Community Structure in Networks

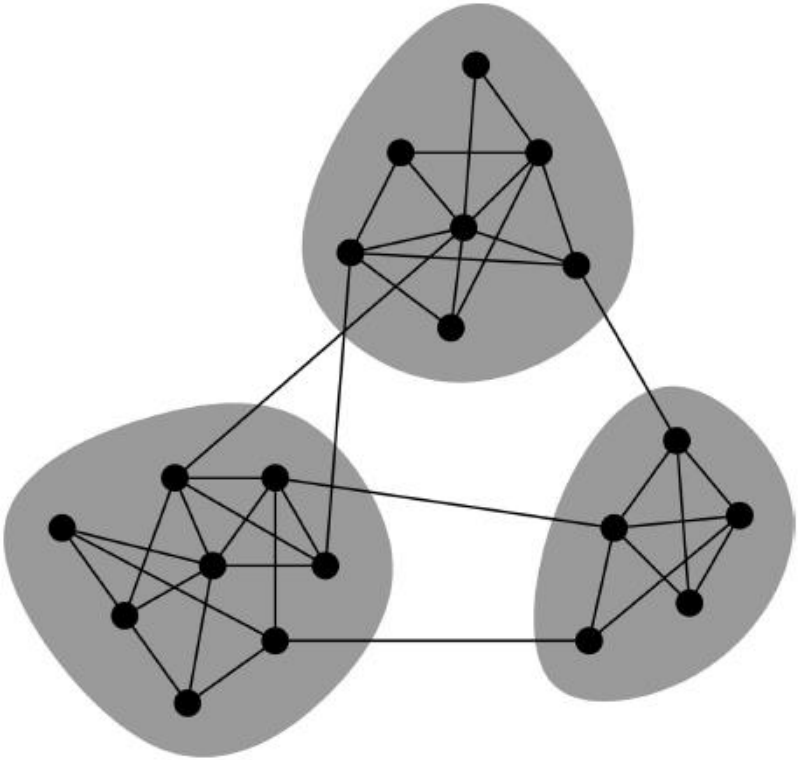
Thanks to Jure Leskovec, Stanford and Panayiotis Tsaparas, Univ. of Ioannina for slides

Agenda

- Strength of Weak Ties
- Structural Holes
- Network Communities
- Community Detection
  - Method 1: Girvan-Newman
  - Method 2: Modularity Optimization

Networks & Communities

- We often think of networks "looking" like this:



- What lead to such a conceptual picture?

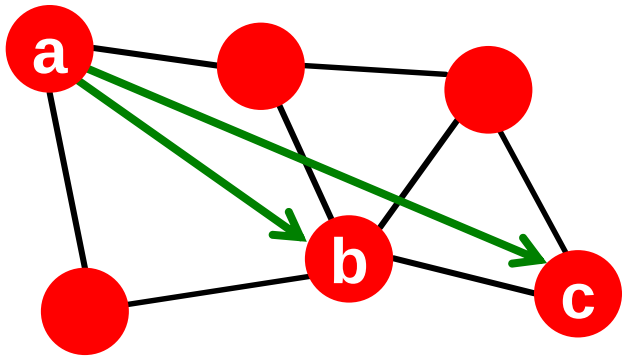
Networks: Flow of Information

- **How information flows through the network?**
  - **What structurally distinct roles do nodes play?**
  - **What roles do different links (short vs. long) play?**
- **How people find out about new jobs?**
  - Mark Granovetter, part of his PhD in 1960s
  - People find the information through personal contacts
- **But:** Contacts were often **acquaintances** rather than close friends
  - **This is surprising:** One would expect your friends to help you out more than casual acquaintances
- **Why is it that acquaintances are most helpful?**

[Granovetter '73]

Granovetter's Answer

- **Two perspectives on friendships:**
  - **Structural:** Friendships that span different parts of the network
  - **Interpersonal:** Friendship between two people is either **strong** or **weak**
- **Structural role: Triadic Closure**

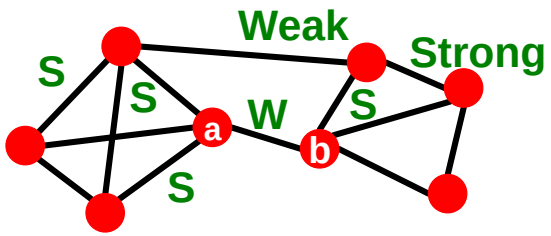


Which edge is more likely, a-b or a-c?

If two people in a network have a friend in common, then there is an increased likelihood they will become friends themselves.

Granovetter's Explanation

- **Granovetter makes a connection between social and structural role of an edge**
- **First point: Structure**
  - Structurally embedded edges are socially **strong**
  - Long-range edges spanning different parts of the network are socially **weak**
- **Second point: Information**
  - Long-range edges allow you to gather information from different parts of the network and get a job
  - Structurally embedded edges are heavily redundant in terms of information access

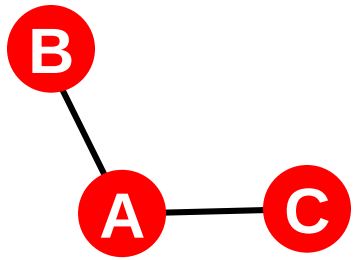


Triadic Closure

- **Triadic closure == High clustering coefficient**

Reasons for triadic closure:

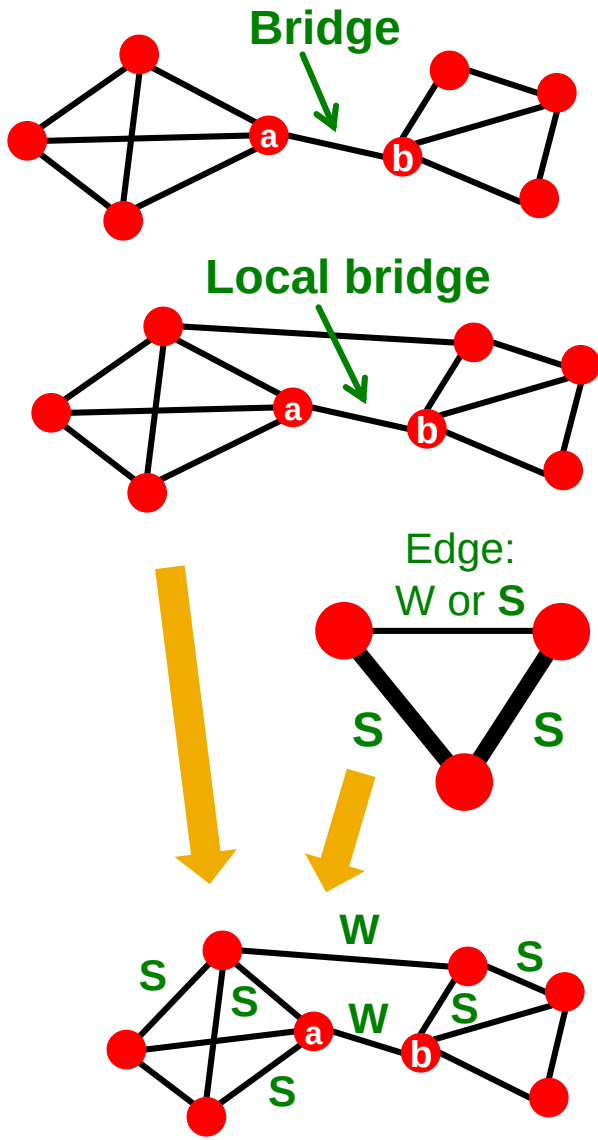
- If B and C have a friend A in common, then:
  - B is **more likely to meet** C
    - (since they both spend time with A)
  - B and C **trust** each other
    - (since they have a friend in common)
  - A has **incentive** to bring B and C together
    - (as it is hard for A to maintain two disjoint relationships)



- **Empirical study by Bearman and Moody:**
  - Teenage girls with low clustering coefficient are more likely to contemplate suicide

Granovetter's Explanation

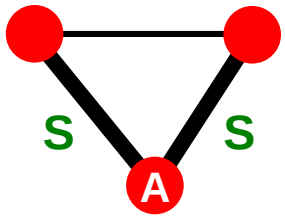
- Define: **Bridge edge**
  - If removed, it disconnects the graph
- Define: **Local bridge**
  - Edge of **Span > 2**
    (**Span** of an edge is the distance of the edge endpoints if the edge is deleted. Local bridges with long span are like real bridges)
- Define: Two types of edges:
  - **Strong** (friend), **Weak** (acquaintance)
- Define: **Strong triadic closure:**
  - Two strong ties imply a third edge
- **Fact:** If strong triadic closure is satisfied then **local bridges are weak ties!**

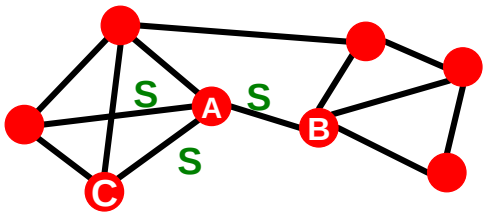


Local Bridges and Weak ties

- **Claim:** If node A satisfies **Strong Triadic Closure** and is involved in at least **two strong ties**, then any **local bridge** adjacent to A must be a **weak tie**.

- **Proof by contradiction:**
  - Assume A satisfies **Strong Triadic Closure** and has **2 strong ties**
  - Let $A - B$ be **local bridge** and a **strong** tie
  - Then $B - C$ must exist because of **Strong Triadic Closure**
  - But then $A - B$ is **not a bridge!** (since **B-C** must be connected due to Strong Triadic Closure property)





Tie strength in real data

- **For many years Granovetter's theory was not tested**
- But, today we have large who-talks-to-whom graphs:
  - Email, Messenger, Cell phones, Facebook
- **Onnela et al. 2007:**
  - Cell-phone network of 20% of country's population
  - **Edge strength:** # phone calls

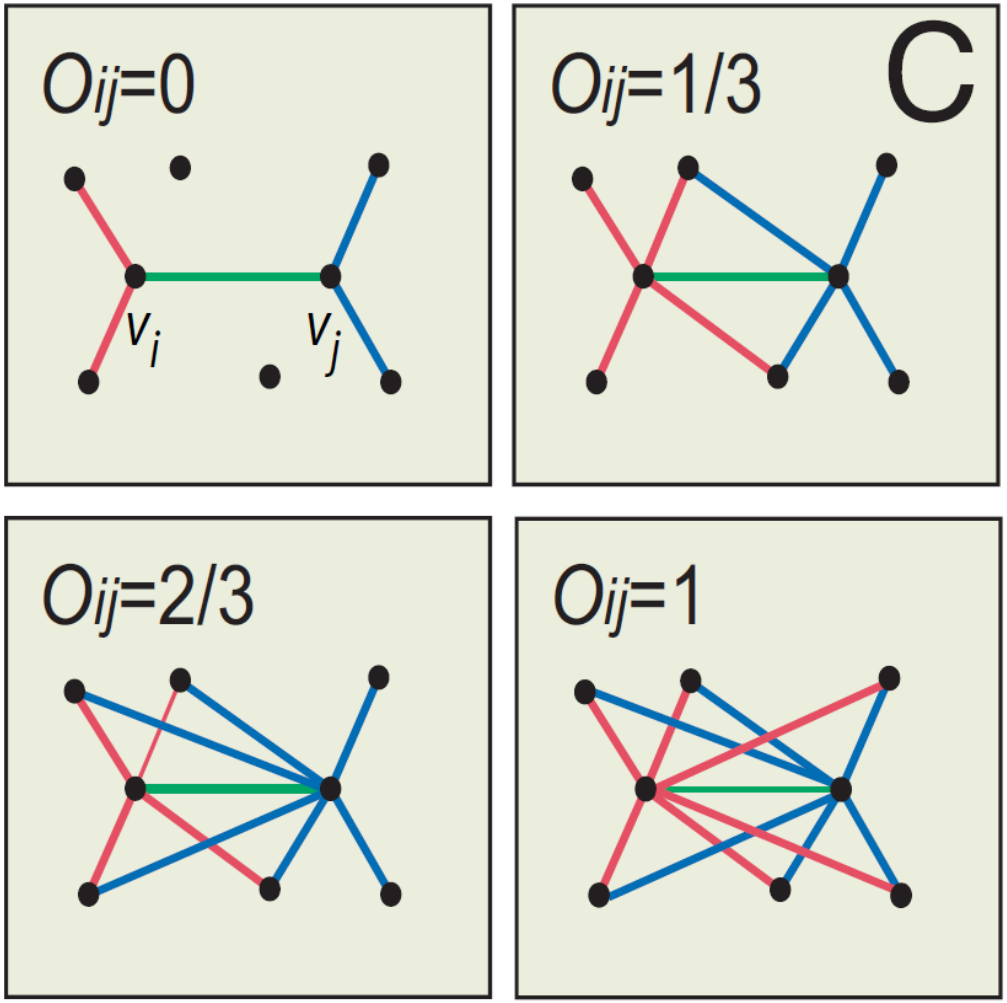
Neighborhood Overlap

- **Edge overlap:**

$$O_{ij} = \frac{N(i) \cap N(j)}{N(i) \cup N(j)}$$

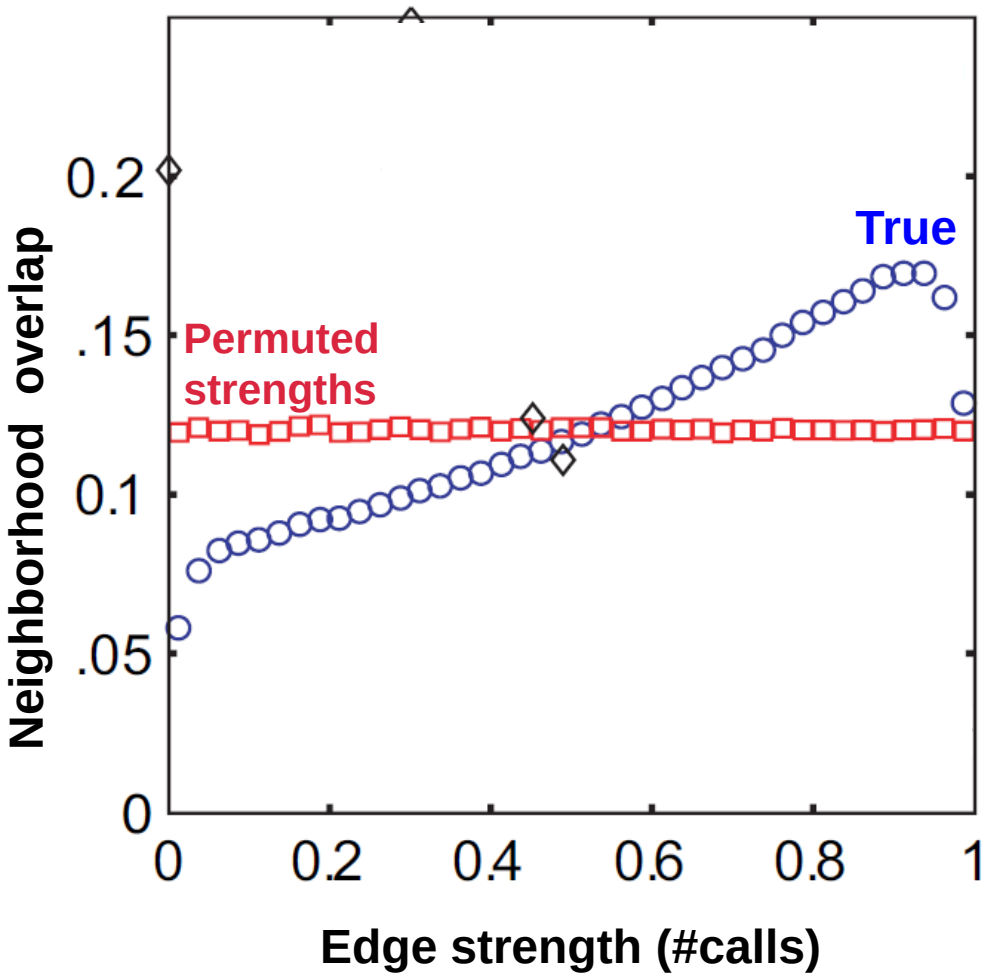
  - $N(i)$ … a set of neighbors of node i

- **Overlap = 0** when an edge is a **local bridge**

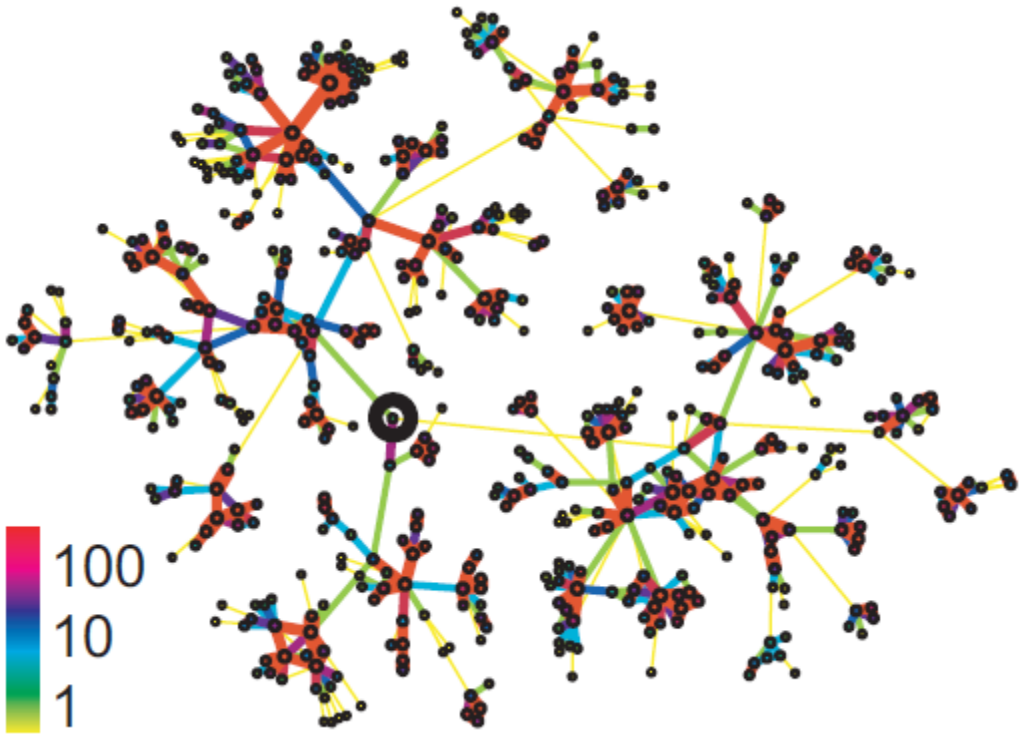


Phones: Edge Overlap vs. Strength

- **Cell-phone network**
- **Observation:**
  - **Highly used links have high overlap!**
- **Legend:**
  - **True:** The data
  - **Permuted strengths:** Keep the network structure but randomly reassign edge strengths

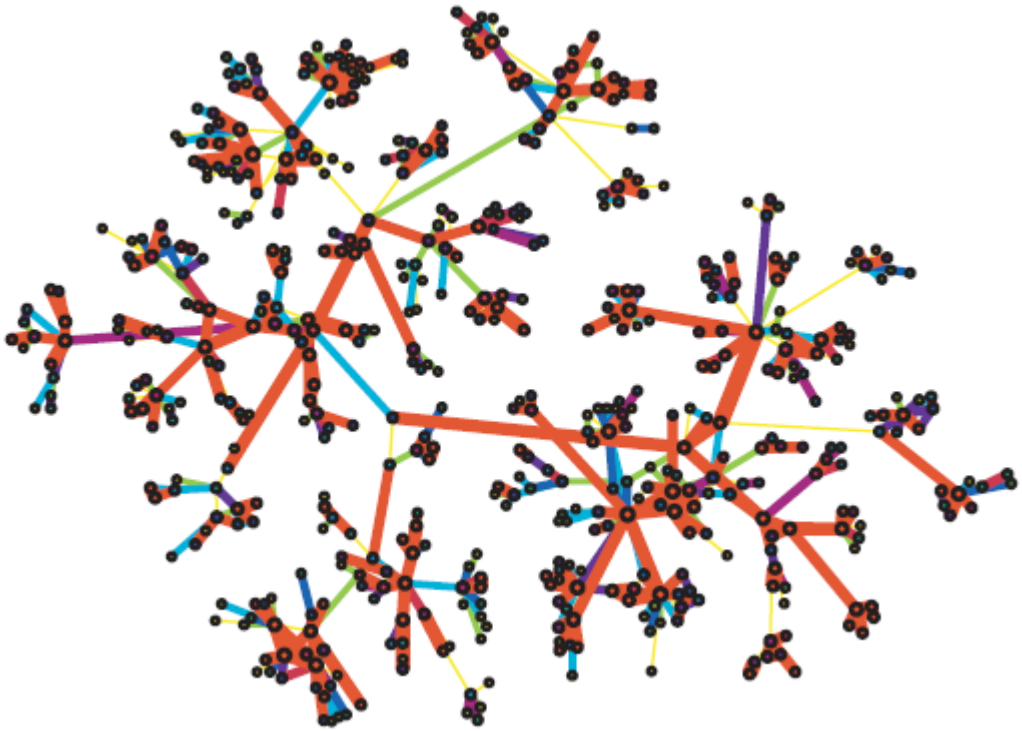


Real Network, Real Tie Strengths



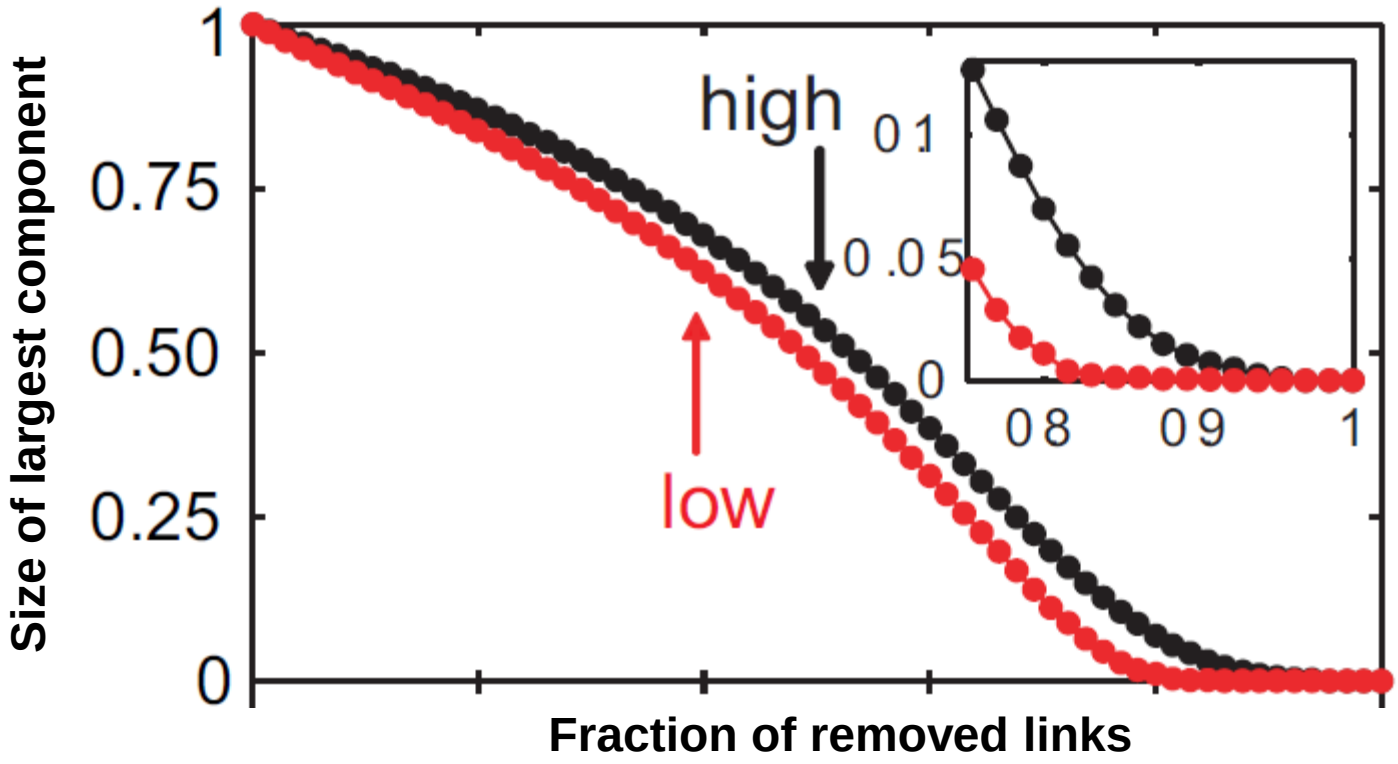
- **Real edge strengths in mobile call graph**
  - Strong ties are more embedded (have higher overlap)

Real Net, Permuted Tie Strengths



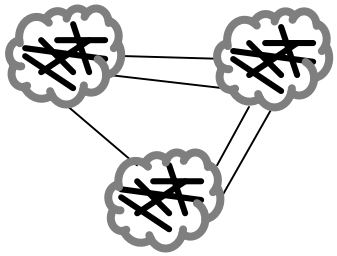
- **Same network, same set of edge strengths** but now **strengths are randomly shuffled**

Link Removal by Strength



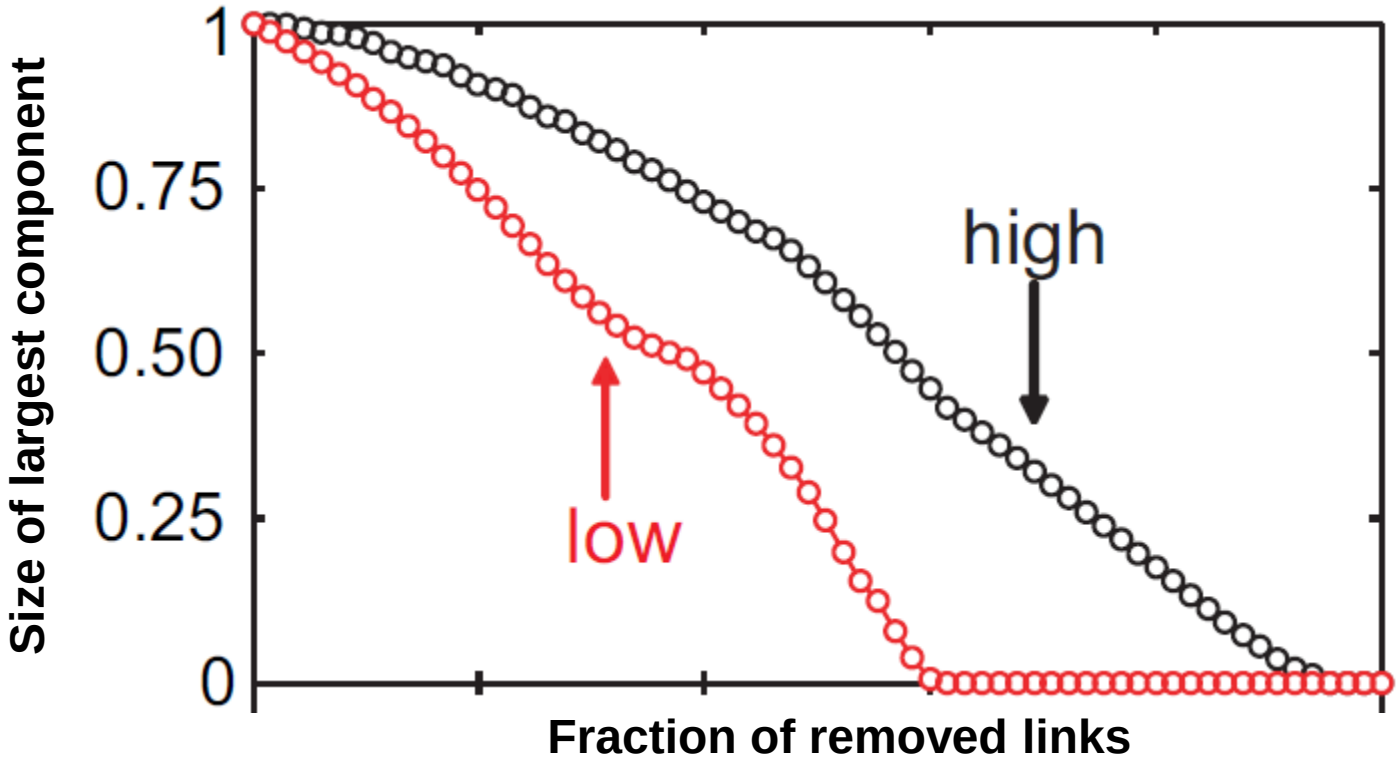
Low disconnects the network sooner

- Removing links by **strength (#calls)**
  - Low to high
  - High to low



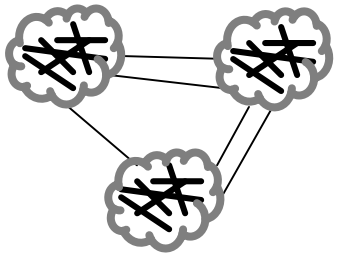
Conceptual picture of network structure

Link Removal by Overlap



Low disconnects the network sooner

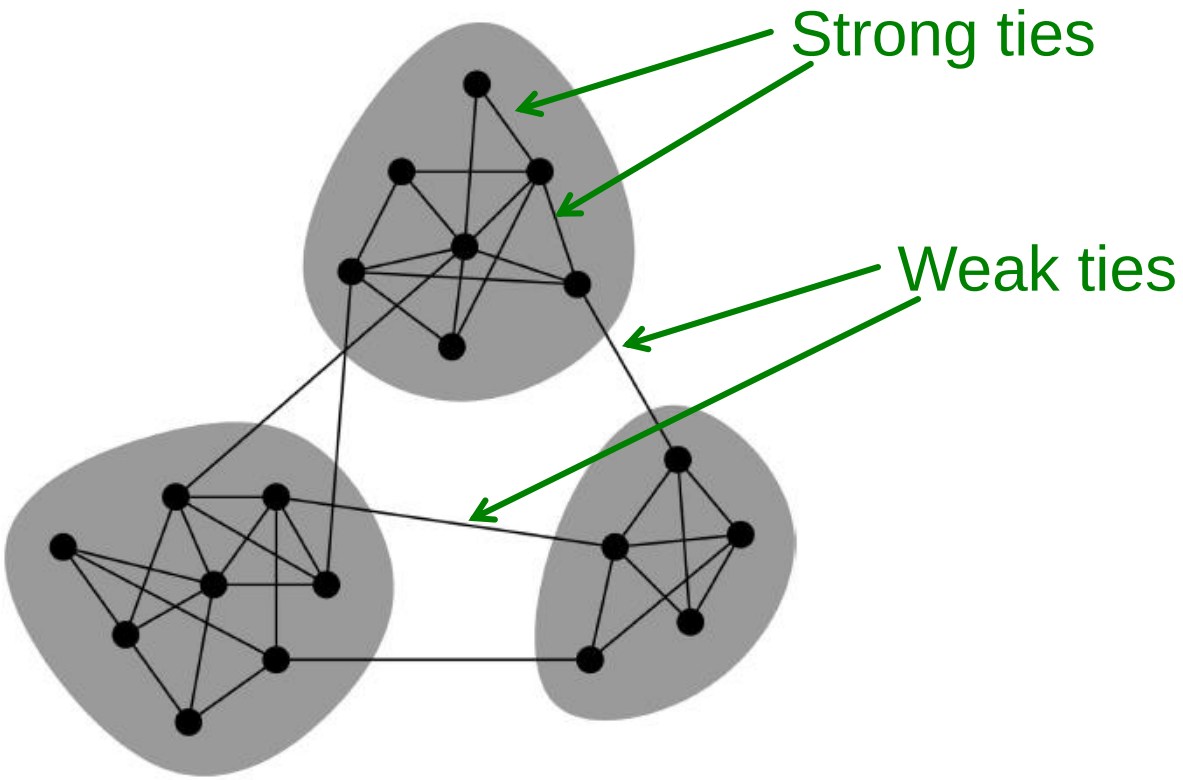
- Removing links based on **overlap**
  - Low to high
  - High to low



Conceptual picture of network structure

Conceptual Picture of Networks

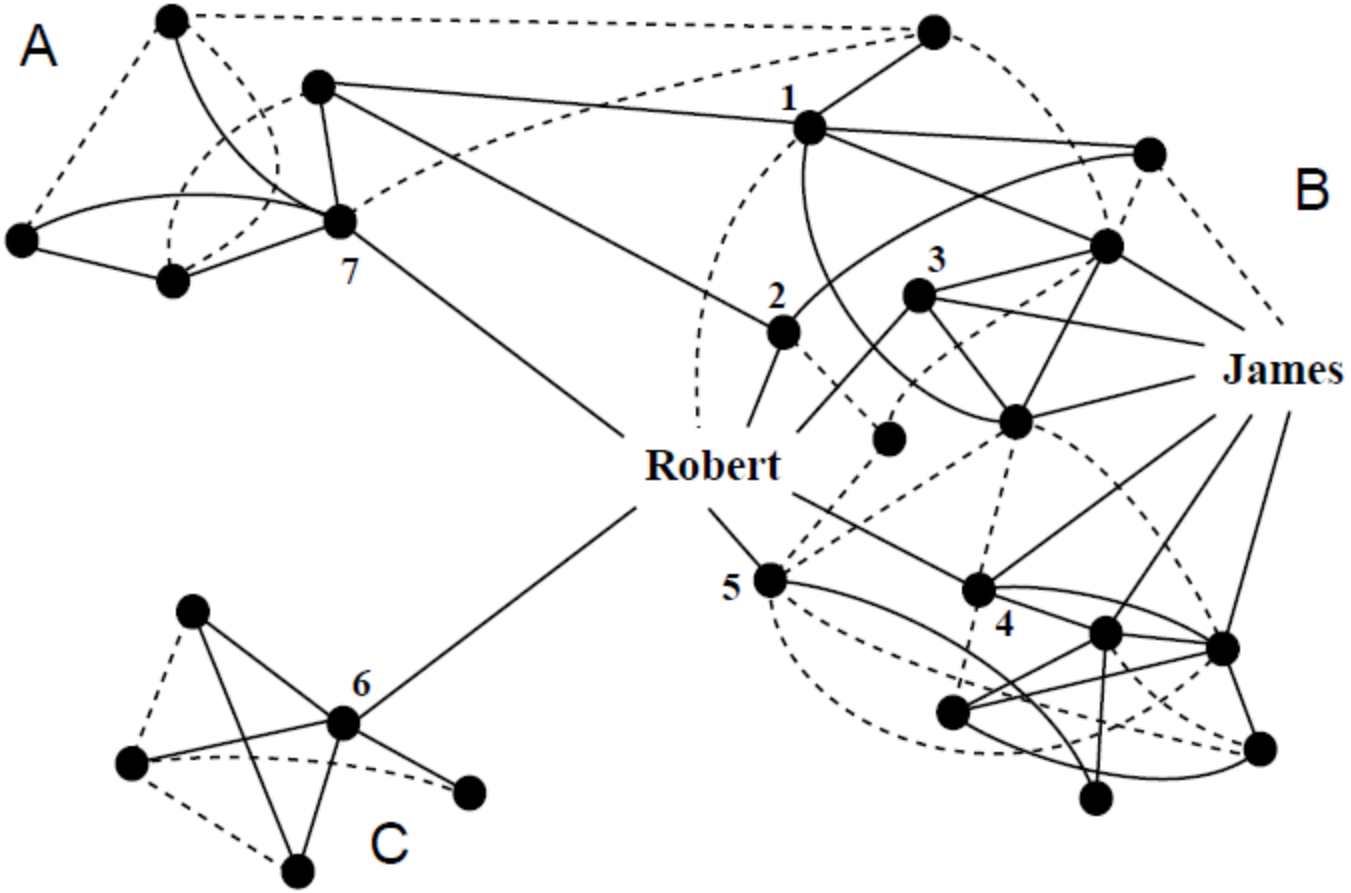
- **Granovetter's theory leads to the following conceptual picture of networks**



Small Detour:
Structural Holes

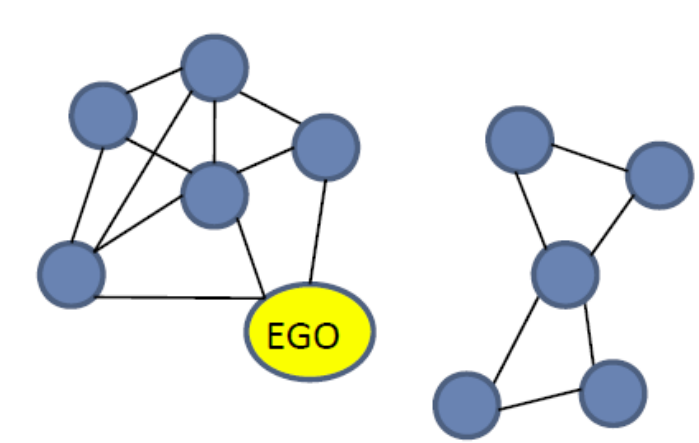
[Ron Burt]

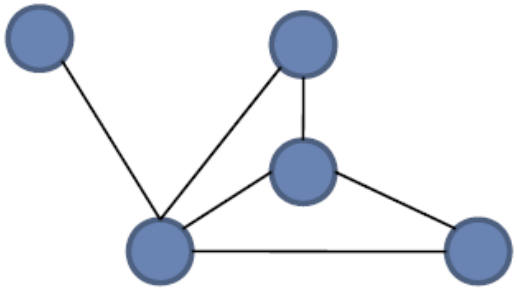
Small Detour: Structural Holes



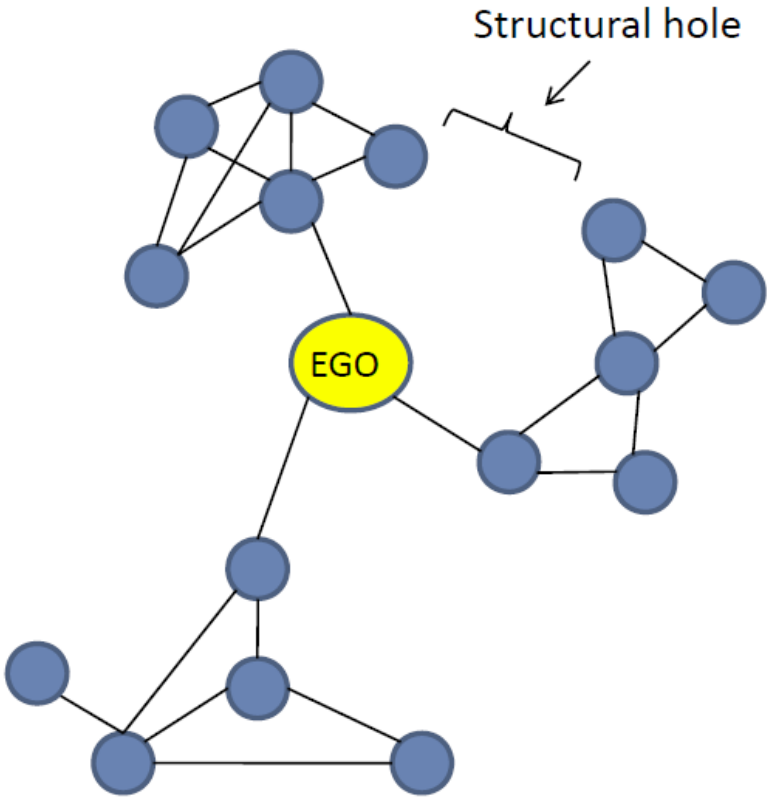
Who is better off, Robert or James?

Structural Holes





Few structural holes

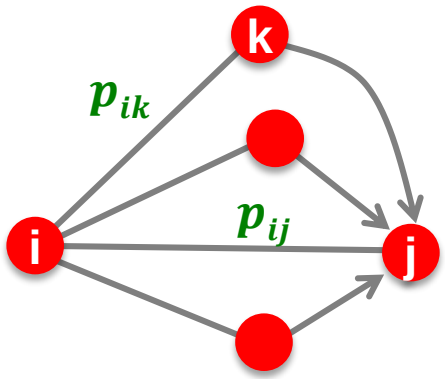


Many structural holes

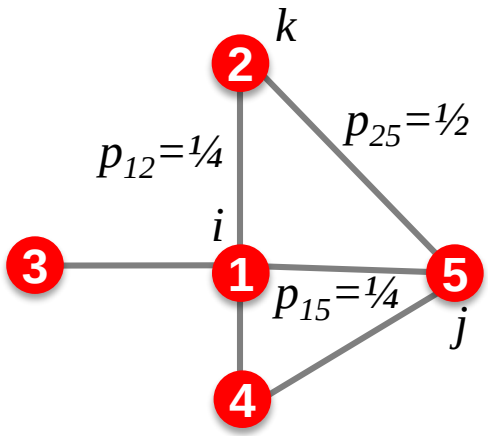
Structural Holes provide ego with access to novel information, power, freedom

Structural Holes: Network Constraint

- **The "network constraint" measure** [Burt]:
  - **To what extent are person's contacts redundant**



$$p_{uv} = 1/d_u$$



p_{uv}

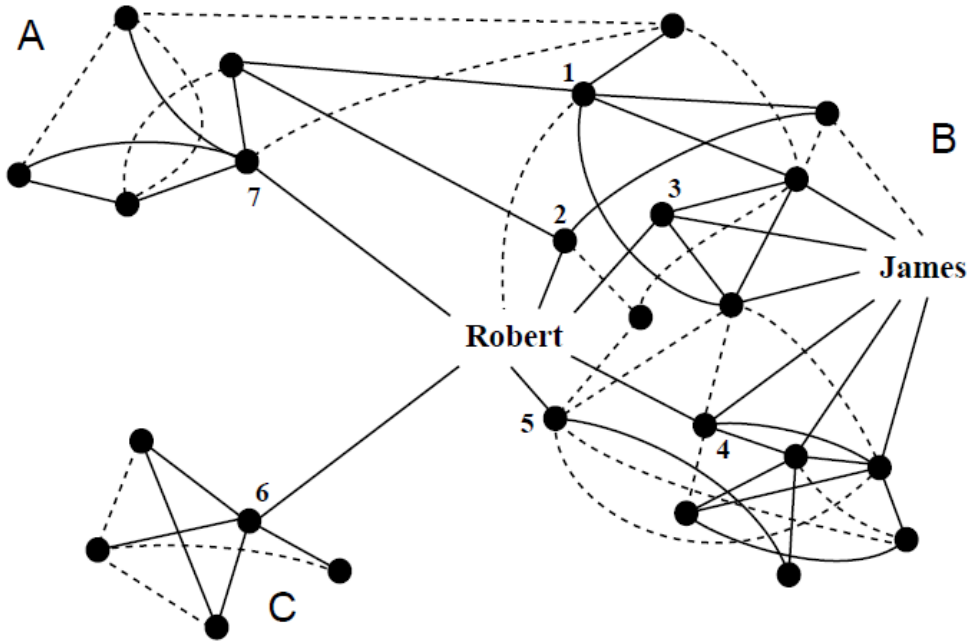
	1	2	3	4	5
1	.00	.25	.25	.25	.25
2	.50	.00	.00	.00	.50
3	1.0	.00	.00	.00	.00
4	.50	.00	.00	.00	.50
5	.33	.33	.00	.33	.00

  - **Low**: disconnected contacts
  - **High**: contacts that are close or strongly tied

$$c_i = \sum_j c_{ij} = \sum_j \left[p_{ij} + \sum_k \left(p_{ik} p_{kj} \right) \right]^2$$

p_{uv} … prop. of u's "energy" invested in relationship with v

Example: Robert vs. James



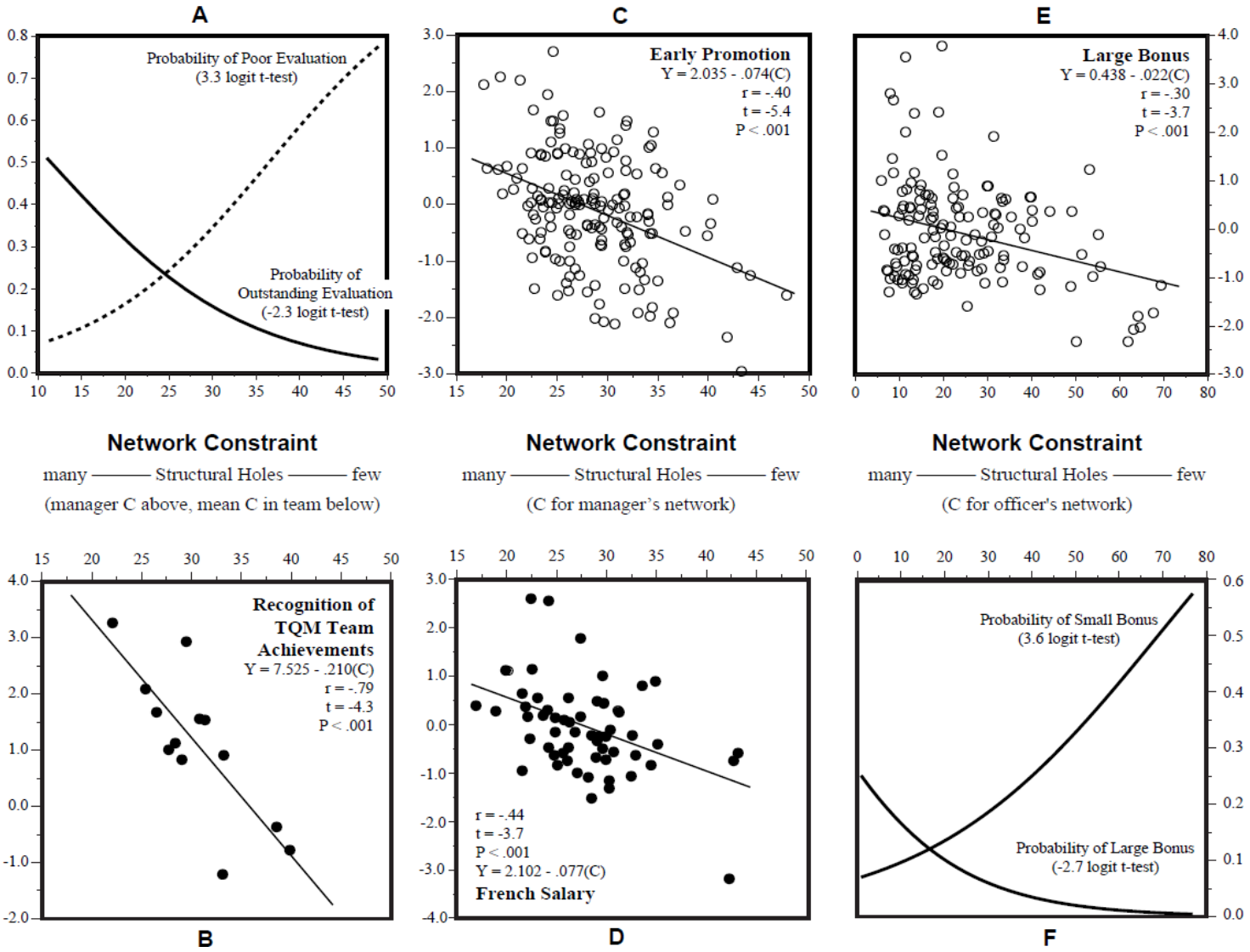
- **Network constraint:**
  - James: $c_J = 0.309$
  - Robert: $c_R = 0.148$

- **Constraint:** To what extent are person's contacts redundant
  - **Low:** disconnected contacts
  - **High:** contacts that are close or strongly tied

Lower *c* is better!

[Ron Burt]

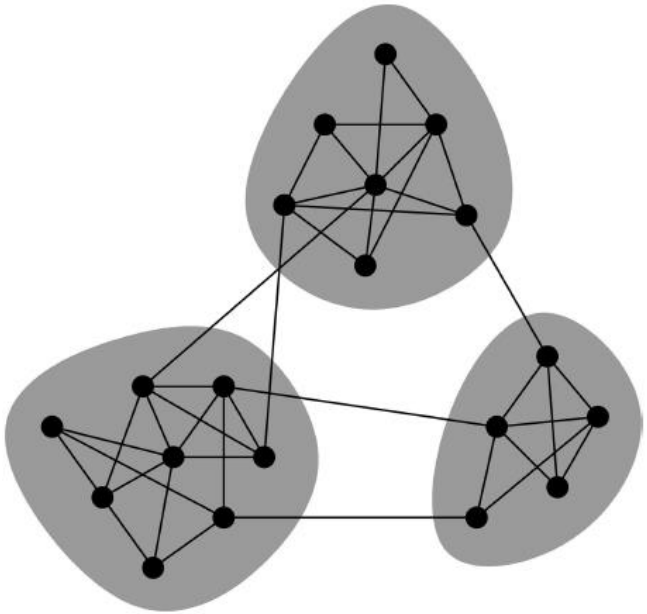
Spanning Holes Matters



Network Communities

Network Communities

- Granovetter's theory suggest that networks are composed of **tightly connected sets of nodes**

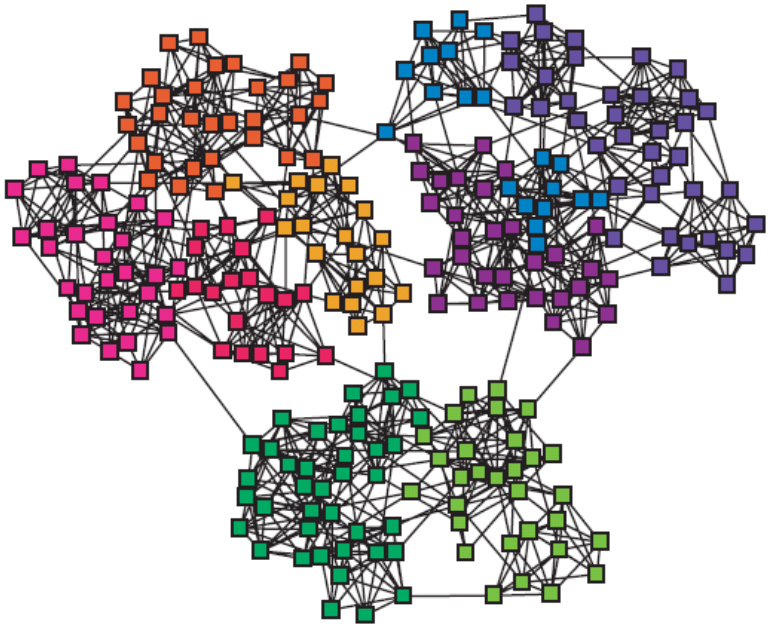


- **Network communities:**
  - Sets of nodes with **lots** of connections **inside** and **few** to **outside** (the rest of the network)

Communities, clusters, groups, modules

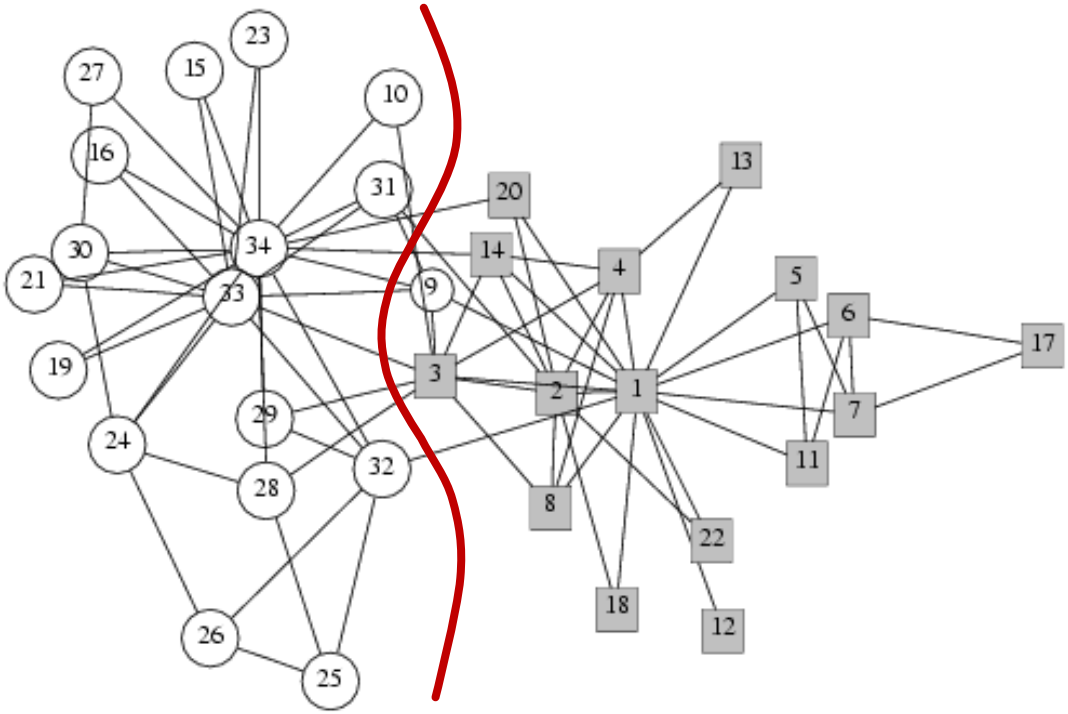
Finding Network Communities

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



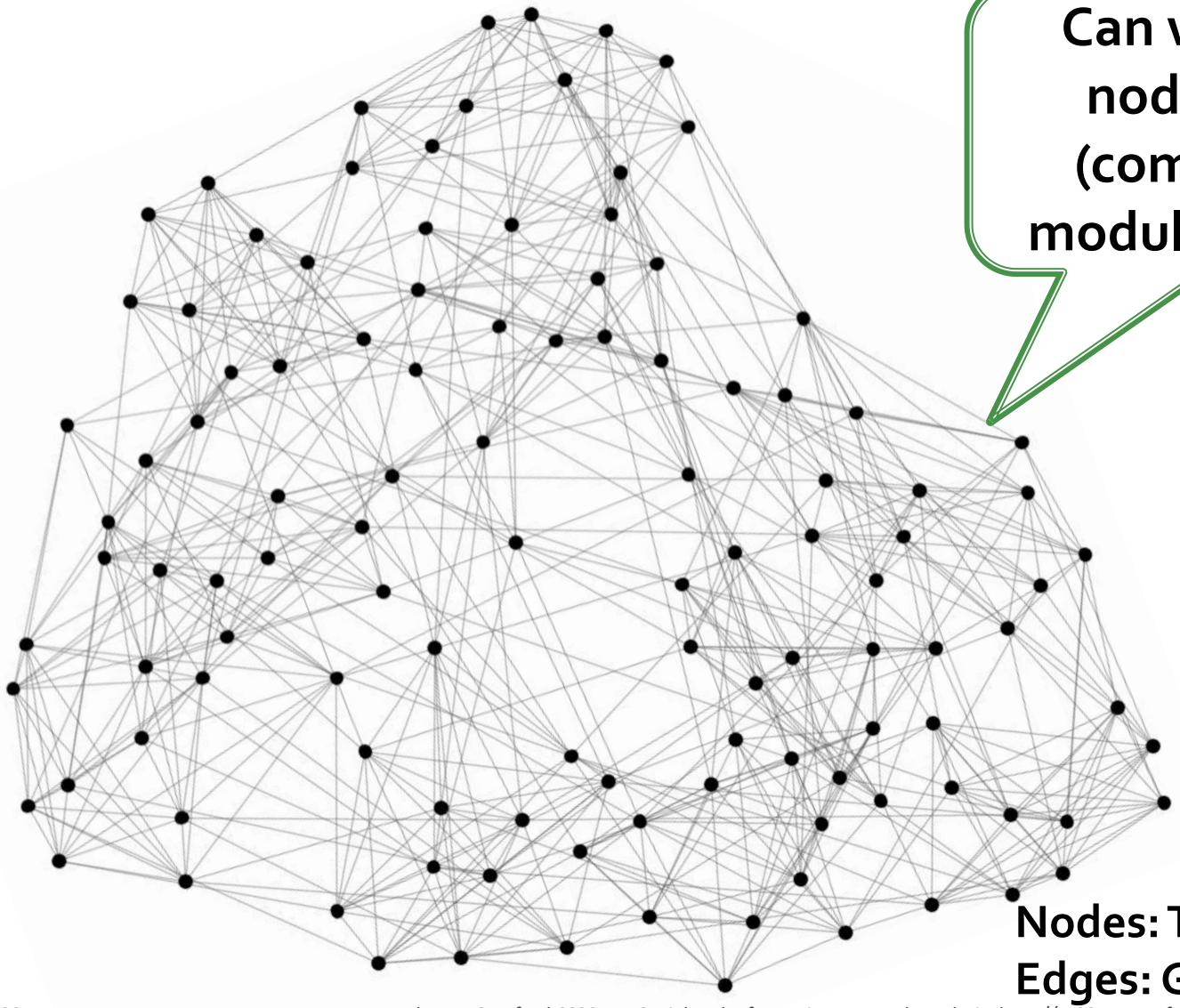
Communities, clusters, groups, modules

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

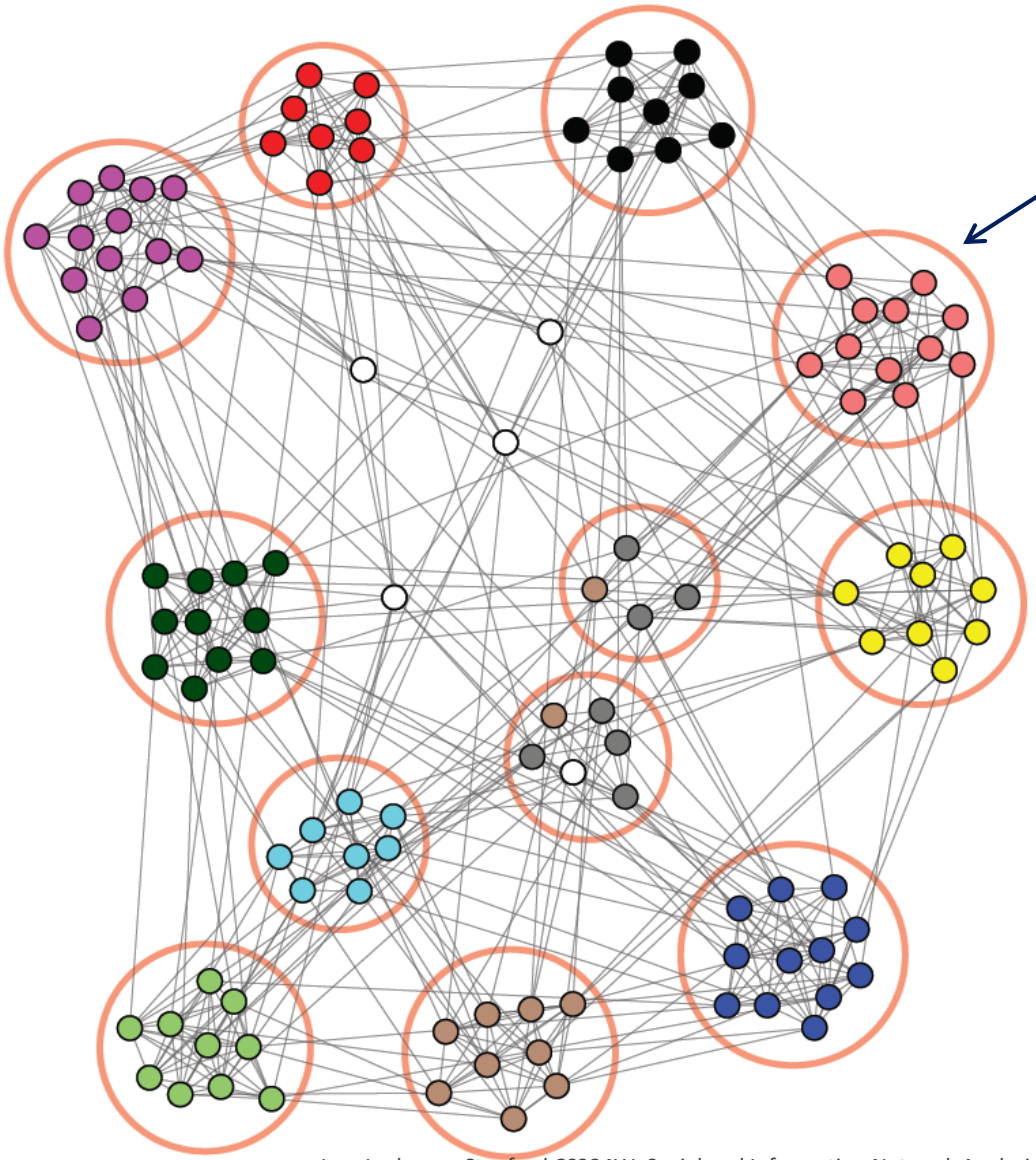
NCAA Football Network



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

Nodes: Teams
Edges: Games played

NCAA Football Network

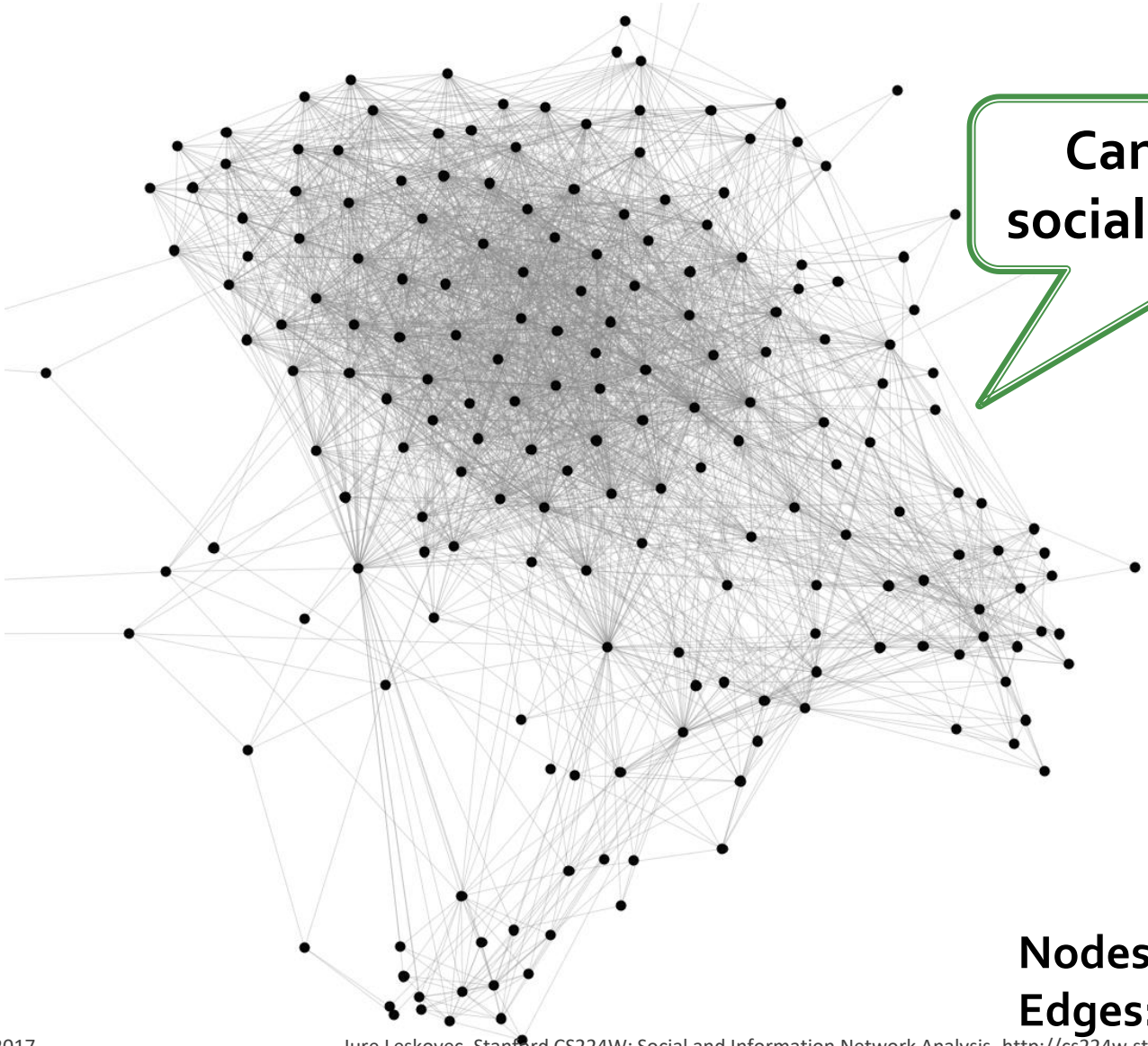


NCAA conferences

- 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

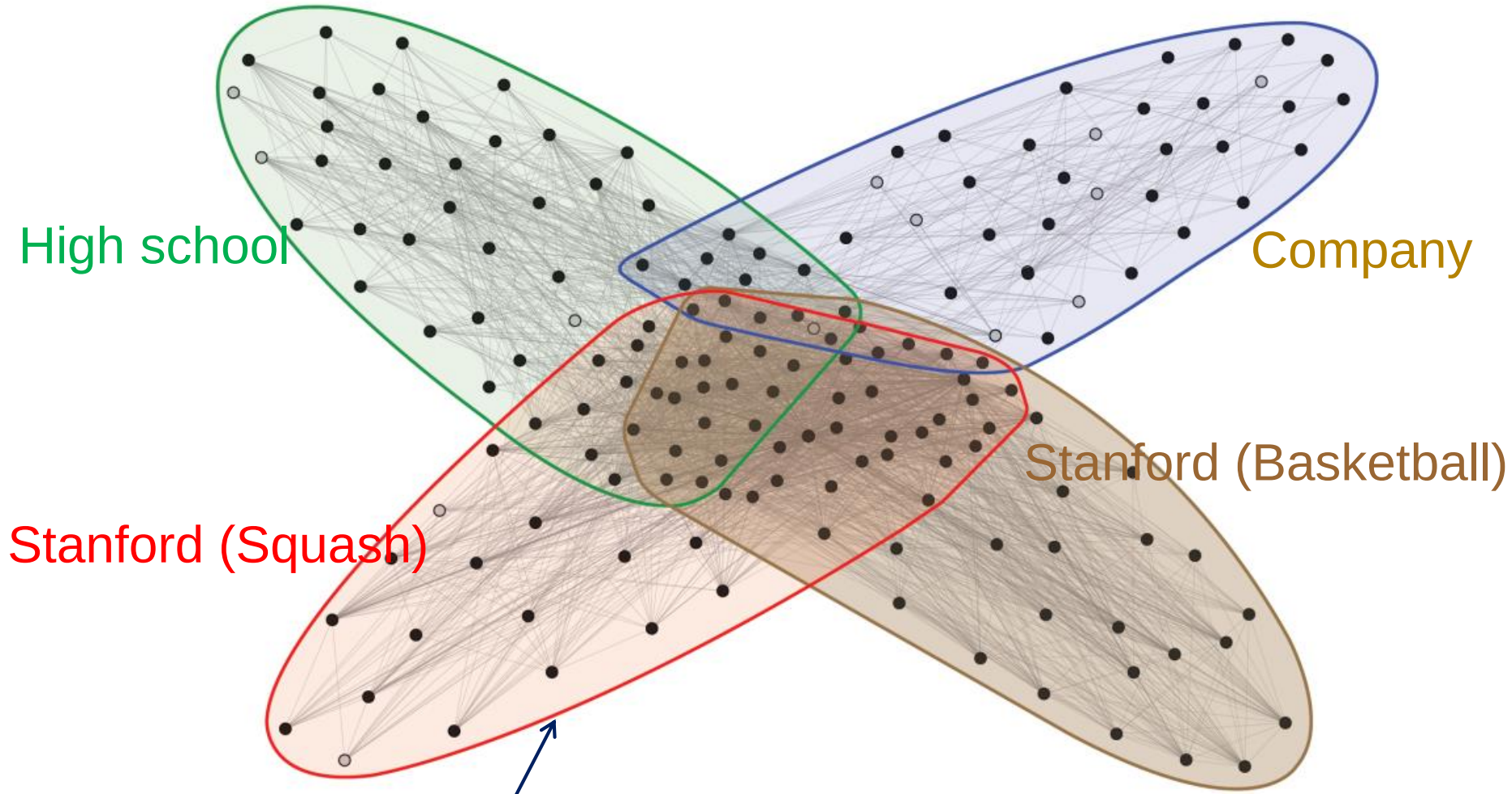
Facebook Ego-network



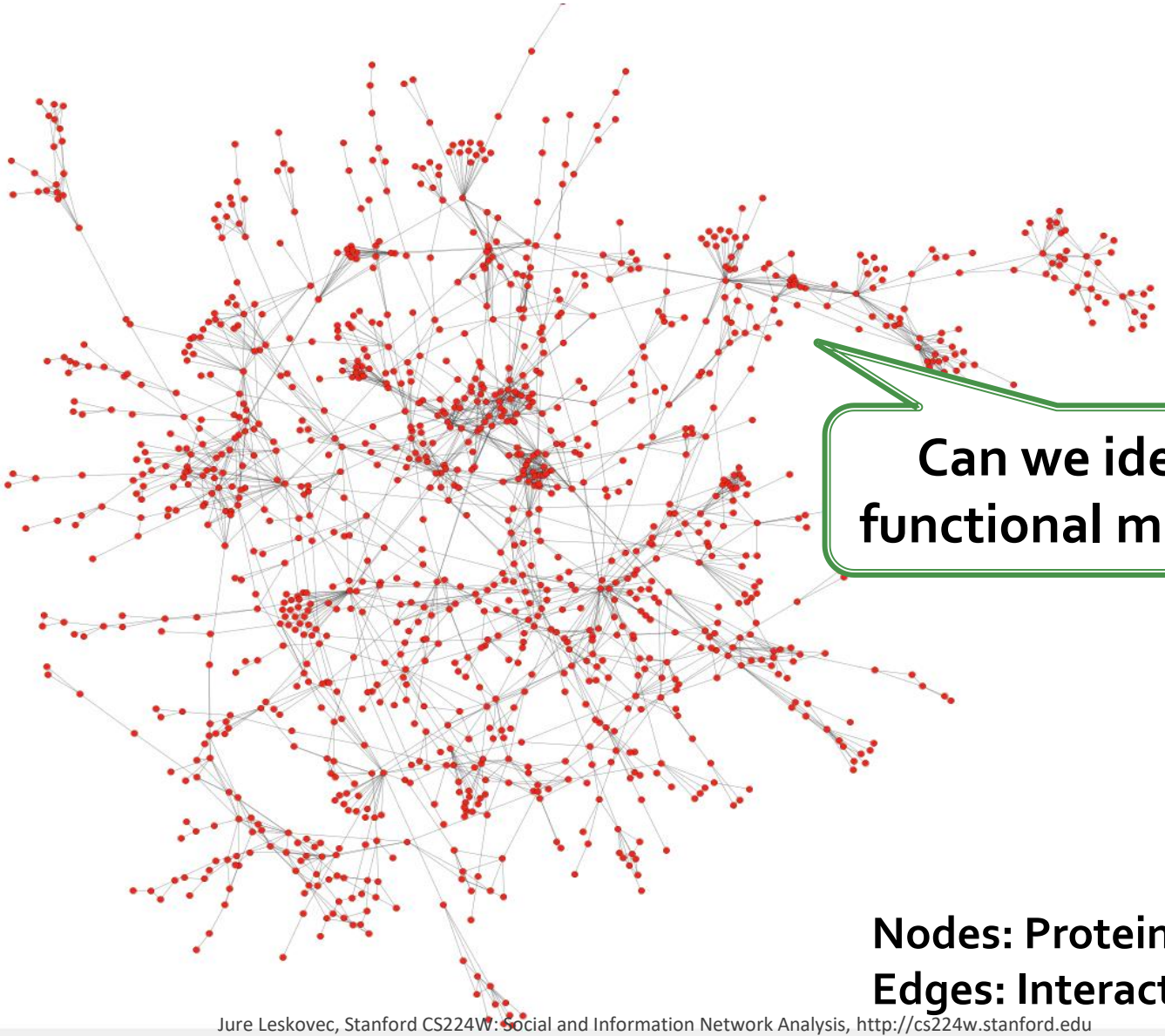
Can we identify social communities?

Nodes: Users
Edges: Friendships

Facebook Ego-network



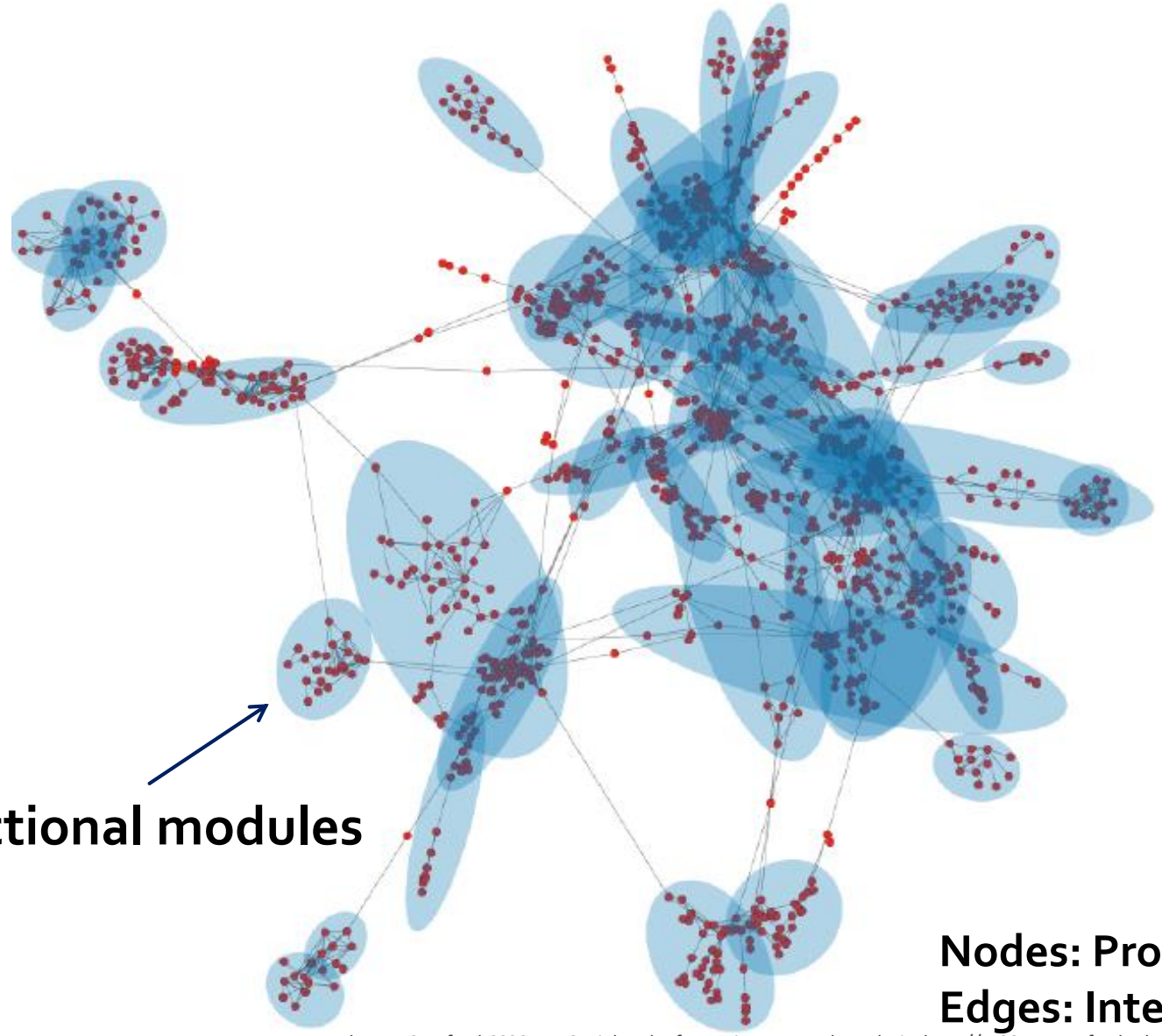
Protein-Protein Interactions



Can we identify functional modules?

Nodes: Proteins
Edges: Interactions

Protein-Protein Interactions

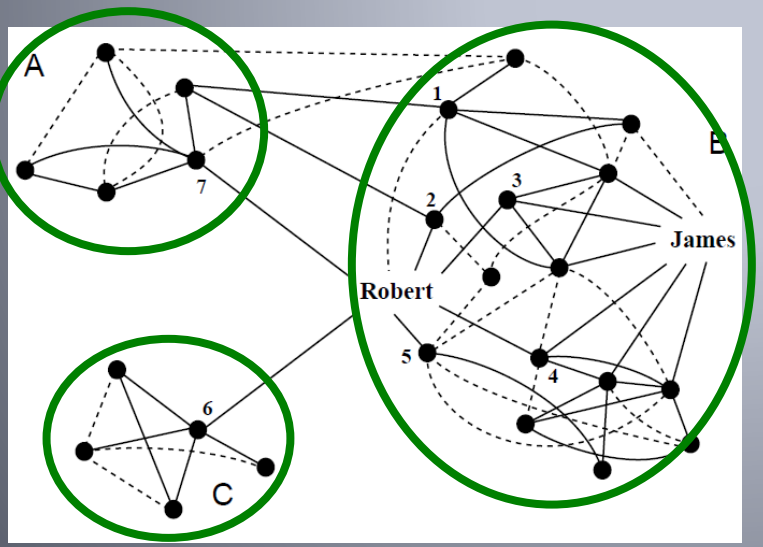


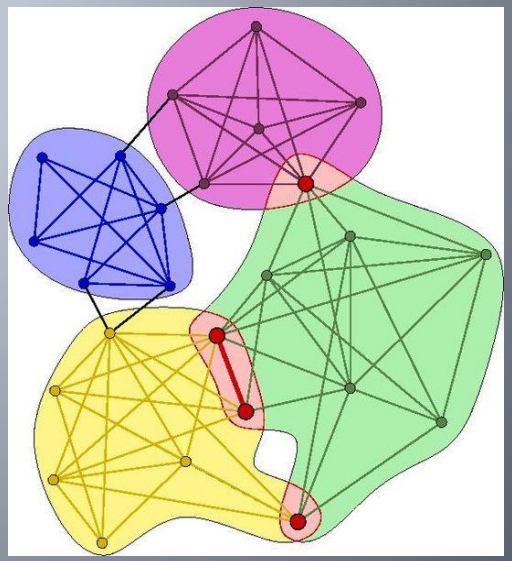
Functional modules

Nodes: Proteins
Edges: Interactions

Community Detection

How to find communities?

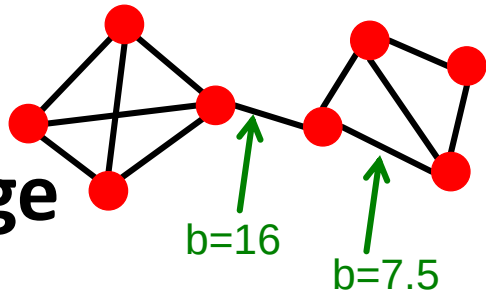


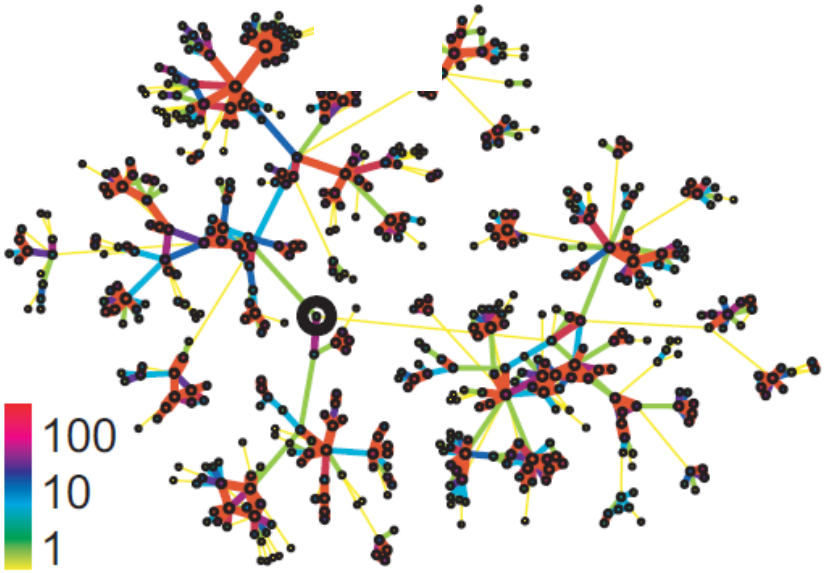


We will work with **undirected** (unweighted) networks

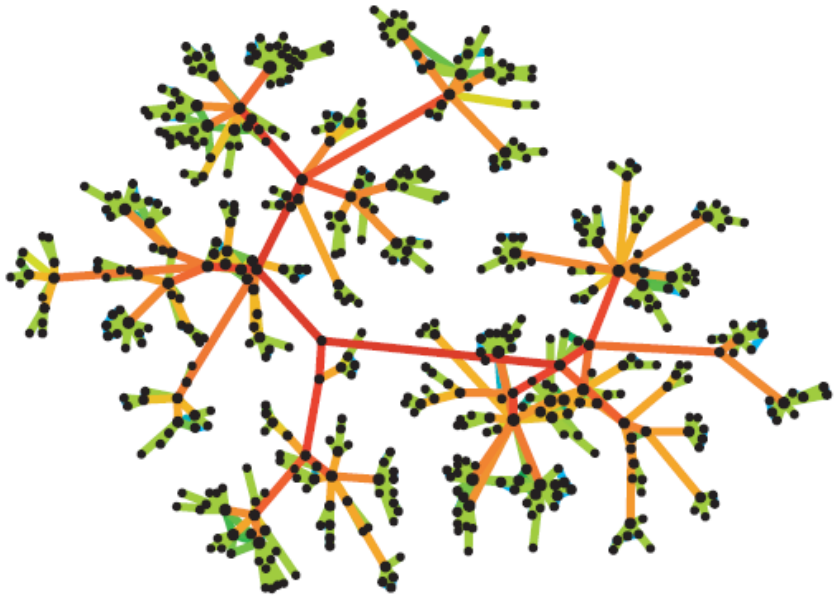
Method 1: Strength of Weak Ties

- **Edge betweenness:** Number of shortest paths passing over the edge
- **Intuition:**





Edge strengths (call volume) in a real network



Edge betweenness in a real network

[Girvan-Newman '02]

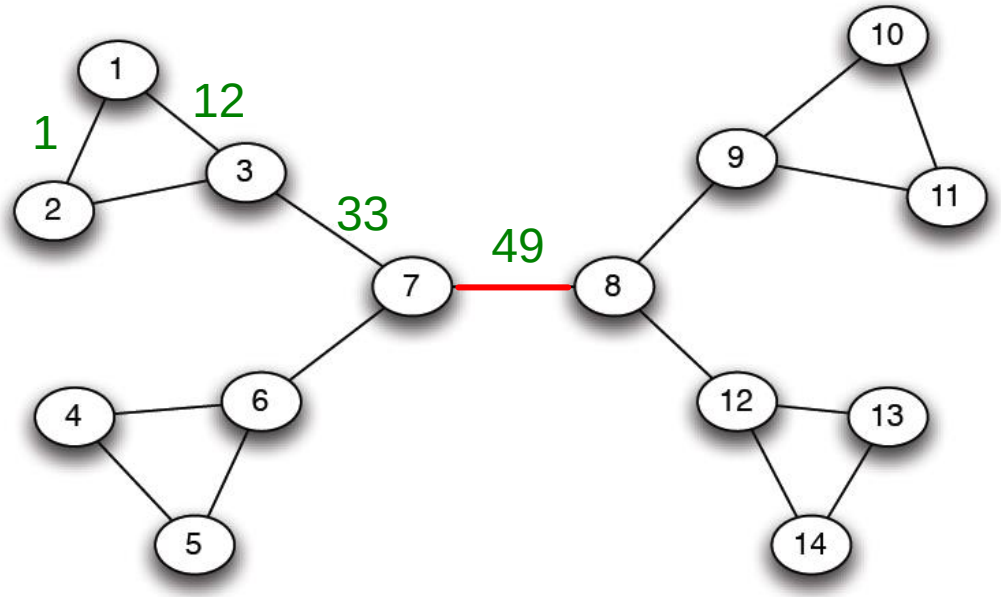
Method 1: Girvan-Newman

- Divisive hierarchical clustering based on the notion of edge **betweenness**:

  Number of shortest paths passing through the edge

- **Girvan-Newman Algorithm:**
  - **Undirected unweighted networks**
  - **Repeat until no edges are left:**
    - Calculate betweenness of edges
    - Remove edges with highest betweenness
  - Connected components are communities
  - Gives a hierarchical decomposition of the network

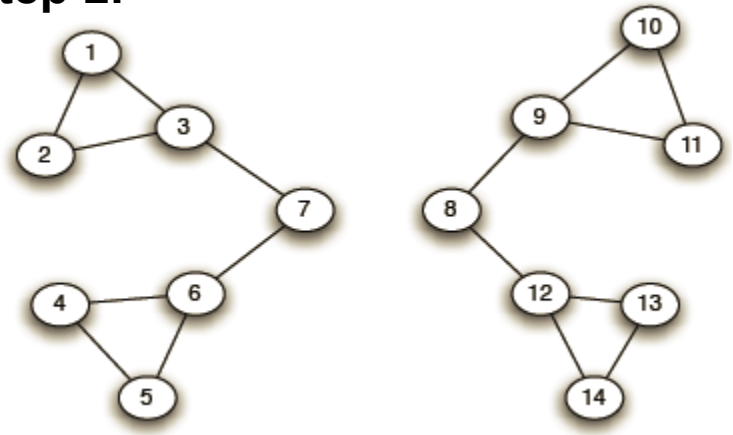
Girvan-Newman: Example



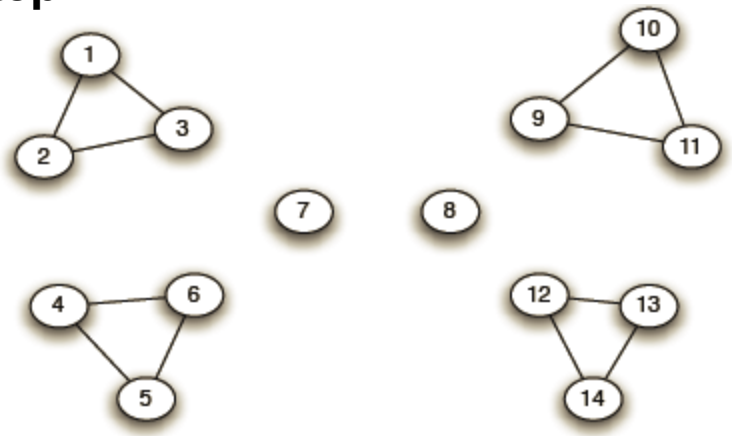
Need to re-compute betweenness at every step

Girvan-Newman: Example

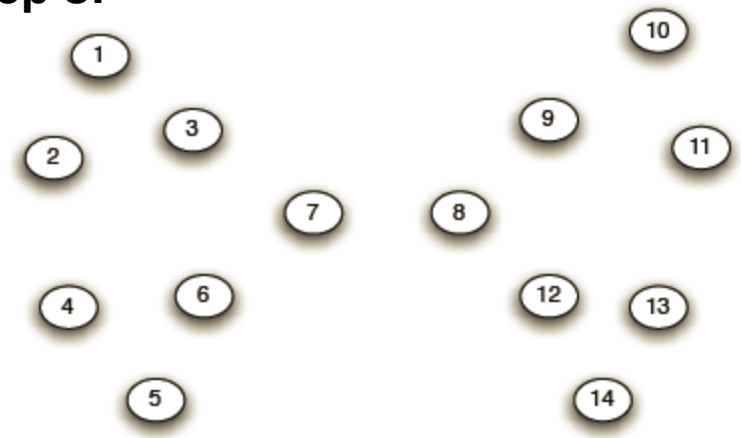
Step 1:



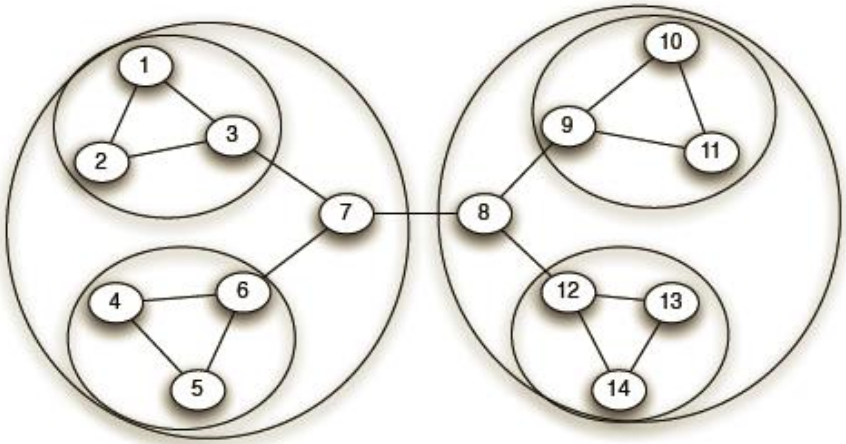
Step 2:



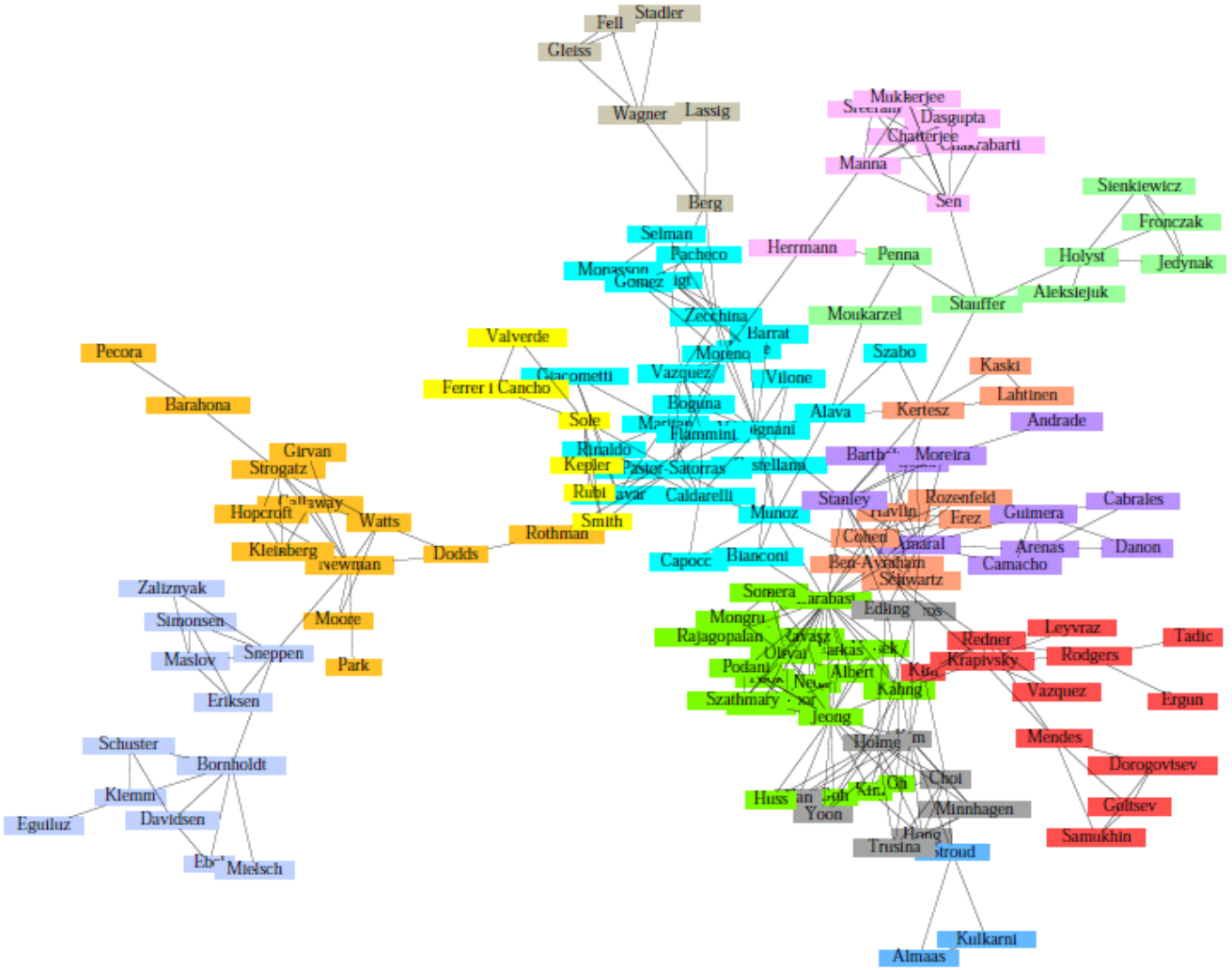
Step 3:



Hierarchical network decomposition:



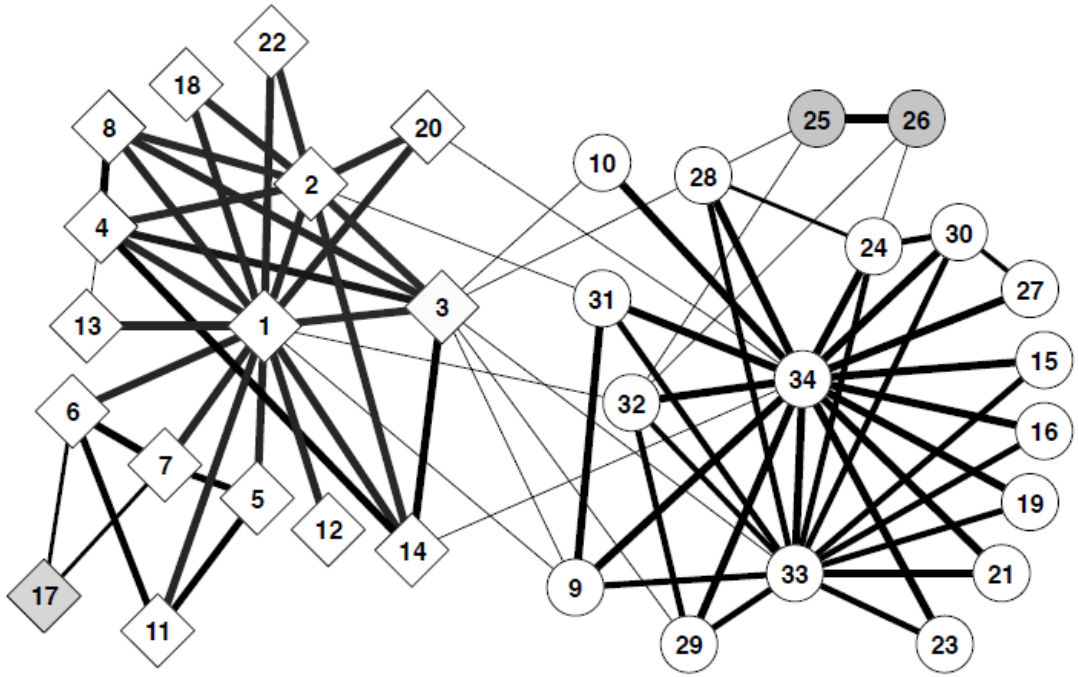
Girvan-Newman: Results

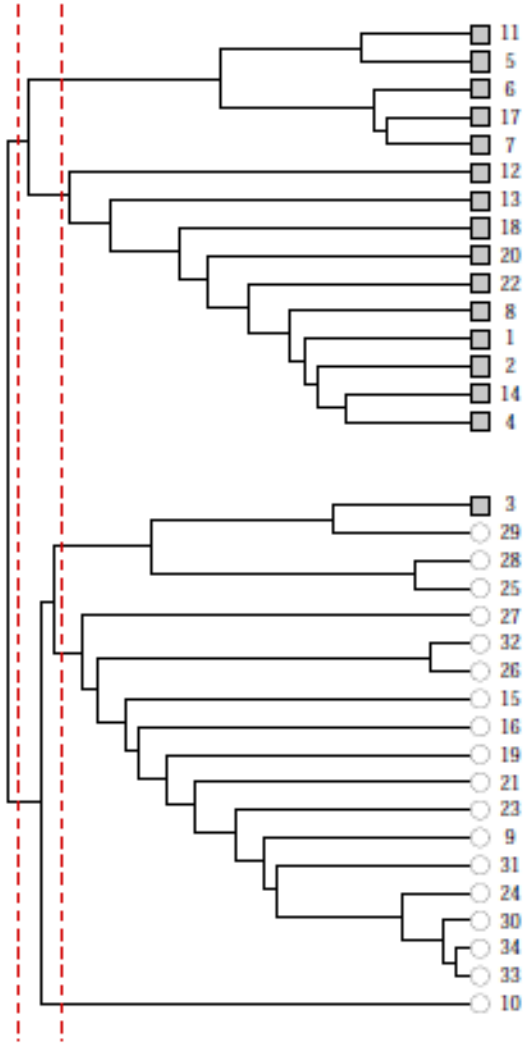


Communities in physics collaborations

Girvan-Newman: Results

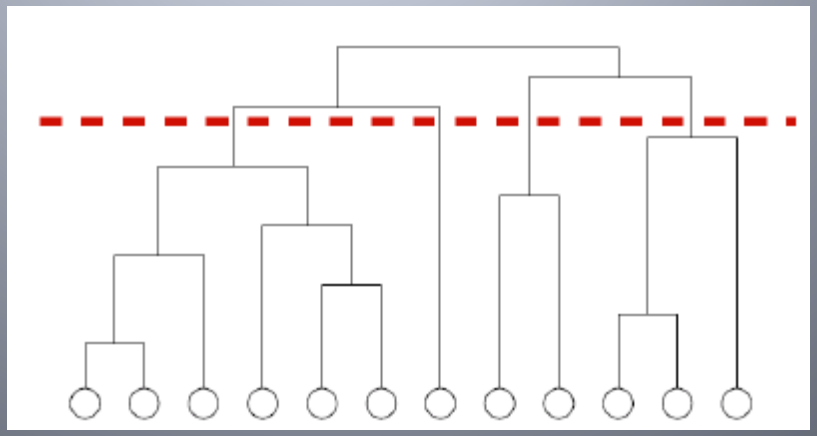
- **Zachary's Karate club:**
  Hierarchical decomposition





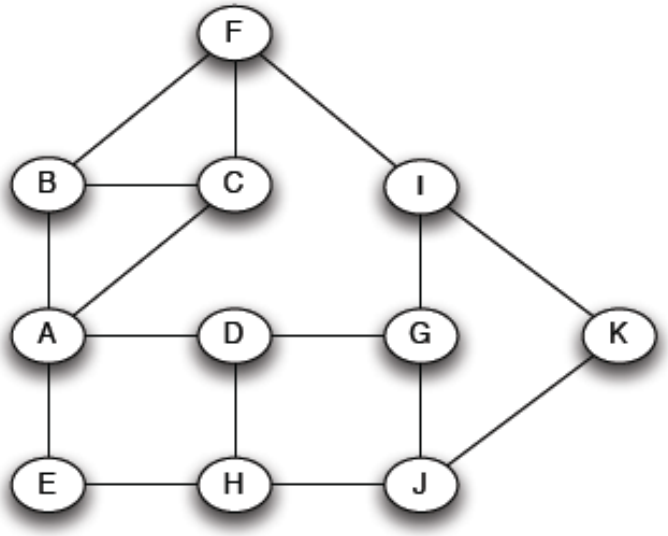
We need to resolve 2 questions

1. How to compute betweenness?
2. How to select the number of clusters?

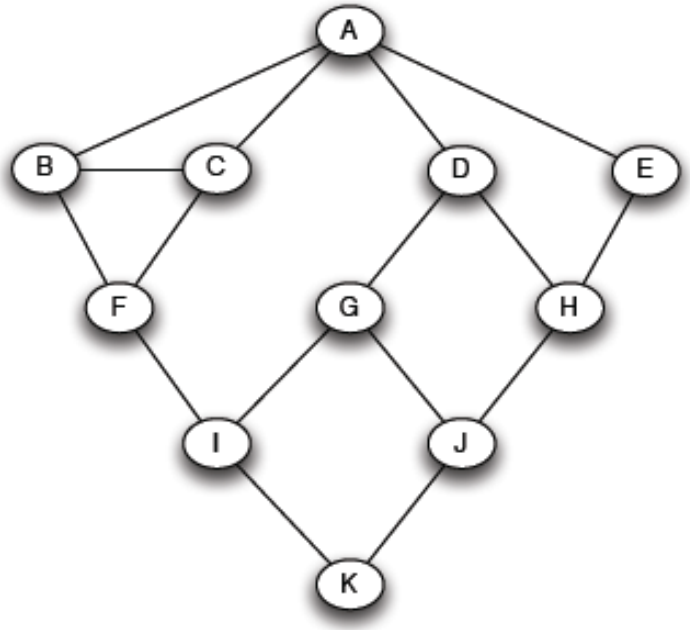


How to Compute Betweenness?

- **Want to compute betweenness of paths starting at node A**

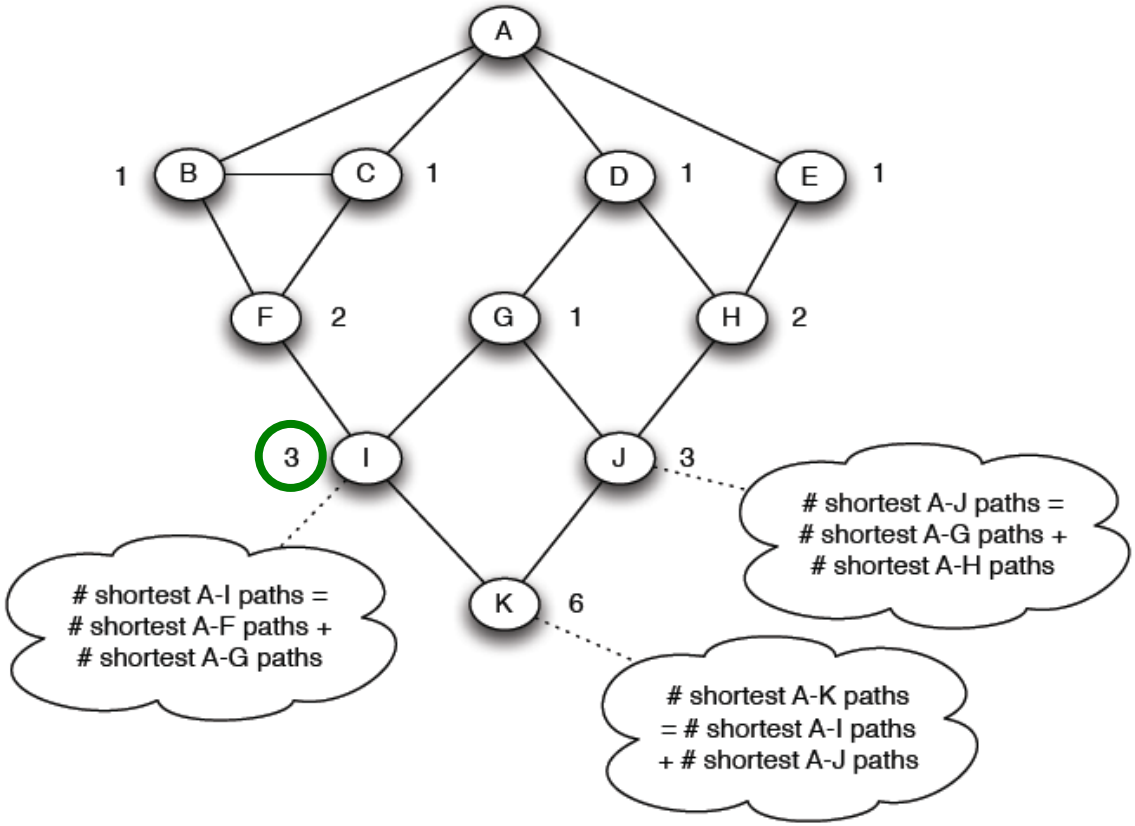


- **Breadth first search starting from A:**



How to Compute Betweenness?

- **Count the number of shortest paths from A to all other nodes of the network:**

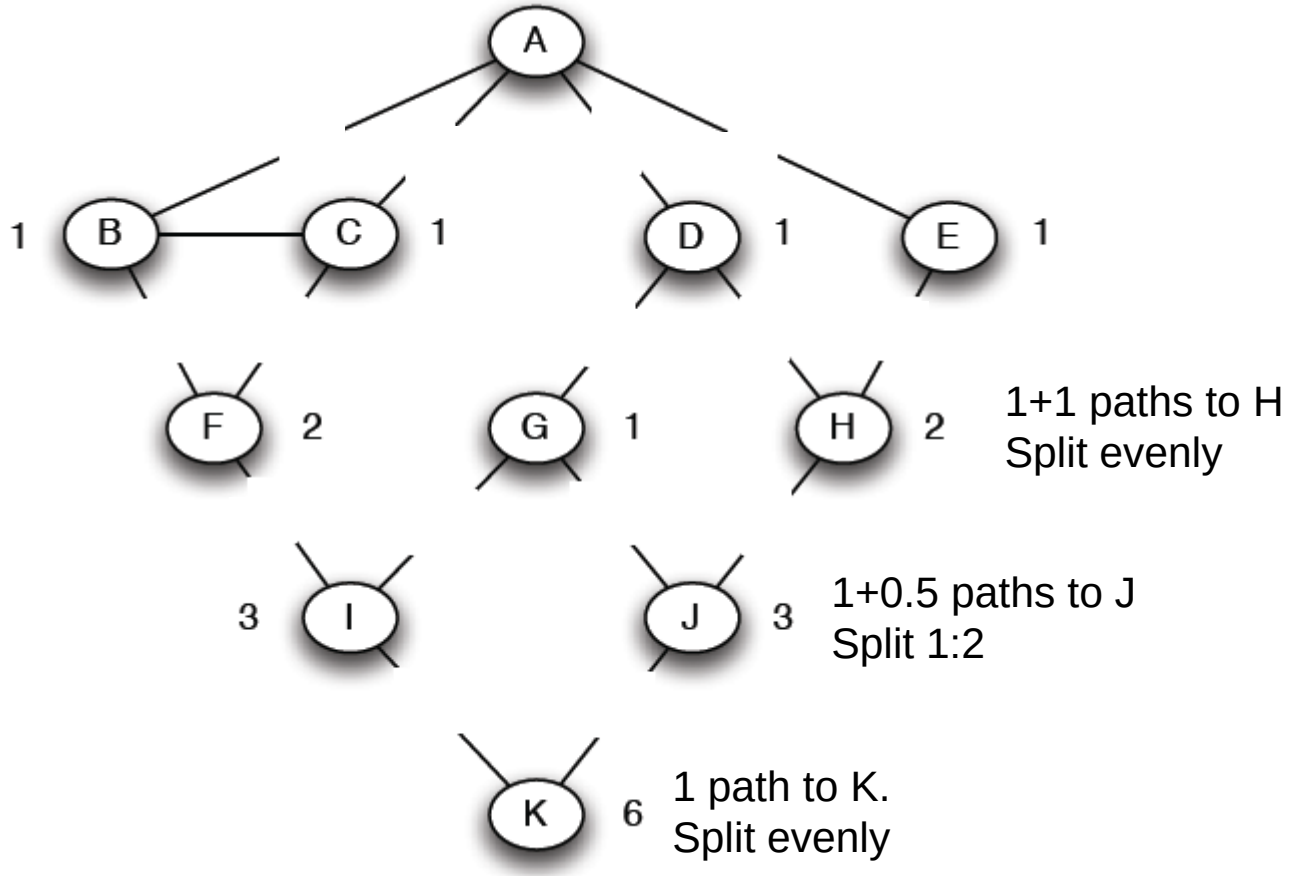


How to Compute Betweenness?

- **Compute betweenness by working up the tree:** If there are multiple paths count them fractionally

The algorithm:

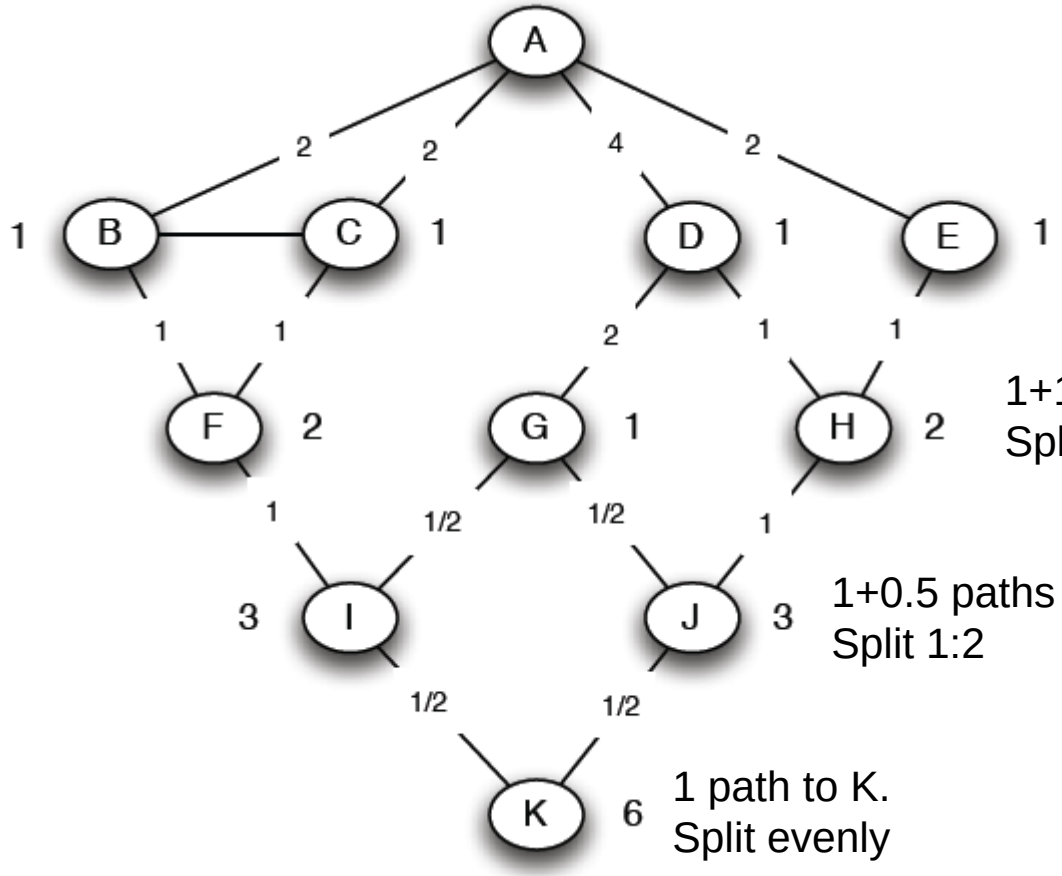
- Add edge **flows**:
  - -- node flow = 1+∑child edges
  - -- split the flow up based on the parent value
- Repeat the BFS procedure for each starting node U



How to Compute Betweenness?

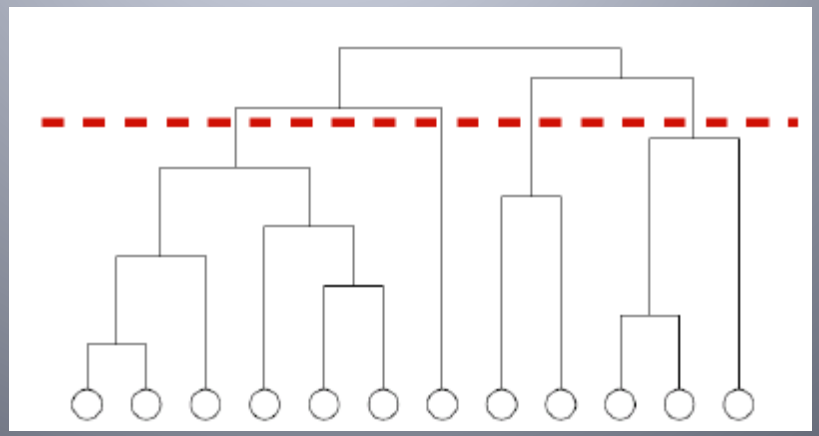
- **Compute betweenness by working up the tree:** If there are multiple paths count them fractionally

The algorithm:
- Add edge **flows**:
  -- node flow = 1+∑child edges
  -- split the flow up based on the parent value
- Repeat the BFS procedure for each starting node U



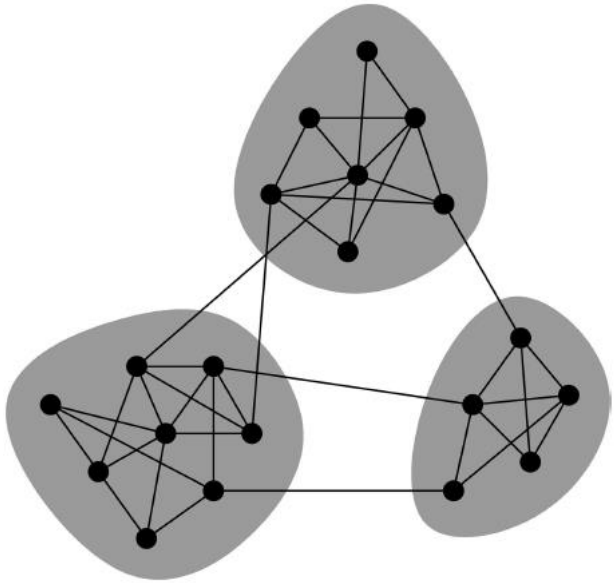
We need to resolve 2 questions

1. How to compute betweenness?
2. How to select the number of clusters?



Network Communities

- **Communities:** sets of **tightly connected nodes**
- Define: **Modularity Q**
  - A measure of how well a network is partitioned into communities
  - Given a partitioning of the network into groups $s \in S$:

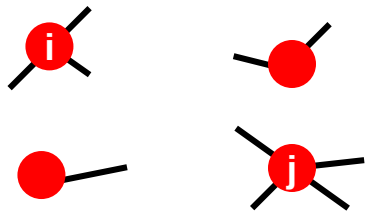


$$Q \propto \sum_{s \in S} [\text{(\# edges within group } s) - \text{(expected \# edges within group } s)]$$

Need a null model!

Null Model: Configuration Model

- **Given real G on n nodes and m edges, construct rewired network G'**
  - Same degree distribution but random connections
  - Consider G' as a multigraph
  - **The expected number of edges between nodes i and j** of degrees k_i and k_j equals to: $k_i \cdot \frac{k_j}{2m} = \frac{k_i k_j}{2m}$
    - The expected number of edges in (multigraph) **G'**:
      - $= \frac{1}{2}\sum_{i \in N}\sum_{j \in N}\frac{k_i k_j}{2m} = \frac{1}{2}\cdot\frac{1}{2m}\sum_{i \in N} k_i \left(\sum_{j \in N} k_j\right) =$
      - $= \frac{1}{4m} 2m \cdot 2m = m$



Note:
$$\sum_{u \in N} k_u = 2m$$

Modularity

- **Modularity of partitioning S of graph G:**
  - **Q $\propto \sum_{s \in S}$ [(# edges within group s) – (expected # edges within group s)]**
  - $$Q(G,S) = \frac{1}{2m} \sum_{s \in S} \sum_{i \in s} \sum_{j \in s} \left(A_{ij} - \frac{k_i k_j}{2m} \right)$$

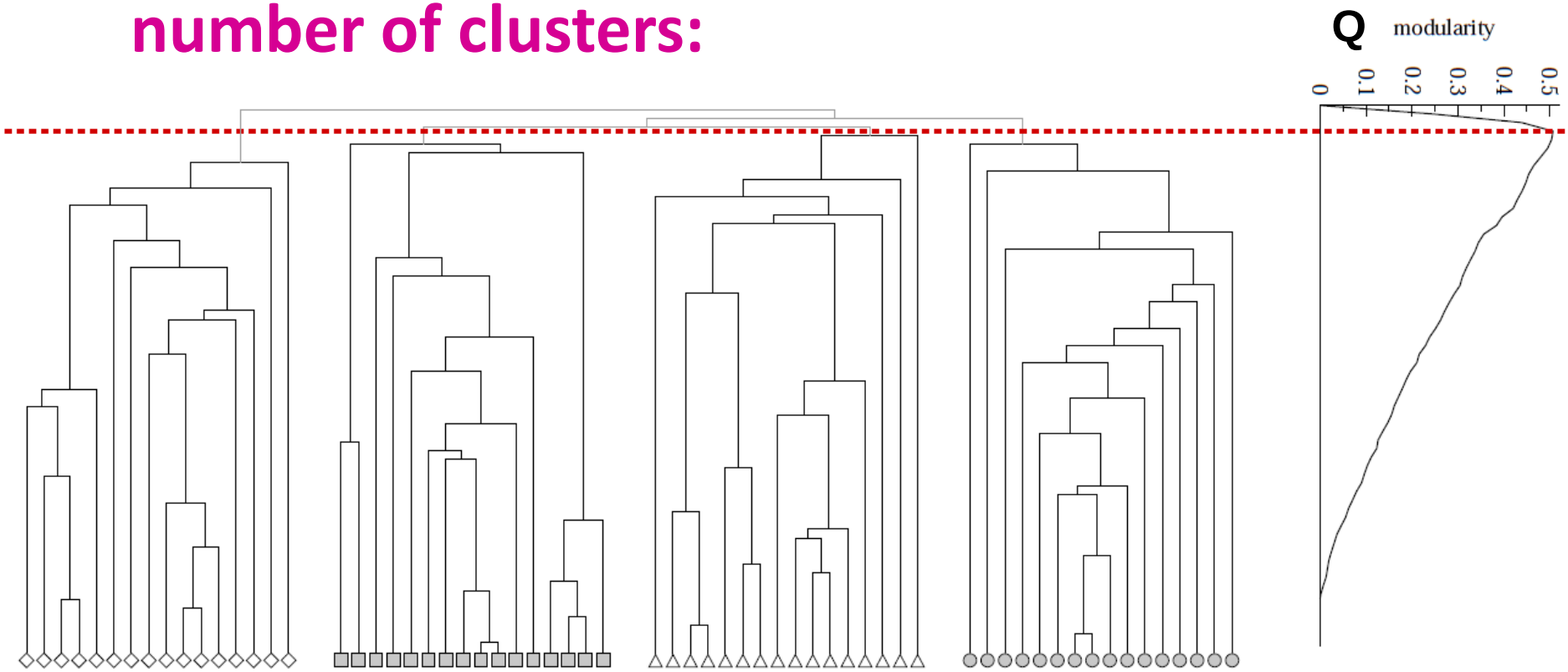
    Normalizing cost.: -1<Q<1

    A_{ij} = 1 if i→j, 0 else

- **Modularity values take range [−1,1]**
  - It is positive if the number of edges within groups exceeds the expected number
  - **0.3-0.7<Q** means significant community structure

Modularity: Number of clusters

- **Modularity is useful for selecting the number of clusters:**



Why not optimize Modularity directly?

Modularity Optimization

Method 2: Modularity Optimization

- **Let's split the graph into 2 communities!**
- **Want to directly optimize modularity!**
  - $\max_S Q(G,S) = \frac{1}{2m}\sum_{s\in S}\sum_{i\in s}\sum_{j\in s}\left(A_{ij} - \frac{k_i k_j}{2m}\right)$
- **Community membership vector s:**
  - s_i **= 1** if node ***i*** is in community **1**
    -1 if node ***i*** is in community **-1**

$\frac{s_i s_j + 1}{2} =$ 1.. if $s_i = s_j$, 0.. else

- $Q(G,s) = \frac{1}{2m}\sum_{i\in N}\sum_{j\in N}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\frac{(s_i s_j + 1)}{2}$

$$= \frac{1}{4m}\sum_{i,j\in N}\left(A_{ij} - \frac{k_i k_j}{2m}\right)s_i s_j$$

Modularity Matrix

Note: each row/col of **B** sums to **0:** $\sum_j A_{ij} = k_i$, $\sum_j \frac{k_i k_j}{2m} = k_i \sum_j \frac{k_j}{2m} = k_i$

- **Define:**
  - **Modularity matrix:** $B_{ij} = A_{ij} - \frac{k_i k_j}{2m}$
  - **Membership:** $s = \{-1, +1\}$
- **Then:** $Q(G,s) = \frac{1}{4m} \sum_{i \in N} \sum_{j \in N} \left(A_{ij} - \frac{k_i k_j}{2m} \right) s_i s_j$

$$= \frac{1}{4m} \sum_{i,j \in N} B_{ij} s_i s_j$$

$$= \frac{1}{4m} \sum_i s_i \underbrace{\sum_j B_{ij} s_j}_{= B_{i\cdot} \cdot s} = \frac{1}{4m} s^T B s$$

- **Task:** Find $s \in \{-1,+1\}^n$ that maximizes **Q(G,s)**

Quick Review of Linear Algebra

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

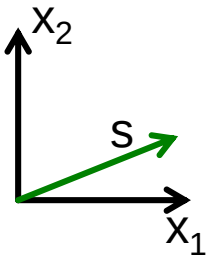
- **Symmetric** matrix **A**
  - That is positive semi-definite: $\boldsymbol{A = U \cdot U^T}$
- Then solutions $\boldsymbol{\lambda, x}$ to equation $\boldsymbol{A \cdot x = \lambda \cdot x}$:
  - **Eigenvectors** $\boldsymbol{x_i}$ ordered by the magnitude of their corresponding **eigenvalues** λ_i ($\lambda_1 \leq \lambda_2 \dots \leq \lambda_n$)
  - $\boldsymbol{x_i}$ are **orthonormal** (orthogonal and unit length)
  - $\boldsymbol{x_i}$ form a coordinate system (basis)
  - If $\boldsymbol{A}$ is positive-semidefinite: $\lambda_i \geq 0$ (and they always exist)
- **Eigen Decomposition theorem:** Can rewrite matrix $\boldsymbol{A}$ in terms of its eigenvectors and eigenvalues: $\boldsymbol{A = \sum_i x_i \cdot \lambda_i \cdot x_i^T}$

Modularity Optimization

- **Rewrite: $Q(G,s) = \frac{1}{4m} s^{\mathrm{T}} B s$ in terms of its eigenvectors and eigenvalues:**

$$= \mathrm{s}^{\mathrm{T}} \left[\sum_{i=1}^{n} x_i \lambda_i x_i^T \right] s = \sum_{i=1}^{n} s^T x_i \lambda_i x_i^T s = \sum_{i=1}^{n} \left(s^T \mathrm{x}_i \right)^2 \lambda_i$$

- **So, if there would be no other constraints on s then to maximize Q, we make $s = x_n$**
  - **Why?** Because $\lambda_n \geq \lambda_{n-1} \geq \cdots$
    - Remember s has fixed length!
    - Assigns all weight in the sum to λ_n (largest eigenvalue)
      - All other $s^T x_i$ terms are **zero** because of orthonormality



Finding the vector s

- **Let's consider only the first term in the summation (because λ_n is the largest):**

$$\max_s Q(G,s) = \sum_{i=1}^{n} \left(s^T x_i\right)^2 \lambda_i \approx \left(s^T x_n\right)^2 \lambda_n$$

- **Let's maximize:** $\sum_{j=1}^{n} s_j \cdot x_{n,j}$ where $s_j \in \{-1,+1\}$
- To do this, we set:
  - $s_j = \begin{cases} +1 & if\ x_{n,j} \geq 0 \ \text{(j−th coordinate of } x_n \geq 0) \\ -1 & if\ x_{n,j} < 0 \ \text{(j−th coordinate of } x_n < 0) \end{cases}$
- **Continue the bisection hierarchically**

Summary: Modularity Optimization

- **Fast Modularity Optimization Algorithm:**
  - Find leading eigenvector x_n of modularity matrix **B**
  - Divide the nodes by the signs of the elements of x_n
  - Repeat hierarchically until:
    - If a proposed split does not cause modularity to increase, declare community indivisible and do not split it
    - If all communities are indivisible, stop
- **How to find x_n? Power method!**
  - Start with random $v^{(0)}$, repeat :
  - When converged ($v^{(t)} \approx v^{(t+1)}$), set $x_n = v^{(t)}$

$$v^{(t+1)} = \frac{Bv^{(t)}}{\left\|Bv^{(t)}\right\|}$$

Summary: Modularity

- **Girvan-Newman:**
  - Based on the "strength of weak ties"
  - Remove edge of highest betweenness
- **Modularity:**
  - Overall quality of the partitioning of a graph
  - Use to determine the number of communities
- **Fast modularity optimization:**
  - Transform the modularity optimization to a eigenvalue problem