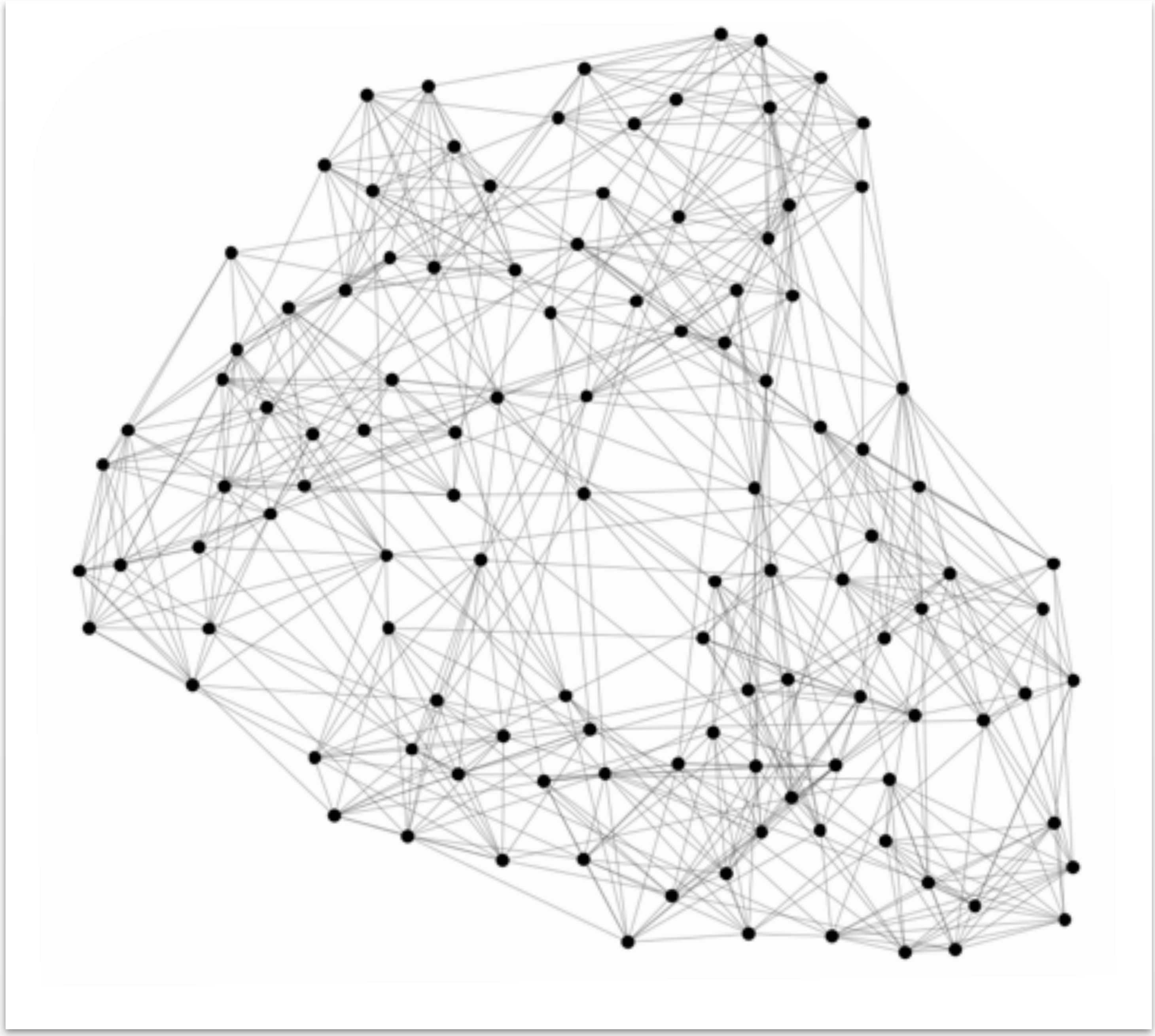
Data Mining Techniques

CS 6220 - Section 3 - Fall 2016

Lecture 19: Social Networks

Jan-Willem van de Meent
(*credit:* Leskovec et al Chapter 10,
Aggarwal Chapter 19)

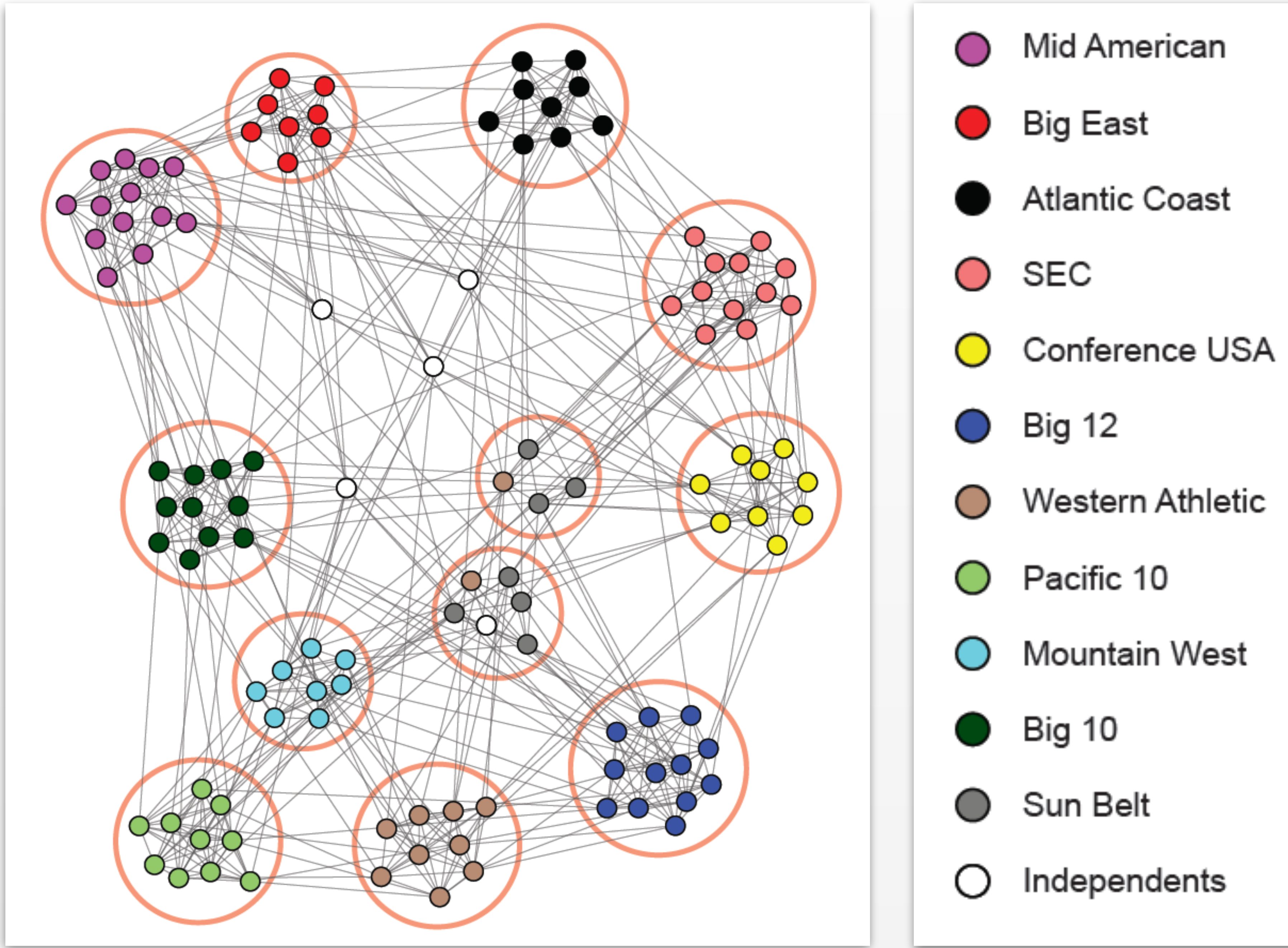
Community Detection



Problem: Can we identify groups of densely connected nodes?

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

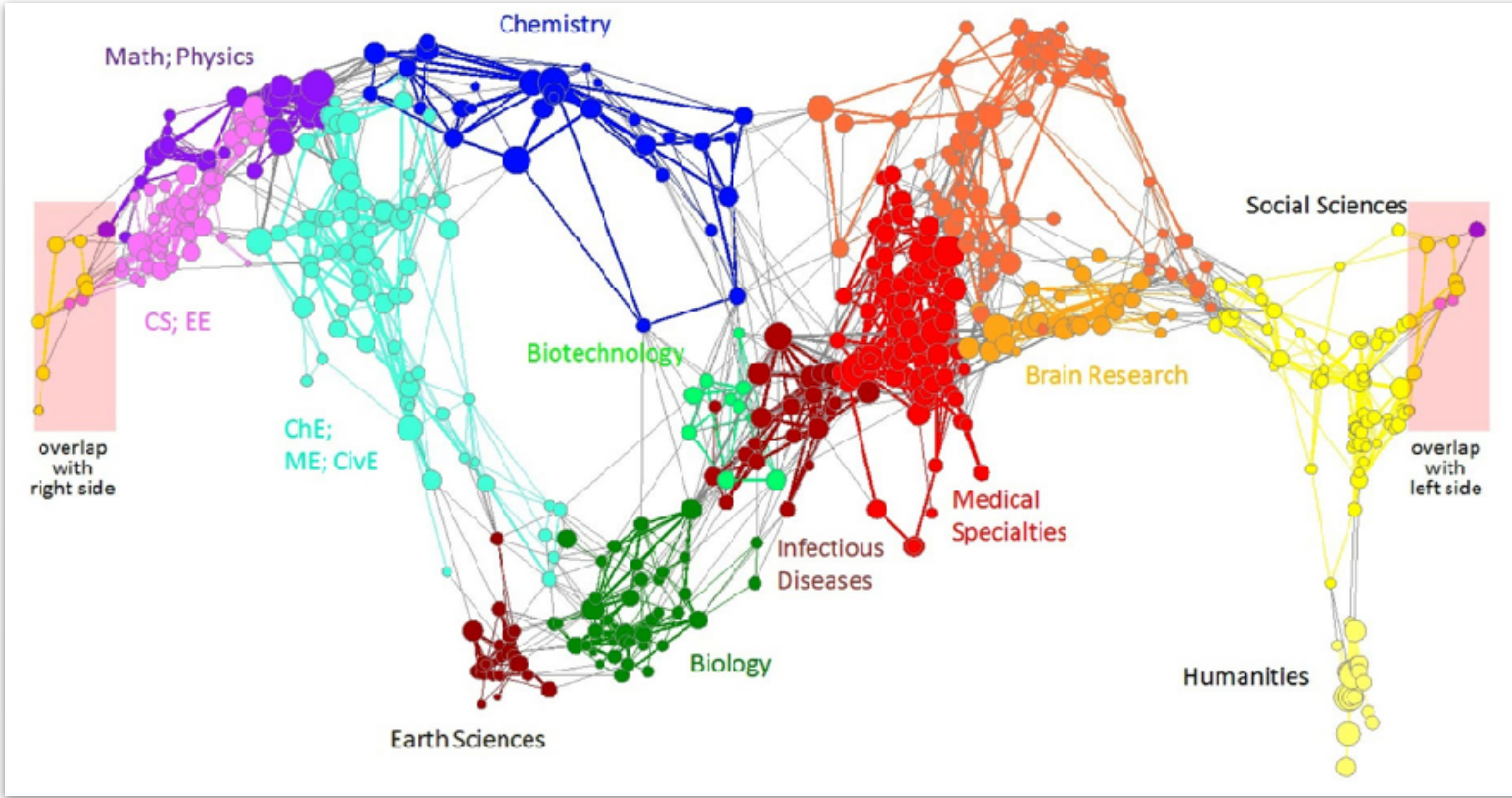
Communities: Football Conferences



Nodes: Football Teams, *Edges:* Matches,
Communities: Conferences

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Communities: Academic Citations

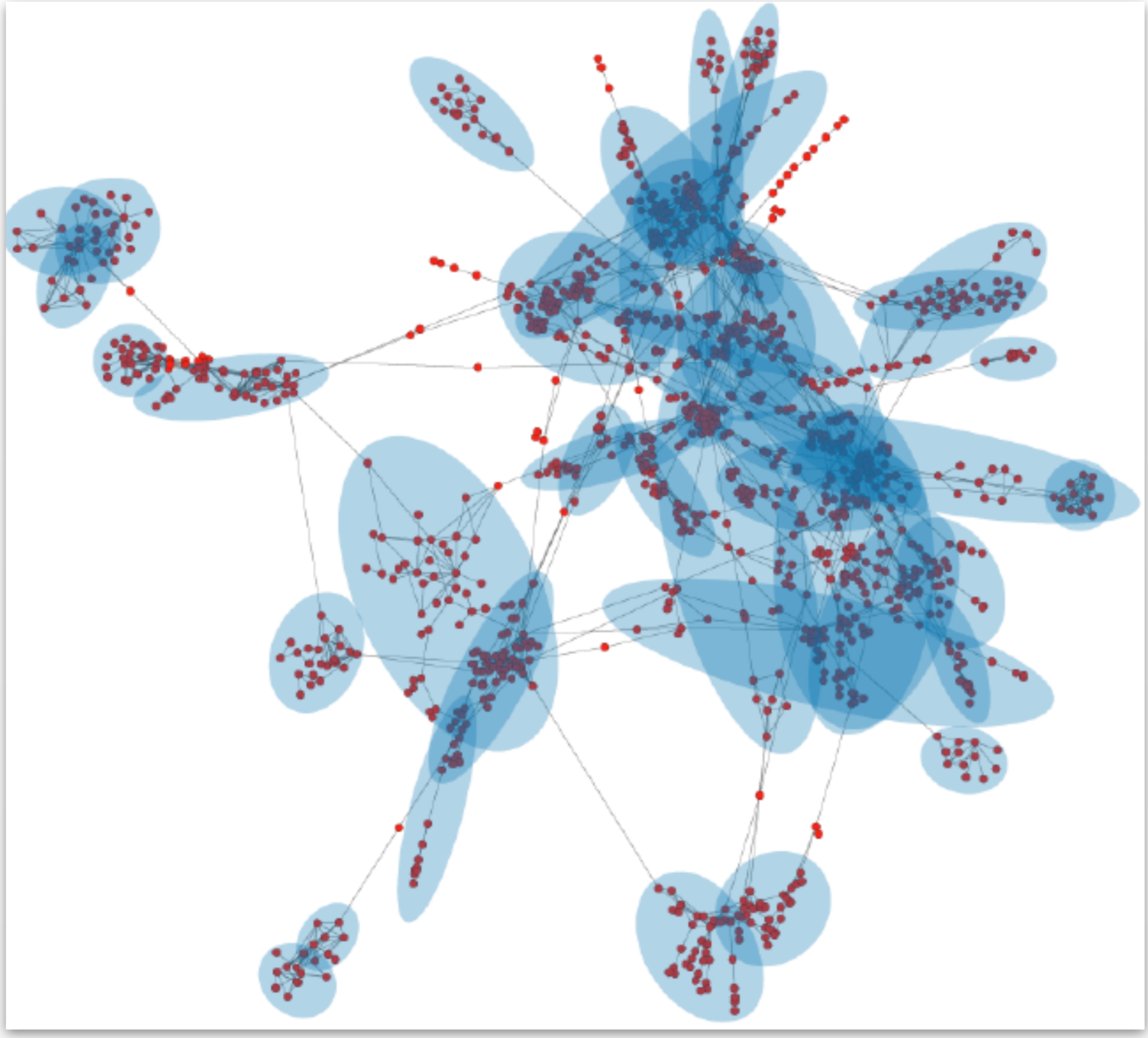


Source: Citation networks and Maps of science [Börner et al., 2012]

Nodes: Journals, *Edges:* Citations,
Communities: Academic Disciplines

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Communities: Protein-Protein Interactions

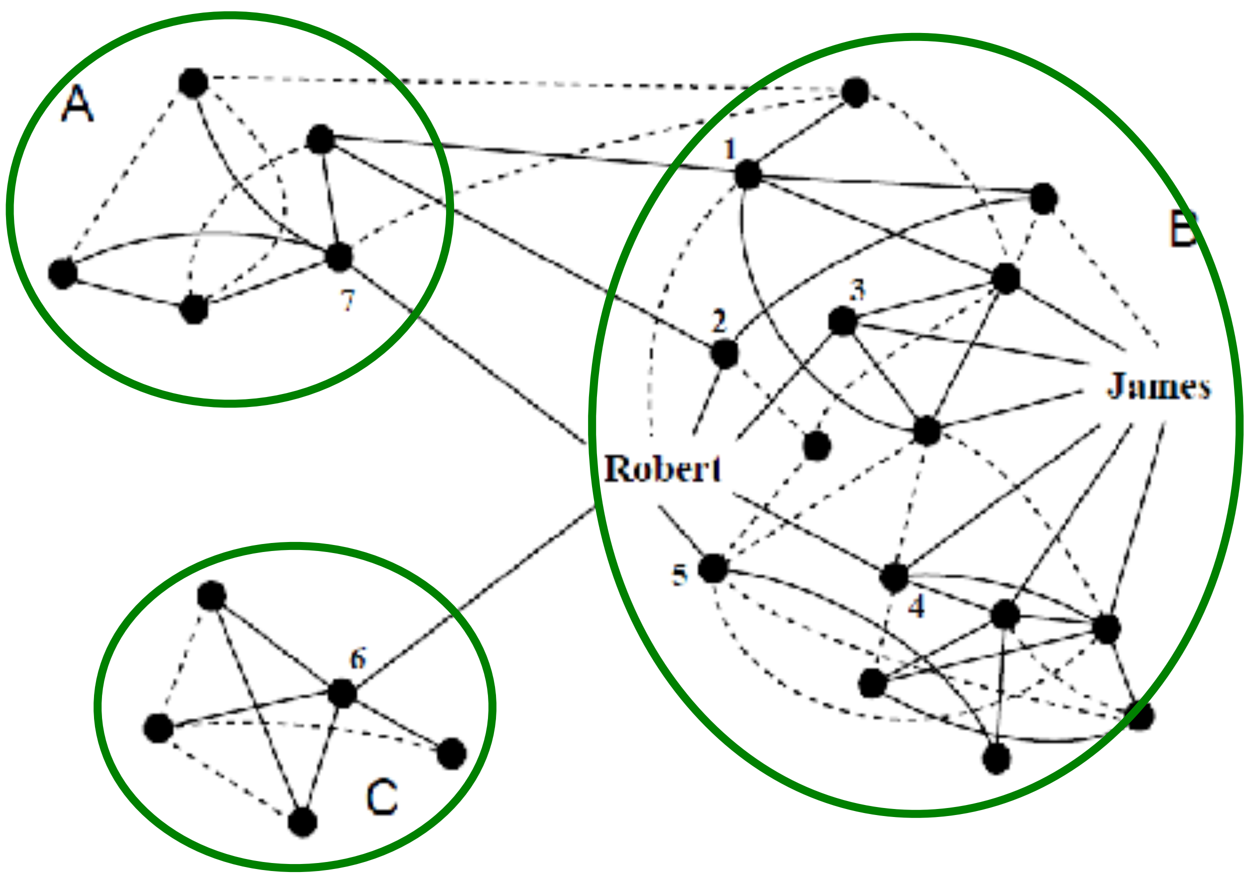


Nodes: Proteins, *Edges:* Physical interactions,
Communities: Functional Modules

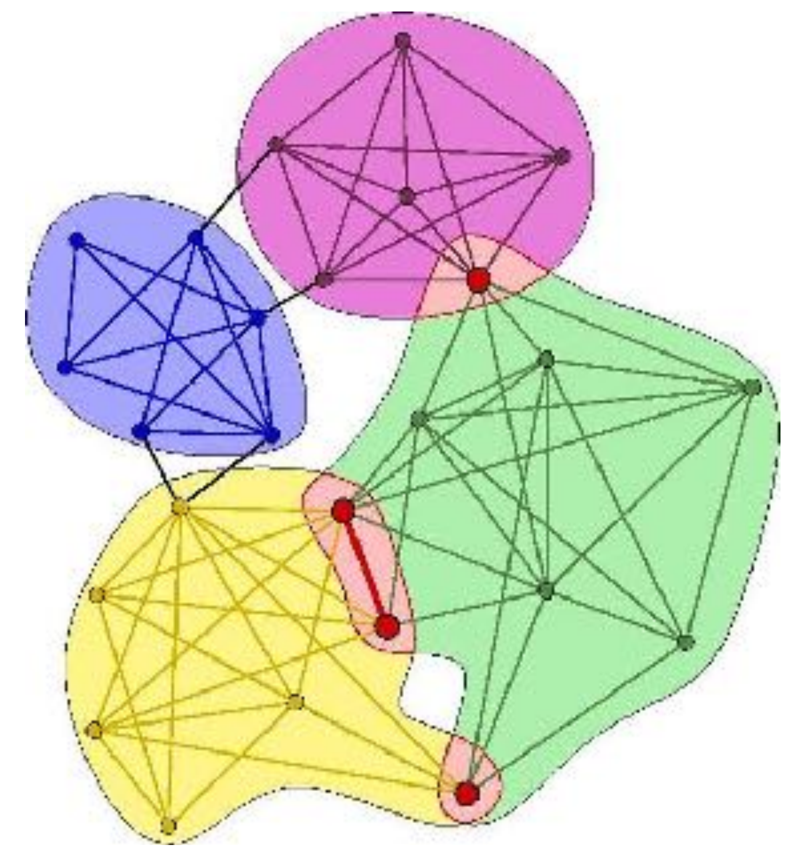
(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Community Detection

Graph Partitioning



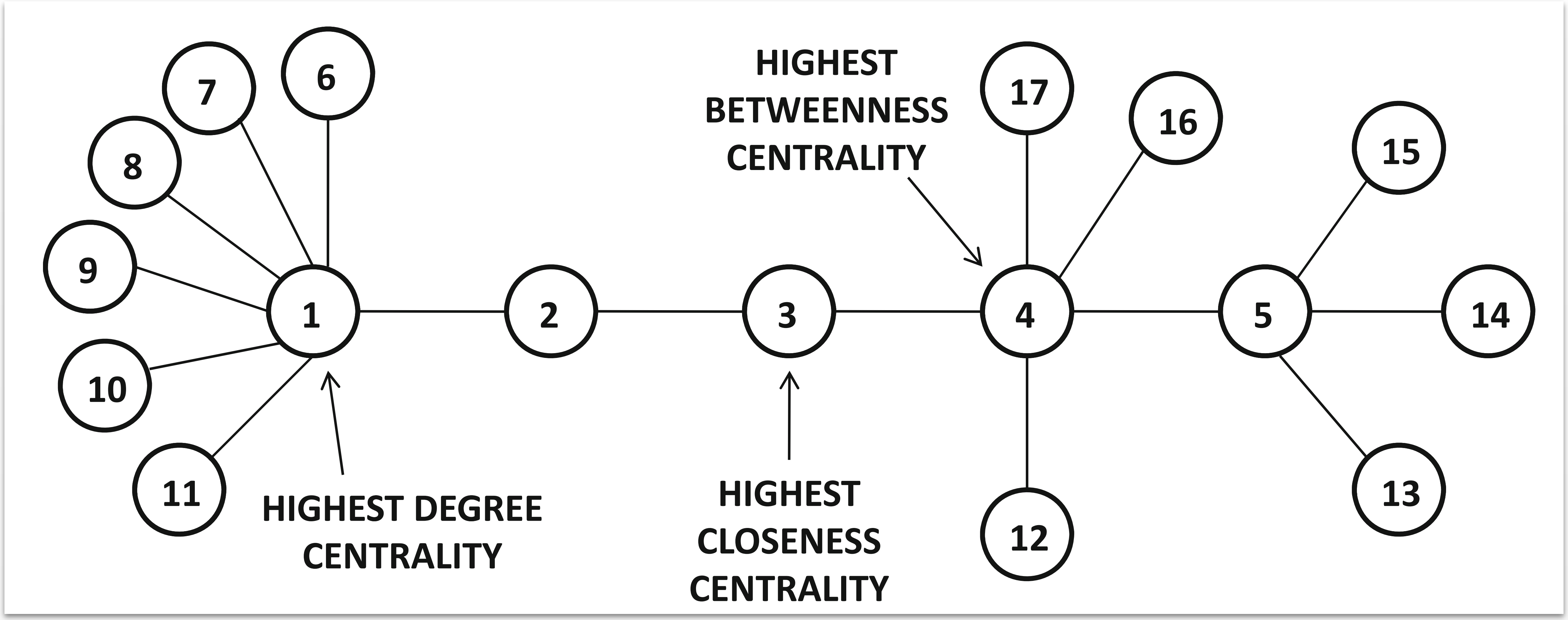
Overlapping Communities



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

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

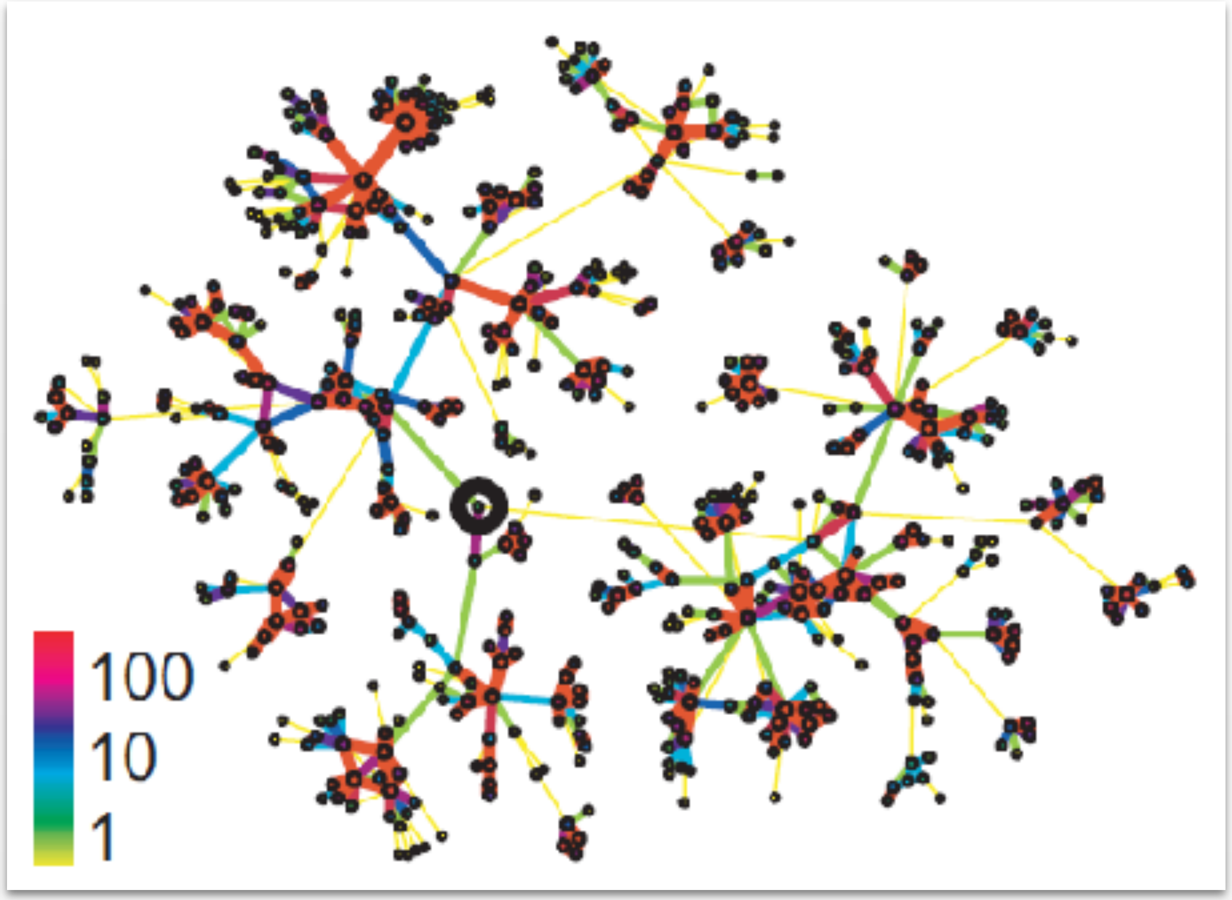
Centrality Measures



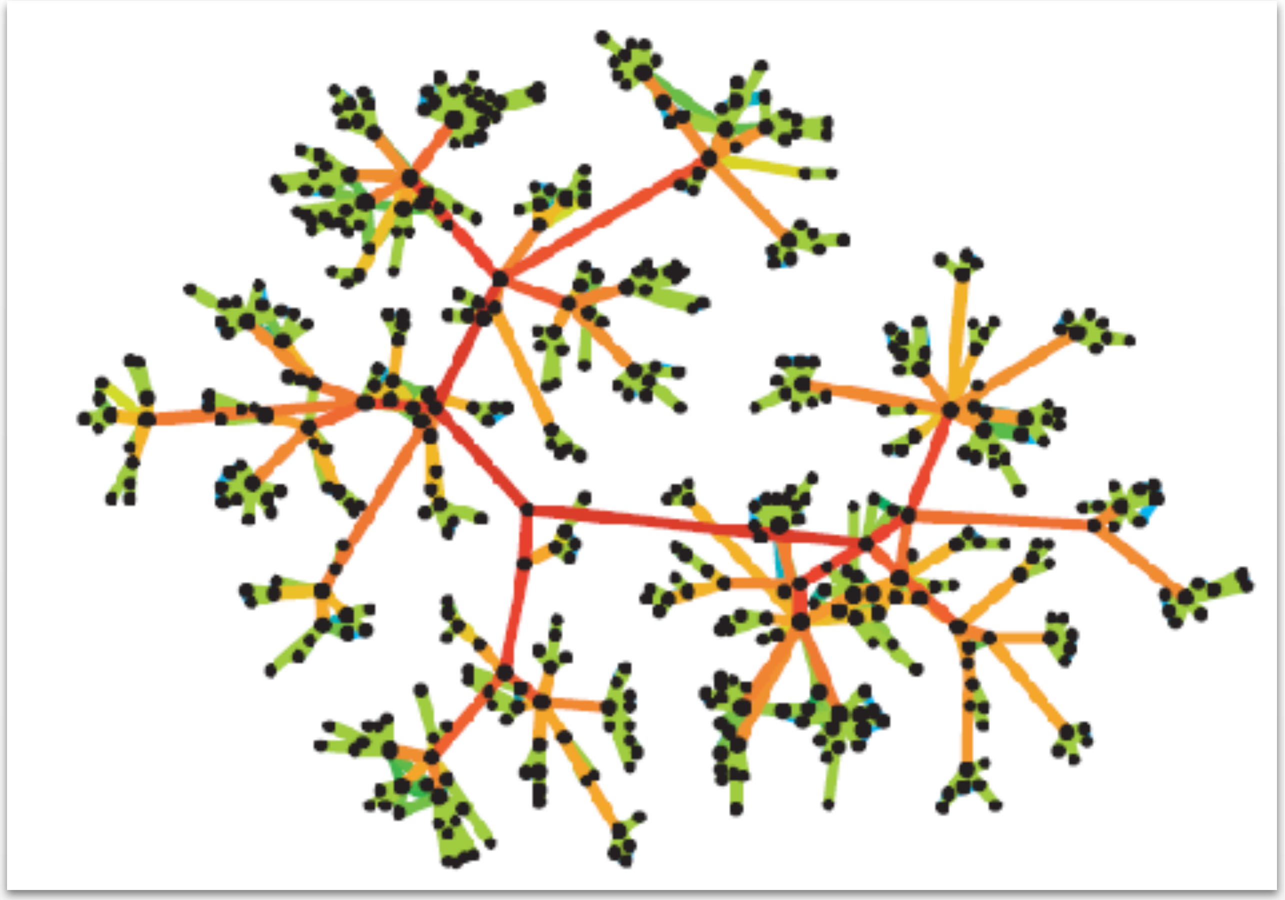
- *Betweenness*: Number of shortest paths
- *Closeness*: Average distance to other nodes
- *Degree*: Number of connections to other nodes

Betweenness

Edge Strength (call volume)



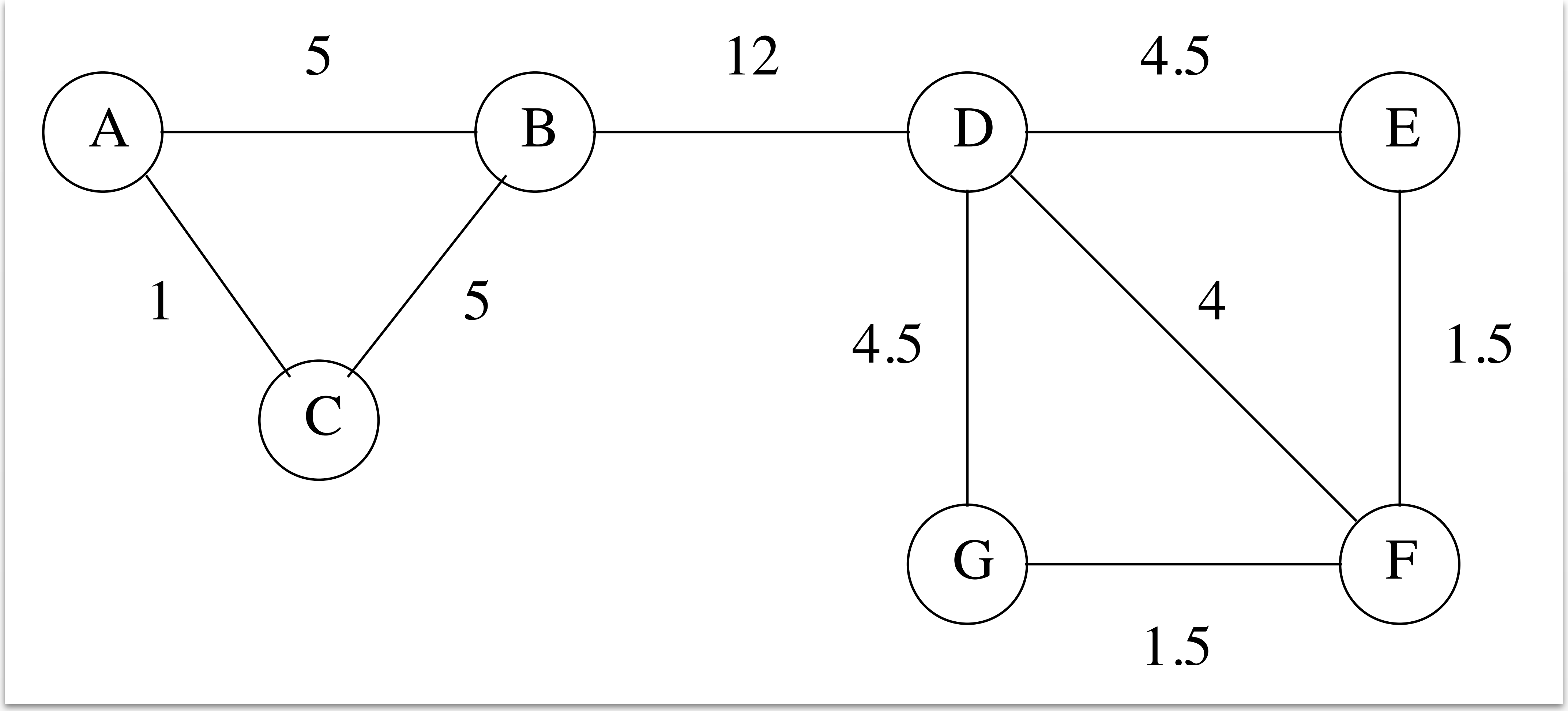
Edge Betweenness



- *Betweenness*: Number of shortest paths passing through a node or edge

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

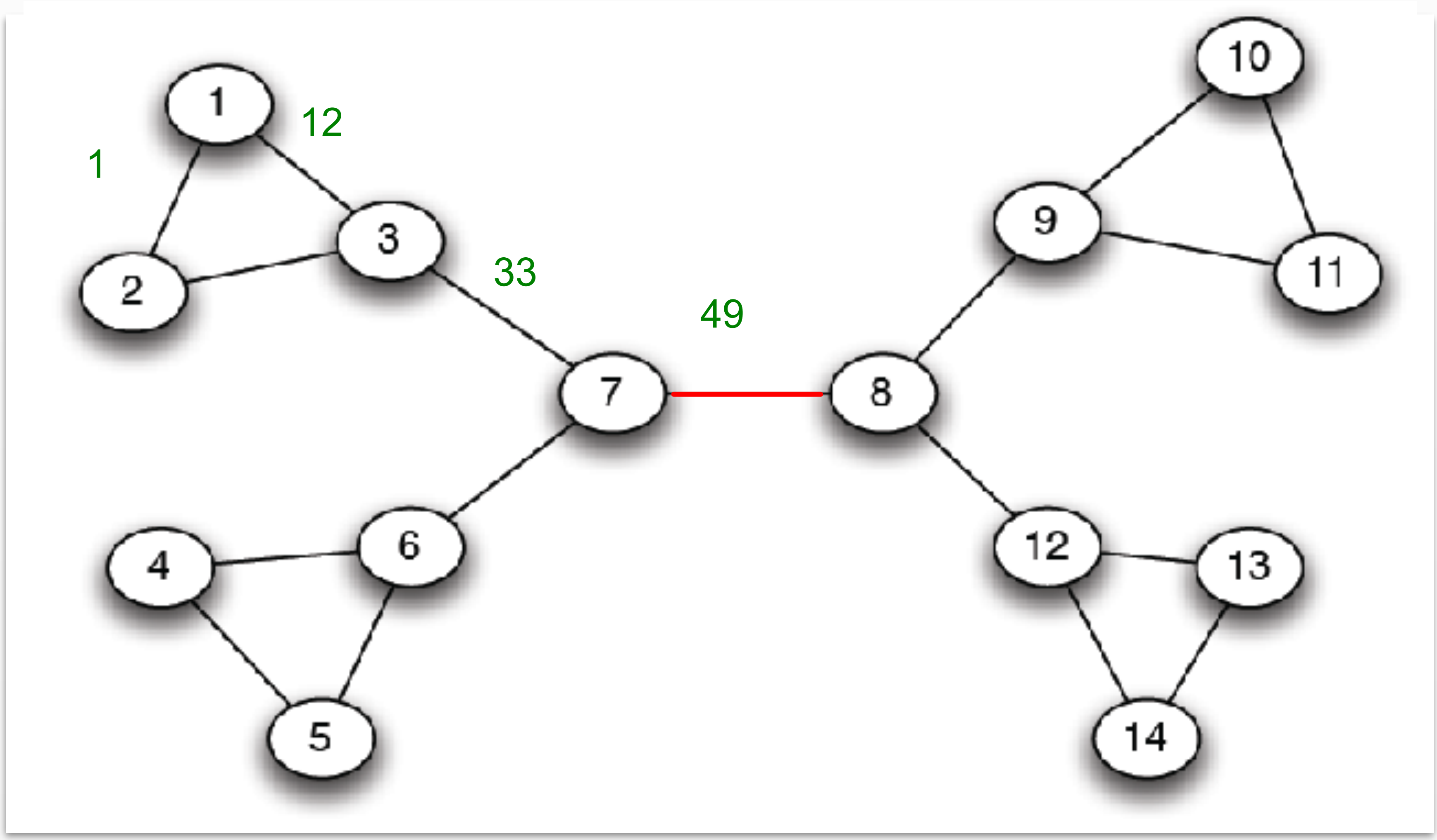
Edge Betweenness



- Count number of shortest paths passing through each edge (*can be done with weighted edges*)
- If there are multiple paths of equal length, then split counts

Girvan-Newman Algorithm

(hierarchical divisive clustering according to betweenness)



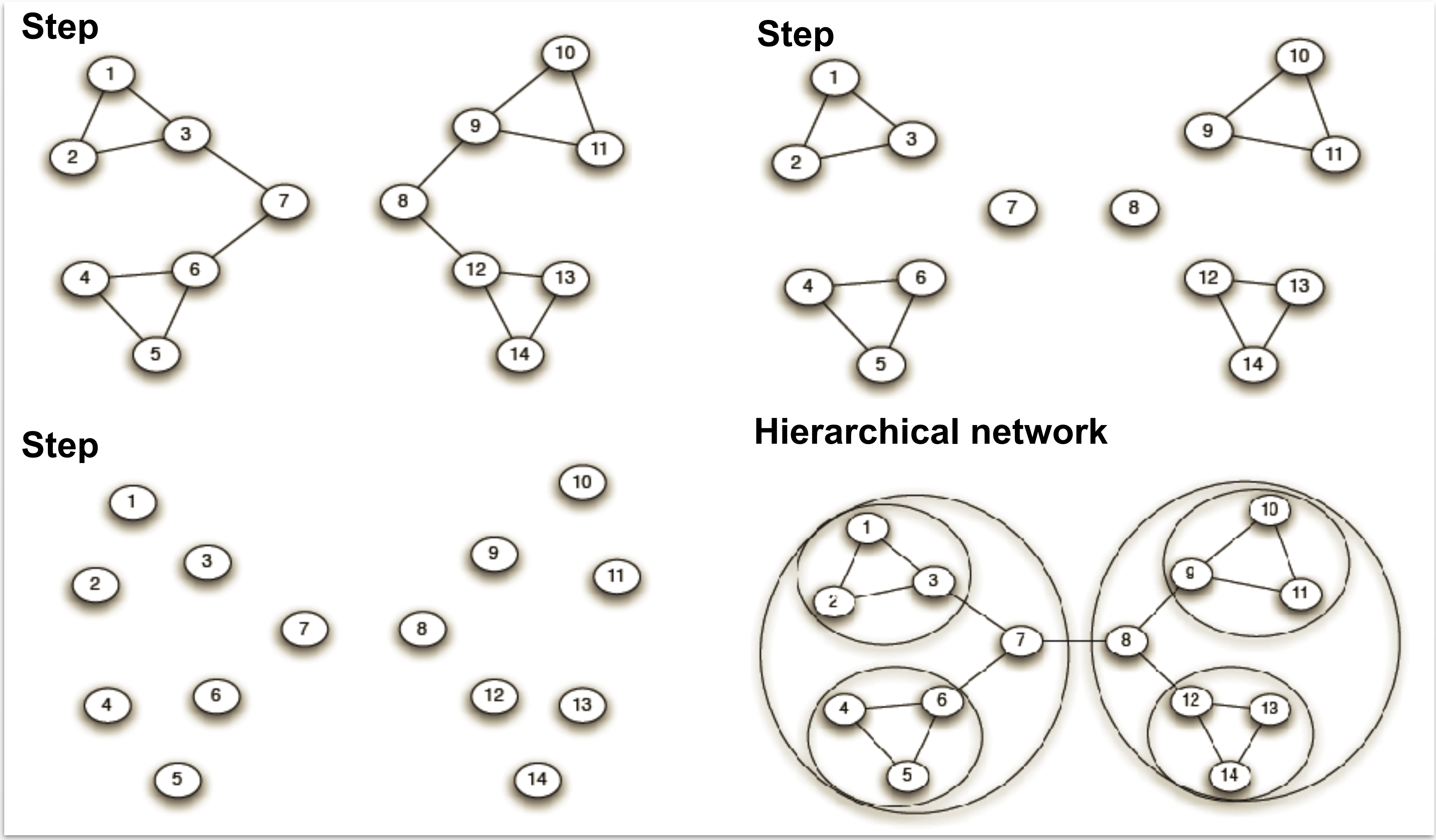
Repeat until k clusters found

1. Calculate betweenness
2. Remove edge(s) with highest betweenness

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

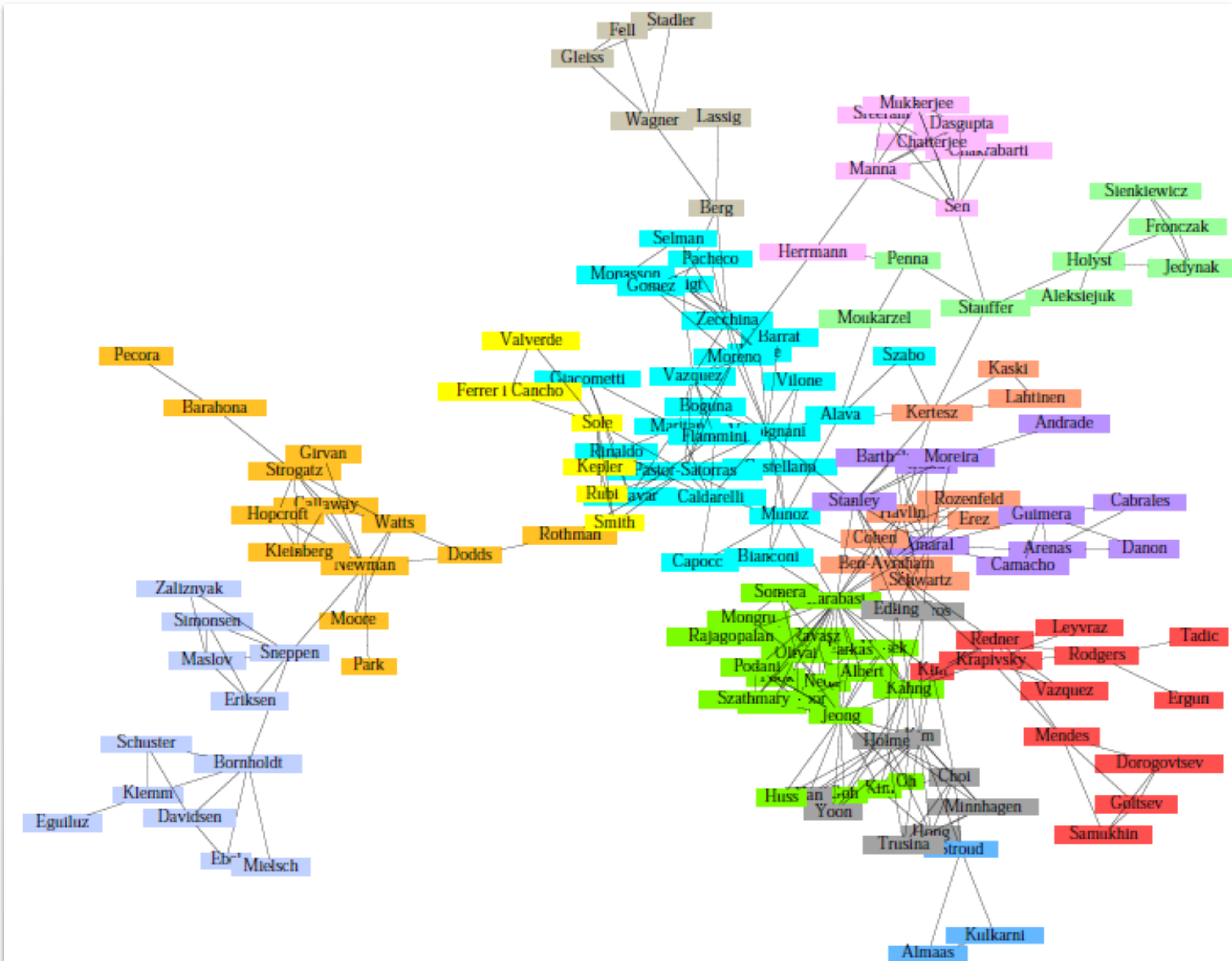
Girvan-Newman Algorithm

(hierarchical divisive clustering according to betweenness)



(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Girvan-Newman: Physics Citations



(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Girvan-Newman

Two problems

1. How can we compute the betweenness for all edges?
2. How can we choose the number of components k?

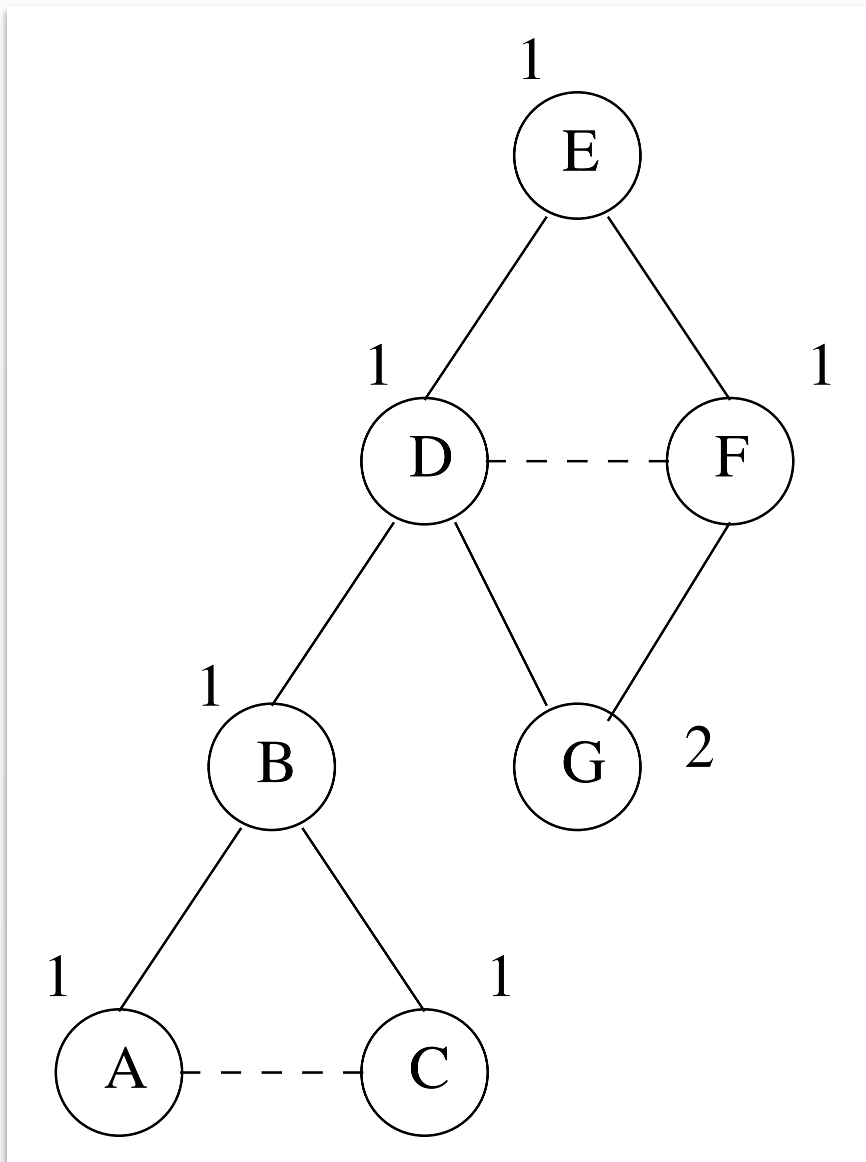
(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Calculating Betweenness

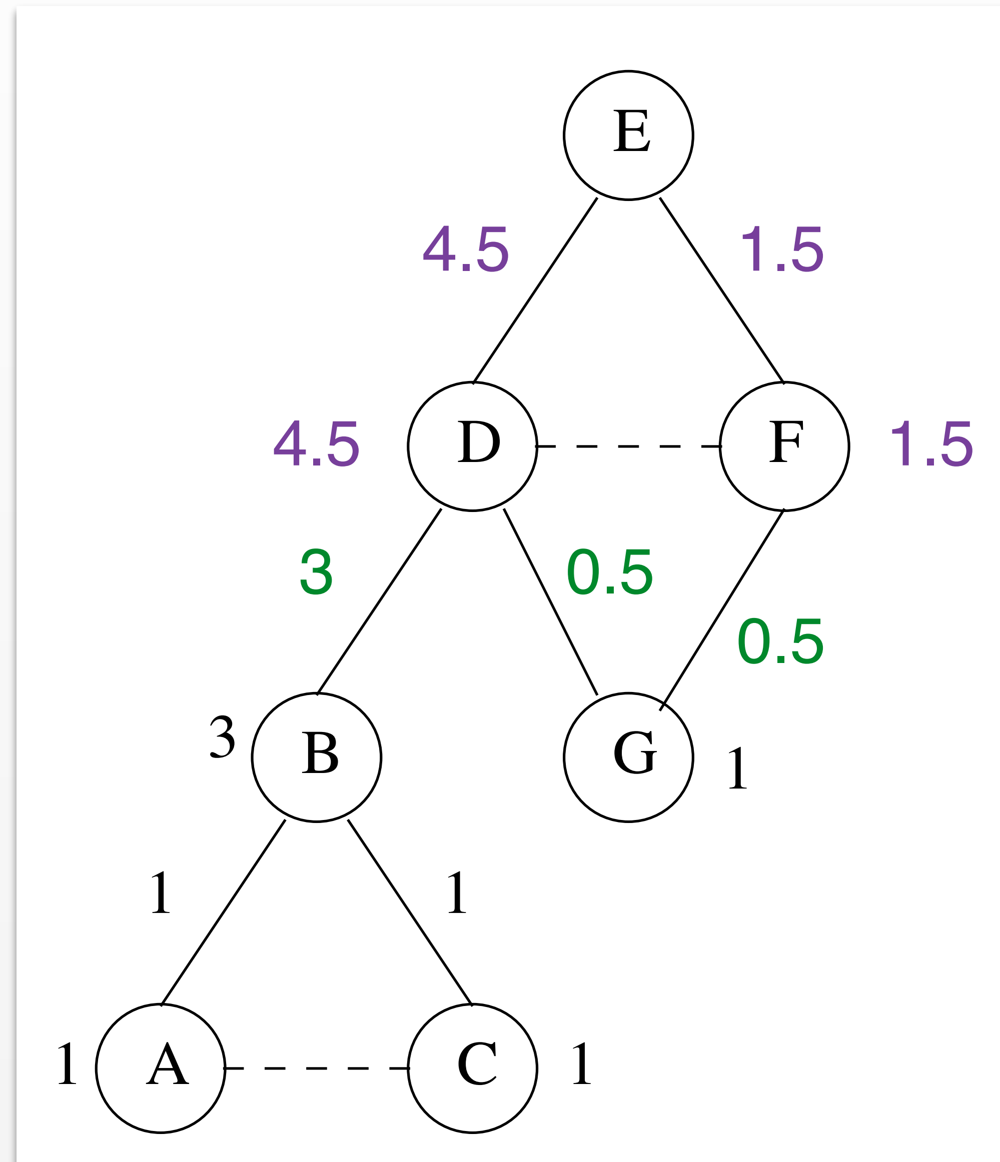
How can we count all shortest paths?

- Loop over nodes in graph
  - Perform breadth-first search to find shortest paths to other nodes
  - Increment counts for edges traversed by shorts paths
- Divide final betweenness by 2 (*since all paths counted twice*)

Counting Shortest Paths



Count number of shortest paths from (E) to each node

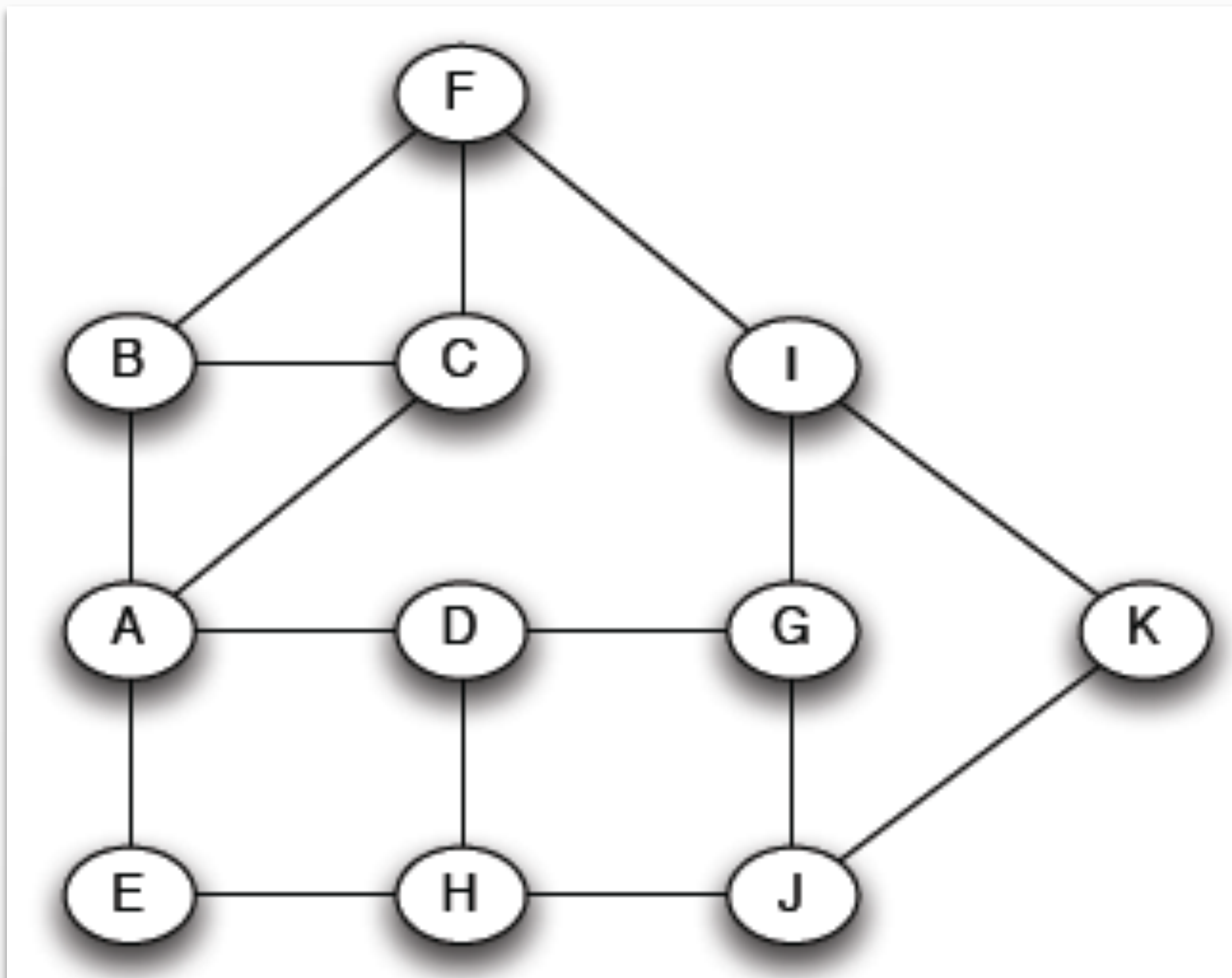


Accumulate credit upwards, dividing across shortest paths

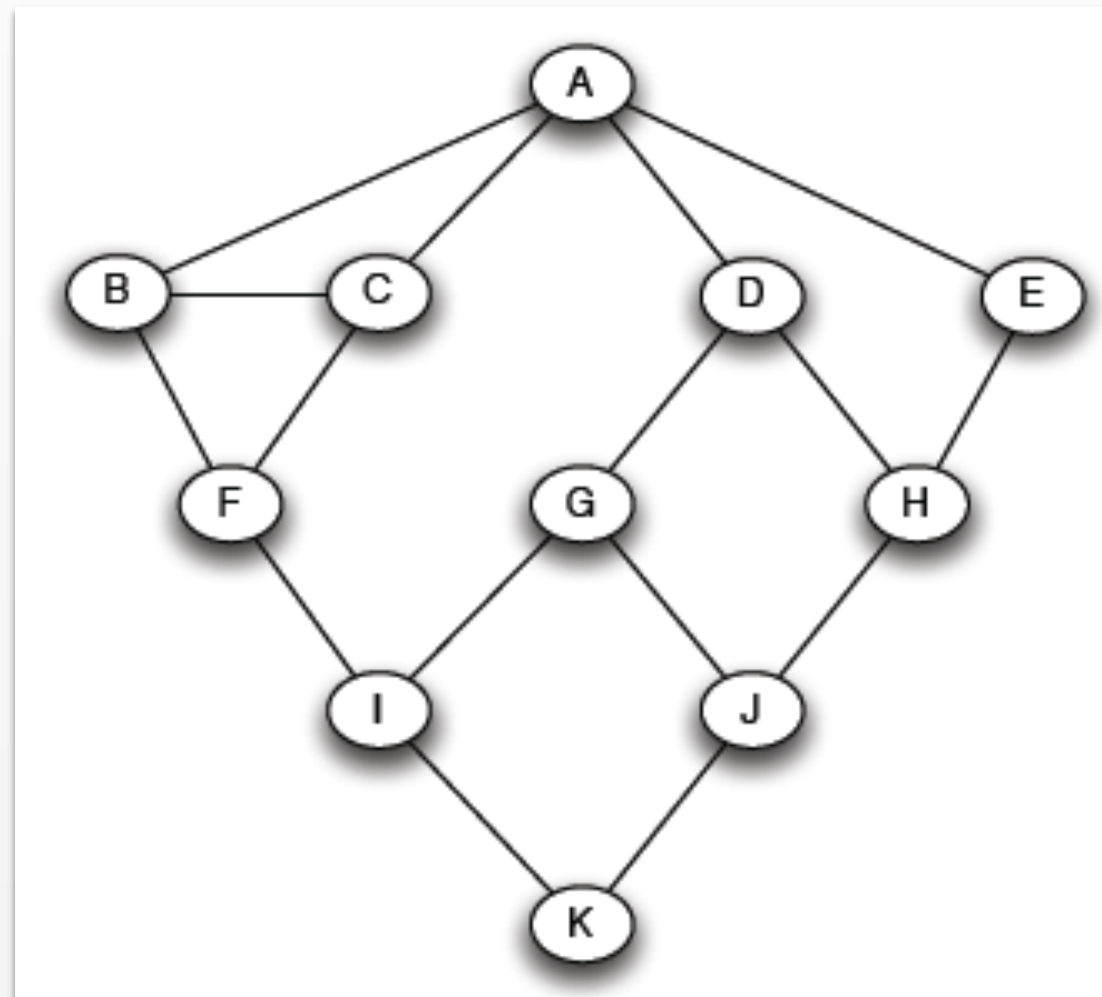
(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Counting Paths: Larger Example

Original Graph

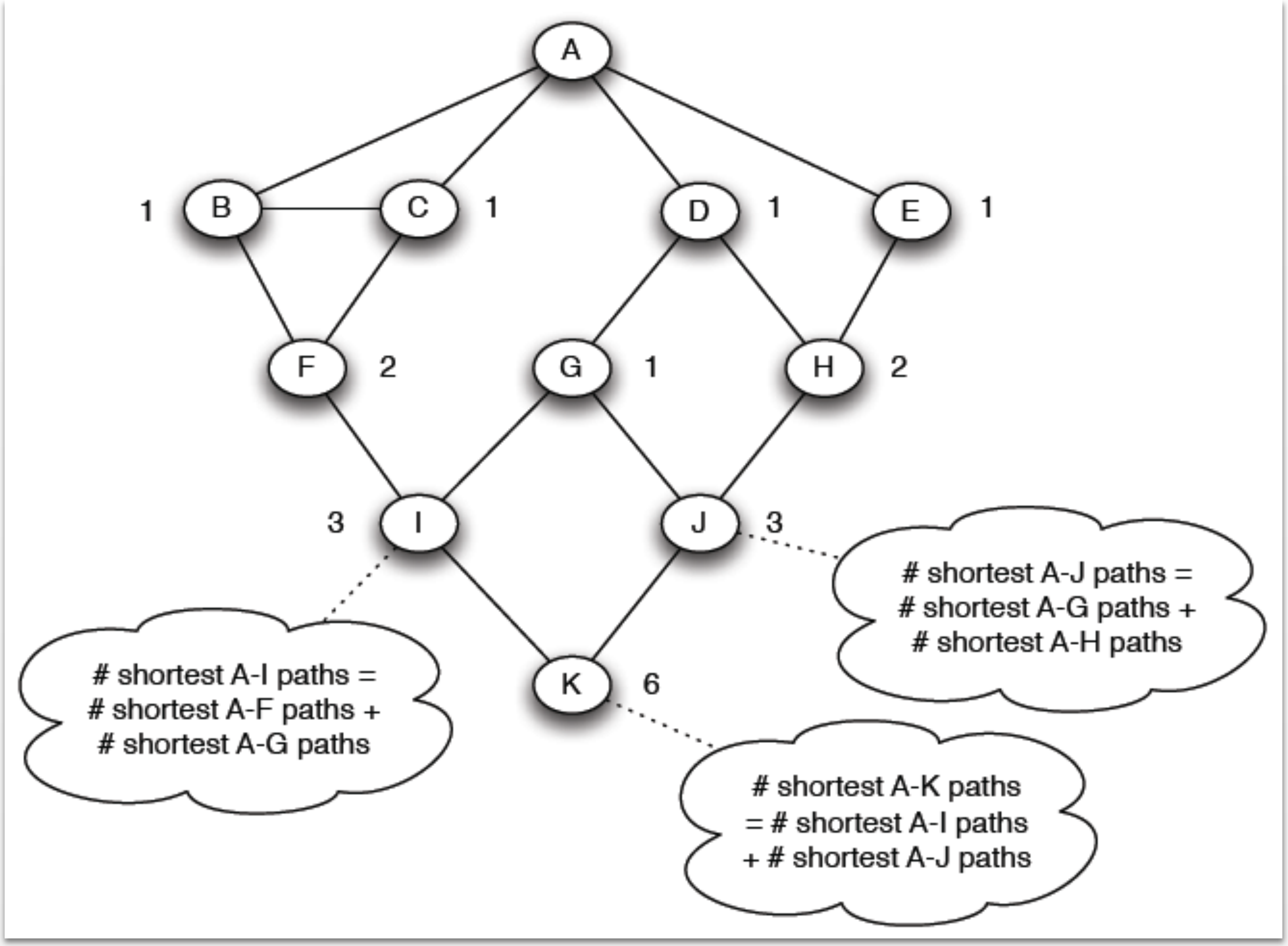


Breadth-first Ordering from A



(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

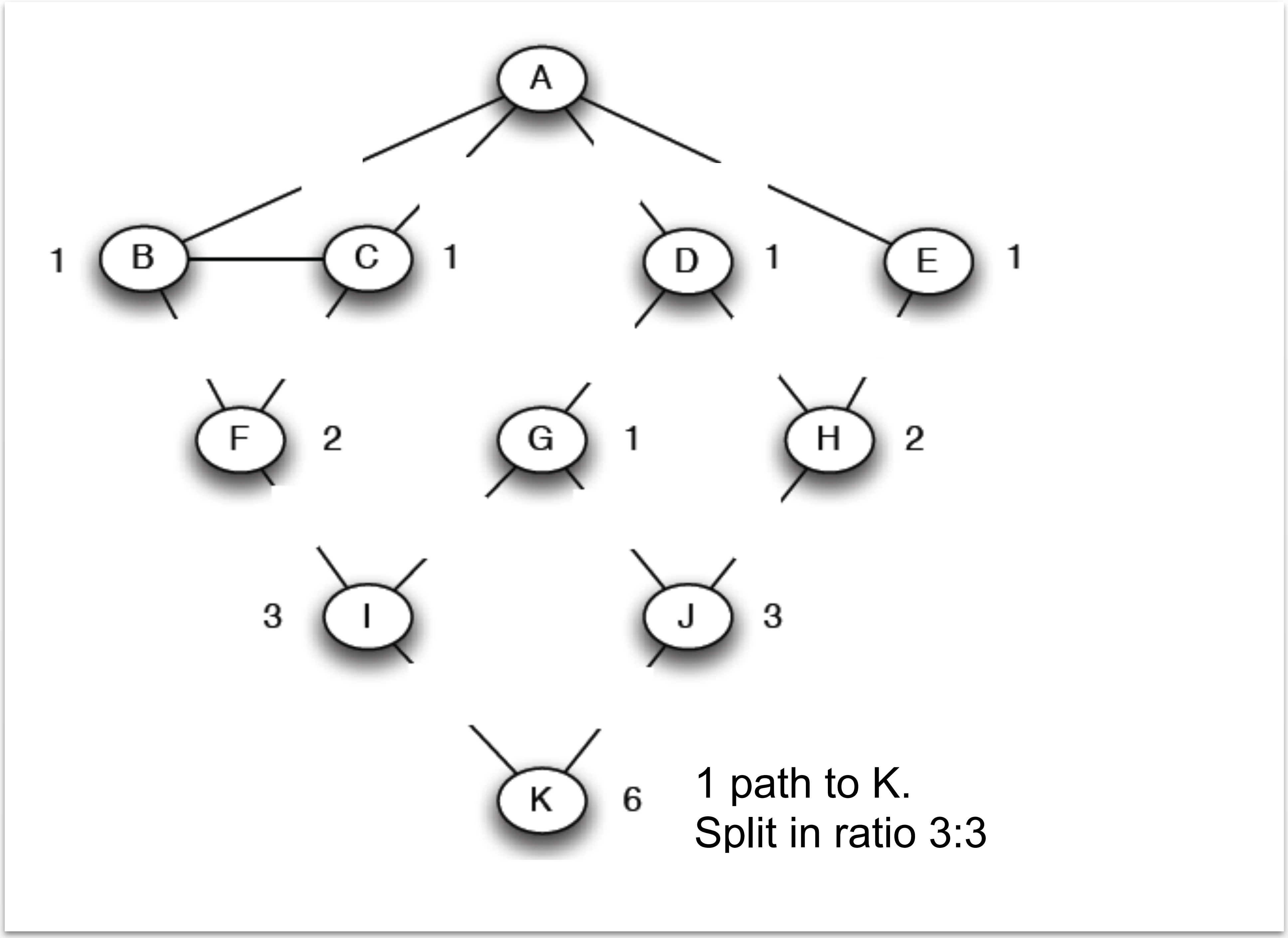
Counting Paths: Larger Example



Step 1. Count number of shortest paths from to each node

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

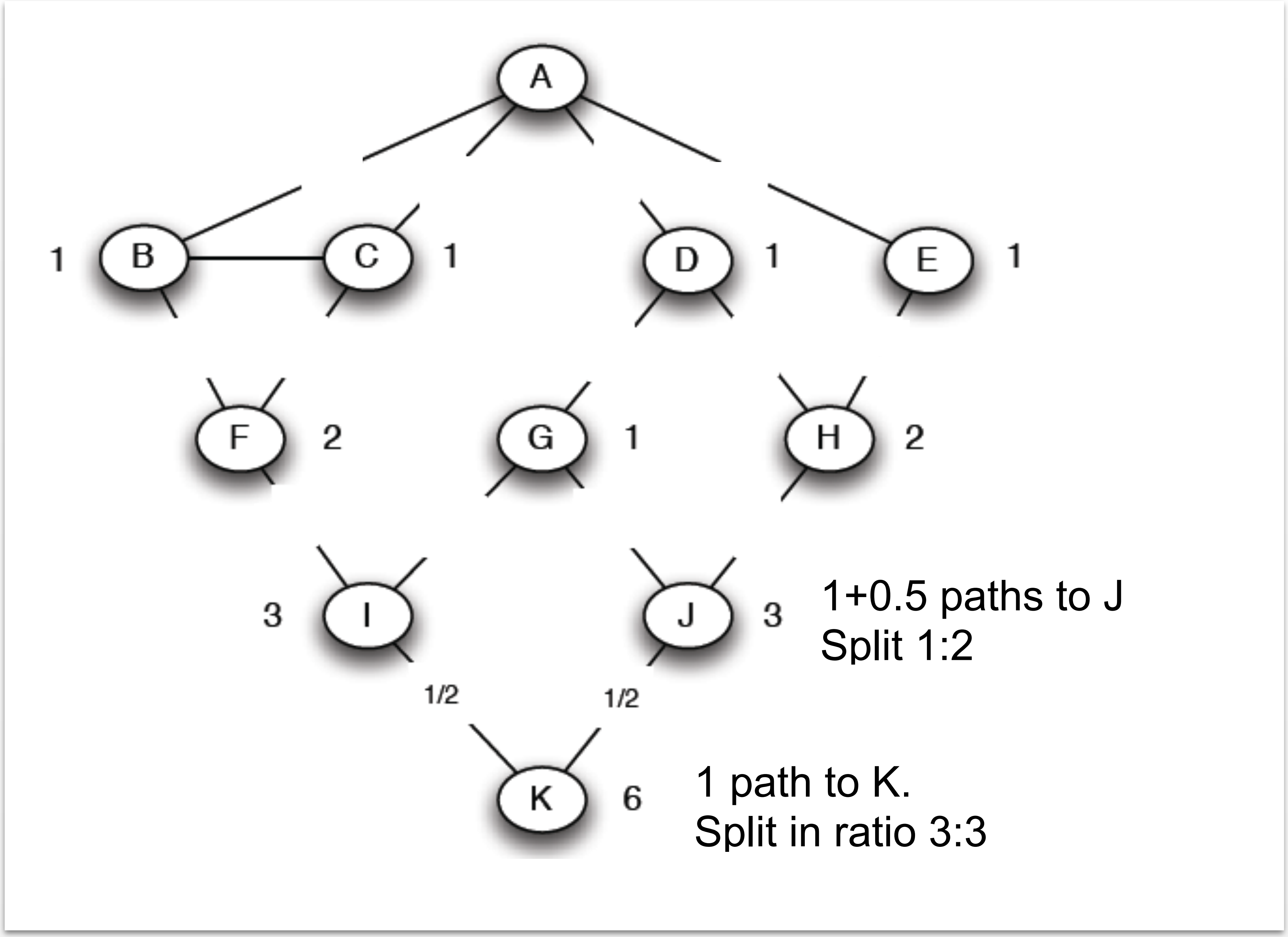
Counting Paths: Larger Example



Step 2. Propagate credit upwards, splitting according to number of paths to parents

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

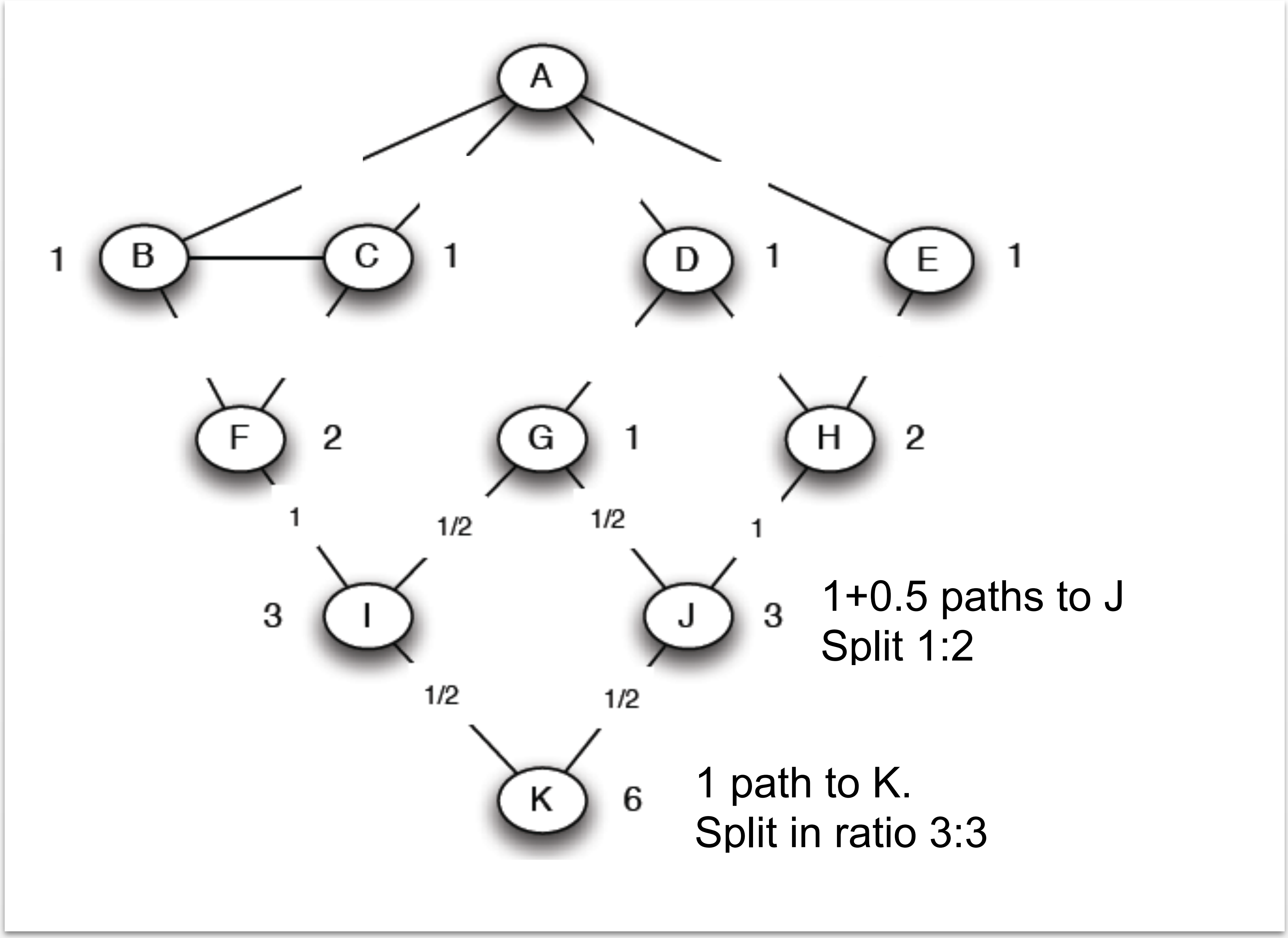
Counting Paths: Larger Example



Step 2. Propagate credit upwards, splitting according to number of paths to parents

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

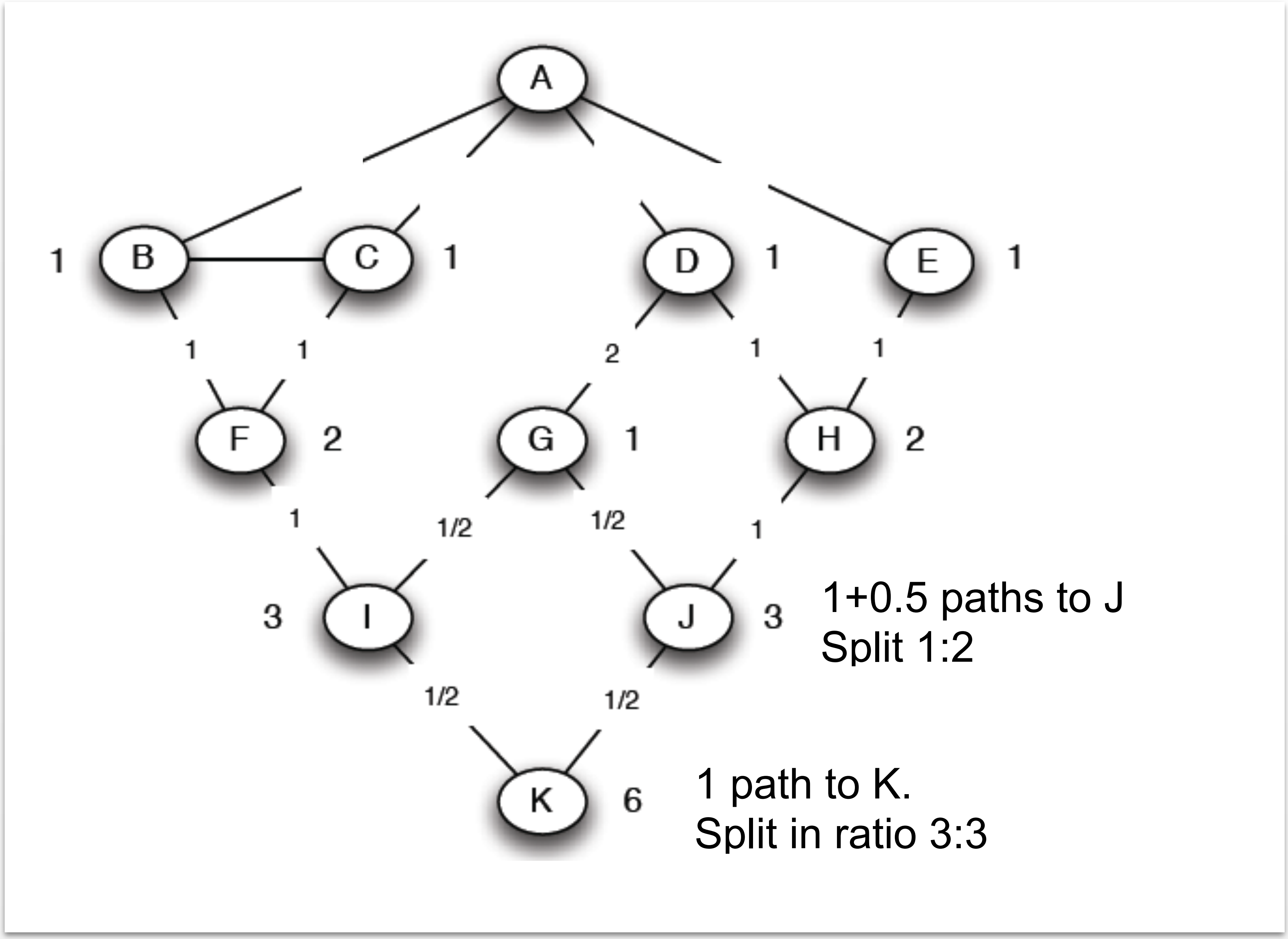
Counting Paths: Larger Example



Step 2. Propagate credit upwards, splitting according to number of paths to parents

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

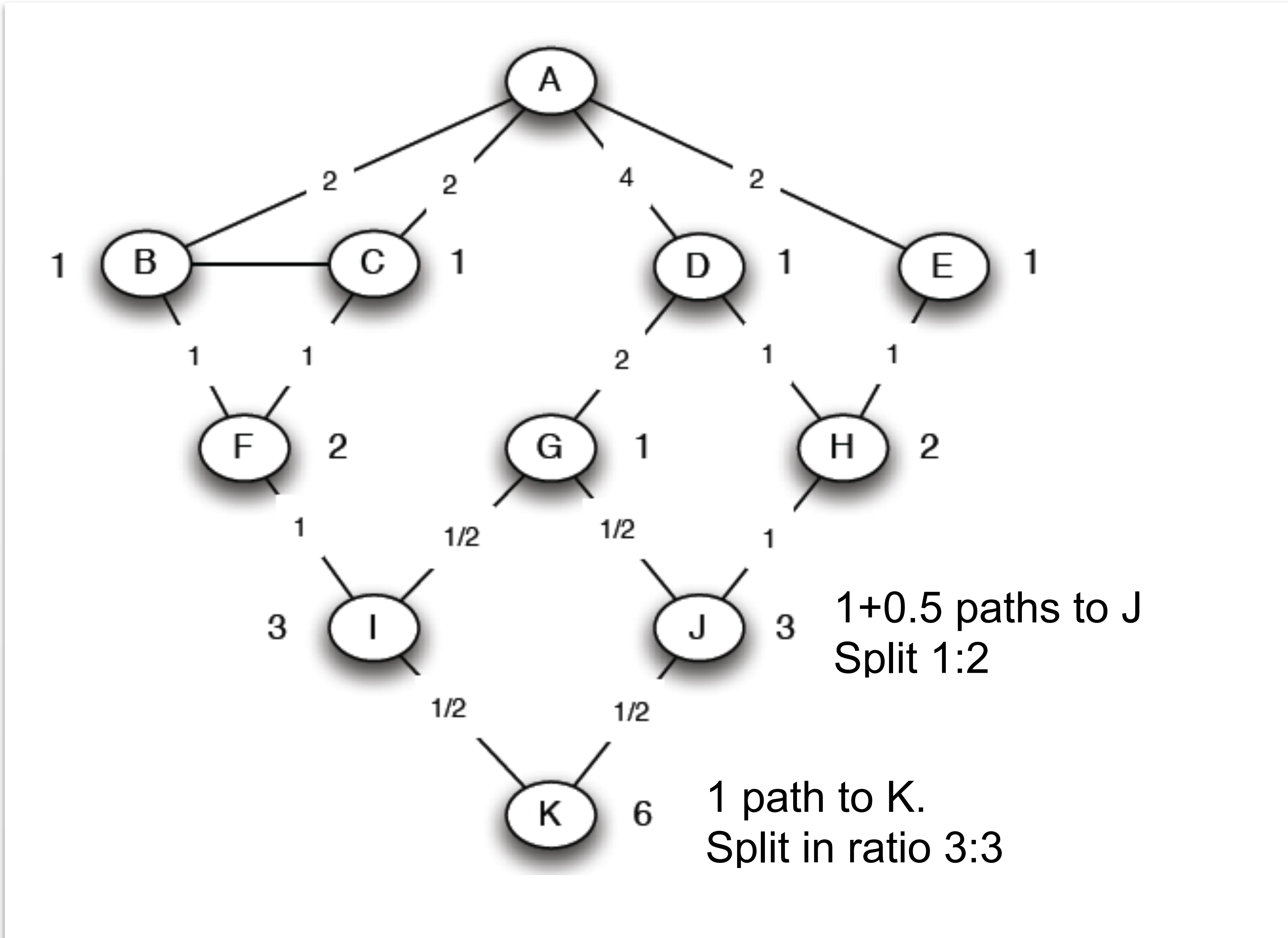
Counting Paths: Larger Example



Step 2. Propagate credit upwards, splitting according to number of paths to parents

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Counting Paths: Larger Example

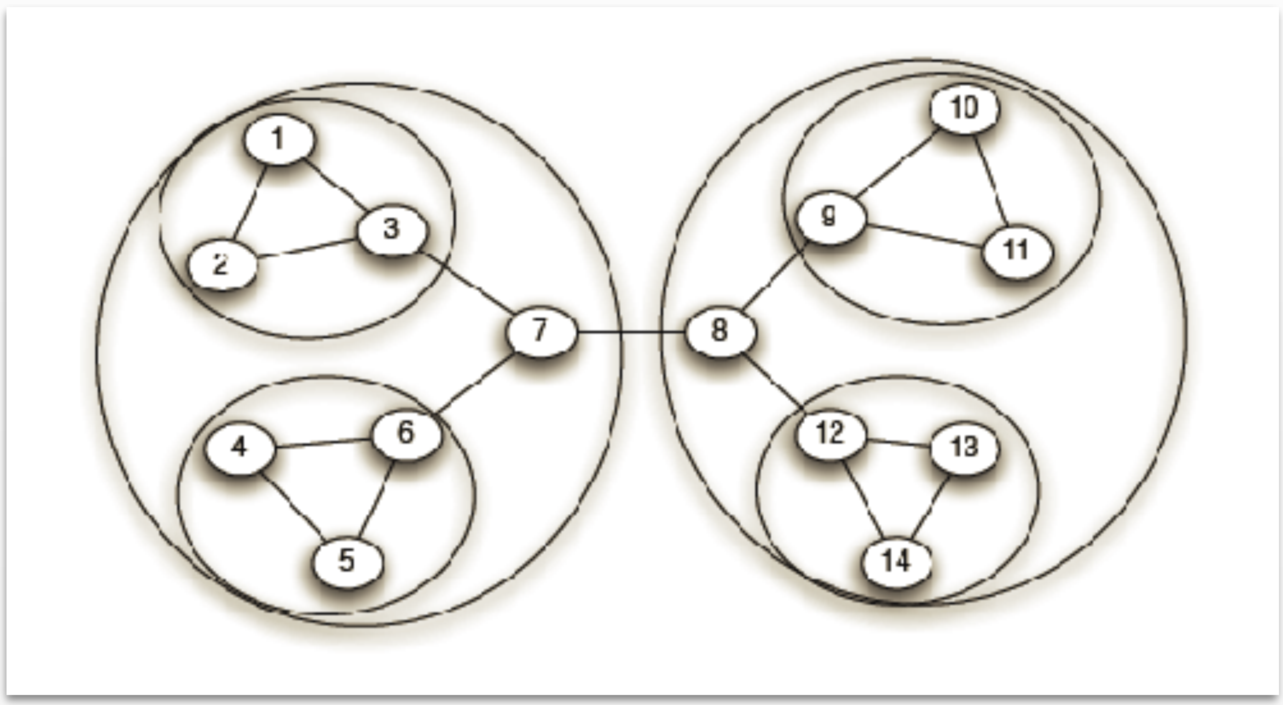


Step 2. Propagate credit upwards, splitting according to number of paths to parents

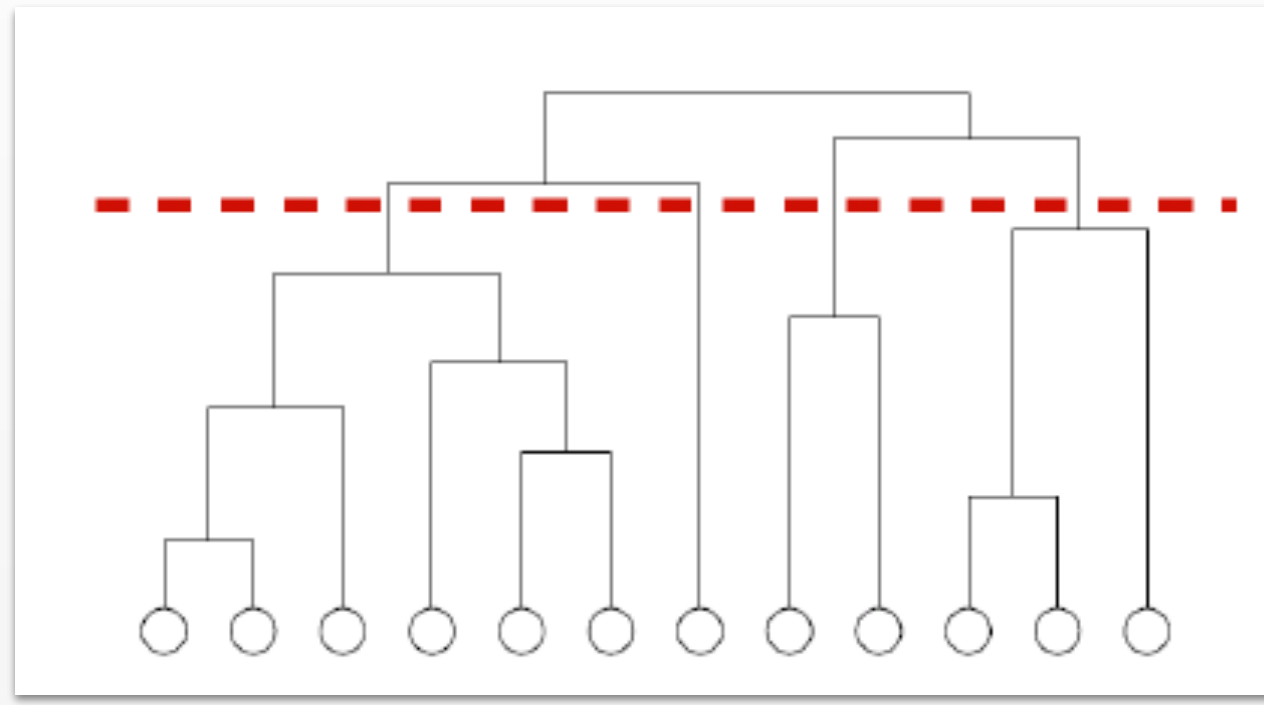
(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Determining the Number of Communities

Hierarchical decomposition



Choosing a cut-off



Analogous problem to deciding on number of clusters in hierarchical clustering

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Modularity

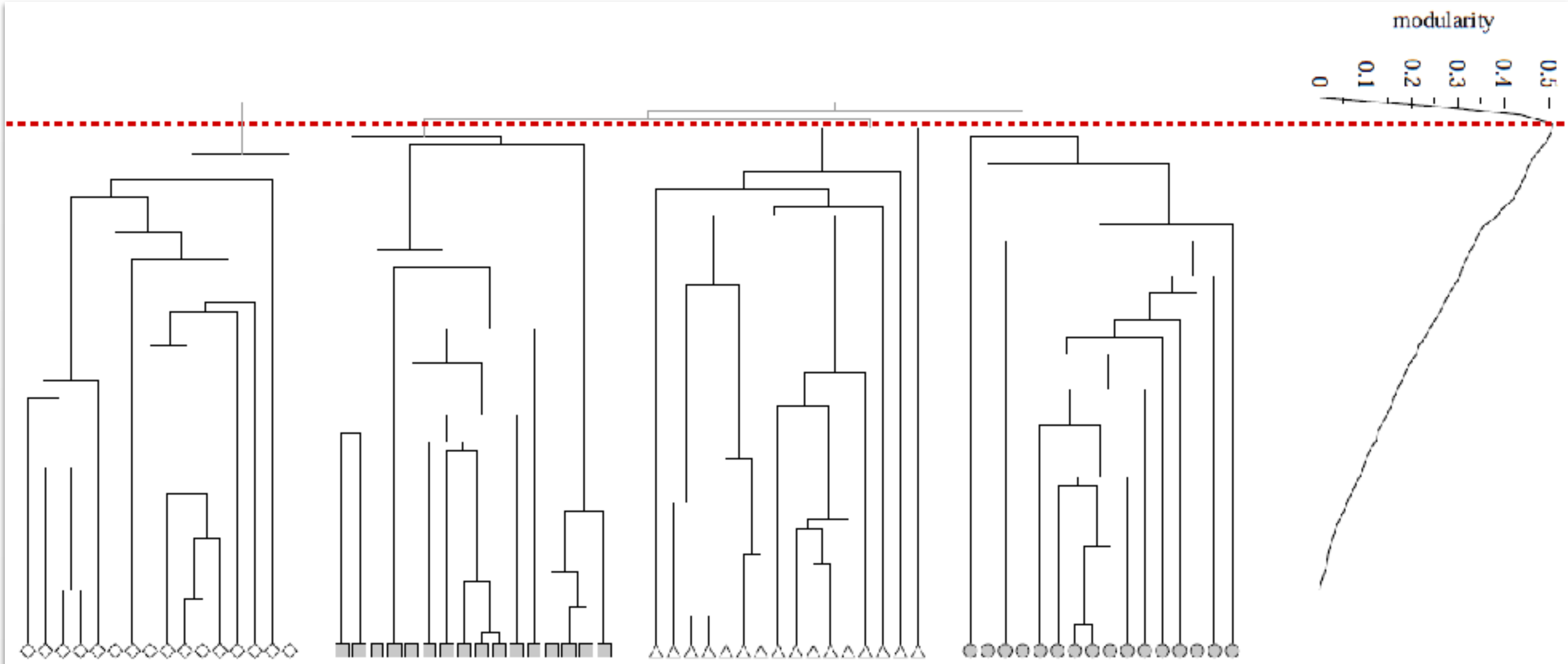
Idea: Compare fraction of edges within module to fraction that would be observed for random connections

$$Q = \frac{1}{2m} \sum_{uv} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \delta(c_u, c_v)$$

- *m:* Number of edges in graph
- A_{uv}: Adjacency matrix (1 if edge exists 0 otherwise)
- k_u: Degree of node *u*
- c_u: Cluster assignment for node u

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Modularity

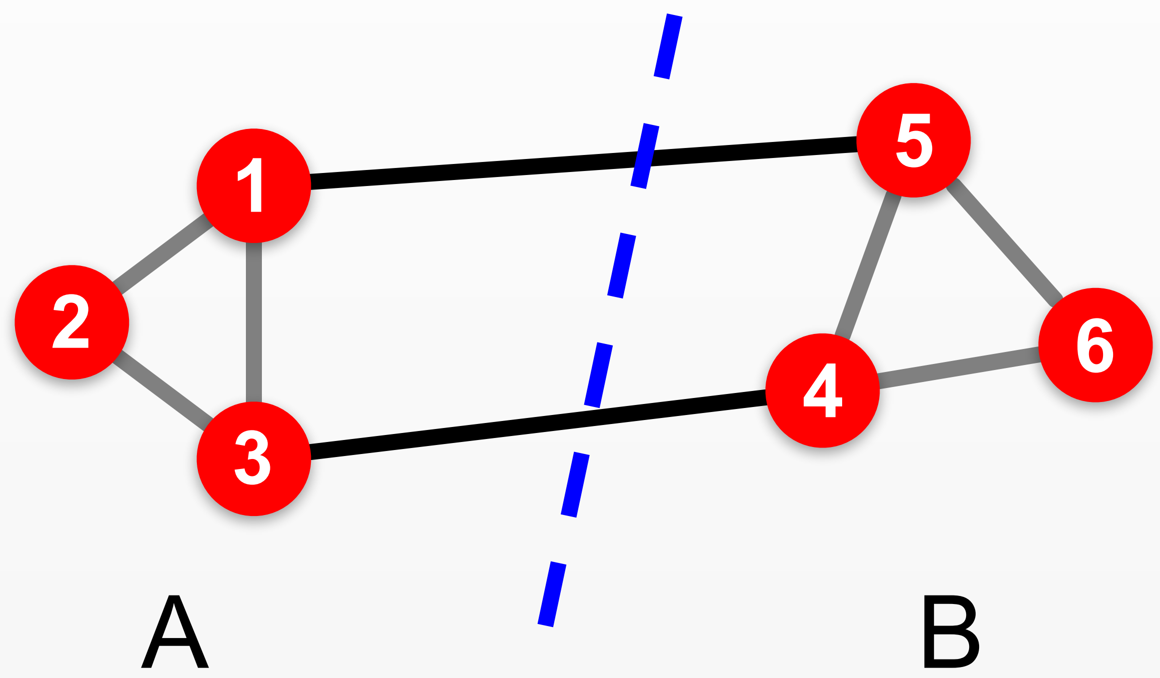


Use modularity to optimize connectivity within modules

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Spectral Clustering

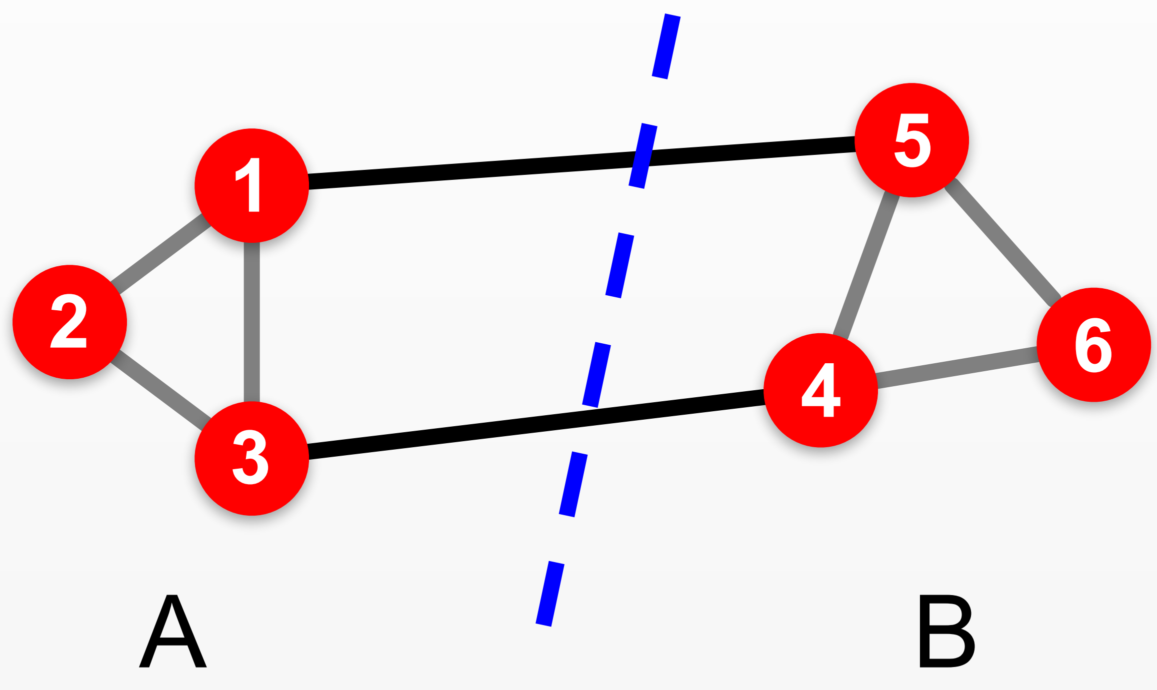
Graph Partitioning



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

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Graph Cuts



Degree

$$d_i = \sum_j \boldsymbol{A}_{ij}$$

Volume

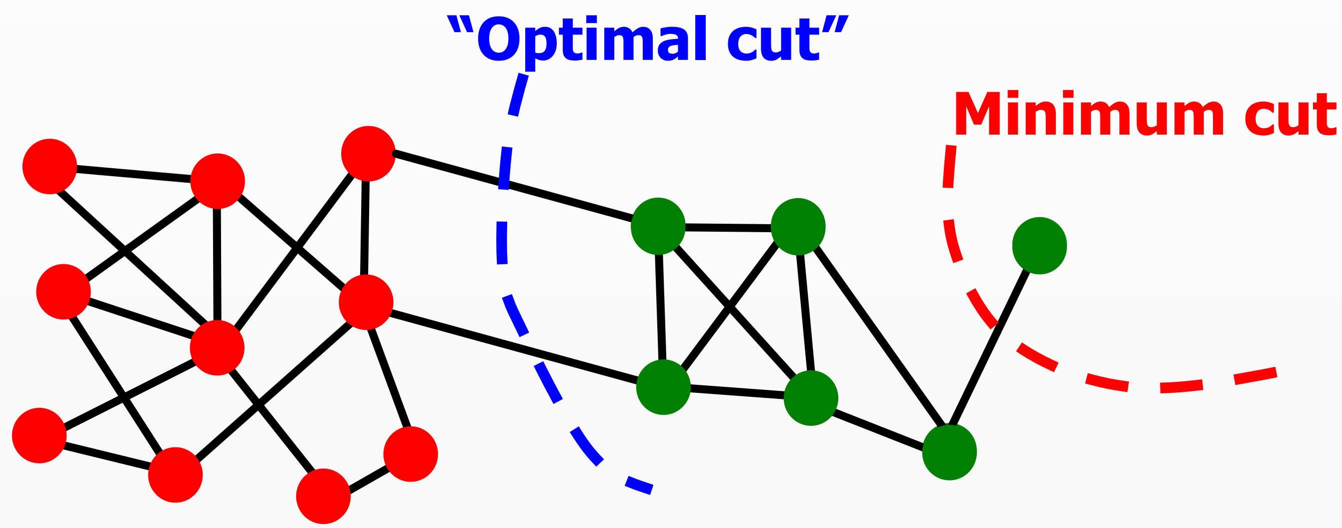
$$\mathrm{vol}(A) = \sum_j d_i$$

Cut

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

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Minimal Cuts

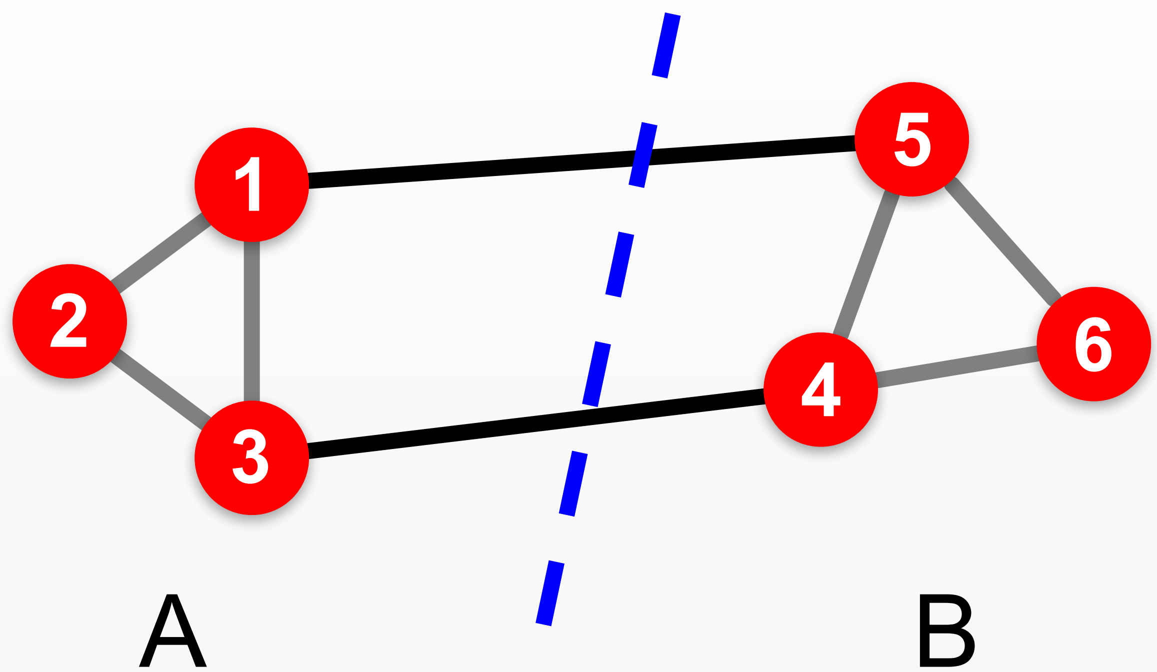


$$\textbf{arg min}_{A,B}\ \textit{cut(A,B)}$$

Problem: minimal cut is not necessarily a good splitting criterion

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Normalized Cuts



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

Degree

$$d_i = \sum_j \boldsymbol{A}_{ij}$$

Volume

$$\text{vol}(A) = \sum_j d_i$$

Cut

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

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Find Optimal Cut [Fiedler'73]

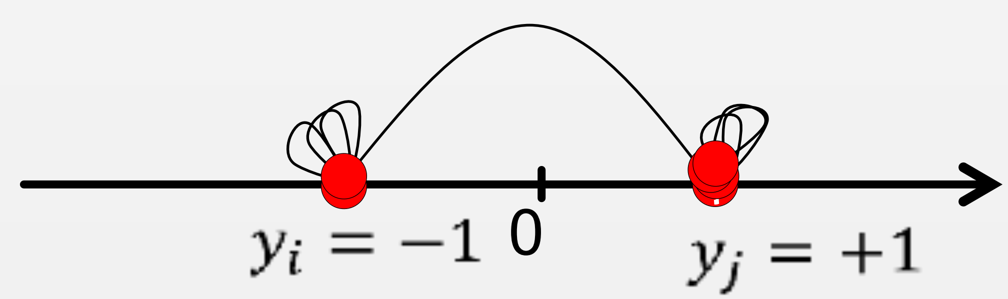
- **Back to finding the optimal cut**
- **Express partition (A,B) as a vector**

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

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

$$y^* = \underset{y \in \{-1,1\}^n}{\operatorname{argmin}} \sum_{(i,j) \in E} (y_i - y_j)^2$$

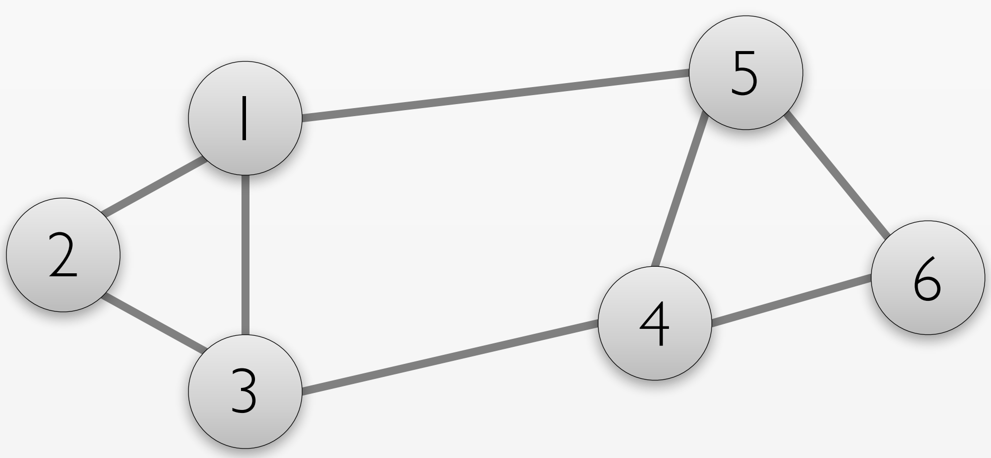
Can't solve exactly. Let's relax y and allow it to take any real value.



(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Matrix Representations

- Adjacency matrix (***A***):
  - ***n*** × ***n*** matrix
  - $A=[a_{ij}]$, $a_{ij}=1$ if edge between node ***i*** and ***j***



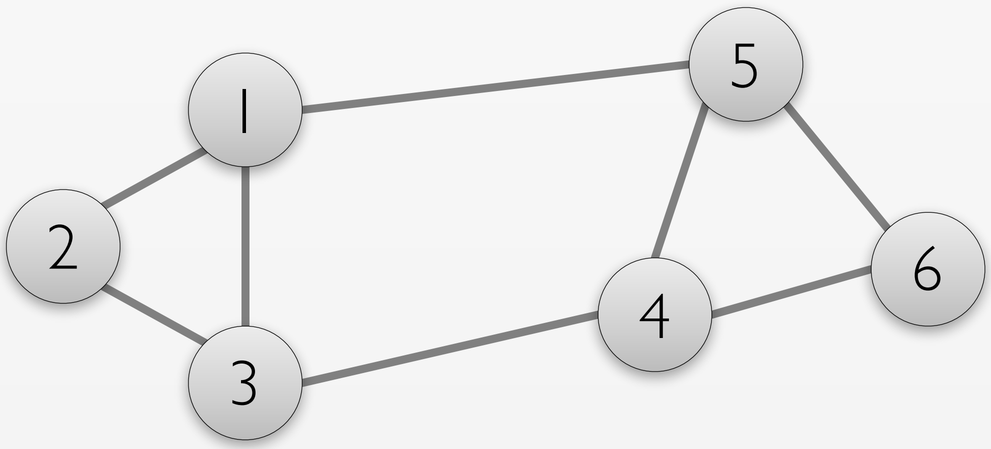
	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0

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

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Matrix Representations

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

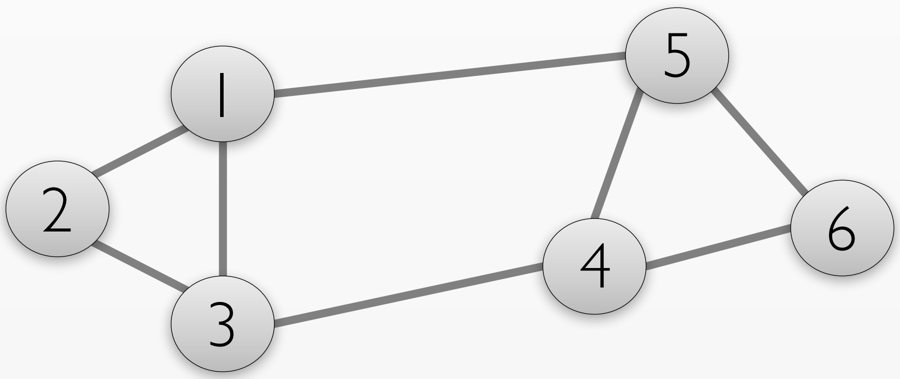


	1	2	3	4	5	6
1	3	0	0	0	0	0
2	0	2	0	0	0	0
3	0	0	3	0	0	0
4	0	0	0	3	0	0
5	0	0	0	0	3	0
6	0	0	0	0	0	2

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

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?**
  - $\boldsymbol{x} = (\mathbf{1}, \dots, \mathbf{1})$ then $\boldsymbol{L} \cdot \boldsymbol{x} = \mathbf{0}$ and so $\boldsymbol{\lambda} = \boldsymbol{\lambda}_{\mathbf{1}} = \mathbf{0}$
- **Important properties:**
  - **Eigenvalues** are non-negative real numbers
  - **Eigenvectors** are real and orthogonal

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Second Eigenvalue

- **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?
  - $x^T L\, x = \sum_{i,j=1}^{n} L_{ij}\, x_i x_j = \sum_{i,j=1}^{n} (D_{ij} - A_{ij})\, x_i x_j$
  - $= \sum_i D_{ii} x_i^2 - \sum_{(i,j)\in E} 2 x_i x_j$
  - $= \sum_{(i,j)\in E} (\underbrace{x_i^2 + x_j^2} - 2 x_i x_j) = \sum_{(i,j)\in E} (x_i - x_j)^2$

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 $x_i^2 + x_j^2$

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

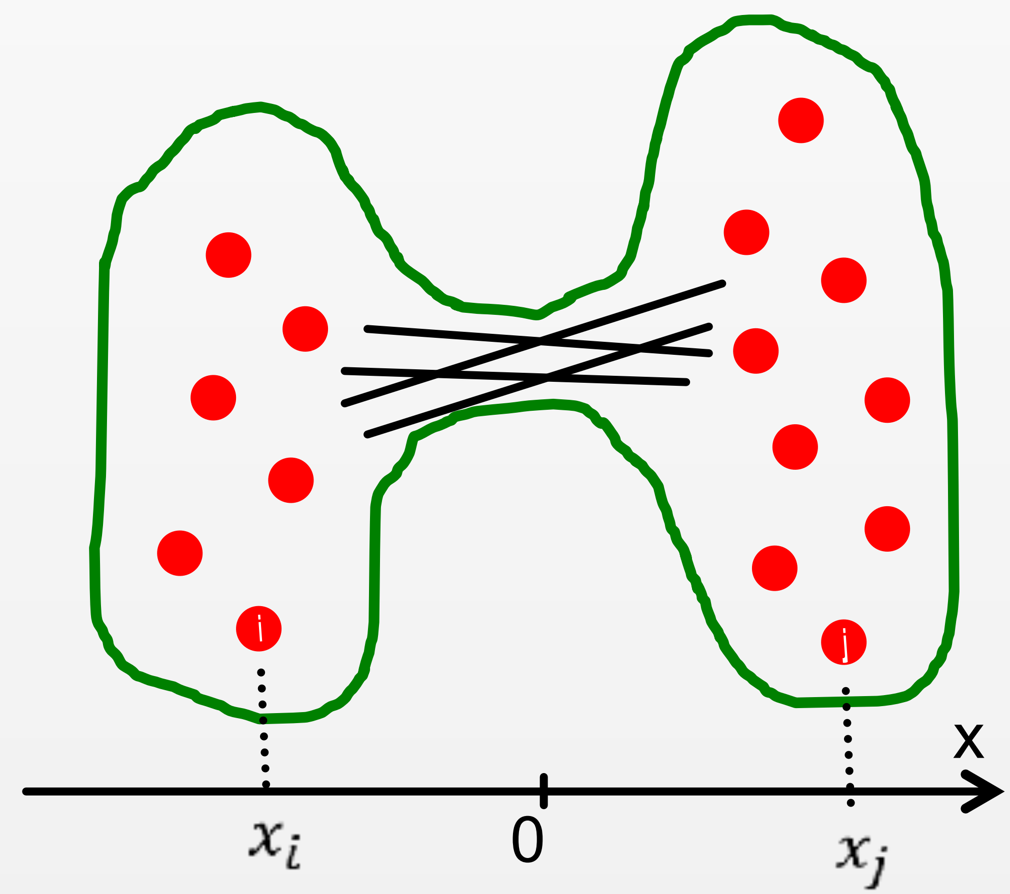
Second Eigenvector of Laplacian

- **What else do we know about x?**
  - x is unit vector: $\sum_i x_i^2 = 1$
  - x is orthogonal to 1st 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)



Balance to minimize

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

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 2nd smallest eigenvalue λ_2 of the Laplacian matrix L
- $\mathbf{x} = \arg\min_{\mathbf{y}} f(y)$: The optimal solution for y is given by the corresponding eigenvector x, referred as the **Fiedler vector**

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

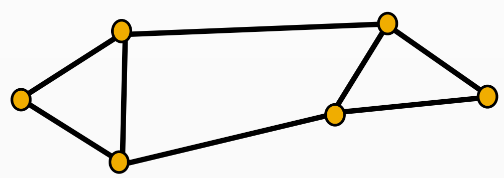
Spectral Clustering Algorithms

- Three basic stages:
  - 1) Pre-processing
    - Construct a matrix representation of the graph
    - More generally, construct similarity matrix
  - 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

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

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

$\lambda =$

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 λ_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?

(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

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

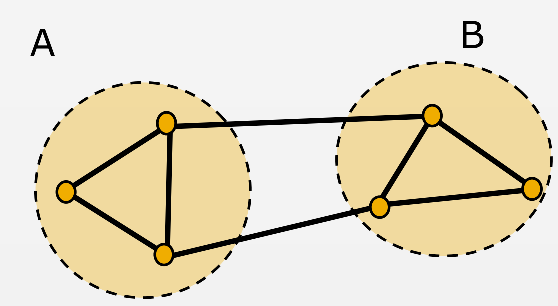
Split at 0:

Cluster A: Positive points

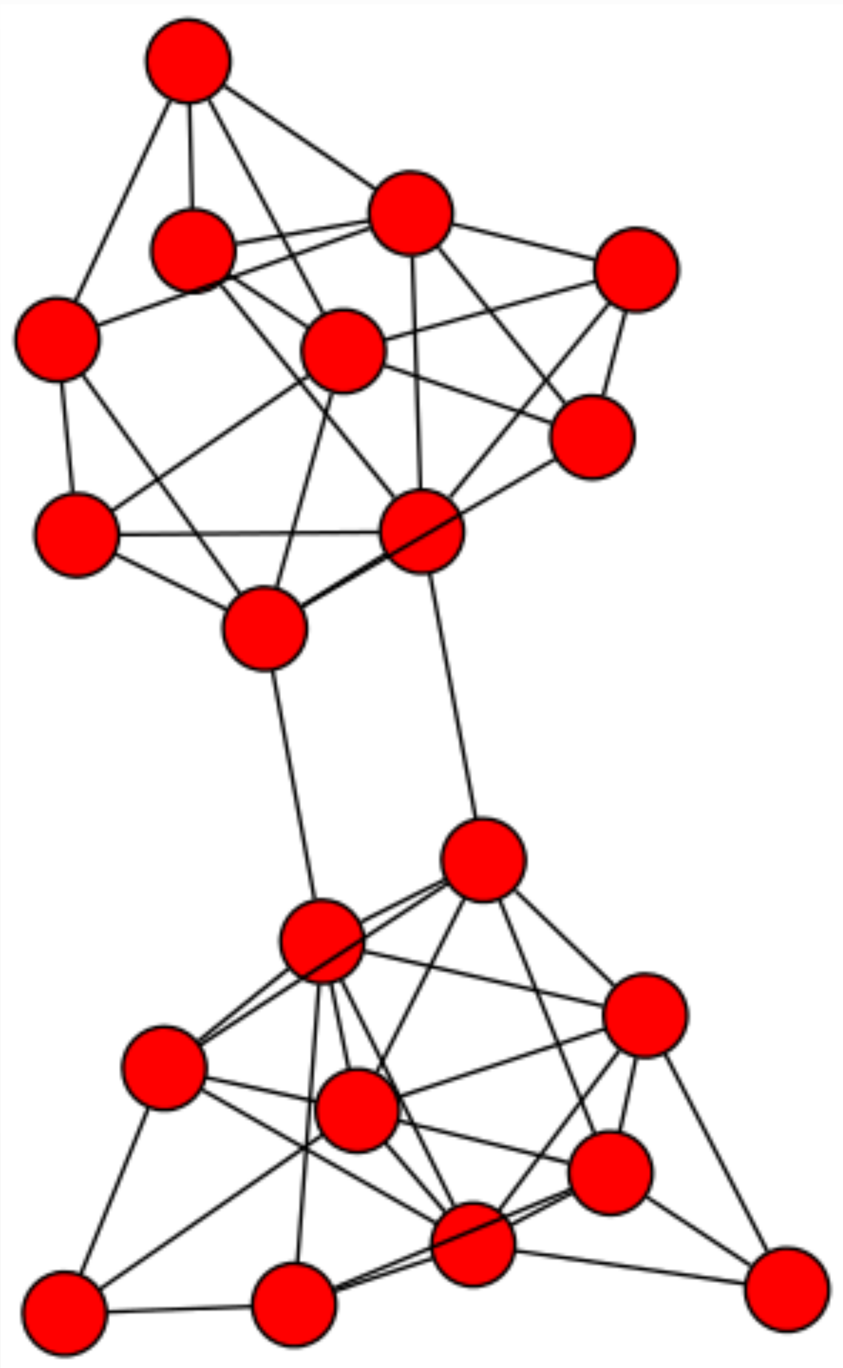
Cluster B: Negative points

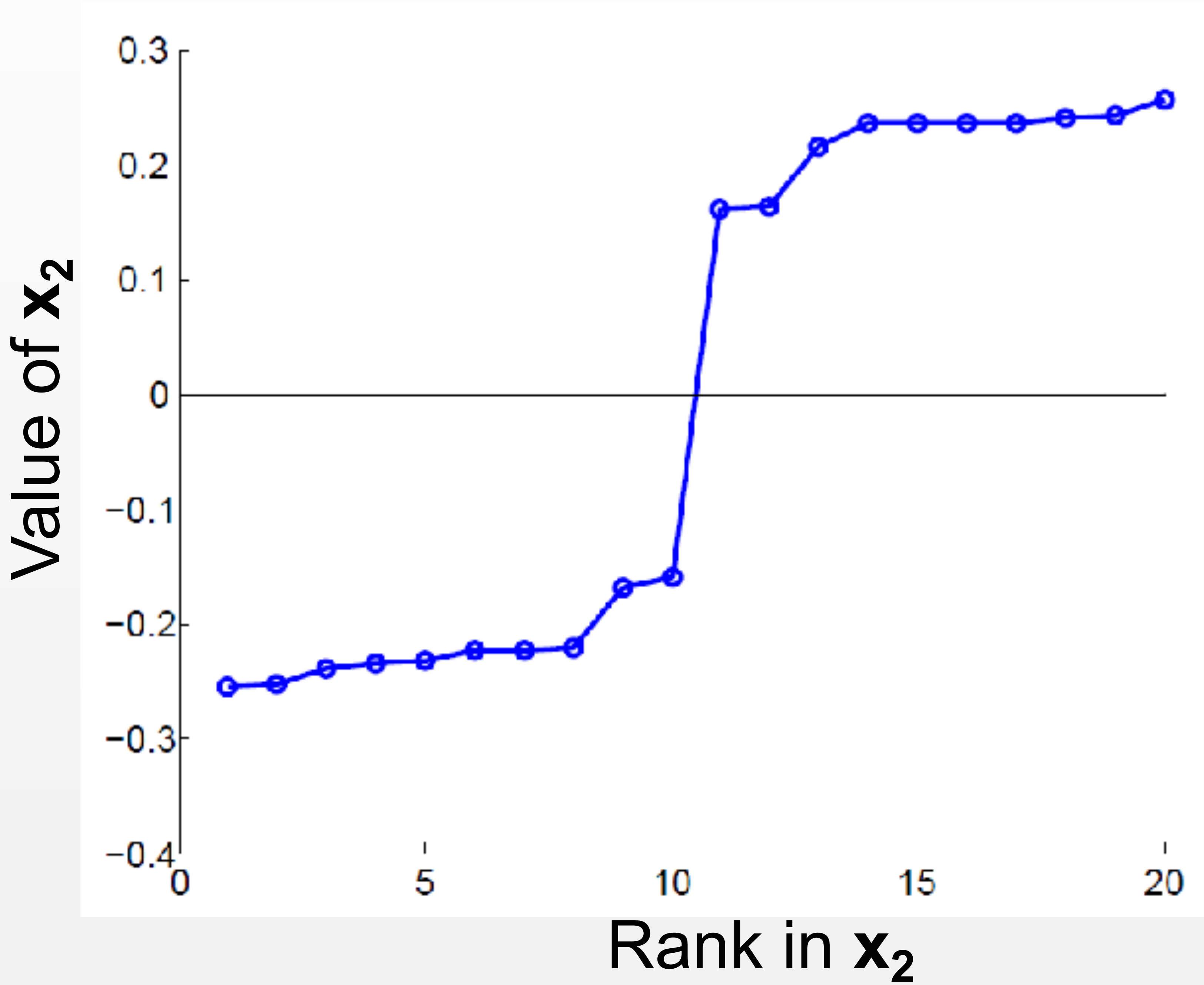
1	0.3
2	0.6
3	0.3

4	-0.3
5	-0.3
6	-0.6



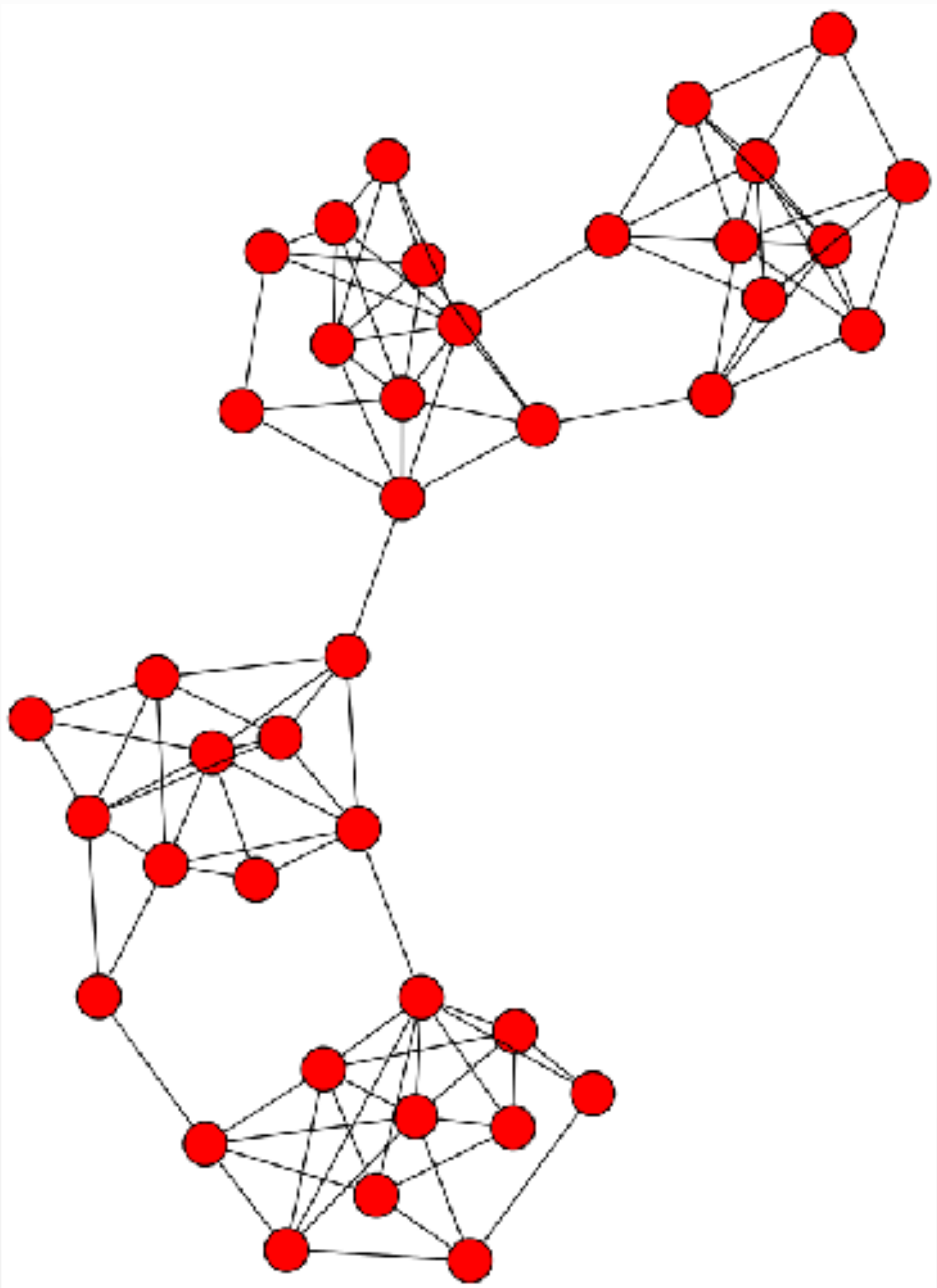
Example: Spectral Partitioning

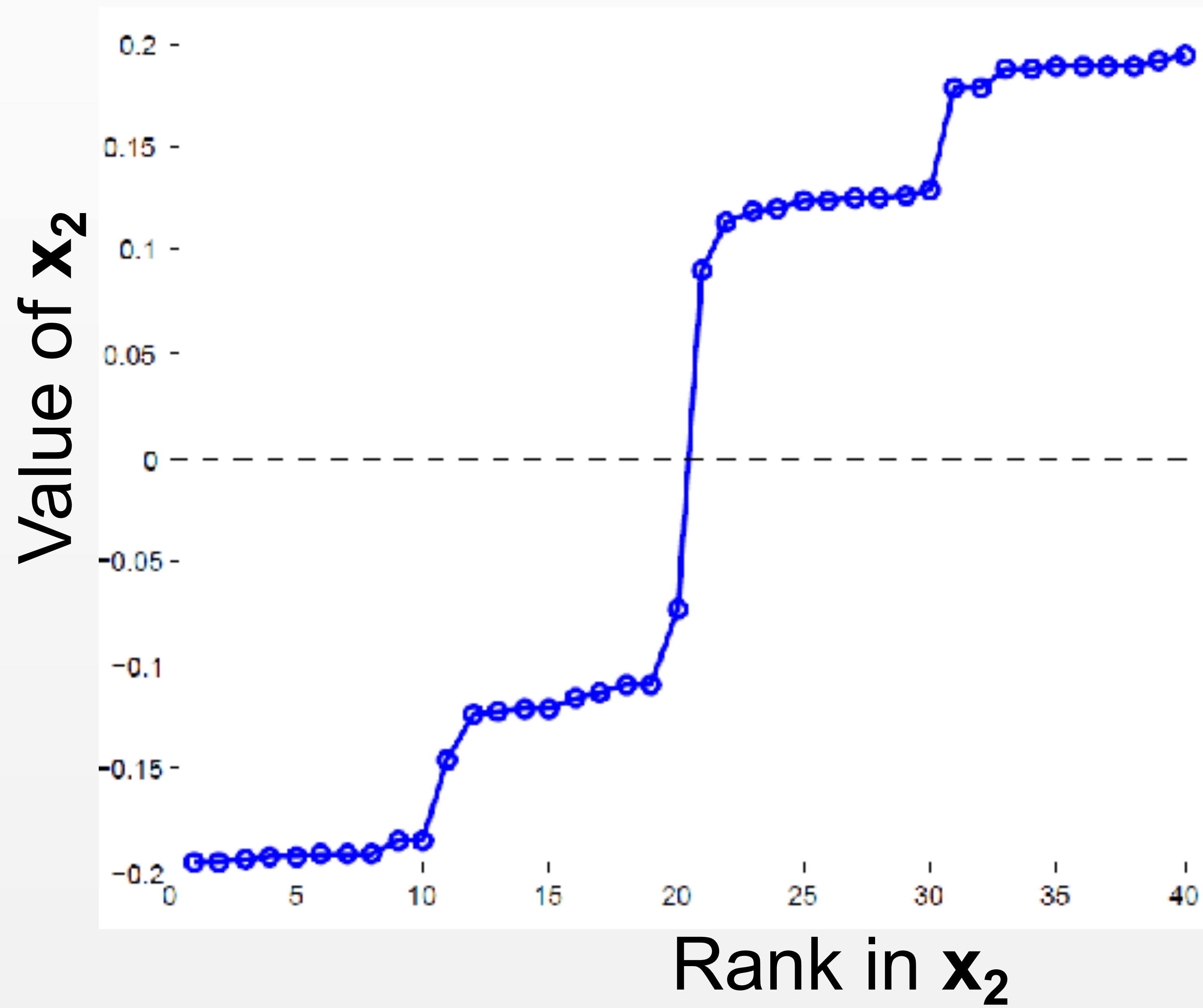




(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

Example: Spectral Partitioning



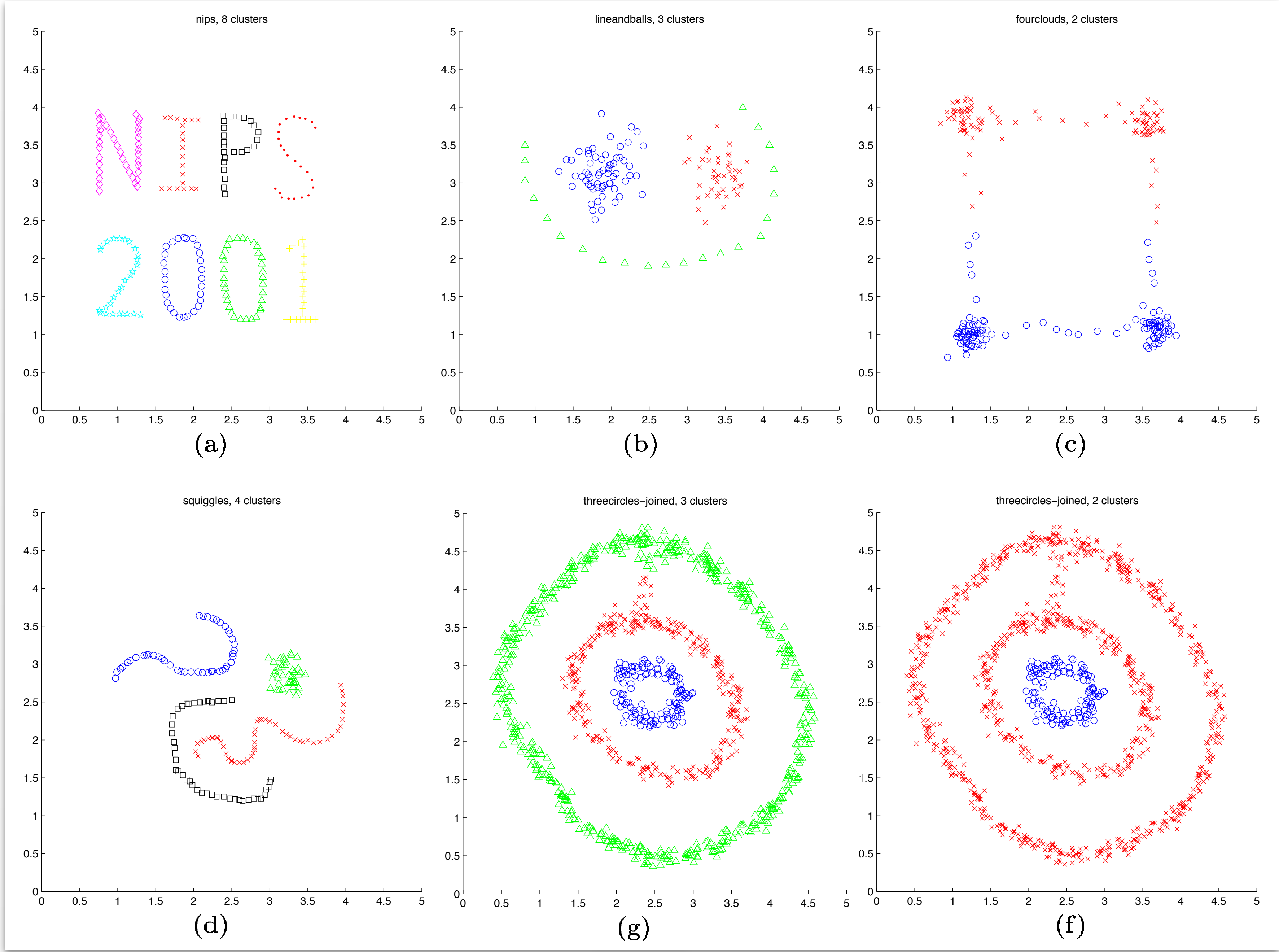


(Adapted from: Mining of Massive Datasets, http://www.mmds.org)

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

Spectral Clustering as General-purpose Method



source: Ng, Jordan and Weiss, NIPS 2001

Define "edge weight" W using some similarity metric (e.g. a kernel function)