf{1}^T = [1, 1, \ldots, 1]$

$$\sum_{i \in V} s_i = \sum_{i \in V_1} (+1) + \sum_{i \in V_2} (-1) = n_1 - n_2 \Rightarrow \mathbf{1}^T \mathbf{s} = n_1 - n_2$$

- Then, the minimum cut criterion for graph bisection is the following

$$\hat{\mathbf{s}} = \arg\min_{s \in \{\pm 1\}^{|V|}} \mathbf{s}^T \mathbf{L} \mathbf{s}, \quad \text{s.t. } \mathbf{1}^T \mathbf{s} = n_1 - n_2$$

- The binary constraints $s \in \{\pm 1\}^{|V|}$ make the optimization problem hard
  - Idea: relax the binary constraints
  - A similar idea can be applied to optimize the modularity function

Recall the Properties of the Laplacian

- The Laplacian matrix is positive semi-definite

$$\mathbf{x}^T \mathbf{L} \mathbf{x} \geq 0, \quad \forall \mathbf{x} \in \mathbb{R}^{|V|}$$

- Spectrum: All eigenvalues of L are real and non-negative

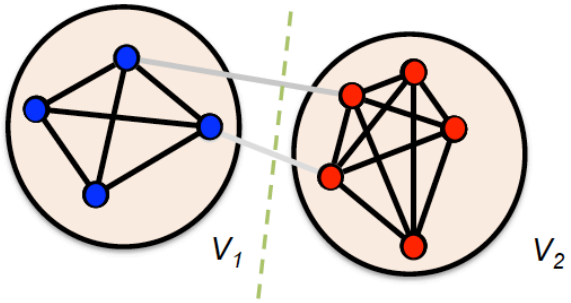
- Spectrum and connectivity
  - The smallest eigenvalue λ_1 of L is zero
  - If the second smallest eigenvalue $\lambda_2 \neq 0$, then G is connected
  - If L has n zero eigenvalues, G has n connected components

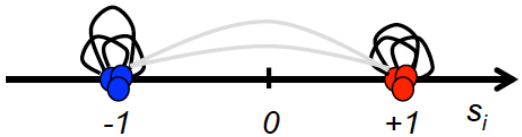
Further Intuition

- Since $\mathbf{s}^T\mathbf{L}\mathbf{s} = \sum_{i,j \in V} L_{ij} s_i s_j = \sum_{(i,j)\in E} (s_i - s_j)^2$, the minimum cut formulation is

$$\hat{\mathbf{s}} = \arg\min_{s \in \{\pm 1\}^{|V|}} \sum_{(i,j)\in E} (s_i - s_j)^2, \quad \text{s.t. } \mathbf{1}^T\mathbf{s} = n_1 - n_2$$

- Does this equivalent cost function make sense? Yes!
  - Edges joining nodes in the same group do not add to the sum
  - Edges joining nodes in different groups add 4 to the sum





Minimize cut:
assign values s_i to nodes i such that few edges cross 0

Minimum Cut Relaxation

- Relax the constraint $s \in \{\pm 1\}^{|V|}$ to $\mathbf{s} \in \mathbb{R}^{|V|}, \quad \| \mathbf{s} \|_2 = 1$

$$\hat{\mathbf{s}} = \arg\min_s \ \mathbf{s}^T \mathbf{L} \mathbf{s}, \ \text{ s.t. } \mathbf{1}^T \mathbf{s} = n_1 - n_2 \text{ and } \mathbf{s}^T \mathbf{s} = 1$$

  - Can be solved using the method of Lagrange multipliers

- Characterization of the solution $\hat{\mathbf{s}}$

$$\hat{\mathbf{s}} = \mathbf{v_2} + \frac{n_1 - n_2}{n} \mathbf{1}$$

  - The second-smallest eigenvector $\mathbf{v_2}$ of L satisfies $\mathbf{1}^T \mathbf{v_2} = 0$
  - Minimum cut is $C(\hat{\mathbf{s}}) = \hat{\mathbf{s}}^T \mathbf{L} \hat{\mathbf{s}} = \mathbf{v}_2^T \mathbf{L} \mathbf{v}_2 \propto \lambda_2$

- If the graph G is disconnected, then we know that $\lambda_2 = 0 = C(\hat{\mathbf{s}})$
  - If G is amenable to bisection, the cut is small and so is λ_2

Theoretical Guarantees

- Consider a partition of G into V_1 and V_2, where $|V_1| \leq |V_2|$
- If G is connected, then the Cheeger inequality guarantees that

$$\frac{\alpha^2}{2k_{max}} \leq \lambda_2 \leq 2\alpha$$

where $\alpha = \frac{C}{|V_1|}$ and k_{max} is the maximum node degree

Then, λ_2 is a useful bound about the size of the minimum cut

Spectral Graph Bisection

- How to obtain the binary cluster labels $s \in \{\pm 1\}^{|V|}$ from $\hat{\mathbf{s}} \in \mathbb{R}^{|V|}$?
  - Maximize the similarity measure $\mathbf{s}^T \hat{\mathbf{s}}$

$$s_i = \text{sign}([\mathbf{v}_2]_i) \begin{cases} +1, & [\mathbf{v}_2]_i > 0 \\ -1, & [\mathbf{v}_2]_i \leq 0 \end{cases}$$

Spectral graph bisection algorithm

1. Compute the Laplacian matrix L with entries $L_{ij} = D_{ij} - A_{ij}$
2. Find the second smallest eigenvector $\mathbf{v}_2$ of L
3. Cluster membership of node i is $s_i = \text{sign}([\mathbf{v}_2]_i)$

- The eigenvector $\mathbf{v}_2$ is known as the Fiedler vector
  - Eigenvalue λ_2 is the Fiedler value or the algebraic connectivity of G

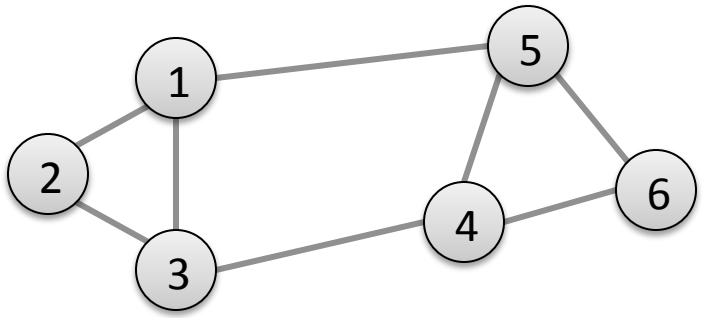
What About the Normalized Case?

- For balanced partitions, we are using the ratio-cut and normalized cut criteria

- Can we perform spectral bisection based on those criteria?
  - Replace the Laplacian matrix with the normalized Laplacian $D^{-1} L$

Adjacency Matrix

- Adjacency matrix A:
  - $n \times n$ matrix
  - $A=[A_{ij}]$, $A_{ij}=1$ if edge between node i and j

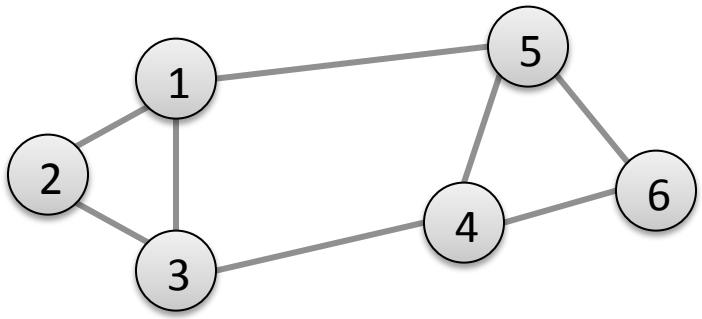


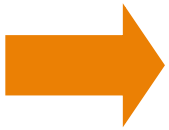
	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0

- Important properties
  - Symmetric matrix
  - Eigenvectors are real and orthogonal

Degree Matrix

- Degree matrix D:
  - $n \times n$ diagonal matrix
  - $D=[D_{ii}]$, D_{ii} = degree of node i

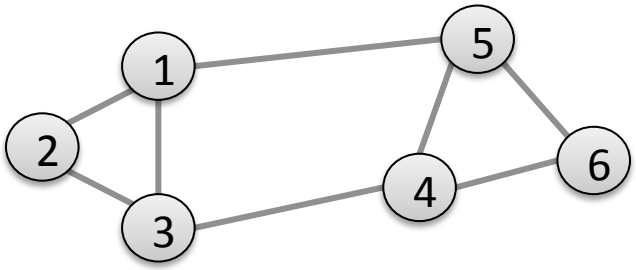




	1	2	3	4	5	6
1	3	0	0	0	0	0
2	0	2	0	0	0	0
3	0	0	3	0	0	0
4	0	0	0	3	0	0
5	0	0	0	0	3	0
6	0	0	0	0	0	2

Laplacian Matrix

- Laplacian matrix L = D - A:
  - n x n symmetric matrix



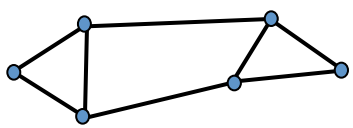
	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

- The Laplacian matrix L has important properties
  - Eigenvalues are non-negative real numbers
  - Eigenvectors are real and orthogonal

Spectral Bisection Algorithm (1/2)

(1) Pre-processing:

- Build Laplacian matrix L of the graph



	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

(2) Decomposition:

- Find eigenvalues λ and eigenvectors v of the matrix L

$\lambda =$

0.0
1.0
3.0
3.0
4.0
5.0

$v =$

0.4	0.3	-0.5	-0.2	-0.4	-0.5
0.4	0.6	0.4	-0.4	0.4	0.0
0.4	0.3	0.1	0.6	-0.4	0.5
0.4	-0.3	0.1	0.6	0.4	-0.5
0.4	-0.3	-0.5	-0.2	0.4	0.5
0.4	-0.6	0.4	-0.4	-0.4	0.0

- Map vertices to corresponding components of λ_2

1	0.3
2	0.6
3	0.3
4	-0.3
5	-0.3
6	-0.6

v_2

How do we now find the clusters?

Spectral Bisection Algorithm (2/2)

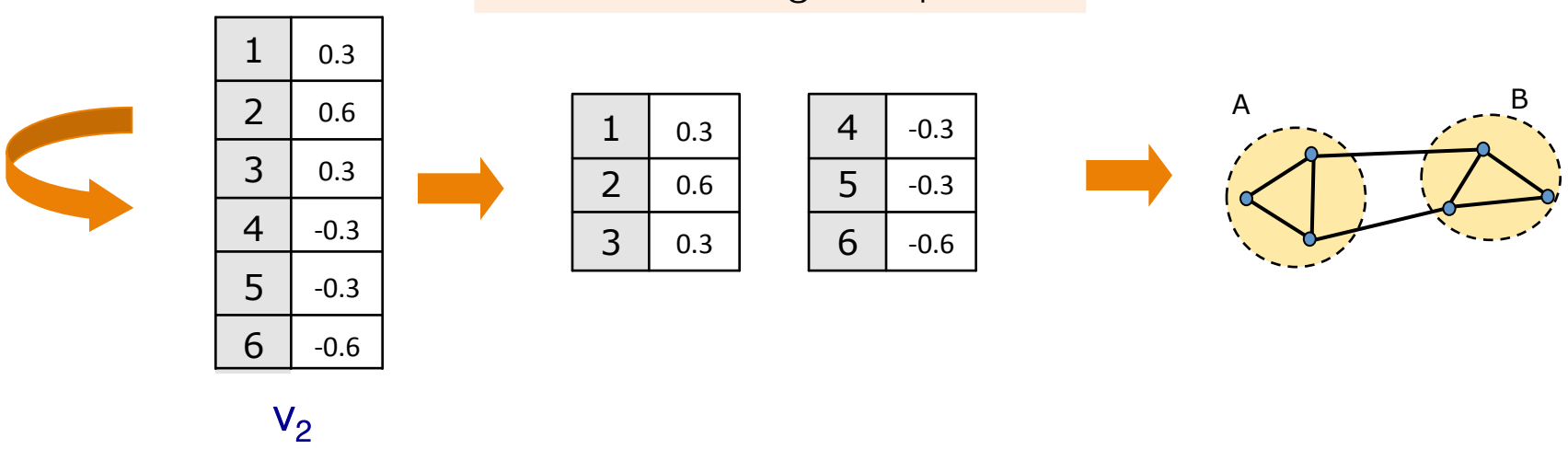
(3) Grouping:

– Assign nodes into one of the two clusters based on the sign of the corresponding component of v_2

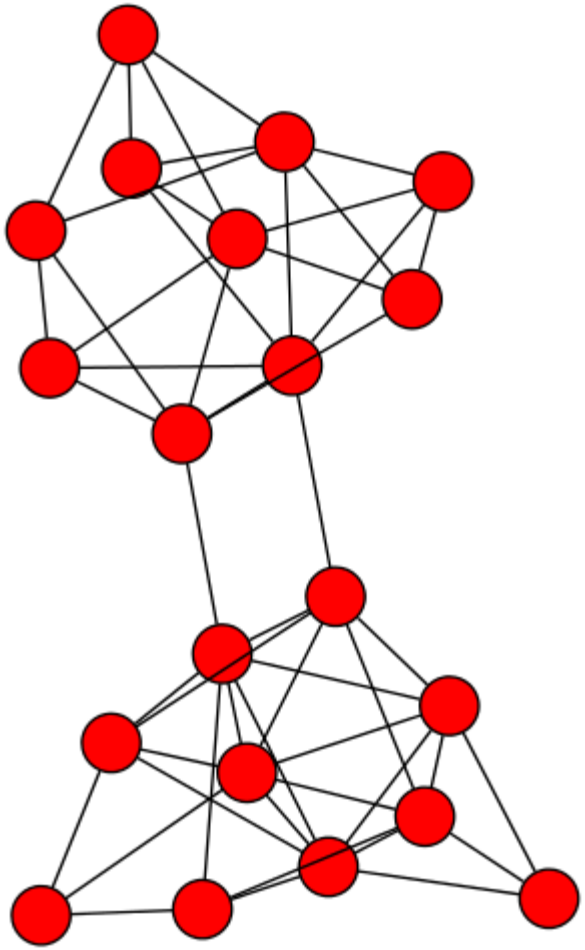
Split at 0:

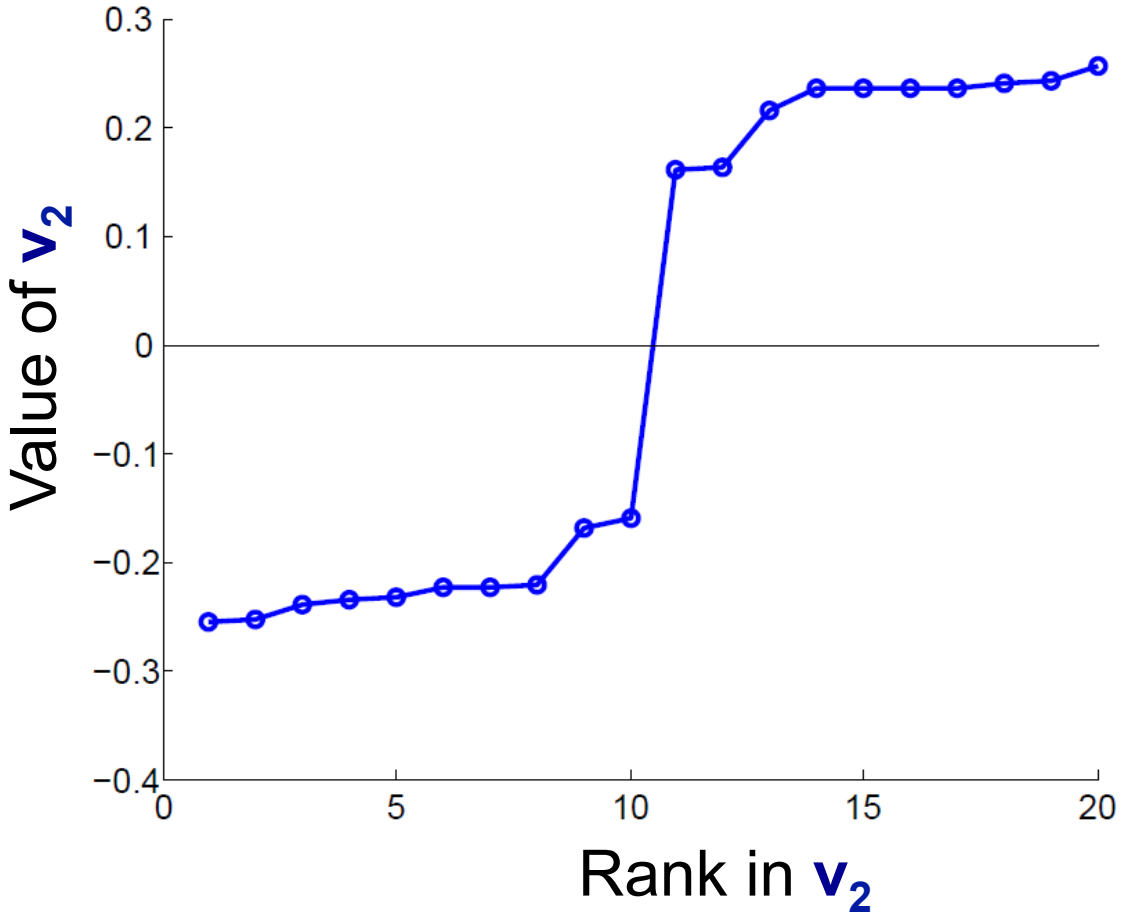
Cluster A: Positive points

Cluster B: Negative points

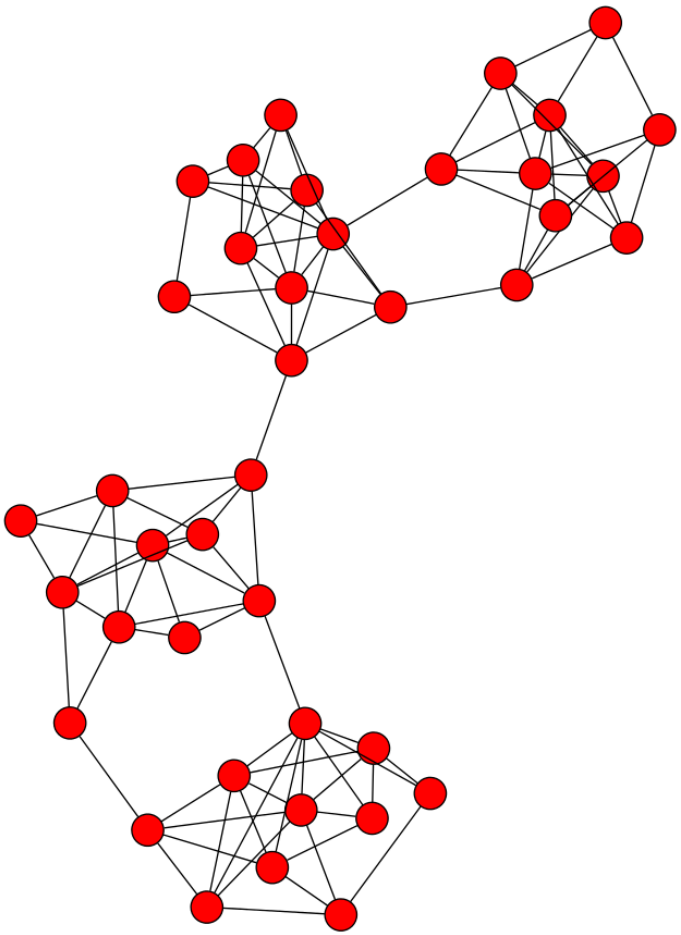


Example: Spectral Partitioning (1/3)

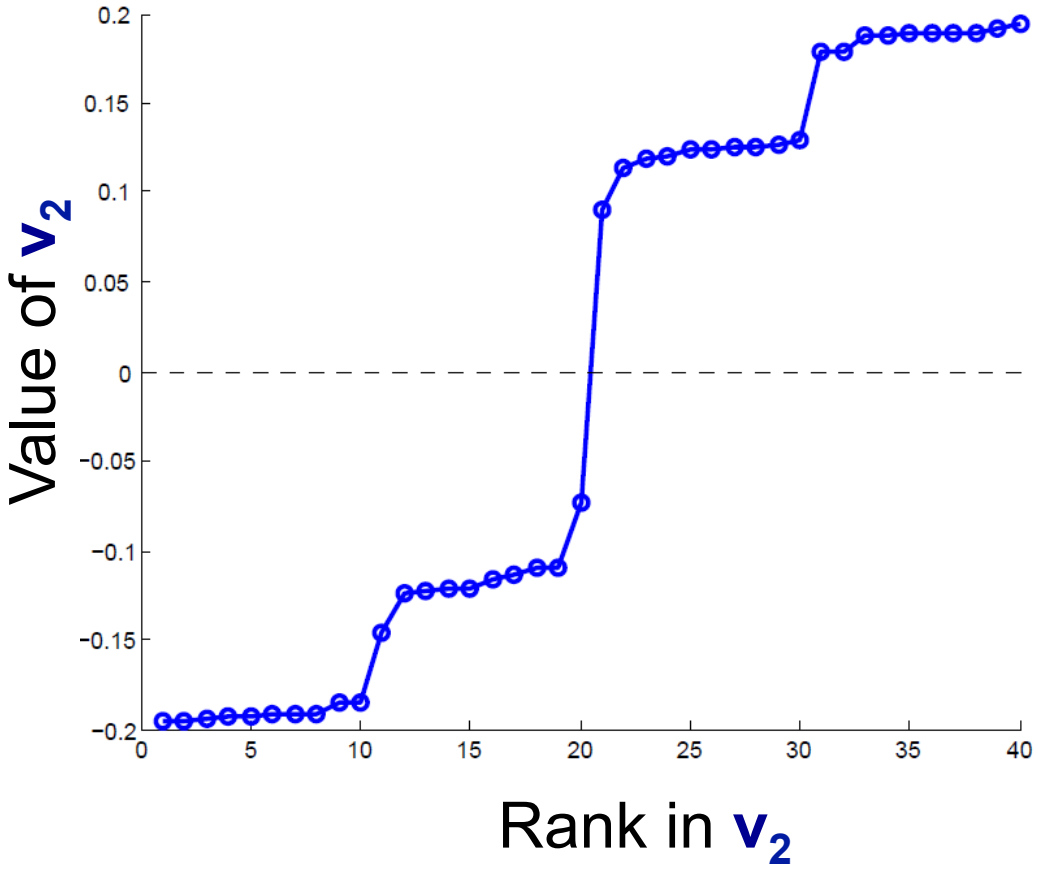




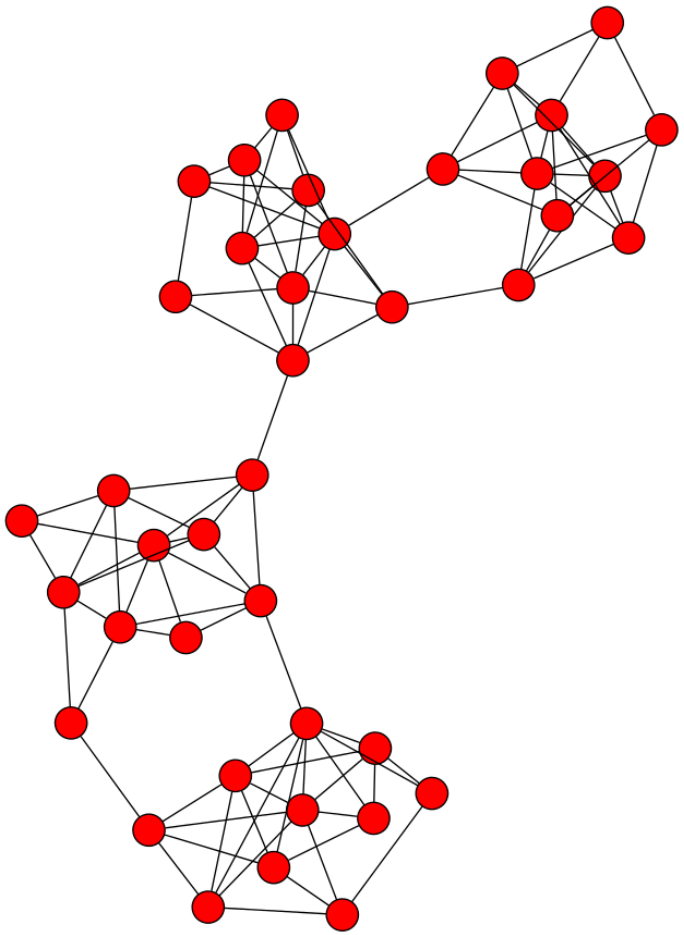
Example: Spectral Partitioning (2/3)

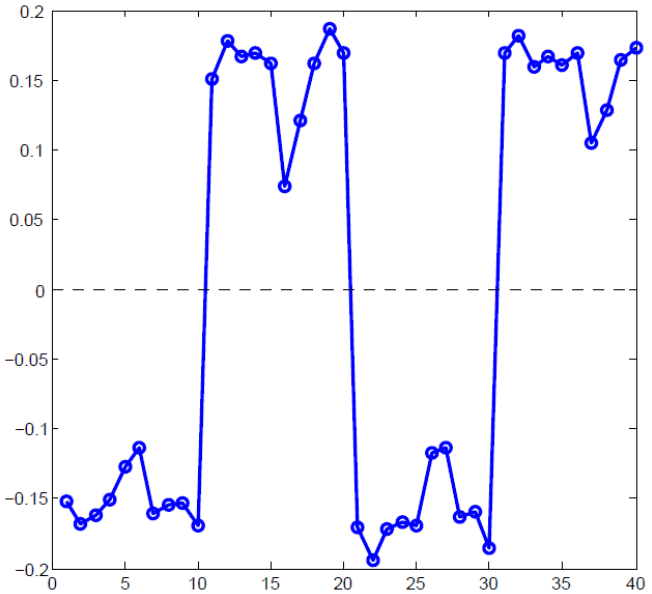


Components of v_2



Example: Spectral Partitioning (3/3)





Components of v_3

k-Way Graph Partitioning

$$W(A,B) = \sum_{i \in A, j \in B} A_{ij}$$

$\bar{A} = V \backslash A$ complement of A

$$\text{cut}(A_1, \ldots, A_k) = \frac{1}{2} \sum_{i=1}^{k} W(A_i, \bar{A}_i)$$

$$\text{ratio-cut}(A_1, \ldots, A_k) = \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \bar{A}_i)}{|A_i|} = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \bar{A}_i)}{|A_i|}$$

$$\text{normalized-cut}(A_1, \ldots, A_k) = \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \bar{A}_i)}{\text{vol}(A_i)} = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \bar{A}_i)}{\text{vol}(A_i)}$$

They can also be expressed using (variants of) the Laplacian matrix

k-Way Spectral Clustering

- How do we partition a graph into k clusters?

- Two basic approaches
  - Recursive bi-partitioning [Hagen et al., '92]
    - Recursively apply the bi-partitioning algorithm in a hierarchical divisive manner
    - Disadvantages: inefficient, unstable
  - Cluster multiple eigenvectors [Shi-Malik, '00]
    - Build a reduced space from multiple eigenvectors
    - Commonly used and preferable approach

Why Use Multiple Eigenvectors?

- Approximation of the optimal cut [Shi-Malik, '00]
  - Can be used to approximate the optimal k-way normalized cut
- Emphasizes cohesive clusters
  - Increases the unevenness in the distribution of the data
  - Associations between similar points are amplified, associations between dissimilar points are attenuated
  - The data begins to "approximate a clustering"
- Well-separated space
  - Transforms data to a new embedded space, consisting of k orthogonal basis vectors
- Multiple eigenvectors prevent instability due to information loss

k-Way (Normalized) Spectral Clustering

Normalized-cut criterion

- Input: Graph G = (V, E) and parameter k
- Output: Clusters $C_1, C_2, \ldots, C_k$ (i.e., cluster assignments of each node)

1. Let A be the adjacency matrix of the graph
2. Compute the normalized Laplacian matrix $\mathbf{L}_n = \mathbf{D}^{-1}\mathbf{L}$
3. Compute the first k eigenvectors of L_n

$$\mathbf{V} = [\mathbf{v}_1 | \mathbf{v}_2 | \ldots | \mathbf{v}_k] \in \mathbb{R}^{n \times k}$$

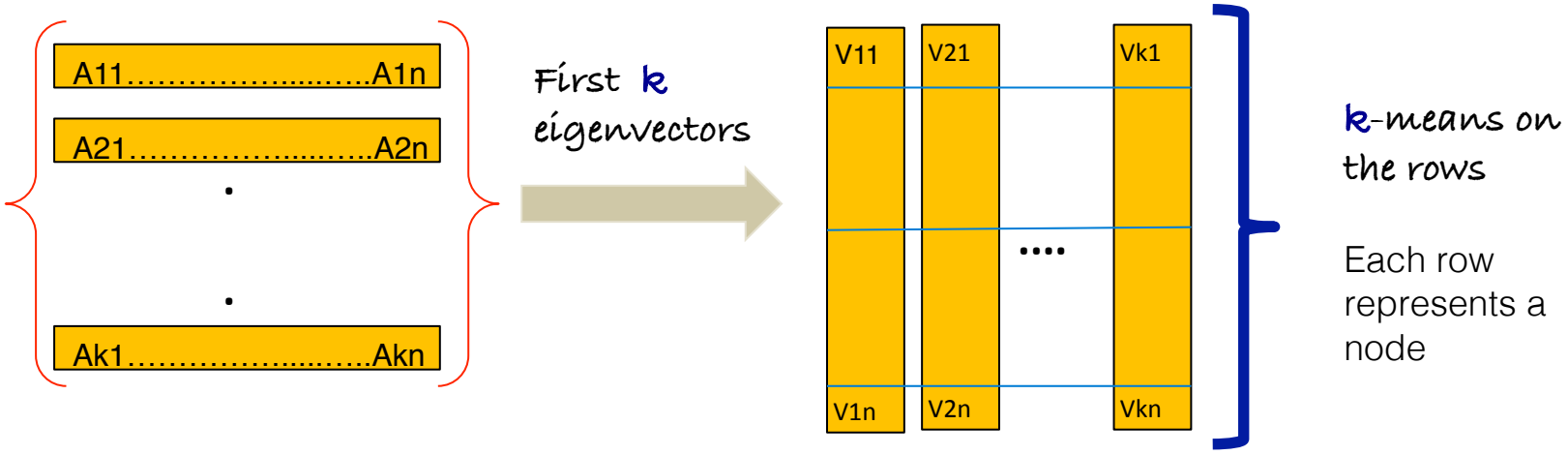
For i=1,…,n, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of V

4. Apply k-means to the points $(y_i)_{i=1,\ldots,n}$ (i.e., rows of V) to find the clusters $C_1, C_2, \ldots, C_k$

[von Luxburg '07], [Shi and Malik '00], [Ng, Jordan, Weiss '02]

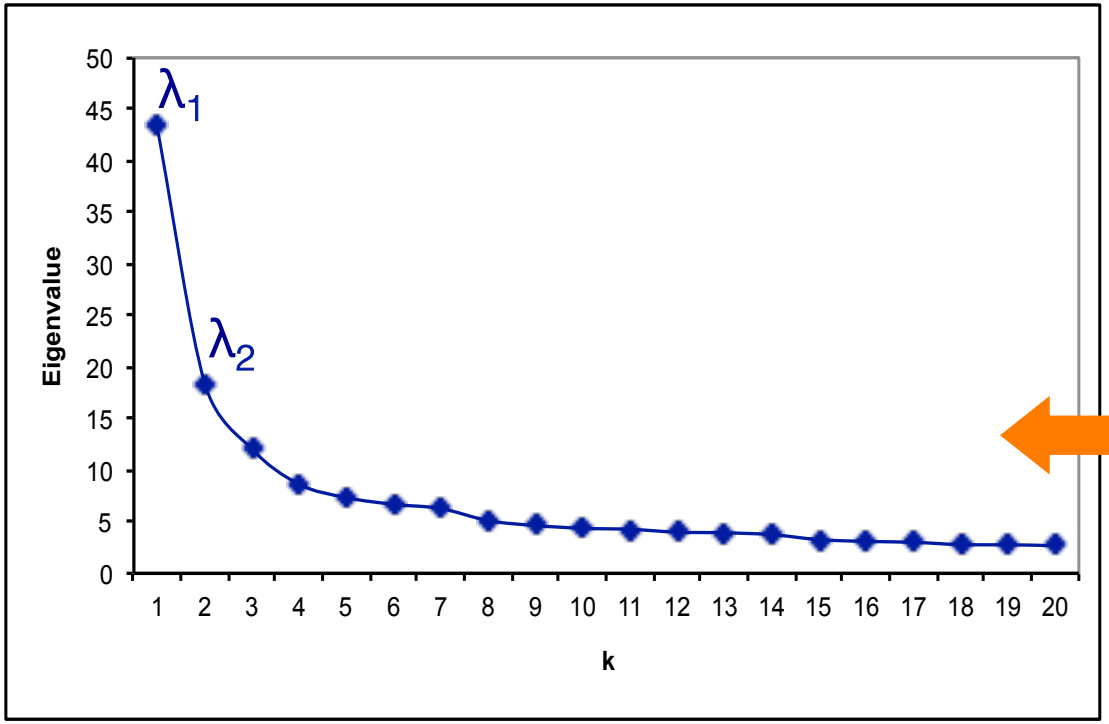
k-Way Spectral Clustering

- Consider the first k eigenvectors of $\mathbf{L}_n = \mathbf{D}^{-1}\mathbf{L}$ as the columns of a matrix
  - Run k-means on the rows of this matrix



How to Select k?

- Eigengap
  - The difference between two consecutive eigenvalues
- Most stable clustering is generally given by the value k that maximizes eigengap



In general, pick k that maximizes:

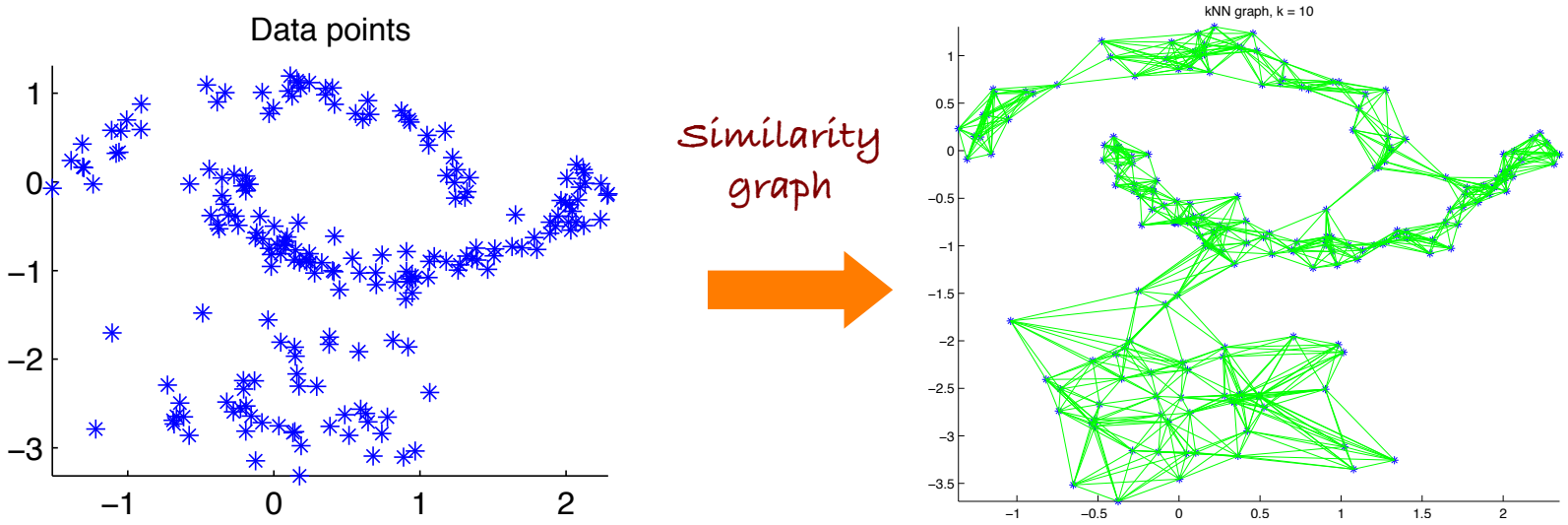
$$|\lambda_{k+1} - \lambda_k|$$

$$\max \Delta_k = |\lambda_2 - \lambda_1|$$

choose k=2

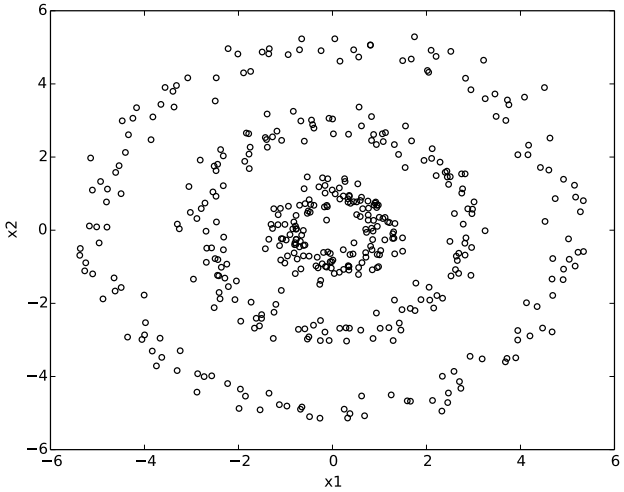
Clustering Non-Graph Data

- Apply graph clustering algorithms on data with no inherent graph structure (e.g., points in a d-dimensional Euclidean space)
- How?
  1. Construct a similarity graph based on the topological relationships and distances between data points
  2. Then, the problem of clustering the set of data points is transformed to a graph clustering problem

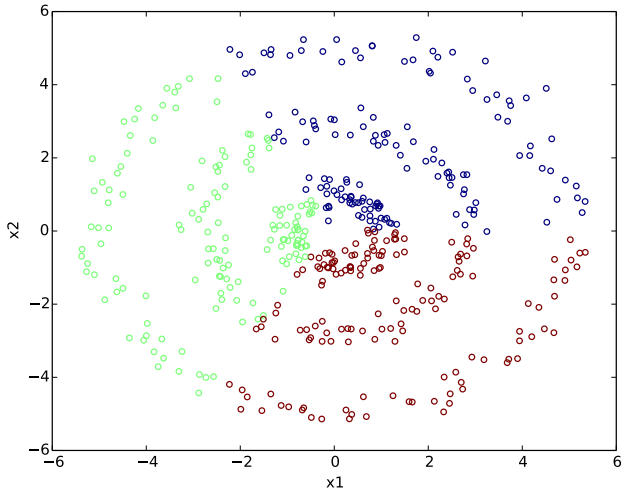


[von Luxburg '07], [Shi and Malik '00], [Ng, Jordan, Weiss '02]

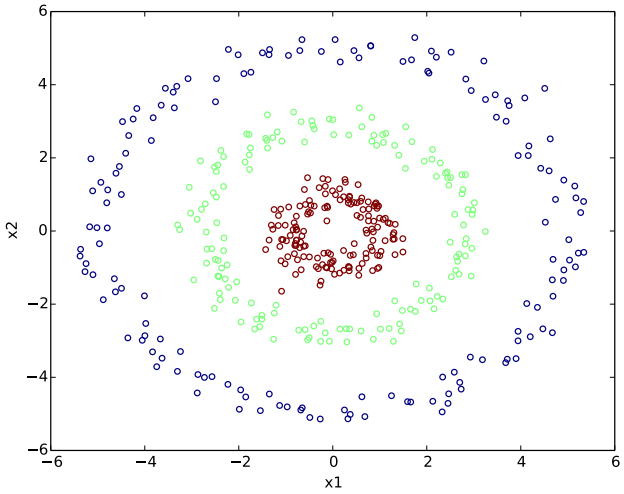
Spectral Clustering vs. k-Means



- 2-dimensional points
- Find k=3 clusters



k-means



spectral clustering

Many Other Partitioning Methods

- METIS
  - Heuristic but works really well in practice
  - http://glaros.dtc.umn.edu/gkhome/views/metis
- Graclus
  - Based on kernel k-means
  - http://www.cs.utexas.edu/users/dml/Software/graclus.html
- Louvain
  - Based on Modularity optimization
  - http://perso.uclouvain.be/vincent.blondel/research/louvain.html
- Clique percolation method
  - For finding overlapping clusters
  - http://angel.elte.hu/cfinder/

Other community evaluation measures

and some observations about their relationship

Community Detection – The Process

- How can we extract the inherent communities of graphs?
- Typically, a two-step approach
  1. Specify a quality measure (evaluation measure, objective function) that quantifies the desired properties of communities
  2. Apply algorithmic techniques to assign the nodes of graph into communities, optimizing the objective function
- Several measures for quantifying the quality of communities have been proposed
  - We have already seen modularity and cut-based criteria
  - In fact, there are many different ways to formalize the notion of communities

Community Evaluation Measures

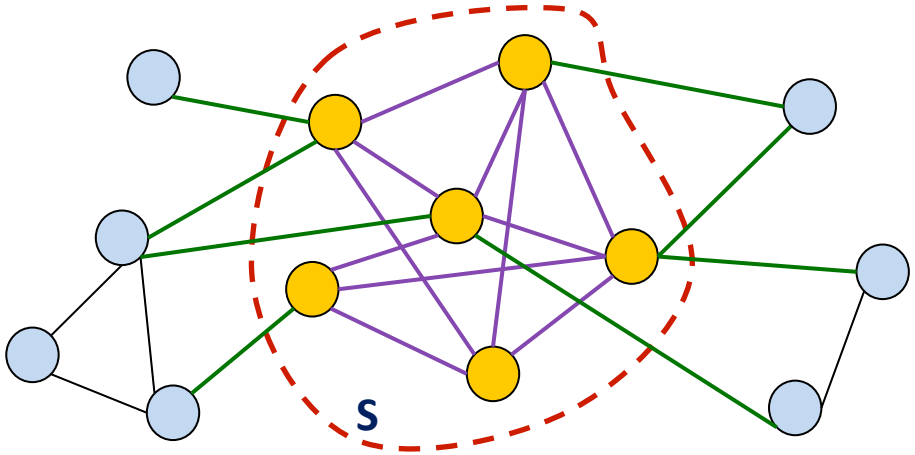
- Focus on
  - Intra-cluster edge density (# of edges within community)
  - Inter-cluster edge density (# of edges across communities)
  - Both two criteria

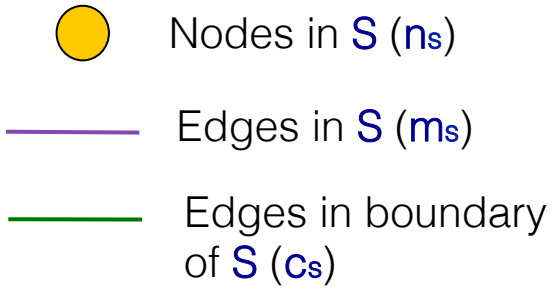
- We group the community evaluation measures according to
  - Evaluation based on internal connectivity
  - Evaluation based on external connectivity
  - Evaluation based on internal and external connectivity
  - Evaluation based on network model

[Leskovec et al. '10], [Yang and Leskovec '12], [Fortunato '10]

Notation

- G = (V, E) is an undirected graph, |V| = n, |E| = m
- S is the set of nodes in the cluster
- n_s = |S| is the number of nodes in S
- m_s is the number of edges in S, $m_s = \left|\{(u,v): u \in S, v \in S\}\right|$
- c_s is the number of edges on the boundary of S, $c_s = \left|\{(u,v): u \in S, v \notin S\}\right|$
- k_u is the degree of node u
- f(S) represent the clustering quality of set S



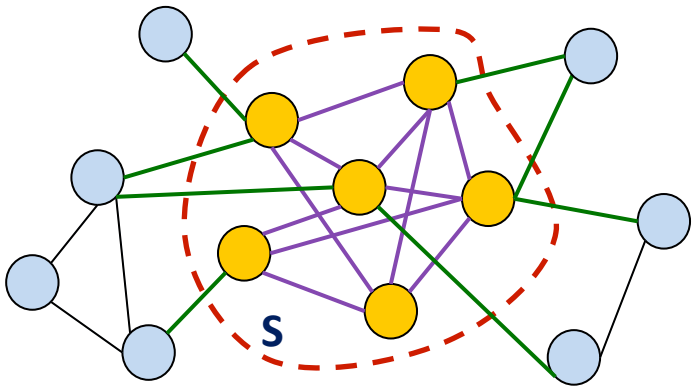


Internal Connectivity (1/3)

- Internal density [Radicchi et al. '04]

$$f(S) = \frac{m_s}{n_s(n_s - 1)/2}$$

Captures the internal edge density of community S



- Edges inside [Radicchi et al. '04]

$$f(S) = m_s$$

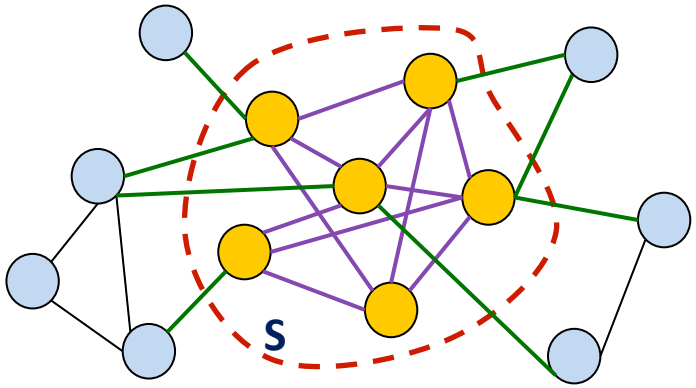
Number of edges between the nodes of S

Internal Connectivity (2/3)

- Average degree [Radicchi et al. '04]

$$f(S) = \frac{2m_s}{n_s}$$

Average internal degree of nodes in S



- Fraction over median degree (FOMD) [Yang and Leskovec '12]

$$f(S) = \frac{\left|\left\{u : u \in S, \left|\left\{(u,v) : v \in S\right\}\right| > d_m\right\}\right|}{n_s}$$

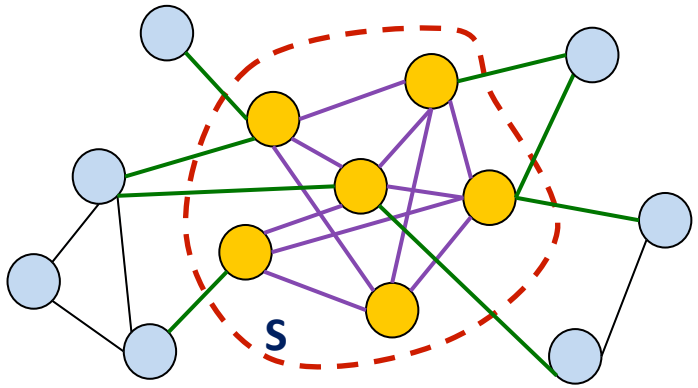
Fraction of nodes in S with internal degree greater than d_m, where d_m = median (d_u)

Internal Connectivity (3/3)

- Triangle participation ratio (TPR) [Yang and Leskovec '12]

$$f(S) = \frac{\left|\left\{u : u \in S, \left\{(v,w) : v,w \in S, (u,v) \in E, (u,w) \in E, (v,w) \in E\right\} \neq \varnothing\right\}\right|}{n_s}$$

Fraction of nodes in S that belong to a triangle

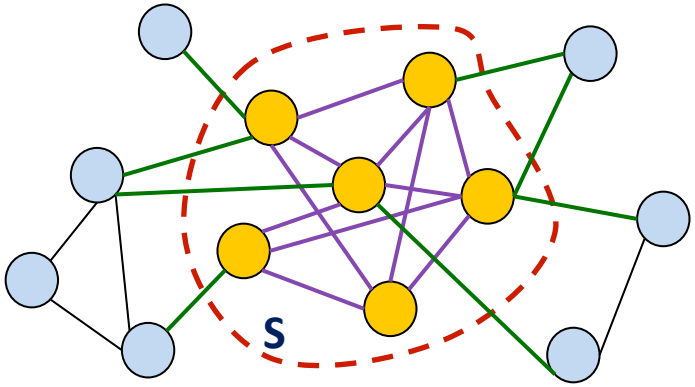


External Connectivity

- Expansion [Radicchi et al. '04]

$$f(S) = \frac{c_s}{n_s}$$

Measures the number of edges per node that point outside S



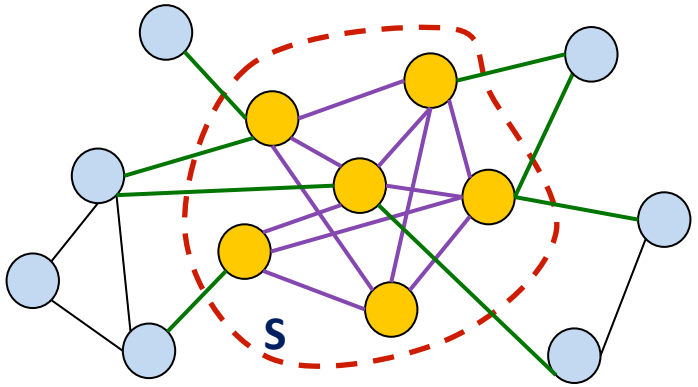
- Ratio cut

Internal and External Connectivity (1/2)

- Conductance [Chung '97]

$$f(S) = \frac{c_s}{2m_s + c_s}$$

Measures the fraction of total edge volume that points outside outside S



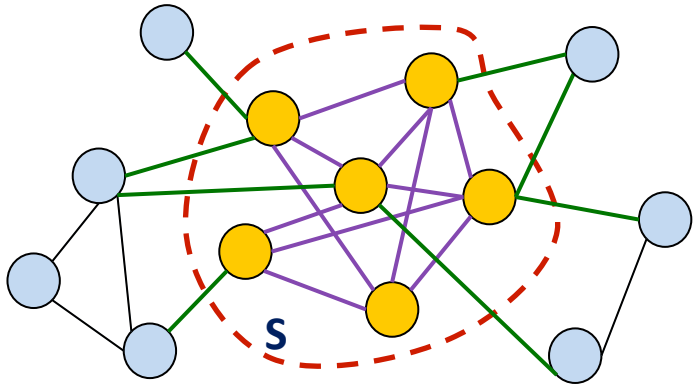
- Normalized cut [Shi and Malic '00]

Internal and External Connectivity (2/2)

- Maximum out degree fraction (Max ODF) [Flake et al '00]

$$f(S) = \max_{u \in S} \frac{\left| \{(u,v) \in E : v \notin S\} \right|}{d_u}$$

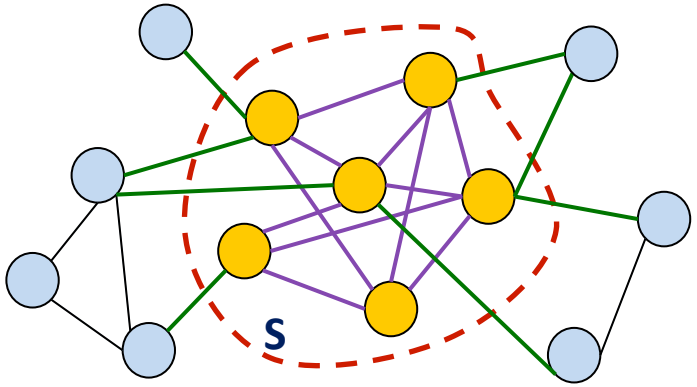
Measures the maximum fraction of edges of a node in S that point outside S



Evaluation Based on Network Model

- Modularity [Newman and Girvan '04], [Newman '06]

Measures the difference between the number of edges in S and the expected number of edges $E(m_s)$ in case of a configuration model

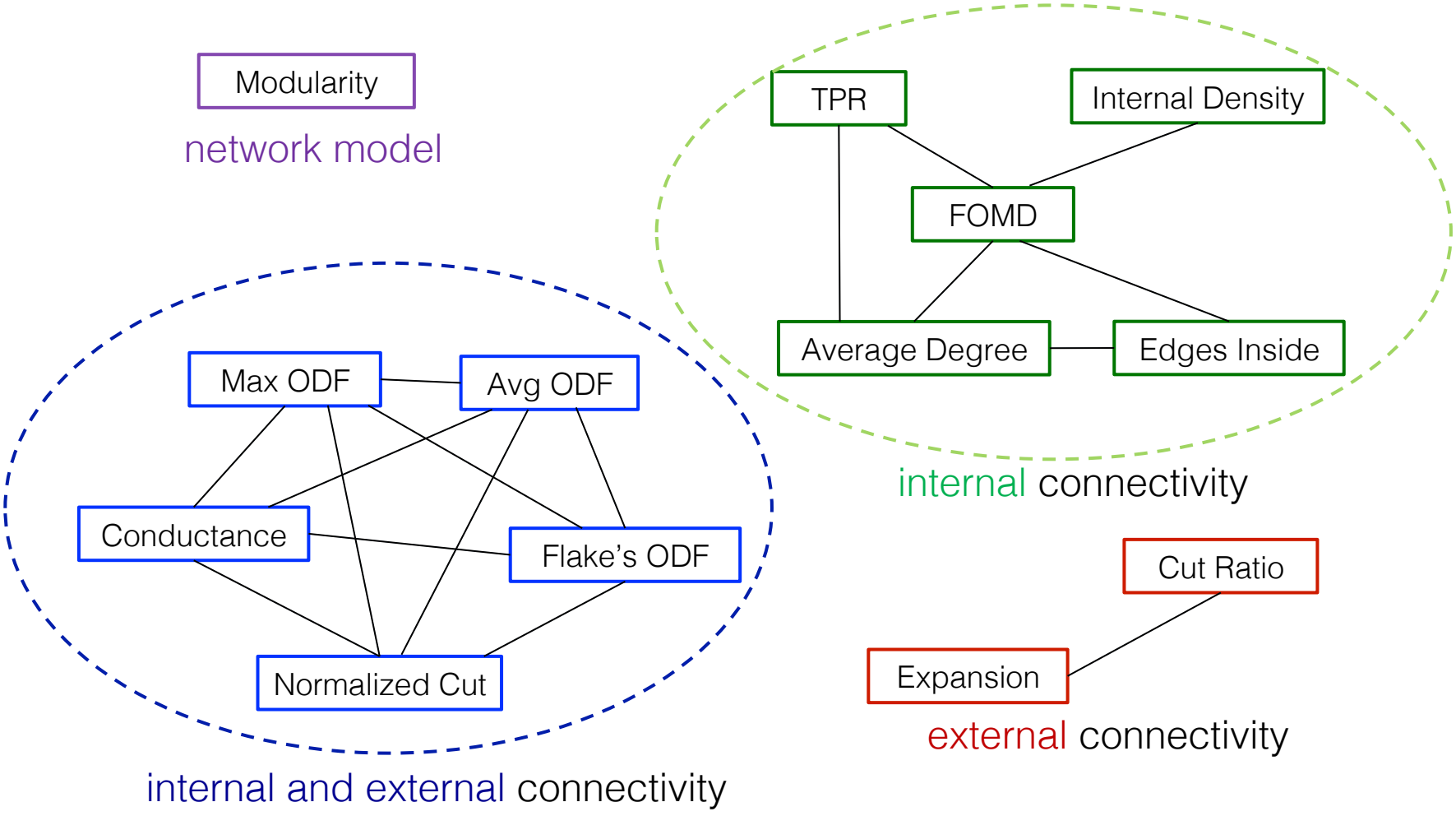


How Different are Those Measures? (1/3)

- Several community evaluation measures (objective criteria) have been proposed
- Is there any relationship between them?

- Consider real graphs with known node assignment to communities (ground-truth information) and test the behavior of the objective measures [Yang and Leskovec '12]
  1. For each of the ground-truth communities S
  2. Compute the score of S using each of the previously described evaluation measures
  3. Form the correlation matrix of the objective measures based on the scores
  4. Apply a threshold in the correlation matrix
  5. Extract the correlations between community objective measures

How Different are Those Measures? (2/3)



Observation: Community evaluation measures form four groups based on their correlation [Yang and Leskovec '12]

How Different are Those Measures? (3/3)

- The different structural definitions of communities are heavily correlated [Yang and Leskovec '12]
- Community evaluation measures form four groups based on their correlation
- These groups correspond to the four main notions of structural communities
  - Communities based on internal connectivity
  - Communities based on external connectivity
  - Communities based on internal and external connectivity
  - Communities based on a network model (modularity)

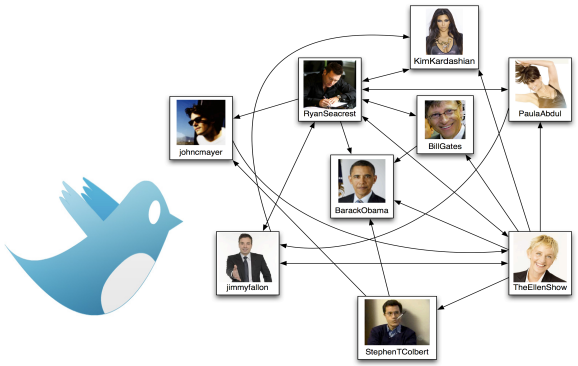
Next Part

- Community detection in directed networks
- Community structure in large scale networks
- Overlapping communities

Community detection in directed networks

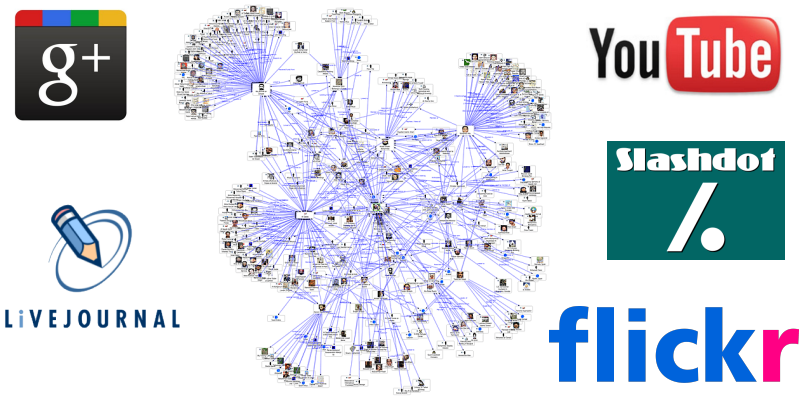
Directed Graphs (1/2)

- A plethora of network data from several applications is from their nature directed

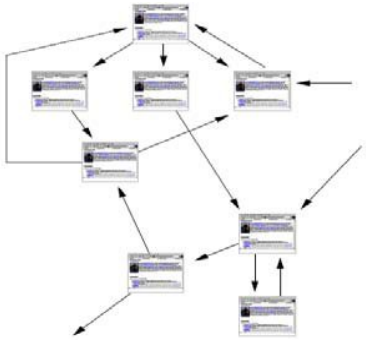


Twitter

[Image: http://sites.davidson.edu/mathmovement/]

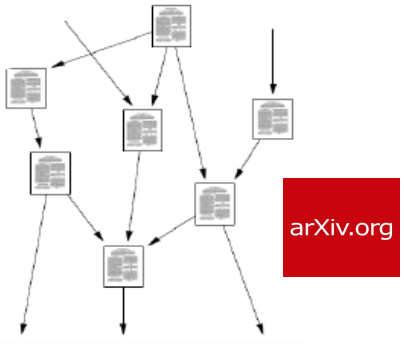


Online Social Networks



Web Graph

Wikipedia



Citation Graph

Challenges (1/2)

- The problem is generally a more challenging task compared to the undirected one
- Existence of asymmetric relationships among entities (non reciprocal) → the nature of interactions are fundamentally different from the one in the undirected case
- Graph concepts for community evaluation (e.g., density)
  - Well theoretically founded for undirected graphs
  - Not enough effort has been put on how to extend these concepts on directed graphs
- Theoretical tools
  - Mainly graph theoretic and linear algebraic tools
  - Have mainly been considered for undirected graphs
  - Not straightforward extension to the directed case

Challenges (2/2)

- No precise and common definition for the problem
- The presence of directed links is possible to imply the existence of other more sophisticated types of clusters that
  - Do not exist in undirected networks
  - Cannot be captured using only density and edge concentration characteristics
- Ignoring directionality and naively transform the graph to undirected in not a good practice

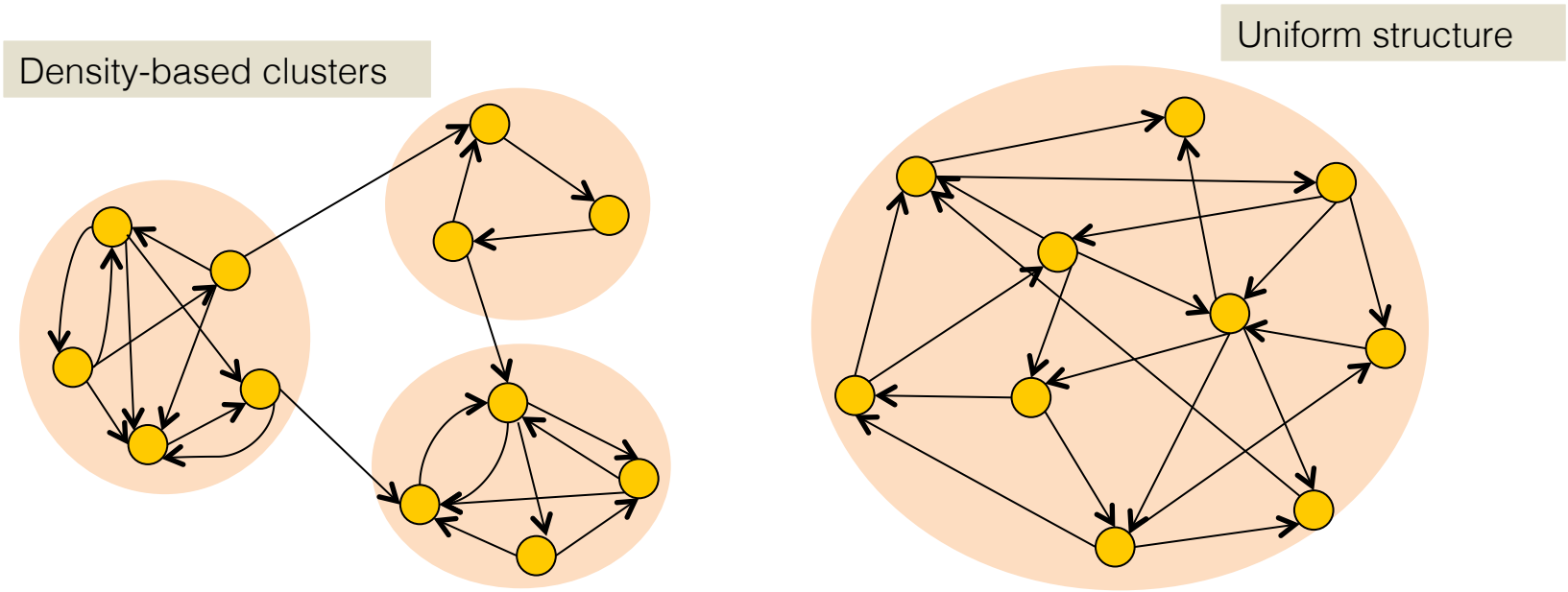
Communities in Directed Networks

A cluster or community in a graph can be considered as a set of nodes that share common or similar features (characteristics)

- Two main definitions/notions (or categories) of clusters in directed networks
  - Density-based clusters
  - Pattern-based clusters

Density-based Clusters

- Follow the typical clustering definition based on edge density characteristics
  - Entirely based on the distribution-density of the edges inside the network
  - Group of nodes with more intra-cluster edges than inter-cluster edges



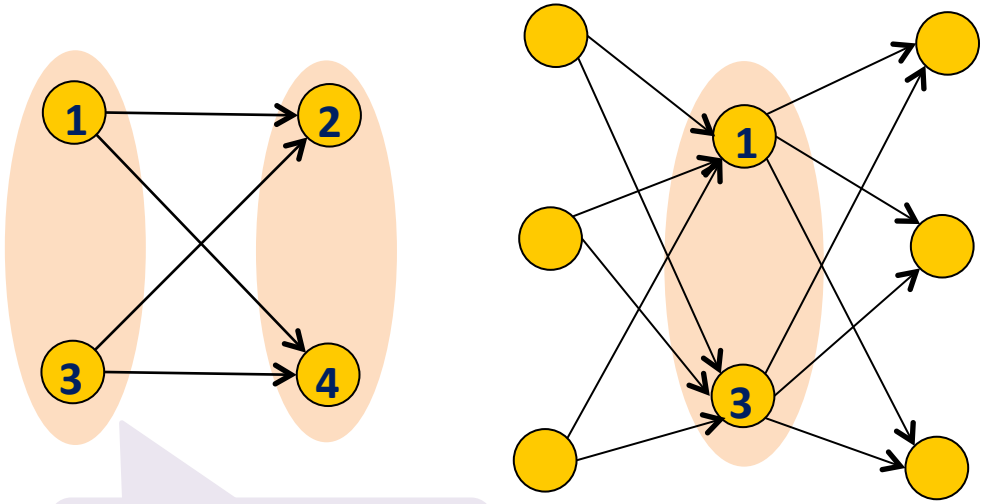
Is It a "Trivial" Task?

- Extending the notion of density-based clusters to directed networks is not always a trivial procedure
- Meaningful extension of the objective criteria (e.g., modularity) used for community evaluation
- Simple graph concepts become more complex
  - E.g., each cluster should be connected [Schaeffer '07]
  - Connectivity in directed graphs
    - Weak connectivity
    - Strong connectivity

Pattern-based Clusters

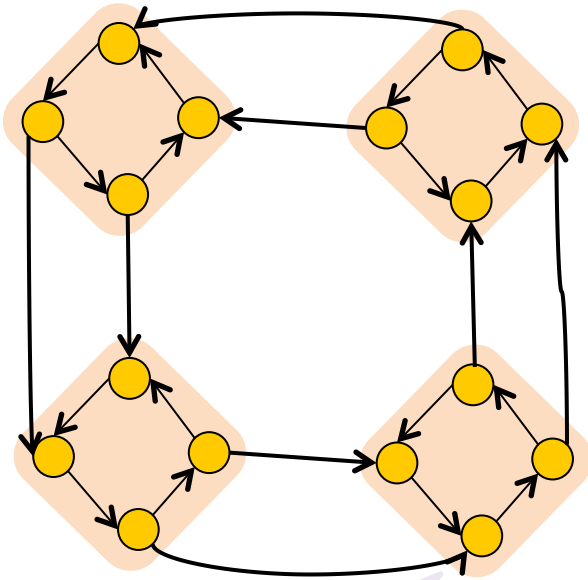
- The density-based definition cannot capture more sophisticated clustering and connectivity patterns
  - Edge density alone may not represent the major clustering criterion
  - Patterns beyond edge density

Citation-based clusters



Nodes point to and pointed by the same group

Flow-based cluster



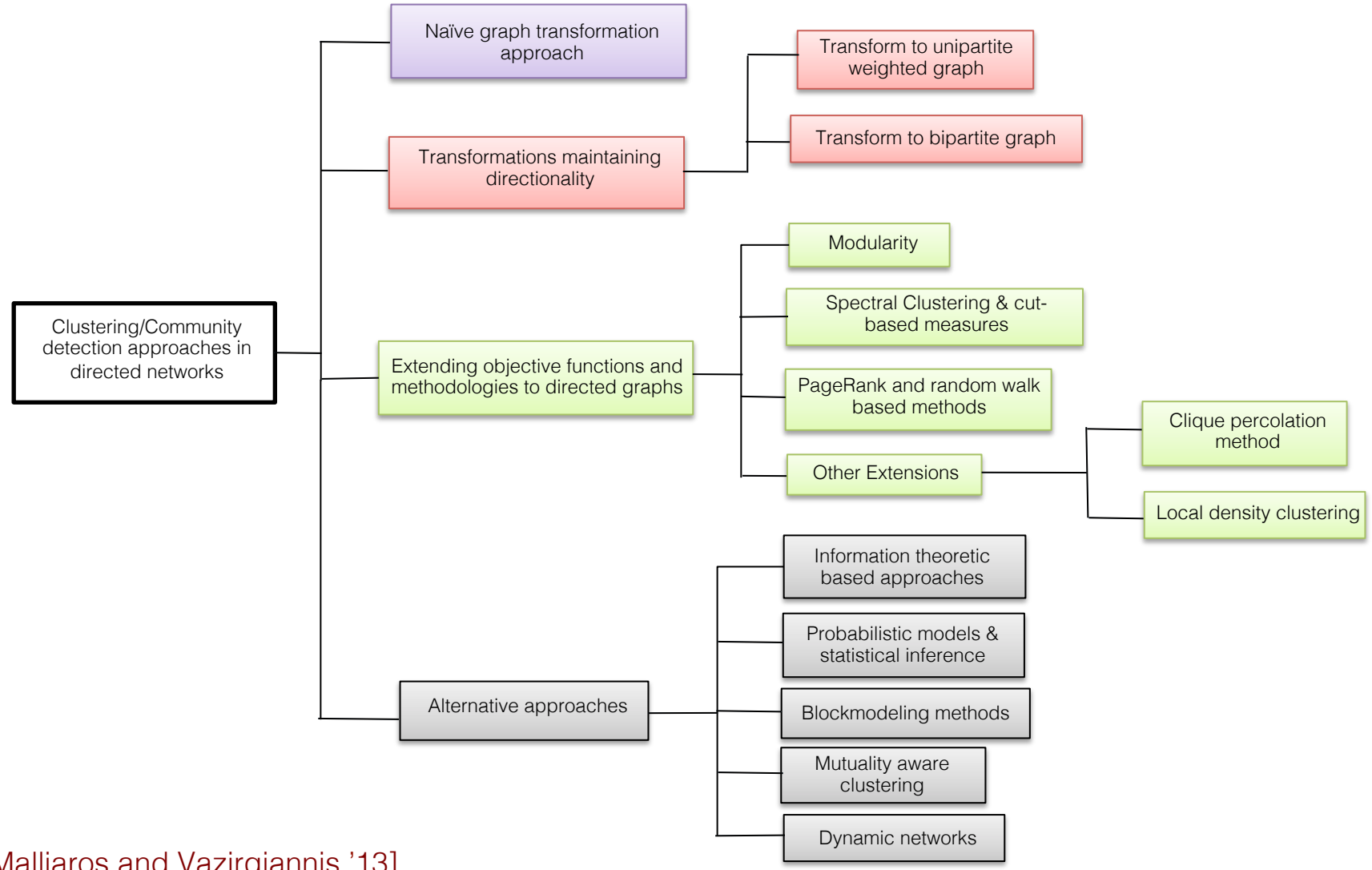
Patterns of flow – groups with persistent flow

Remarks on Clustering Notions

- Both types of clusters may co-exist in a directed graph
  - Combined density-based and pattern-based clusters
- E.g., many methods adopt the citation-based clustering notion and are also able to identify density-based clusters

- Key point:
  - Apply appropriate transformations to enhance a density-based method with pattern-based clustering features

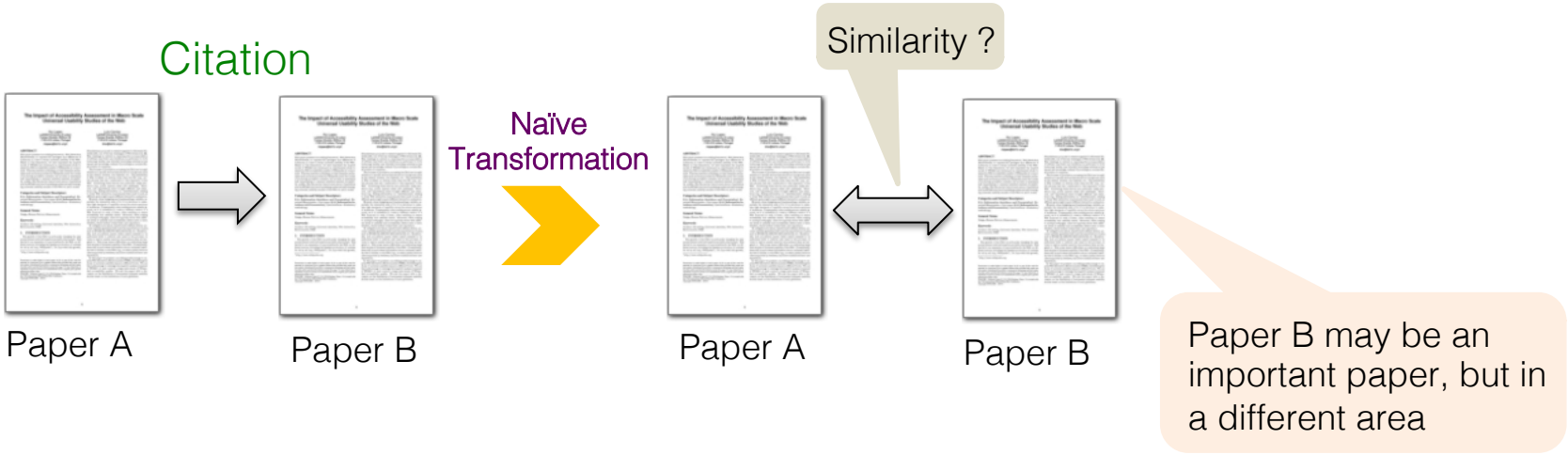
Taxonomy of Algorithms



[Malliaros and Vazirgiannis '13]

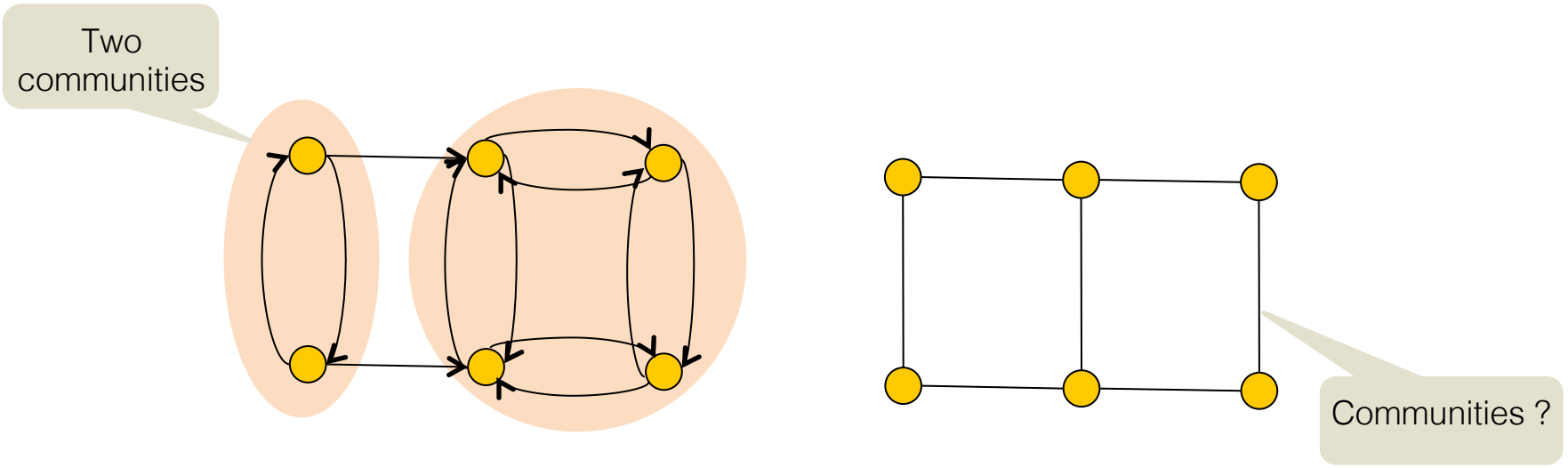
Naïve Graph Transformation (1/2)

- Discard edge directionality and treat graphs as undirected → Apply algorithms for undirected graphs
- Several drawbacks: information represented by edges' direction is ignored
- Data ambiguities
  - Ambiguities and to some degree incorrect information are introduced in the graph



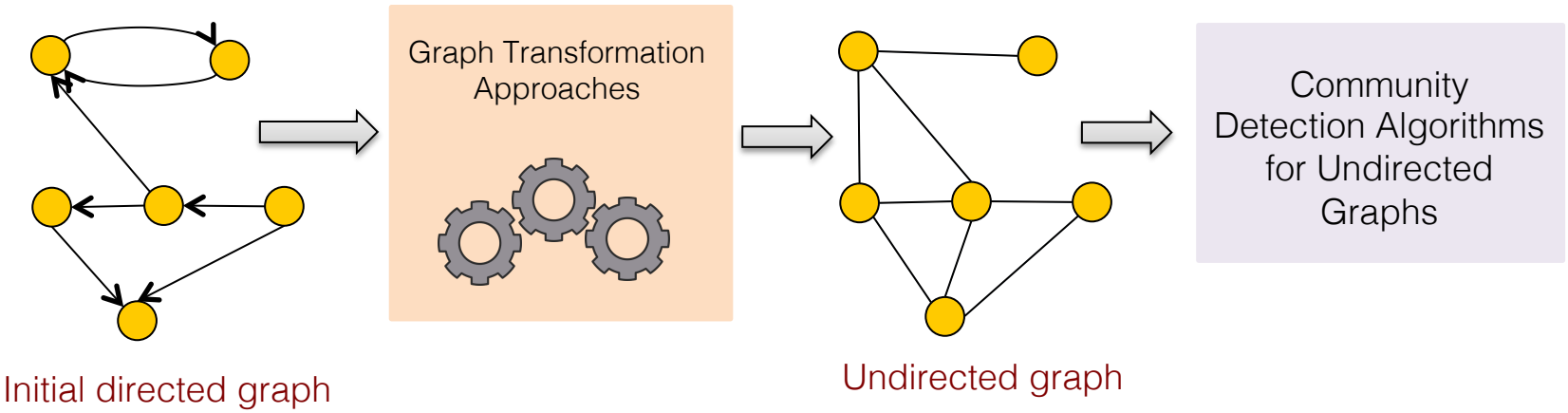
Naïve Graph Transformation (2/2)

- Deviations in clustering results
  - Ambiguities introduced in the data, may have impact to the final outcome of the clustering algorithm
  - Valuable information is not utilized in the clustering process
  - E.g., clusters that exist in the initial directed network, may not be identified at the transformed one



Transformations Maintaining Directionality

1. Transform the directed graph to undirected (unipartite / bipartite)
2. Edges' direction information is retained as much as possible (e.g., by introducing weights on the edges of the transformed graph)
3. Apply already proposed community detection algorithms designed for undirected graphs
4. The extracted communities will also correspond to the communities of the initial graph

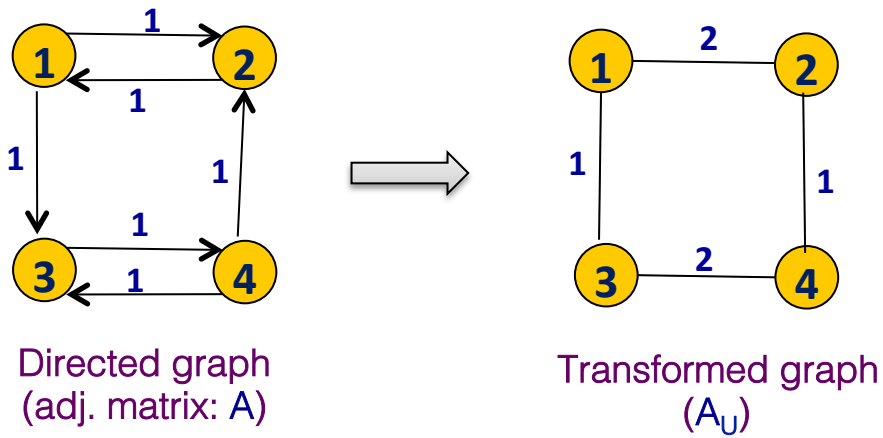


Transformation to Unipartite Weighted Graph (1/4)

- Idea: transform the directed graph to undirected
  - Information about directionality is incorporated via edge weights
- Graph symmetrizations [Satuluri and Parthasarathy '11]

$$\mathbf{A}_U = A + A^T$$

Adjacency matrix of directed graph



Directed graph (adj. matrix: A)

Transformed graph (A_U)

- Same number of edges
- Edges in both direction:
  - Add as edge weight the sum of the weights in the initial graph

Transformation to Unipartite Weighted Graph (2/4)

- The previous symmetrization maintains intact the edge set (discard directions – new edge weights)
- Observation: Meaningful clusters can be groups of nodes that share similar incoming and/or outgoing edges
  - Edges should appear between similar nodes (in-link and out-link node similarity)

- Bibliometric symmetrization [Satuluri and Parthasarathy '11]

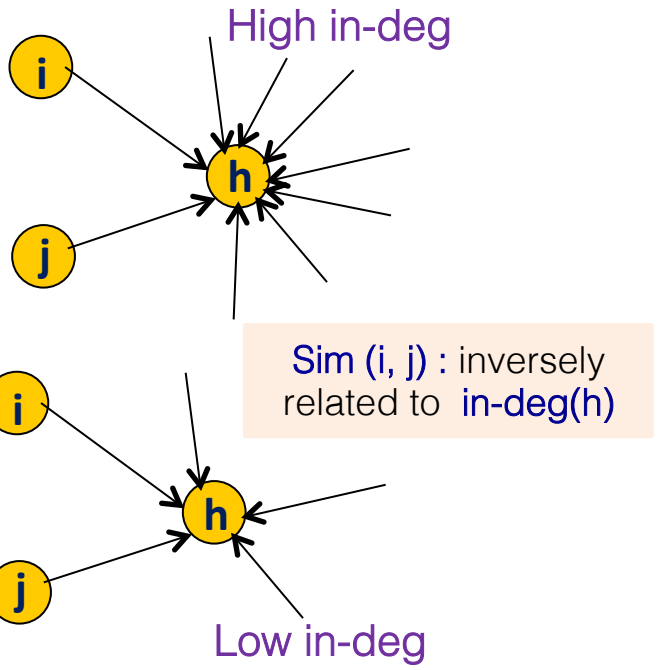
$$\mathbf{A}_U = \mathbf{A}\mathbf{A}^T + \mathbf{A}^T\mathbf{A}$$

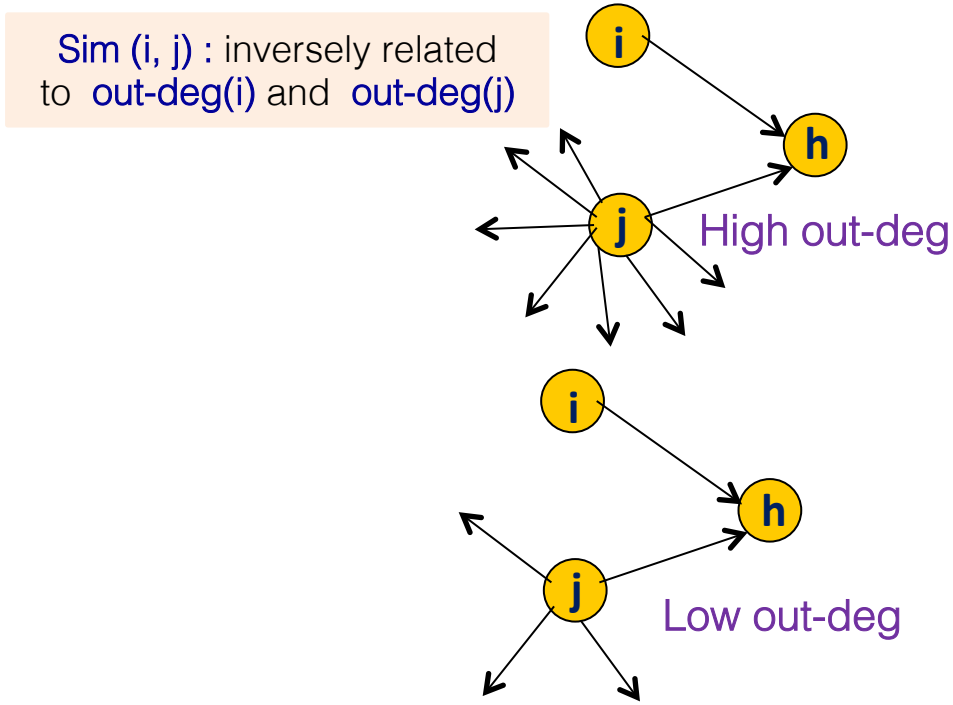
  - $\mathbf{C} = \mathbf{A}\mathbf{A}^T$: Bibliographic coupling matrix (captures the number of common outgoing edges between each pair of nodes)
  - $\mathbf{B} = \mathbf{A}^T\mathbf{A}$: Co-citation strength matrix (captures the number of common incoming edges between each pair of nodes)

Introduce new edges based on the number of common outgoing and incoming edges

Transformation to Unipartite Weighted Graph (3/4)

- The degree distribution of real-world networks is heavy-tailed
- Nodes with high degree would share a lot of common edges with other nodes (higher similarity)
- How can we define a similarity measure between the nodes of a directed graph, taking into account in- and out- degree?





Transformation to Unipartite Weighted Graph (4/4)

- Degree discounted symmetrization

$$B = D_{out}^{-a} A D_{in}^{-\beta} A^T D_{out}^{-a}$$

Bibliographic coupling matrix

$$C = D_{in}^{-\beta} A^T D_{out}^{-a} A D_{in}^{-\beta}$$

Co-citation matrix

- Adjacency matrix of symmetrized undirected graph

$$A_U = B + C$$

Typically, $\alpha = \beta = 0.5$ [Satuluri and Parthasarathy '11]

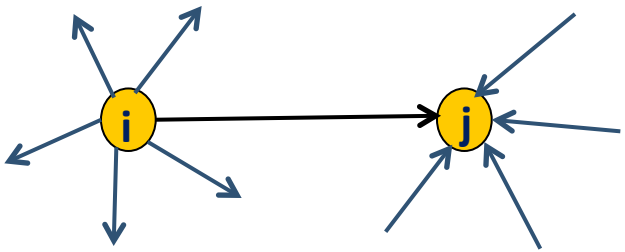
Modularity for Directed Graphs

- Initially introduced for the case of undirected graphs

  Q = (fraction of edges within communities) -
  (expected fraction of edges)

- In directed graphs, the existence of a directed edge (i, j) between nodes i and j depends on the out-degree of i and in-degree of j

$$Q_d = \frac{1}{m} \sum_{i,j} \left(\mathbf{A}_{ij} - \frac{k_i^{out} k_j^{in}}{m} \right) \delta(c_i, c_j)$$



Consider that:

- i has high out-deg and low in-deg
- j has high in-deg and low out-deg

More probable to observe edge (i, j) than edge (j, i)

[Arenas et al. '07], [Leicht and Newman '08]

Modularity Optimization

- Goal: Assign the nodes into two communities, X and Y
- Let $s_i, \forall i \in V$ be an indicator variable where $s_i = +1$ if i is assigned to X and $s_i = -1$ if i is assigned to Y

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

$$= \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{k_i^{out}k_j^{in}}{m}\right)(s_i s_j + 1)$$

$$= \frac{1}{2m}\sum_{ij} B_{ij} s_i s_j = \frac{1}{2m} s^T \mathrm{B} s$$

$$B_{ij} = A_{ij} - \frac{k_i^{out}\, k_j^{in}}{m}$$ Modularity matrix (nonsymmetric)

[Leicht and Newman '08]

- Transpose Q_d (scalar) and take the average

$$Q_d = \frac{1}{4m} s^T \left(B + B^T\right) s$$

- Now $B + B^T$ is symmetric

Spectral optimization of modularity

1. Compute the eigenvector that corresponds to the largest positive eigenvalue of $B + B^T$
2. Assign the nodes to communities X and Y according to the signs of the corresponding components in the eigenvector
3. Repeated bisection (for more than two communities)

Laplacian Matrix for Directed Graphs

- Undirected networks: use the eigenvector that corresponds to the second smallest eigenvalue of the Laplacian matrix (Fiedler vector) to obtain a bipartition of the nodes
  - Solution to the normalized cut objective function

- Laplacian matrix for directed graphs

$$L_d = I - \frac{\Pi^{1/2} P \Pi^{-1/2} + \Pi^{-1/2} P^T \Pi^{1/2}}{2}$$

  - P is the transition matrix and $\Pi = \text{diag}(\pi_1, \pi_2, \ldots, \pi_n)$ the stationary distribution of the random walk
    - Cheeger inequality holds for L_d

[Chung '07], [Zhou et al. '05], [Li and Zhang '10]

Directed Spectral Clustering Algorithm

Input: Directed graph $G = (V, E)$

Output: A partition of the vertex set V into two groups

1. Define a random walk over G with transition matrix P
2. Form the normalized Laplacian matrix L_d
3. Compute the eigenvector u_2 of L_d that corresponds to the second smallest (non zero) eigenvalue
4. Partition the vertex set V into two parts
   a. $S = \{i \in V \mid u_2(i) \geq 0\}$
   b. $S' = \{i \in V \mid u_2(i) < 0\}$

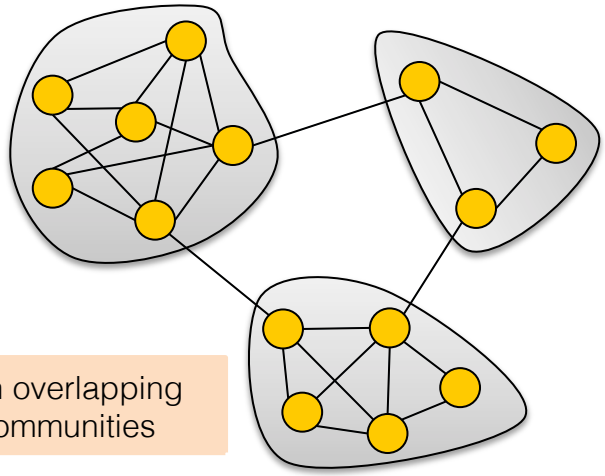
- The algorithm can be extended to the case of a k-partition
  - Eigenvectors of the k smallest eigenvalues of L_d

[Zhou et al. '10], [Gleich '06]

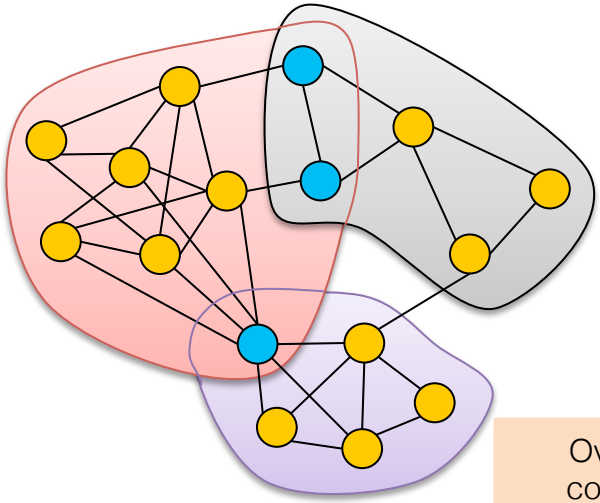
Overlapping community structure

Overlapping Community Detection

- Most of the methods presented so far perform hard clustering
  - The graph is divided into communities (clusters, modules)
  - Each node is assigned to a single community
- In many cases, nodes can simultaneously belong to more than one communities
  - Overlapping communities



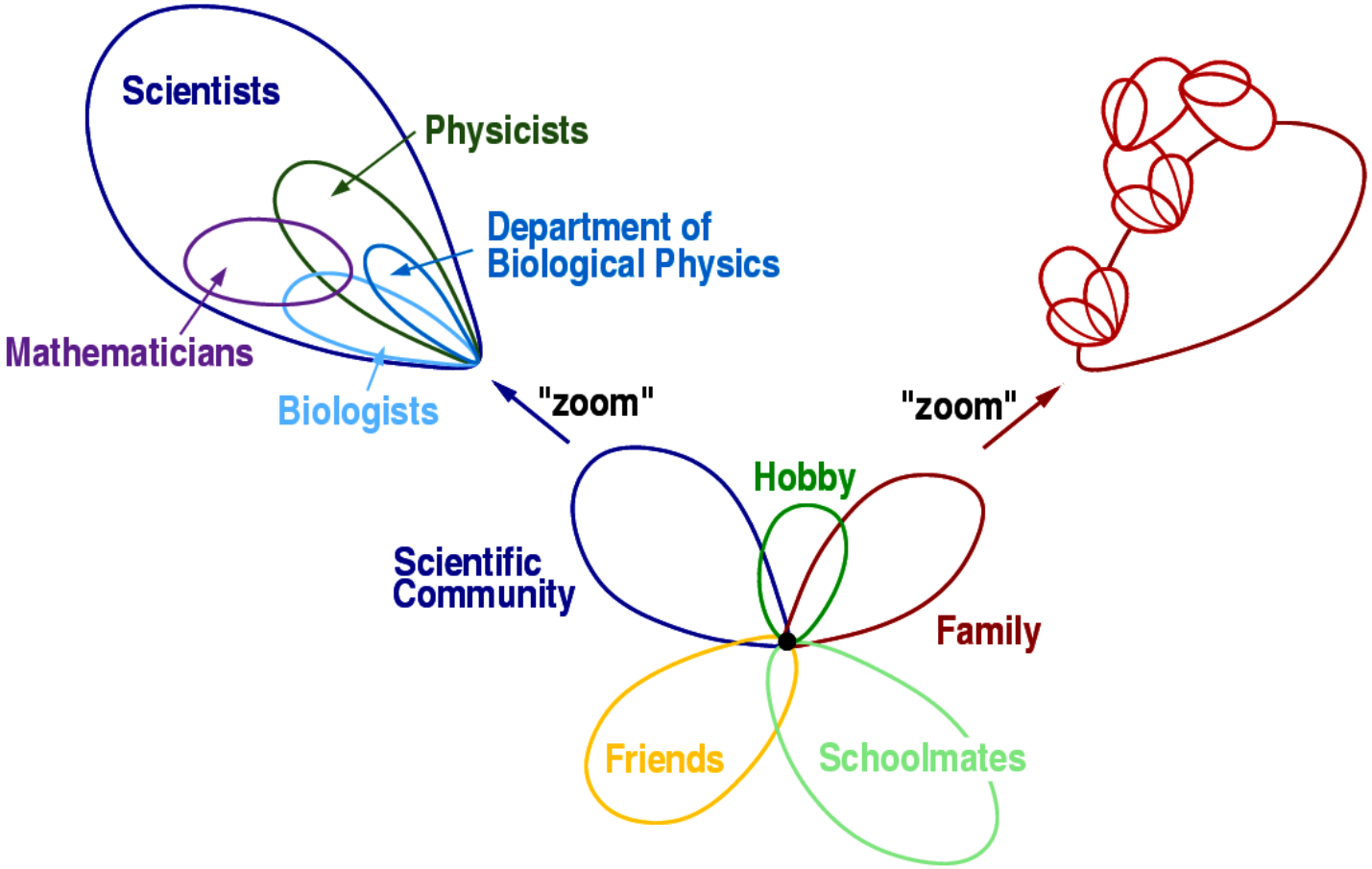
Non overlapping communities



Overlapping communities

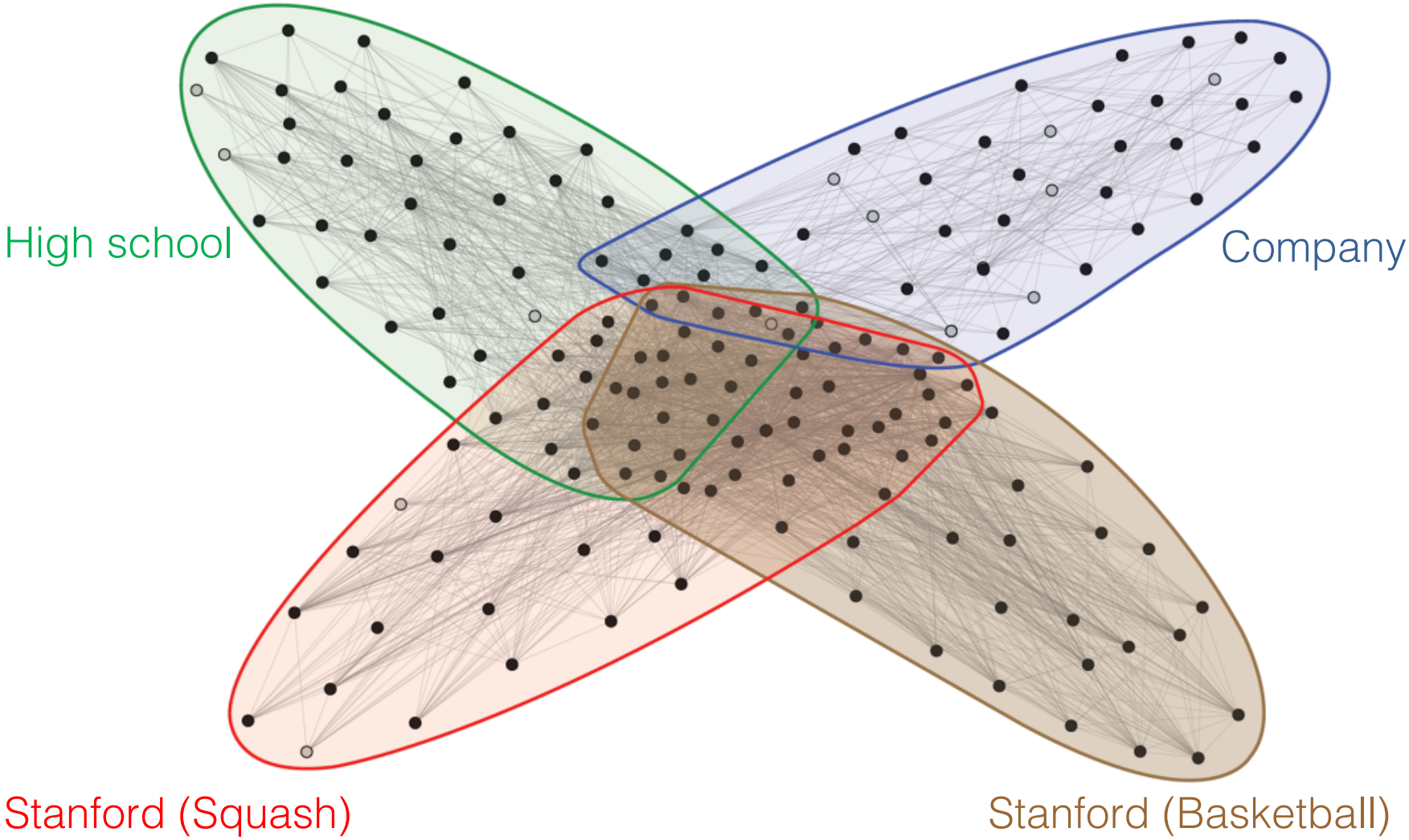
Overlaps of Social Circles (1/2)

A node can belong to many social “circles”



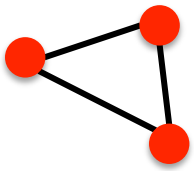
[Palla et al., ‘05]

Overlaps of Social Circles (2/2)

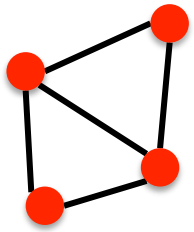


Clique Percolation Method (CPM)

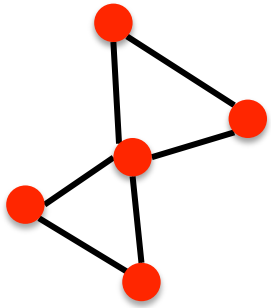
- Two nodes belong to the same community if they can be connected through adjacent k-cliques
  - k-clique:
    - Fully connected graph on k nodes
  - Adjacent k-cliques
    - Overlap in k-1 nodes
- k-clique community
  - Set of nodes that can be reached through a sequence of adjacent k-cliques



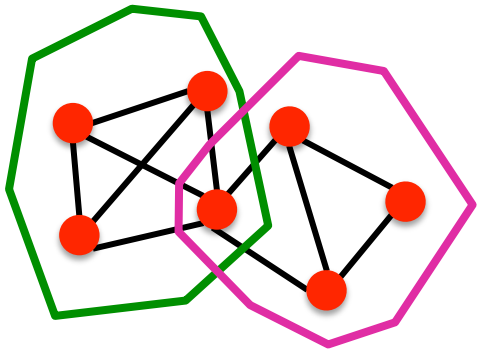
3-clique



Adjacent 3-cliques



Non-adjacent 3-cliques

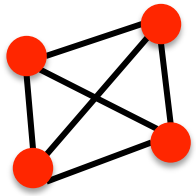


Two overlapping 3-clique communities

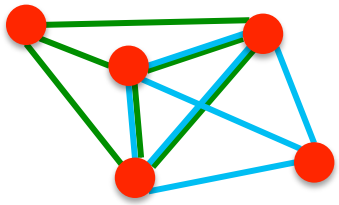
[Palla et al., '05] **Code:** http://www.cfinder.org/

Clique Percolation Method (CPM)

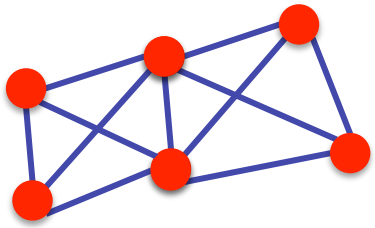
- Two nodes belong to the same community if they can be connected through adjacent k-cliques



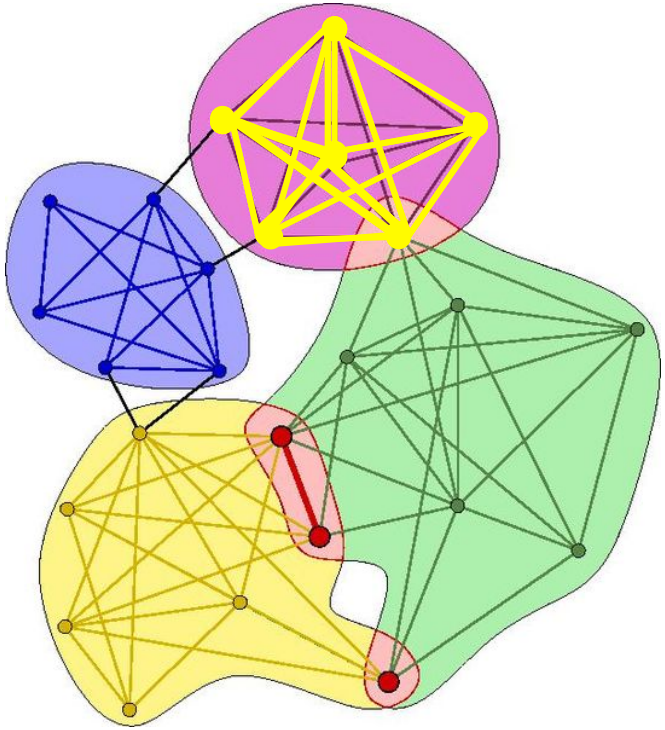
4-clique



Adjacent 4-cliques



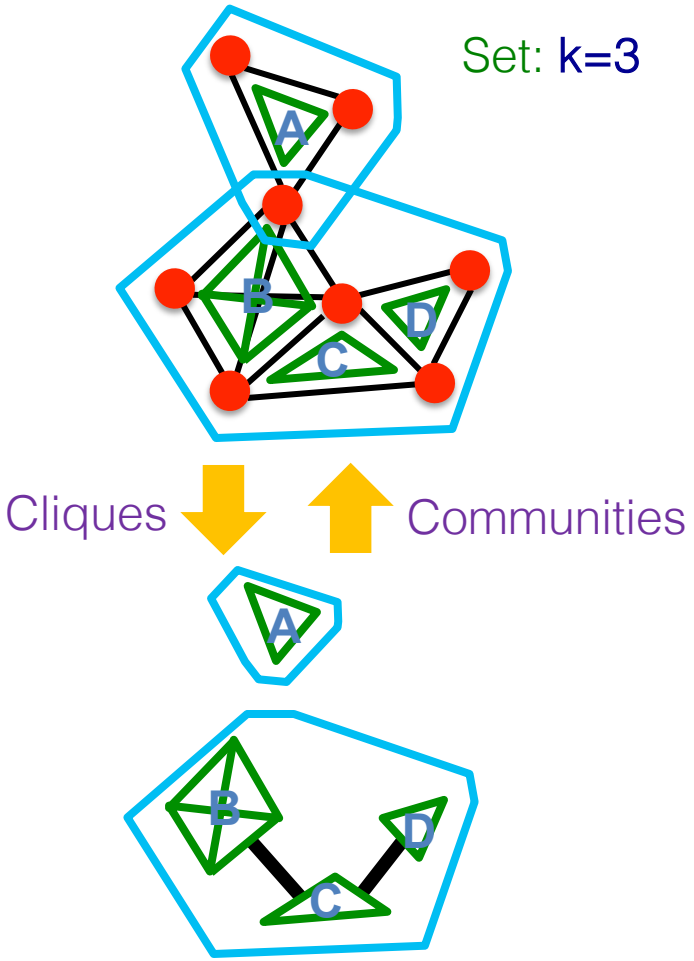
Non-adjacent 4-cliques



Communities for k=4

CPM: Steps

- Clique Percolation Method
  - Find maximal-cliques
    - Def. Clique is maximal if no superset is a clique
  - Clique overlap super-graph:
    - Each clique is a super-node
    - Connect two cliques if they overlap in at least k-1 nodes
  - Communities:
    - Connected components of the clique overlap matrix



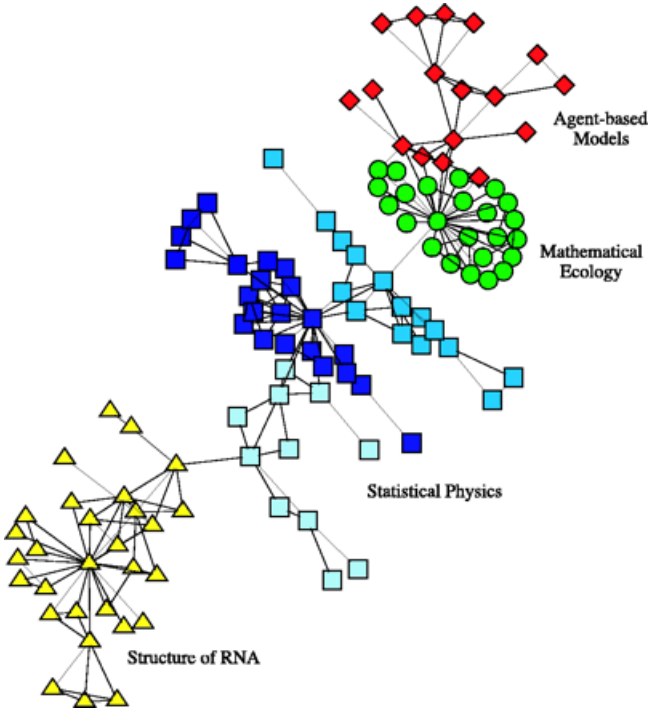
- How to set k ?
  - Set k so that we get the "richest" (most widely distributed cluster sizes) community structure

CPM – Extensions and Discussion

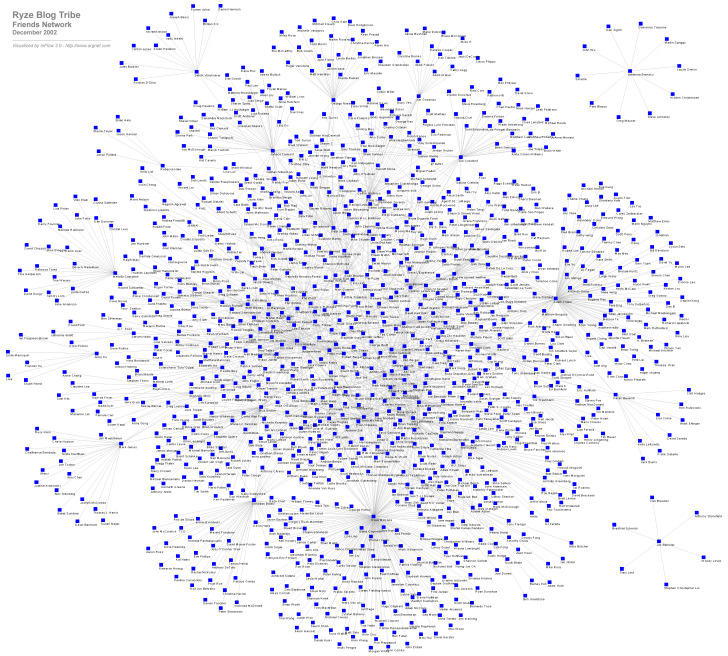
- Several extensions of the Clique Percolation Method
  - Weighted graphs [Farkas et al. '07]
  - Bipartite graphs (overlapping bicliques) [Lehmann et al. '08]
  - Scalable ("fast") implementation of CPM [Kumpula et al. '08]
  - Parallel implementation of CFinder [Pollner et al. '12]
- Drawbacks of CPM [Fortunato '10]
  - Assumes that the graph has a large number of cliques
  - It may fail to detect communities in graphs with a small number of cliques
  - In case of graphs with many cliques → a single community that covers the whole graph
  - How to set parameter k to identify meaningful communities?

Community structure of large-scale graphs

Community Structure in Small vs. Large Graphs



Small scale collaboration network (Newman)



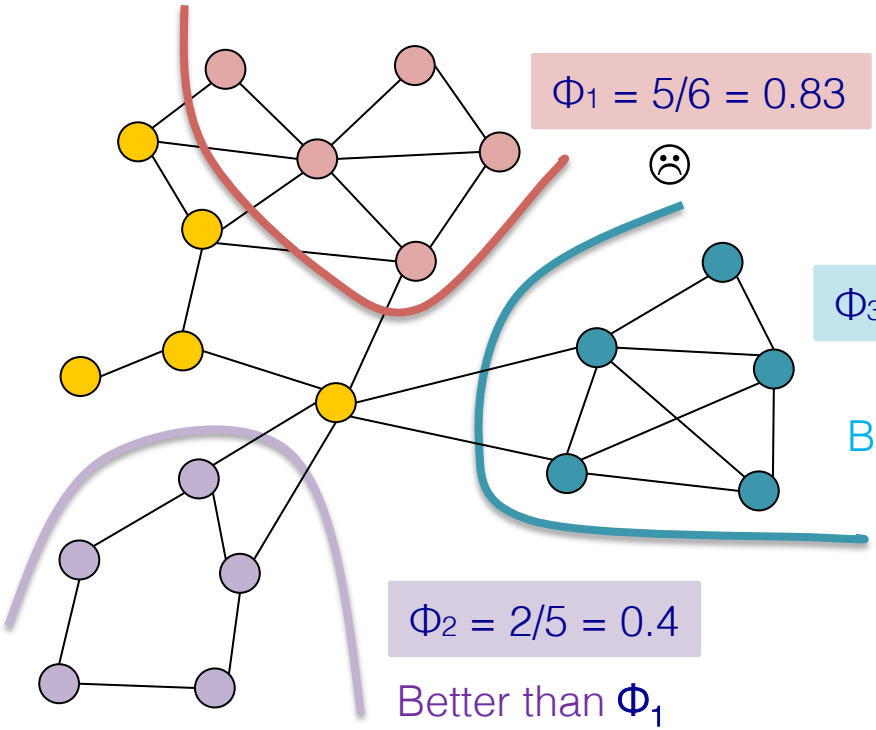
Blog network
http://www.ryze.com

Examine the Structural Differences

- Use conductance Φ(S) as a community evaluation measure
  - Smaller value for conductance implies better community-like properties [Leskovec et al. '09]

Φ(S) = # outgoing edges / # edges within

lower is better



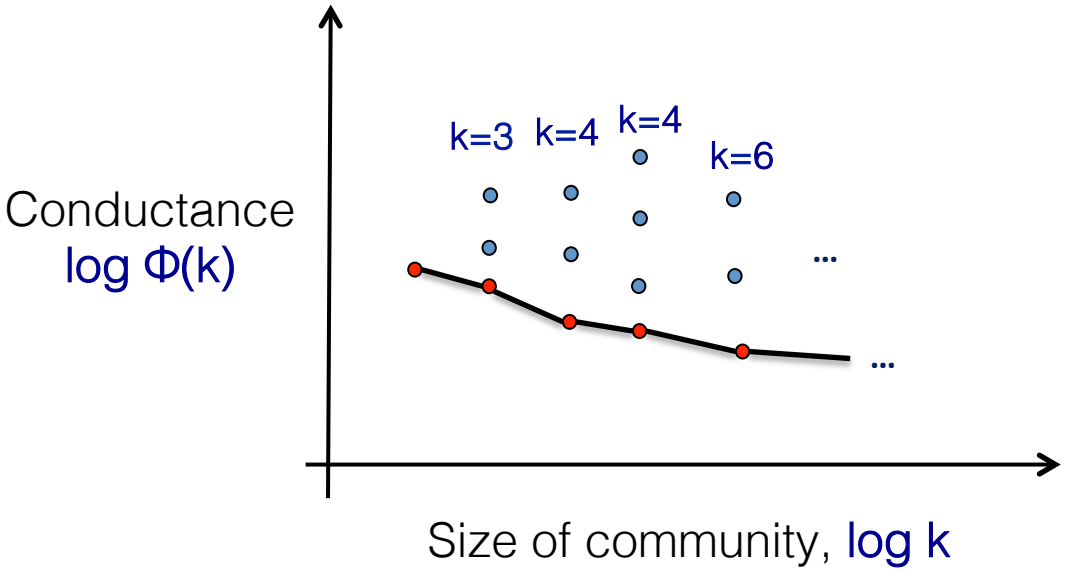
Find the best community of k = 5 nodes

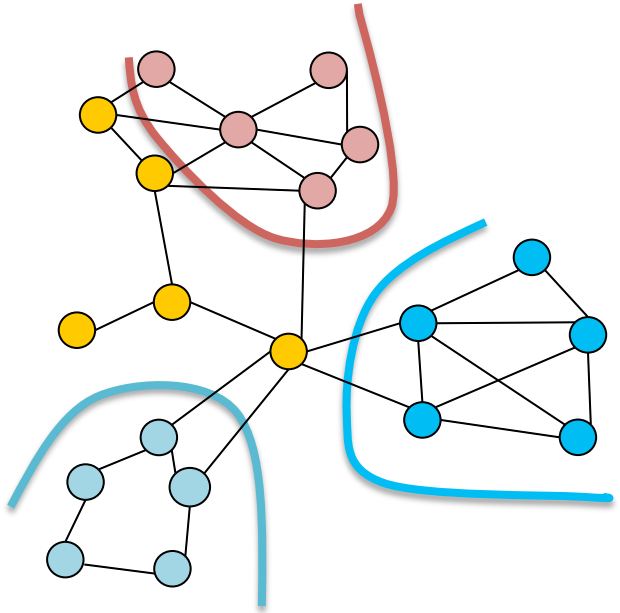
$$\Phi(k) = \min_{S \subset V, |S|=k} \Phi(S)$$

Example by J. Leskovec, ICML 2009

Network Community Profile plot

- Network Community Profile (NCP) plot [Leskovec et al. '09]
  - Plot the best conductance score (minimum) $\Phi(k)$ for each community size k



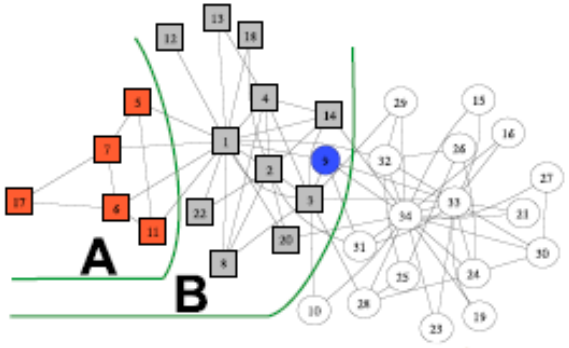


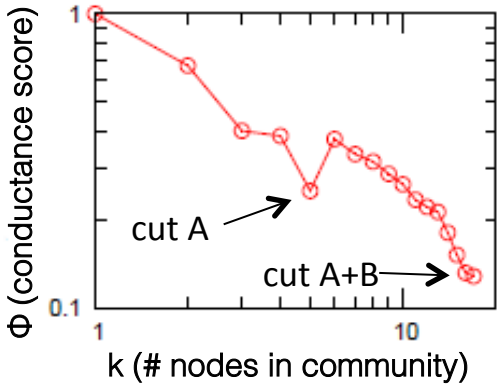
NCP plot of real graphs

?

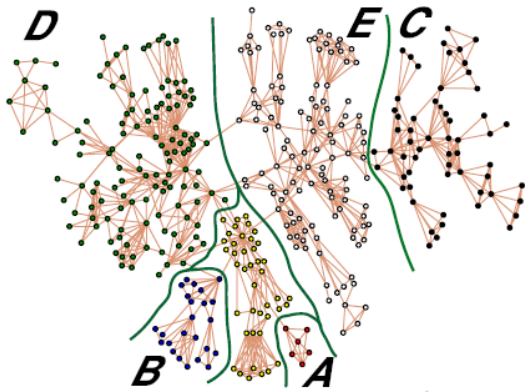
NCP Plot Examples

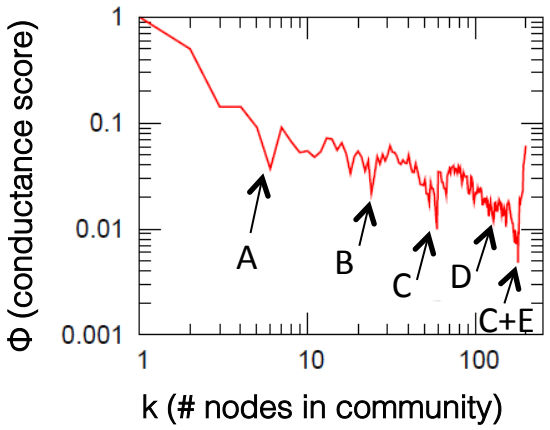
Small scale networks





Zachary's karate club social network

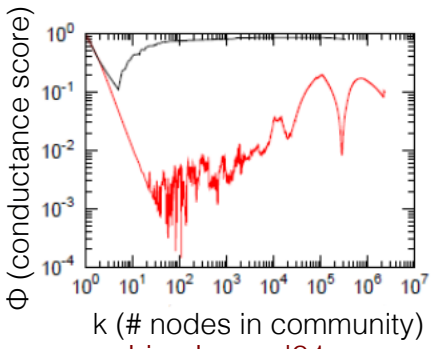




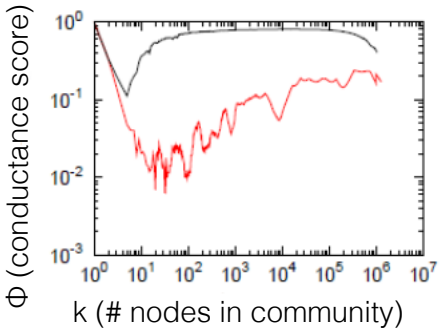
Newman's collaboration network

[Leskovec et al. '09]

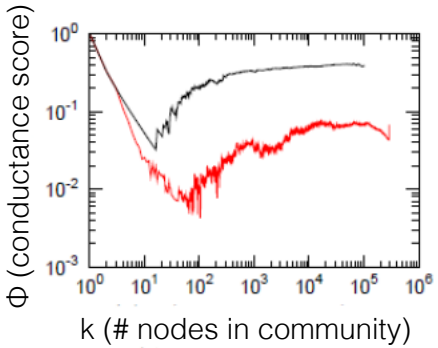
NCP Plot of Large Real Graphs



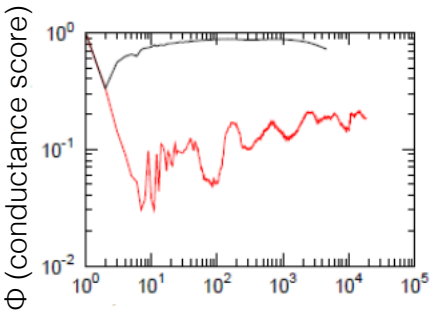
LiveJournal01



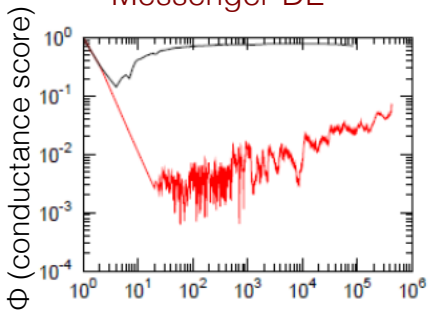
Messenger-DE



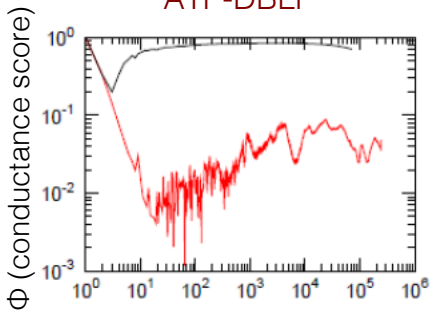
ATP-DBLP



Cit-Hep-Th



Web-Google



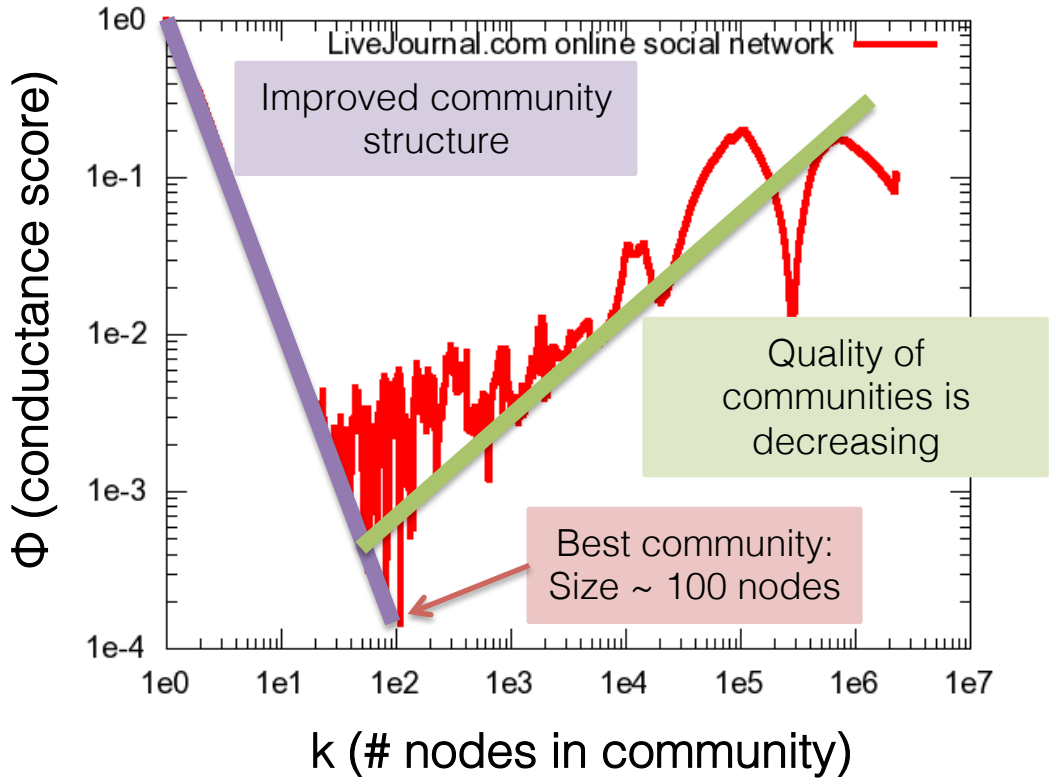
Amazon-All

Large scale networks

Any common property?

[Leskovec et al. '09]

Observation in Large Graphs

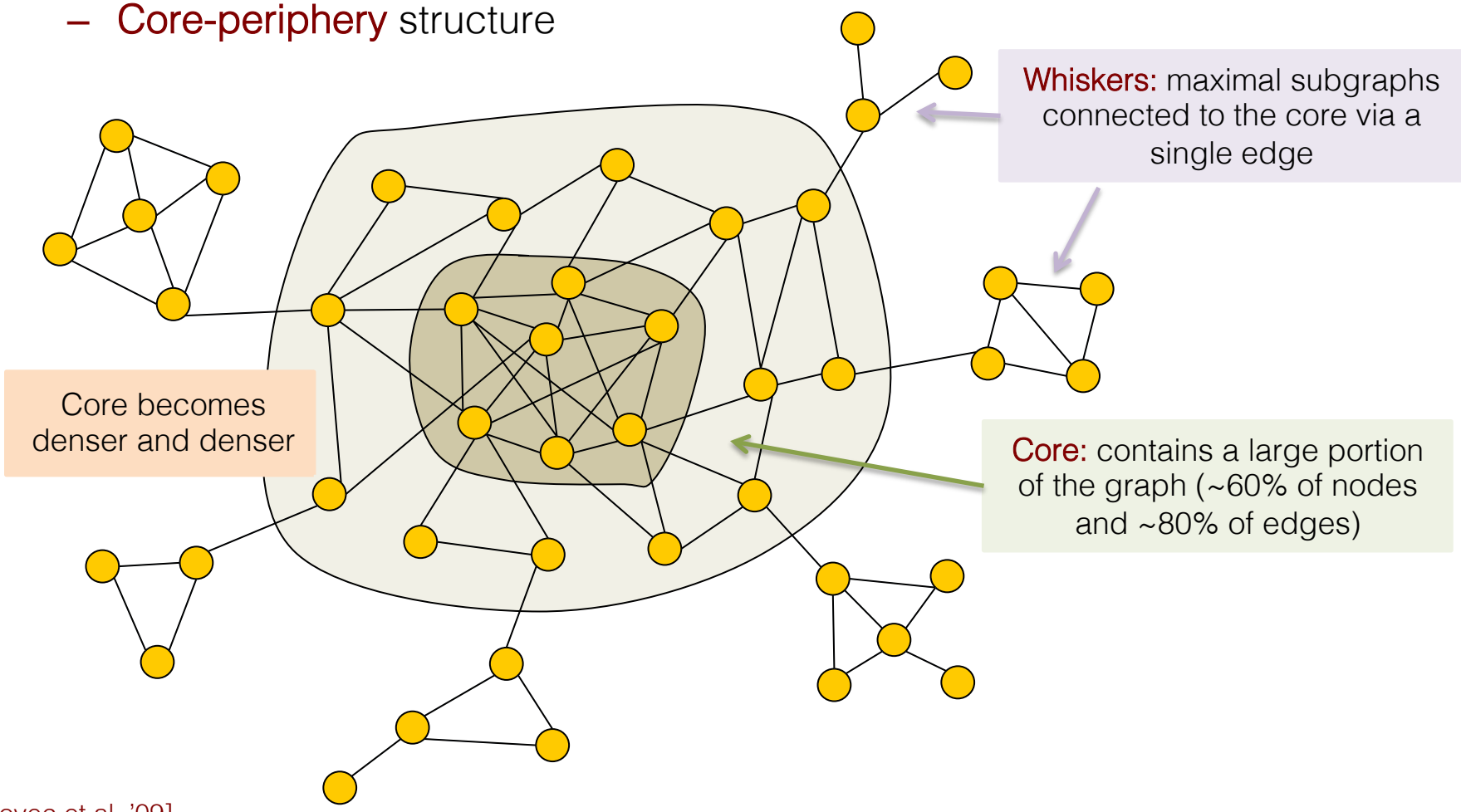


LiveJournal social network
|V| = 5M, |E| = 42M

Figure: http://snap.stanford.edu/ncp/
Slide by J. Leskovec, ICML 2009

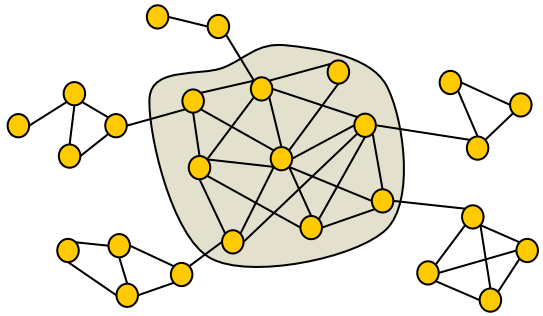
Explanation: Core-Periphery Structure

- How can we explain the observed structure of large graphs?
  - Core-periphery structure



[Leskovec et al. '09]

Core-Periphery Structure



- Core-periphery structure
  - Core
  - Whiskers
- Whiskers
  - Non-trivial structure → more than random (shape and size)
- Q: What is happening if we remove whiskers (periphery) from graphs?
  - Almost nothing. The whiskers are replaced by 2-whiskers (subgraphs connected to the core with 2 edges)
- The core itself has core-periphery structure
- Important point: Whiskers are also responsible for the best communities in large graphs (lowest point of NCP plot)

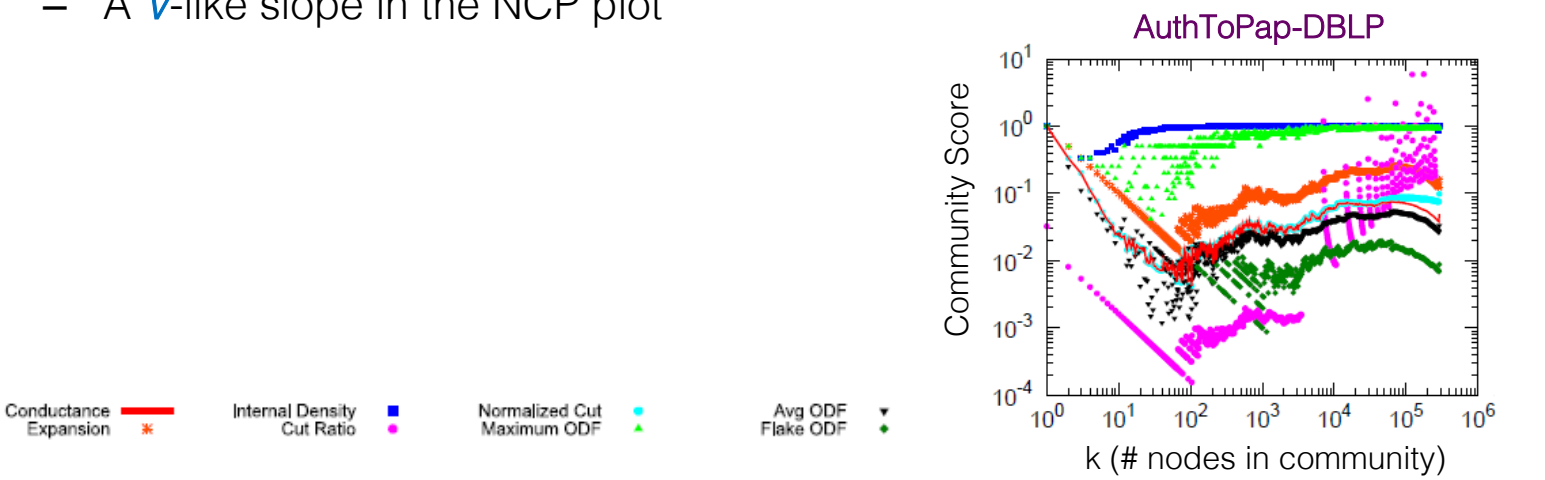
Match the observations made using the properties of the Kronecker graphs

Similar Structural Observations

- Jellyfish model for the Internet topology [Tauro et al. '01]
- Min-cut plots [Chakrabarty et al. '04]
  - Perform min-cut recursively
  - Plot the relative size of the minimum cut
- Robustness of large scale social networks [Malliaros, Megalooikonomou, Faloutsos '12, '15]
  - Robustness estimation based on the expansion properties of graphs
  - Social networks are expected to have low robustness due to the existence of communities → the (small number of) inter-community edges will act as bottlenecks
  - Large scale social graphs tend to be extremely robust
  - Structural differences (in terms of robustness and community structure) between different scale graphs

Clustering Algorithms and Objective Criteria

- Question 1: Is the observed property an effect of the used community detection algorithm (Metis + flow based method)?
  - A: No. The qualitative shape of the NCP plot is the same, regardless of the community detection algorithm [Leskovec et al. '09]
- Question 2: Is the observed property an effect of the conductance community evaluation measure?
  - A: No. All the objective criteria that based on both internal and external connectivity, show an almost similar qualitatively behavior [Leskovec et al. '10]
  - A *V*-like slope in the NCP plot



Conclusions

- Large scale real-world graphs
  - Core-periphery structure
  - No large, well defined communities
  - Structural differences between different scale graphs

- Community detection algorithms should take into account these structural observation
  - Whiskers correspond to the best (conductance-based) communities
  - Need larger high-quality clusters?
  - Bag of whiskers: union of disjoint (disconnected) whiskers are mainly responsible for the best high-quality clusters of larger size (above 100)

Survey Articles

- S. Fortunato. Community detection in graphs. Physics Reports 486, 75-174, 2010
- S. E. Schaeffer. Graph clustering. Computer Science Review, 1(1): 27–64, 2007
- F. D. Malliaros and M. Vazirgiannis. Clustering and community detection in directed networks: A survey. Physics Reports 533, 95-142, 2013
- S. Fortunato and D. Hric. Community detection in networks: a user guide. Physics Reports 659, 1-44, 2016
- S. Papadopoulos, Y. Kompatsiaris, A. Vakali, and P. Spyridonos. Community detection in Social Media: Performance and application considerations. Data Min Knowl Disc, 24:515–554, 2012

Summary

- Clustering and community detection algorithms
  - Modularity optimization
  - Spectral clustering
  - Communities in directed networks
  - Observations about the community structure of the graph

Next Lecture

- Node similarity
- Link prediction

Thank You!