

**Note to other teachers and users of these slides:** We would be delighted if you found this our material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. If you make use of a significant portion of these slides in your own lecture, please include this message, or a link to our web site: <http://www.mmds.org>

# Analysis of Large Graphs: Community Detection

Mining of Massive Datasets

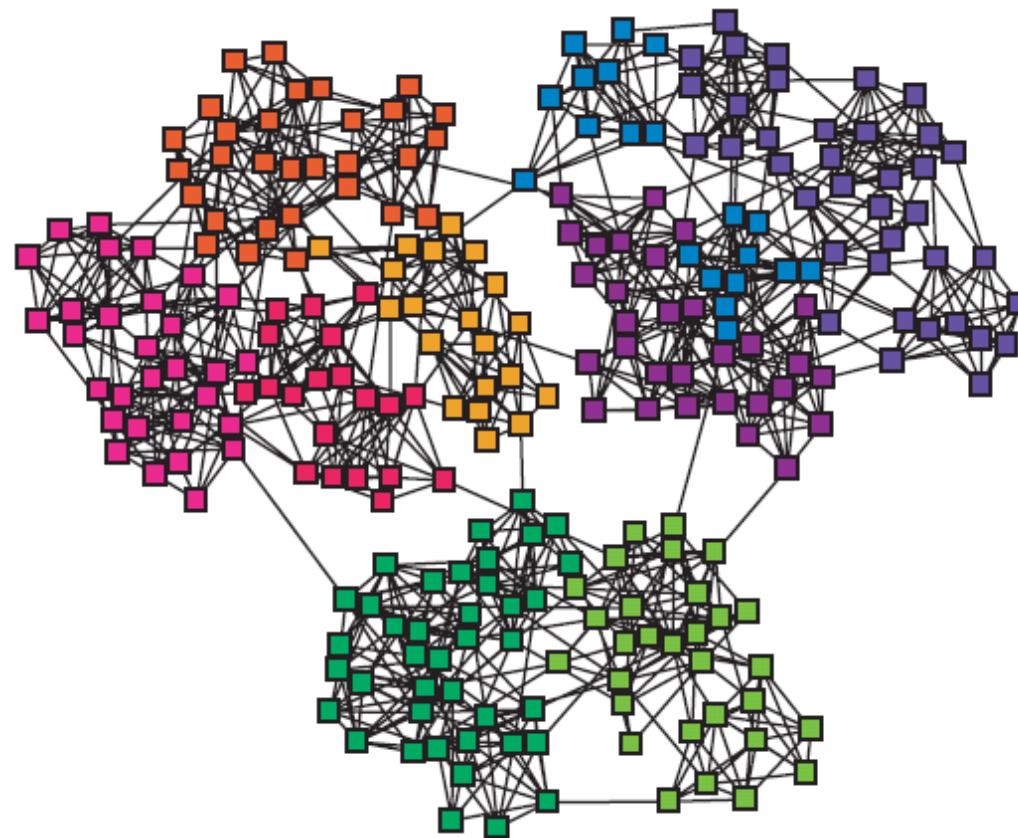
Jure Leskovec, Anand Rajaraman, Jeff Ullman  
Stanford University

<http://www.mmds.org>

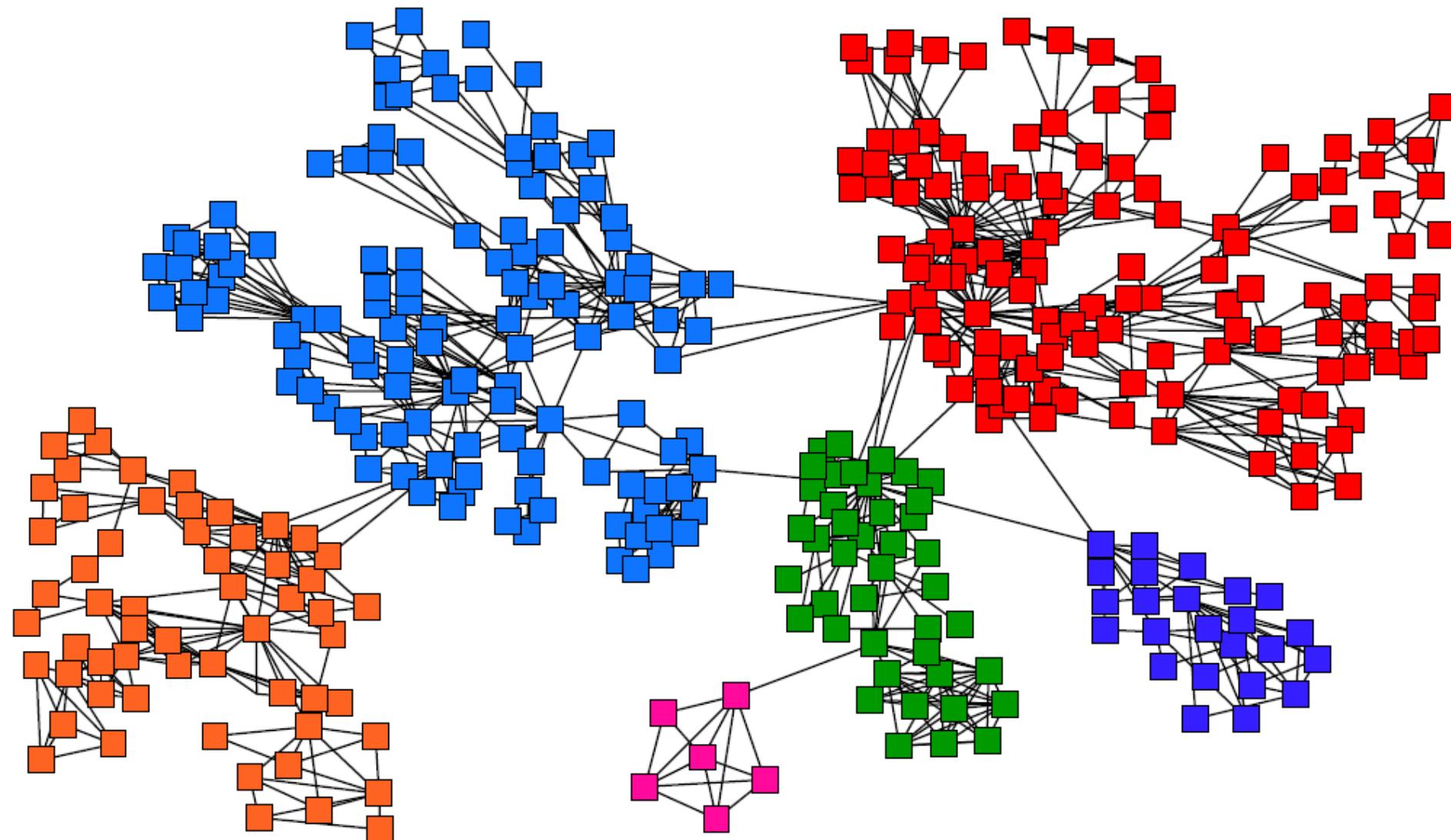


# Networks & Communities

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

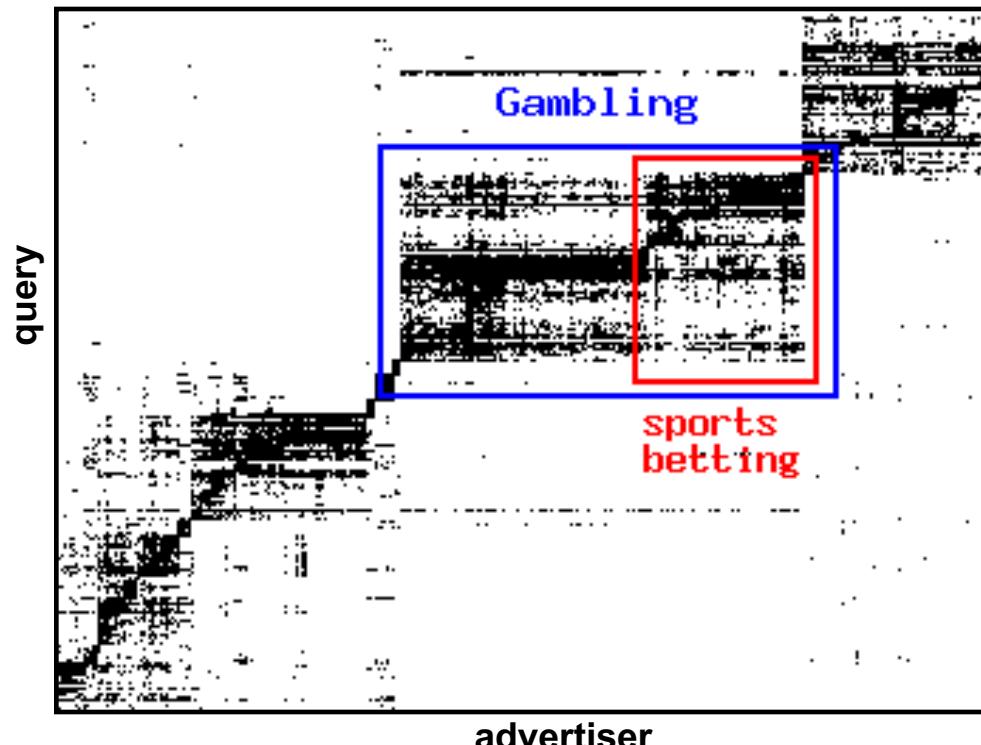


# Goal: Find Densely Linked Clusters



# Micro-Markets in Sponsored Search

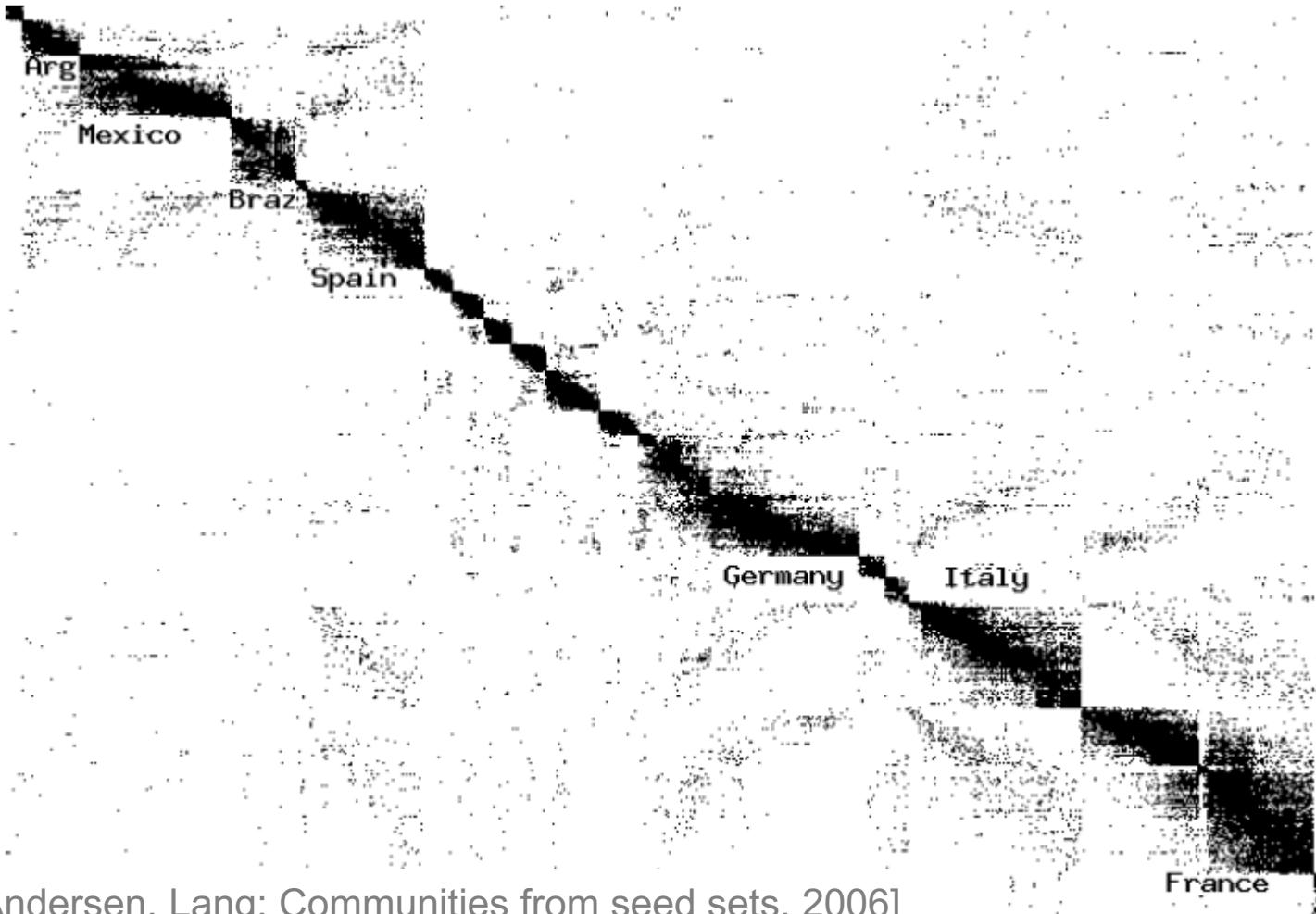
- Find micro-markets by partitioning the query-to-advertiser graph:



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

# Movies and Actors

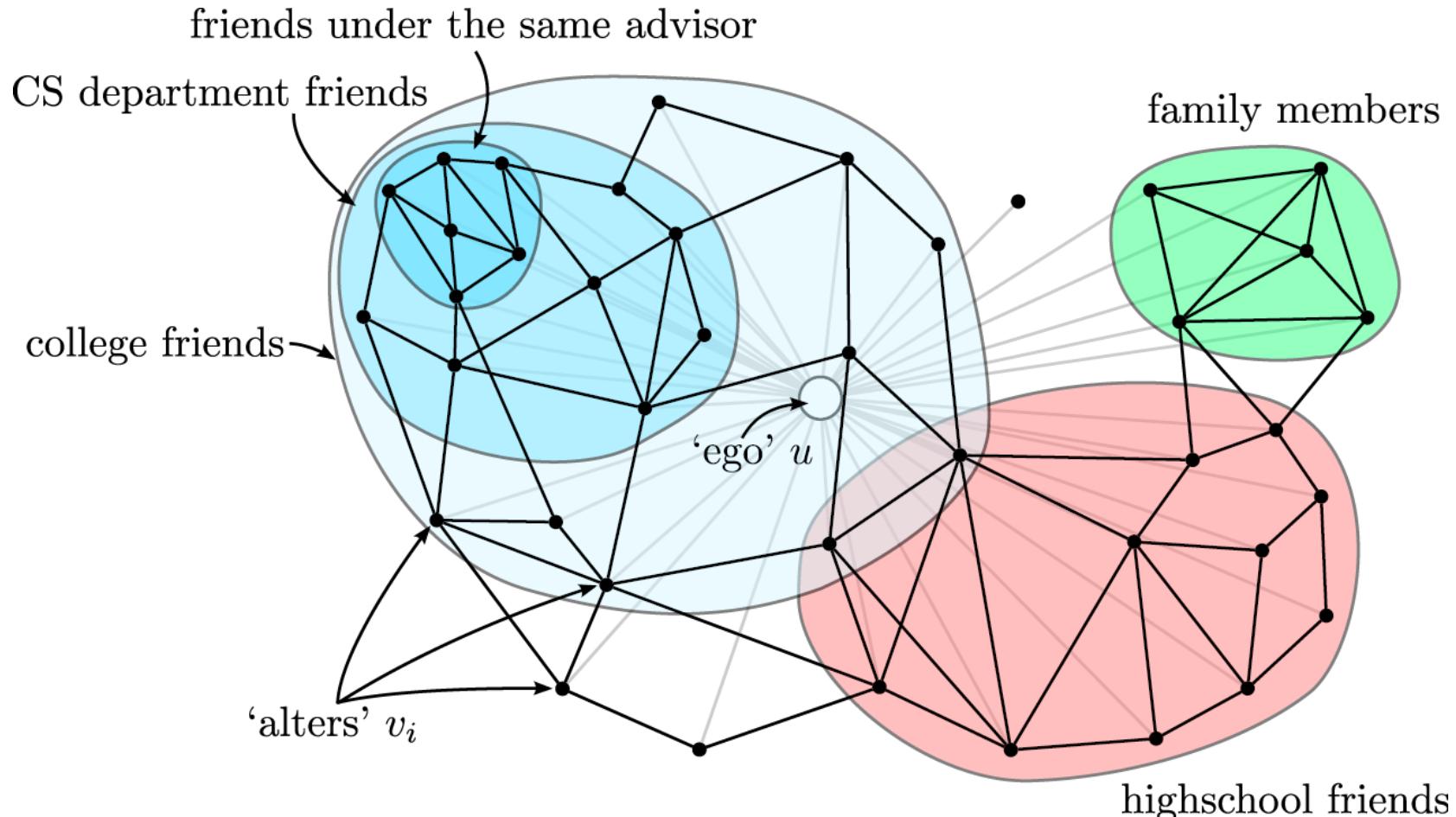
## ■ Clusters in Movies-to-Actors graph:



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

# Twitter & Facebook

## ■ Discovering social circles, circles of trust:



## WEIGHTED METRIC GRAPHS → Clustering

Clustering. Given a set  $U$  of  $n$  objects labeled  $p_1, \dots, p_n$ , classify into coherent groups.

↑  
photos, documents, micro-organisms

$$\langle S, \text{distance}(\bar{x}, \bar{y}) \rangle$$

$S \subseteq \mathbb{R}^n$

Distance function. Numeric value specifying "closeness" of two objects:

$$\{ \text{distance}(p_i, p_j)$$

$$\uparrow$$

number of corresponding pixels whose intensities differ by some threshold

$$G = \{C_1, C_2, \dots, C_k\}$$

Fundamental problem. Divide into clusters so that points in different clusters **are far apart**.

- Routing in mobile ad hoc networks.
- Identify patterns in gene expression.
- Document categorization for web search.
- Similarity searching in medical image databases
- Skycat: cluster  $10^9$  sky objects into stars, quasars, galaxies.

$$\begin{aligned} C_r &\subseteq U \\ C_r \cap C_g &= \emptyset \\ \bigcup_{r=1}^k C_r &= U \end{aligned}$$



$$D(i, \delta) \equiv d(p_i, p_\delta)$$

## Clustering of Maximum Spacing

k-Clustering. Divide objects into k non-empty groups.

Distance function. Assume it satisfies several natural properties.

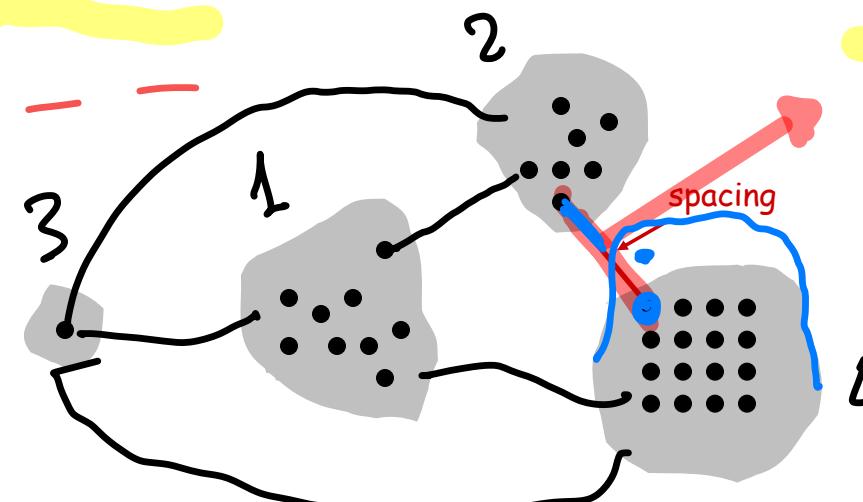
- $d(p_i, p_j) = 0$  iff  $p_i = p_j$  (identity of indiscernibles)
- $d(p_i, p_j) \geq 0$  (nonnegativity)
- $d(p_i, p_j) = d(p_j, p_i)$  (symmetry)

Spacing. Min distance between any pair of points in different clusters.

Clustering of maximum spacing. Given an integer  $k$ , find a  $k$ -clustering of maximum spacing.

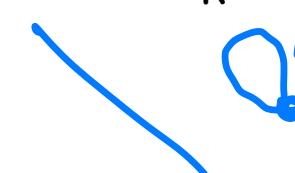
$$P = \langle P_x, P_y \rangle$$

$$U \subseteq \mathbb{R}^2$$



$$\rightarrow \text{SPACING}(C_5, C_{5_2}) = ?$$

$$k = 4$$



$C_i = \{C_1, C_2, \dots, C_k\}$

space:  $U \times C_i \rightarrow \mathbb{R}^+$

$$\forall i, \delta = 1, \dots, k$$

$$d(C_i, C_\delta) = \text{space}(C_i, C_\delta) =$$

$$\min \{ d(p_i, p_\delta) : p_i \in C_i, p_\delta \in C_\delta \}$$

MAIN IDEA:

PROBLEM MODEL

$$G: \bar{P} \rightarrow \{1, \dots, K\}$$
$$\zeta(3) = K-1$$

USE WEIGHTED GRAPHS TO REPRESENT THE INSTANCE!

$G(V, E)$ ,  $d: E \rightarrow \mathbb{R}^+$ , where

- $V = U = \{p_1, \dots, p_n\}$ .
- $E = \{\text{ALL non-ordered pairs in } V\}$  (i.e. complete graph)
- $\text{Cost}(e) = d(p_i, p_j)$  for any  $e = \{p_i, p_j\}$

SOL. ADMISS:

COLLEZIONE ??

$$G = \{C_1, \dots, C_K\} \quad ??$$
$$C_j \subseteq P$$

COMB  
RAPP.  
COME STRUT.  
UNA K-PART.

??

$$\bigcup_{S=1}^K C_S = P$$
$$\forall S_1 \neq S_2 : C_{S_1} \cap C_{S_2} = \emptyset$$

Instance:  $\mathbb{R}^2 \rightarrow \langle S, d \rangle$

GEOMETRICA:

- $P \subseteq \mathbb{R}^2$ .  $|P| = n$
- $P = \{P_1, P_2, \dots, P_i, \dots, P_n\}$
- $P_i = \langle P_{i,x}, P_{i,y} \rangle \quad P_{i,x}, P_{i,y} \in \mathbb{R}$
- $d: \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}^+$
- $d(P_i, P_j) \leq :$
- $K \in \mathbb{N} \quad 1 \leq K \leq n-1$
- $I \leftarrow \text{IST.}$

$C \leftarrow \text{SOL. ADM.}$

$\{C_1 = \{v_1\}, \dots, C_n = \{v_n\}\} \in \mathbb{C} \rightarrow \text{NO - SOL. AMM.}$

Greedy Clustering Algorithm

$K=3$

Init. Conf.

$n$  clusters

$n-K$

Single-link k-Clustering algorithm.

- Form a graph on the vertex set  $U$ , corresponding to  $n$  clusters.
- Find the **closest** pair (edge) of objects  $(p, p')$  such that  $p$  &  $p'$  are not in the same cluster, and add **an edge** between them: so merging 2 clusters.
- Repeat  $n-k$  times until there are exactly  $k$  clusters.

$|U|=n$

$\downarrow$  1 Merge (min Edg)

$\downarrow$  2 Merge (min Edg)

$\downarrow$   $n-2$  Merge

$\vdots$   $k$  Merge

$n-k$  min Edg

$-k$   
 $\{C_1, C_2, \dots, C_k\}$   
 Lb  
 SOL-AMM.

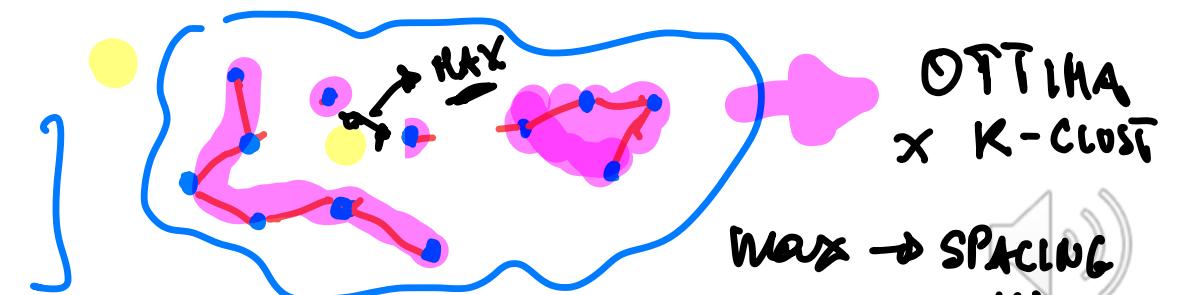
Key Obs. 1. This procedure is precisely Kruskal's algorithm (except we stop when there are  $k$  connected components).

Key Obs. 2. Equivalent to finding an MST  $T$  and deleting the  $k-1$  most expensive edges from  $T$  (thus forming  $k$  connected components).

$K=4 \rightarrow K-1=3$

Proofs of 1 and 2: Exercises

$$[e_1 \leq e_2 \leq e_3 \leq \dots \leq e_K \leq e_{K+1} \leq \dots \leq e_m]$$



OTTIMA  
 x K-CLOSEST

max → SPACING  
 MINIMO

## Greedy Clustering Algorithm: Analysis

$T^*$  ē un MST per G

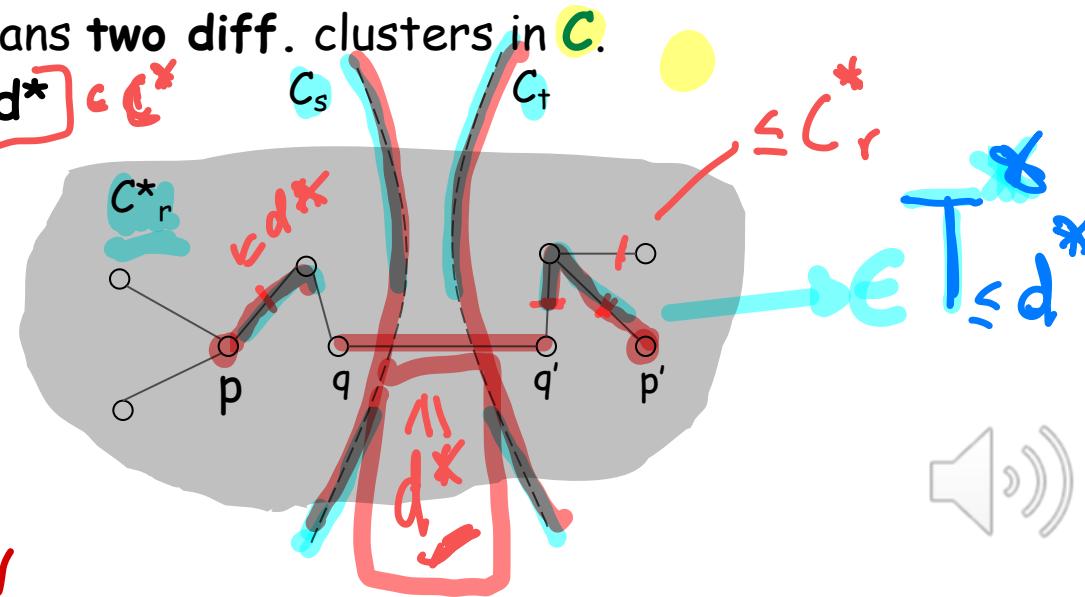
Theorem. Let  $C^*$  denote the clustering  $C^*_1, \dots, C^*_k$  formed by deleting the  $k-1$  most expensive edges of an MST. Then,  $C^*$  is a  $k$ -clustering of maximal spacing.

$$d^* = \text{spacing} = d(C^*)$$

III  
 ↗(e) dove  
 e ē il  $(k-1)$ -mo  
 arco + pesante

spacing ( $C$ )  $\leq d^*$

$d^* = \text{spacing } (C^*)$



$$C_r^* \subseteq P \equiv \{\text{points}\}$$

included

$$\geq e_{k-1} \downarrow * = d^*$$

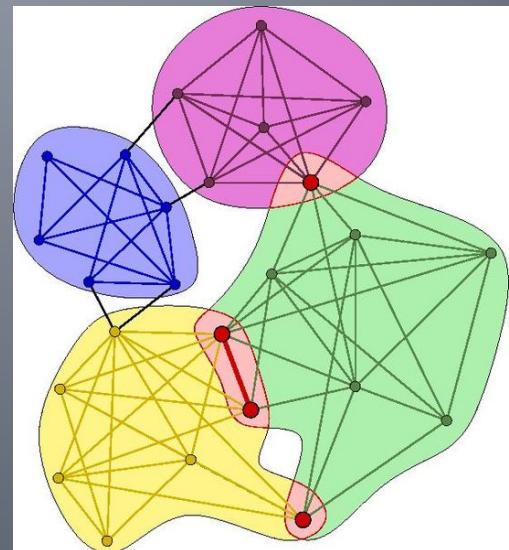
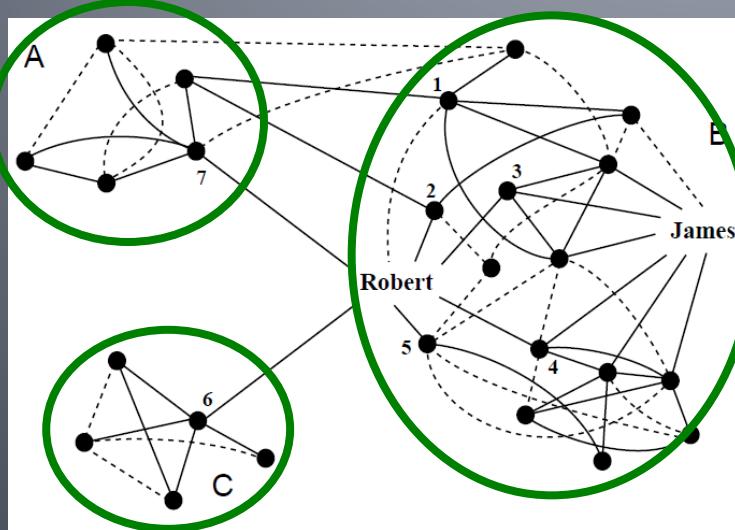
there must exist!



- UNWEIGHTED (NON-METRIC!) GRAPHS

# Community Detection

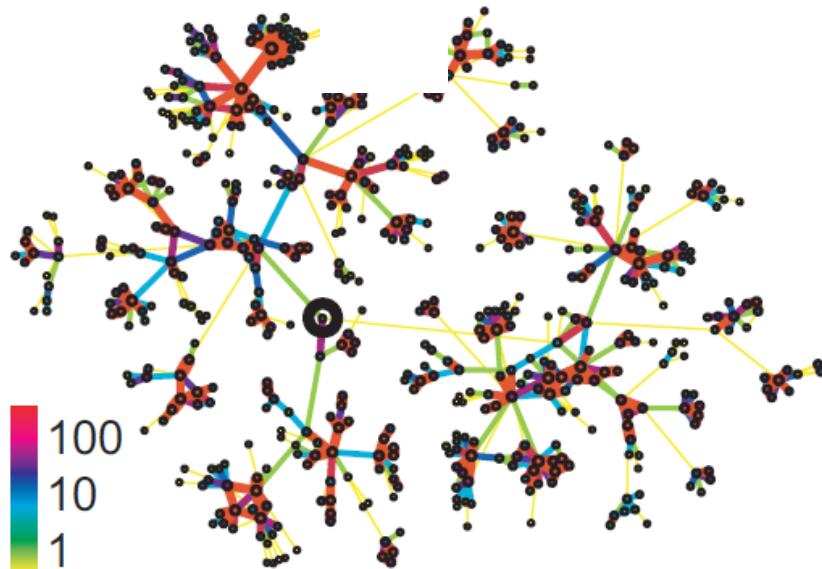
## How to find communities?



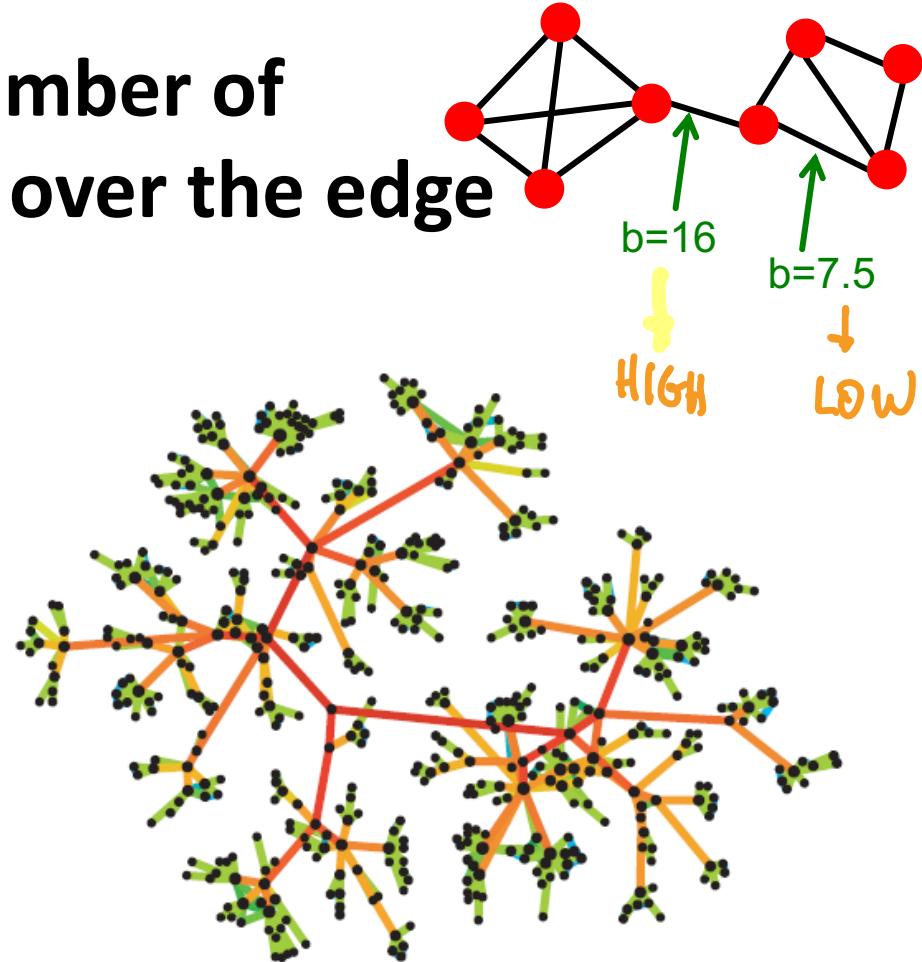
We will work with **undirected** (unweighted) networks

# 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

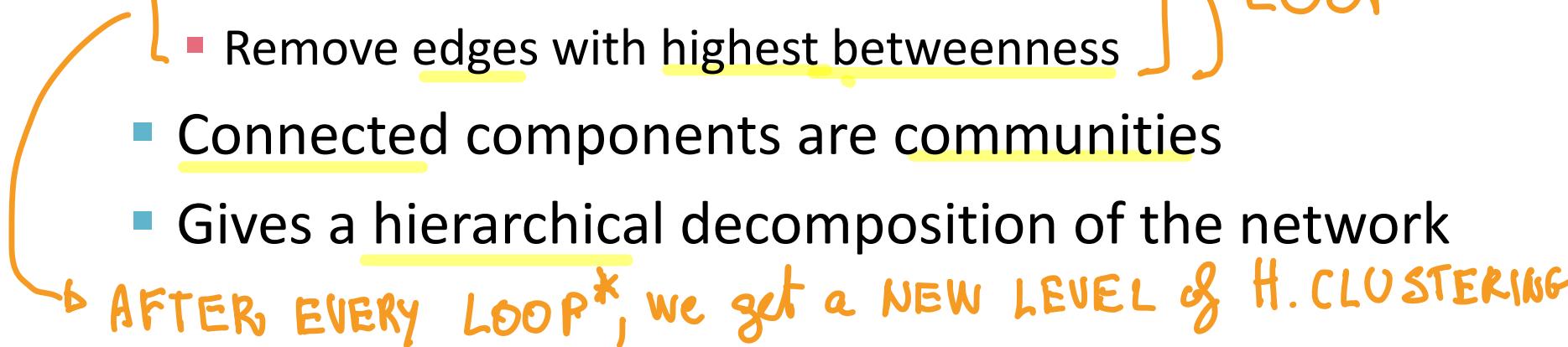


Edge betweenness  
in a real network

# Method 1: Girvan-Newman

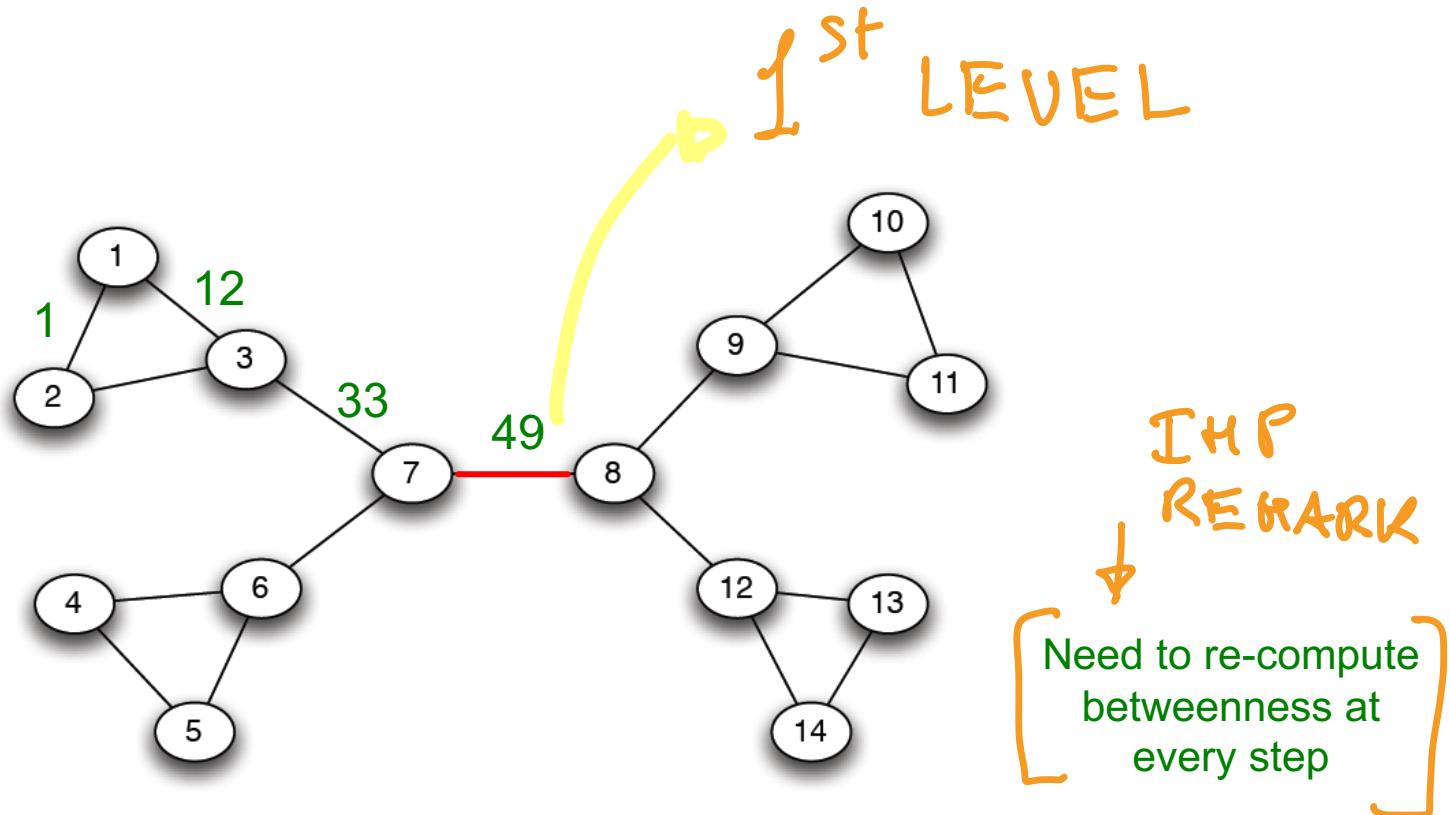
- Divisive hierarchical clustering based on the notion of edge **betweenness**:  
Number of shortest paths passing through the edge
- **Girvan-Newman Algorithm:**
  - Undirected unweighted networks
  - **Repeat until no edges are left:**
    - Calculate betweenness of edges
    - Remove edges with highest betweenness
  - Connected components are communities
  - Gives a hierarchical decomposition of the network

AFTER EVERY LOOP\*, we get a NEW LEVEL of H. CLUSTERING



} GN's  
LOOP\*

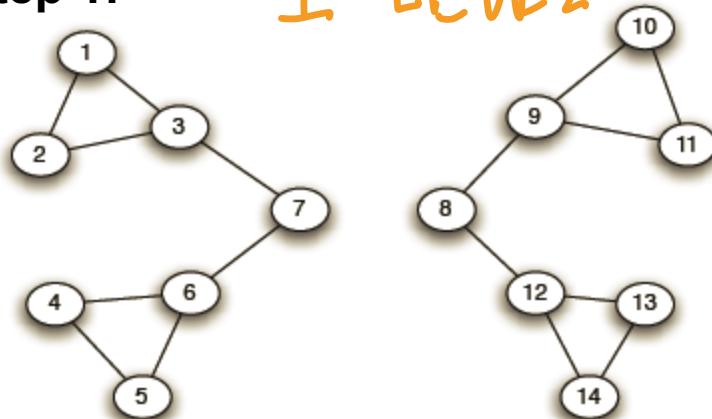
# Girvan-Newman: Example



# Girvan-Newman: Example

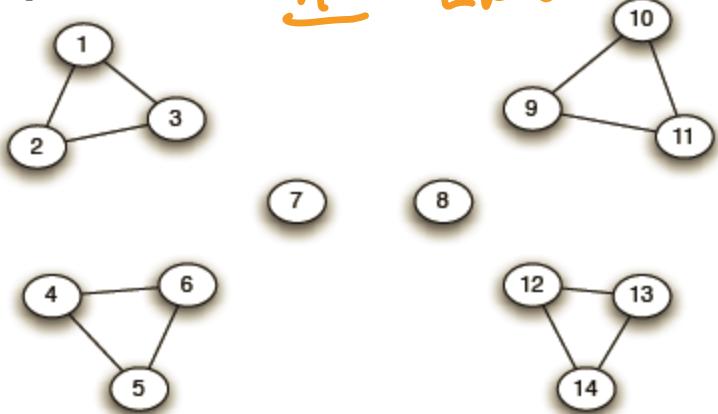
Step 1:

I LEVEL



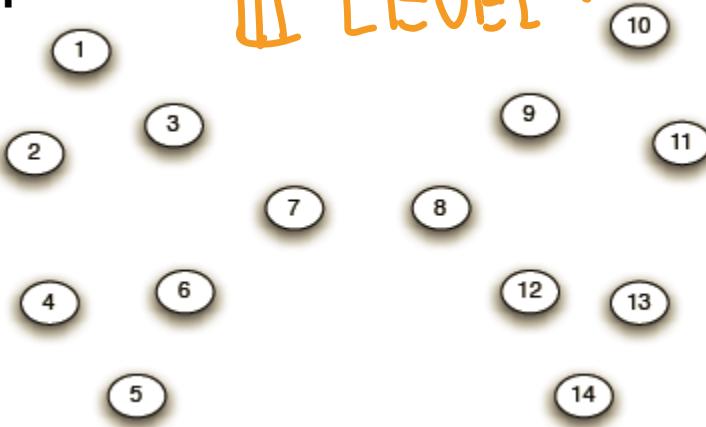
Step 2:

II LEVEL

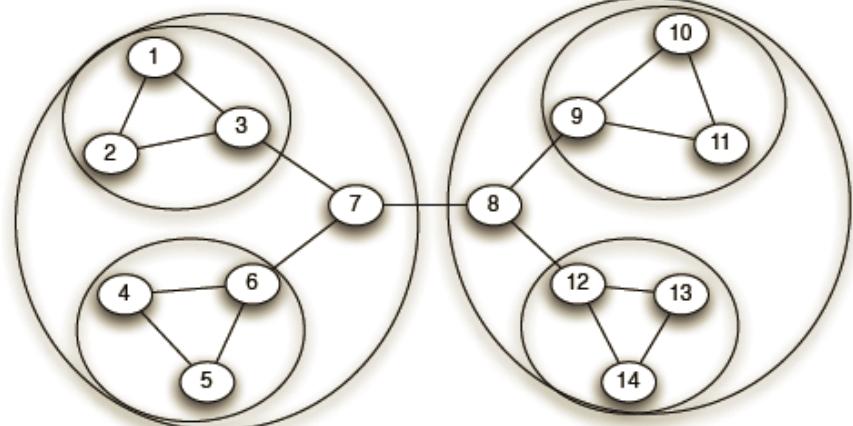


Step 3:

III LEVEL ??



Hierarchical network decomposition:



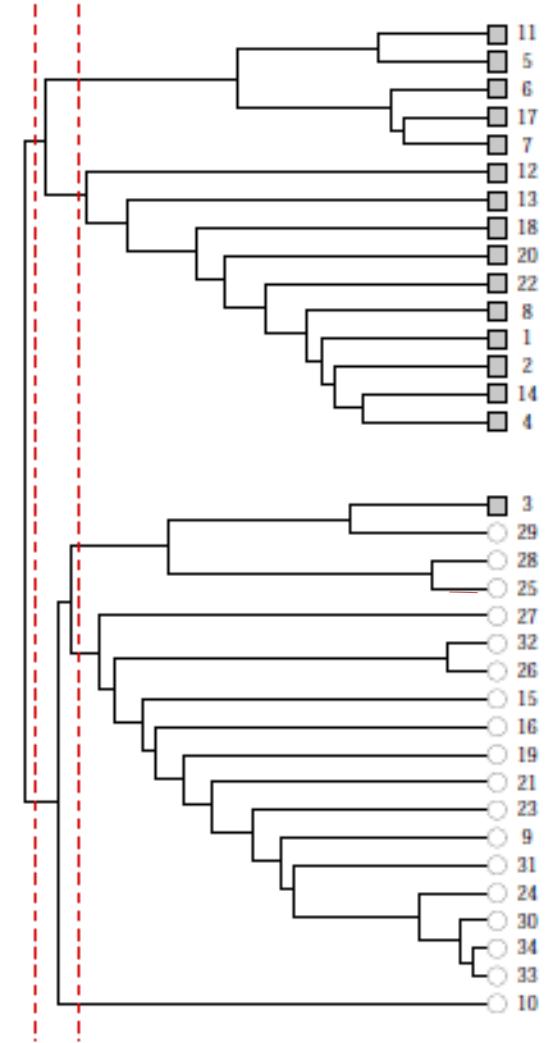
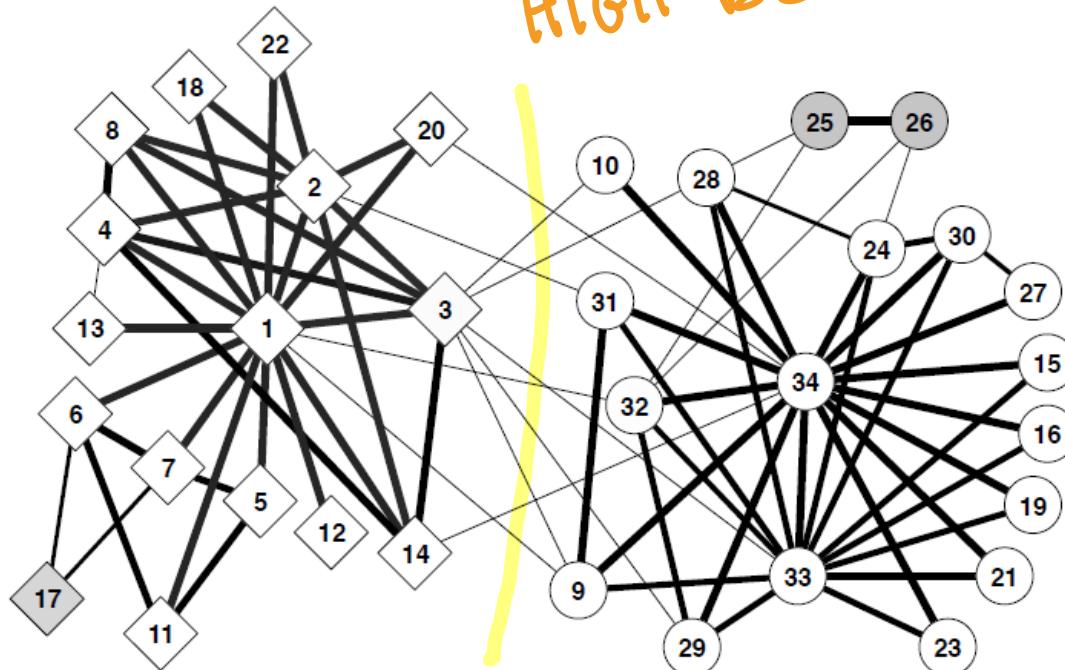
# Girvan-Newman: Results



Communities in physics collaborations

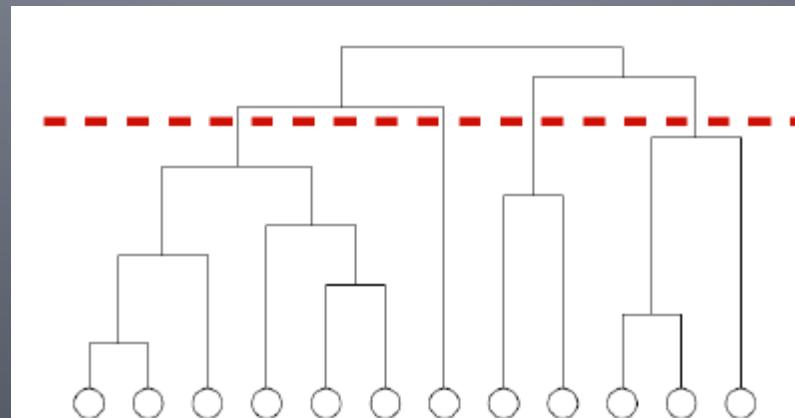
# Girvan-Newman: Results

- 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?



# GN' ALGO ( $G(V, E)$ ) for BETWEENNESS of every edge $e \in E$

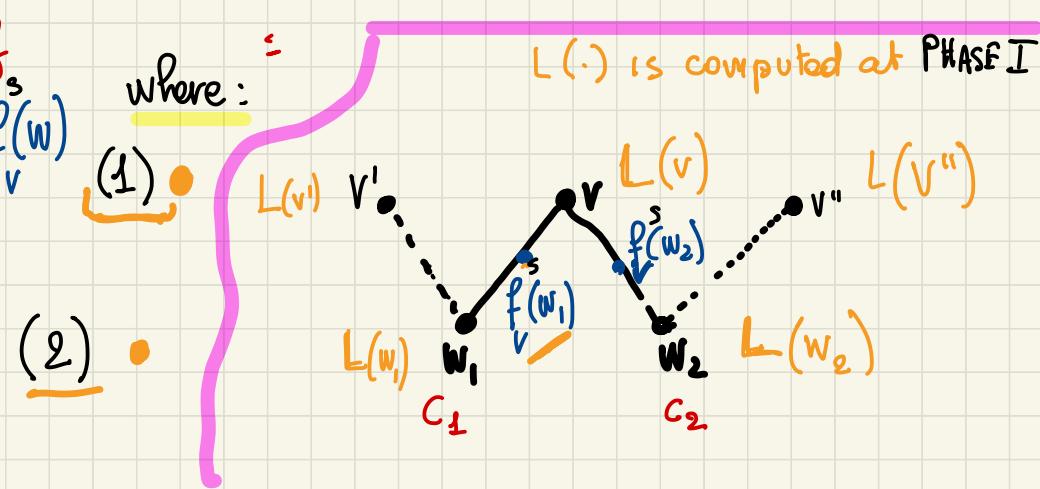
- PHASE I:
- (i)  $\forall s \in V$ , compute  $[BFS(G, s) \equiv \{L_1, L_2, \dots, L_d\}]$
  - (ii)  $\forall v \neq s$ : Compute (Top-down on  $BFS(G, s)$ ):  $\left\{ \begin{array}{l} \text{it is easy from } BFS(G, s) \\ \text{LABEL } L(v) \equiv \# \text{spaths}(s-v) \end{array} \right.$

PHASE II: By a Bottom-up procedure (start from each LEAF of  $BFS(G, s)$ )  
 assign CREDITS  $c(v)$  to each  $v \neq s$ , where inductively:

$$\left\{ \begin{array}{l} c^s(v) = 1 \quad \text{if } v \text{ is a leaf} \\ c^s(v) = 1 + \sum_{w \in \text{SON}(v)} c^s(w) \cdot f_v^s(w) \end{array} \right.$$

$$\text{SON}(v) = \{\text{sons of } v\}$$

$$f_v^s(w) = \frac{L(v)}{\sum_{v' \in \text{PARENTS}(w)} L(v')}$$

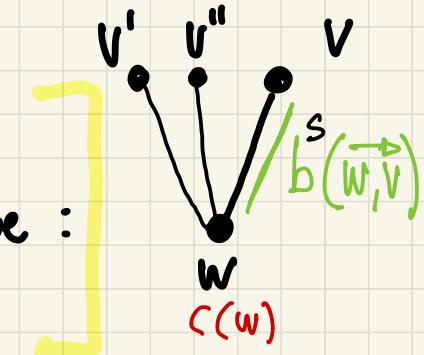


• PHASE III : From Eqs (1)+(2), assign to every edge  $(\vec{w}, v)$ , the value  $f_v^s(w)$  computed in PHASE 2

• At the end of PHASE III, we get :

(i)  $\forall$  source  $s \in V$ ,  $\forall$  edge  $e = (\vec{w}, v)$ , the value :

$$b^s(\vec{w}, v) := f_v^s(w) \cdot c^s(v)$$

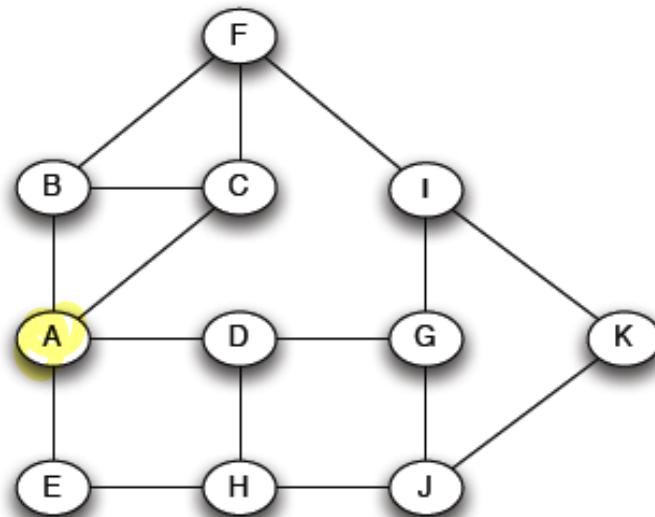


(ii) The Betweenness of edge  $(w, v)$  is  $B(w, v) \equiv \sum_{\text{source } s \in V} b^s(\vec{w}, v)$

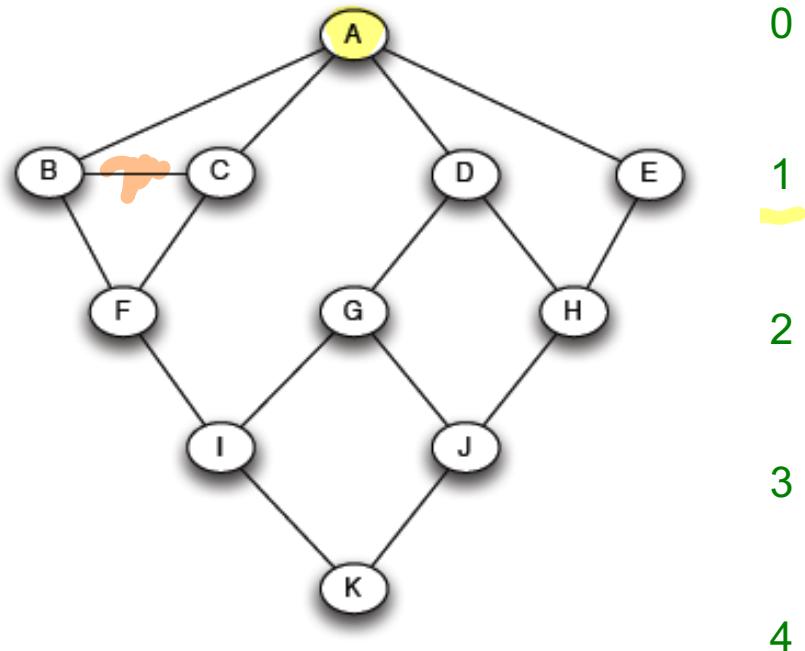
(\*) NOTE : If an edge  $e$  never appear as a BFS-edge for some source  $s$ , Then its betweenness( $e$ ) is  $0$  !

# How to Compute Betweenness?

- Want to compute betweenness of paths starting at node A

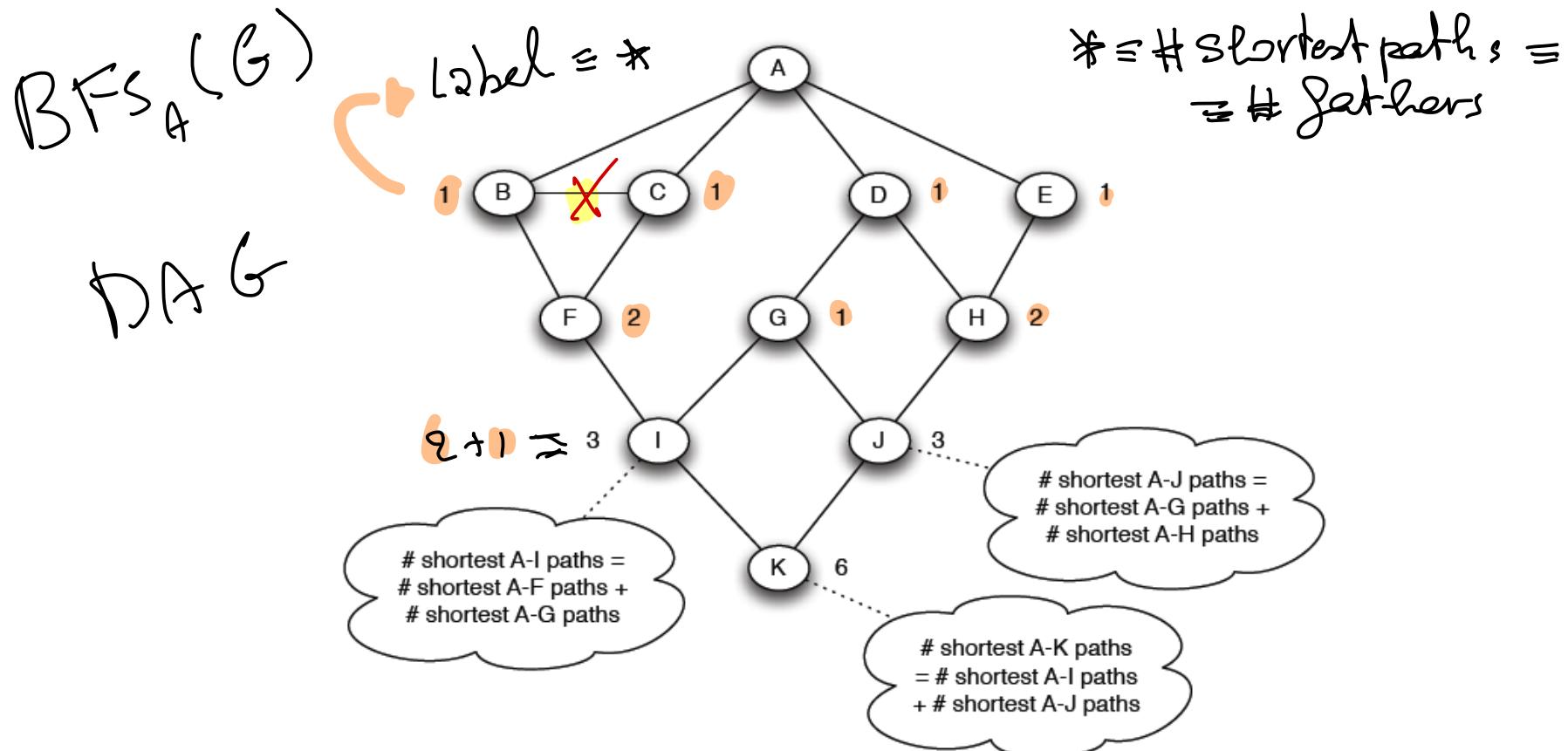


- Breath first search starting from A:



# 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

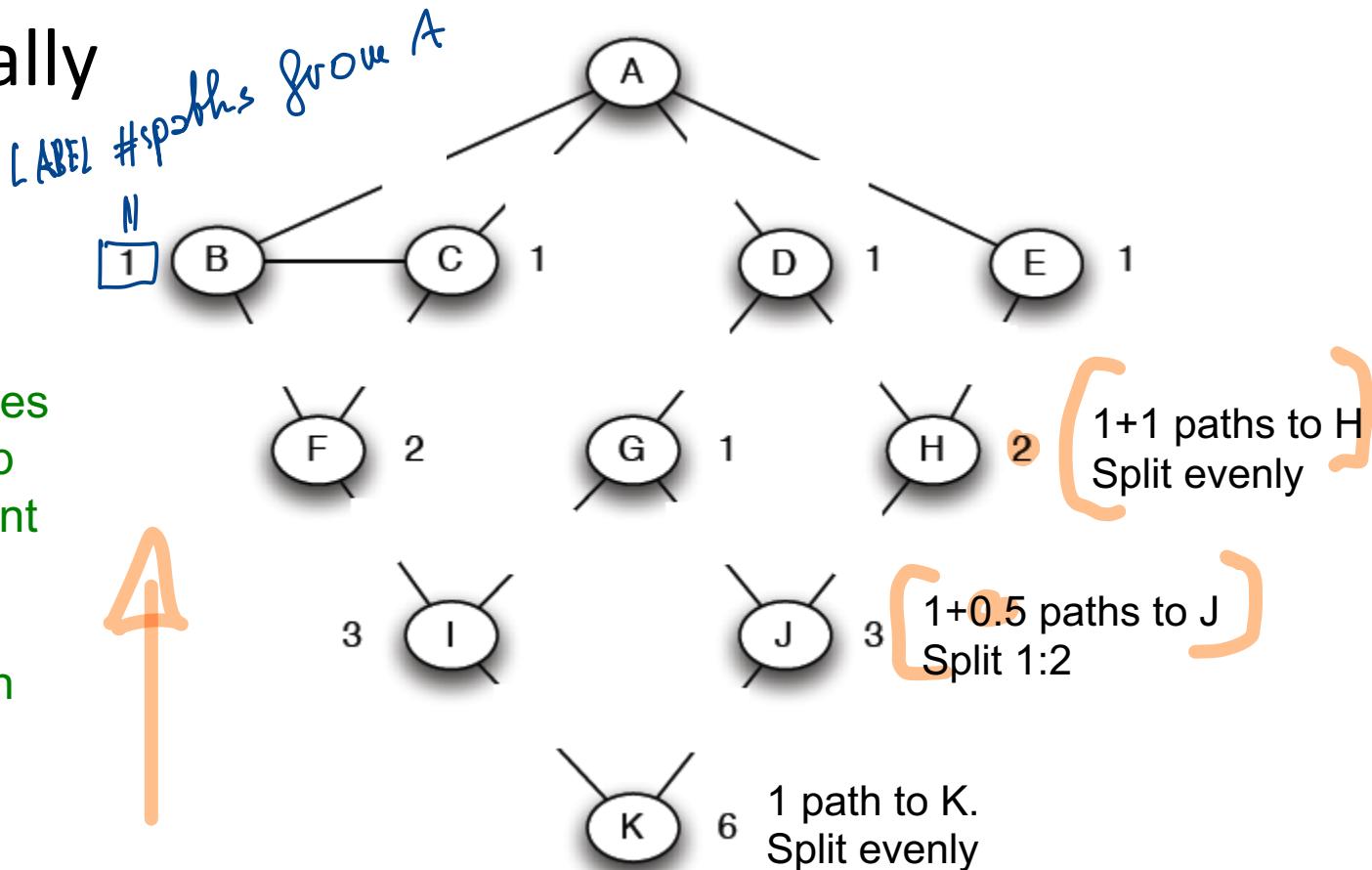
The algorithm:

- Add edge flows:

- node flow =  $1 + \sum_{\text{child edges}}$

- split the flow up based on the parent value

- Repeat the BFS procedure for each starting node  $U$



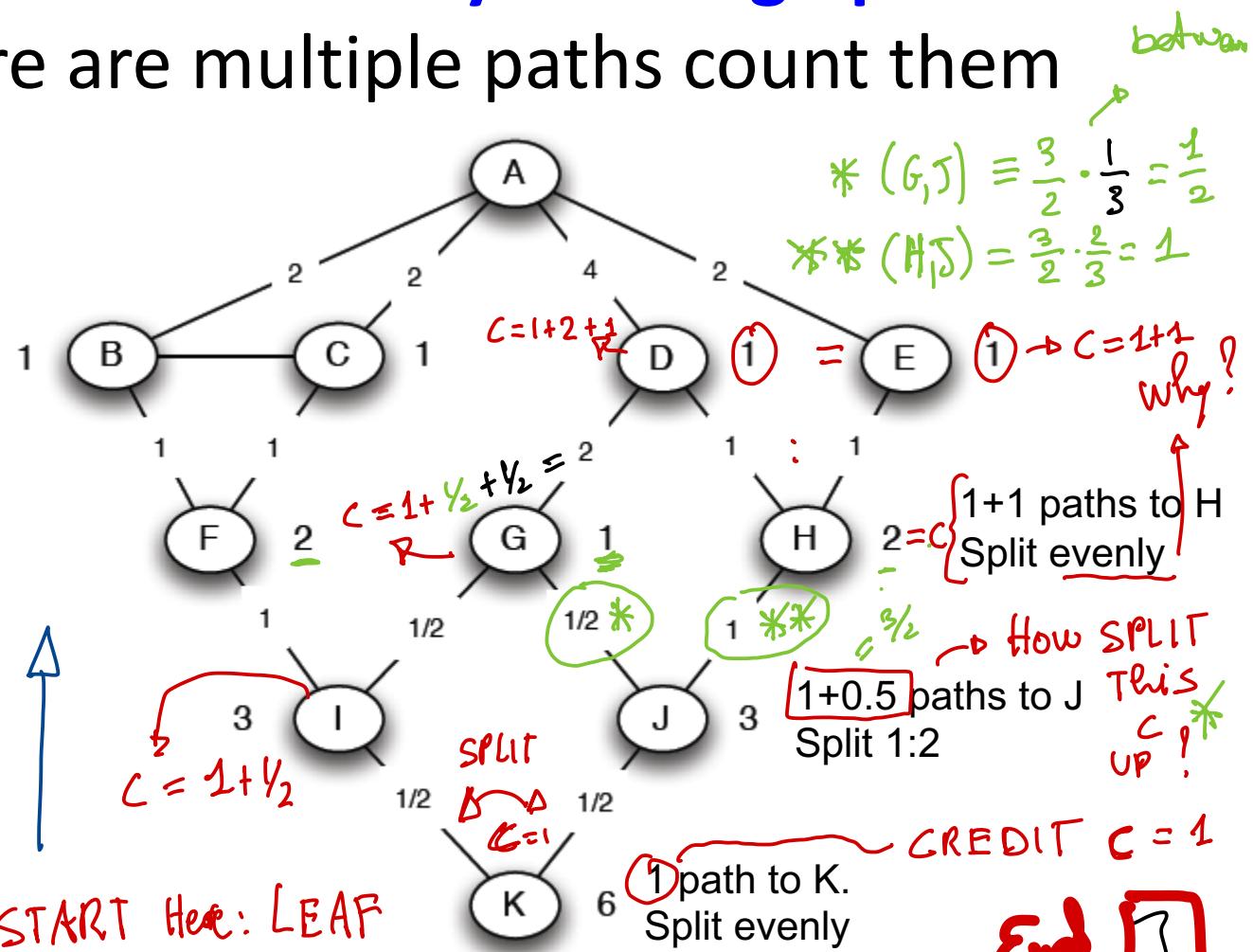
L'ALGO di GN per la BETWEENNESS  
va fatta sul libro CHPT 10

# How to Compute Betweenness?

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

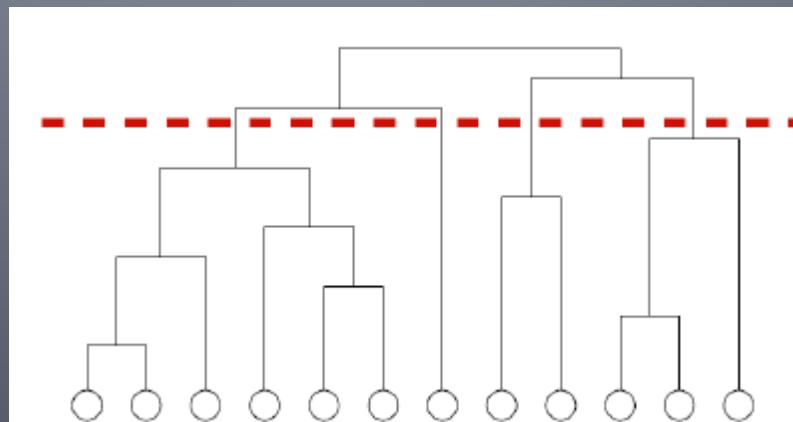
## The algorithm:

- Add edge flows:
    - node flow =  $flow_{\text{in}} \approx 1 + \sum_{\text{child edges}}$
    - split the flow up based on the parent value
  - Repeat the BFS procedure for each starting node  $U$



# We need to resolve 2 questions

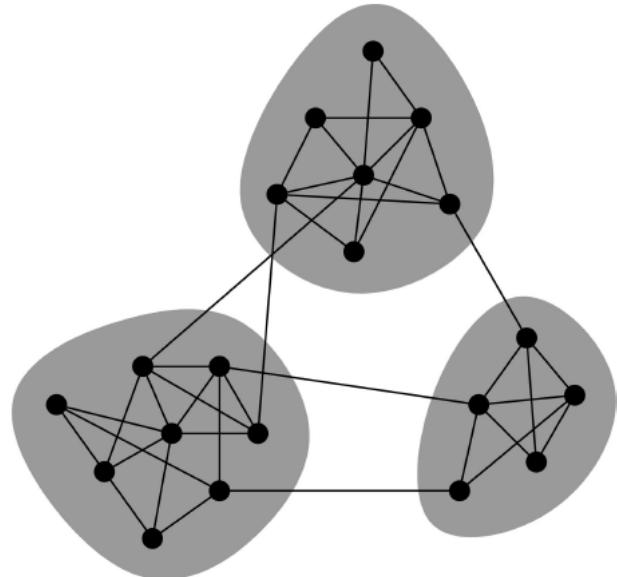
1. How to compute betweenness?
2. How to select the number of clusters?



# 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  $s \in S$ :

$$Q \propto \sum_{s \in S} [ (\# \text{ edges within group } s) - \underbrace{(\text{expected } \# \text{ edges within group } s)}_{\text{Need a null model!}} ]$$



# Null Model: Configuration Model

- Given real  $G$  on  $n$  nodes and  $m$  edges, construct rewired network  $G'$

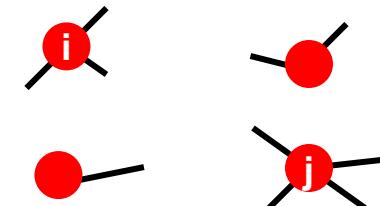
- Same degree distribution but random connections
- Consider  $G'$  as a multigraph
- The expected number of edges between nodes  $i$  and  $j$  of degrees  $k_i$  and  $k_j$  equals to:

$$k_i \cdot \frac{k_j}{2m} = \frac{k_i k_j}{2m}$$

- The expected number of edges in (multigraph)  $\mathbf{G}'$ :

- $= \frac{1}{2} \sum_{i \in N} \sum_{j \in N} \frac{k_i k_j}{2m} = \frac{1}{2} \cdot \frac{1}{2m} \sum_{i \in N} k_i (\sum_{j \in N} k_j) =$
- $= \frac{1}{4m} 2m \cdot 2m = m$

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



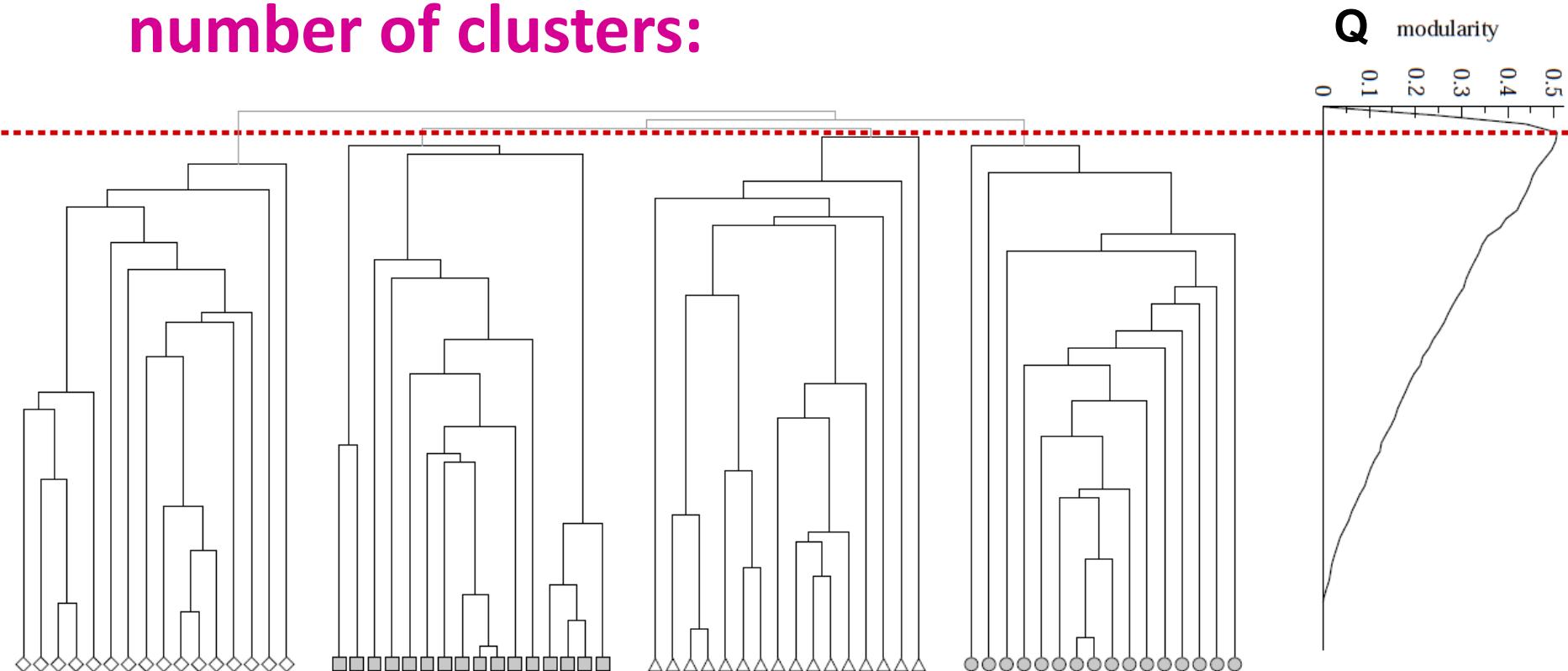
# Modularity

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

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

# Modularity: Number of clusters

- Modularity is useful for selecting the number of clusters:

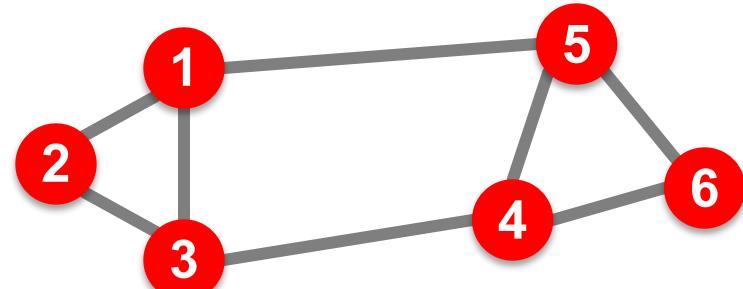


Next time: Why not optimize Modularity directly?

# Spectral Clustering

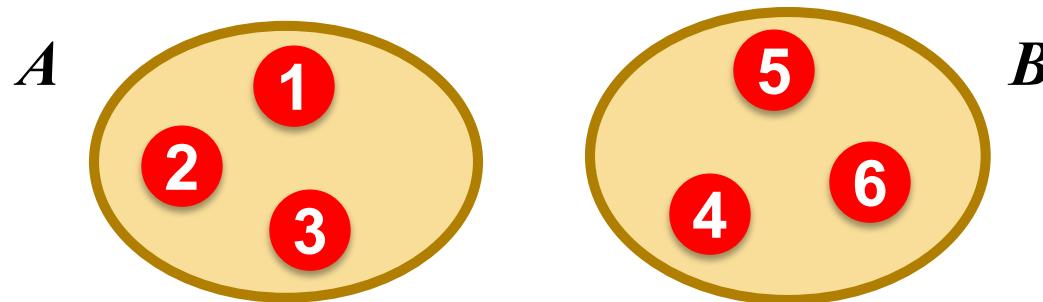
# Graph Partitioning

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



- Bi-partitioning task:

- Divide vertices into two disjoint groups  $A, B$

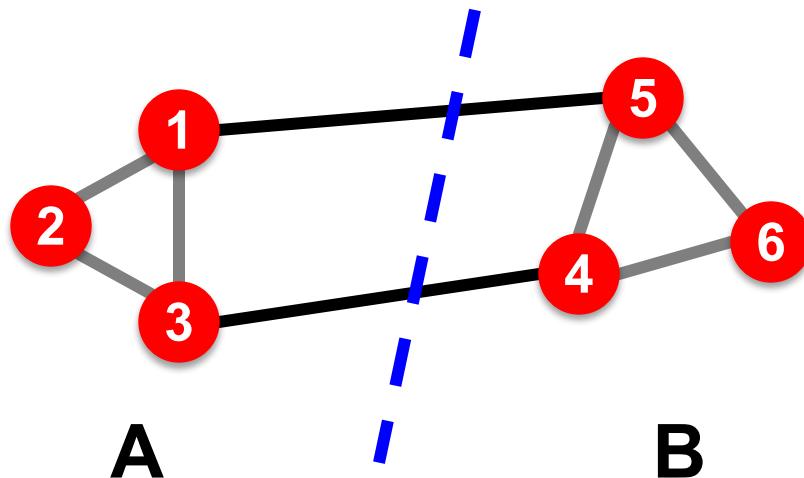


- Questions:

- How can we define a “good” partition of  $G$ ?
  - How can we efficiently identify such a partition?

# Graph Partitioning

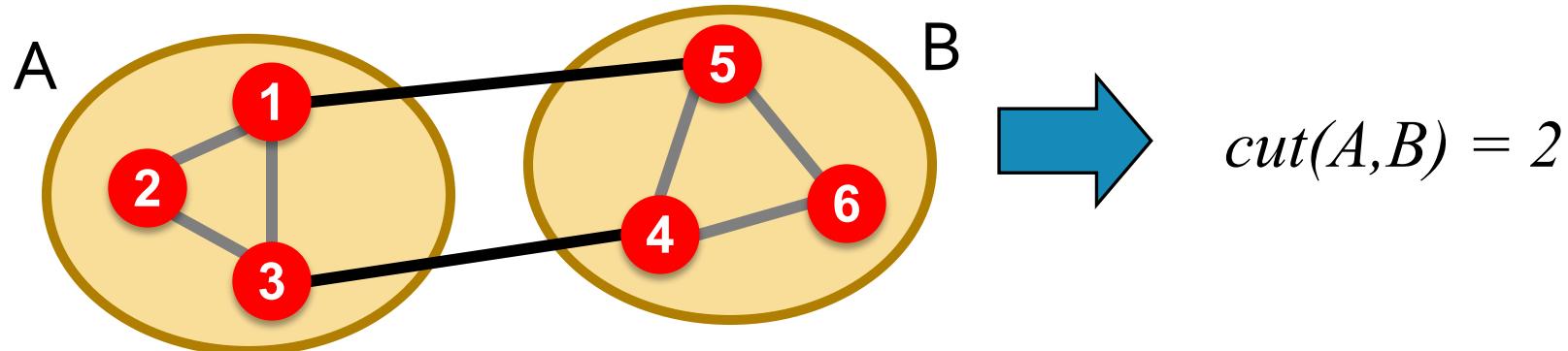
- What makes a good partition?
  - Maximize the number of within-group connections
  - Minimize the number of between-group connections



# 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:

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

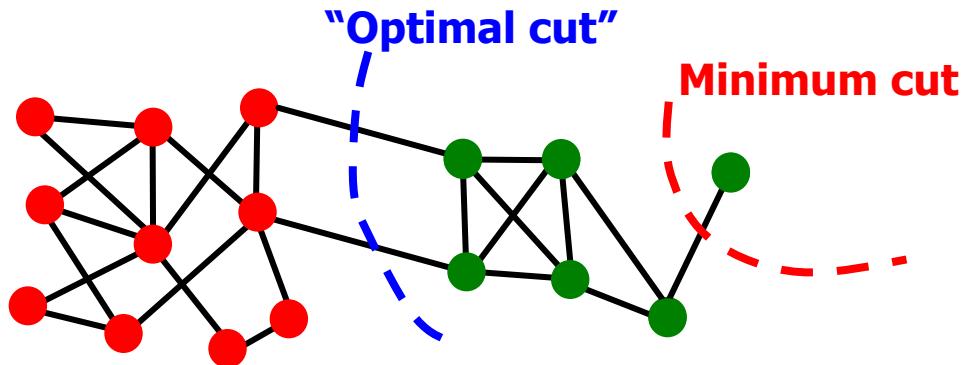


# Graph Cut Criterion

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

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

- 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

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

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

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

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

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

- **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  $j$

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

- **Spectral Graph Theory:**

- Analyze the “spectrum” of matrix representing  $G$
- **Spectrum:** Eigenvectors  $\mathbf{x}_i$  of a graph, ordered by the magnitude (strength) of their corresponding eigenvalues  $\lambda_i$ :  $\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$

$$\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$$

# Example: d-regular graph

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

$A \cdot x = \lambda \cdot x$  What is  $\lambda$ ? What  $x$ ?

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

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

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

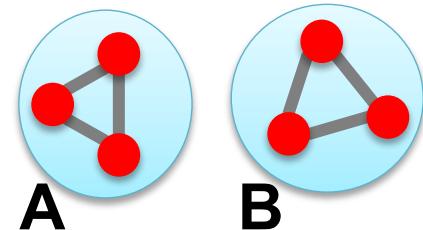
# $d$ is the largest eigenvalue of $A$

- $G$  is  $d$ -regular connected,  $A$  is its adjacency matrix
- **Claim:**
  - $d$  is largest eigenvalue of  $A$ ,
  - $d$  has multiplicity of **1** (there is only **1** eigenvector associated with eigenvalue  $d$ )
- **Proof: Why no eigenvalue  $d' > d$ ?**
  - To obtain  $d$  we needed  $x_i = x_j$  for every  $i, j$
  - This means  $x = c \cdot (1, 1, \dots, 1)$  for some const.  $c$
  - **Define:**  $S$  = nodes  $i$  with maximum possible value of  $x_i$
  - Then consider some vector  $y$  which is not a multiple of vector  $(1, \dots, 1)$ . So not all nodes  $i$  (with labels  $y_i$ ) are in  $S$
  - Consider some node  $j \in S$  and a neighbor  $i \notin S$  then node  $j$  gets a value strictly less than  $d$
  - **So  $y$  is not eigenvector! And so  $d$  is the largest eigenvalue!**

# Example: Graph on 2 components

## ■ What if $G$ is not connected?

- $G$  has 2 components, each  $d$ -regular



## ■ What are some eigenvectors?

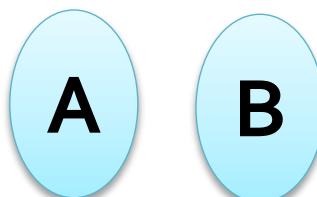
- $x = \text{Put all } 1\text{s on } A \text{ and } 0\text{s on } B \text{ or vice versa}$

- $x' = (\underbrace{1, \dots, 1}_{|A|}, \underbrace{0, \dots, 0}_{|B|})$  then  $A \cdot x' = (d, \dots, d, 0, \dots, 0)$

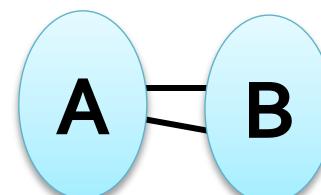
- $x'' = (0, \dots, 0, \underbrace{1, \dots, 1}_{|B|})$  then  $A \cdot x'' = (0, \dots, 0, d, \dots, d)$

- And so in both cases the corresponding  $\lambda = d$

## ■ A bit of intuition:



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

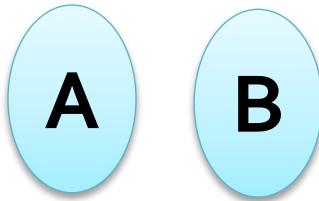


$$\lambda_n - \lambda_{n-1} \approx 0$$

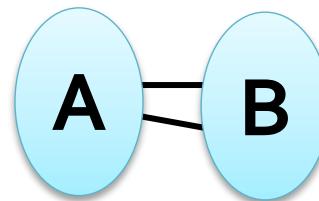
2<sup>nd</sup> largest eigval.  
 $\lambda_{n-1}$  now has  
value very close  
to  $\lambda_n$

# More Intuition

## More intuition:



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



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

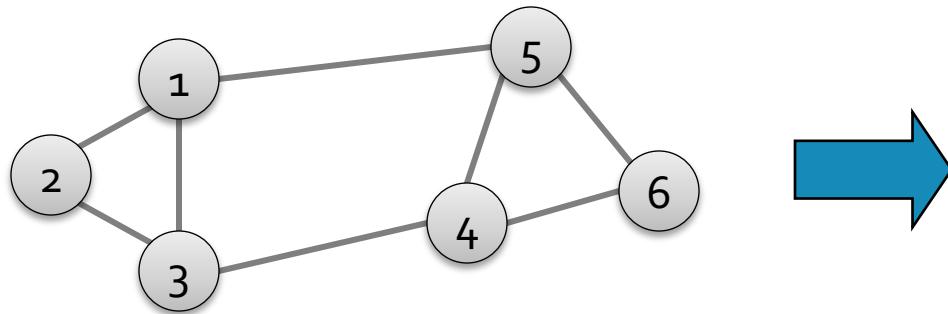
2<sup>nd</sup> largest eigval.  
 $\lambda_{n-1}$  now has  
value very close  
to  $\lambda_n$

- If the graph is connected (right example) then we already know that  $x_n = (1, \dots, 1)$  is an eigenvector
- Since eigenvectors are orthogonal then the components of  $x_{n-1}$  sum to **0**.
  - Why? Because  $x_n \cdot x_{n-1} = \sum_i x_n[i] \cdot x_{n-1}[i]$
- So we can look at the eigenvector of the 2<sup>nd</sup> largest eigenvalue and declare nodes with positive label in **A** and negative label in **B**.
- **But there is still lots to sort out.**

# Matrix Representations

## ■ Adjacency matrix ( $A$ ):

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



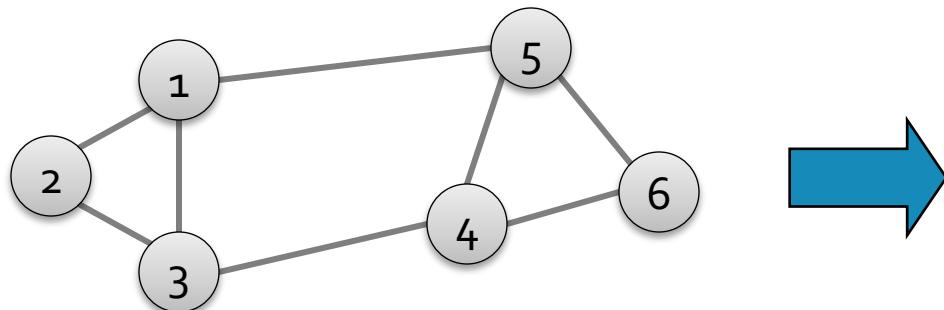
	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0

## ■ Important properties:

- Symmetric matrix
- Eigenvectors are real and orthogonal

# Matrix Representations

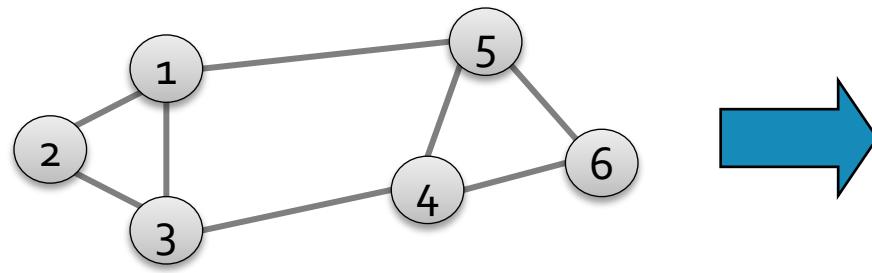
- **Degree matrix (D):**
  - $n \times n$  diagonal matrix
  - $D = [d_{ii}]$ ,  $d_{ii}$  = degree of node  $i$



	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

# Matrix Representations

- Laplacian matrix ( $L$ ):
  - $n \times n$  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

- What is trivial eigenpair?

$$L = D - A$$

- $x = (1, \dots, 1)$  then  $L \cdot x = 0$  and so  $\lambda = \lambda_1 = 0$
- Important properties:
  - Eigenvalues are non-negative real numbers
  - Eigenvectors are real and orthogonal

# Facts about the Laplacian L

- (a) All eigenvalues are  $\geq 0$
- (b)  $x^T L x = \sum_{ij} L_{ij} x_i x_j \geq 0$  for every  $x$
- (c)  $L = N^T \cdot N$

- That is,  $L$  is positive semi-definite
- **Proof:**
  - (c) $\Rightarrow$ (b):  $x^T L x = x^T N^T N x = (xN)^T (Nx) \geq 0$ 
    - As it is just the square of length of  $Nx$
  - (b) $\Rightarrow$ (a): Let  $\lambda$  be an eigenvalue of  $L$ . Then by (b)  $x^T L x \geq 0$  so  $x^T L x = x^T \lambda x = \lambda x^T x \Rightarrow \lambda \geq 0$
  - (a) $\Rightarrow$ (c): is also easy! Do it yourself.

# $\lambda_2$ as optimization problem

- Fact: For symmetric matrix  $M$ :

$$\lambda_2 = \min_x \frac{x^T M x}{x^T x}$$

- What is the meaning of  $\min x^T L x$  on  $G$ ?

- $x^T L x = \sum_{i,j=1}^n L_{ij} x_i x_j = \sum_{i,j=1}^n (D_{ij} - A_{ij}) x_i x_j$
- $= \sum_i D_{ii} x_i^2 - \sum_{(i,j) \in E} 2x_i x_j$
- $= \sum_{(i,j) \in E} (\underbrace{x_i^2 + x_j^2}_{\text{Node } i \text{ has degree } d_i} - 2x_i x_j) = \sum_{(i,j) \in E} (x_i - x_j)^2$

Node  $i$  has degree  $d_i$ . So, value  $x_i^2$  needs to be summed up  $d_i$  times.  
But each edge  $(i,j)$  has two endpoints so we need  $x_i^2 + x_j^2$

# Proof:

$$\lambda_2 = \min_x \frac{x^T M x}{x^T x}$$

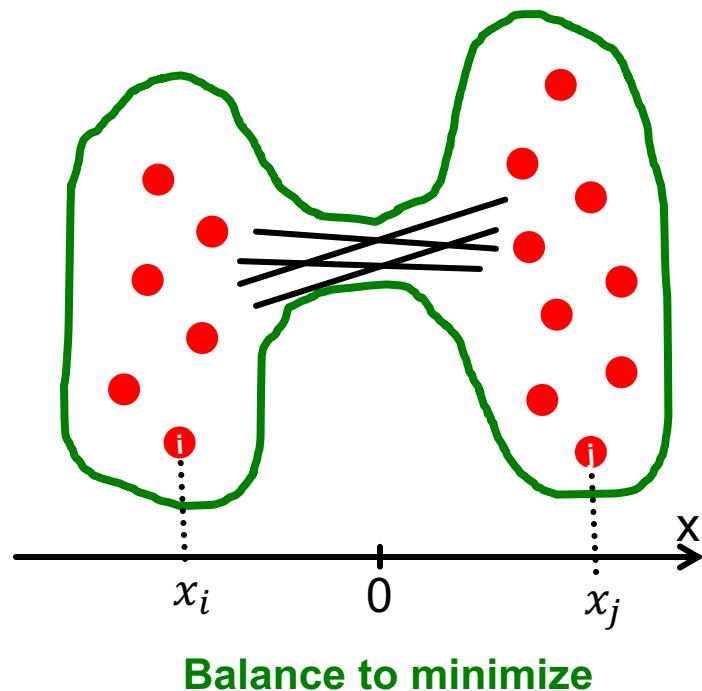
- Write  $x$  in axes of eigenvectors  $w_1, w_2, \dots, w_n$  of  $M$ . So,  $x = \sum_i^n \alpha_i w_i$
- Then we get:  $Mx = \sum_i \alpha_i \underbrace{Mw_i}_{\lambda_i w_i} = \sum_i \alpha_i \lambda_i w_i$
- So, what is  $x^T M x$ ?  $= 0 \text{ if } i \neq j$   
 $1 \text{ otherwise}$
- $x^T M x = (\sum_i \alpha_i w_i)(\sum_i \alpha_i \lambda_i w_i) = \sum_{ij} \alpha_i \lambda_j \alpha_j \underbrace{w_i w_j}_{= 0 \text{ if } i \neq j \text{ or } 1 \text{ if } i = j}$
- $= \sum_i \alpha_i \lambda_i w_i w_i = \sum_i \lambda_i \alpha_i^2$
- To minimize this over all unit vectors  $x$  orthogonal to:  $w = \min$  over choices of  $(\alpha_1, \dots, \alpha_n)$  so that:  
 $\sum \alpha_i^2 = 1$  (unit length)  
 $\sum \alpha_i = 0$  (orthogonal to  $w_1$ )
- To minimize this, set  $\alpha_2 = 1$  and so  $\sum_i \lambda_i \alpha_i^2 = \lambda_2$

# $\lambda_2$ as optimization problem

- What else do we know about  $x$ ?
  - $x$  is unit vector:  $\sum_i x_i^2 = 1$
  - $x$  is orthogonal to 1<sup>st</sup> eigenvector ( $1, \dots, 1$ ) thus:  
 $\sum_i x_i \cdot 1 = \sum_i x_i = 0$
- Remember:

$$\lambda_2 = \min_{\substack{\text{All labelings} \\ \text{of nodes } i \text{ so} \\ \text{that } \sum x_i = 0}} \frac{\sum_{(i,j) \in E} (x_i - x_j)^2}{\sum_i x_i^2}$$

We want to assign values  $x_i$  to nodes  $i$  such  
that few edges cross 0.  
(we want  $x_i$  and  $x_j$  to subtract each other)



# Find Optimal Cut [Fiedler'73]

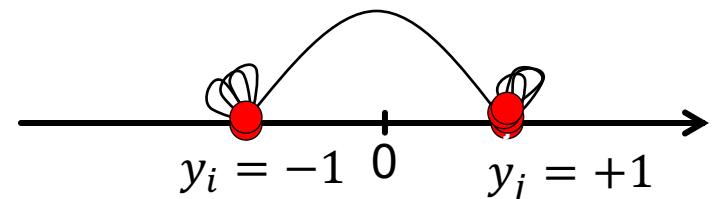
- Back to finding the optimal cut
- Express partition (A,B) as a vector

$$y_i = \begin{cases} +1 & \text{if } i \in A \\ -1 & \text{if } i \in B \end{cases}$$

- We can minimize the cut of the partition by finding a non-trivial vector  $x$  that **minimizes**:

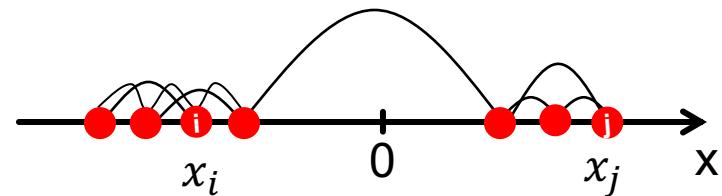
$$\arg \min_{y \in [-1, +1]^n} f(y) = \sum_{(i, j) \in E} (y_i - y_j)^2$$

Can't solve exactly. Let's relax  $y$  and allow it to take any real value.



# Rayleigh Theorem

$$\min_{y \in \Re^n} f(y) = \sum_{(i,j) \in E} (y_i - y_j)^2 = y^T L y$$



- $\lambda_2 = \min_y f(y)$ : The minimum value of  $f(y)$  is given by the 2<sup>nd</sup> smallest eigenvalue  $\lambda_2$  of the Laplacian matrix  $L$
- $x = \arg \min_y f(y)$ : The optimal solution for  $y$  is given by the corresponding eigenvector  $x$ , referred as the **Fiedler vector**

# Approx. Guarantee of Spectral

- Suppose there is a partition of  $\mathbf{G}$  into  $\mathbf{A}$  and  $\mathbf{B}$  where  $|A| \leq |B|$ , s.t.  $\alpha = \frac{(\# \text{ edges from } A \text{ to } B)}{|A|}$   
then  $2\alpha \geq \lambda_2$ 
  - This is the approximation guarantee of the spectral clustering. It says the cut spectral finds is at most 2 away from the optimal one of score  $\alpha$ .
- Proof:**

- Let:  $a=|A|$ ,  $b=|B|$  and  $e= \# \text{ edges from } \mathbf{A} \text{ to } \mathbf{B}$
- Enough to choose some  $x_i$  based on  $\mathbf{A}$  and  $\mathbf{B}$  such that:  $\lambda_2 \leq \underbrace{\frac{\sum(x_i - x_j)^2}{\sum_i x_i^2}}_{\lambda_2 \text{ is only smaller}} \leq 2\alpha$  (while also  $\sum_i x_i = 0$ )

$\lambda_2$  is only smaller

# Approx. Guarantee of Spectral

## ■ Proof (continued):

- 1) Let's set:  $x_i = \begin{cases} -\frac{1}{a} & \text{if } i \in A \\ +\frac{1}{b} & \text{if } i \in B \end{cases}$
- Let's quickly verify that  $\sum_i x_i = 0$ :  $a\left(-\frac{1}{a}\right) + b\left(\frac{1}{b}\right) = 0$

$$\begin{aligned} \text{■ 2) Then: } \frac{\sum(x_i - x_j)^2}{\sum_i x_i^2} &= \frac{\sum_{i \in A, j \in B} \left(\frac{1}{b} + \frac{1}{a}\right)^2}{a\left(-\frac{1}{a}\right)^2 + b\left(\frac{1}{b}\right)^2} = \frac{e \cdot \left(\frac{1}{a} + \frac{1}{b}\right)^2}{\frac{1}{a} + \frac{1}{b}} = \\ &e \left(\frac{1}{a} + \frac{1}{b}\right) \leq e \left(\frac{1}{a} + \frac{1}{a}\right) \leq e \frac{2}{a} = 2\alpha \end{aligned}$$

e ... number of edges between A and B

Which proves that the cost achieved by spectral is better than twice the OPT cost

# Approx. Guarantee of Spectral

## ■ Putting it all together:

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

- where  $k_{max}$  is the maximum node degree in the graph
  - Note we only provide the 1<sup>st</sup> part:  $2\alpha \geq \lambda_2$
  - We did not prove  $\lambda_2 \geq \frac{\alpha^2}{2k_{max}}$
- Overall this always certifies that  $\lambda_2$  always gives a useful bound

# 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:

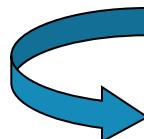
- Build Laplacian matrix  $L$  of the graph



	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  $\lambda$  and eigenvectors  $x$  of the matrix  $L$



$$\lambda =$$

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

- Map vertices to corresponding components of  $\lambda_2$

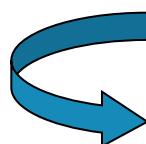
1	0.3
2	0.6
3	0.3
4	-0.3
5	-0.3
6	-0.6



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

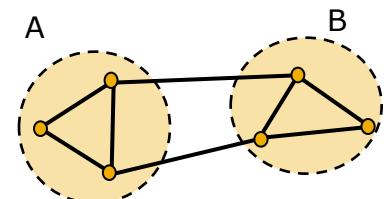


**Split at 0:**

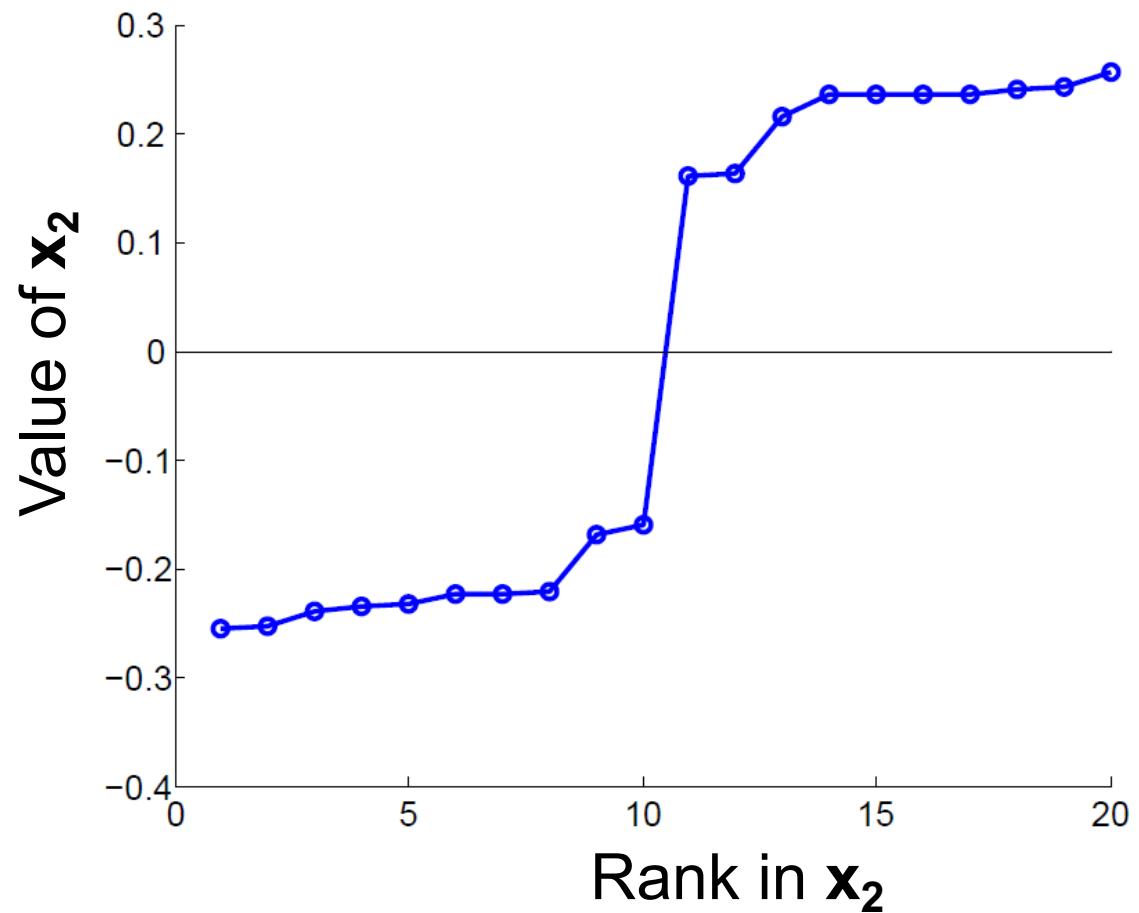
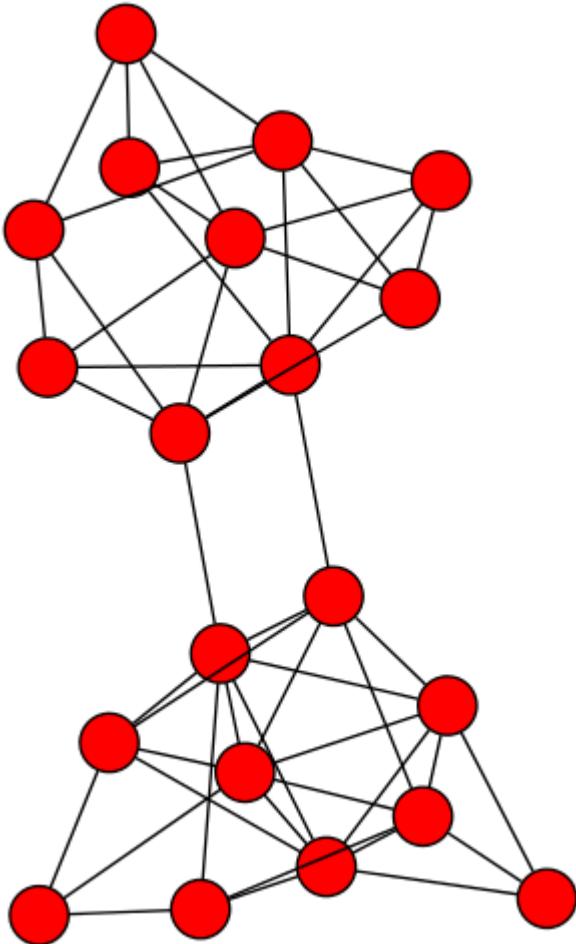
**Cluster A: Positive points**  
**Cluster B: Negative points**

1	0.3
2	0.6
3	0.3

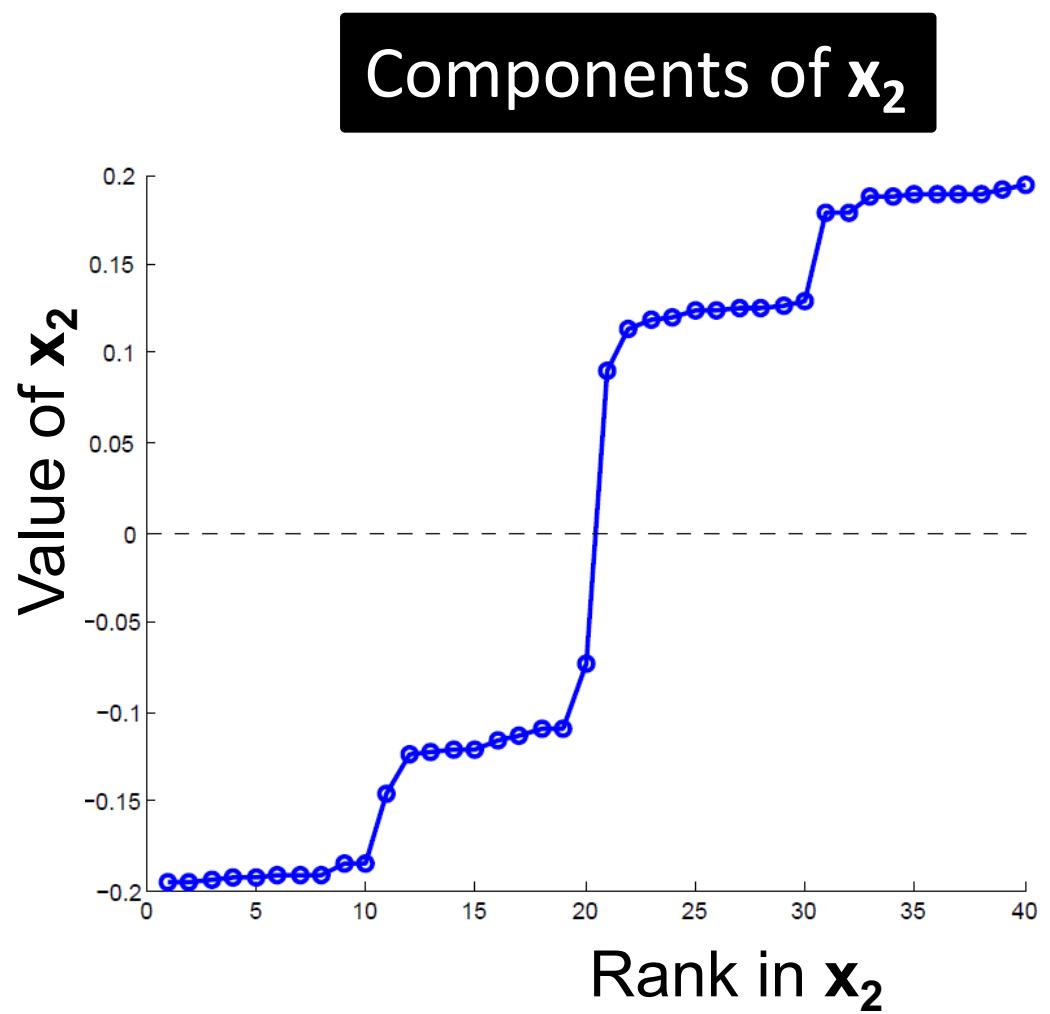
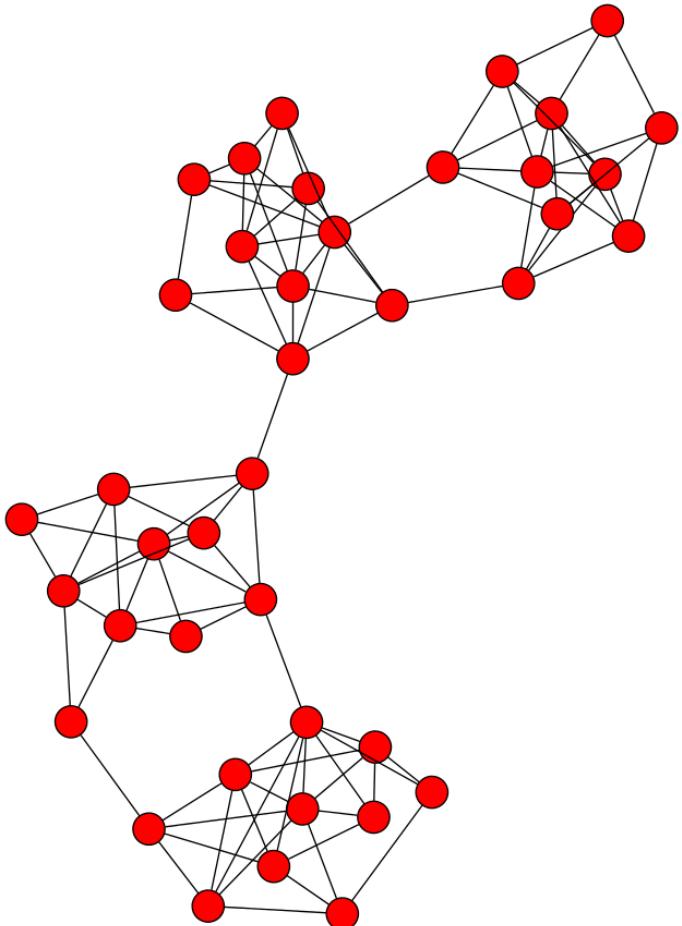
4	-0.3
5	-0.3
6	-0.6



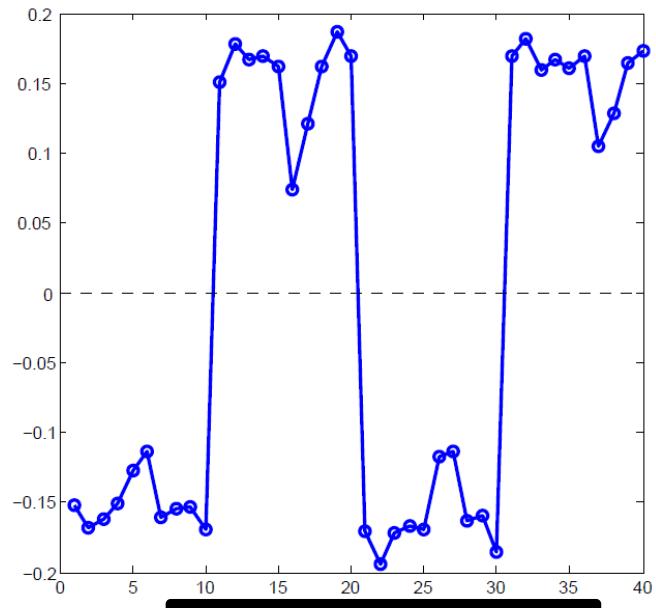
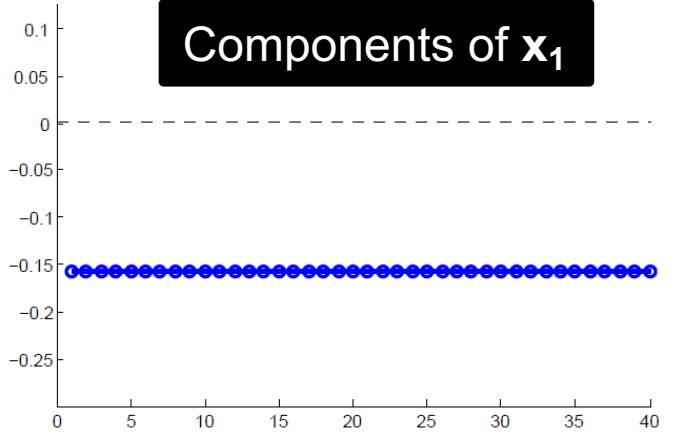
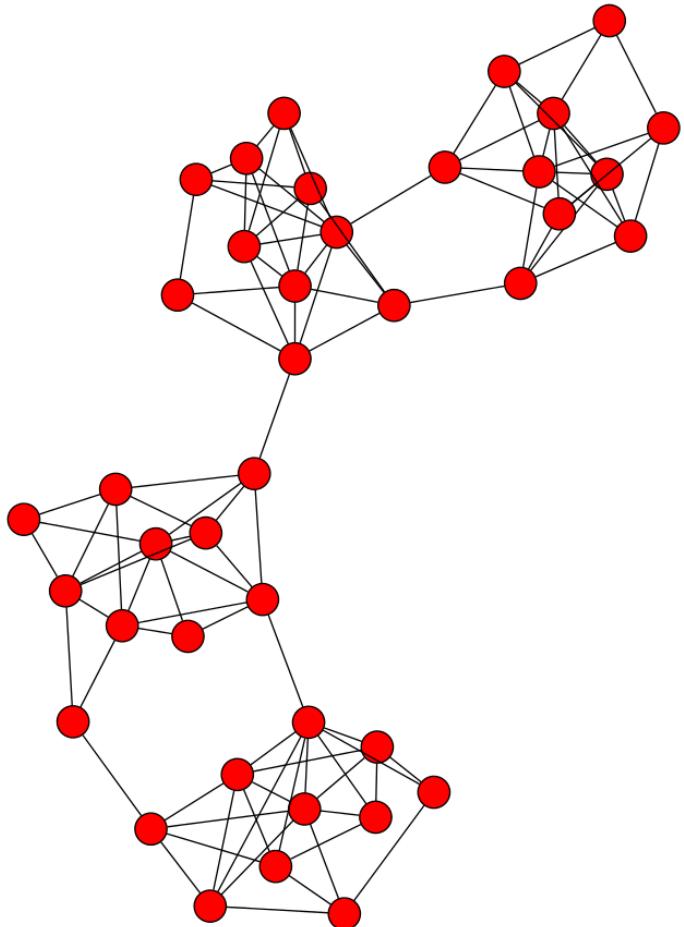
# Example: Spectral Partitioning



# Example: Spectral Partitioning



# 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...

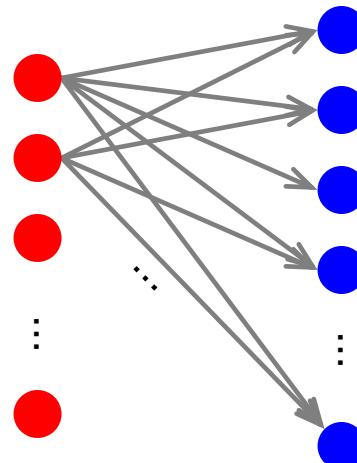
# Why use multiple eigenvectors?

- **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

# 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!**

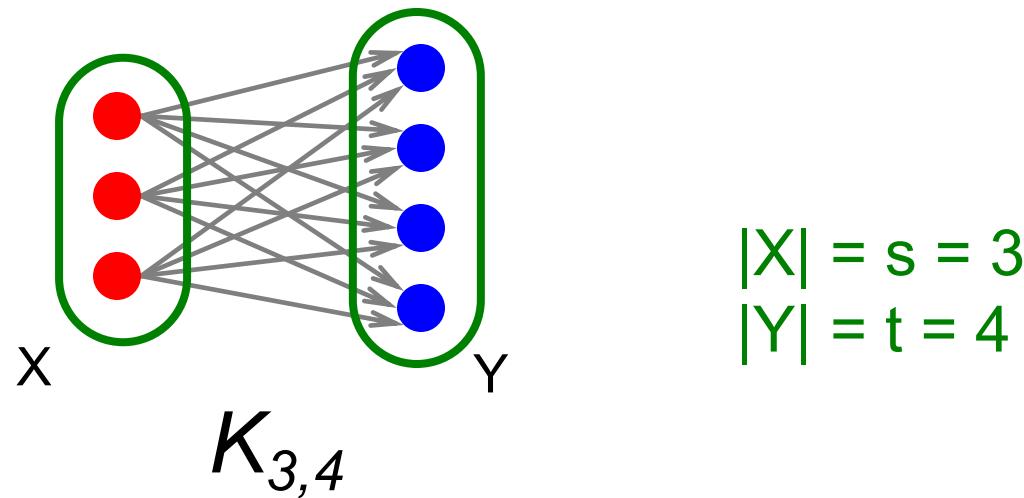
**Intuition:** Many people all talking about the same things

# 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”



Fully connected

# Frequent Itemset Enumeration

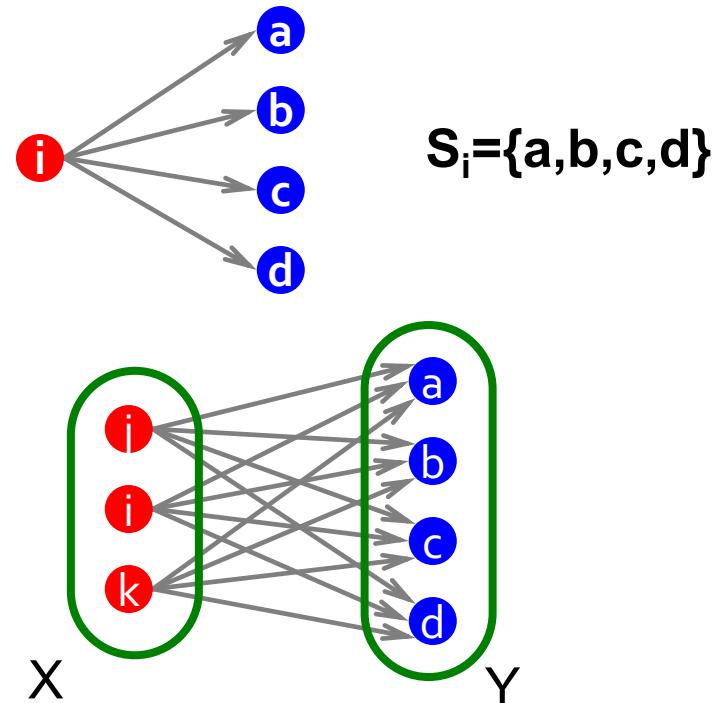
- **Market basket analysis.** Setting:
  - **Market:** Universe  $U$  of  $n$  items
  - **Baskets:**  $m$  subsets of  $U$ :  $S_1, S_2, \dots, S_m \subseteq U$   
( $S_i$  is a set of items one person bought)
  - **Support:** Frequency threshold  $f$
- **Goal:**
  - Find all subsets  $T$  s.t.  $T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

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

Frequent itemsets = complete bipartite graphs!

## ■ How?

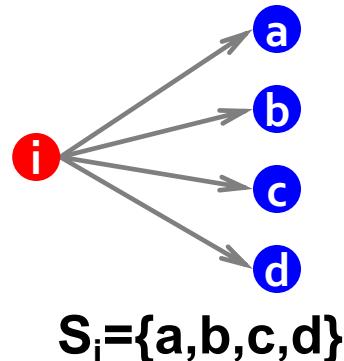
- View each node  $i$  as a set  $S_i$  of nodes  $i$  points to
- $K_{s,t} =$  a set  $Y$  of size  $t$  that occurs in  $s$  sets  $S_i$
- Looking for  $K_{s,t} \rightarrow$  set of frequency threshold to  $s$  and look at layer  $t$  – all frequent sets of size  $t$



**s** ... minimum support ( $|X|=s$ )  
**t** ... itemset size ( $|Y|=t$ )

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

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

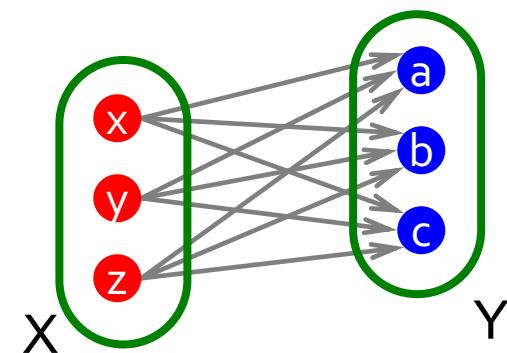
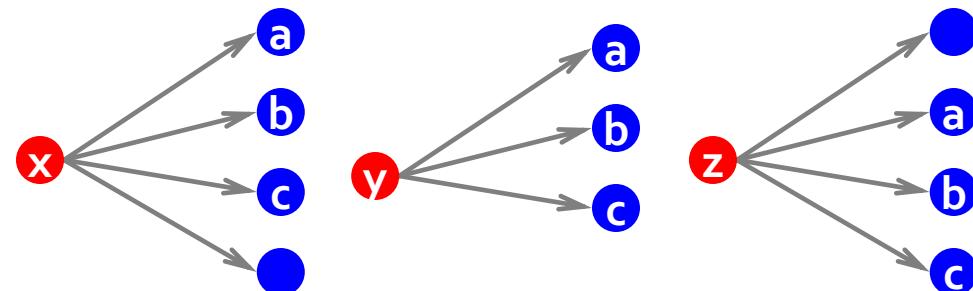


Find frequent itemsets:  
**s** ... minimum support  
**t** ... itemset size

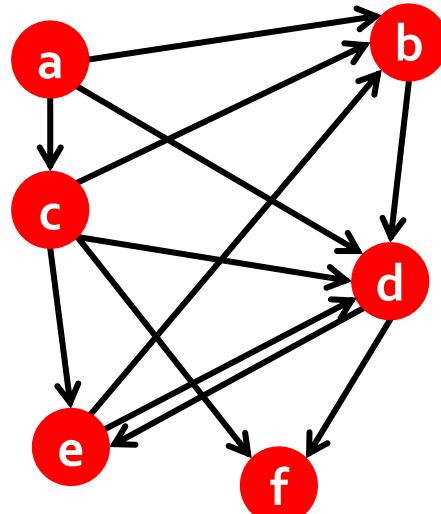
We found  $K_{s,t}$ !

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

Say we find a **frequent itemset**  $Y=\{a,b,c\}$  of supp  $s$   
 So, there are  $s$  nodes that link to all of  $\{a,b,c\}$ :



# Example (1)



**Itemsets:**

$$a = \{b, c, d\}$$

$$b = \{d\}$$

$$c = \{b, d, e, f\}$$

$$d = \{e, f\}$$

$$e = \{b, d\}$$

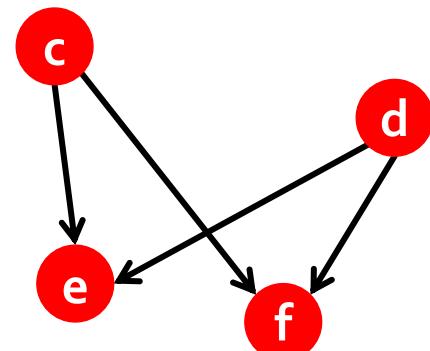
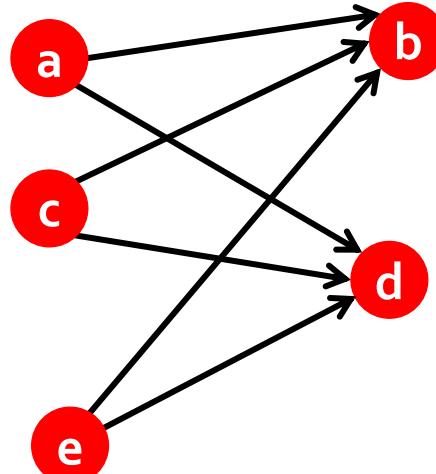
$$f = \{\}$$

■ **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

NSW Rural Fire Service Internet Site  
NSW Fire Brigades  
Sutherland Rural Fire Service  
CFA: County Fire Authority  
“The National Cente...ted Children’s Ho...  
CRAFTI Internet Connexions-INFO  
Welcome to Blackwoo... Fire Safety Serv...  
The World Famous Guestbook Server  
Wilberforce County Fire Brigade  
NEW SOUTH WALES FIR...ES 377 STATION  
Woronora Bushfire Brigade  
Mongarlowe Bush Fire – Home Page  
Golden Square Fire Brigade  
FIREBREAK Home Page  
Guises Creek Volunt...fficial Home Page...

### Nodes on the left

New South Wales Fir...ial Australian Links  
Feuerwehrlinks Australien  
FireNet Information Network  
The Cherrybrook Rur...re Brigade Home Page  
New South Wales Fir...ial Australian Links  
Fire Departments, F... Information Network  
The Australian Firefighter Page  
Kristiansand brannv...dens brannvesener...  
Australian Fire Services Links  
The 911 F,P,M., Fir...mp; Canada A Section  
Feuerwehrlinks Australien  
Sanctuary Point Rural Fire Brigade  
Fire Trails “l...ghters around the...  
FireSafe – Fire and Safety Directory  
Kristiansand Firede...departments of th...