Chapter 1: Graph Analysis and Mining

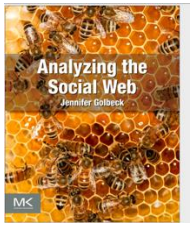
Lynda Tamine-Lechani
lynda.lechani@irit.fr
https://www.irit.fr/~Lynda.Tamine-Lechani/

Graph Analysis and Mining

- Chapter description

  Study the properties and different types of graphs, learn methods for communauty detection and link prediction. The course focuses on social networks (real-world graphs) but the notions and methods presented are mostly applicable to all types of graphs.

- Material



Tutorial:

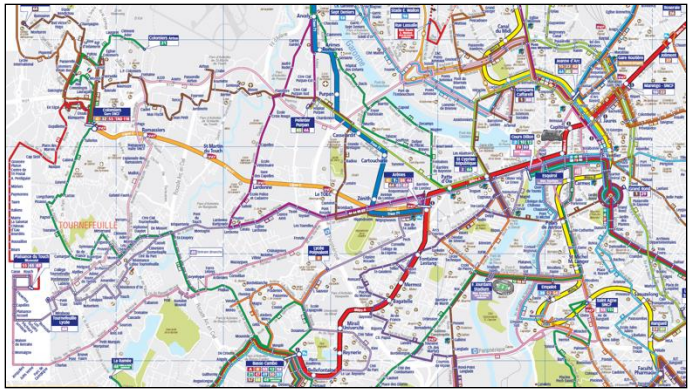
Modelling data with networks KDD'18
(https://ivanbrugere.github.io/kdd2018/)

- Chapter Outline
  1. Basics on graphs
  2. Graph properties
  3. Connectivity
  4. Graph generation models
  5. Communauty detection
  6. Link prediction

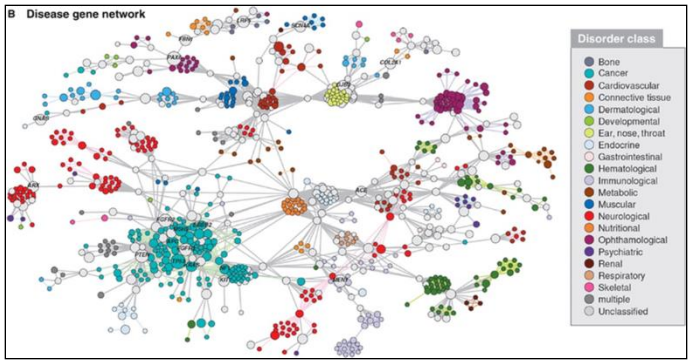
Graphs are everywhere

Transport graph

Source: Tisséo

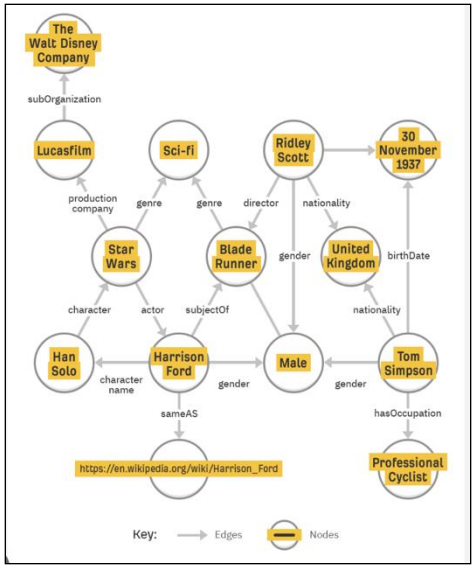


Biology graph

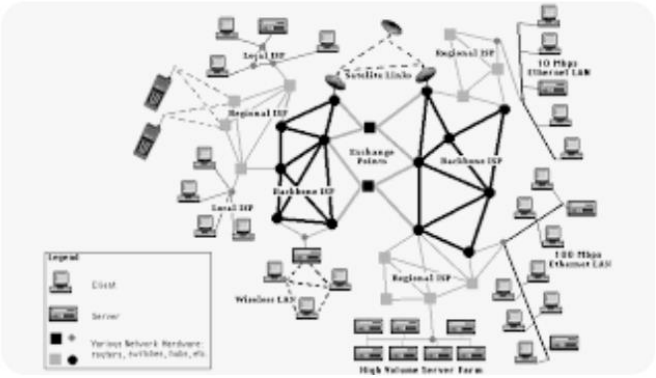


Source: Human disease classification in the postgenomic era: A complex systems approach to human pathobiology, 2007

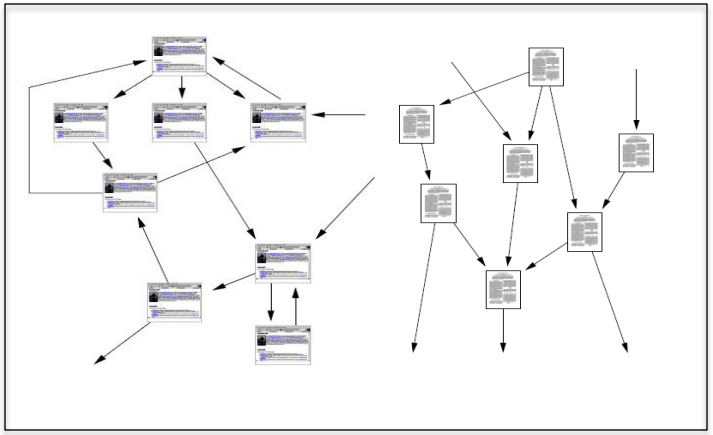
Knowledge graph



Internet graph

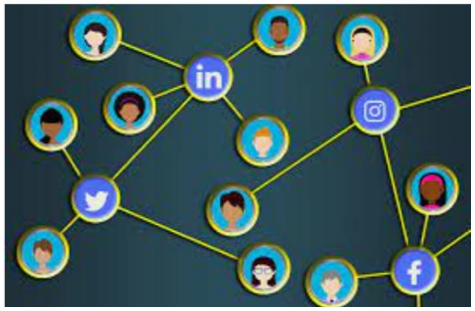


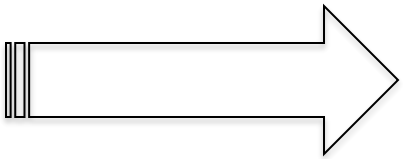
Citation graph

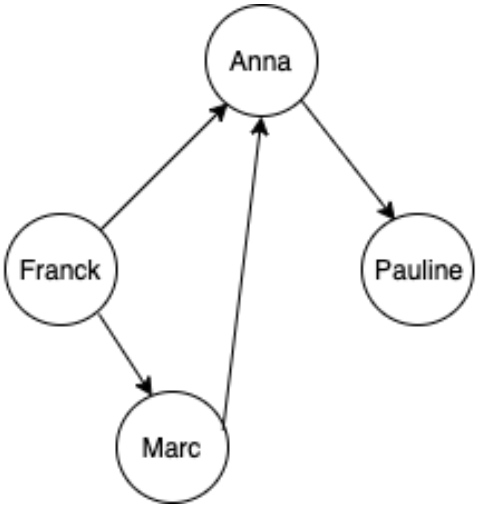


In this course, **focus** on Social Web as a specific graph

- Social Web as a real-world **graph**







Facebok friendships: **nodes** (users or actors); **edges** (undirected friendships)

Twitters fellowships: **nodes** (users or actors); **edges** (directed fellowships)

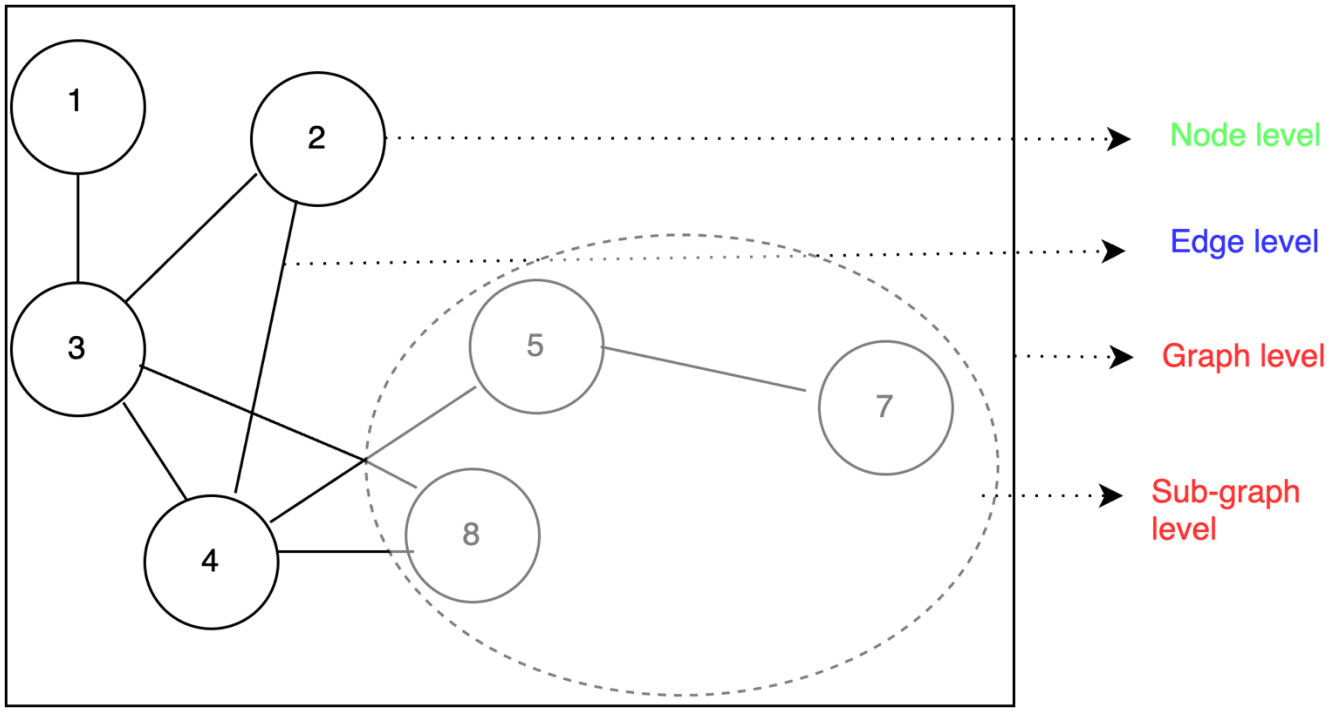
Mobile phone calls: **nodes** (users or actors); **edges** (directed calls)

Bibliographic citations: **nodes** (publications); **edges** (directed citations)

...

Traditional tasks on (social) graphs

- Types of tasks



Traditional tasks on (social) graphs

- **Community detection**
  - Identify linked, cohesive clusters of nodes

    eg., identify cinephils in social networks

- **Link prediction**
  - Predict whether two nodes are (will) be linked, related

    eg., recommend items similar to those clicked,bought

- **Node classification**
  - Predict the type (label) of a given node

    eg., identify active users

- **Graph similarity**
  - Measure how two graphs are similar

    eg., How much similarly behave two communities from FaceBook and Twitter?

Nodes and Edges

- A (social) network is a graph composed of:
  - **Nodes:** also called **nodes**, **users**, **actors**, **vertices**
  - **Edges:** also called **links**, **ties**, **connections, relationships**
    - An edge can have a **direction**
      - ✓ **A directed graph** includes edges with end-points
      - ✓ **An undirected graph** does not includes edges with end-points
    - An edge can have a **weigh** that reflects its **importance**
      - ✓ A **weighted graph** includes edges with different weights (importance)
      - ✓ An **unweigthed graph** includes edges with undifferentiated weights

Special graphs

- Homogeneous graph
  - 1 type of nodes U, 1 type of edges U-U
    - eg., Person-to-person email graph
- Bipartite, Tripartite graph
  - Bipartite: two node types U, V; 1 edge type U-V (no edge U-U, V-V)
    - eg., human-disease graph
  - Tripartite: three node types U, V, W ; 2 edge types U-V, V-W
    - eg., graph of recipients, ingredients compunds graph
- Multi-relational graph
  - 1 type of node, multiple types of edges
    - eg., Social networks with user nodes and multiple relationships: friend, follow, ...
- Heterogeneous graph
  - Multiple types of nodes, multiple types of edges
    - Scientific collaboration : authors, papers as nodes; citation betwen papers, collaboration between authors, authors like papers, etc.

Special graphs

- How to build a graph?
  - Identify the node types, the nodes
  - Identify the edge types and the node types they involve
    - eg., Person-to-person email graph

- Choose the right representation
  - Matrix, tensor, ... : unique vs. multiple representations
  - Bear in mind the target domain, downstream task

- How to make implementation efficient?
  - Characterize graph properties, measures
  - Sample the graph: node sampling, edge sampling, subgraph sampling,...

Nodes and Edges

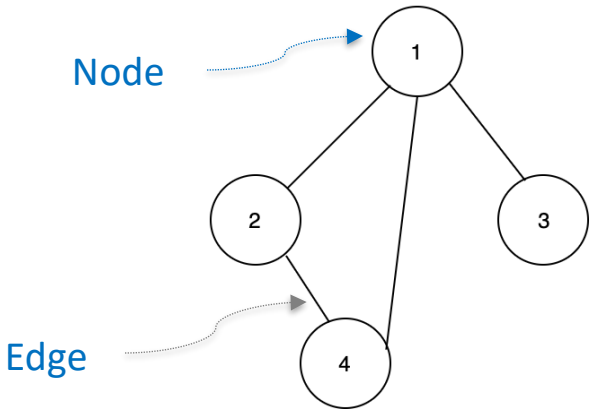
- Let us formalize a homogeneous graph:

$$G = (V, E)$$
$$V = \{v_1, v_2, \ldots, v_n\}, |V| = n$$
$$E = \{e_1, e_2, \ldots, e_m\}, |E| = m$$
$$|E_{max}| = \binom{n}{2} = \frac{n(n-1)}{2}$$

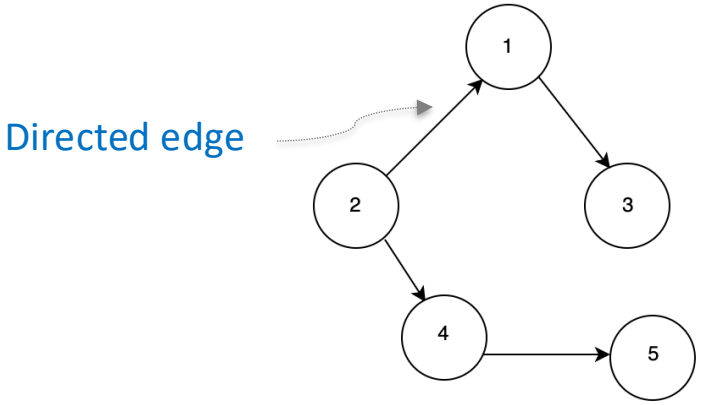
G'=(V',E') is a subgraph of G=(V,E) iif

$$V' \subseteq V$$
$$E' \subseteq (V' \times V') \cap E$$

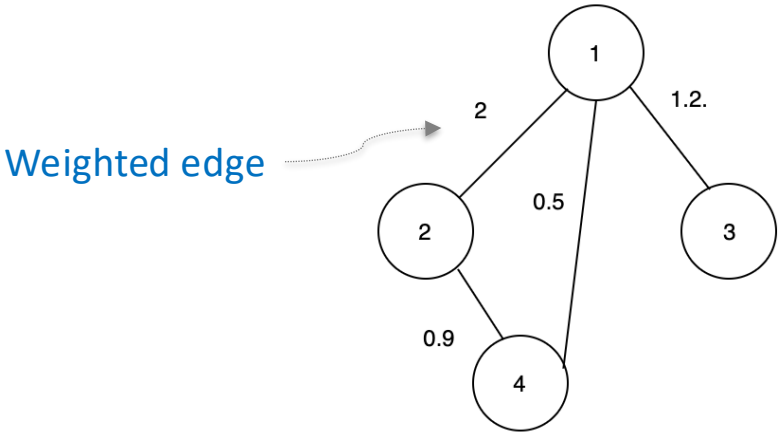
Nodes and Edges



G=(V,E), undirected, unweighted graph
V= {1, 2, 3, 4}, n= 4
E={(1,2), (1, 3), (1,4), (2,4)}, m=4



G=(V,E), undirected, unweighted graph
V= {1, 2, 3, 4, 5}, n= 5
E={(1,3), (2, 1), (2,4), (4,5)}, m=4



G=(V,E, W), undirected, weighted graph
V= {1, 2, 3, 4, 5}, n= 5
E={(1,3), (2, 1), (2,4), (4,5)}, m=4
w(1,2)=2
w(1,3)=1.2
w(1,4)=0.5
W(2,4)=0.9

Types of graphs

- Neighbours and degree
  - The set of nodes connected via an edge: for node *v*, *N(v)* is the set of neighbours.
  - The number of node'neighbours represents its **degree** $d_v = |N(v)|$
  - In directed graphs, we define:

    incoming neighbours $\mathrm{N}_{\mathrm{in}}(\mathrm{v})$, **outgoing neighbours** $\mathrm{N}_{\mathrm{out}}(\mathrm{v})$,

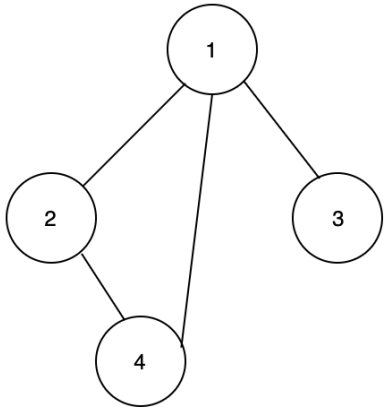
    in-degree d_v^{in} and **out-degree d_v^{out} :**
    - **d_v^{in}** : number of edges pointing to node v
    - **d_v^{out}**: number of edges pointing away from node *v*

Neighberhood

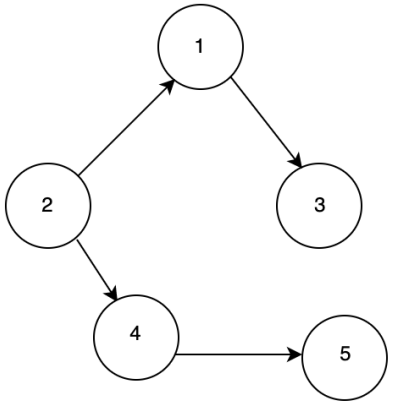
- Neighbours and degree
  - The set of nodes connected via an edge: for node *v*, *N(v)* is the set of neighbours.
  - The number of node'neighbours represents its **degree** $d_v = /N(v)/$
  - In directed graphs, we define:

    incoming neighbours $N_{in}(v)$, **outgoing neighbours** $N_{out}(v)$,

    in-degree d_v^{in} and **out-degree d_v^{out} :**
    - **d_v^{in}** : number of edges pointing to node v
    - **d_v^{out}**: number of edges pointing away from node *v*



$N(1) = \{1, 2, 3,\}$,
$d(1) = 3; d(2) = 2; d(3) = 1; d(4) = 2$



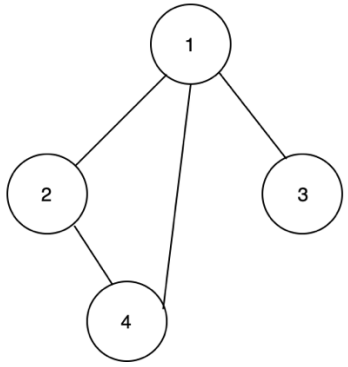
$d_1^{in} = 1, d_1^{out} = 1$
$d_2^{in} = 0, d_1^{out} = 1$
$d_3^{in} = 1, d_3^{out} = 0$

Neighberhood

- Sum degree, average degree

$$\sum_{i=1}^{i=n} d_i = 2m$$

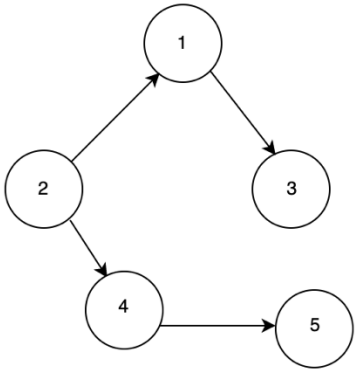
$$\overline{d} = \frac{1}{n}\sum_{i=1}^{i=n} d_i = \frac{2\times m}{n}$$



$\bar{d}$=2

- Degree properties

$$\sum_{i=1}^{i=n} d_i^{in} = \sum_{i=1}^{i=n} d_i^{out}$$



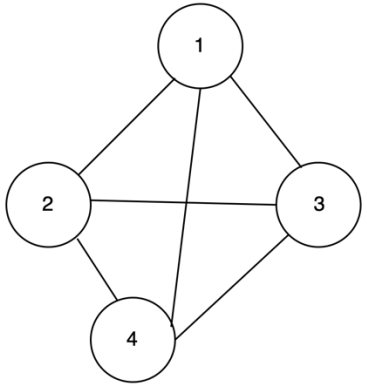
$\sum_{i=1}^{n} d_i^{in}$= 4

$\sum_{i=1}^{n} d_i^{out}$= 4

- Complete graph

A graph with the number of edges $\mathrm{E} = \mathrm{E}_{\max}$ is a **complete graph.**

The **average degree** of a complete graph is *n-1*



$\bar{d}$=3

Classical representations

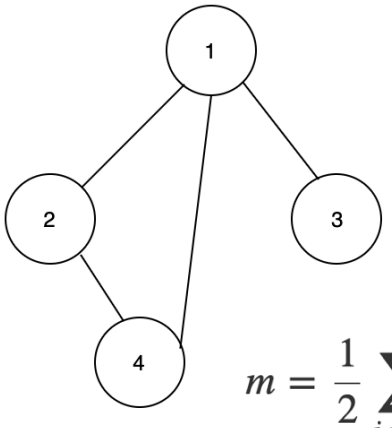
- Beyond visualization, need of mathematical and computational tools for graph representation
  - Adjacency matrix

A_{ij}=1 if **there is an edge** from node i to node j ie. i is **one-hop neighbour** of j

A_{ij} = 0 otherwise

Undirected, unweighted

Friendship



$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

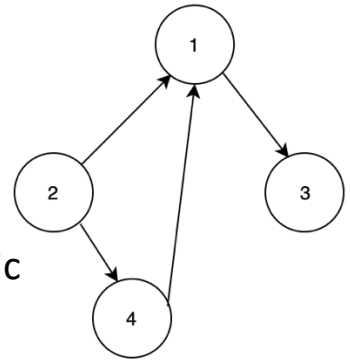
Adjacency matrix is symetric

$A_{ii}=0$, $A_{ij}=A_{ji}$ ($A=A^T$)

$$m = \frac{1}{2}\sum_{i,j=1}^{n} A_{ij} \;, \bar{d} = \frac{2m}{n}$$

Directed, unweighted

Fellowship



$$A^{in} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$A^{out} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

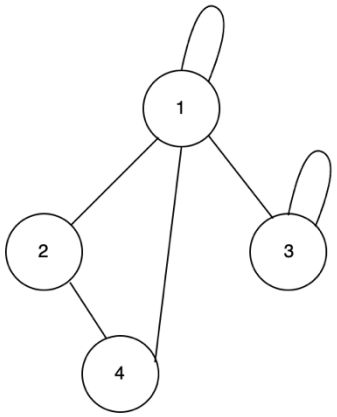
Adjacency matrix is not symetric

$A_{ii}=0$, $\exists\, i,j A_{ij} \neq A_{ji}$ ($A \neq A^T$)

Classical representations

Undirected, self loops

rarely in SN (e.g., Proteins)



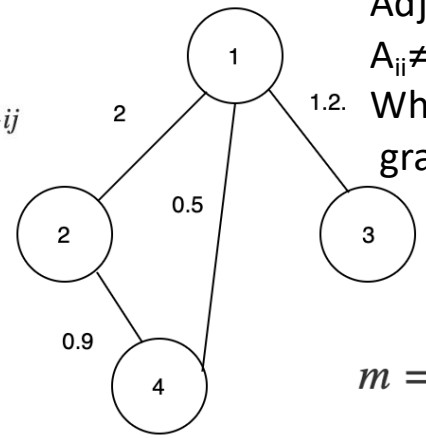
$A_{ii} \neq 0$, $A_{ij} = A_{ji}$

$$m = \frac{1}{2} \sum_{i,j=1, i \neq j}^{n} nonzero(A_{ij}) + \sum_{i,j=1}^{n} A_{ij}$$

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

Undirected, weighted, signed

A weight label for edges (e.g., Review)



Adjacency matrix is symetric

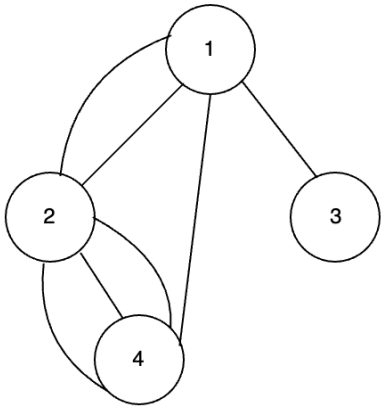
$A_{ii} \neq 0$, $A_{ij} = A_{ji}$ (weight of edge e_{ij})

When Aij =-1/+1/+/-, it is a signed graph

$$m = \frac{1}{2} \sum_{i,j=1}^{n} nonzero(A_{ij})$$

Multi-graph, undirected (with/without self loops):

Multiple edges between nodes (e.g.,*Collaboration)*



Adjacency matrix is symetric

$A_{ii} = 0$, $A_{ij} = A_{ji}$

$$m = \frac{1}{2} \sum_{i,j=1}^{n} nonzero(A_{ij})$$

$$A = \begin{pmatrix} 0 & 2 & 1 & 1 \\ 2 & 0 & 0 & 3 \\ 1 & 0 & 0 & 0 \\ 1 & 3 & 0 & 0 \end{pmatrix}$$

Classical representations

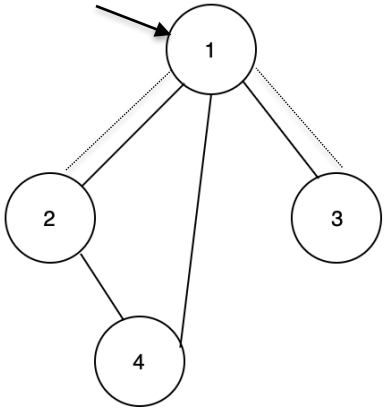
- Adjacencey matrix and neighberhood
  - Number of **common neighbours** between pairs of nodes computed using adjacency matrix
  - Common neighbours define paths of length 2
  - Number of common neighbours between node *i* and node *j*

$$\sum_k A_{i,k}A_{j,k} = A_i \cdot A_j$$

  - More generally

$$A \times A^T = A^2$$

Node 1 is a common neighbour between Nodes 2 and 3



$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

Common neighbours (2,3)= $A_1 X A_4$
=1

Classical representations

- Representation of special graphs (see TD1)
  - Bipartite, Tripartite
    - Transform into homogeneous graphs: 1 projection per edge-type, ie. 1matrix for bi-partite graphs, 2 matric for tripartite graphs

- Multi-relational
  - Transform into homogeneous graphs: 1 adjacency matrix per edge type
  - Binary edge representation between nodes

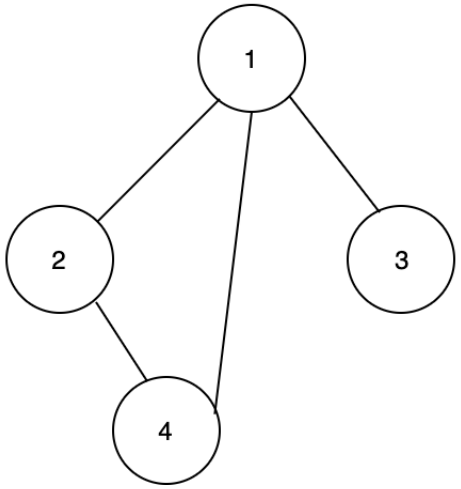
- Heterogeneous
  - Transform into homogeneous graphs: 1 adjacency matrix per pair node type- edge type
  - Use a tensor instead of matrices

Classical representations

Most real-world networks are sparse $E << E_{max}$, so adjacency matrix is filled with zeros (0).

- WWW: $n=319,717$, average degree = 9.65
- LinkedIn: $n= 6,946,668$, average degree = 8.87
- DPLB: $n= 317,080$, average degree = 6.62

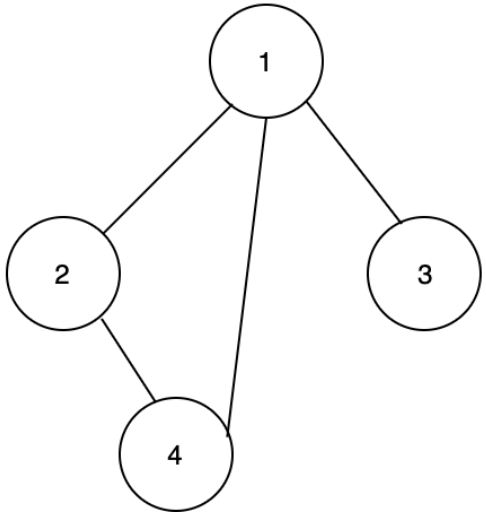
- Adjacency list
  - For each node, list of nodes that are connected to



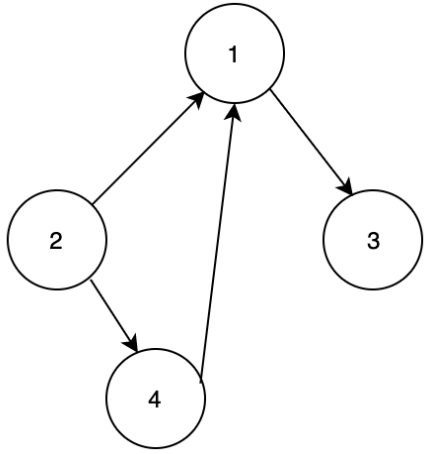
Node	Nodes connected to
1	2, 3, 4
2	1, 4
3	1
4	1, 2

Classical representations

- Edge list
  - List of edges in the graph *e(i,j)*, denotes v_i is connected to v_j
  - Consider graph properties (e.g., directed vs. indirected)



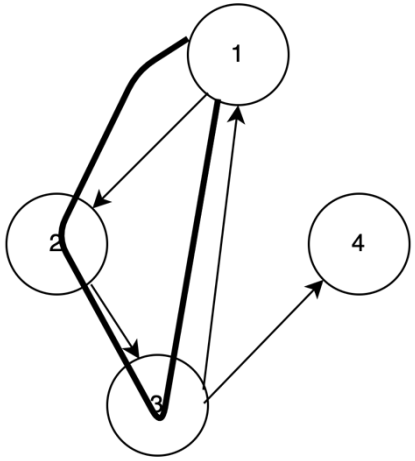
(1,2)
(1,3)
(1,4)
(2,4)



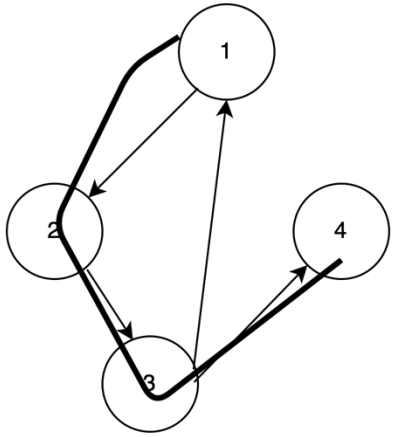
(1,3)
(2,1)
(2,4)
(4,1)

Walk, Path, Diameter

- Walk
  - A walk is a sequence of edges that share end points (**incident edges**) visited one after another
  - The length of a walk is the number of visited edges
- Path
  - A specific walk between nodes *u* and *v* where nodes and edges are distinct (felow the direction if any)
  - The **length l of a path P_{uv} between nodes u, v (distance)** is the number of visited edges in **P_{uv}**
  - **Number of paths** P_{uv} between nodes u, v
    - ✓ l = 1 if $A_{uv} = 1$
    - ✓ l = 2 if $A_{uk}A_{kv}=1$ else $A_{uk}A_{kv}=0$
    - ✓ More generally: l= h,
  - The **shortest path** is the path between two nodes that has the shortest length $p_{uv}^h = [A^h]_{uv}$
  - In a directed graph, a path follows the direction of edges



Length of walk = 3
Special path: cycle



Length of path = 3
$P_{14} = 3$

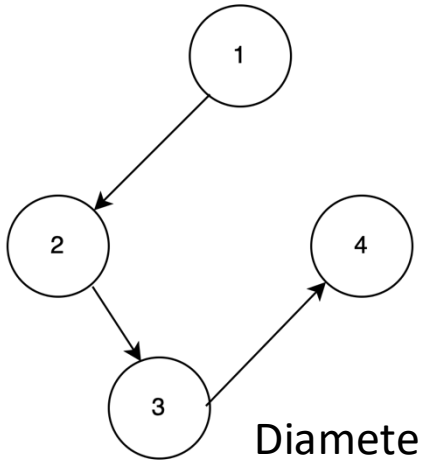
Walk, Path, Diameter

- Random Walk
  - A walk is a sequence of edges that share end points (**incident edges**) visited one after another
  - The next node to be visited is **randomly selected** among the neighbours
  - For all edges starting at node i, we could consider a uniform jump such as:

$$\text{undirected graph: } p_{ij} = \frac{1}{d_i}, j \in N(i)$$
$$\text{directed graph: } p_{ij} = \frac{1}{d_i}, j \in N_{out}(i)$$
$$\sum_j p_{ij} = 1, i, j \; p_{ij} \geq 0$$

Back to graph notions: Component

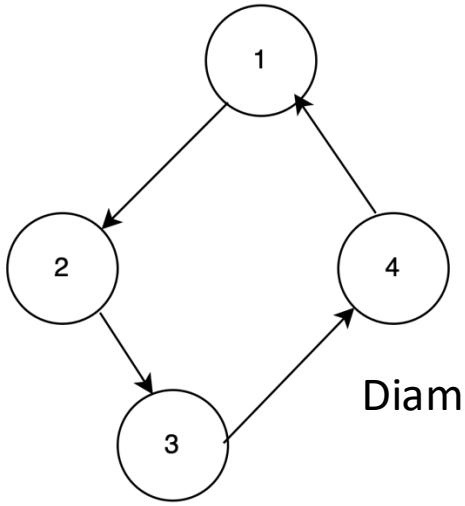
- Diameter
  - Length of the **longest shortest path** between **any pairs** of nodes in the graph



Diameter=3

- Component
  - In an undirected graph, a **component** is a subgraph where there is a **path between each pair of nodes**
  - In a directed graph, a **strong component** is a subgraph where for each pair of nodes (u, v) there is a path in the two directions (u->v, v->u)
  - **Average path length** in a component, strong component

$$\bar{l} = \frac{1}{2|E_{max}|} \sum_{i,j,i \neq j} l_{i,j}$$



Diameter=4

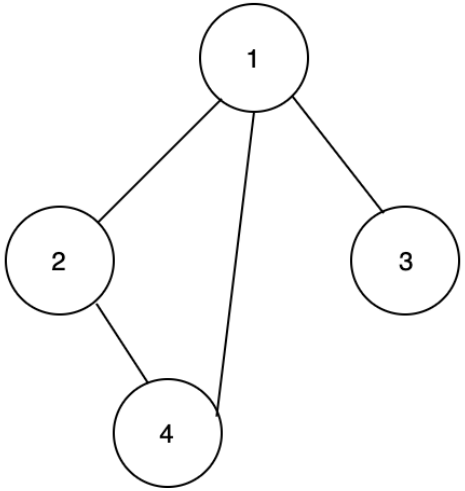
Key properties of graphs

- Degree distribution
  - Probability that a randomly chosen node has degree d
  - Generally, plot histogram

$$\pi(d) = \{d_k, k = 1, \ldots, n\}$$

$$d_k = \frac{n_k}{n}$$

Number of nodes with degree *k*

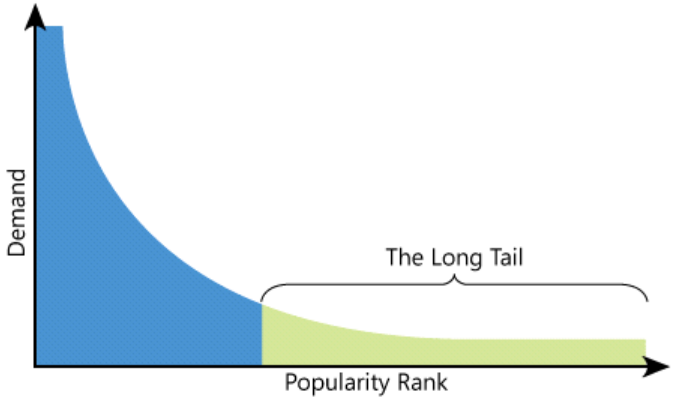


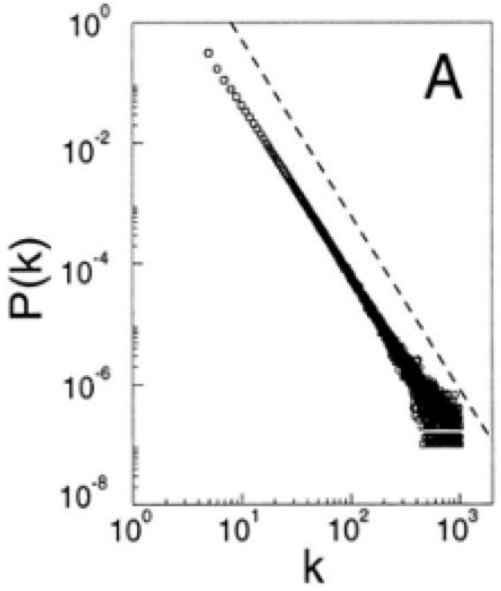
P(1)=1/4; p(2)=1/2; p(3)=1/4

  - In social networks, degree distribution follows a power law: many users with very few neighbours (eg., friends). Phenomena of **long tail**

The total sales volume of unpopular items, taken together, is very significant.

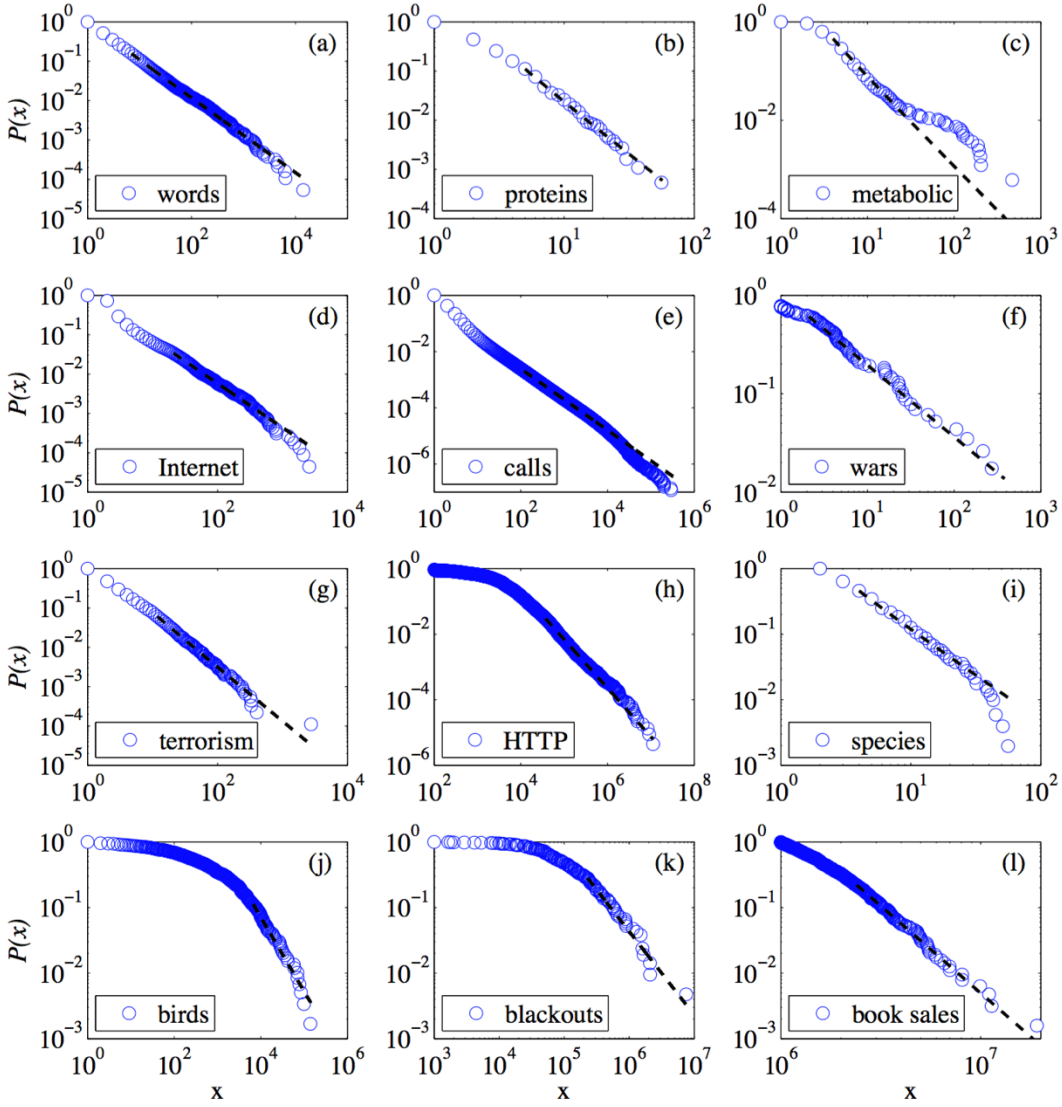
57% of Amazon's sales is from the long tail





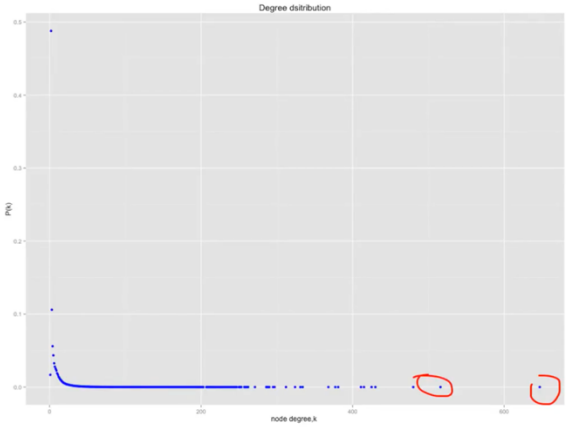
Key properties of graphs

- Power law distribution is almost everywhere



Key properties of graphs

- Check the law degree distribution
  - Pick a popularity measure (see slides below) and compute it for all the graph nodes
  - Compute d_k, the fraction of nodes having degree k
  - Plot a log-log graph, where the x-axis represents $\ln\ k$ and the y-axis represents $\ln\ p_k$
  - If a power-law distribution exists, we should observe a straight line



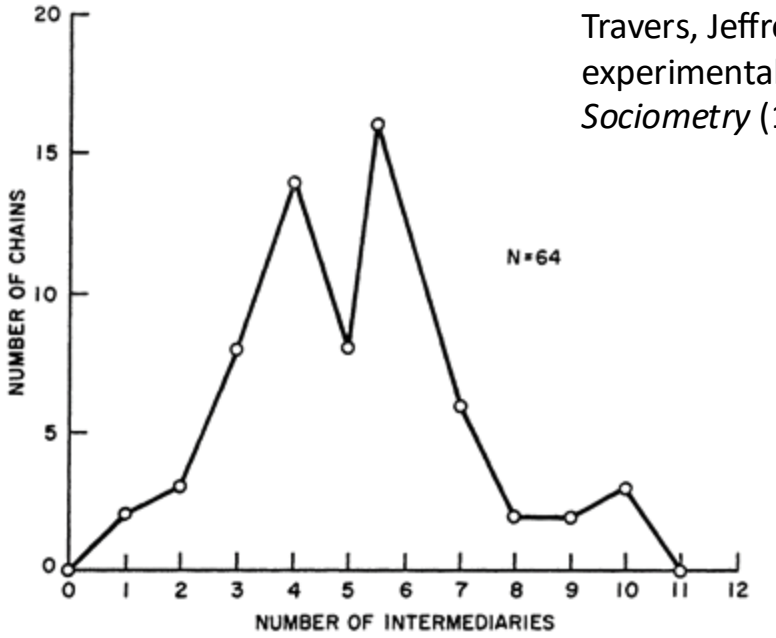
Key properties of graphs

- Six degree separation – related to the "*small world*" hypothesis

  *« Any two people selected randomly from the population are on average separated no more that by **six intermediate connections** »*

  A famous experiment conducted by Travers and Milgram in 1969

  - Subjects (296) were asked to send a chain letter to his acquaintance in order to reach a target person
  - The average path length is around **5.2**

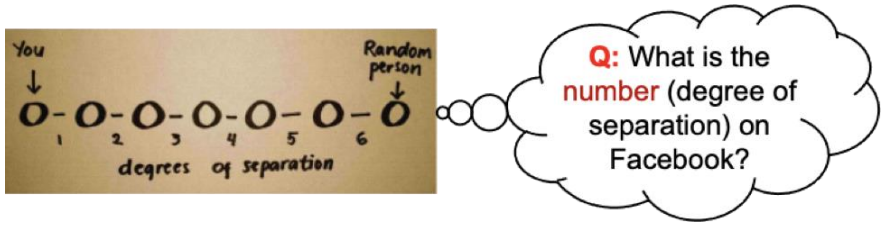


Travers, Jeffrey, and Stanley Milgram. "An experimental study of the small world problem." *Sociometry* (1969): 425-443.

Key properties of graphs

Verified on Facebook

Average length of 3.5 in 2016



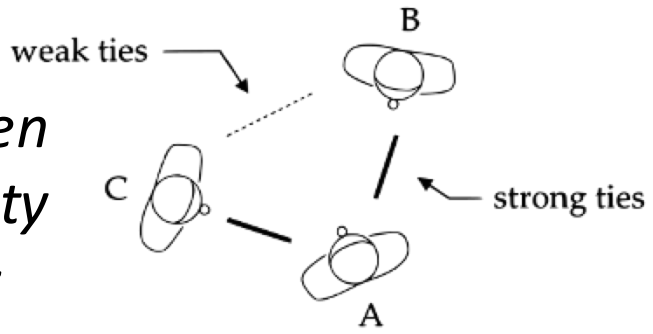
...and other networks

	Network	Type	n	m	ℓ
Social	Film actors	Undirected	449 913	25 516 482	3.48
	Company directors	Undirected	7 673	55 392	4.60
	Math coauthorship	Undirected	253 339	496 489	7.57
	Physics coauthorship	Undirected	52 909	245 300	6.19
	Biology coauthorship	Undirected	1 520 251	11 803 064	4.92
	Telephone call graph	Undirected	47 000 000	80 000 000	
	Email messages	Directed	59 812	86 300	4.95
	Email address books	Directed	16 881	57 029	5.22
	Student dating	Undirected	573	477	16.01
	Sexual contacts	Undirected	2 810		
Information	WWW nd.edu	Directed	269 504	1 497 135	11.27
	WWW AltaVista	Directed	203 549 046	1 466 000 000	16.18
	Citation network	Directed	783 339	6 716 198	
	Roget's Thesaurus	Directed	1 022	5 103	4.87
	Word co-occurrence	Undirected	460 902	16 100 000	
Technological	Internet	Undirected	10 697	31 992	3.31
	Power grid	Undirected	4 941	6 594	18.99
	Train routes	Undirected	587	19 603	2.16
	Software packages	Directed	1 439	1 723	2.42
	Software classes	Directed	1 376	2 213	5.40
	Electronic circuits	Undirected	24 097	53 248	11.05
	Peer-to-peer network	Undirected	880	1 296	4.28
Biological	Metabolic network	Undirected	765	3 686	2.56
	Protein interactions	Undirected	2 115	2 240	6.80
	Marine food web	Directed	134	598	2.05
	Freshwater food web	Directed	92	997	1.90
	Neural network	Directed	307	2 359	3.97

Average path length

Key properties of graphs

- Weak tie hypothesis



Hypothesis: *if A is linked to both B and C, then there is greater-than-chance probability (p>0.5) that B and C are linked to each other*

- in other words: « A friend's friend is also my friend » (a.k.a **homophily**)
- A and B, A and C are linked with strong ties, while B and C are linked with weak ties

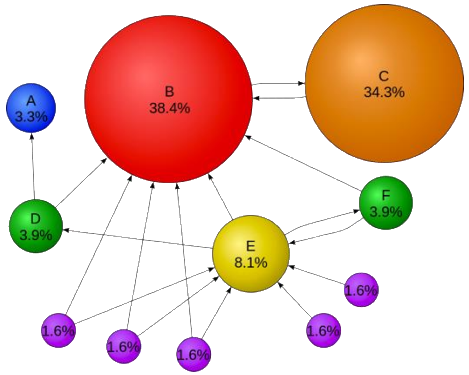
The weak tie hypothesis is used to model information spread (diffusion) through the network

- Information propagates through weak ties rather than strong ties

Measuring connectivity in networks

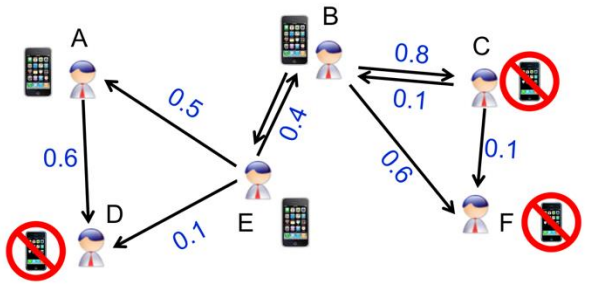
- Crucial need of measures for identifying structural patterns
  - Central nodes

    Influencers, discussers, ...



  - Transitivity, [Reciprocity]

    Types of interaction, influence



  - Similar nodes

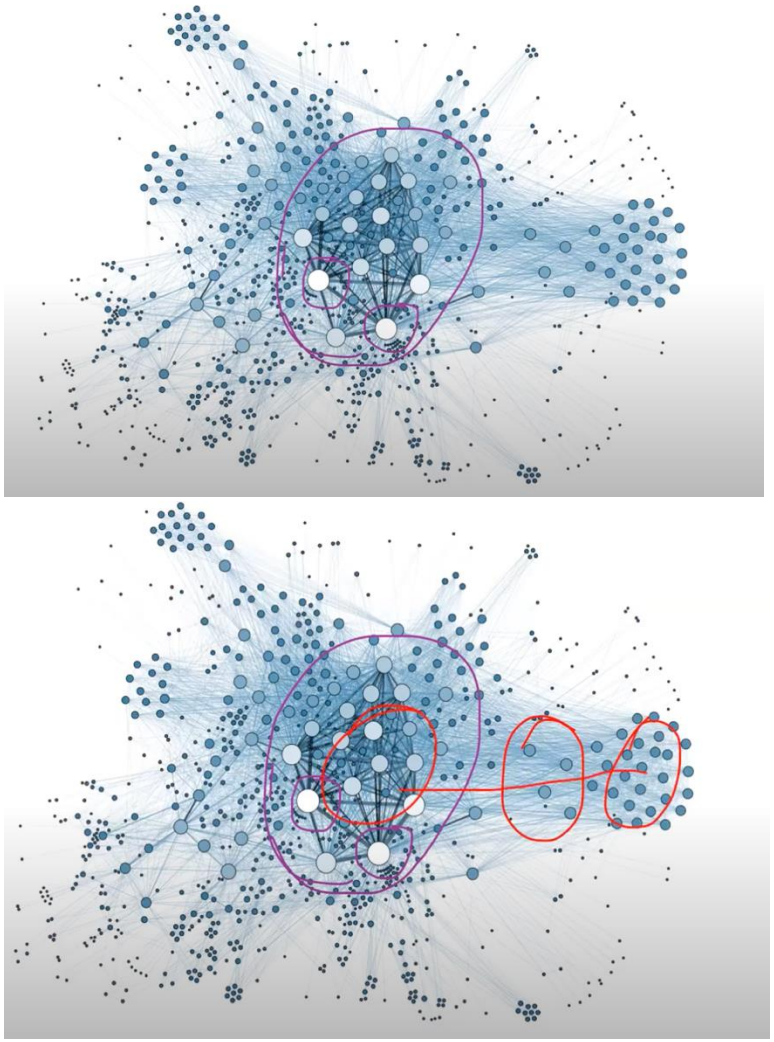
    Nodes that behave similarly

Centrality

- Centrality as a measure of importance
  - Not all nodes are equally important
  - A node is **important** if its connectivity with other nodes in the network makes it **visible**
  - Visibility is not only measured by **direct edges**, but also by **indirect edges** through intermediaries

Knock and Burt distinguished two types of visibility : **centrality** and **prestige**

  - Importance in terms of **those who are connected to the node**
  - Importance in terms of **how is the node connected to other nodes**
  - Importance in terms of **how fast you reach the others**



Importance in terms of those who are connected to the node

- Degree Centrality

  Based on the number of nearest neighbours

$$C_D(i) = \frac{1}{n-1} \sum_j A_{ij}$$

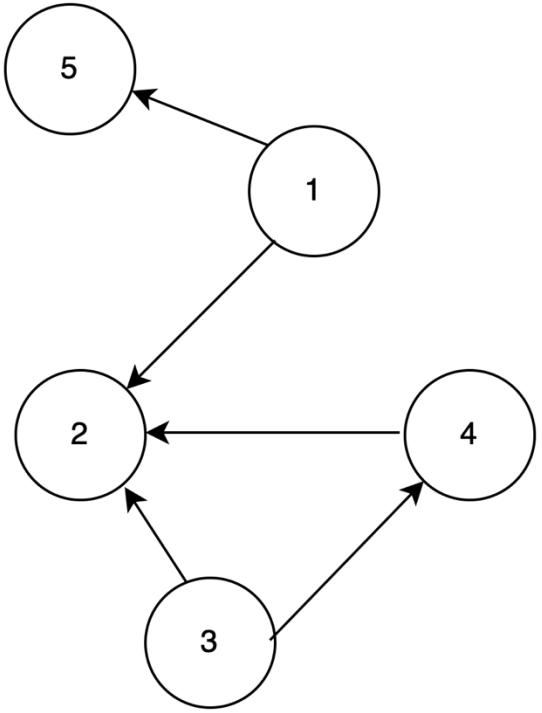
Possible other normalisations: sum(d_i)

Compute C_D(i)-IN (**prestige**), C_D(out) or a combination C_D(i)-IN, CD(out) in the case of directed graphs.

C_D(i)-IN is mostly used.

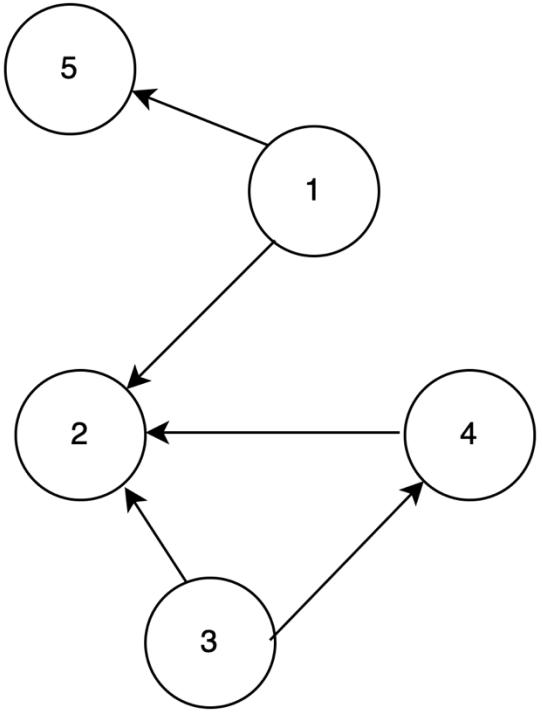
- *High value*: direct contact with many other nodes
- *Low value*: not active, peripheral node

Importance in terms of those who are connected to the node



Node	d_{in}	d_{out}	C_D	Rank
1				
2				
3				
4				
5				

Importance in terms of those who are connected to the node



Node	d_{in}	d_{out}	C_D-IN	Rank
1	0	2	2/(5-1)	2
2	**3**	**0**	**3/4**	**1**
3	0	2	2/4	2
4	1	1	2/4	2
5	1	0	1/4	3

Weaknesses

- ✓ Likely that more than one edge have similar degrees
- ✓ Having more friends is not a salient feature of importance. Importance of friends matters

Importance in terms of those who are connected to the node

- Eigen vector centrality

  The importance of a node depends on the importance of its neighbours (recursive definition). For directed graphs, we can use incoming and outcoming edges.

  We can basically consider that node's centrality is the summation of its neihbour's centralities. Let A be the adjacency matrix:

$$C_E(i) = \frac{1}{\lambda} \sum_{i \neq j} A_{ij} C_E(j)$$

  Matrix formulation: let C_B $(C_B(1), \ldots, C_B(n))$ be the eigen vector centralities of the n nodes in the graph:

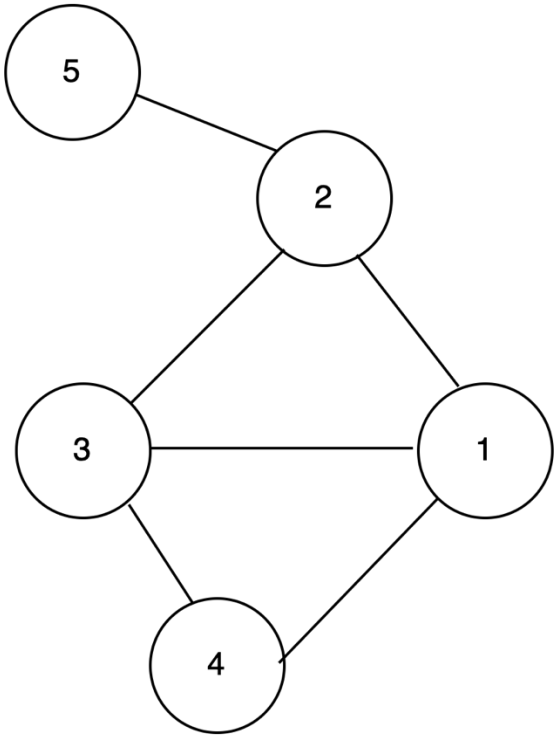
$$\lambda C_E = A^T C_E$$

  So C_B is the eigen vector of matrix A^T

Importance in terms of those who are connected to the node

- How to compute the Eigen Vector?
  - Select the **largest eigen value** λ
  - The corresponding eigenvector of λ is C_e.
  - Based on the Perron-Frobenius theorem, all the components of $C_e$will be positive
  - The components of C_e are the eigenvector centralities for the graph.

Importance in terms of those who are connected to the node



$$A = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$\lambda = (2.68, -1.74, -1.27, 0.33, 0.00)$$

Maximal eigen value

$$C_E = \begin{pmatrix} 0.4119 \\ 0.58.25 \\ 0.4119 \\ 0.5237 \\ 0.2169 \end{pmatrix}$$

Weaknesses

- Problem with directed graphs: centrality only passes throu outgoing edges and in long directed acyclic graphs, centrality becomes zero

Importance in terms of those who are connected to the node

- Katz vector centrality

$$C_K(i) = \alpha \sum_{j=1} A_{ji} C_K(j) + \beta$$

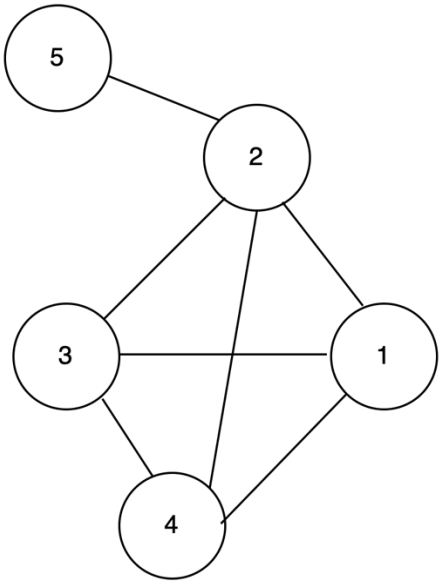
Control

Bias

When α=0, the eigenvector centrality is not considered and all nodes have equal centrality value β. As α increases, the effect of β is reduced.

In practice: $\boldsymbol{\alpha < 1/\lambda}$, where λ is the largest eigenvalue of $\boldsymbol{A^T}$

Importance in terms of those who are connected to the node



$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

$$\lambda = (-1.68, -1.0, -1.0, 0.35, 3.2)$$

Maximal eigen value

$$C_k = \begin{pmatrix} 1.14 \\ 1.31 \\ 1.31 \\ 1.14 \\ 0.85 \end{pmatrix}$$

Weaknesses

- In directed graphs, an important node (high centrality), passes **all** its centrality over **all** of its out-links
- Solution: apply a tax proportional to out-going links

Importance in terms of those who are connected to the node

- Page Rank centrality as a revised version of the Katz eigen vector

$$C_P(i) = \alpha \sum_{j=1} A_{ji} \frac{C_P(j)}{d_j^{out}} + \beta$$

Proportion of the outgoing links

$$C_P(i)^{t+1} = \alpha C_P(i)^{t} + \beta$$

$\alpha < \frac{1}{\lambda}$, *where* λ *is the greatest eigenvalue of* $A^T D^{-1}$

In indirected graphs, $\lambda = 1$, $\alpha < 1$

Importance in terms of those who are connected to the node

- Page Rank alternative approach: commonly known on the web
  - Key idea from Sergey Brin et Larry Page, 1998: each web page implicitly transfers some importance to linked pages throughout outgoing links. A web page is important as many important pages point to it. Becomes the ***Google score.***
  - Formally

    Let A be the adjacency matrix of graph $G(V,E)$, $|E|=n$

    Let W be the transition matrix of graph G, where w_{ij} is the transition probability from node i to node j $w_{ij}=A_{ij}/d_i$

    W is a column stochastic matrix

    How to compute $PR(i)$?

    ✓Initialize the n-dimension vector PR^0 = (1/n, ...1/n)

    ✓ Iteratively compute

$$PR(i)^{t+1} = \sum_{j \to i} \frac{PR(j)^t}{d_j^{out}}$$

PageRank node i at step t+1

PageRank node i at step t

All nodes linking to node i

Out degree of node j

Importance in terms of those who are connected to the node

- Matrix formulation

The transposed transition matrix of graph G

$$PR^{t+1} = W^T \times PR^t$$

The n-dimensionnal PageRank vector at step t+1

The n-dimensionnal PageRank vector at step t

- Extension to a random surfer model

Compute the pagerank of page p as the probability that a random surfer starts at a random page and ends at page p.

$$PR^{t+1} = \frac{(1-d)}{n} + d \times W^T \times PR^t$$

Probability that the random surfer "teleports" and not uses the outlinks

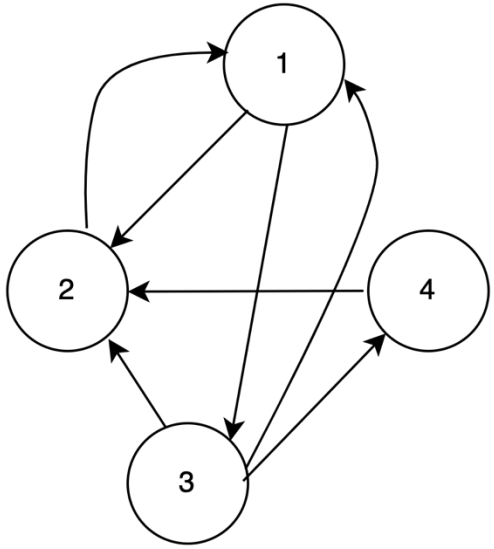
Probability that the random surfer surfs throughout the outgoing links

d=1, we have a "naïve" pagerank, d=0, every page has a pagerank 1/n

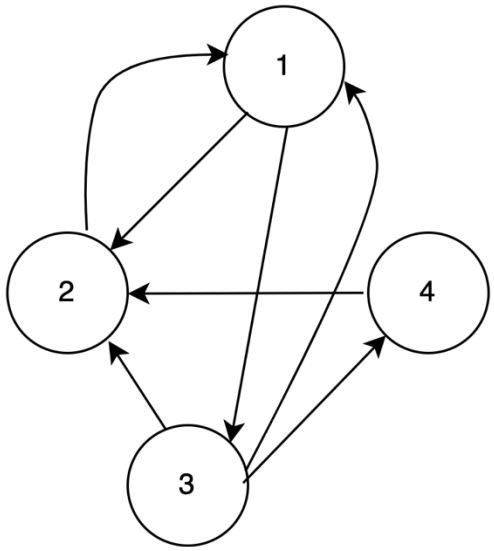
Generally $d=0.85$

After a small number of iterations, the solution will converge to the true PageRank solution.

Importance in terms of those who are connected to the node



Importance in terms of those who are connected to the node



$$W = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 \\ 1 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 1/3 \\ 0 & 1 & 0 & 0 \end{pmatrix} \qquad W^T = \begin{pmatrix} 0 & 1 & 1/3 & 0 \\ 1/2 & 0 & 1/3 & 1 \\ 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 0 \end{pmatrix}$$

Outlinks of node 1

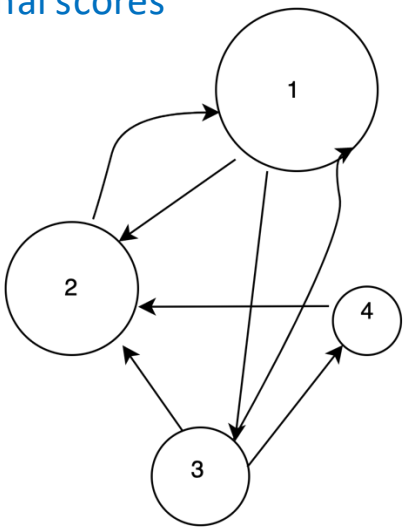
$PR^0 = (0.25, 0.25, 0.25, 0.25)$

$$PR^1 = W^T PR^0$$

$$PR^1 = \begin{pmatrix} 0.33 \\ 0.46 \\ 0.13 \\ 0.08 \end{pmatrix} \qquad PR^2 = \begin{pmatrix} 0.50 \\ 0.29 \\ 0.17 \\ 0.04 \end{pmatrix} \qquad PR^3 = \begin{pmatrix} 0.35 \\ 0.35 \\ 0.25 \\ 0.06 \end{pmatrix}$$

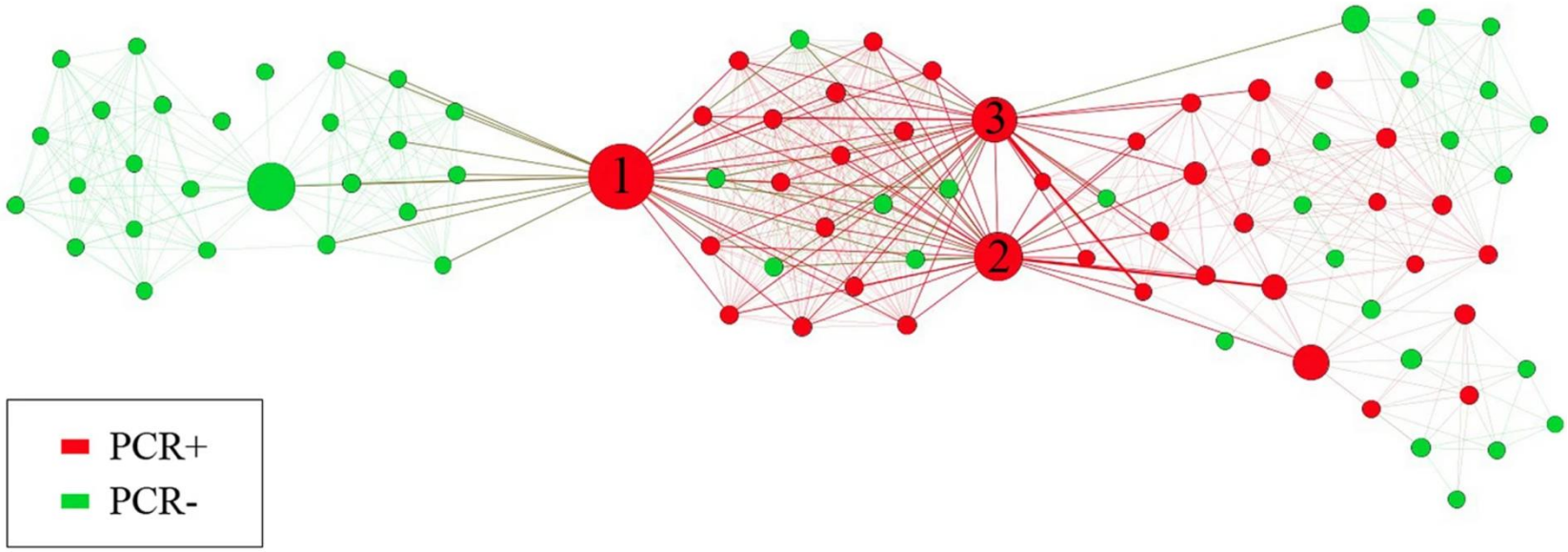
$$PR^{16} = \begin{pmatrix} 0.40 \\ 0.33 \\ 0.20 \\ 0.07 \end{pmatrix} \qquad PR^{17} = \begin{pmatrix} 0.40 \\ 0.34 \\ 0.20 \\ 0.07 \end{pmatrix}$$

Convergence to final scores



Importance in terms of those who are connected to the node

- Betweeness centrality



The network shown under the Atlas 2 distribution highlighting the 3 most important key players in the network.

Source: Nature (2021).
A case study of university student networks and the COVID-19 pandemic using a social network analysis approach in halls of residence
José Alberto Benítez-Andrades, Tania Fernández-Villa, Carmen Benavides, Andrea Gayubo-Serrenes, Vicente Martín & Pilar Marqués-Sánchez

Importance in terms of how is the node connected to other nodes

- Betweeness centrality

Measures how much a node has control over all possible pairs of nodes. If it is part of the shortest paths of many pairs, it will be central

For undirected graphs

$$C_B(i) = \frac{2}{(n-1)(n-2)} \sum_{i \neq j \neq k} \frac{p_{jk}(i)}{p_{jk}}$$

Number of **shortest paths** between nodes j and k **that pass through node i**

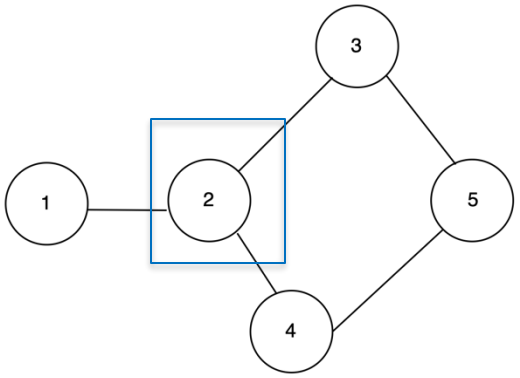
Number of **shortest paths** between nodes j and k

For directed graphs

$$C_B(i) = \frac{1}{(n-1)(n-2)} \sum_{i \neq j \neq k} \frac{p_{jk}(i)}{p_{jk}}$$

Maximal number of **node pairs not including node i**

Importance in terms of those who are connected to the node



$C_B(1)=$
$C_B(2)=$
$C_B(3)=$
$C_B(4)=$
$C_B(5)=$

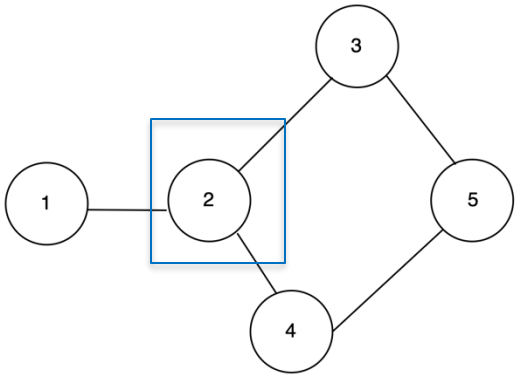
Number of shortest paths

	1	2	3	4	5
1					
2					
3					
4					
5					

Node	C_B	Rank
1		
2		
3		
4		
5		

How to compute shortest paths?
Use Djikstra algorithm (See your course on graph theory)
Better solution: Brandes algorithm (See your course on graph theory)

Importance in terms of those who are connected to the node



Number of shortest paths

	1	2	3	4	5
1	-	1	1	1	2
2	1	-	1	1	2
3	1	1	-	2	1
4	1	1	2	-	1
5	2	2	1	1	-

n= 5 ; undirected graph, scale with 2/ (5-1)(5-2) = 1/6

$C_B(1)$=0 – no shortest path through node 1

(1,3) (1,4) (1,5) (3,4) (3,5) (4,5)

$C_B(2)$= 1/6 x (1/1) + (1/1) + (2/2) + (1/2) + 0 + 0

(1,2) (1,4) (1,5) (2,4) (2,5) (4,5)

$C_B(3)$= 1/6x (0+ 0+ 1/2 + 0 + 1/2 + 0)

$C_B(4)$= 1/6 x ?

$C_B(5)$= 1/6 x ?

Node	C_B	Rank
1	0	
2	1/6* 7/2	
3	1/6	
4	?	
5	?	

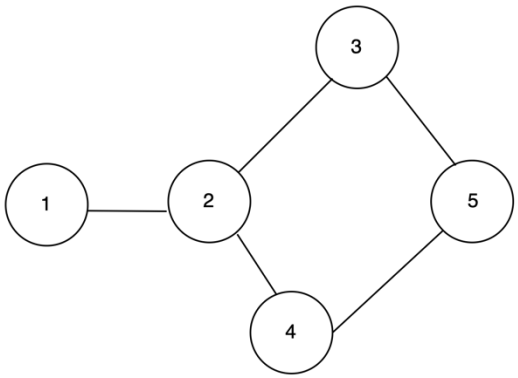
Importance in terms of those who are connected to the node

- Closeness centrality: how close are the other nodes

$$C_C(i) = \frac{1}{\bar{l}_i}$$

$$\bar{l}_i = \frac{1}{n-1} \sum_{k \neq i} l_{ik}$$

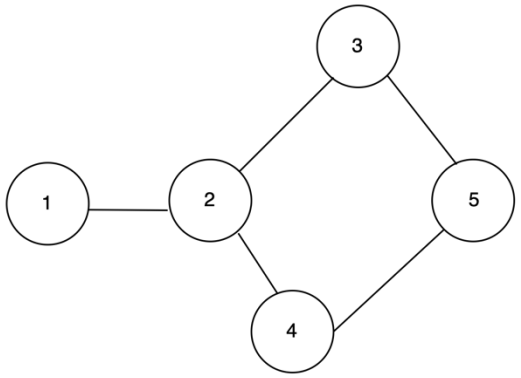
Close nodes are expected to have a smaller average shortest path length to other nodes



Node	C_C	Rank
1		
2		
3		
4		
5		

Importance in terms of those who are connected to the node

- Closeness centrality: how close are the other nodes



$$C_C(i) = \frac{1}{\bar{l}_i}$$

$$\bar{l}_i = \frac{1}{n-1} \sum_{k \neq i} l_{ik}$$

Close nodes are expected to have a smaller average shortest path length to other nodes

n=5; multiply by 1/4

(1,2) (1,3) (1,4) (1,5)

Average(l1)= ¼ (1 + 2 + 2 + 3)

$C_C(1)=1/((1+2+2+3)/4)$

$C_C(2)=1/((1+1+1+2)/4)$

$C_C(3)= 1/((?)/4)$

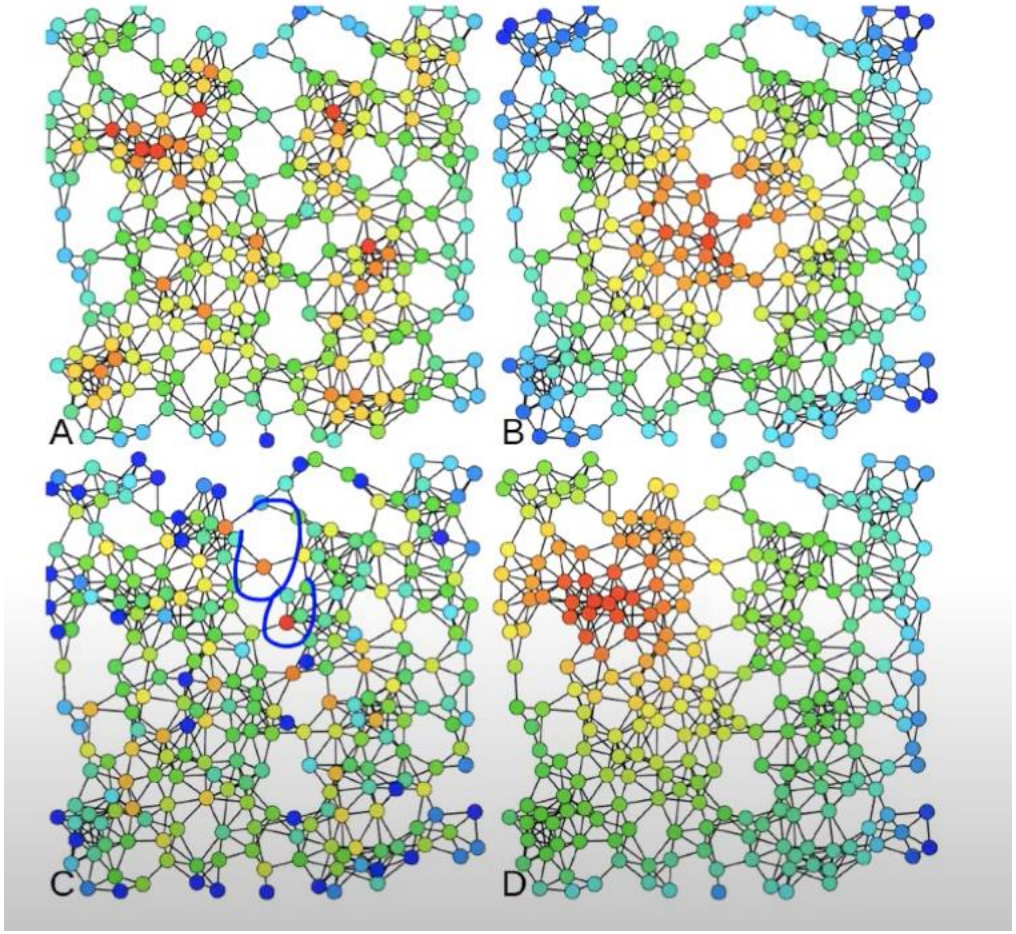
$C_C(4)= 1/(?)/4)$

$C_C(5)= 1/(?)/4)$

Node	C_C	Rank
1	1/2	
2	4/5	
3	?	
4	?	
5	?	

Connectivity measures: let us compare

© Claudio Roccinin



A) Degree centrality , B) Closeness centrality, C) Betweeness centrality, D) Eigen vector centrality

Connectivity measures: let us compare

- Generally, different centrality metrics will be positively correlated
- When they are not, there is likely something interesting about the network

	Low Degree	Low Closeness	Low Betweenness
High Degree		Embedded in cluster that is far from the rest of the network	Ego's connections are redundant - communication bypasses him/her
High Closeness	Key player tied to important/active players		Probably multiple paths in the network, ego is near many people, but so are many others
High Betweenness	Ego's few ties are crucial for network flow	Very rare cell. Would mean that ego monopolizes the ties from a small number of people to many others.	

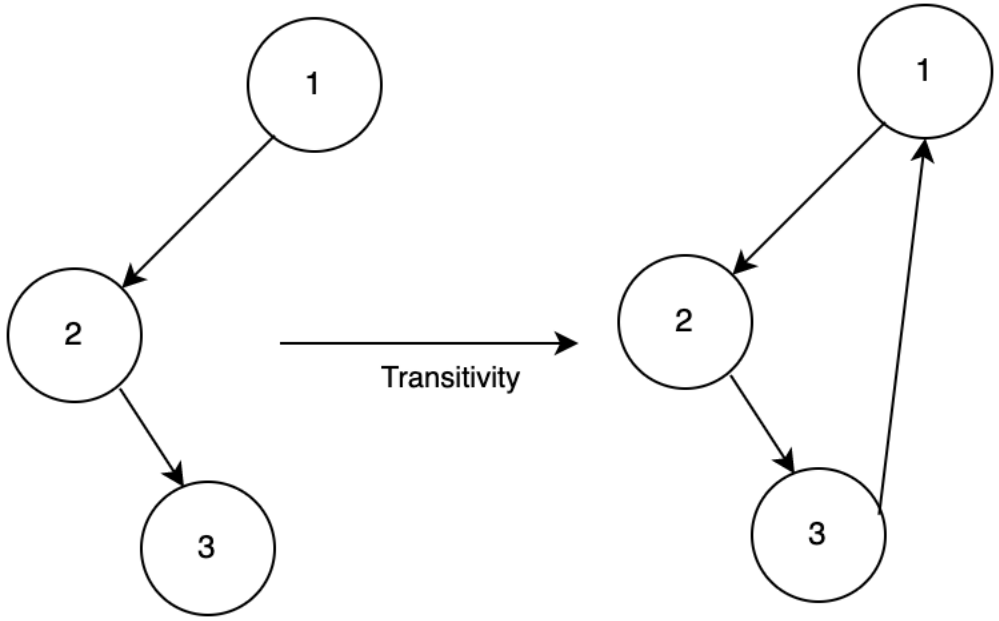
This slide is developed by James Moody

Transitivity, Reciprocity

- Transitivity

  Well-known relation property in Mathematics: if a R b and b R c then a R c, where a,b,c are items, R a relationship.

  Does this property holds in a graph?



Transitivity, Reciprocity

- Transitivity: how to measure?

  In undirected graphs, use the local clustering coefficient: how much is the graph dense in comparison to a complete graph

- Clustering coefficient

  Probability that two nodes are connected conditioned to having a **shared neighbour**

  - At a node level: consider paths of length 2

$$C_i = \frac{|e_{jk} : v_j, v_k \in N_i, e_{jk} \in E|}{d_i(d_i - 1)}$$

Directed graph

$$C_i = \frac{2|e_{jk} : v_j, v_k \in N_i, e_{jk} \in E|}{d_i(d_i - 1)}$$

Undirected graph

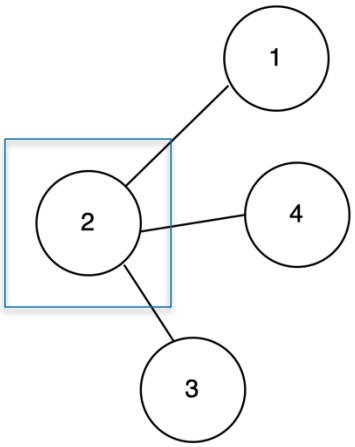
d_i: degree of node i

  - At the graph level

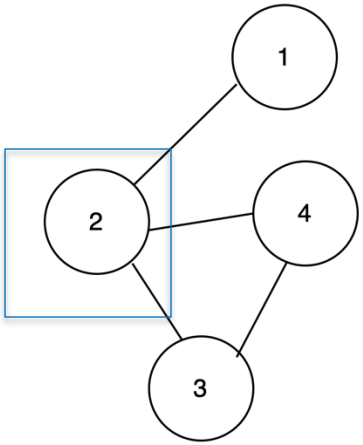
$$C = \frac{1}{n} \sum_{i=1}^{i=n} C_i$$

n: total number of nodes in graph G

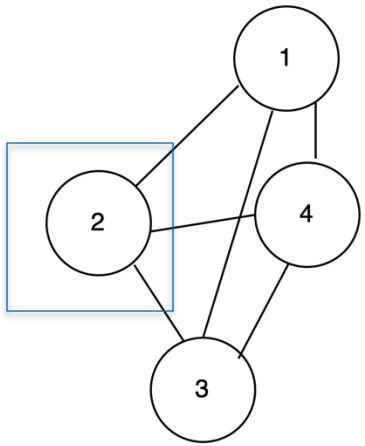
Transitivity, Reciprocity



C_2=0 (no edge between node 2' neighbours)

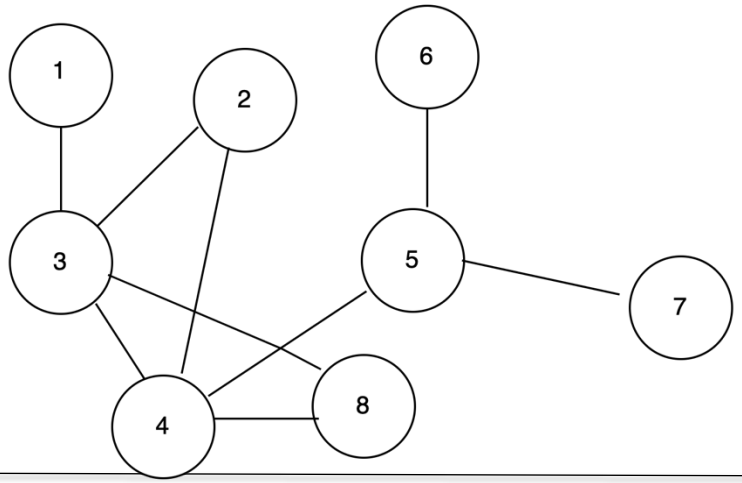


d_2 = 3 ; one edge (3,4) between node 2' neighbours
C_2=1/(3*(3-1))=1/6



d_2 = 3 ; 3 edges (3,4) ; (4,1 ; (1,3) between node 2' neighbours
C_2=3/ (3*(3-1))=1/2

Let us compute for an entire graph



Node	C_i
1	0
2	1/2*(2-1)
3	2/4*(4-1)
4	2/4*(4-1)
5	?
6	?
7	?
8	?

Similarity

- Similarity as a measure of **structural equivalence**

  We examine the overlap in the neighberhood of nodes

  - Node similarity: assume that nodes themselves are included in the neighberhood to avoid undefined values (ie. N_i includes node i)

Using the *Jaccard* measure

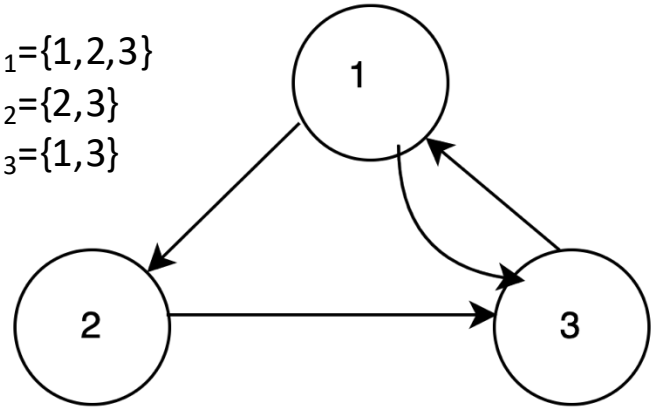
$$Sim_{jacc}(i,j) = \frac{|N_i \cap N_j|}{|N_i \cup N_j|}$$

Using the *cosine* measure

$$Sim_{cos}(i,j) = \frac{|N_i \cap N_j|}{\sqrt{|N_i||N_j|}}$$

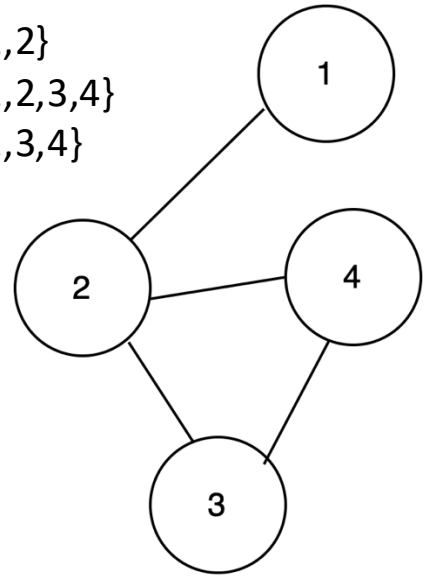
Similarity

N_1={1,2,3}
N_2={2,3}
N_3={1,3}



Node pair (i,j)	Sim $_{jacc}$	Sim $_{cos}$
(1,2)	2/3	$2/(3x2)^{½}$
(1,3)	2/3	$2/(3X2)^{½}$
(2,3)	1/3	$1/(2X2)^{½}$

N_1={1,2}
N_2={1,2,3,4}
N_3={2,3,4}



Node pair (i,j)	Sim$_{jacc}$	Sim $_{cos}$
(1,2)	2/4=1/2	$2/(2x4)^{½}$
(1,3)	?	?
(1,4)	?	?
(2,3)	3/4	$3/(4x3)^{½}$
(2,4)	?	?
(3,4)	?	?

Similarity

- Similarity **significance**

  A similarity measure is informative if it significantly differs from **random** neighberhood.

  Compare the similarity computed between nodes i and j with the measure expected if edges are randomly picked.

  Let us consider nodes i, j. Expected similarity of nodes i, j

$$Sim_E(i,j) = (d_i * d_j)/n$$

Neighberhood overlap:

$$|N_i \cap N_j| = \sum_k A_{ik}A_{jk}$$

Significance:

$$\sigma_{sign}(i,j) = \sum_k A_{ik}A_{jk} - \frac{d_i d_j}{n}$$

Similarity

- Normalized **significance using the Pearson correlation**

Significance:

$$\sigma_{pers}(i,j) = \frac{\sigma_{sign}(i,j)}{\sqrt{\sum_k (A_{ik} - \overline{A}_i)^2} \sqrt{\sum_k (A_{jk} - \overline{A}_i)^2)}}$$

$\sigma \in [-1,1]$

$$\overline{A}_i = \frac{1}{n} \sum_k A_{ik}$$

If σ reveals a significant difference, then nodes i, j are significantly similar.

Random Graph Models

- Well-konwn random graph models:
  - Regular lattice model
  - Small world model
  - Preferential attachment model

Random Graph Models

- Random graph

  Generate a graph based on a number of nodes n and/or number of edges m, probability of generating edges m: useful to study graph properties. Two main types of random graphs graphs:

  - G(n,p): Let us consider a graph $G(V,E)$, with n nodes and m edges. Any edges can be generated independently with probability p.

  - G(n,m): Let us consider that both n nodes and m edges are fixed. the number of possible graphs with n nodes and m edges is:

$$|\Omega| = \binom{\binom{n}{2}}{m}$$

Random Graph Models

- Main properties of random graphs
  - When n is large, G(n,p) and G(n,m) are similar
  - The expected degree of of a node in G(n,p) is d=(n-1)xp
  - The expected number of edges in G(n,p) is $\binom{n}{2}$
  - The expected degree (model distribution) in G(n,p) is:

$$p(d_i = d) = \binom{n-1}{d} p^d (1-p)^{n-1-d}$$

fellows a Poisson distribution, **not a power law distribution**

  - The expected local clustering coefficient of each node is p (underestimate this coefficient)

Preferential Attachment Model

- Assumptions

Barabási, Albert-László, and Réka Albert. "Emergence of scaling in random networks." *science* 286.5439 (1999): 509-512.

"*The more connected a node is, the more likely it is to receive new links*"

- Each new node is connected to existing nodes with a probability that is proportional to the number of already existing links
- Formally, probability of node k to be connected to node i

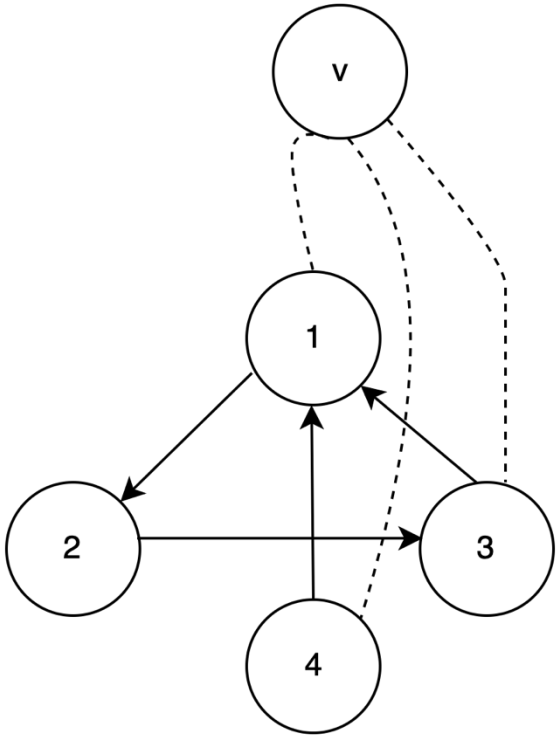
$$p_i = \frac{d_i}{\sum_j d_j}$$

- At the beginning, n0 nodes. At each step t, add a node v and m0 edges (v, v') (m0<n0) with a probability proportional to $d_{v'}$

Preferential Attachment Model

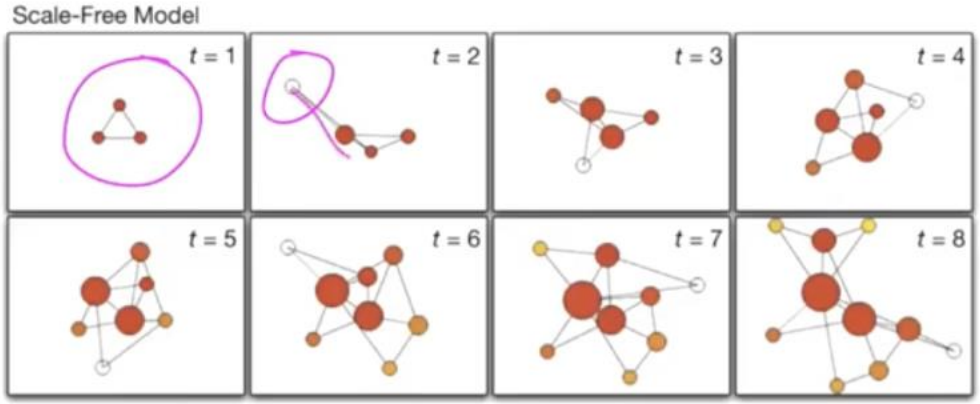
- Assumptions

$$p_i = \frac{d_i}{\sum_j d_j}$$



$P_1 = 2/4 = 1/2$
$P_2 = 1/4$
$P_3 = 1/4$
$P_4 = 0$

Barabási, Albert-László, and Réka Albert. "Emergence of scaling in random networks." *science* 286.5439 (1999): 509-512.



Preferential Attachment Model

- Algorithm generation (© Book, social media and mining)

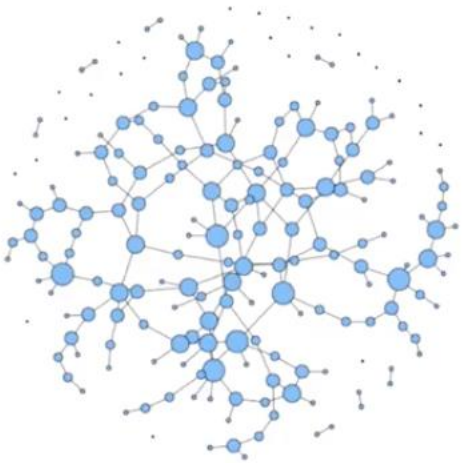
Algorithm 4.2 Preferential Attachment

Require: Graph $G(V_0, E_0)$, where $|V_0| = m_0$ and $d_v \geq 1 \; \forall \; v \in V_0$, number of expected connections $m \leq m_0$, time to run the algorithm t

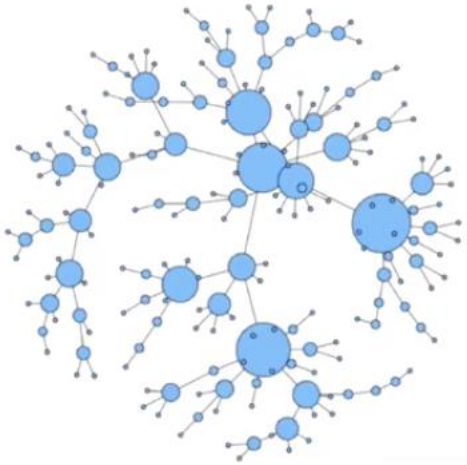
1: **return** A scale-free network
2: //Initial graph with m_0 nodes with degrees at least 1
3: $G(V, E) = G(V_0, E_0)$;
4: **for** 1 to t **do**
5: $V = V \cup \{v_i\}$; // add new node v_i
6: **while** $d_i \neq m$ **do**
7: Connect v_i to a random node $v_j \in V, i \neq j$ (i.e., $E = E \cup \{e(v_i, v_j)\}$) with probability $P(v_j) = \frac{d_j}{\sum_k d_k}$.
8: **end while**
9: **end for**
10: Return $G(V, E)$

Preferential Attachment Model

- Random graph model vs. preferential attachment model



Random graph model



Preferential attachment graph model

Preferential Attachment Model

- Properties
  - Degree

$$p(d) = \frac{2|E|^2}{d^3}$$

  - Clustering coefficient

$$C = \frac{m_0 - 1}{8} \frac{(ln\, t)^2}{t}$$

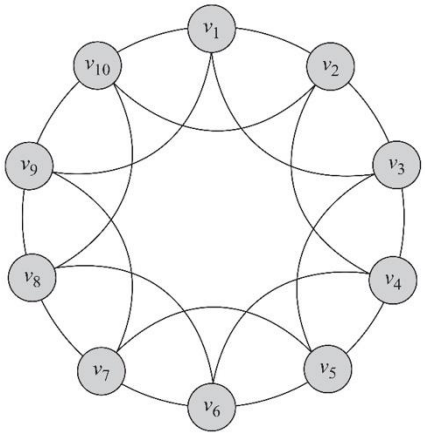
  - Average path length

$$\bar{l} \sim \frac{ln\,|E|}{ln\,(ln\,|E|)}$$

Small-world model - Regular Lattice

- In real-world (social) networks, many nodes have a limited and often at least, a fixed number of edges (remember the long-tail distribution). From the graph theory view, equivalent to embed nodes in **regular networks.**

- **A regular lattice is special regular network** where we identify patterns in node connections. For networks with degree d, each node is connected to $d/2$ previous nodes and $d/2$ fellowing nodes
  - The lattice has a **high**, but **fixed**, clustering coefficient
  - The lattice has a **high** average path length



Small World Model

- **Small world generation**

  The Watts-Stregatz (WS) random graph model characterized by:

  - a parameter $0 \leq \beta \leq 1$ controls randomness in the model
    - ✓ $\beta = 0$, the model is basically a regular lattice
    - ✓ $\beta = 1$, the model becomes a random graph

Watts, Duncan J., and Steven H. Strogatz. "Collective dynamics of 'small-world'networks." *nature* 393.6684 (1998): 440-442.

  - Short average path length
  - High clustering coefficient
  - In practice most of the nodes have similar degrees

Small World Model

- Algorithm generation (© Book, social media and mining)

Rewiring: *take an edge, change one of its end-points randomly*

Algorithm 4.1 Small-World Generation Algorithm

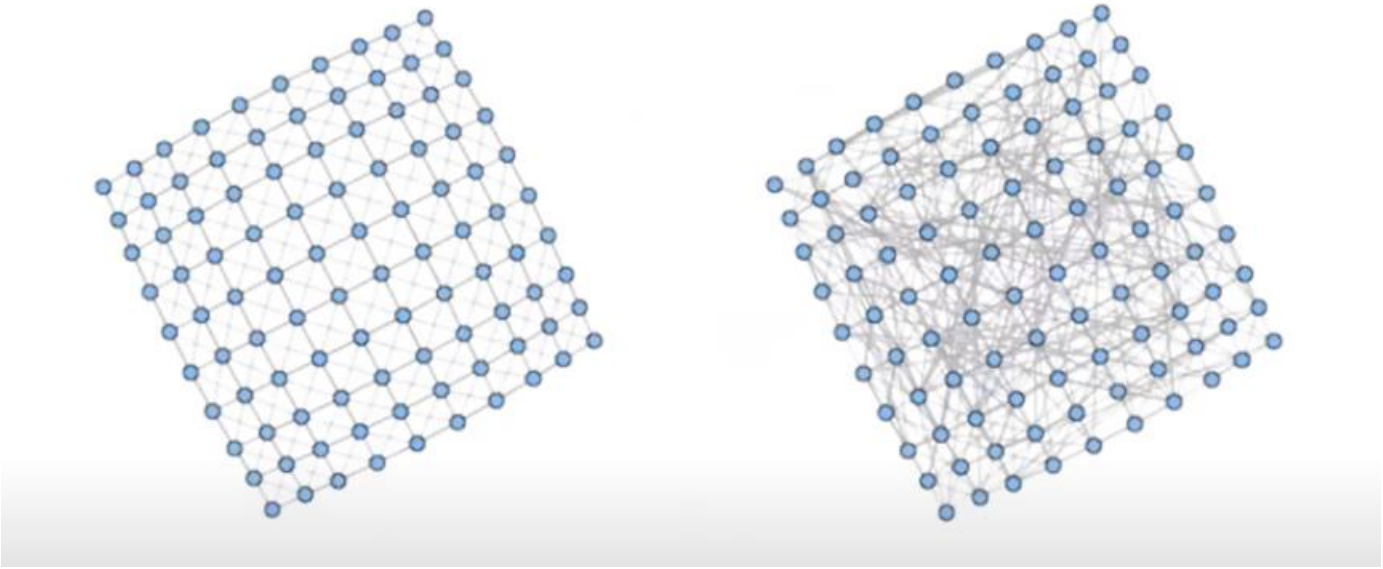
Require: Number of nodes $|V|$, mean degree c, parameter β

1: **return** A small-world graph $G(V, E)$
2: G = A regular ring lattice with $|V|$ nodes and degree c
3: **for** node v_i (starting from v_1), and all edges $e(v_i, v_j)$, $i < j$ **do**
4: v_k = Select a node from V uniformly at random.
5: **if** rewiring $e(v_i, v_j)$ to $e(v_i, v_k)$ does not create loops in the graph or multiple edges between v_i and v_k **then**
6: rewire $e(v_i, v_j)$ with probability β: $E = E - \{e(v_i, v_j)\}$, $E = E \cup \{e(v_i, v_k)\}$;
7: **end if**
8: **end for**
9: Return $G(V, E)$

Small World Model

- Algorithm generation (© Book, social media and mining)

Rewiring: *take an edge, change one of its end-points randomly*



Lattice graph

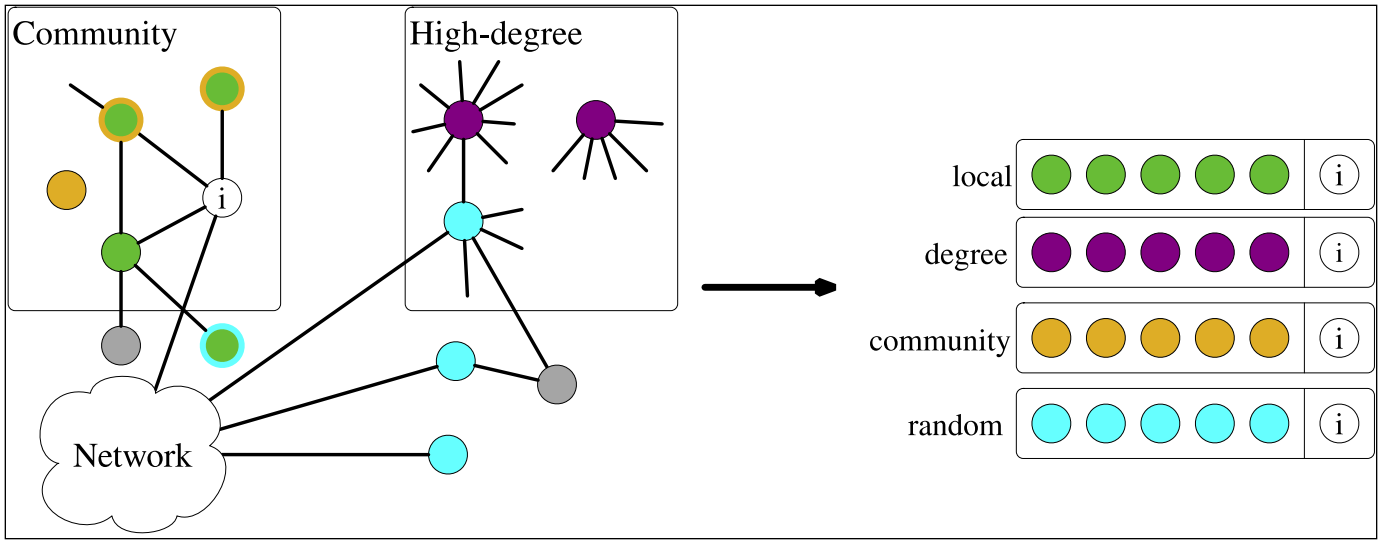
Avg. Path length = 3.58
Clust. Coeff = 0.49

Small world graph with 20% rewiring

Avg. Path legth = 2.32
Clust coeff = 0.19

Graphs and data complexity: How to sample?

© Ivan Bugere, Tutorial on representation learning of graphs



Local: target nodes and sample their neighbours
Degree: target central nodes
Community: target k communities
Random: random choice of nodes and edges

Community in Social Media

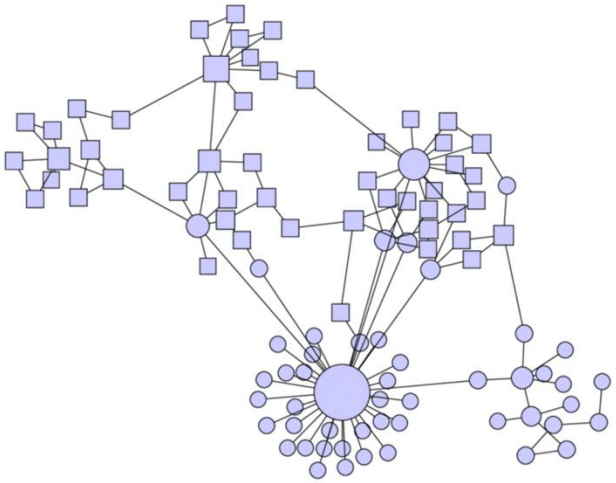
- What is a (social) community?

Lack of universal definition.

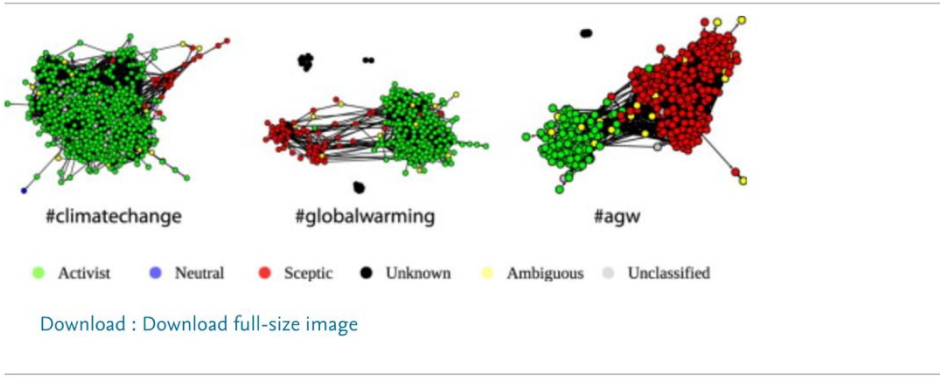
A (social) community is formed by a group of actors such that those within the group interact with each other more frequently that those outside the group, a.k.a **group cluster**, **cohesive subgroup**, **module.**

- Exlicit vs. Implicit groups
  - **Explicit** groups: formed by user subscriptions
    - ✓ Not all sites provide community platforms
    - ✓ Not all people want to make effort to join groups
  - **Implicit** (real-world) groups: implicitly formed by social interactions along time
    - ✓ Groups can change dynamically based on their interactions (independently of subscription "link')
    - ✓ Support for interaction understanding, navigation, visualization, other tasks (eg. propagation, influence measurement)

Community in Social Media



Collaboration network in Research gate
Source: Anderson, Katharine. (2011).
Skill Specialization and the Formation of Collaboration Networks.



Climate change polarisation "sceptic" vs. "activist"
Source: Global Climate Change. (2015).
ywel T.P. Williams, James R. McMurray, Tim Kurz, F. Hugo Lambert,

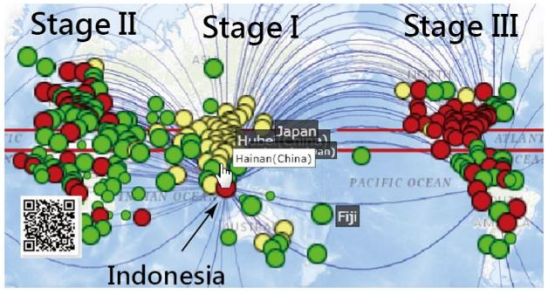
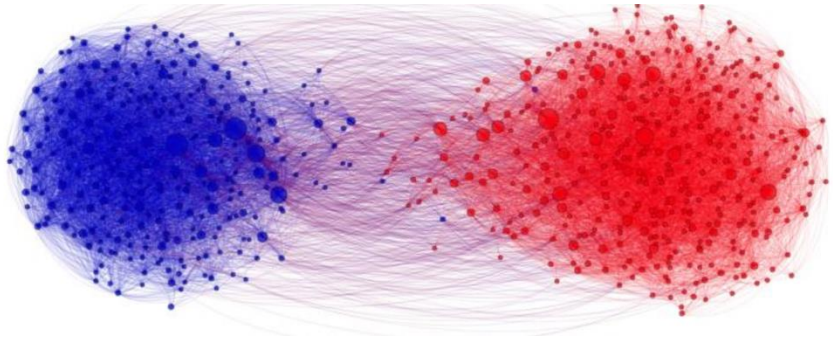


Figure 3. Indonesia compared to three clusters separated by social network analysis (SNA) using the correlation pattern across three stages in yellow, green, and red, respectively. (Note: the curve means the close relation (correlation coefficient (CC) > 0.8) between two countries/regions according to SNA guidelines (Note: readers are invited to scan the QR-code to examine the detail on a dashboard)).

Using Social Network Analysis to Identify Spatiotemporal Spread Patterns of COVID-19 around the World
Source: Environmental Research and Public Health (2021).
Kyent-Yon Yie, Tsair-Wei Chien , Yu-Tsen Yeh, Willy Chou, Shih-Bin Su



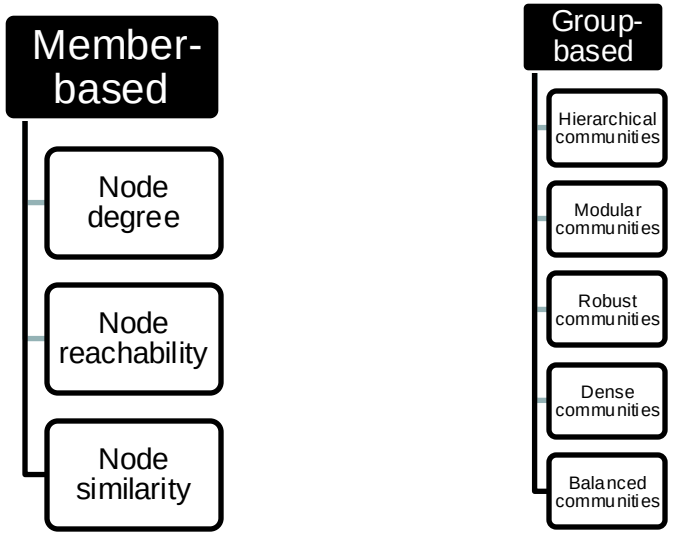
Democratic/Republican political blogs
(Source: Adamic, 2004)

Community in Social Media

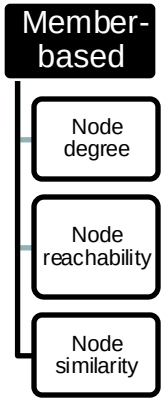
- Community detection

  Process of identifying clusters of nodes with **strong intra-connections** and **weak inter-connections**

- Algorithms for community detection
  - Input: graph G(V,E)
  - Output:
    - Non overlapping communities: Partition of subgraphs over G $C_1(V1,E_1),...C_k(V_k,E_k)$
    - Overlapping communities: set of subgraphs over G $C_1(V1,E_1),...C_k(V_k,E_k)$
  - Approaches



Member-Based Community Detection



- Key idea

  Examine **node characteristics** and identify nodes with **similar characteristics** to be considered as a community.

  Which characteristics:

  - **Degree:** Nodes with similar degrees (e.g., ***cliques***) are in the same community
  - **Reachability:** Nodes connected with shortest paths are in the same community (e.g., ***k-cliques***)
  - **Similarity:** Similar nodes are in the same community

Degree - Cliques

- What is a clique?

A maximum **complete subgraph** where all nodes are direct neighbours

- Communities as cliques: identify subgraphs with the fellowing characteristics:
  - ***The maximum clique***: with the maximal number of nodes
  - ***All the maximal cliques***: that are not subgraphs of a larger cliques

Source: Book, Social Media Mining

Algorithm 1 Brute-Force Clique Identification

Require: Adjacency Matrix A, Vertex v_x

1: **return** Maximal Clique C containing v_x
2: CliqueStack = $\{\{v_x\}\}$, Processed = $\{\}$;
3: **while** CliqueStack not empty **do**
4: C=pop(CliqueStack); push(Processed,C);
5: v_{last} = Last node added to C;
6: $N(v_{last}) = \{v_i | A_{v_{last},v_i} = 1\}$.
7: **for all** $v_{temp} \in N(v_{last})$ **do**
8: **if** $C \bigcup \{v_{temp}\}$ is a clique **then**
9: push(CliqueStack, $C \bigcup \{v_{temp}\}$);
10: **end if**
11: **end for**
12: **end while**
13: Return the largest clique from Processed

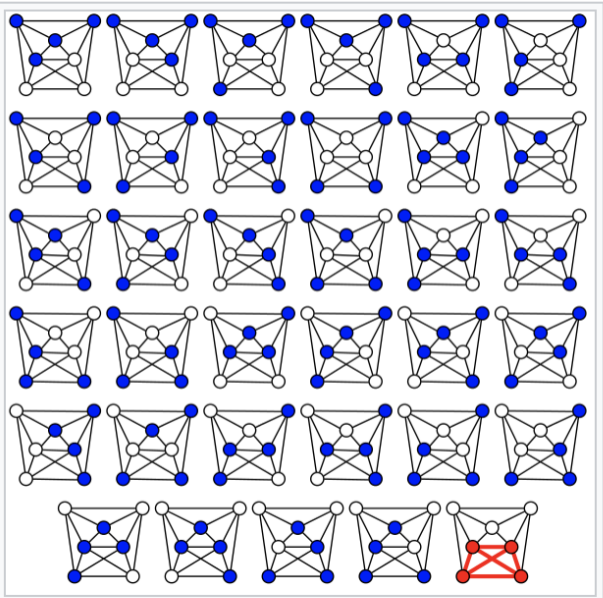
Begin with a node x and fellow the edged through adjacency matrix A

Push the neigbours in the candidate clique

Check the clique properties

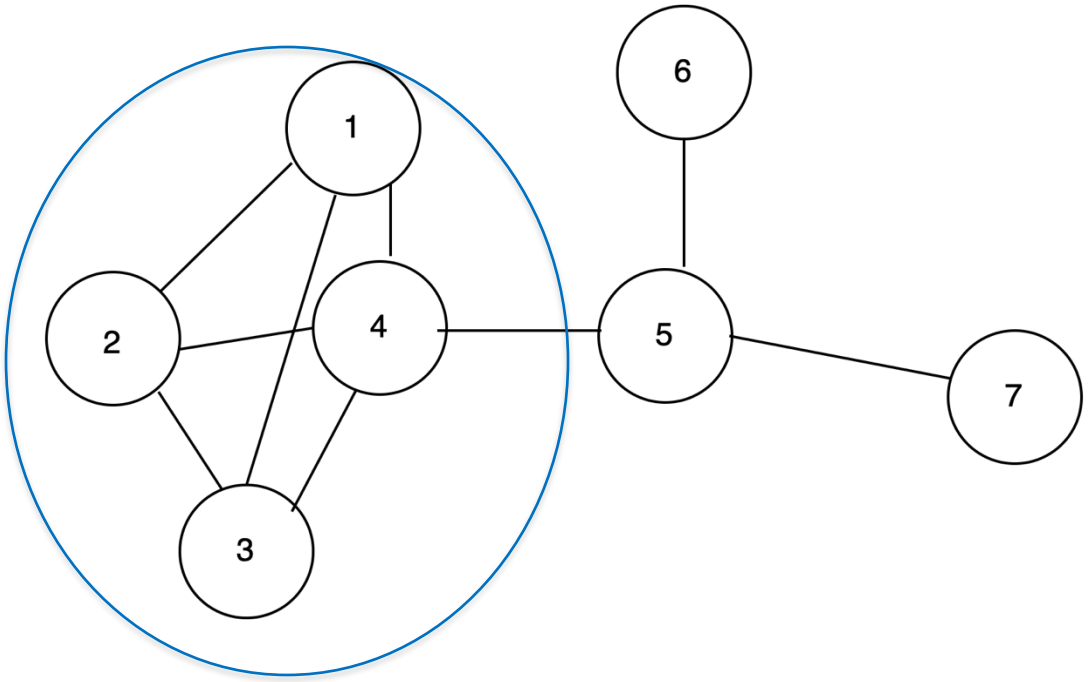
Community as the largest clique built upon node x (includes node x)

Degree - Communities as Cliques



The brute force algorithm finds a 4-clique in this 7-vertex graph (the complement of the 7-vertex path graph) by systematically checking all C(7,4) = 35 4-vertex subgraphs for completeness.

Source: Wikipedia



Nodes 1,2,3 and 4 form a clique

Degree - Cliques

- Issues

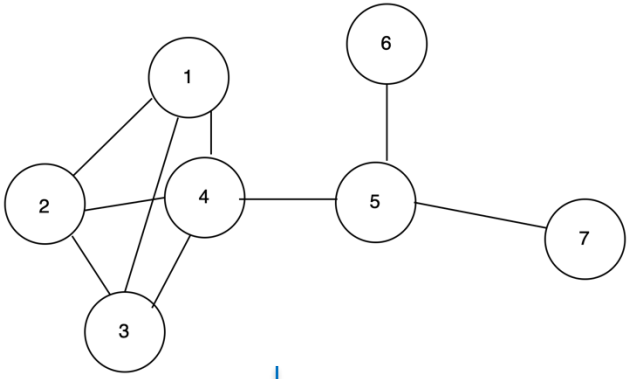
Finding cliques and maximum cliques (using a brut-force algorithm) are NP-hard problems, costly in terms of time complexity.

Impractical for large networks

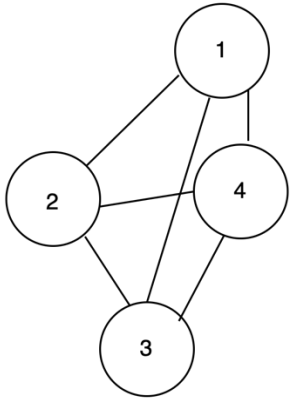
- Solution using a pruning procedure
  - Sample a sub-network from the given network, and find a clique in the sub-network (e.g., use a greedy approach, see algorithm Slide ???)
  - Suppose the clique above is size k, in order to find out a larger clique, all nodes with degree $<= k\text{-}1$ should be removed
  - Repeat until the network is small enough

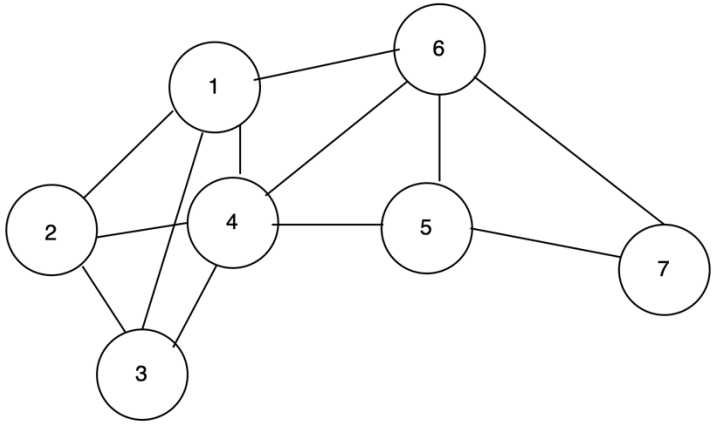
Larger clique is of size > k, so each node has a degree >=k-1.
Thus, we can prune nodes with degree <=k

Degree - Cliques



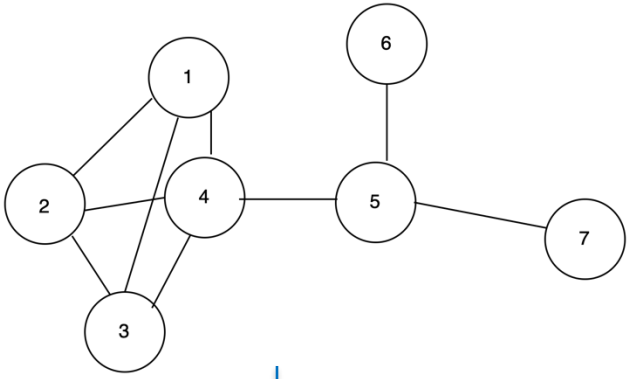
Find a clique >=3





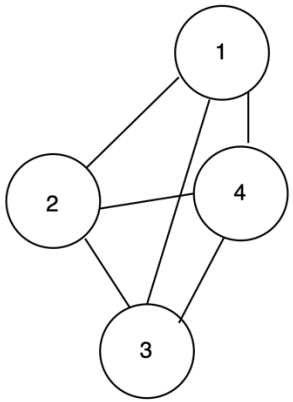
Find a clique >=4

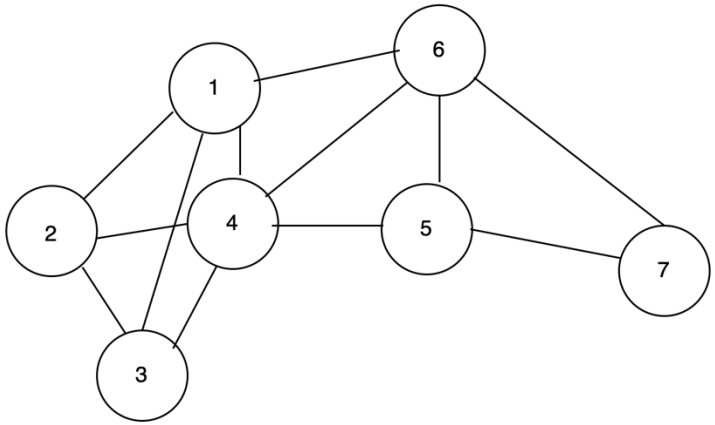
Degree - Cliques



Find a clique >=3,
Remove
All nodes with
degree <= (3-1)-1=1

Remove nodes 5,6,7
Obtained clique:
{1,2,3,4}





Find a clique >=4, remove
All nodes with degree <= (4-1)-1=2

Remove nodes 5,7
Remove node 6
Obtained clique: {1,2,3,4}

Weaknesses
- Cliques are rare
- Cliques are not robust: removing a single edge could lead to destroy the clique
- Real-world communities are overlapping, do not form separate cliques

Degree - Clique Percolation Method (CPM)

- Definitions

k-clique. A k-clique is a fully connected set of k nodes, i.e. every pair of its nodes is connected by an edge in the graph.

k-cliques adjacency. Two k-cliques are adjacent iif and they **share $k-1$ nodes**.

k-clique (CPM) community. A k-clique community (or cpm community) is the set of the nodes belonging to a maximal set of k-cliques that can be reached from each other through a series of adjacent k-cliques.

- General algorithm

Input: graph G, parameter k

Procedure

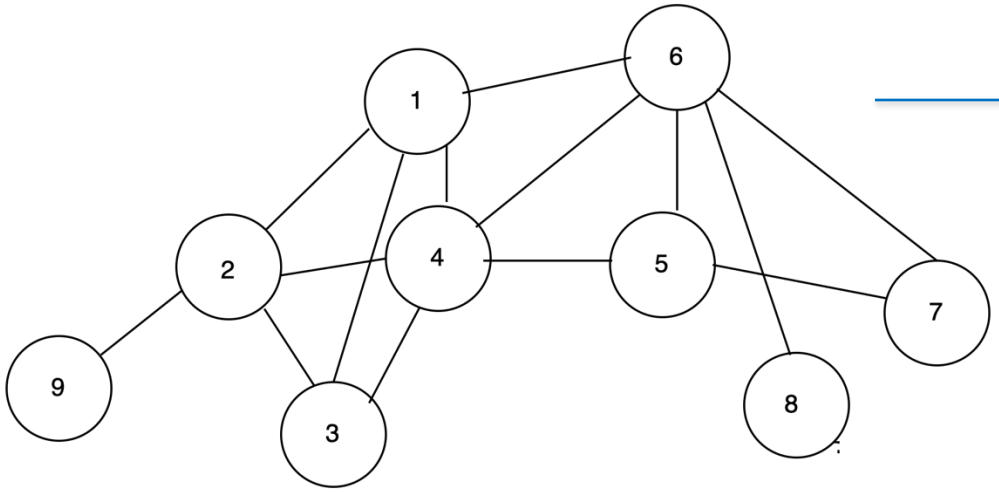
- Find out all k-cliques in G
- **Build a click graph based on k-clique adjacency**
- Identify the k-clique communities

Algorithm 1 Exact CPM algorithm

```
1: UF ← Union-Find data structure
2: Dict ← Empty Dictionary
3: for each k-clique c_k ∈ G do
4:     S ← ∅                                  ▷ communities of c_k to merge
5:     for each (k − 1)-clique c_{k−1} ⊂ c_k do
6:         if c_{k−1} ∈ Dict.keys() then
7:             p ← UF.Find(Dict[c_{k−1}])
8:         else
9:             p ← UF.MakeSet()
10:            Dict[c_{k−1}] ← p
11:        S ← S ∪ {p}
12:    UF.Union(S)
```

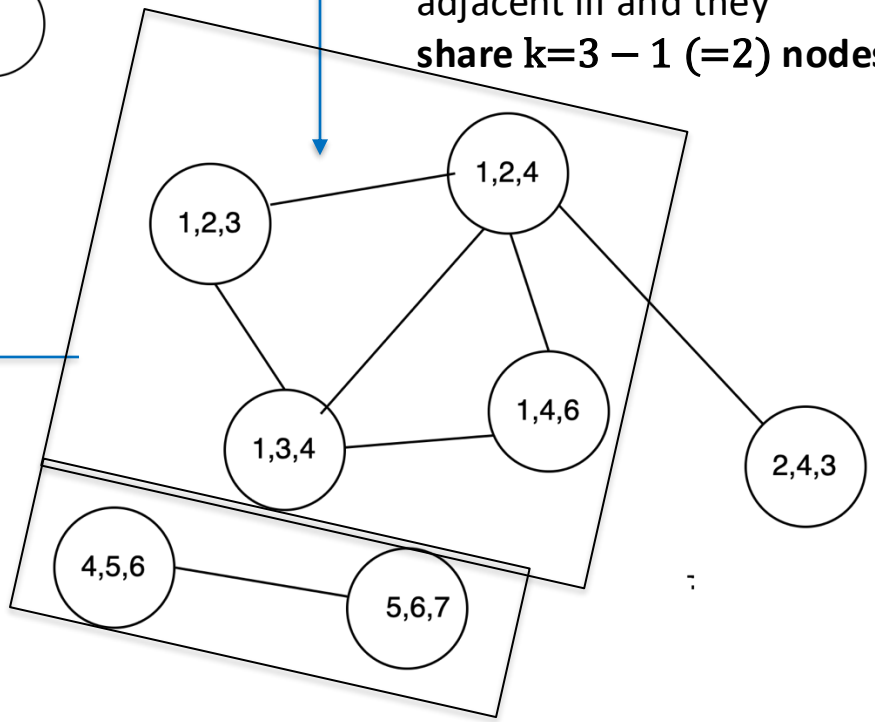
© Book, Social Media Mining

Degreee - Clique Percolation Method (CPM)



Cliques of size 3 (where each pair of nodes is connected)
{1,2,3}, {1,3,4}, {1,2,4}, {1,4,6}, {2,4,3}, {4,5,6}, {5,6,7}

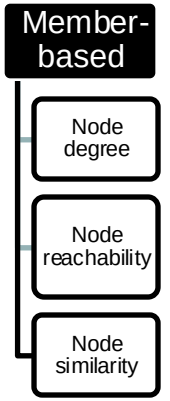
adjacent iif and they **share k=3 – 1 (=2) nodes**



Communities
{4,5,6,7}
{1,2,3,4,6}

Weaknesses
- Nodes are assumed to belong to the same community if there is a path between them (See definition of k-clique) regardless of the distance

Member-Based Community Detection



- Key idea

  Examine **node characteristics** and identify nodes with **similar characteristics** to be considered as a community.

  Which characteristics:

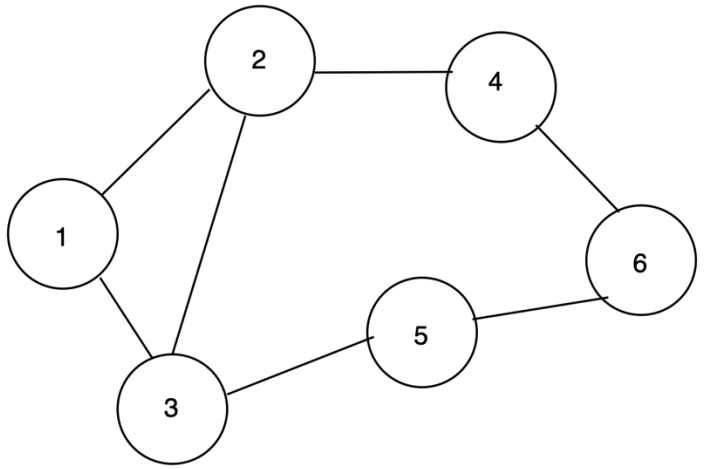
  - Degree**:** Nodes with similar degrees (e.g., ***cliques***) are in the same community
  - **Reachability:** Nodes connected with shortest paths are in the same community (e.g., ***k-cliques***)
  - Similarity**:** Similar nodes are in the same community

Reachability - K-clique, K-club

- Key idea: narrow the nodes in the same community, notion of **reachability in k-hops**

k-clique. A maximal subgraph in which the **maximal distance between any nodes is <= k**.

k-club. k-clique with an additional constraint: nodes on the shortest path should be parts of the subgraph (ie., diameter <=k)



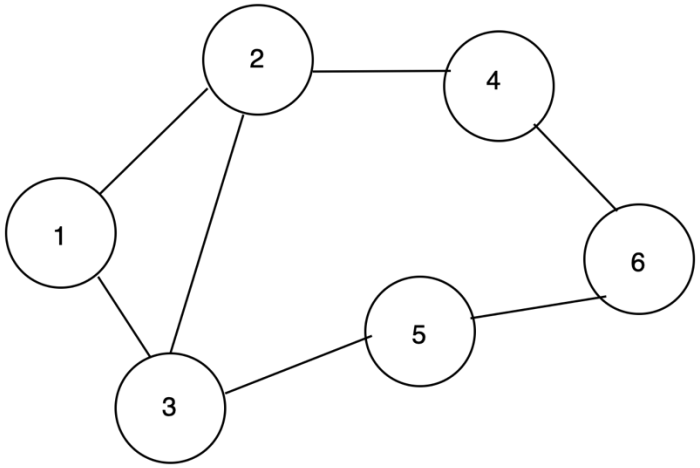
Cliques
2-clique:
2-clubs:

Reachability - K-clique, K-club

- Key idea: narrow the nodes in the same community, notion of **reachability in k-hops**

k-clique. A maximal subgraph in which the maximal distance between any nodes is <= k.

k-club. k-clique with an additional constraint: nodes on the shortest path should be parts of the subgraph (ie., diameter <=k)



Cliques {1,2,3}: based on 1-hop neighbouring

2-clique: maximal distance between nodes is **2**: {1,2,3,4,5}, {2,3,4,5,6},

2-clubs: maximal distance between nodes is **2**

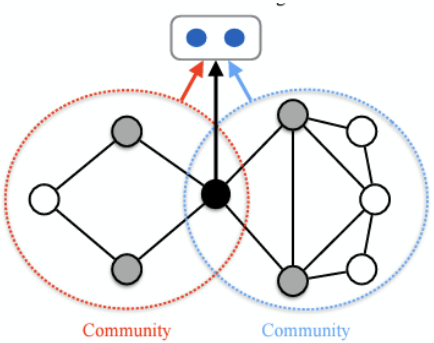
{1,2,3,4}
{1,2,3,5}

Derived from {1,2,3,4,5}: shortest path between 4 and 5 passes through node which is outside of the clique

{2,3,4,5,6}

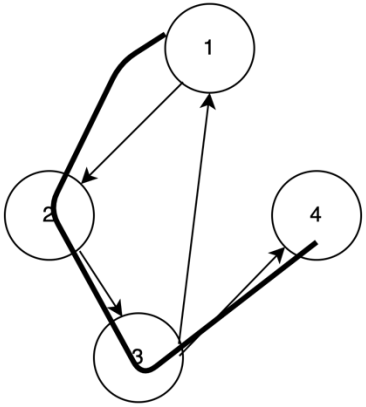
Reachability – Use node embeddings

- Community-enhanced deep walk
  - Key assumptions
    - Each node may belong to multiple communities
    - Each sequence of nodes built with a random walk belong to one community
    - A community is determined with: 1) its sequence distribution over communities; 2) the community distribution of its nodes over nodes



The embeding of a node is Determined by both its neighbors and communities

  - Key steps
    - For each node, calculate a walk sequence
    - Learn node embeddings that maximze the predictions of neighbour nodes and belonging communities



Node 1, walk sequence: 1, 2, 3, 4

Tu & al. A unified framework for community detection and network representation learning, IEEE Transactions on Knowledge and Data Engineering, 2016

Reachability – Use node embeddings

- Community-enhanced deep walk
  - For a node *v* and sequence *s*, compute the probability assignment to community *c*

$$P(c|s) \approx P(v|c) \times P(c|s)$$

  - Maximze *the* predictions of **neighbour nodes** and **belonging communities**

$$L = -\left(\frac{1}{|s|}\sum_{u,v \in s} \underbrace{\log(P(v|u))}_{\text{neighbour nodes}} + \underbrace{\log(P(v|c))}_{\text{Belonging communities}}\right)$$

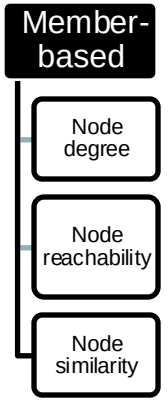
$$P(v|u) = \frac{\exp(z_u^T z_v)}{\sum_{n \in V} \exp(z_u^T z_n)}$$

$$P(v|c) = \frac{\exp(z_v^T z_c)}{\sum_{n \in V} \exp(z_v^T c_n)}$$

Similar to deep walk

Tu & al. A unified framework for community detection and network representation learning, IEEE Transactions on Knowledge and Data Engineering, 2016

Member-Based Community Detection



- Key idea

  Examine **node characteristics** and identify nodes with **similar characteristics** to be considered as a community.

  Which characteristics:

  - Degree**:** Nodes with similar degrees (e.g., ***cliques***) are in the same community
  - Reachability**:** Nodes connected with shortest paths are in the same community (e.g., ***k-cliques***)
  - **Similarity:** Similar nodes are in the same community

Similarity

- Key idea: Similar (most similar) nodes belong to the same community.

  Perform classical clustering methods (e.g., k-means) on node similarities:

Jaccard similarity

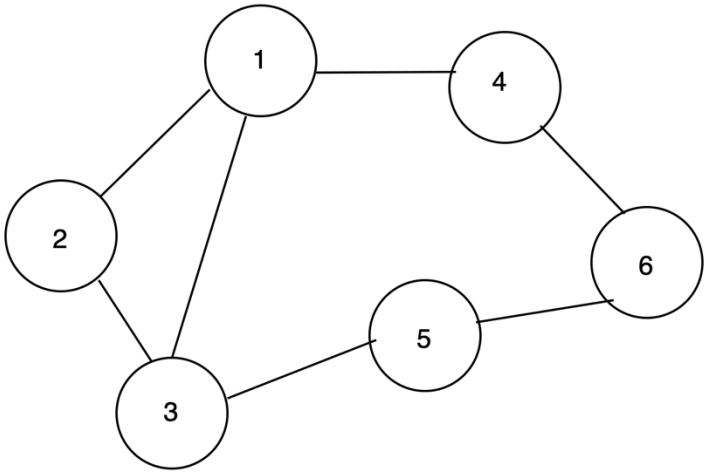
$$Sim_{jacc}(i,j) = \frac{|N_i \cap N_j|}{|N_i \cup N_j|}$$

cosine similarity

$$Sim_{cos}(i,j) = \frac{|N_i \cap N_j|}{\sqrt{|N_i||N_j|}}$$

Similarity

- Key idea: Similar (most similar) nodes belong to the same community.



1. Build a similarity matrix (jaccard, cosin,..)

	1	2	3	4	5	6
1	?	3/4	?	?	?	?
2	?	?	?	?	1/5	
3	?	?	?	?	?	?
4	?	?	?	?	?	1/2
5	?	?	?	?	?	?
6	?	?	?	?	?	?

2. Perform k-means (e.g. 2 means)
i=0, $C_1 = \{2\}$, $C_2 = \{6\}$

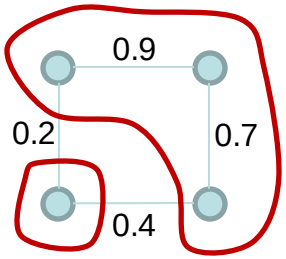
Similarity

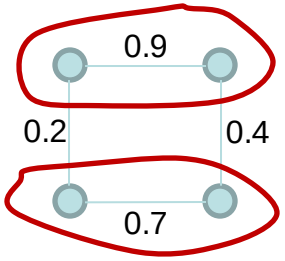
- Apply a threshold on similarities to delimit the communities

Similarity-based community detection method

1. Set a community threshold σ
2. Compute the similarity for each pair of nodes in the given graph
3. If the similarity of two nodes is over the threshold σ, the two nodes form the same community.

$\sigma = 0.5$





Weakness

Need to compute the similarity between each pair of nodes. Do not scale for large networks

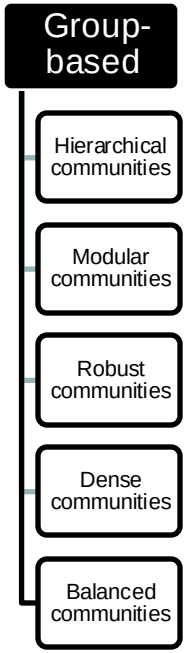
Group-Based Community Detection

- Key idea

  Examine **group characteristics** and identify those **having key characteristics** to be considered as a community.

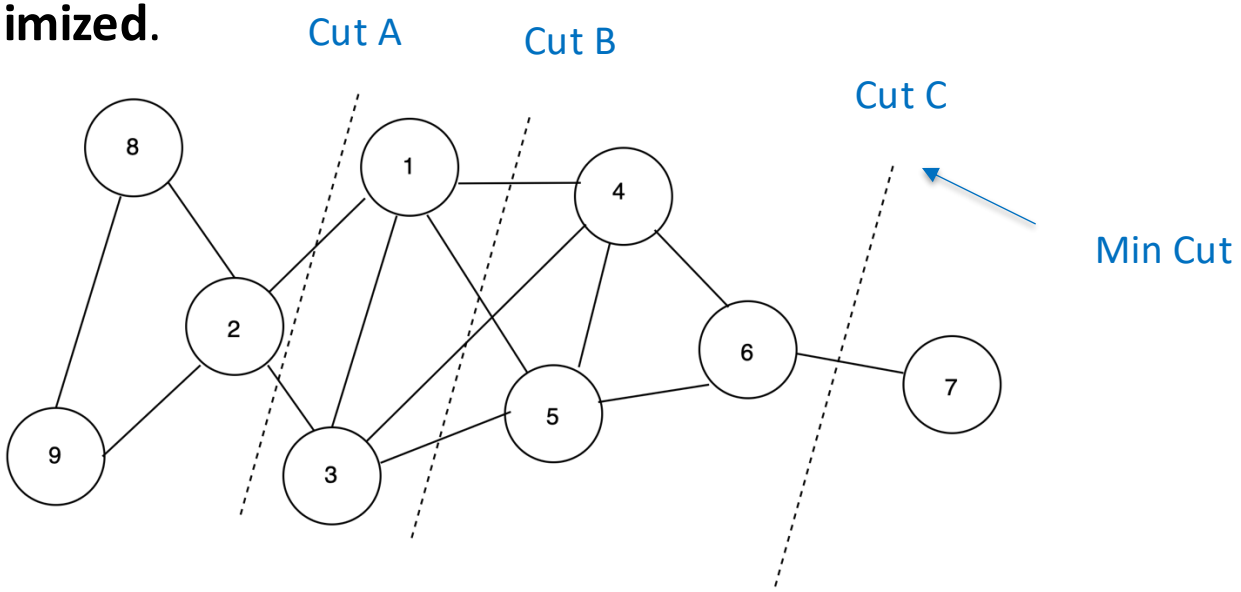
  Which characteristics:

  - **Balanced:** spectral clustering
  - **Robust:** k-connected graphs
  - **Modular:** graphs with maximal modularity
  - **Dense**: Quasi-cliques
  - **Hierarchical**: hierarchical clusters



Balanced communities

- Cut the network into two partitions such that the number of edges crossed by the cut is minimal
  - **Cut partitioning (cut):** partitions obtained after cut. **The size of the cut is the number of edges that are cut**
  - **Min-cut :** find the cut-partitions, such as the number of edges between two partitions is **minimized**.



Balanced communities

- **Ratio-Cut**

$$Ratio\ Cut(\pi) = \frac{1}{k}\sum_{i=1}^{i=k}\frac{cut(C_i, \overline{C}_i)}{|C_i|}$$

π: *partition*
k: partition size
$|C_i|$: number of nodes in C_i
$Vol(C_i)$: sum of degrees in C_i
$\overline{C}_i$ complement of C_i
$cut(C_i, \overline{C}_i)$ size of the cut

Favors larger communities (minimization of Ratio Cut)

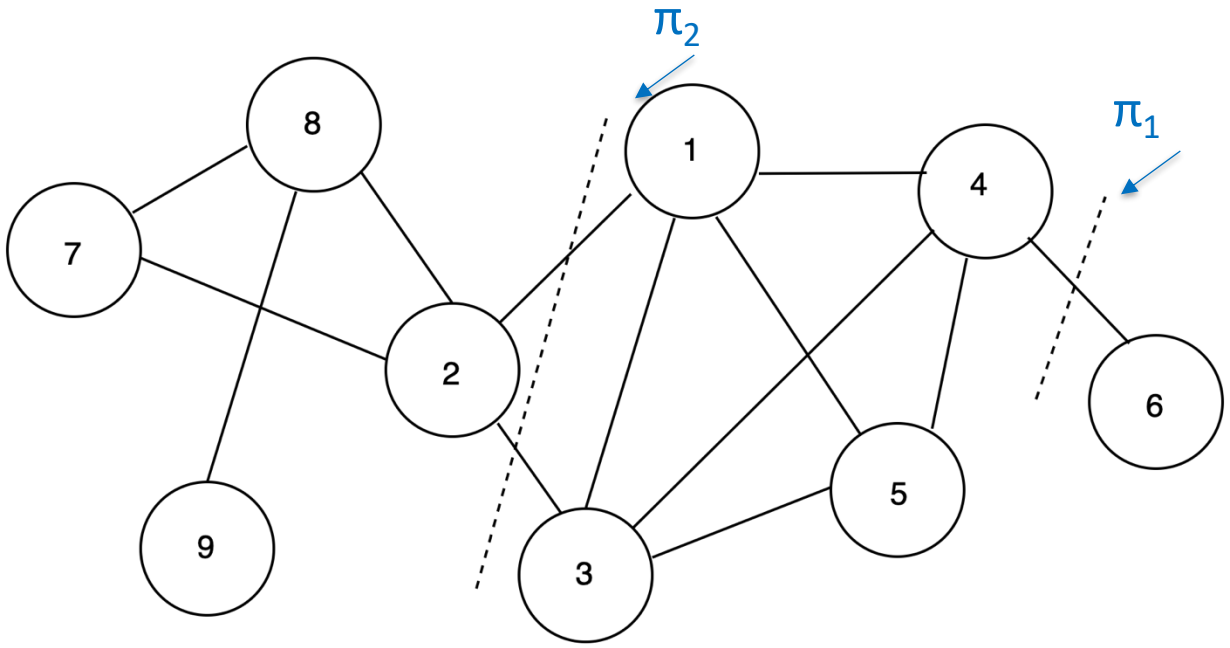
- **Normalized Cut**

$$Normalized\ Cut(\pi) = \frac{1}{k}\sum_{i=1}^{i=k}\frac{cut(C_i, \overline{C}_i)}{vol(C_i)}$$

Weakness:

Imbalanced partitions: we would like a cut that favors large communities over small ones

Balanced communities



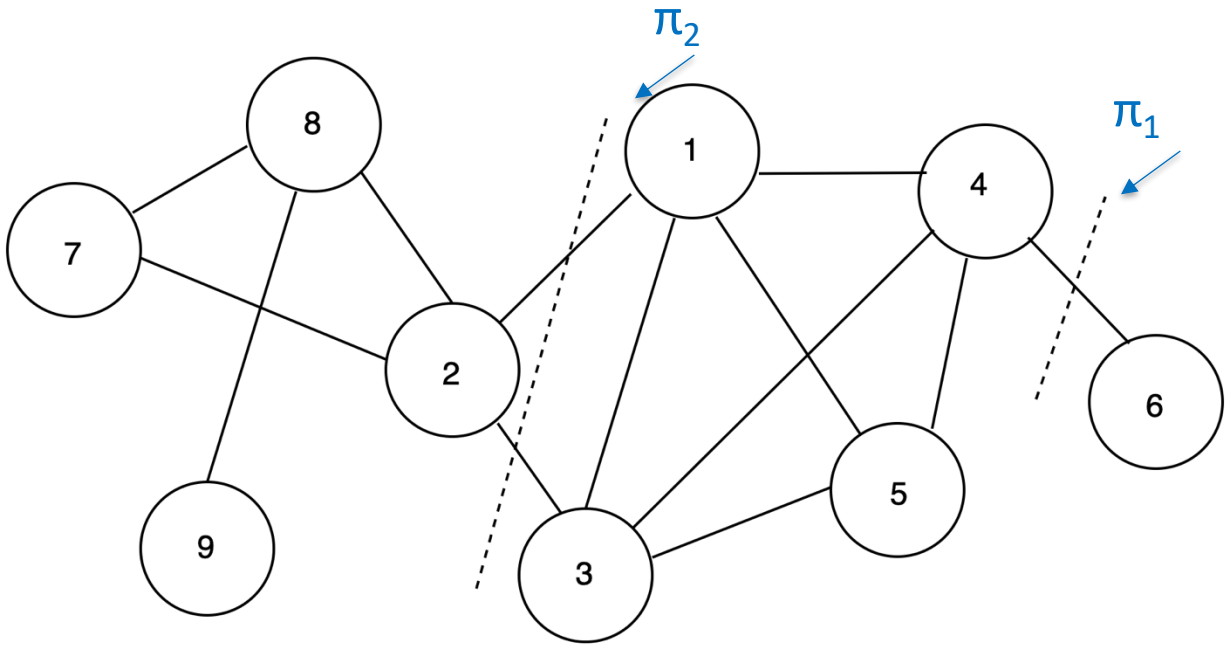
Ratio Cut (π_1) =

Ratio Cut (π_2) =

Cut (π_1) =

Normalized Cut (π_2) =

Balanced communities



Ratio Cut (π_1) = ½ (1/1+ 1/8) = 9/16 = 0.56

Ratio Cut (π_2) = ½ (2/4+ 2/5) = 9/20 = 0.45 Normalized

Normalized Cut (π_1) = ½ (1/1+ 1/27) = 14/27 = 0.52

Normalized Cut (π_2) = ½ (2/12+ 2/16) = 7/48 = 0.15

We can see that Ratio Cut (π_2) < Ratio Cut (π_1), and Normalied Cut (π_2) < Normalized Cut (π_1)

Balanced communities

- **Matrix formulation**

$$Ratio\ Cut(\pi) = \frac{1}{k}\sum_{i=1}^{i=k}\frac{cut(\pi_i, \overline{\pi}_i)}{|C_i|}$$

$$= \frac{1}{k}\sum_{i=1}^{i=k}\frac{X_i^T(D-A)X_i)}{X_i^T X_i}$$

$$= \frac{1}{k}\sum_{i=1}^{i=k} X_i^T(D-A)\widehat{X}_i$$

$$\widehat{X}_i = \frac{\widehat{X}_i}{(\widehat{X}_i^T\widehat{X}_i)^{1/2}}$$

- ✓ π: *partition*
- ✓ X_{ij}= 1 if i belongs to community j, 0 otherwise
- ✓ D: Diagonal of the degree matrix
- ✓ X^TAX: number of edges that are inside of community i
- ✓ $X^{TD}X$: number of edges that are connected to members of community i
- ✓ XT(D-A)X is the numebr of edged in the cut that separates community from other nodes

The i^{th} diagonal of $X^T(D-A)X$ is equivalent to $cut(\pi_i, \overline{\pi}_i)$

Balanced communities

- **Matrix formulation**

L laplacian matrix

Optimal solution for spectral optimization:

$$min_{\widehat{X}} Tr(\widehat{X}^T L \widehat{X})$$

$\widehat{X}$ is the top eigenvectors with the smallest eigenvalues

Balanced communities

- **Matrix formulation**

$$Ratio\ Cut(\pi) = \frac{1}{k}\sum_{i=1}^{i=k}\frac{cut(\pi_i, \overline{\pi}_i)}{|C_i|}$$

$$= \frac{1}{k}\sum_{i=1}^{i=k}\frac{X_i^T(D-A)X_i)}{X_i^T X_i}$$

$$= \frac{1}{k}\sum_{i=1}^{i=k} X_i^T(D-A)\widehat{X}_i$$

$$\widehat{X}_i = \frac{\widehat{X}_i}{(\widehat{X}_i^T\widehat{X}_i)^{1/2}}$$

- ✓ π: *partition*
- ✓ X_{ij}= 1 if i belongs to community j, 0 otherwise
- ✓ D: Diagonal of the degree matrix
- ✓ X^TAX: number of edges that are inside of community i
- ✓ $X^{TD}X$: number of edges that are connected to members of community i
- ✓ XT(D-A)X is the numebr of edged in the cut that separates community from other nodes

The i^{th} diagonal of $X^T(D-A)X$ is equivalent to $cut(\pi_i, \overline{\pi}_i)$

Balanced communities

- **Matrix formulation**

L laplacian matrix

Optimal solution for spectral optimization:

$$min_{\widehat{X}} Tr(\widehat{X}^T L \widehat{X})$$

$\widehat{X}$ is the top eigenvectors with the smallest eigenvalues

Density-based groups

- A community is viewed as a **dense** subgraph
- How to measure graph density?

  A subgraph $G_s(V_s,E_s)$ is a $\gamma - dense$ quasi-clique if:

$$\frac{|E_s|}{|V_s|(|V_s-1)/2} \geq \gamma$$

- How to identify $\gamma - dense$ quasi cliques in graph G(V,E)?

  Use a similar strategy that of cliques

  - Sample a subgraph, and find a maximal $\gamma - dense$ quasi-clique say of size k)
  - Prune nodes with degree $< k\gamma$

Hierarchical clustering

- Goal: build a hierarchical structure of communities based on network structure. Two main clustering approaches
  - Divisive Hierarchical Clustering: top-down
  - Agglomerative Hierarchical clustering: bottom-up

- Divisive clustering: naïve approach

level $i = 0$

$G_s = \text{Graph}(G,E)$; $CG_s(i) = [Gs]$

Repeat

- Partition each subgraph in $CG_s(i)$ at level i in a set of clusters $SCG_s(i)$
- $CG_s(i) = \text{Union}(SCGs(i))$
- $i = i+1$

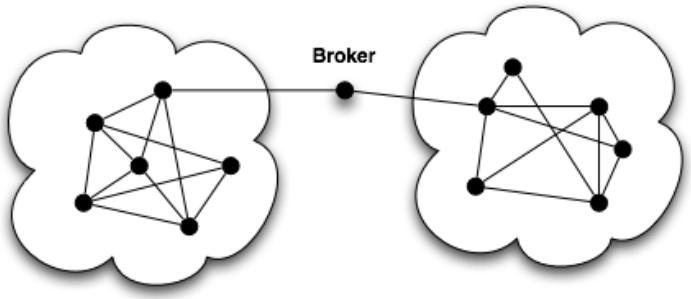
Until reaching singleton graphs

- Key components
  - How to detect the characterize the community frontiers: focus on important nodes, important edges?

Hierarchical clustering

- **Girvman-Newman algorithm**: focuses on edges that are most likely between communities
  - Find the **edges** with the least **strength (high betweeness).**
  - Remove the edge and update the corresponding strength of each edge
  - Recursively apply the above two steps until a network is decomposed in a set of components
  - Each component is a **community**

- Edge betweeness: the key element of the Girvman Newman algorithm
  - Measures the bridgness of an edge between two communities. The higher is the measure for an edge, the more it bridges between commuities

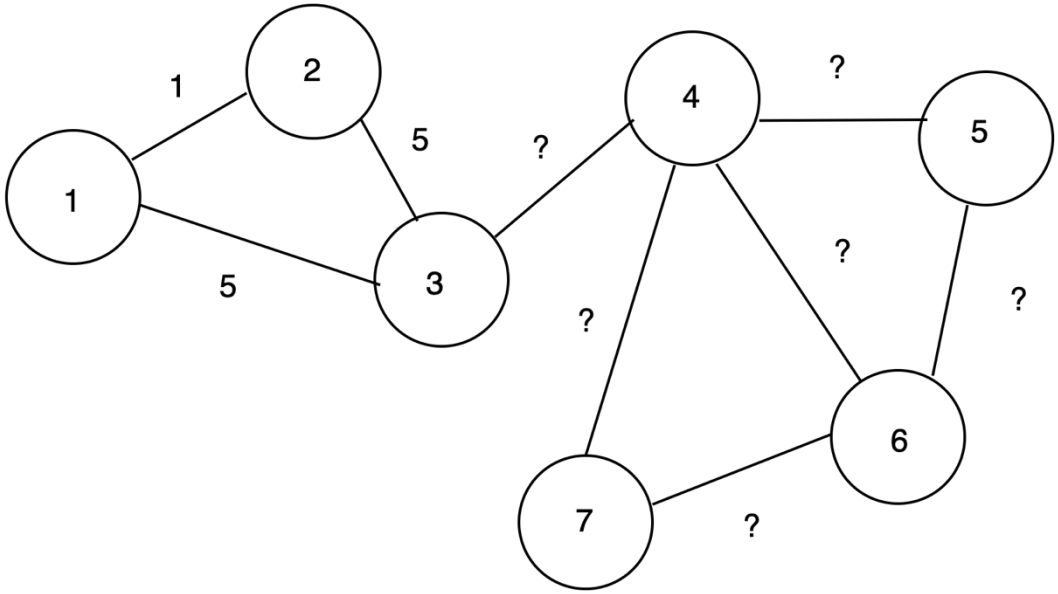


Hierarchical clustering

- Edge betweeness

  Remember **node betweeness**: number of shortest paths passing through a node. **Edge betweeness** is the the number of shortest paths passing through the edge. If there are multiple paths of equal length, then split counts.

Naïve count



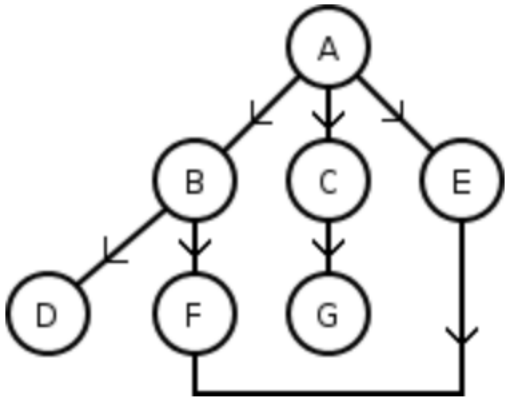
Hierarchical clustering

- Beyond the naïve count: need of an algorithm
- How to compute **edge betweeness**?
  - Compute betweenness of paths starting at node n
  - **Breath first search (BFS) starting from n**
  - Count the number of shortest paths from n to all other nodes of the network
  - Compute edge betweenness score by working up the tree:
    - ✓ If there are multiple paths count them fractionally
  - **Repeat the BFS procedure for each node of the network**
  - Add edge betweeness scores

Hierarchical clustering

- Breath first search (BFS)

  (see your course graph theory)



Source: Wikipédia

Algorithm 2.3 Breadth-First Search (BFS)

Require: Initial node v, graph/tree $G(V, E)$, queue Q

```
1: return An ordering on how nodes are visited
2: Enqueue v into queue Q;
3: visitOrder = 0;
4: while Q not empty do
5:     node = dequeue from Q;
6:     if node not visited then
7:         visitOrder = visitOrder +1;
8:         Mark node as visited with order visitOrder; //or print node
9:         Enqueue all neighbors/children of node into Q;
10:    end if
11: end while
```

Source: ©Book Social Media and Mining

Hierarchical clustering

- The Girvan-Newman algorithm

Newman-Girvan, 2004

Algorithm: Edge Betweenness

Input: graph G(V,E)

Output: Dendrogram/communities

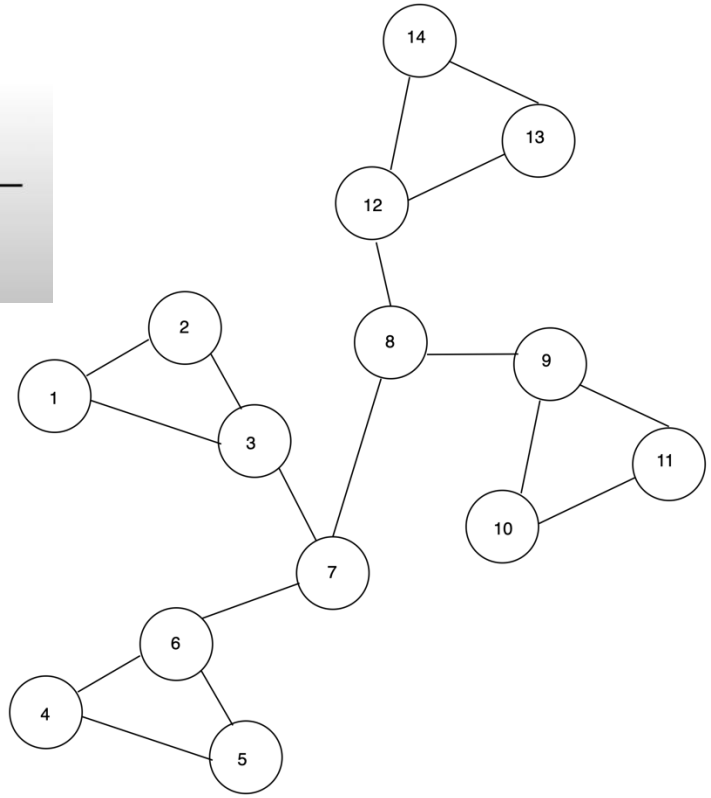
repeat

  For all $e \in E$ compute edge betweenness $C_B(e)$;

  remove edge e_i with largest $C_B(e_i)$;

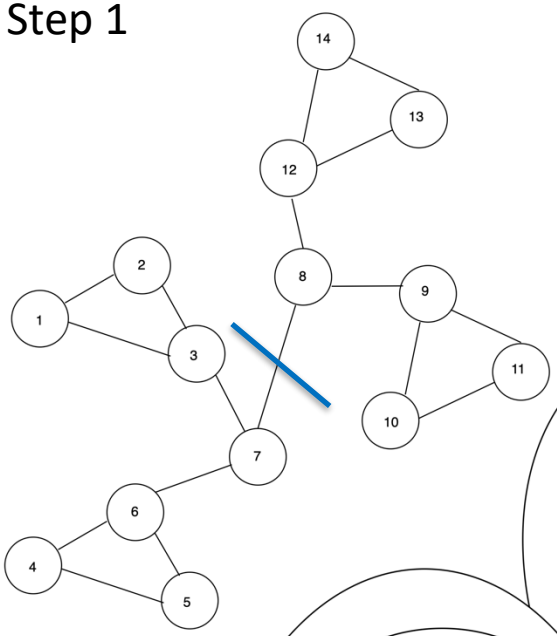
until *edges left*;

If bi-partition, then stop when graph splits in two components (check for connectedness)

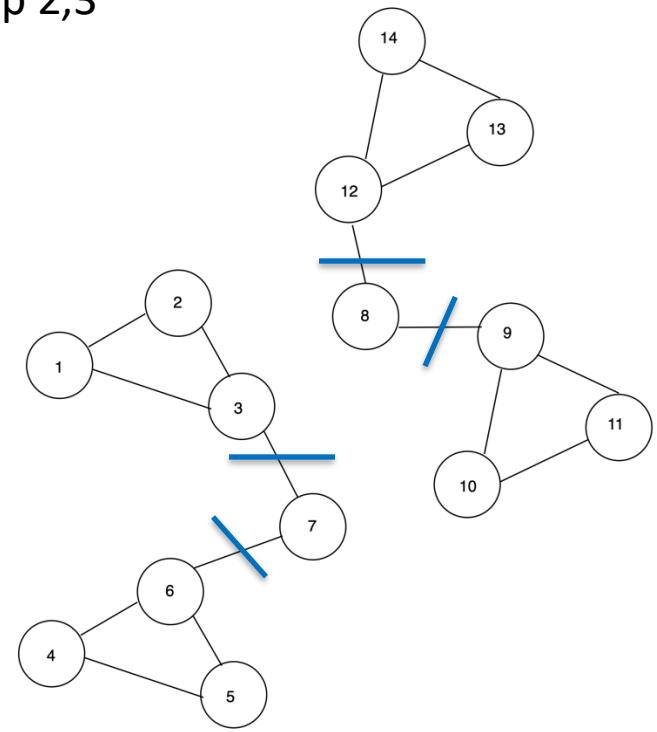


Hierarchical clustering

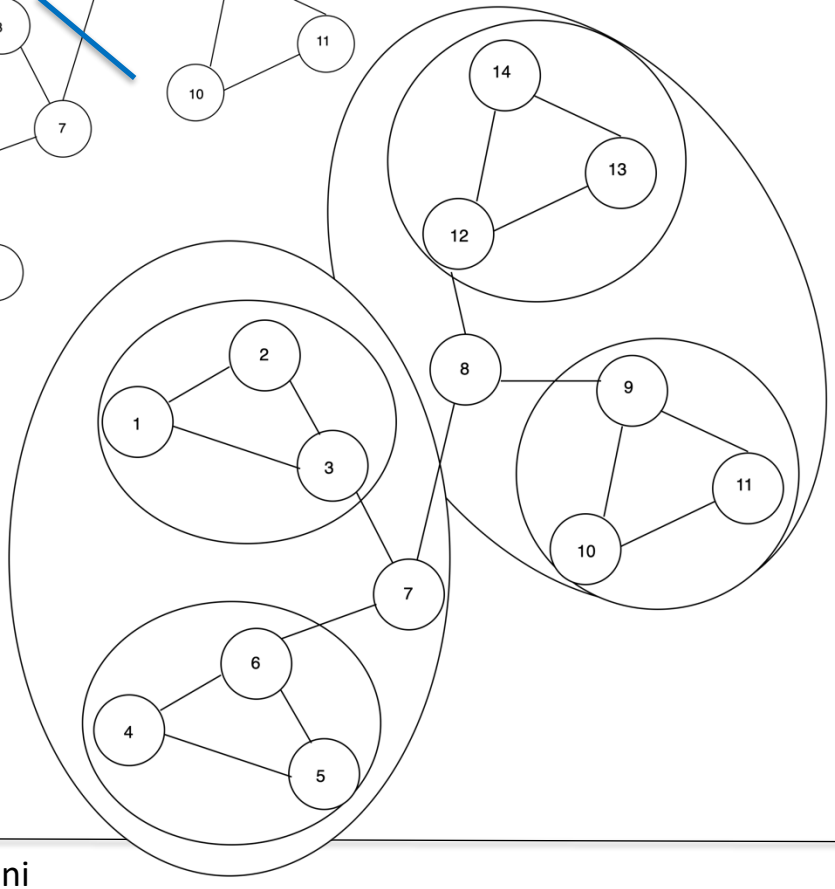
Step 1



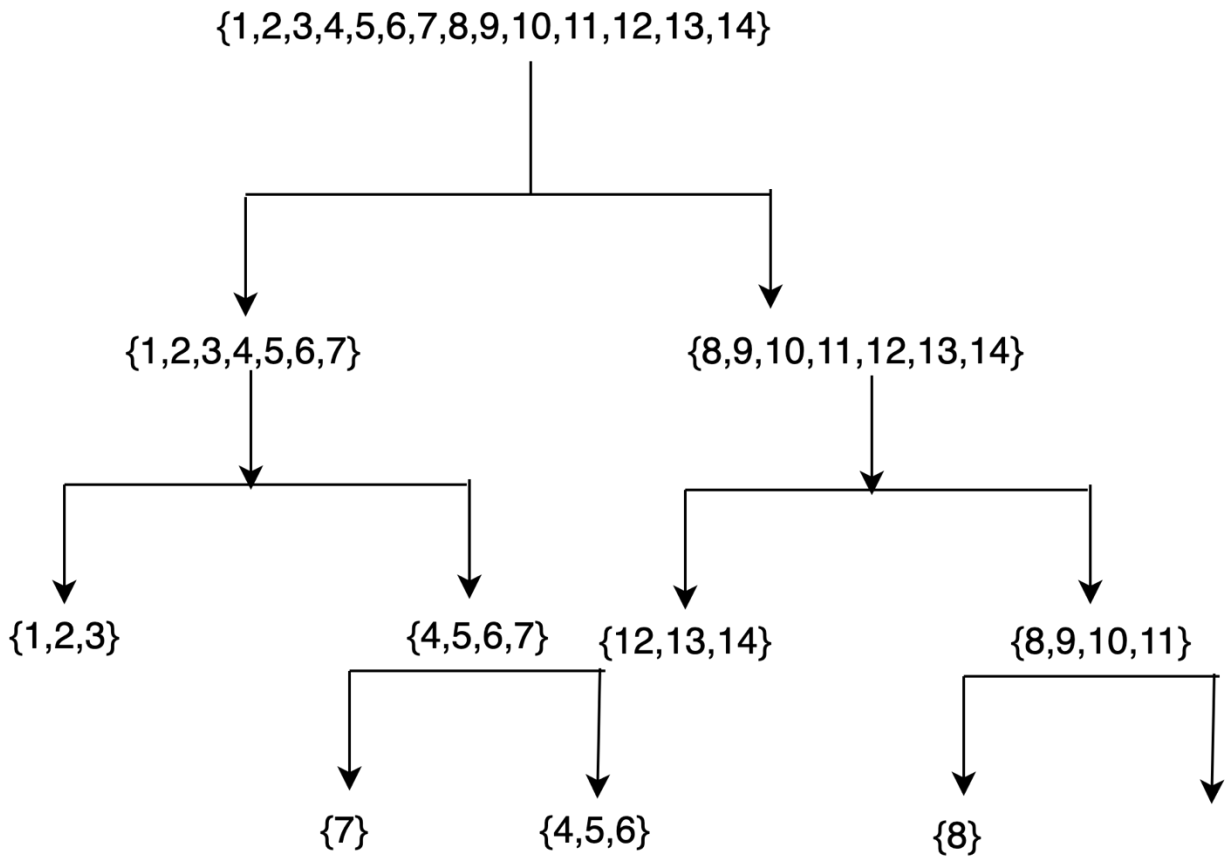
Step 2,3



Step 4



Hierarchical clustering



- Weaknesses

  Edge betweeness requires high complex computation

  One removal of an edge will lead to the recomputation of betweeness for all edges

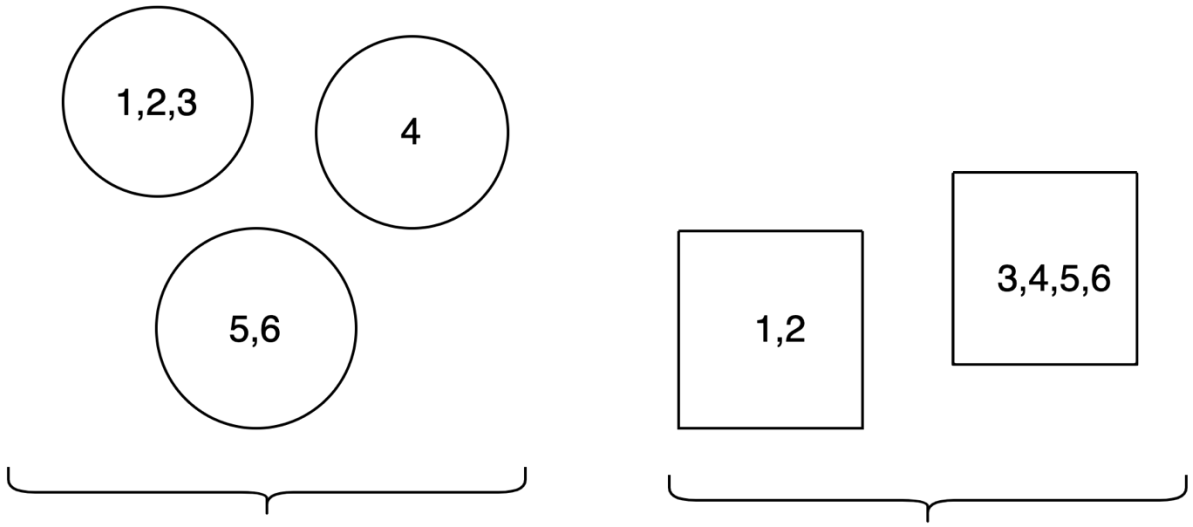
Evaluation of community detection

- Objective: measure in what extent the automatically built nodes' communities are "realistic" ie. map with **real-world communities.**
  - ✓ Assignment node-community (ies) is **known a-priori**: **ground truth is available**
  - ✓ Assignment node-community (ies) is **unknown a-priori**. Ground truth **is not available.**

- Two main evaluation approaches
  - Evaluation **with ground truth**
    - ✓ Recall, precision, F-measure
    - ✓ Normalized Mutual Information (NMI), Entropy
  - Evaluation **without ground truth**
    - ✓ Evaluation of the structure: based on clustering quality measures
    - ✓ Extrinsic evaluation using external tasks using the detected communities

Evaluation with ground truth

- Evaluation setting
  - The number of automatically detected communities can be different from the ground truth
  - No clear community mapping between the automatically detected communities and the ground truth



Ground Truth

Clustering result

Evaluation with ground truth

- Measures
  - Precision (P), Recall (R), F-Measure, Accuracy: pairwise community memberships

		Ground truth	
		Same community	Not same community
Clustering result	Same community	TP	FP
	Not same community	FN	TN

TP : True Positive FP : False Positive FN : False Négative TN : True Negative

$$P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}$$

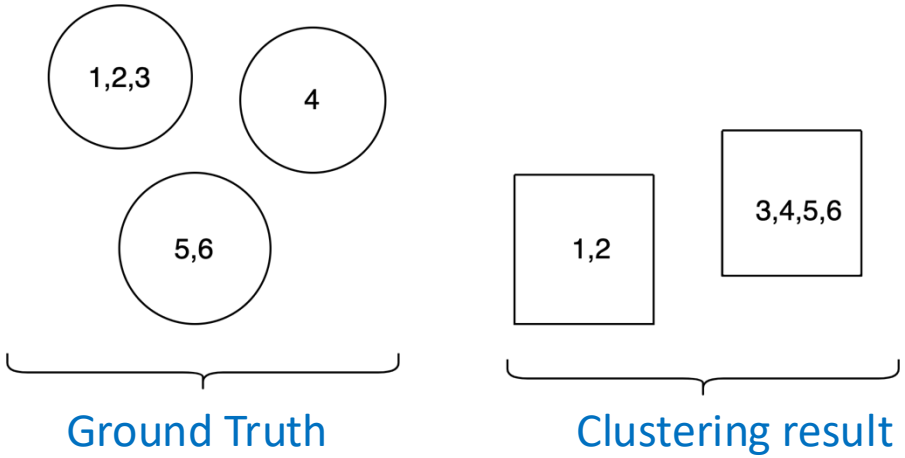
$$F - measure = \frac{2 \times P \times R}{P + R}$$

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)}$$

  - True Positive Rate (TPR), False Positive Rate (FPR)

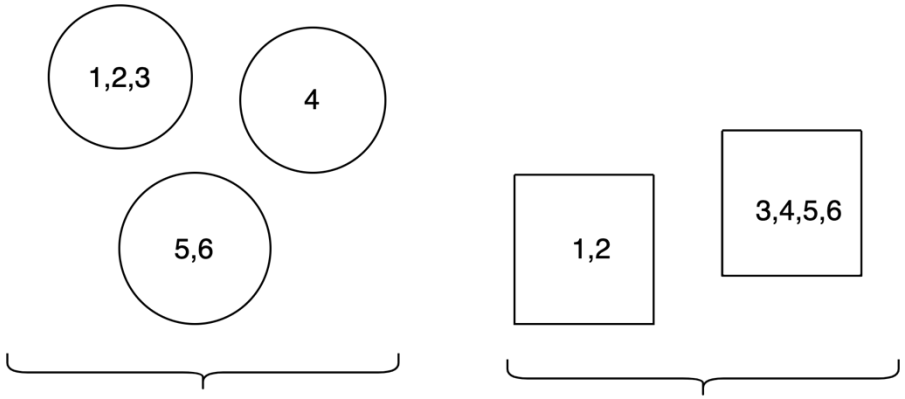
$$FPR = \frac{FP}{FP + TN} \qquad TPR = \frac{TP}{TP + FN}$$

Evaluation with ground truth



Community C		Ground truth	
		Same community	Not same community
Clustering result	Same community		
	Not same community		

Evaluation with ground truth



Community C		Ground truth	
		Same community	Not same community
Clustering result	Same community	2	4
	Not same community	2	6

1,2 ; 5,6 : TP
1,3;2,3 : FN
3,4; 4,5; : FP
1,4;1,5;1,6;2,4;2,5;2,6 : TN

P = 2/6
R= 2/4
F-measure= (2* 1/3* 1/2)/(1/3+1/2))

Accuracy=8/14
FPR=4/10
TPR =2/4

Evaluation with ground truth

- **Mutual information**

Measures the quantity of information hold by the partition Y by computing the matching between two partitions X and Y related to the clustering result and ground truth partitions. The mutual information is based on the entropy measure.

$$NMI(X;Y) = \frac{I(X;Y)}{\sqrt{H(X)H(Y)}}$$

– Let partition a (π^a), partition b (π^b), be the the clustering result and ground truth partitions. We compute the NMI measure as:

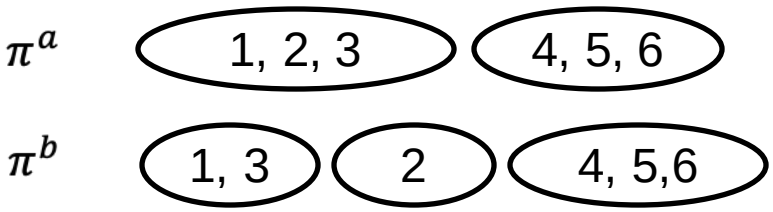
$$NMI(\pi^a, \pi^b) = \frac{\sum_{h=1}^{k^{(a)}} \sum_{\ell=1}^{k^{(b)}} n_{h,\ell} \log\left(\frac{n \cdot n_{h,l}}{n_h^{(a)} \cdot n_\ell^{(b)}}\right)}{\sqrt{\left(\sum_{h=1}^{k^{(a)}} n_h^{(a)} \log \frac{n_h^a}{n}\right)\left(\sum_{\ell=1}^{k^{(b)}} n_\ell^{(b)} \log \frac{n_\ell^b}{n}\right)}}$$

✓ The minimal value 0 corresponds to a random partition. The identified communities do not bring any information regarding the ground truth.

✓ The maximal value of H(π) is log n. Thus the IMN is low for k(a) or k(b) =n, total number of nodes in the graph.

Evaluation with ground truth

Partition a (π^a): [1, 1, 1, 2, 2, 2]
Partition b (π^b): [1, 2, 1, 3, 3, 3]



$n = 6$
$k^{(a)} = 2$
$k^{(b)} = 3$

	n_h^a
h=1	3
h=2	3

	n_l^b
l=1	2
l=2	1
l=3	3

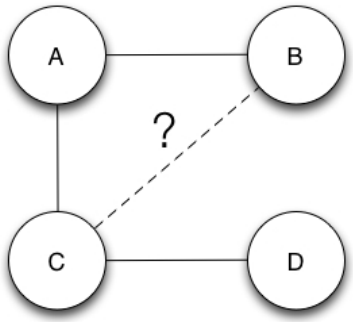
$n_{h,l}$	l=1	l=2	l=3
h=1	2	1	0
h=2	0	0	3

$$NMI(\pi^a, \pi^b) = \frac{\sum_{h=1}^{k^{(a)}} \sum_{\ell=1}^{k^{(b)}} n_{h,\ell} \log\left(\frac{n \cdot n_{h,l}}{n_h^{(a)} \cdot n_\ell^{(b)}}\right)}{\sqrt{\left(\sum_{h=1}^{k^{(a)}} n_h^{(a)} \log \frac{n_h^a}{n}\right)\left(\sum_{\ell=1}^{k^{(b)}} n_\ell^{(b)} \log \frac{n_\ell^b}{n}\right)}} \quad =0.8278$$

Link Prediction

- Link prediction task

  Predict **the new links (that would appear)** based on the existing links



- Some scenarios of link prediction in social networks
  - Predict a link **user-user** (e.g., follow). Given a user u and users he follows $N(u)$ at time t, predict if at time t' $(t'>t)$, user u would follow user u' such that u' do not belong to $N(u)$ at time t
  - Predict a link **user-item** (e.g. buy). Given a user at time t, infer behavioral links that are consistent with the user profile, network structure at time t' (t'>t)
  - Predict a link **user-community**. Given a community structure at time t, predict if user u will join a community at time t' (t'>t).

Link Prediction

- Practical formulation of the link prediction task: two main approaches
  - **Links missing**

✓Remove a **sample** set of links and then aim to predict them.

Various techniques of **sampling**:

- **Remove** randomly edges from the graph
  - Select only the edges that span a particular geodesic distance
- Select only the edges with a threshold degree

…

✓Predict the removed links

- **Links over time (if time is provided)**

- Choose a **threshold date t0'** to split data in 2
- Set Graph `G[t0..t0']` **before** the threshold date **and** Graph `G[t1..t1']` **after** the threshold date
- Predict the links that are in `G[t1..t1']` **but not in** `G[t0..t0']`

Link Prediction

- Main approaches
  - **Proximity-based measures**
    - Intuition: "*Nodes which are "close" belong to the same circle and likely to become linked themselves*"

      Link prediction heuristics measure how "close" nodes are

  - **Supervised learning**
    - Build a training set including nodes *v* with a) positive neighbours N(v) and b) negative neighbours $N^-(v)$
    - Generate features
    - Train a binary classification model: two labels (1: Link) vs. (0: No link)
    - Test the model on new nodes, graphs

Link Prediction – Proximity-based measures

- Methodology
  - For each pair of nodes `(u,v)`, compute score `Sim(u,v)` based on proximity-based measures
    - *Links missing: consider removed links*
    - *Links over time:* consider links in `G[t1..t1']`
  - Sort pairs `(u,v)` by the decreasing order of scores `Sim(u,v)`
  - Predict top `n` pairs `(u,v)`

Proximity-based link prediction

- Node-based similarity measures
  - *Common neighbours*: the more common neighbours that two nodes share, the more similar they are

$$\sigma(x, y) = |N(x) \cap N(y)|$$

  - *Jaccard score*: the likelihood of a node that is a neighbour of either x or y to be a common neighbour

$$\sigma(x, y) = \frac{|N(x) \cap N(y)|}{|N(x) \cup N(y)|}$$

  - *Adamic / Adar*: If two nodes share a neighbour and that neighbour is a rare neighbour, it should have a higher impact on their similarity

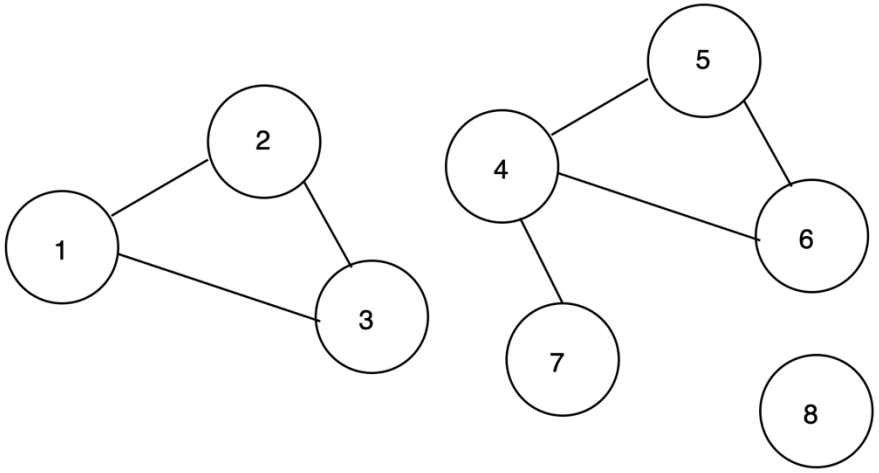
$$\sigma(x, y) = \sum_{z \in N(x) \cap N(y)} \frac{1}{log|N(z)|}$$

  - *Preferential attachment*: nodes of higher degree have a higher likeliness of getting connected to incoming nodes

$$\sigma(x, y) = |N(x)| \cdot |N(y)|$$

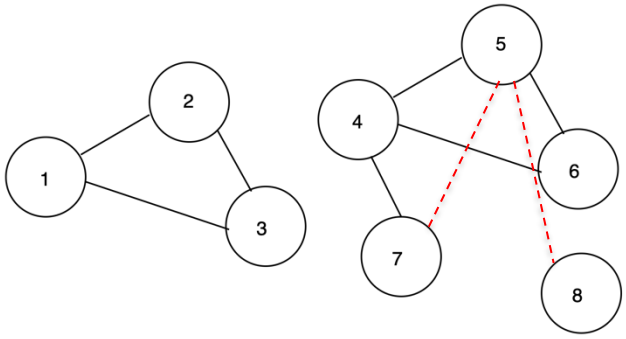
Similarity-based measures

Predict the more likely missing edges using node-based neighberhood measures



Similarity-based measures

Let us use the Jaccard and Adamic Adar measures for node 5



N_5={4,5,6}
N_7 ={4,7}
N_8={8}
……

Candidate missing edge(i,j)	Sim_{jacc}	Rank	Sim Adam/Adar	Rank
(2,4)	0		0	
(2,5)	?		?	
(2,6)	?		?	
(2,7)				
(2,8)	?		?	
(3,4)	?		?	
(3,5)				
(5,7)	1/4		1/log 3	
(5,8)	0		0	

Weakness

Node-based topological similarity measures (common neighbours, Jaccard, Adamic/Adar, preferential attachment) perform the best but do not scale well

Proximity-based link prediction

- Other measures could be used
  - Path-based measures (eg. SimRank)

$$\sigma(x,y) = \gamma \frac{\sum_{x' \in N(x)} \sum_{y' \in N(y)} \sigma(x', y')}{|N(x)||N(y)|}$$

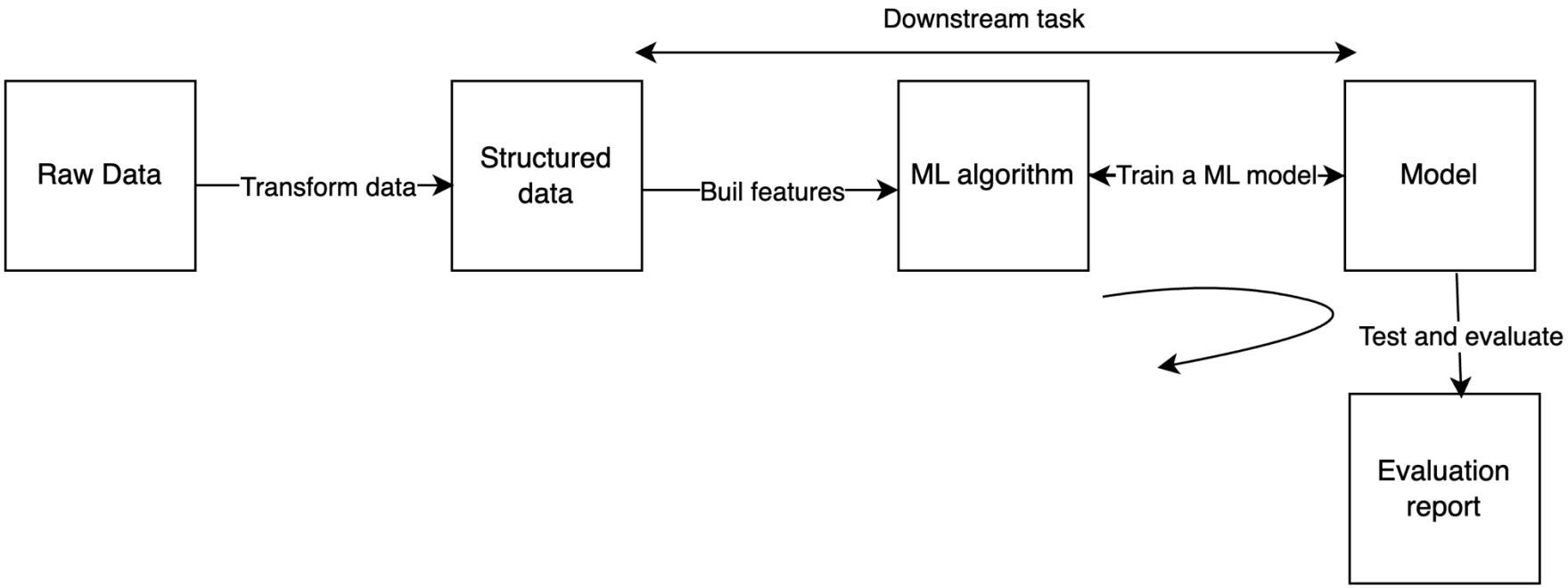
  - Global neighberhood-based measures (eg. Katz measure)

$$\sigma(x,y) = \sum_{l=1}^{\infty} \beta^l |paths_{x,y}^{<l>}|$$

  - ...

Link Prediction – Traditional supervised learning

- Methodology



Link Prediction – Supervised learning

- Methodology

1. ML training

- Build feature-based vectors of positive, negative examples on the training subgraph
- Which features?
  - ✓ Topological features: similarity, centrality, etc.
  - ✓ Aggregated features: belonging to same community?
  - ✓ Content-based proximity measures: likes, city, affiliation, etc.
- Train a classification model
  - ✓ Decision tree
  - ✓ Logistic regression
  - ✓ Neural model
  - ✓ …

2. Predict

- Classify test links
- Sort by decreasing order of positive labels' likelihood

Link prediction as a binary classification problem

- A challenging classification problem
  - A very large number of possible edges (quadratic in number of nodes)
  - Highly unbalanced class distribution
    - ✓ Positive examples: linear growth with the number of nodes
    - ✓ Negative examples : quadratic growth with the number of nodes

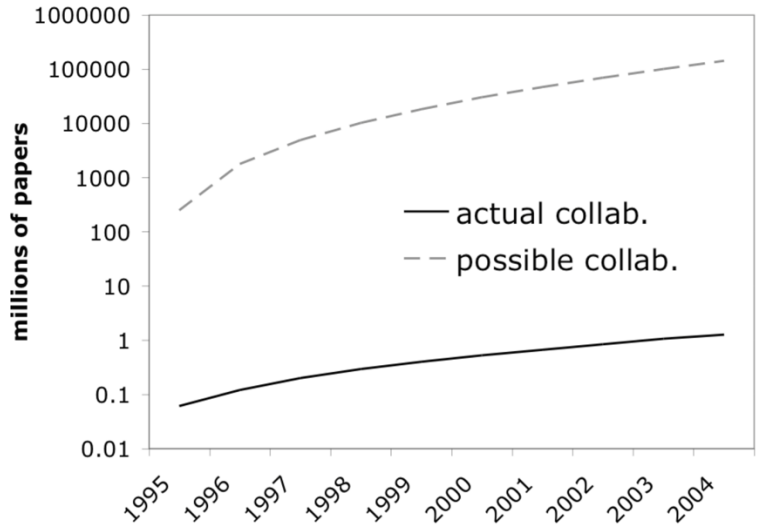


Figure 1. Logarithmic plot of actual and possible collaborations between DBLP authors, 1995-2004.

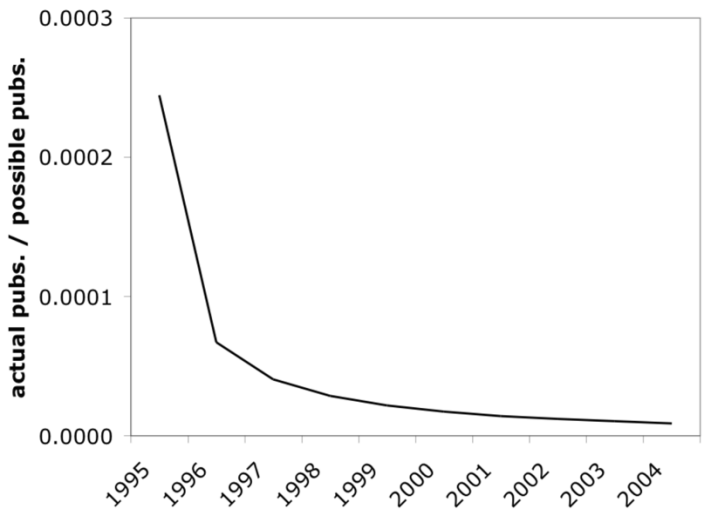


Figure 2. Publications of DBLP authors as a proportion of possible collaborations, 1995-2004.

Source: M. Rattigan, D. Jensen. The case for anomalous link discovery. ACM SIGKDD Explorations Newsletter. v 7, n 2, pp 41-47, 2005

Evaluation of Link prediction

- Traditional measures
  - ✓Accuracy

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)}$$

Basic evaluation of a classifier:

*Given a test set with **positive** and **negative** links, how many links are accurately classified?*

Issue: the test set has only positive examples. We need to add negative examples (pairs of nodes without a new link)

Issue of class imbalance! (refer to your course in machine learning)

Evaluation of Link prediction

- Traditional measures
  - Precision (P), Recall (R)

$$P = \frac{TP}{TP + FP} \qquad R = \frac{TP}{TP + FN}$$

  - How to measure?
    Consider top k links and then compute P, R (and Accuracy)

  - Other measures: Average Precision (AP), Mean Average Precision (MAP)
    - For each node, rank the links based on a score (eg. similarity)
    - Compute the precision P at each correctly ranked edge
    - Compute the average precision AP
    - Average AP over K nodes, obtain MAP

K increases, P tends to decrease and R tends to increase, thus, FN transformed in TP or FN

Evaluation of Link prediction

Node 1: Total Expected true edges: 5

Rank of predicted edge	Edge	R	P
1	TP	1/5= 0.2	1/1=1
2	TN		
3	TN		
4	TP	2/5=0.4	2/4= 0.5
5	TP	3/5= 0.6	3/5= 0.6
6	TN		
7	TP	4/5=0.8	4/7= 0.57
8	TN		

AP = (1+0.5.+ 0.6+0.57)/4
= 0.66

Node 2: Total Expected true edges:4

Rank of predicted edge	Edge	R	P
1	TP	1/4= 0.25	1/1=1
2	TP	2/4=0.5	2/2=1
3	TN		
4	TN		
5	TP	3/4= 0.75	3/5= 0.6
6	TN		
7	TP	4/4=1	4/7= 0.57
8	TN		

AP = (1+1+ 0.6+0.57)/4
= 0,79

MAP = (0.66+0.79)/2
= 0.72