大数据计算及应用(五)

Mining Social-Network Graphs

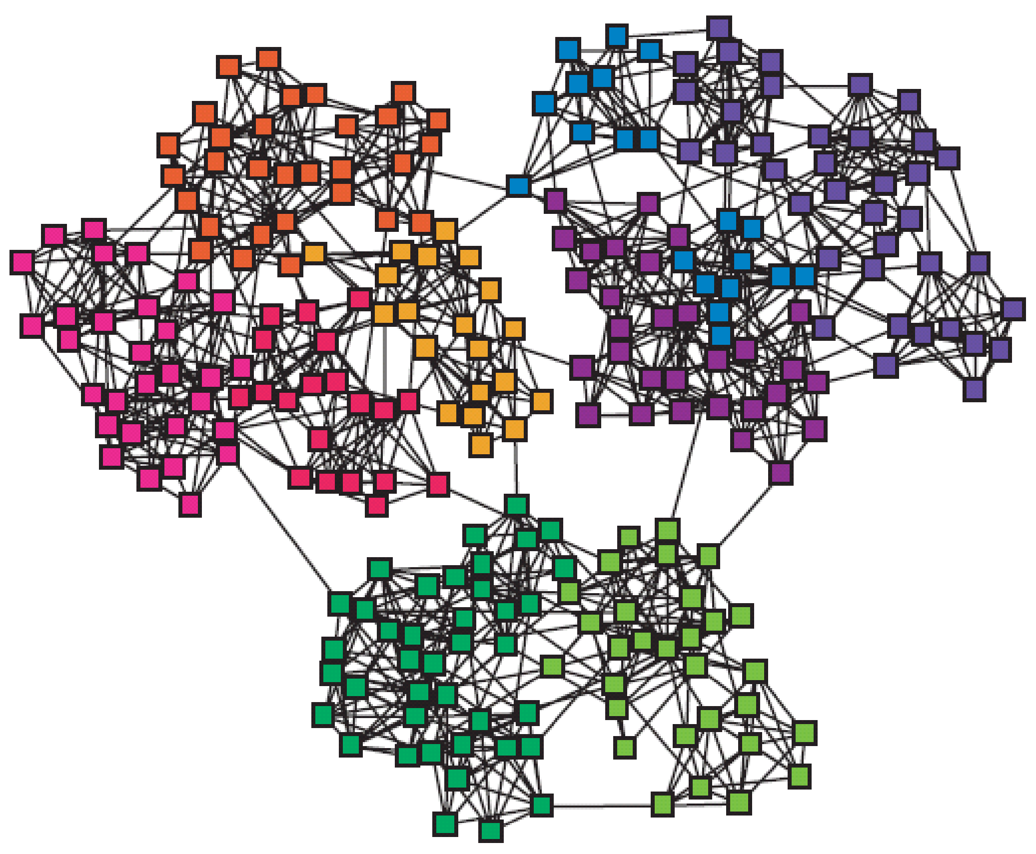
Slides adapted from http://www.mmds.org

Agenda

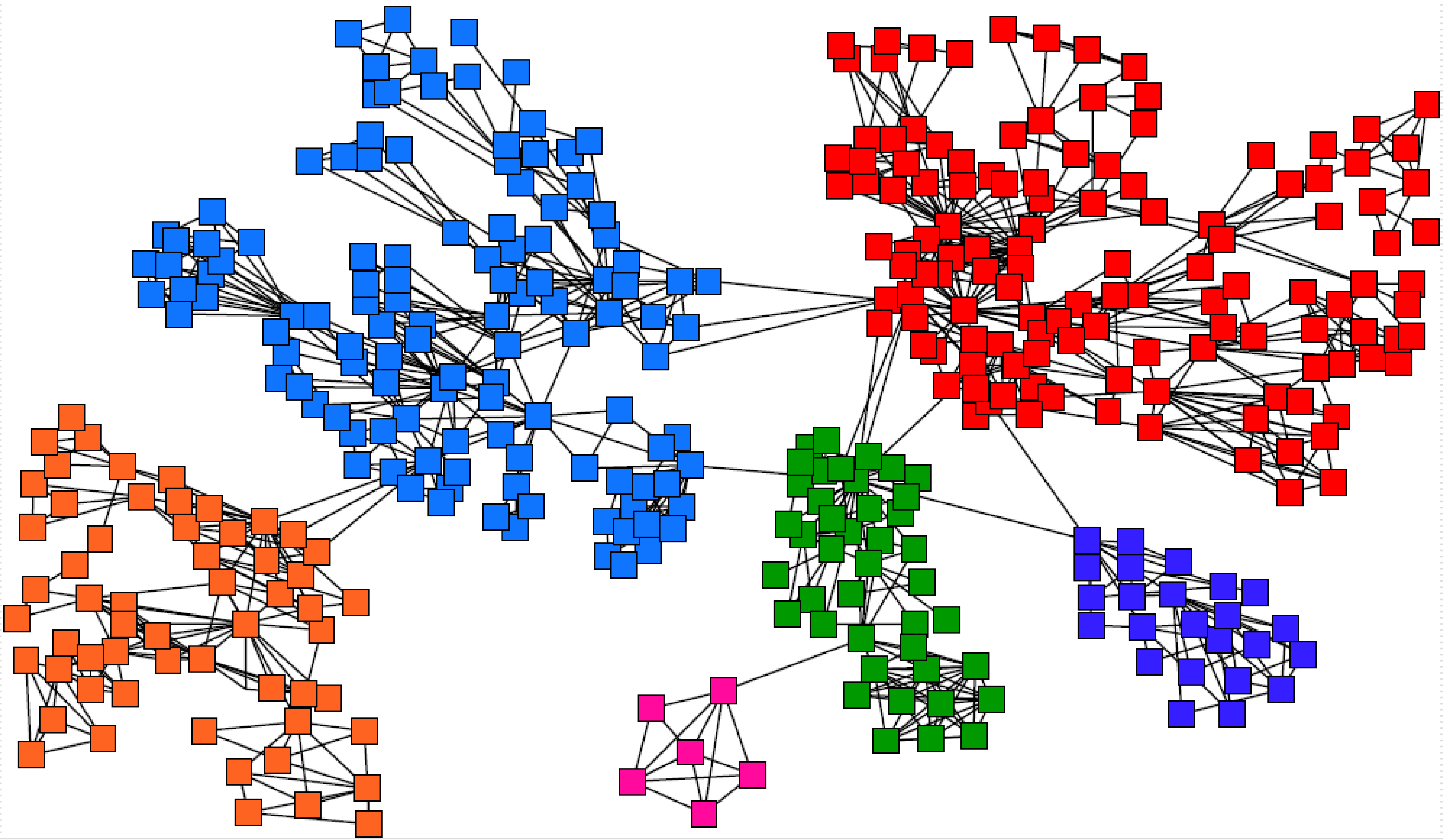
High dim. data	Graph data	Infinite data	Machine learning	Apps
Locality sensitive hashing	**PageRank,** SimRank	Filtering data streams	SVM	Recommender systems
Clustering	Community Detection	Web advertising	Decision Trees	Association Rules
Dimensionality reduction	Spam Detection	Queries on streams	Perceptron, kNN	Duplicate document detection

Networks & Communities

- We often think of networks being organized into modules, cluster, communities:

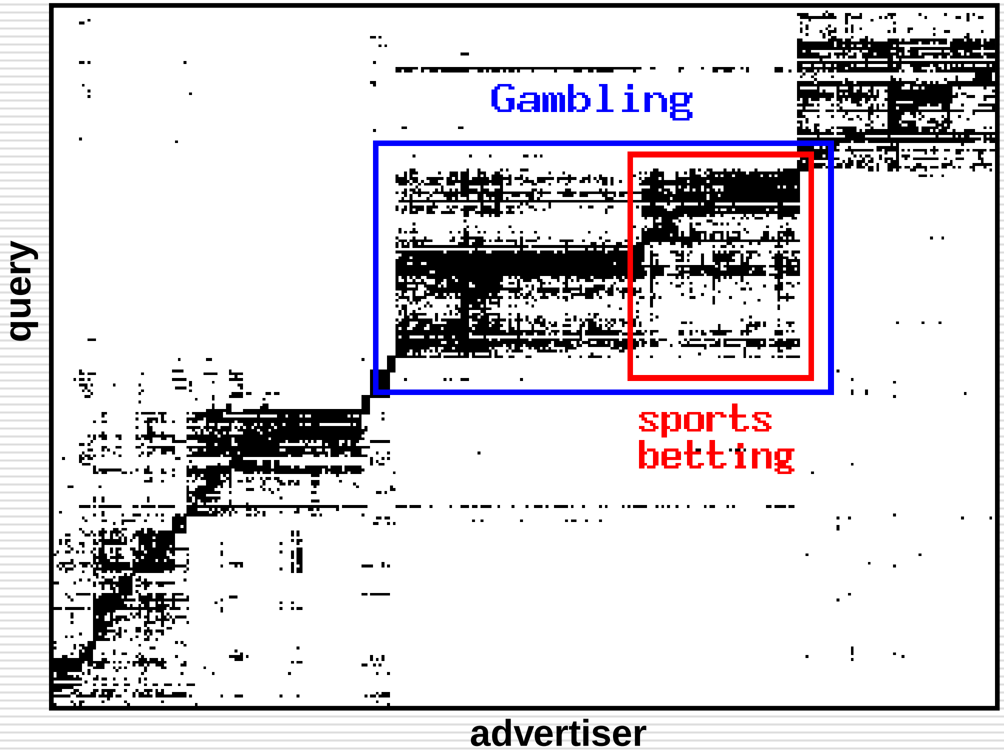


Goal: Find Densely Linked Clusters



Micro-Markets in Sponsored Search

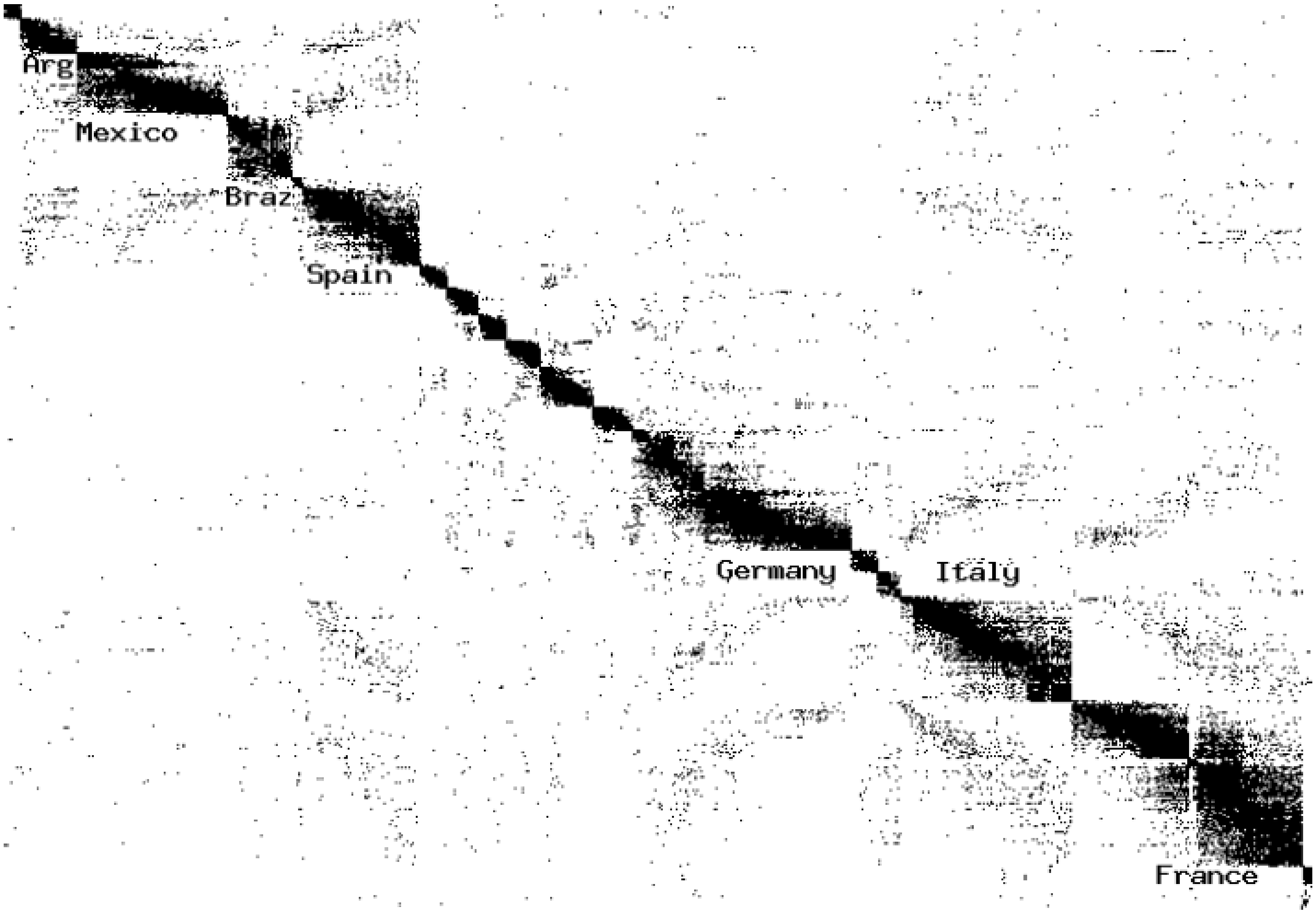
- **Find micro-markets by partitioning the query-to-advertiser graph:**



[Andersen, Lang: Communities from seed sets, 2006]

Movies and Actors

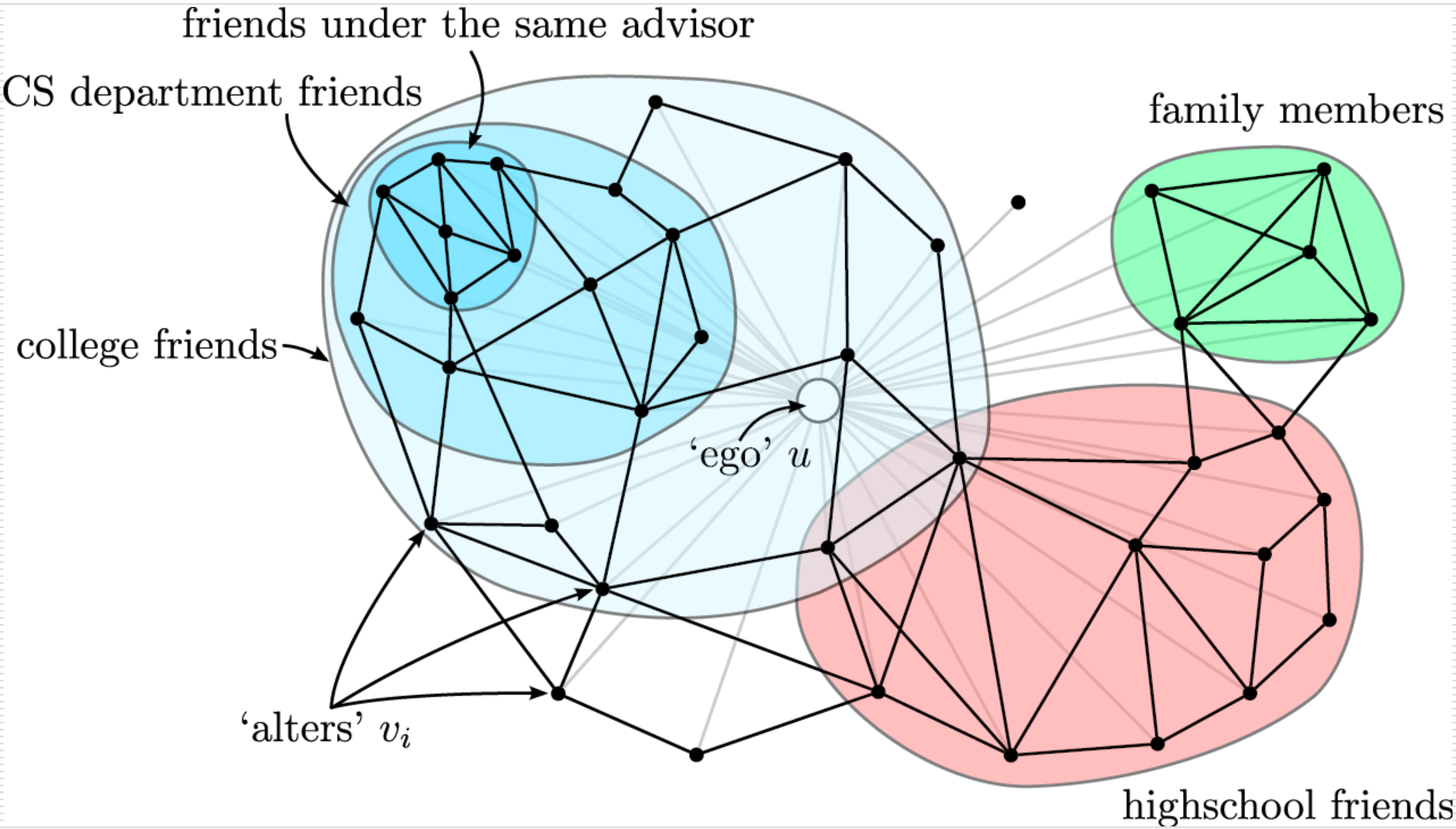
- **Clusters in Movies-to-Actors graph:**



[Andersen, Lang: Communities from seed sets, 2006]

Twitter & Facebook

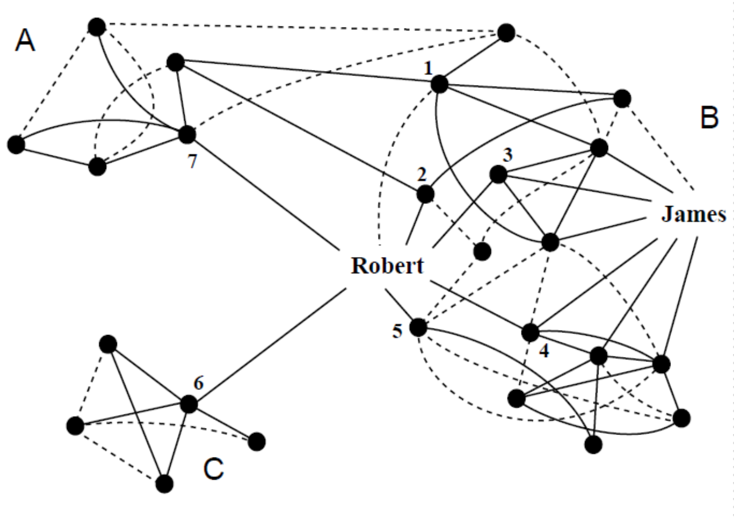
- Discovering social circles, circles of trust:

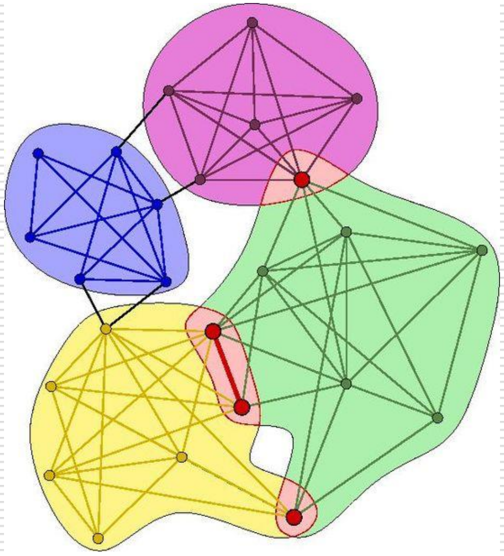


[McAuley, Leskovec: Discovering social circles in ego networks, 2012]

Community Detection

How to find communities?

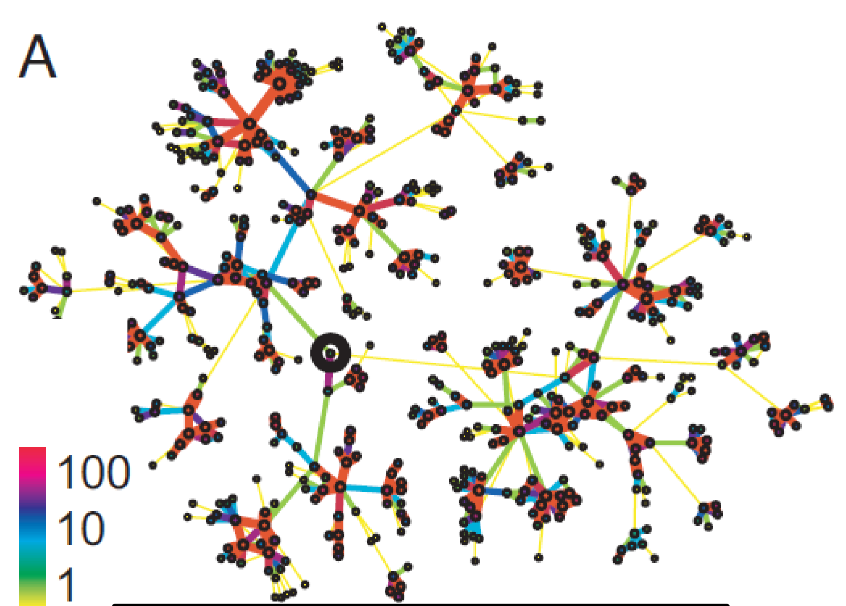




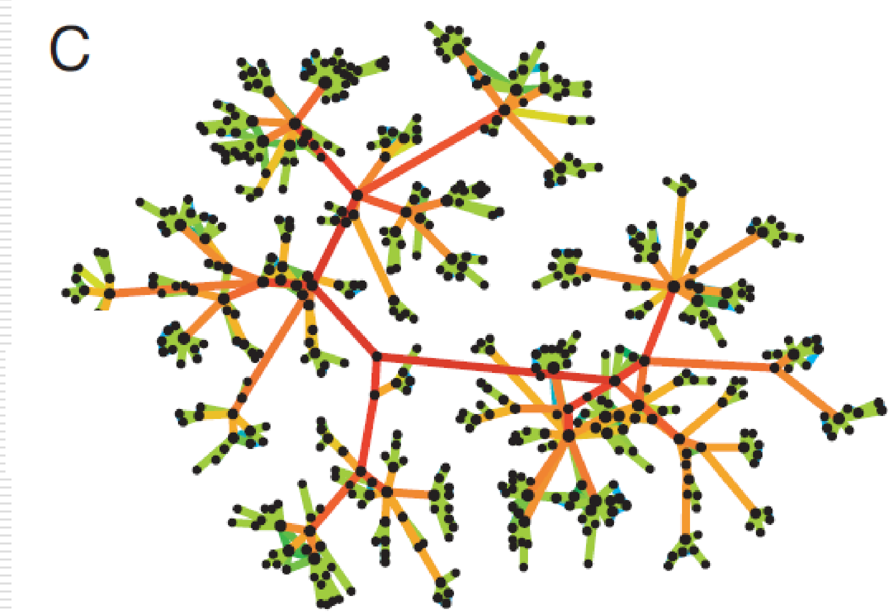
We will work with undirected (unweighted) networks

Method 1: Strength of Weak Ties

Intuition:

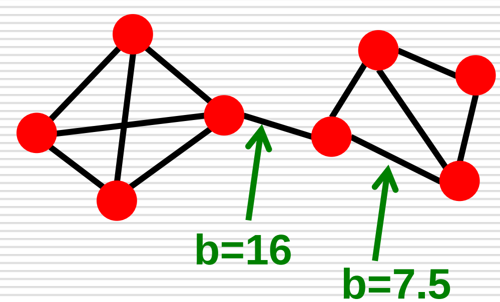


Edge strengths (call volume) in a real network



Edge betweenness in a real network

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

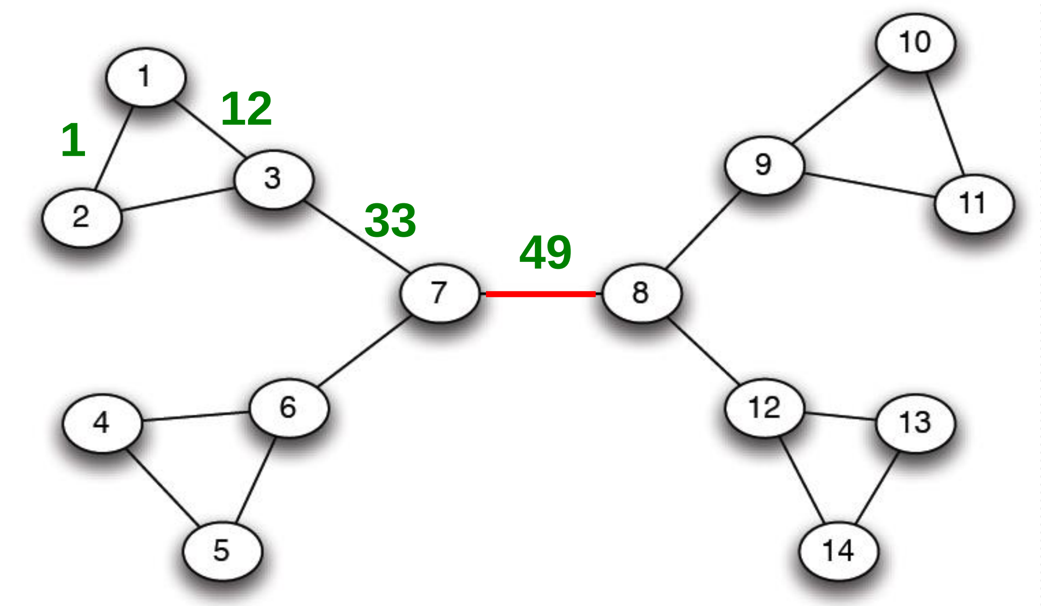


[Girvan-Newman '02]

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

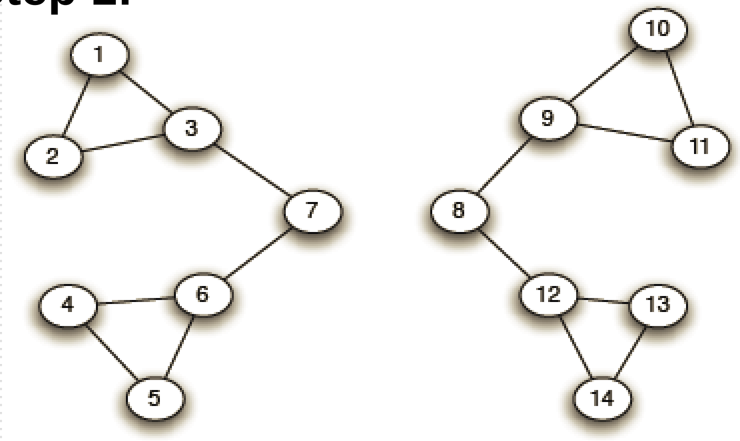
Girvan-Newman: Example



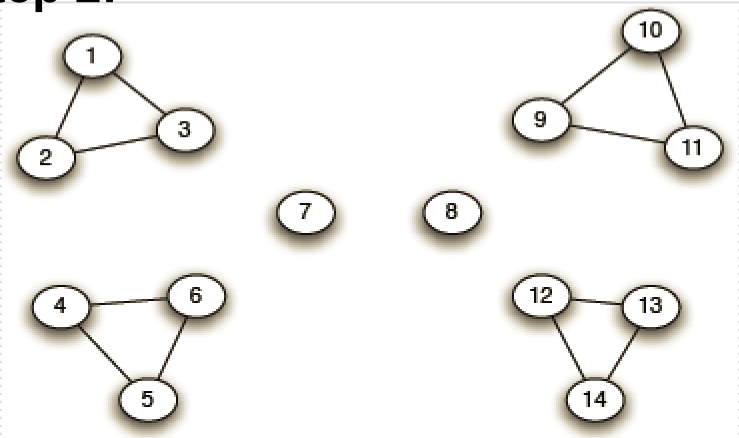
Need to re-compute betweenness at every step

Girvan-Newman: Example

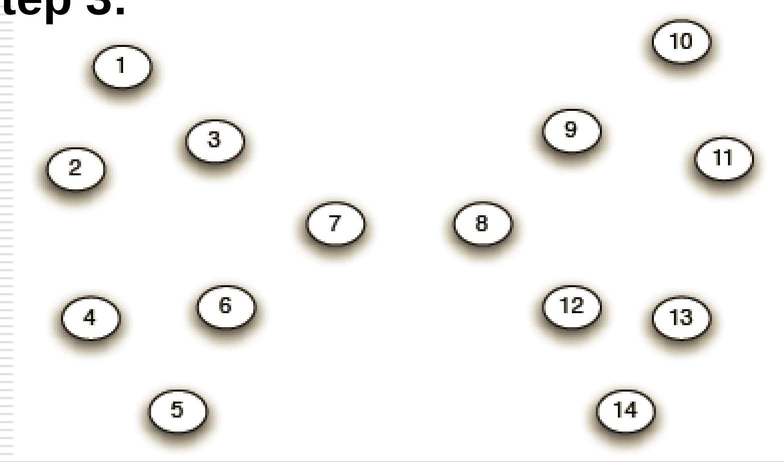
Step 1:



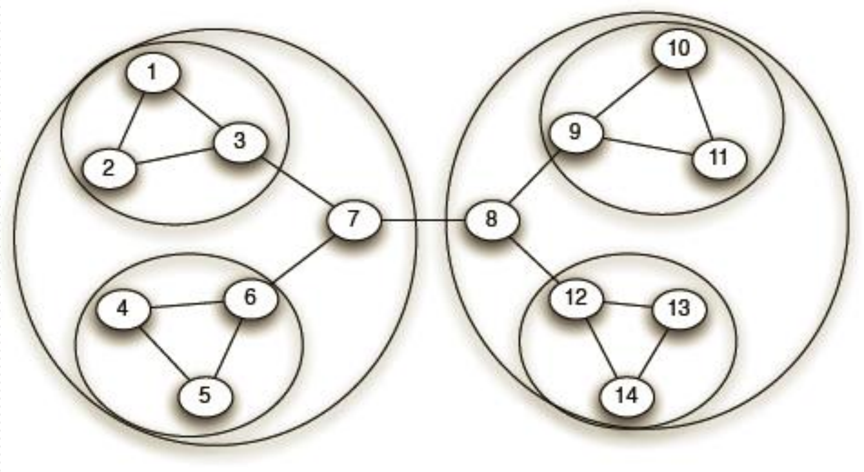
Step 2:



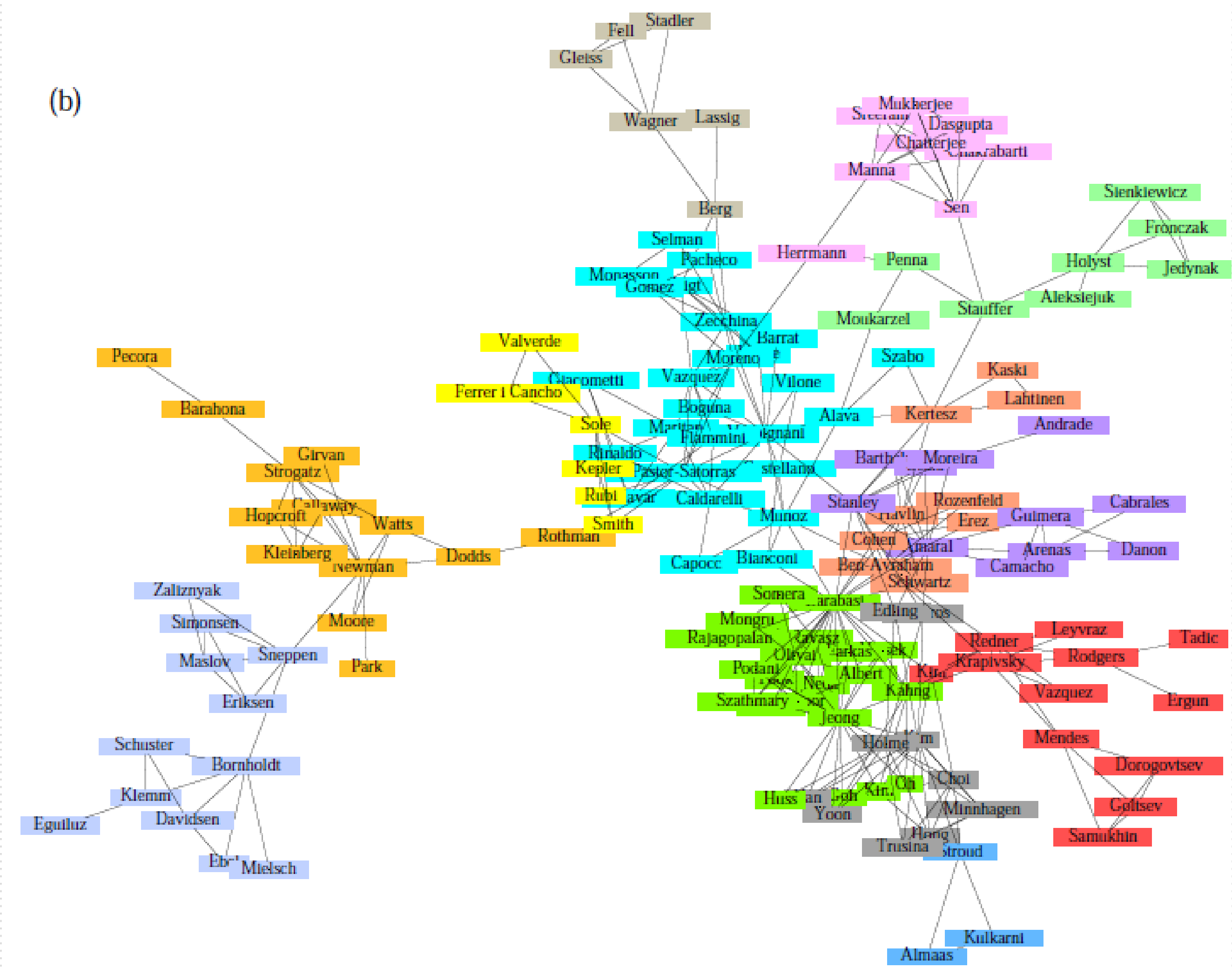
Step 3:



Hierarchical network decomposition:



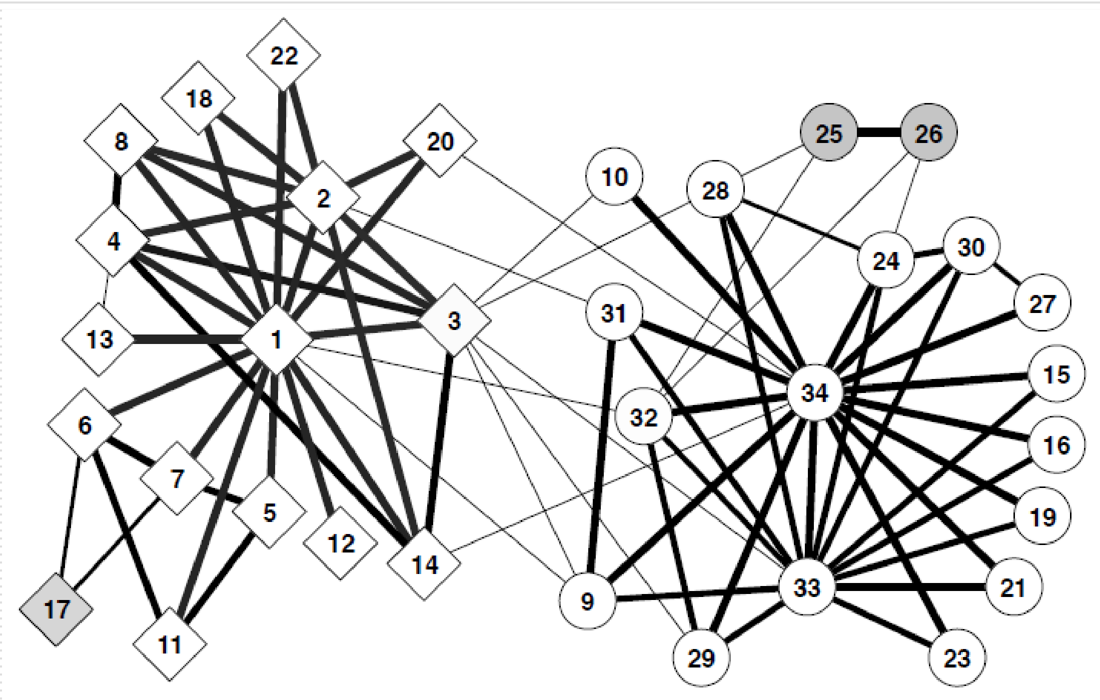
Girvan-Newman: Results

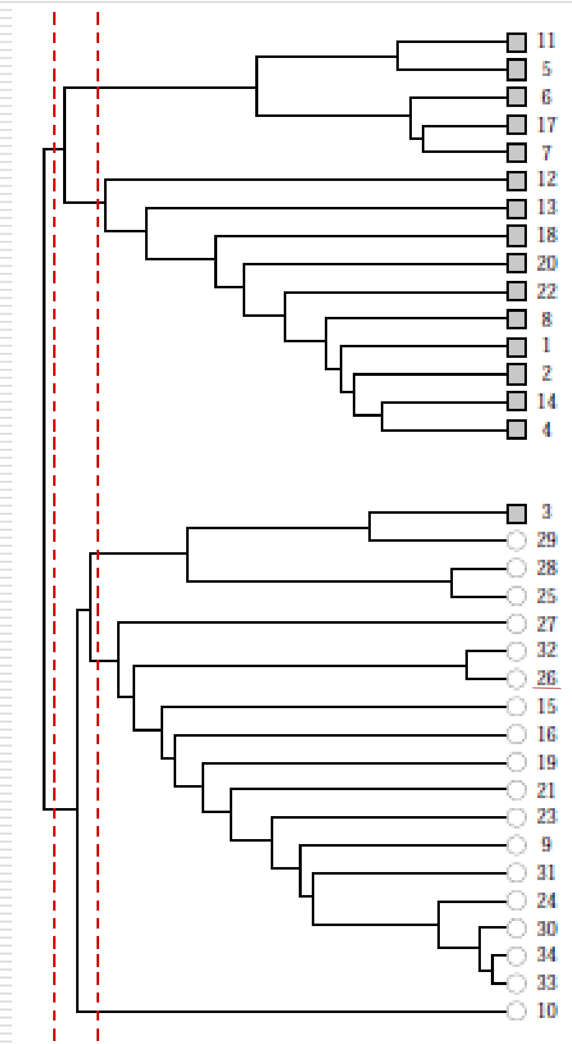


Communities in physics collaborations

Girvan-Newman: Results

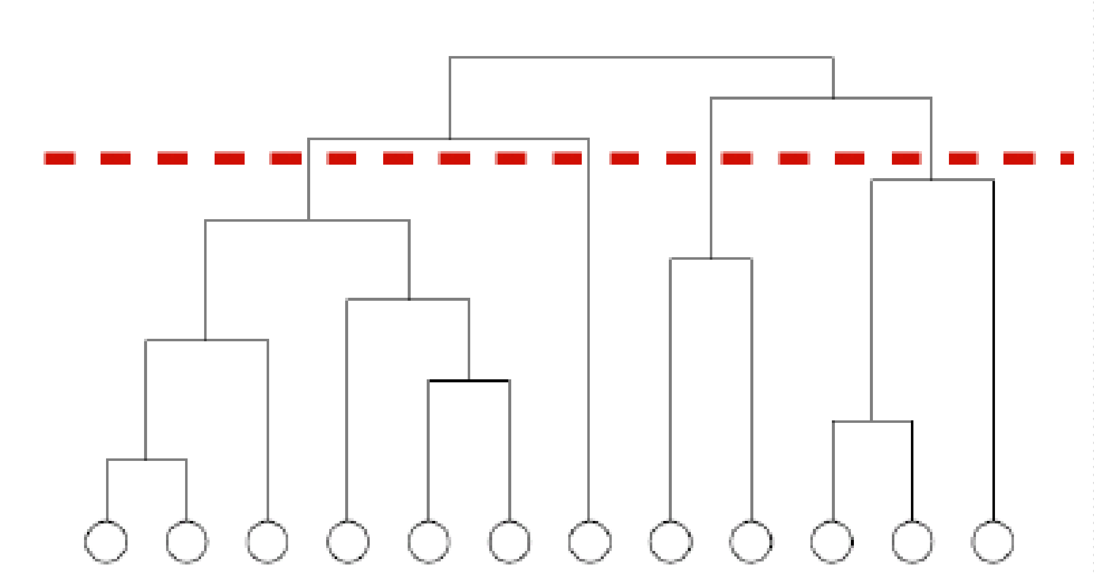
- **Zachary's Karate club:** Hierarchical decomposition





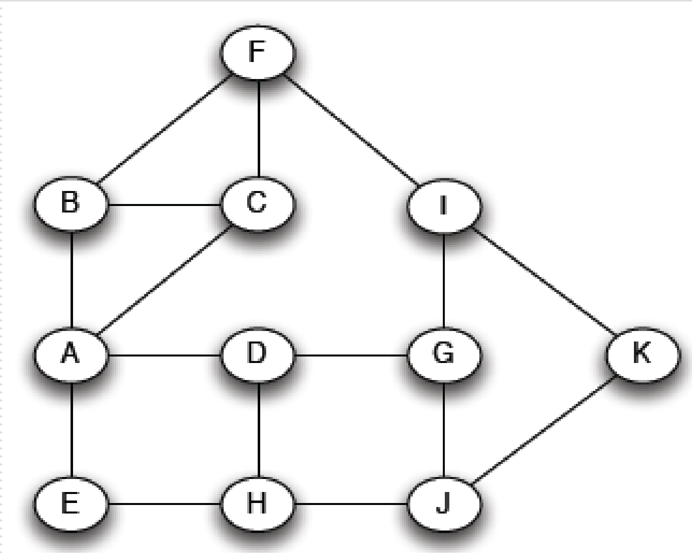
We need to resolve 2 questions

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

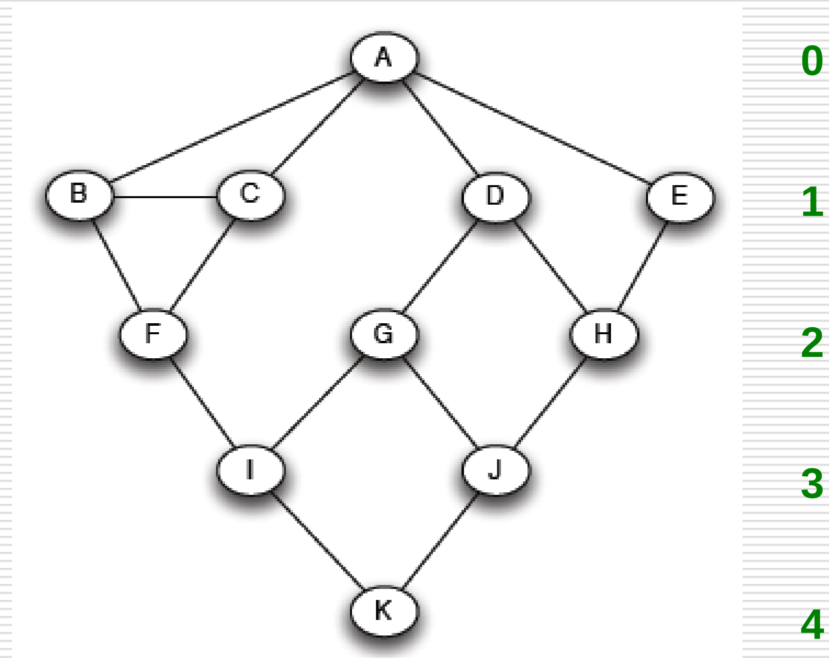


How to Compute Betweenness?

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

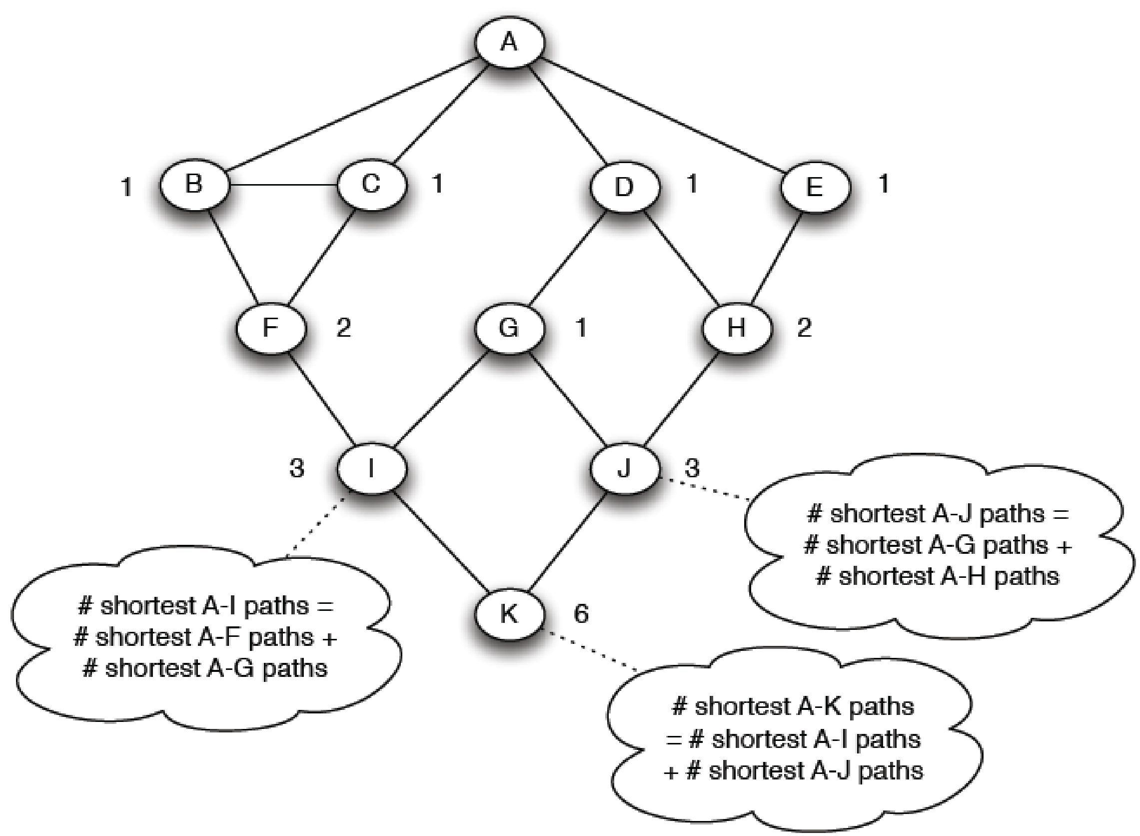


- **Breath first search starting from A:**



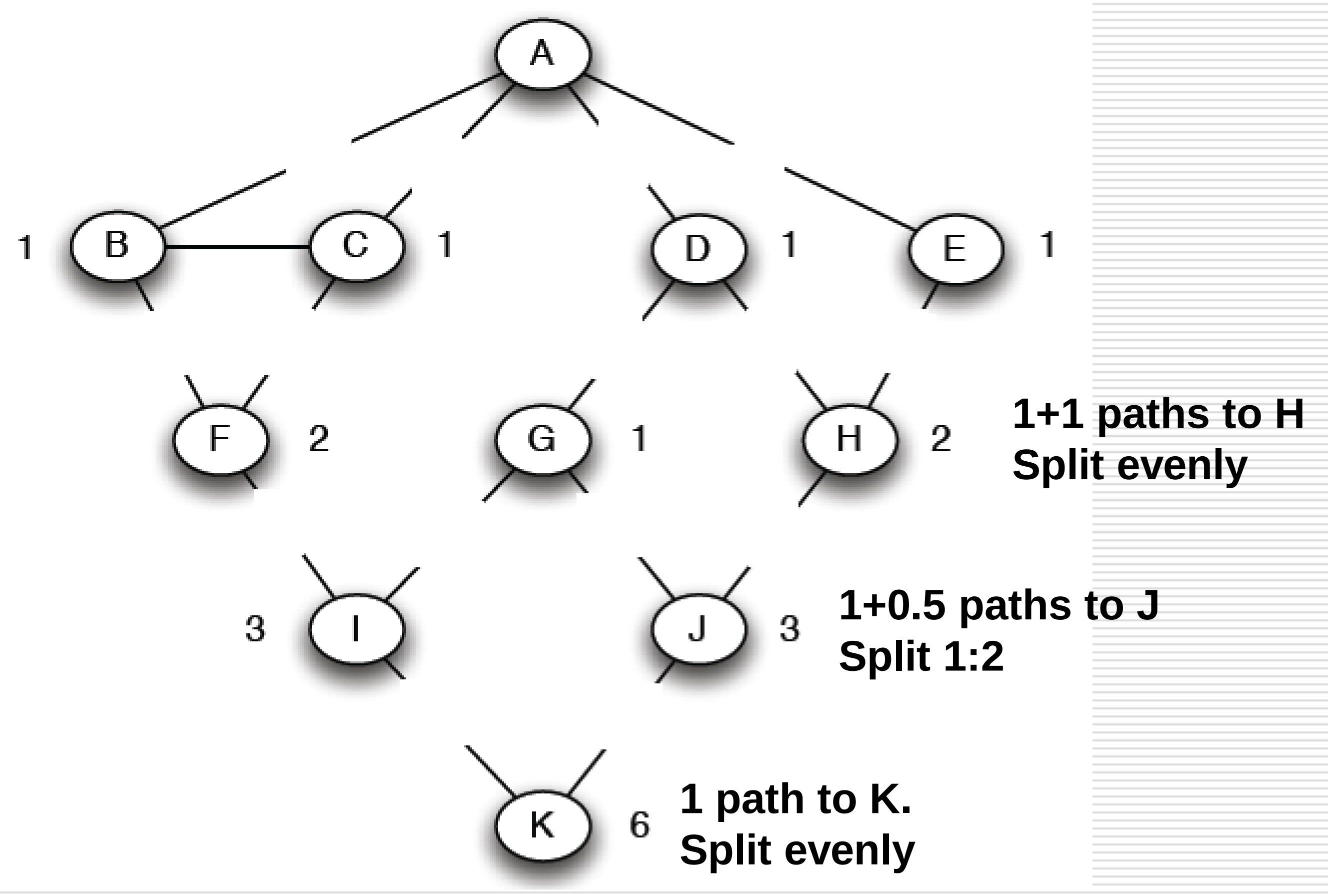
How to Compute Betweenness?

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



How to Compute Betweenness?

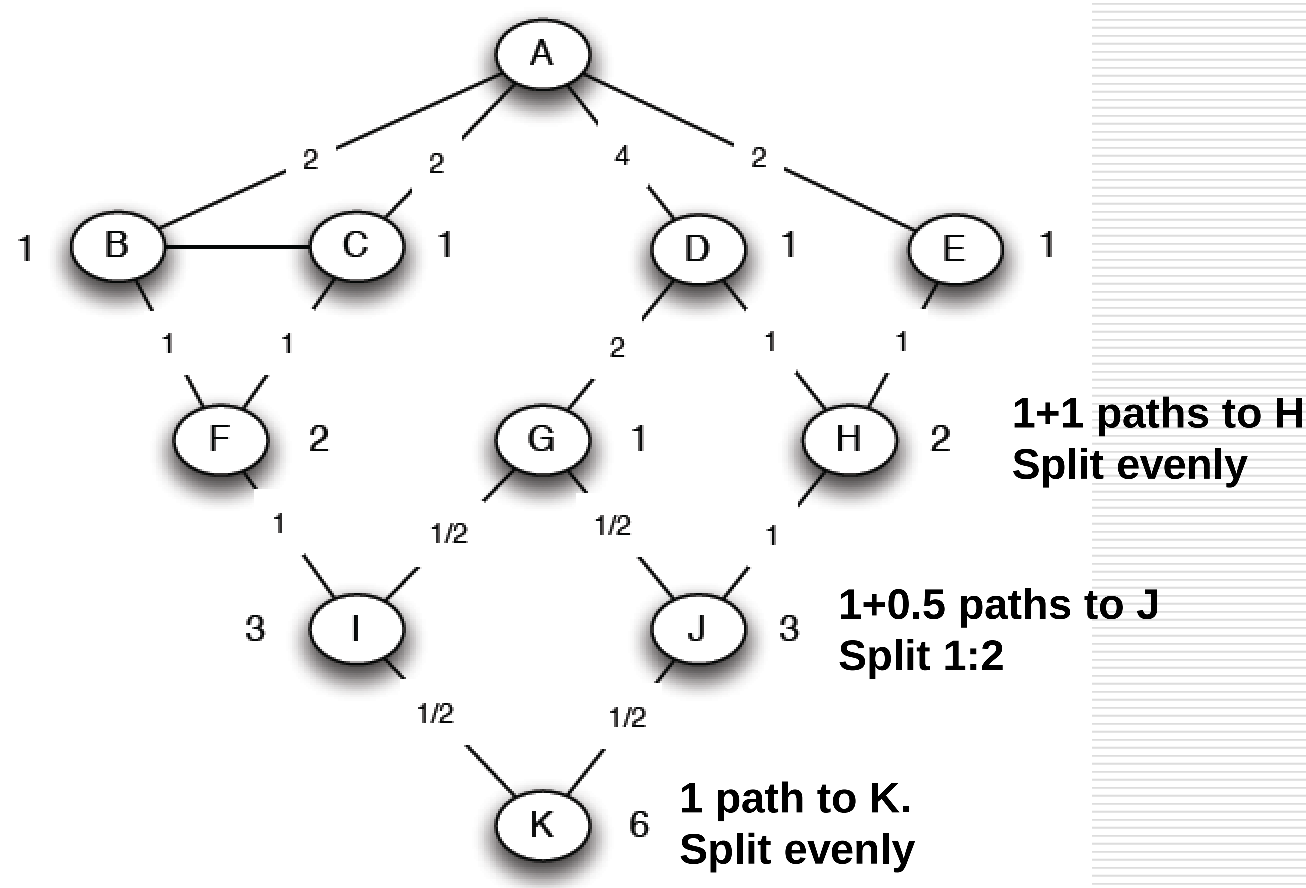
- **Compute betweenness by working up the tree:** If there are multiple paths count them fractionally



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

How to Compute Betweenness?

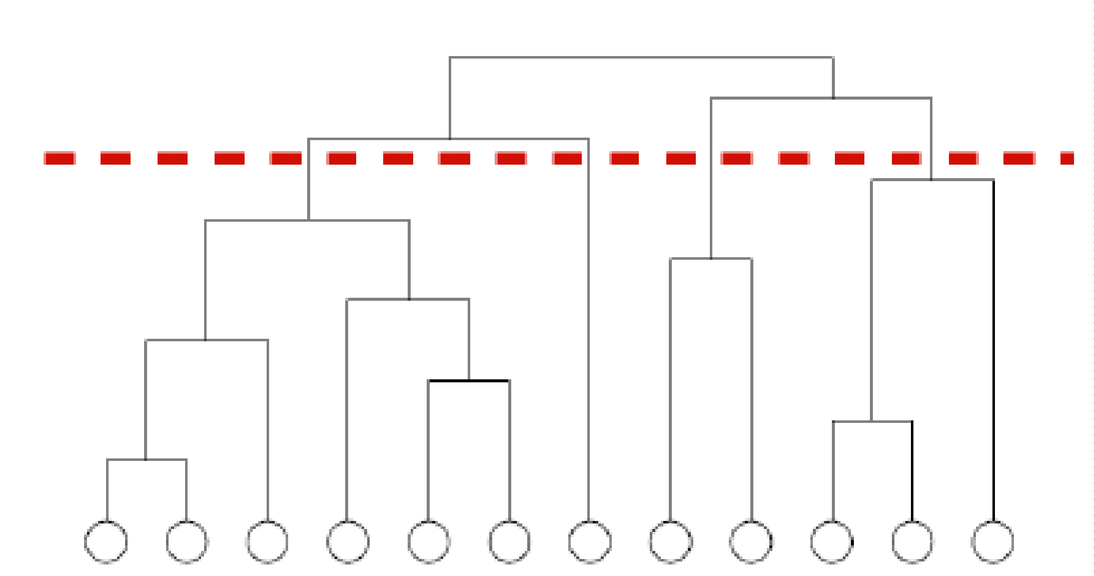
- **Compute betweenness by working up the tree:** If there are multiple paths count them fractionally



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

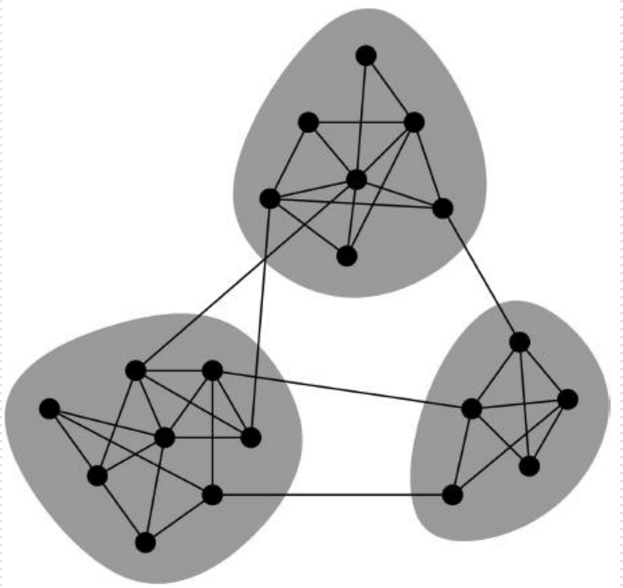
1. **How to compute betweenness?**
2. **How to select the number of clusters?**



Network Communities

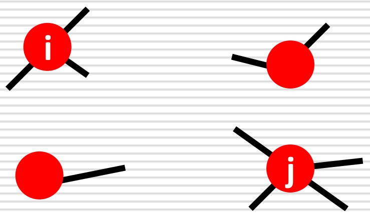
- **Communities:** sets of **tightly connected nodes**
- Define: **Modularity Q**
  - A measure of how well a network is partitioned into communities
  - Given a partitioning of the network into groups $s \in S$:

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



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

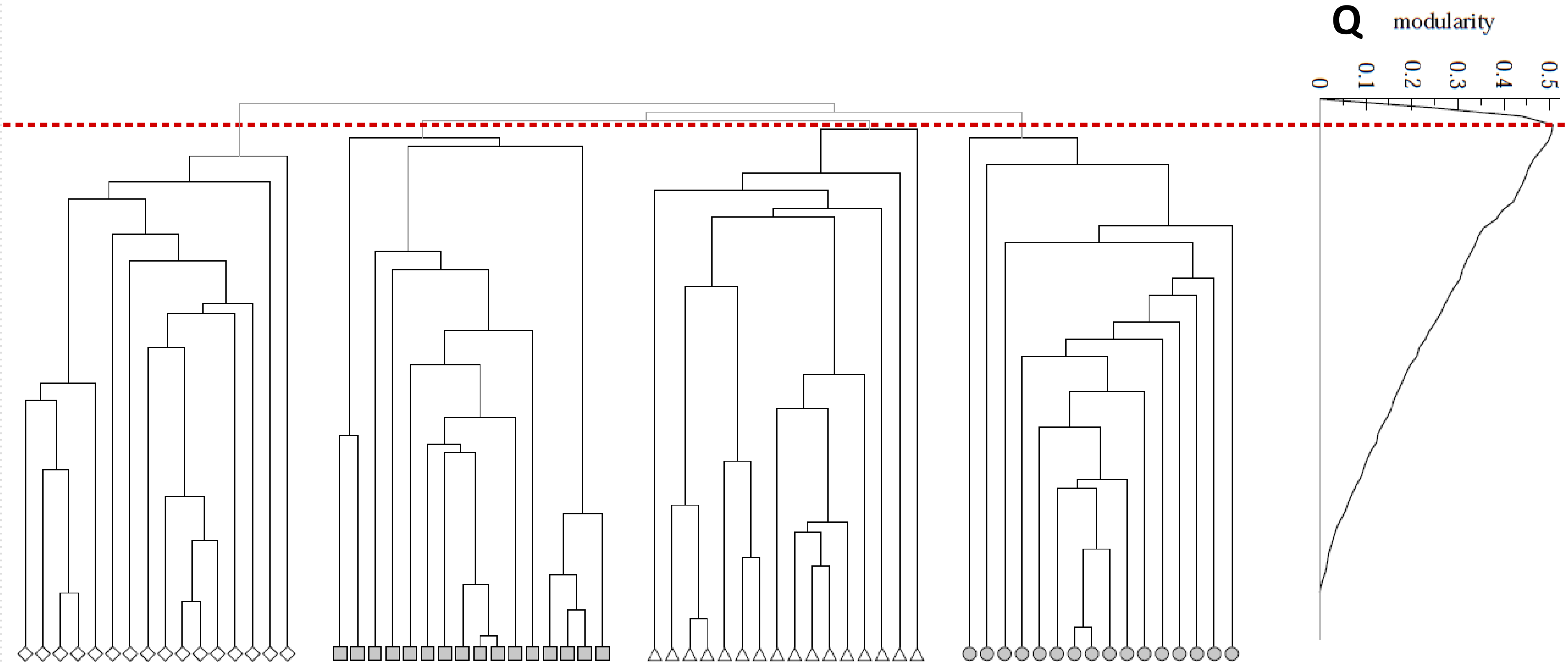
- **Modularity of partitioning S of graph G:**
  - **Q $\propto \sum_{s \in S}$ [(# edges within group *s*) – (expected # edges within group *s*)]**
  - $$Q(G,S) = \underbrace{\frac{1}{2m}}_{\text{Normalizing cost.: } -1<Q<1} \sum_{s \in S} \sum_{i \in s} \sum_{j \in s} \left(A_{ij} - \frac{k_i k_j}{2m}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

$A_{ij} = 1$ if $i \rightarrow j$, 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

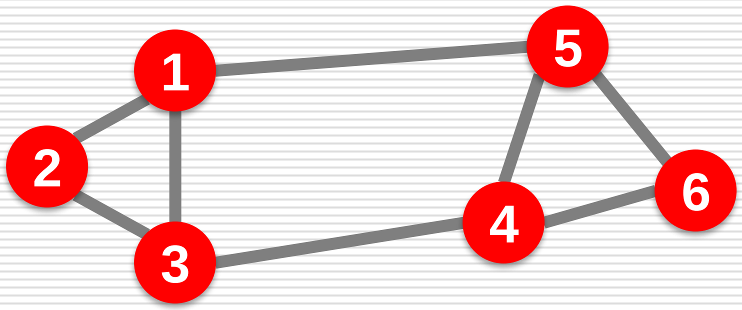
- **Modularity is useful for selecting the number of clusters:**



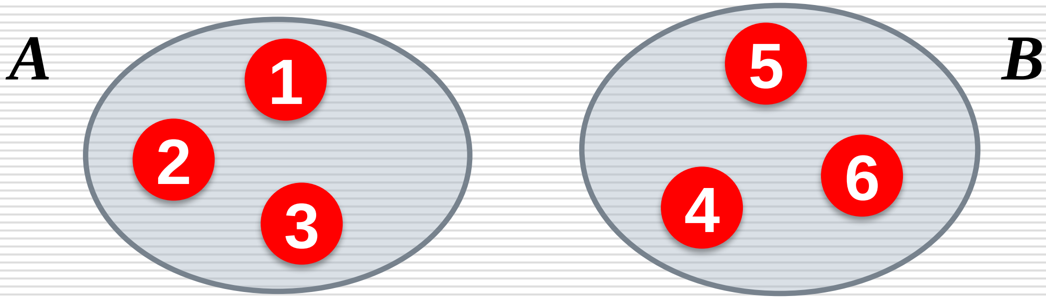
Spectral Clustering

Graph Partitioning

- **Undirected graph $G(V, E)$:**



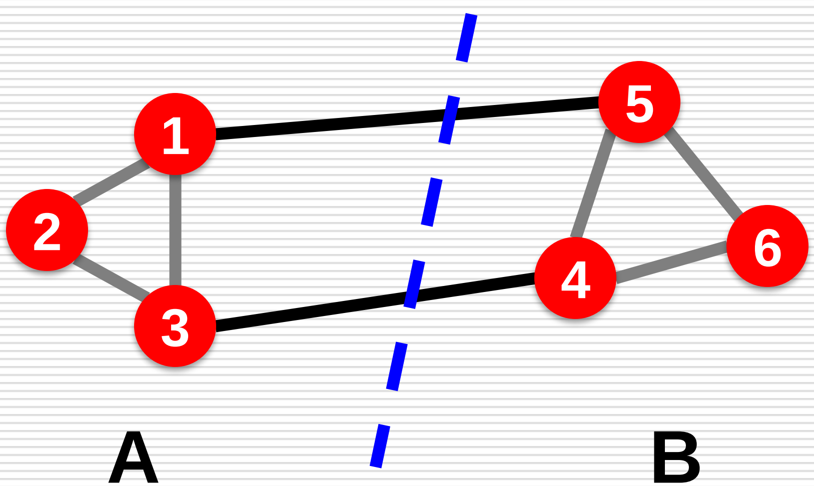
- **Bi-partitioning task:**
  - Divide vertices into two disjoint groups A, B



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

Graph Partitioning

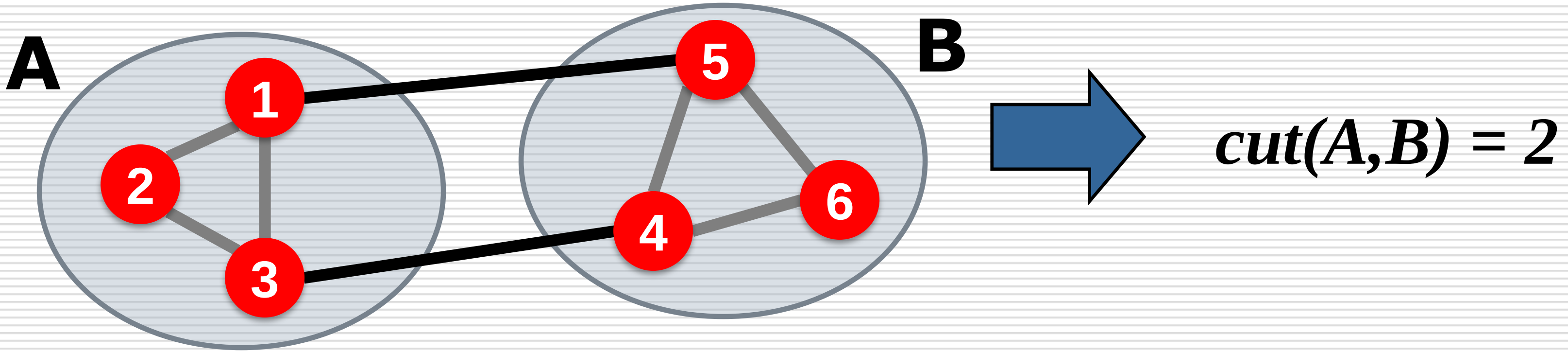
- **What makes a good partition?**
  - Maximize the number of within-group connections
  - Minimize the number of between-group connections



Graph Cuts

- Express partitioning objectives as a function of the "edge cut" of the partition
- **Cut:** Set of edges with only one vertex in a group:

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

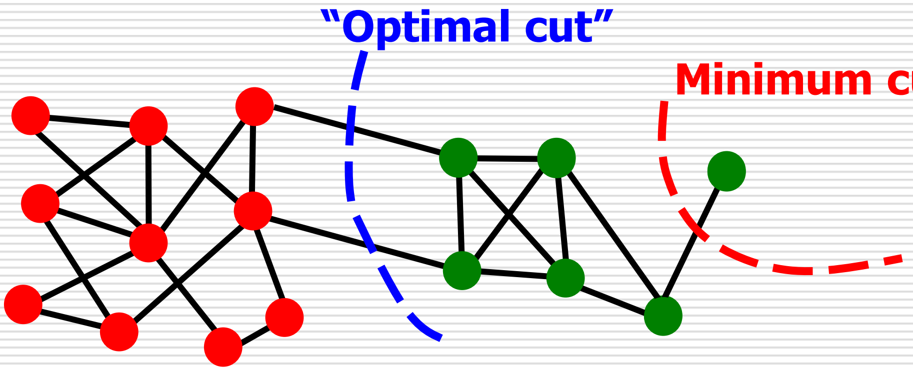


Graph Cut Criterion

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

$$\textbf{arg min}_{A,B}\ \textit{cut(A,B)}$$

- **Degenerate case:**



- **Problem:**
  - Only considers external cluster connections
  - Does not consider internal cluster connectivity

Graph Cut Criteria

- **Criterion: Normalized-cut** [Shi-Malik, ’97]
  - Connectivity between groups relative to the density of each group

$$ncut(A,B) = \frac{cut(A,B)}{vol(A)} + \frac{cut(A,B)}{vol(B)}$$

$\boldsymbol{vol(A)}$**:** total weight of the edges with at least one endpoint in $\boldsymbol{A}$: $\boldsymbol{vol(A) = \sum_{i \in A} k_i}$

- **Why use this criterion?**
  - Produces more balanced partitions

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

Spectral Graph Partitioning

- ***A***: adjacency matrix of undirected **G**
  - $\boldsymbol{A_{ij}}$ **=1** if $(\boldsymbol{i}, \boldsymbol{j})$ is an edge, else **0**
- ***x*** is a vector in $\Re^n$ with components $(\boldsymbol{x_1}, \dots, \boldsymbol{x_n})$
  - Think of it as a label/value of each node of $\boldsymbol{G}$
- **What is the meaning of $A \cdot x$?**

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad y_i = \sum_{j=1}^{n} A_{ij} x_j = \sum_{(i,j) \in E} x_j$$

- **Entry y_i is a sum of labels x_j of neighbors of i**

What is the meaning of Ax?

- j^{th} **coordinate of $A \cdot x$:**
  - Sum of the ***x***-values of neighbors of ***j***
  - Make this a new value at node ***j***

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

$$\boldsymbol{A \cdot x = \lambda \cdot x}$$

- **Spectral Graph Theory:**
  - Analyze the "spectrum" of matrix representing $\boldsymbol{G}$
  - **Spectrum:** Eigenvectors $\boldsymbol{x_i}$ of a graph, ordered by the magnitude (strength) of their corresponding eigenvalues $\boldsymbol{\lambda_i}$:

$$\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$$

$$\lambda_1 \le \lambda_2 \le \ldots \le \lambda_n$$

Example: d-regular graph

- Suppose all nodes in $\boldsymbol{G}$ have degree $\boldsymbol{d}$ and $\boldsymbol{G}$ is connected
- **What are some eigenvalues/vectors of $\boldsymbol{G}$?**

  $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{\lambda} \cdot \boldsymbol{x}$ What is λ? What $\boldsymbol{x}$?

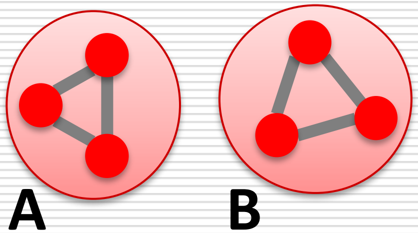
  - **Let's try: $\boldsymbol{x} = (\mathbf{1}, \mathbf{1}, \dots, \mathbf{1})$**
  - **Then: $\boldsymbol{A} \cdot \boldsymbol{x} = (\boldsymbol{d}, \boldsymbol{d}, \dots, \boldsymbol{d}) = \boldsymbol{\lambda} \cdot \boldsymbol{x}$. So: $\boldsymbol{\lambda} = \boldsymbol{d}$**
  - **We found eigenpair of $\boldsymbol{G}$: $\boldsymbol{x} = (\mathbf{1}, \mathbf{1}, \dots, \mathbf{1})$, $\boldsymbol{\lambda} = \boldsymbol{d}$**

Remember the meaning of $\boldsymbol{y} = \boldsymbol{A} \cdot \boldsymbol{x}$:

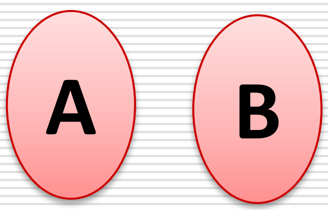
$$y_j = \sum_{i=1}^{n} A_{ij} x_i = \sum_{(j,i) \in E} x_i$$

Example: Graph on 2 components

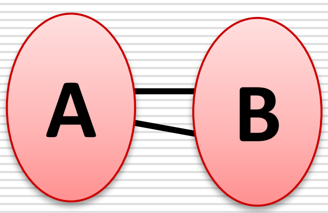
- **What if G is not connected?**
  - G has **2** components, each d-regular



- **What are some eigenvectors?**
  - $x =$ Put all 1s on A and 0s on B or vice versa
    - $x' = (\underbrace{1, \dots, 1}_{|A|}, \underbrace{0, \dots, 0}_{|B|})$ **then** $\mathrm{A} \cdot x' = (d, \dots, d, 0, \dots, 0)$
    - $x'' = (0, \overset{|A|}{\dots}, 0, 1, \overset{|B|}{\dots}, 1)$ **then** $A \cdot x'' = (0, \dots, 0, d, \dots, d)$
    - And so in both cases the corresponding $\lambda = d$
- **A bit of intuition:**



$\lambda_n = \lambda_{n-1}$



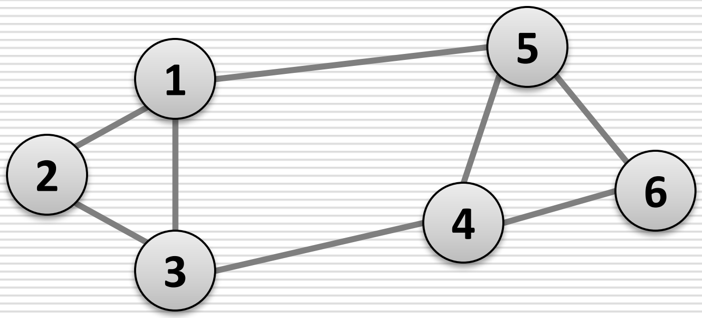
$\lambda_n - \lambda_{n-1} \approx 0$

2nd largest eigval. λ_{n-1} now has value very close to λ_n

Matrix Representations

Adjacency matrix (A):

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



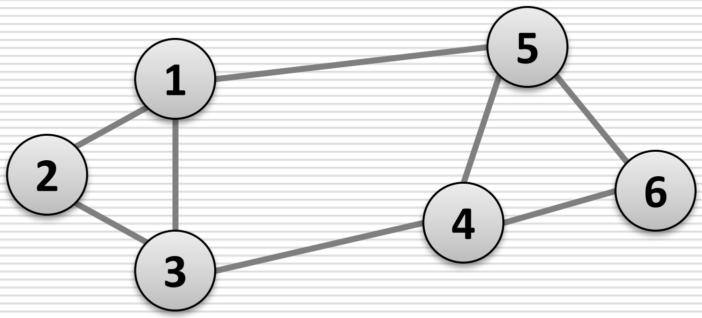
	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0

Important properties:

- Symmetric matrix
- Eigenvectors are real and orthogonal

Matrix Representations

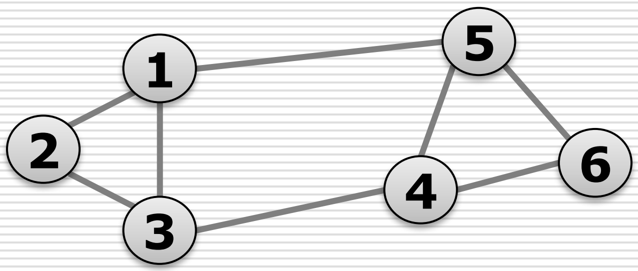
- **Degree matrix (D):**
  - $n \times n$ diagonal matrix
  - $D=[d_{ij}]$, d_{ii} = degree of node i



	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

Matrix Representations

- **Laplacian matrix (L):**
  - $\boldsymbol{n \times n}$ 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

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

- **What is trivial eigenpair?**
  - $\boldsymbol{x = (1, \ldots, 1)}$ then $\boldsymbol{L \cdot x = 0}$ and so $\boldsymbol{\lambda = \lambda_1 = 0}$
- **Important properties:**
  - **Eigenvalues** are non-negative real numbers
  - **Eigenvectors** are real and orthogonal

So far…

- **How to define a "good" partition of a graph?**
  - Minimize a given graph cut criterion

- **How to efficiently identify such a partition?**
  - Approximate using information provided by the eigenvalues and eigenvectors of a graph

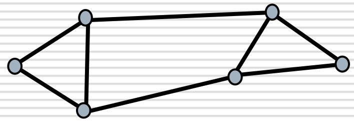
- **Spectral Clustering**

Spectral Clustering Algorithms

- **Three basic stages:**
  - **1) Pre-processing**
    - Construct a matrix representation of the graph
  - **2) Decomposition**
    - Compute eigenvalues and eigenvectors of the matrix
    - Map each point to a lower-dimensional representation based on one or more eigenvectors
  - **3) Grouping**
    - Assign points to two or more clusters, based on the new representation

Spectral Partitioning Algorithm

- **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 ***x*** of the matrix ***L***
  - Map vertices to corresponding components of λ_2

λ=

0.0
1.0
3.0
3.0
4.0
5.0

X =

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

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

How do we now find the clusters?

Spectral Partitioning

- **3) Grouping:**
  - Sort components of reduced 1-dimensional vector
  - Identify clusters by splitting the sorted vector in two
- **How to choose a splitting point?**
  - Naïve approaches:
    - Split at **0** or median value
  - More expensive approaches:
    - Attempt to minimize normalized cut in 1-dimension (sweep over ordering of nodes induced by the eigenvector)

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

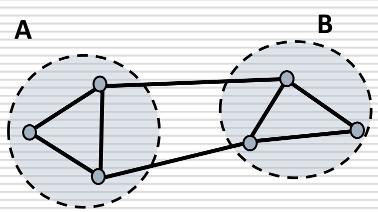
Split at 0:

Cluster A: Positive points

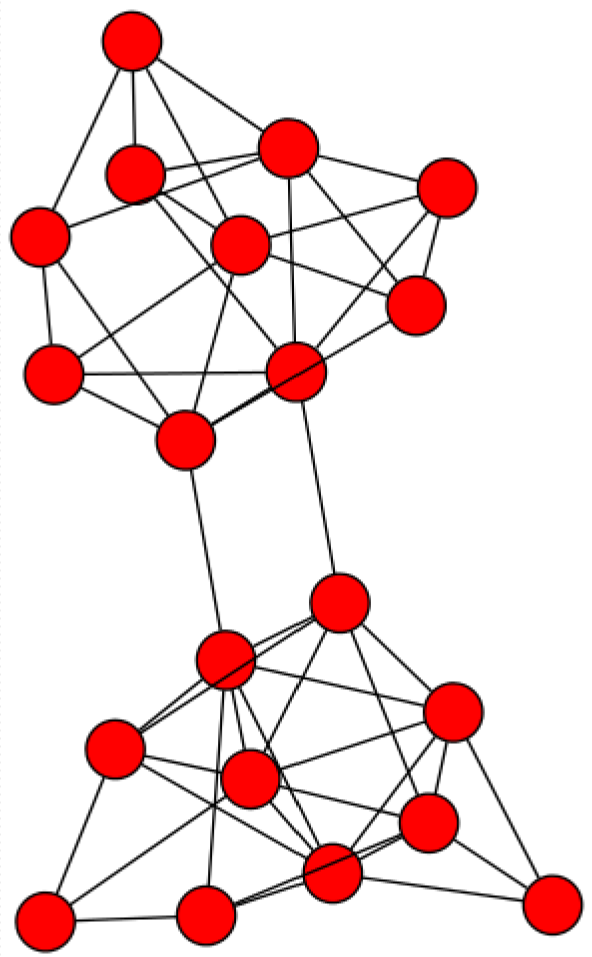
Cluster B: Negative points

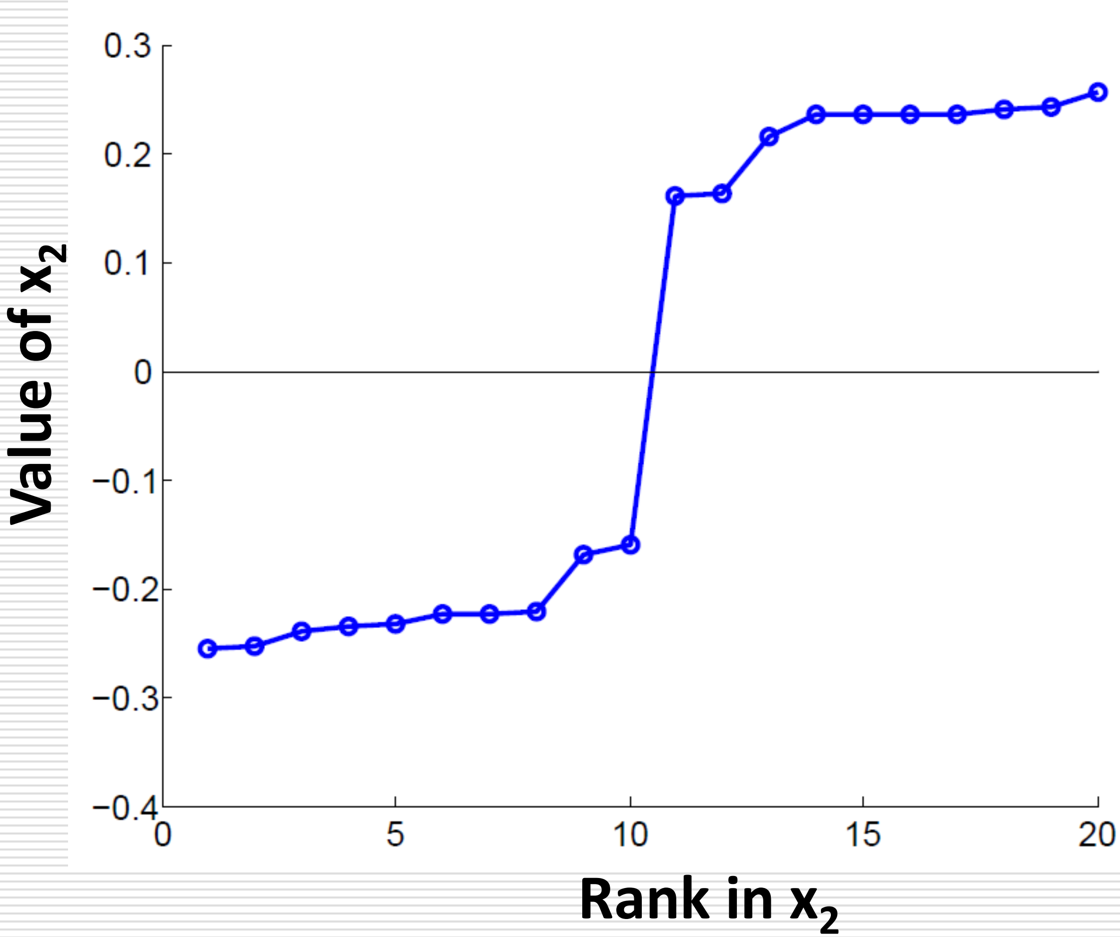
1	0.3
2	0.6
3	0.3

4	-0.3
5	-0.3
6	-0.6

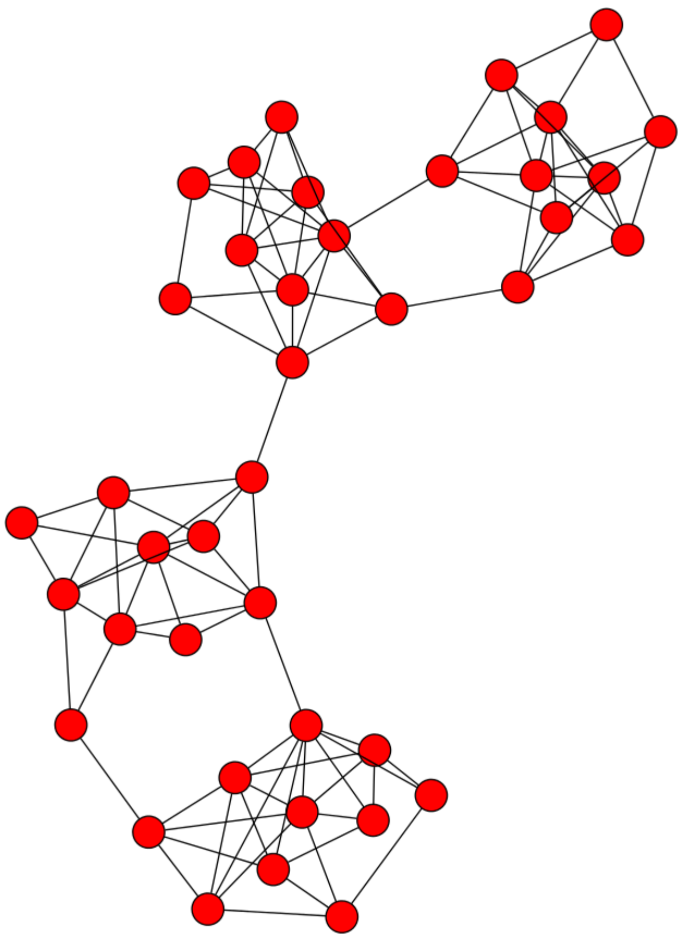


Example: Spectral Partitioning

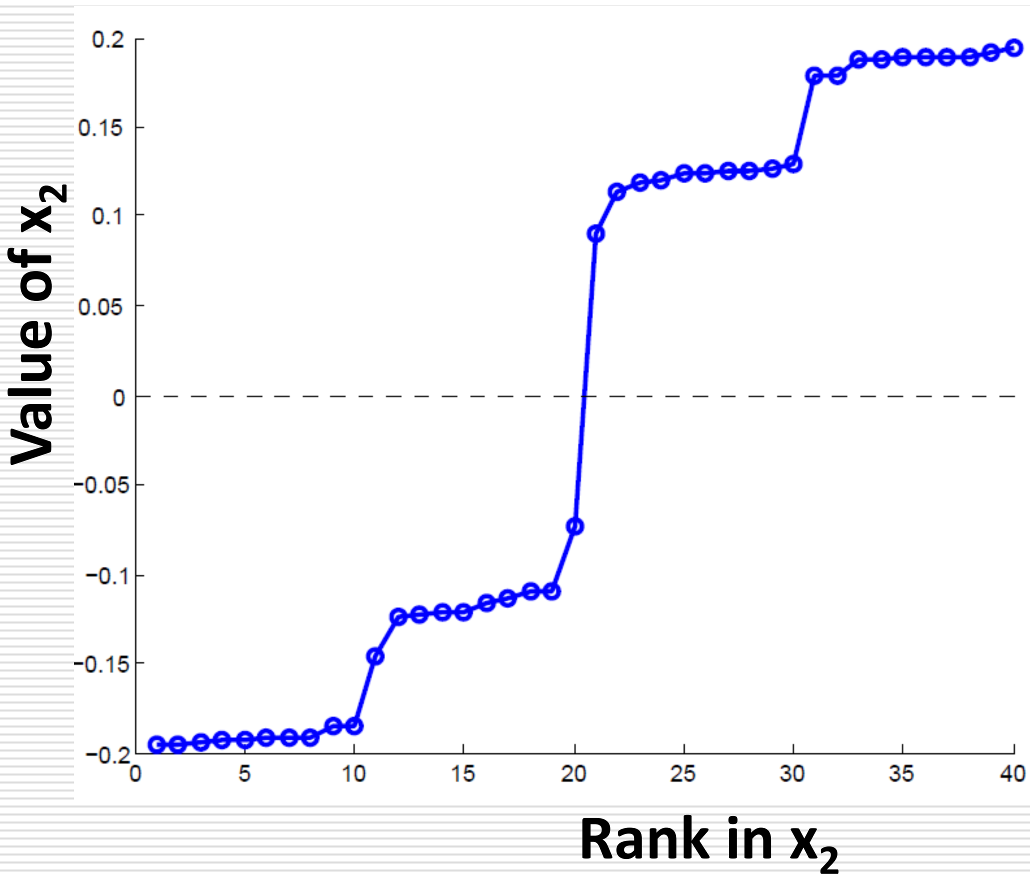




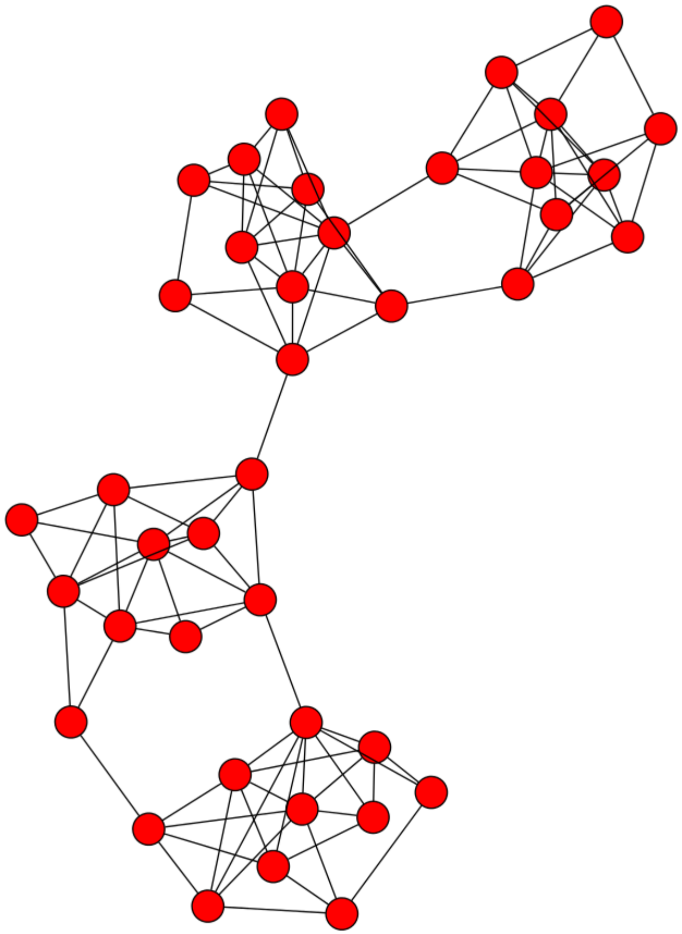
Example: Spectral Partitioning

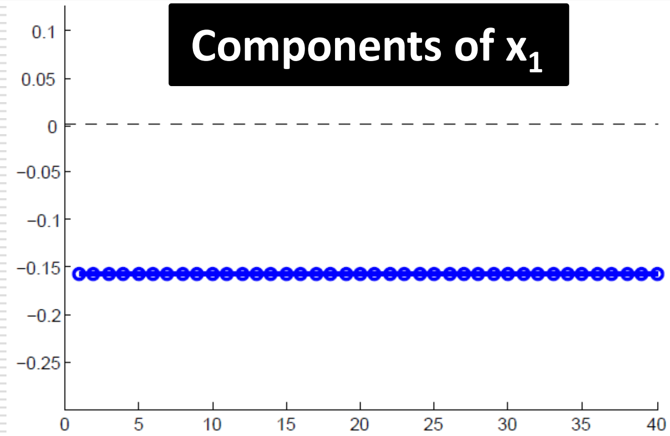


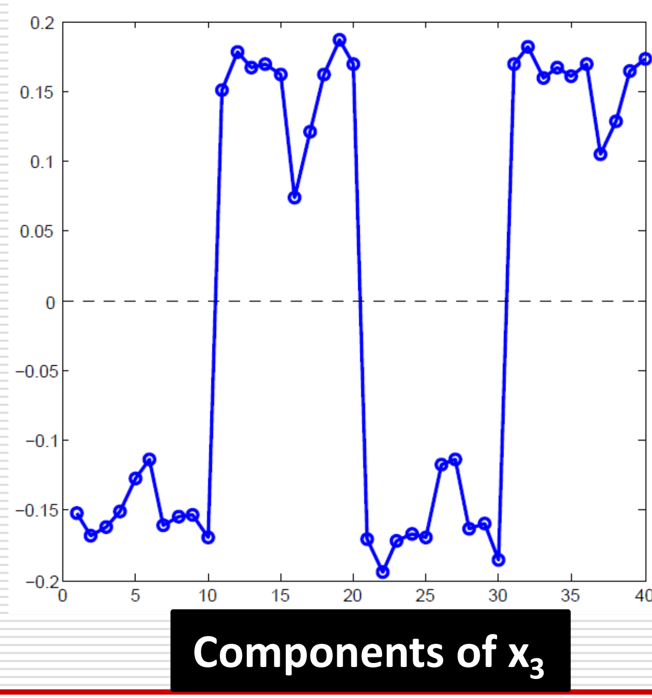
Components of x_2



Example: Spectral Partitioning







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 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 in recent papers
    - A preferable approach…

Why use multiple eigenvectors?

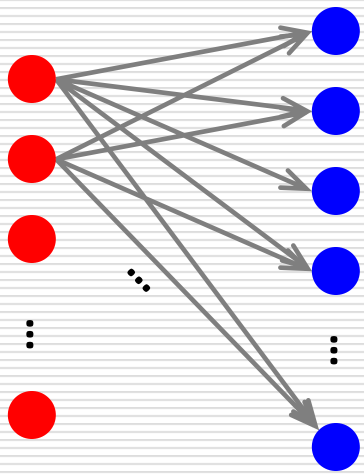
- **Approximates the optimal cut** [Shi-Malik, '00]
  - Can be used to approximate optimal *k*-way normalized cut
- **Emphasizes cohesive clusters**
  - Increases the unevenness in the distribution of the data
  - Associations between similar points are amplified, associations between dissimilar points 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

Analysis of Large Graphs: Trawling

[Kumar et al. '99]

Trawling

- Searching for small communities in the Web graph
- What is the signature of a community / discussion in a Web graph?



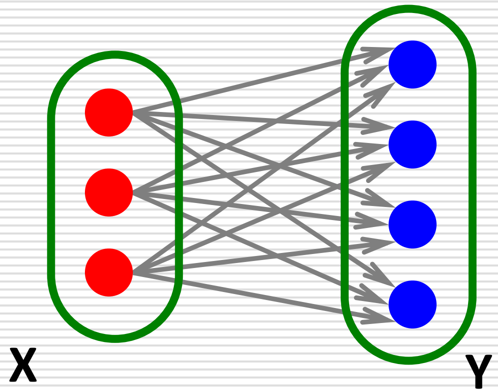
Dense 2-layer graph

Use this to define "topics":
What the same people on
the left talk about on the right
Remember HITS!

Intuition: Many people all talking about the same things

Searching for Small Communities

- **A more well-defined problem:**
  Enumerate complete bipartite subgraphs $K_{s,t}$
  - Where $K_{s,t}$: s nodes on the "left" where each links to the same t other nodes on the "right"



$K_{3,4}$

|X| = s = 3
|Y| = t = 4

Fully connected

Frequent Itemset Enumeration

[Agrawal-Srikant '93]

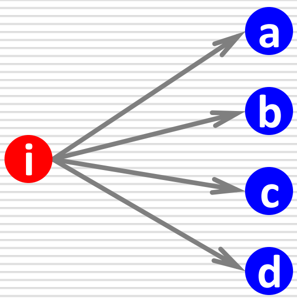
- **Market basket analysis.** Setting:
  - **Market:** Universe $\boldsymbol{U}$ of $\boldsymbol{n}$ items
  - **Baskets:** $\boldsymbol{m}$ subsets of $\boldsymbol{U}$: $\boldsymbol{S_1, S_2, \ldots, S_m \subseteq U}$
    ($\boldsymbol{S_i}$ is a set of items one person bought)
  - **Support:** Frequency threshold $\boldsymbol{f}$
- **Goal:**
  - **Find all subsets $\boldsymbol{T}$ s.t. $\boldsymbol{T \subseteq S_i}$ of at least $\boldsymbol{f}$ sets $\boldsymbol{S_i}$**
    (items in $\boldsymbol{T}$ were bought together at least $\boldsymbol{f}$ times)
- **What's the connection between the itemsets and complete bipartite graphs?**

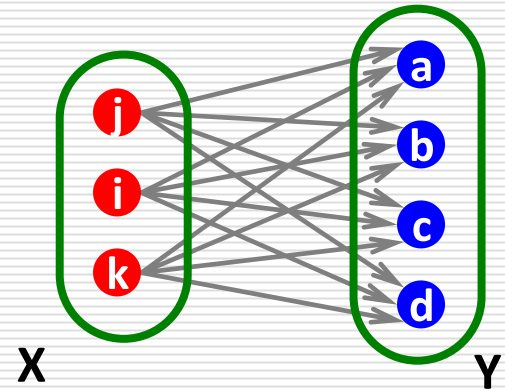
From Itemsets to Bipartite $K_{s,t}$

[Kumar et al. '99]

Frequent itemsets = complete bipartite graphs!

- **How?**
  - View each node ***i*** as a set S_i of nodes ***i*** points to
  - $K_{s,t}$ = a set ***Y*** of size ***t*** that occurs in ***s*** sets S_i
  - Looking for $K_{s,t}$ → set of frequency threshold to ***s*** and look at layer ***t*** – all frequent sets of size ***t***



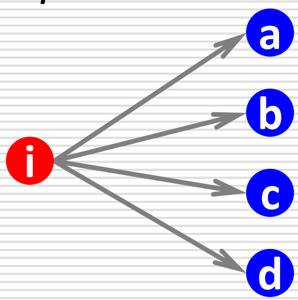


s … minimum support (|X|=s)
t … itemset size (|Y|=t)

From Itemsets to Bipartite $K_{s,t}$

[Kumar et al. '99]

View each node *i* as a set S_i of nodes *i* points to



S_i={a,b,c,d}

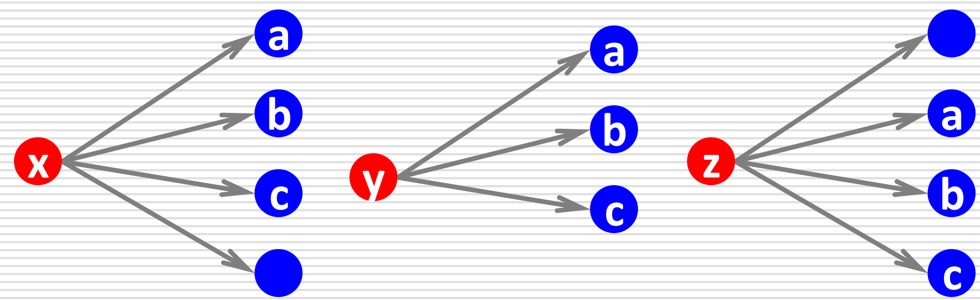
Find frequent itemsets:

s … minimum support

t … itemset size

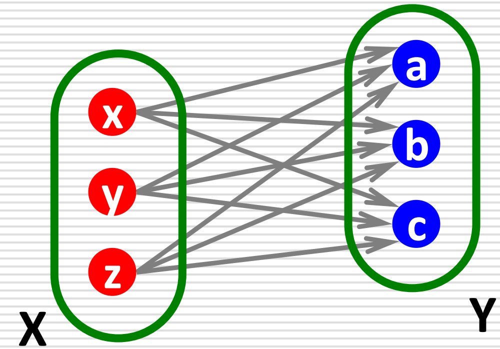
Say we find a frequent itemset *Y={a,b,c}* of supp *s*

So, there are *s* nodes that link to all of {a,b,c}:

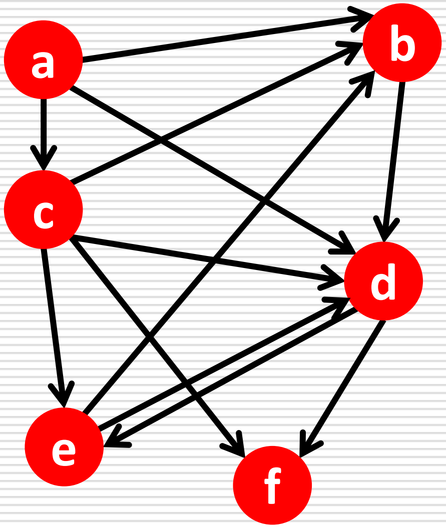


We found $K_{s,t}$!

$K_{s,t}$ = a set *Y* of size *t* that occurs in *s* sets S_i



Example (1)



Itemsets:

a = {b,c,d}

b = {d}

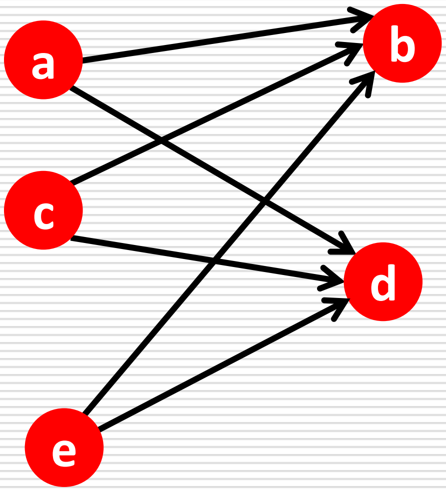
c = {b,d,e,f}

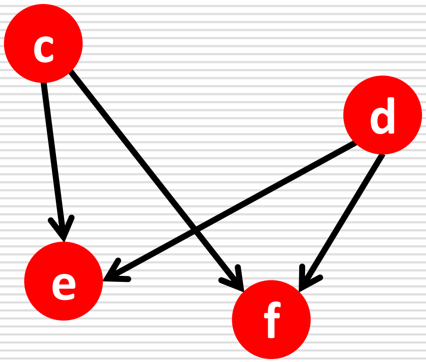
d = {e,f}

e = {b,d}

f = {}

- **Support threshold s=2**
  - **{b,d}**: support 3
  - **{e,f}**: support 2
- And we just found 2 bipartite subgraphs:





Example (2)

Example of a community from a web graph

A community of Australian fire brigades

Nodes on the right	Nodes on the left
NSW Rural Fire Service Internet Site	New South Wales Fir...ial Australian Links
NSW Fire Brigades	Feuerwehrlinks Australien
Sutherland Rural Fire Service	FireNet Information Network
CFA: County Fire Authority	The Cherrybrook Rur...re Brigade Home Page
"The National Cente...ted Children's Ho...	New South Wales Fir...ial Australian Links
CRAFTI Internet Connexions-INFO	Fire Departments, F... Information Network
Welcome to Blackwoo... Fire Safety Serv...	The Australian Firefighter Page
The World Famous Guestbook Server	Kristiansand brannv...dens brannvesener...
Wilberforce County Fire Brigade	Australian Fire Services Links
NEW SOUTH WALES FIR...ES 377 STATION	The 911 F,P,M., Fir...mp; Canada A Section
Woronora Bushfire Brigade	Feuerwehrlinks Australien
Mongarlowe Bush Fire – Home Page	Sanctuary Point Rural Fire Brigade
Golden Square Fire Brigade	Fire Trails "l...ghters around the...
FIREBREAK Home Page	FireSafe – Fire and Safety Directory
Guises Creek Volunt...fficial Home Page...	Kristiansand Firede...departments of th...

[Kumar, Raghavan, Rajagopalan, Tomkins: Trawling the Web for emerging cyber-communities 1999]

Acknowledgement

- Slides are adapted from:
  - Prof. Jeffrey D. Ullman
  - Dr. Anand Rajaraman
  - Dr. Jure Leskovec