NETWORK ANALYSIS & MINING

unit - 2

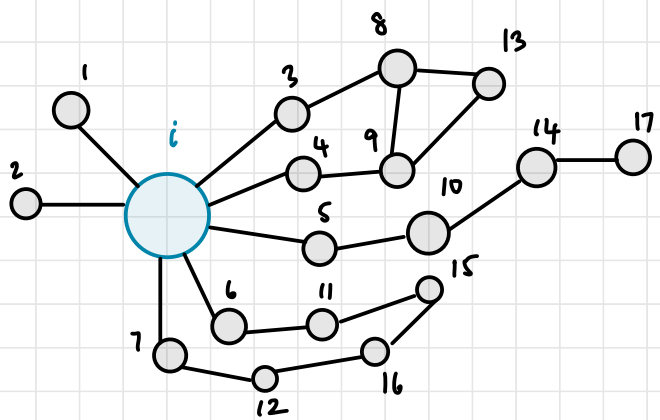
feedback/corrections: vibha@pesu.pes.edu

VIBHA MASTI

Centrality Recap

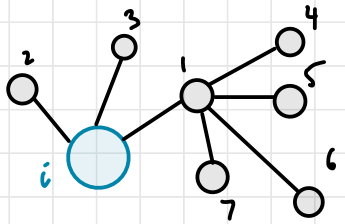
	Low Degree	Low Closeness	Low Betweenness
High Degree		Embedded in cluster that is far from the rest of the network (a)	Ego's connections are redundant - communication bypasses him/her
High Closeness	Key player tied to important important/ active alters (b)		Probably multiple paths in the network, ego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

(a) High degree, Low closeness: embedded in a cluster far from the rest of the network



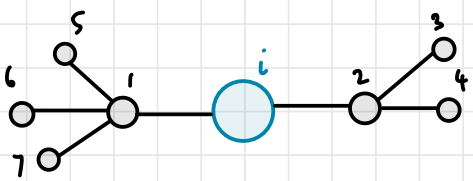
$d_i = 7$ $\quad C_i = \frac{17}{33} = 0.515$

(b) High closeness, low degree: tied to important alters / key player



$d_i = 3$ $\quad c_i = \frac{7}{11} = 0.636$

(c) High betweenness, low degree: ego's few ties crucial



$b_i = 12$ $\quad d_i = 2$ $\quad c_i = \frac{1}{12} = 0.583$

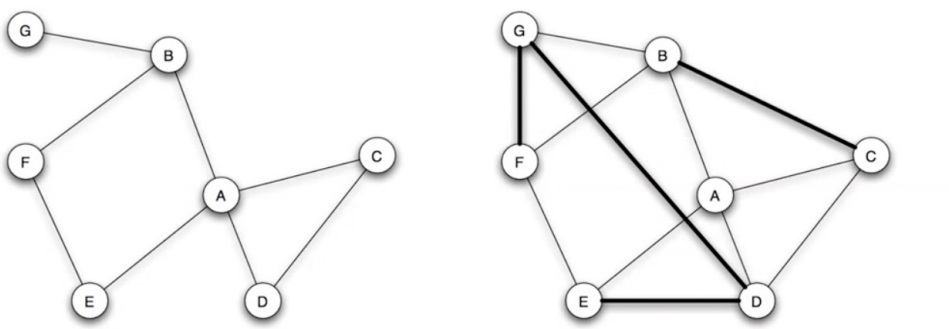
Network Formation First Steps – Dyad, Triad, Bridge

Dyad

- 2 actors and ties linking
- Kind of relation
- Dyad census – properties of tie
  - mutual
  - asymmetric
  - null (no tie)
- Connections between 2 nodes; ties:
  - Granovetter's strength of weak ties (strength)
  - Multiplexity of tie (multiple ties with another node)

Triadic Closure

- If 2 people have a common friend, increased likelihood of them becoming friends



(a) *Before new edges form.* (b) *After new edges form.*

BC, GF, ED result of triadic closure
GD not a result of triadic closure

Types of Closure

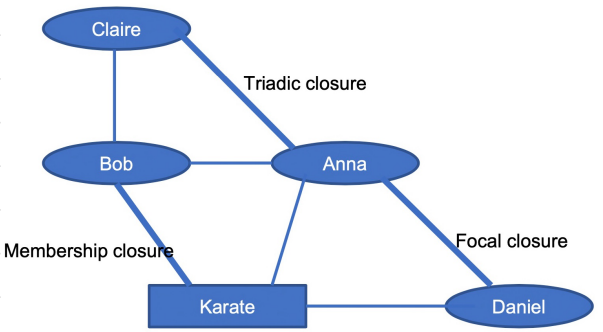
Triadic Closure
- common friend
- A–B, B–C; A–C becomes

Foci closure
- two individuals with shared interests connect
- two students in class preparing for GRE/GATE become close

Membership closure
- existing connections influenced to join organisation
- members of the same club/activity
- Google Dev Club, PIL – encourage friends to join

Co-Evolution of Social Network

- All types of triadic closure together



  - Bob intros Anna to Claire (triadic closure)
  - Karate club intros Anna to Daniel (focus: Karate)
  - Anna intros Bob to Karate (membership closure)

- Co-evaluation of social affiliation network

Triadic Closure & Common Friends

- Observed that probability of triadic closure grows linearly with number of friends (email communicatim dataset)

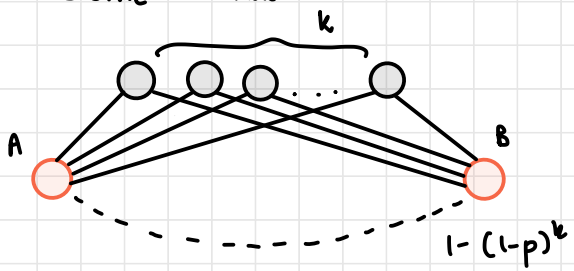
- p = Probability that A & B become friends when they have one common friend

- 1-p = Probability of not becoming friends

- $(1-p)^k$ = Probability of not becoming friends after having k common friends

- $1-(1-p)^k$ = Probability of A & B becoming friends after having k common friends

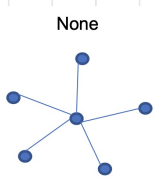
- Pressure to become friends

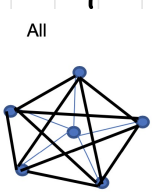


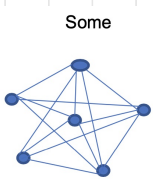
Triadic Closure and Density of Ego's Friendship Network

- Two extreme scenarios:
  1. My friends do not know each other
  2. Each of my friends knows the other

- Strength of friendship network
  - Ratio of no. of connections to total possible no. of connections between a node's friends/connections





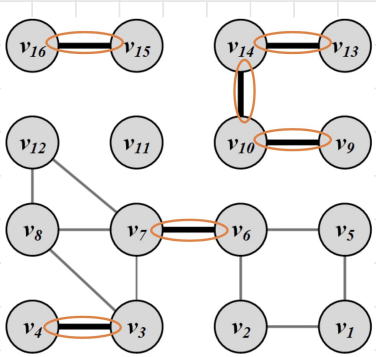


n = 9

strength of friendship = $\frac{9}{10}$

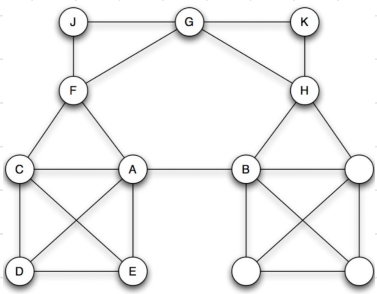
Bridges

- Bridges: edges whose removal will increase number of connected components



Local Bridge

- Removal of local bridge increases the distance between A and B



- It is called a local bridge if they have no nodes in common
- Span of local bridge: no. of hops to connect two end nodes if edge removed

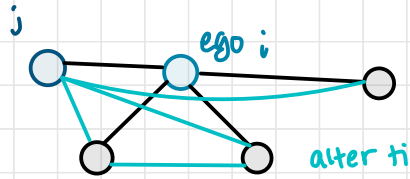
Measuring Cohesion

1. Shortest path: path between 2 nodes with shortest length
   - v_i & $v_j \rightarrow \ell_{ij}$
2. n-Hop neighbourhood: of a node; set of nodes within n hops in shortest path from the node
   - adjacency matrix to n^{th} power: n-hop connections and no. of shortest paths
3. Density: proportion of dyadic connections present

$$\text{Density}_{undir} = \frac{L}{n(n-1)/2} \qquad \text{Density}_{dir} = \frac{L}{n(n-1)}$$

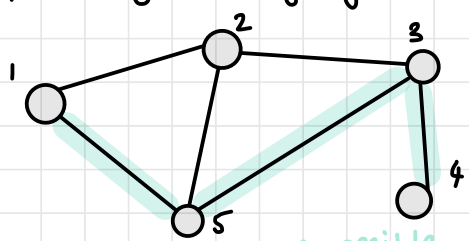
Problems:
- Decreasing density with increase in network size
- Increasing density with increase in cohesive sub-groups
- Ties mainly from a single person

ties mainly through j



Density(i) = $\frac{4}{6}$

4. Diameter and average path length
   - Diameter: longest geodesic/ longest shortest path between any 2 nodes
   - Average path length: average geodesic

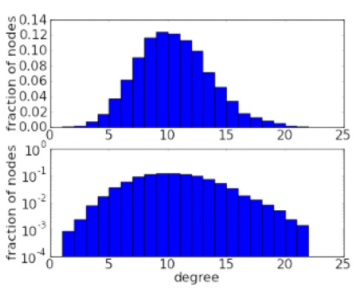


diameter =
max(1,2,3,1,1,2,1,1,1,2)

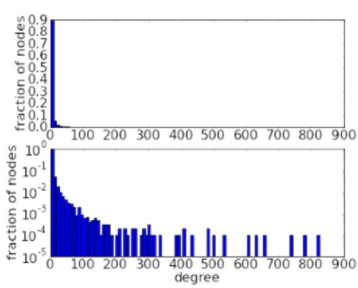
= 3

Degree Distribution

- Probability distribution of the degrees over the whole network
- p_x = fraction of vertices in a network with degree x



A binomial degree distribution of a network with 10,000 nodes and average degree of 10. The top histogram is on a linear scale while the bottom shows the same data on a log scale.



A power law degree distribution of a network with 10,000 nodes and average degree of around 7. The top histogram is on a linear scale while the bottom shows the same data on a log scale.

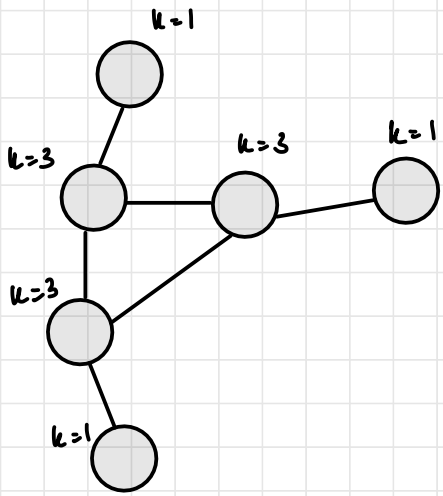
Entropy of Degree Distribution

$H = \sum - p_k \log p_k$

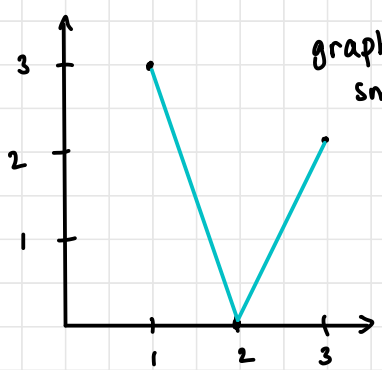
high entropy - random

ASSORTAVITY

- Preference for a network's nodes to attach to others that are similar
- Disassortative: attach to others that are different
- Plot of average degree of neighbours of a node vs degree k of node
- Eg:



- Assortative: increasing slope
- Disassortative: decreasing slope



graph too small

degree	avg degree of neigh
1	3
2	0
3	7/3

Assortative Mixing

- Assortative Mixing coefficient r
- Perfect assortativity: normalised associativity matrix with 1's on the diagonal (r=1)
- Perfect disassociativity: all diagonal elements 0 (r=-1)

Figure 8.2. A U.S. High School Friendship Network in 1994 between Races. Eighty percent of the links exist between members of the same race (from [Currarini et al., 2009]).

8.1.1 Measuring Assortativity for Nominal Attributes

Consider a scenario where we have nominal attributes assigned to nodes. As in our example, this attribute could be race or nationality, gender, or the like. One simple technique to measure assortativity is to consider the number of edges that are between nodes of the same type. Let $t(v_i)$ denote the type of node v_i. In an undirected graph[2] , $G(V, E)$, with adjacency matrix A, this measure can be computed as follows,

$$\frac{1}{m} \sum_{(v_i, v_j) \in E} \delta(\, t(v_i), t(v_j)\,) = \frac{1}{2m} \sum_{ij} A_{ij}\, \delta(\, t(v_i), t(v_j)\,), \tag{8.1}$$

where m is the number of edges in the graph, $\frac{1}{m}$ is applied for normalization, and the factor $\frac{1}{2}$ is added because G is undirected. $\delta(.,.)$ is the Kronecker delta function:

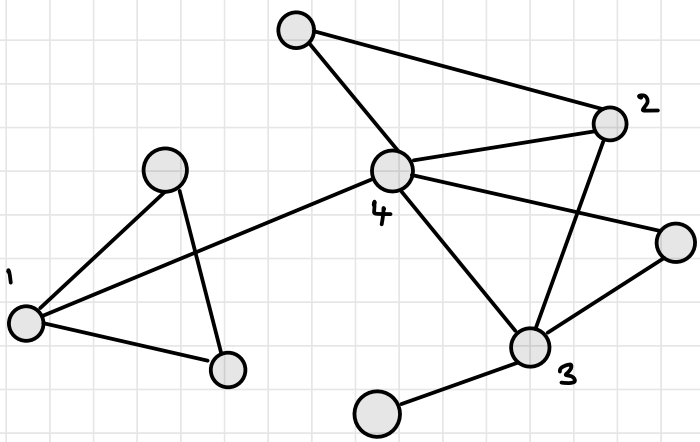
$$\delta(x, y) = \begin{cases} 0, & \text{if } x \neq y; \\ 1, & \text{if } x = y. \end{cases} \tag{8.2}$$

Rich Club

- Nodes with many connections are clustered together

- Rich club coefficient for a degree k
  - n = no. of nodes with degree > k
  - Rich Club Coefficient = $\frac{\text{no. of edges between n nodes}}{\text{max no. of edges between n nodes}}$

k = 3

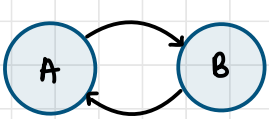


coefficient $= \frac{4}{6}$

Reciprocity & Transitivity

1. Reciprocity

- Measure of likelihood of vertices in a directed network to be mutually linked - dyadic relationship
- Closed loops of length 2
- Only in directed graphs
- Eg: mutual followers on Twitter, Tumblr, Instagram



- Reciprocity: no. of reciprocal pairs in the graph / max

$$R = \frac{\sum_{i,j,i<j} A_{ij} A_{ji}}{\lfloor \frac{|E|}{2} \rfloor} = \frac{2}{|E|} \times \frac{1}{2} Tr(A^2)$$

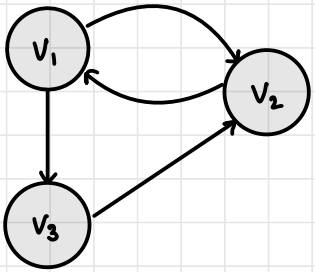
trace (sum of diagonal)

max reciprocity → $\lfloor \frac{|E|}{2} \rfloor$ ← number of edges

where $Tr(A^2) = A_{1,1} + A_{2,2} + \ldots + A_{n,n}$

- A^2 stores the 2-hop adjacency matrix. Diagonal elements are nodes with 2-hop closed loops

- Eg:



$$A = \begin{matrix} \\ \\ \\ \end{matrix} \begin{matrix} \text{dest} \\ \end{matrix} \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$A^2 = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

$|E| = 4$

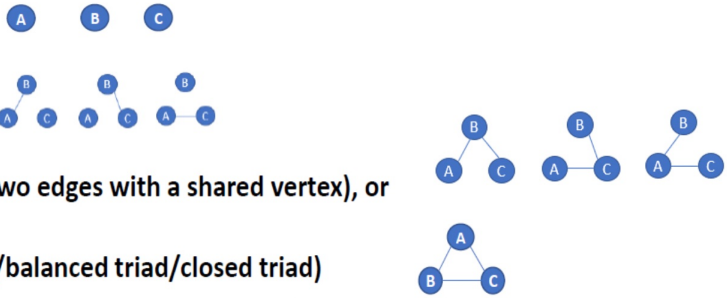
$Tr(A^2) = 2$

$\therefore R = \frac{2}{4} = \frac{1}{2}$ (fraction of edges in net that are reciprocal)

2. Transitivity

- Undirected graph: 4 possible triadic relationships

1. no ties,
2. one tie,
3. two ties (open triad – two edges with a shared vertex), or
4. all three ties(transitive/balanced triad/closed triad)



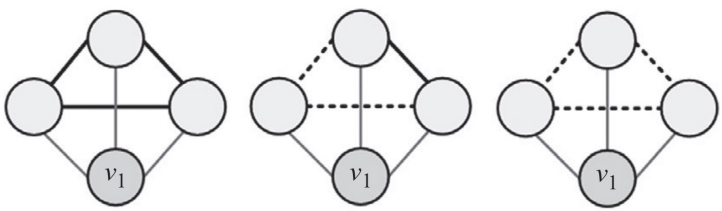
- Count of relative prevalence of 4 types of relations across all possible triples : sense of population
  - isolation, couples only, structural holes, clusters
- Transitive linking: if $v_1 \to v_2$ and $v_2 \to v_3$ lead to formation of $v_3 \to v_1$, transitive linking
- If a friend of my friend is also my friend
- Higher transitivity ⇒ denser graph ⇒ closer to complete graph (all nodes linked to all other nodes)
- Measuring transitivity : global & local clustering coefficients
- Real world: friendships highly transitive (high average local clustering coefficients)

(a) Local Clustering Coefficient (Watts and Strogatz)

- Transitivity at the node level (undirected graphs)
- How strongly pairs of neighbours of a node are connected

$$C(v_i) = \frac{\text{no. of pairs of neighbours of } v_i \text{ that are connected}}{\text{no. of pairs of neighbours of } v_i}$$

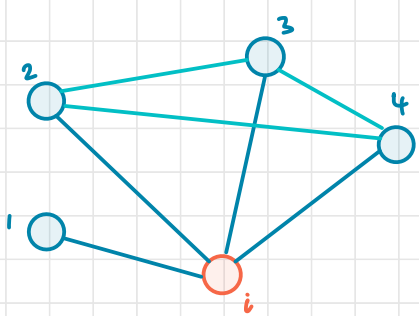
$d_i C_2$



$C(v_1) = 1$ $C(v_1) = 1/3$ $C(v_1) = 0$

Figure 3.10. Change in Local Clustering Coefficient for Different Graphs. Thin lines depict connections to neighbors. Solid lines indicate connected neighbors, and dashed lines are the missing connections among neighbors.

Q: Find local clustering coefficient of v_i



$$C(v_i) = \frac{3}{{}^4C_2} = \frac{3}{6} = \frac{1}{2}$$

- Low local clustering coefficient of v_i $\Rightarrow$ v_i controls flow of info & is influential (more structural holes around v_i)

(b) Global Clustering Coefficient (Watts and Strogatz)

- Average local clustering coefficient of all nodes

$$C_{WS} = \frac{1}{n} \sum_{i=1}^{n} C(v_i)$$

- High clustering coefficient : networks dominated by vertices of low degree

(c) Global Clustering Coefficient (Newman, Strogatz and Watts)

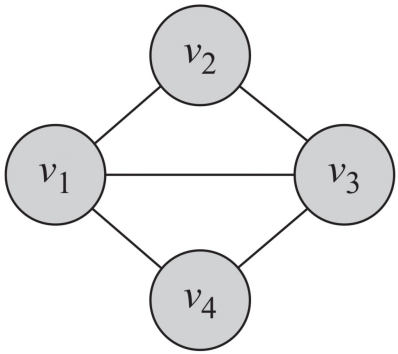
- Closed path of length 2: fully connected triangle
  - $v_1 \leftrightarrow v_2$, $v_2 \leftrightarrow v_3$, $v_3 \leftrightarrow v_1$ (undirected)
- Clustering coefficient

$$C = \frac{|\text{closed paths of length 2}|}{|\text{paths of length 2}|}$$

- Alternative calculation: every triangle has 3 closed paths of length 2

$$C = \frac{\text{number of triangles} \times 3}{\text{number of connected triples}}$$

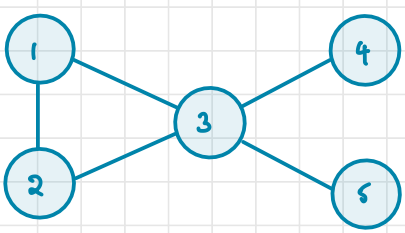
Q: Find global (Newman, Strogatz and Watts) clustering coefficient of



$$C = \frac{\text{no. of triangles} \times 3}{\text{no. of triples}} = \frac{2 \times 3}{2 \times 3 + 2} = \frac{6}{8} = \frac{3}{4}$$

$v_2v_1v_4, v_2v_3v_4$

Q: Find local clustering coefficient of node 3, the global (avg) clustering coefficient and the global (NSW) clustering coefficient



$$CC(1) = \frac{1}{{}^2C_2} = 1$$

$$CC(2) = \frac{1}{{}^2C_2} = 1$$

$$CC(3) = \frac{1}{{}^4C_2} = \frac{1}{6}$$

$$CC(4) = 0$$

$$CC(5) = 0$$

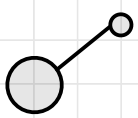
$$CC_{avg} = \frac{1}{5}\left(1 + 1 + \frac{1}{6} + 0 + 0\right) = \frac{13}{30} = 0.43$$

$$CC_{NSW} = \frac{1 \times 3}{1 \times 3 + 5} = \frac{3}{8}$$

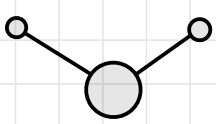
Efficient Way to Find Triples

- For a node i in an undirected graph:
  - (a) $d_i = 1$, triples centered at $i = 0$
  - (b) $d_i = 2$, triples centered at $i = 1$
  - (c) $d_i = 3$, triples centered at $i = 3$
  - (d) $d_i = 4$, triples centered at $i = 6$
  - (e) $d_i = n$, triples centered at $i = \frac{n(n-1)}{2}$ or ${}^nC_2$

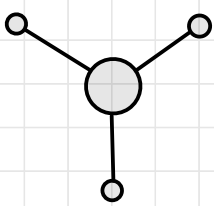
(a)



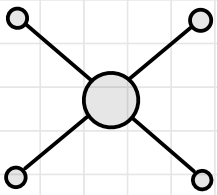
(b)



(c)



(d)



GROUPS & SUBSTRUCTURES

- Identifying substructures within a network

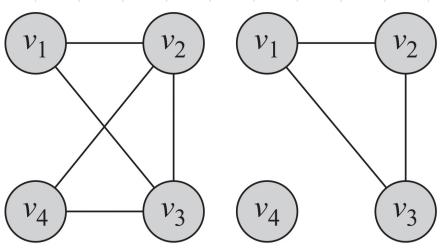
- Approaches
  - Top down
  - Bottom up

1. TOP DOWN APPROACH

- Look at entire network as a whole and identify sub-structures
- Holes / weak spots in the overall network structure define lines of division
- Finding these weak spots: undirected and directed graphs

(a) Undirected Graph — connected & disconnected components

- If a pair of vertices has no path between them, the network is termed as disconnected (else: connected)
- Component: maximal subset of network that is connected (every pair of vertices has a path connecting them)

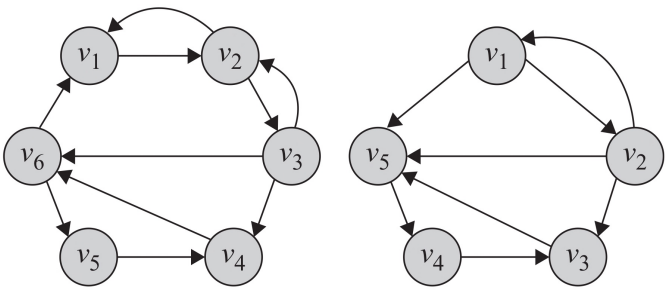


(a) Connected (b) Disconnected

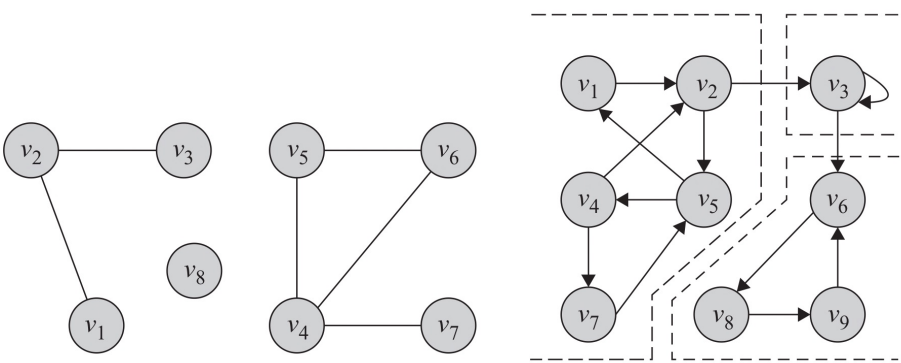
(b) Directed Graph – strongly connected and weakly connected

- Components can be either strongly connected or weakly connected
- Strongly connected: every pair of vertices u and v has a directed path from u to v as well as from v to u

- Weakly connected: if replacing all the directed edges with undirected ones leads to the formation of a connected component



(c) Strongly connected (d) Weakly connected



(a) A Graph with Three Components (b) A Graph with Three Strongly Connected Components

Figure 2.10. Components in Undirected and Directed Graphs.

- Out components of node A: set of vertices reachable from node A via directed paths
- In components of node A: set of vertices (including A) from which there is a directed path to A
- All members of strongly connected component have same in-component

- All members of strongly connected component of A are members of A's out-component

The Bow-Tie Structure of the Web

- Web divided (broadly) into large pieces

- Pieces
  (a) One giant strongly connected component (SCC)
  (b) In: nodes upstream of the SCC – can reach the SCC but cannot be reached from it
  (c) Out: nodes downstream of the SCC – can be reached from the SCC but cannot reach the SCC

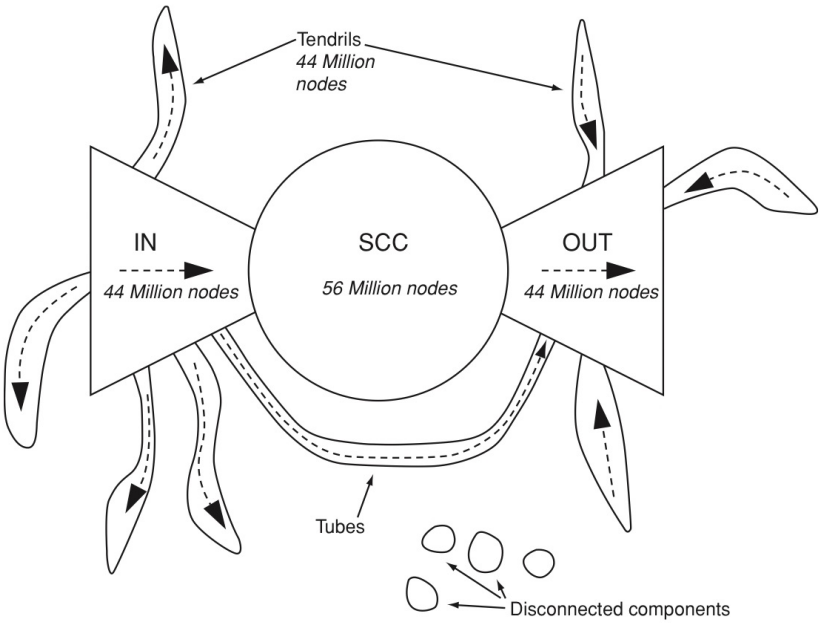


Figure 13.7. A schematic picture of the bow-tie structure of the Web (image from Broder et al., [80]). Although the numbers are now outdated, the structure has persisted.

GIANT COMPONENT

- Real, undirected networks: typically one large component - giant component - and large number of small components

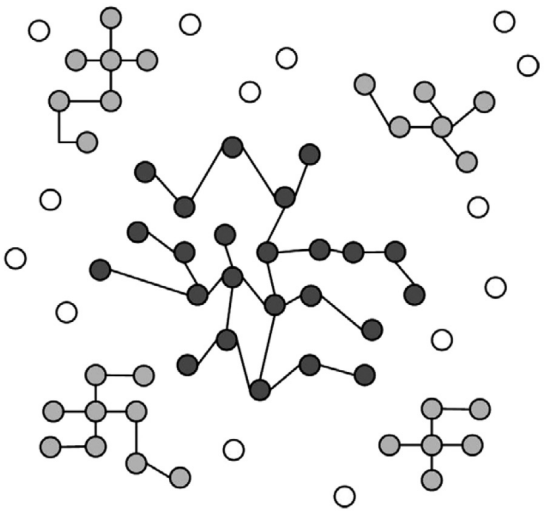


Figure 6.11. Network Segmentation. The network is decomposed into a giant component (dark gray), star components (medium gray), and singletons (light gray).

Preferential Attachment

- Albert-László Barabási, 1999 : most complex networks evolve as a result of preferential attachment (rich get richer)

- New node joining a network : likely to attach to a node with high degree
  - connection likelihood α degree of target node

Blocks and Cut Points

- Finding weak spots - if a node were removed, would the structure become divided?

- Called cut points; divide network into blocks

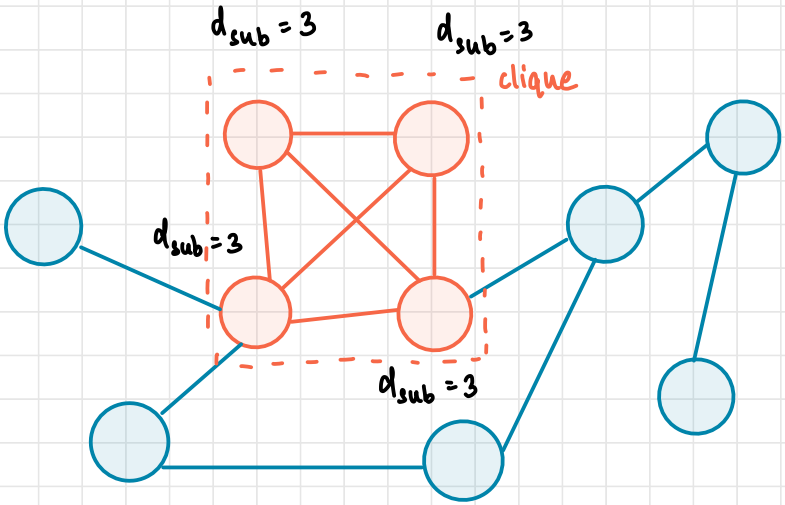
2. BOTTOM UP APPROACH

- How dense connections are built up from dyads and triads

Clique

- Maximal subset of vertices in a graph that is complete (all nodes directly connected to all other nodes)

- Clique of k members: all members in subgraph have a degree of k-1 (if subgraph is seen as an independent graph)



- Smallest clique: dyad

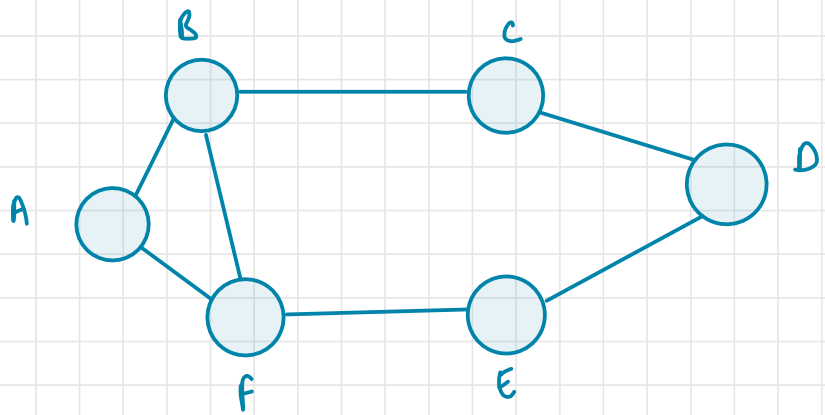
- Cliques can overlap

- Definition of clique is too rigid; can relax restriction:
  1. N-clique / N-clan
  2. k-plex / k-core

1. N-Clique

sometimes enforced, sometimes relaxed

- Any (maximal) set of S nodes where the geodesic path between every pair of nodes is $\leq N$
- Strict clique : 1-clique (all nodes connected to all other nodes in a subgraph)



- {A,B,F} : 1-clique
- {C,D,E} : 2-clique (not max)
- {B,C,D,E} : 2-clique (not max) notice: path E-F-B not in subgraph even though it is geodesic
- {F,B,C,D,E} : maximal 2-clique
- {A,B,C,D,E,F} : maximal 3-clique

2. K-clan

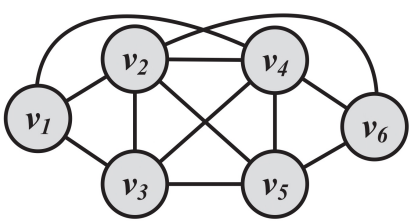
- Restrict k-cliques : all geodesics of subgraph pass through subgraph

- All ties among actors occur through other members of the group
- Diameter $\leq k$
- For same graph, $\{B, C, D, E\}$ is not a 2-clan
- $\{B, C, D, E, F\}$: 2-clan

3. k-plex

- All nodes have minimal degree of $|S| - k$ where $|S|$ is the number of nodes in the subgraph of vertices S
- If $k=1$, all nodes in S have a degree d_s of at least $|S| - 1$ (strict clique / clique)

$$d_s \geq |S| - 1$$



1-plex : $\{v_2, v_3, v_4, v_5\}$

2-plex : $\{v_1, v_2, v_3, v_4, v_5\}, \{v_2, v_3, v_4, v_5, v_6\}$

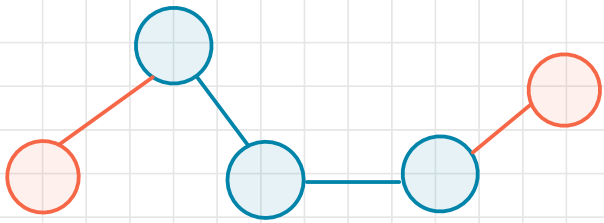
3-plex : $\{v_1, v_2, v_3, v_4, v_5, v_6\}$

Figure 6.5. Maximal k-plexes for $k = 1$, 2, and 3.

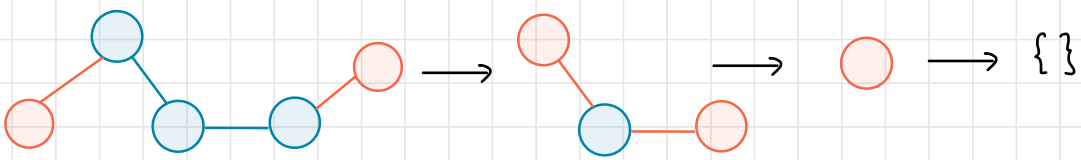
4. k-core

- All nodes in subgraph have a degree of at least k
- 0-core: all graphs are 0-cores (even isolated nodes)

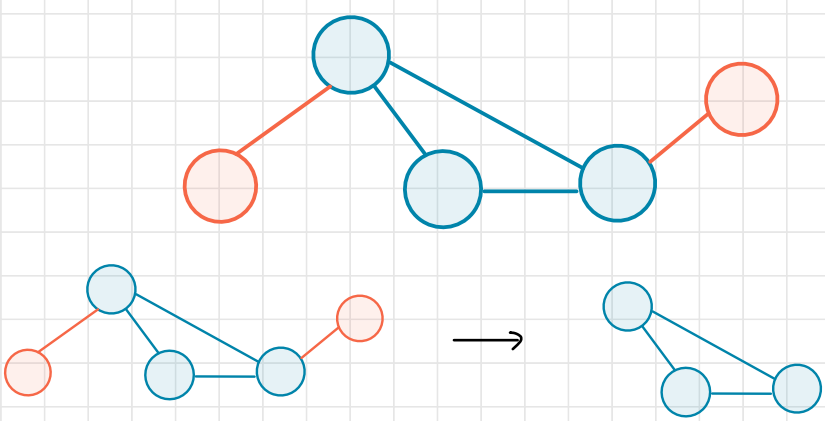
- 1-core: graph with no isolate
- 2-core: at least 2 neighbours (no pendulum structure)
- The graph below has 2 pendulum structures (not a 2-core)



- To find 2-core, recursively remove all nodes with degree < 2 until a 2-core graph is left



- Graph above has no 2-core
- Modified graph with a non-empty 2-core



Algorithm to find k-core

- Start with original graph and remove all nodes with degree $< k$
- Remove all edges connected to removed nodes
- Update degrees of nodes in the graph after deletion
- Repeat until all nodes have degree $\geq k$ (k-core formed)

5. k-Crust

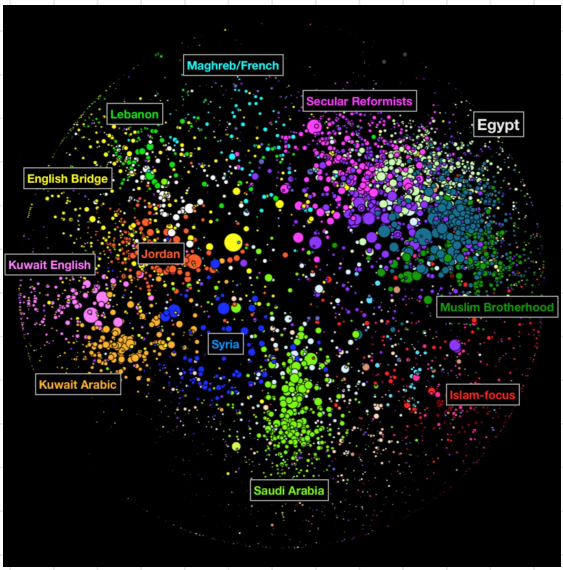
- What is left in the graph after removal of k-core
- Loosely connected periphery

6. k-corona

- Subgraph of k-core where all nodes have exactly k neighbours in the core

COMMUNITY DETECTION

- Communities: structures of nodes with strong internal connections and weak connections to outside the structure
- Maximally decoupled
- Algorithms:
  1. Clique Percolation method
  2. Girvan Newman
  3. Louvain



1. CLIQUE PERCOLATION METHOD (CPM)

- Assume communities formed from a set of cliques and edges connecting them
- Use clique as seed

Input

- Clique size k (no. of nodes)
- Network

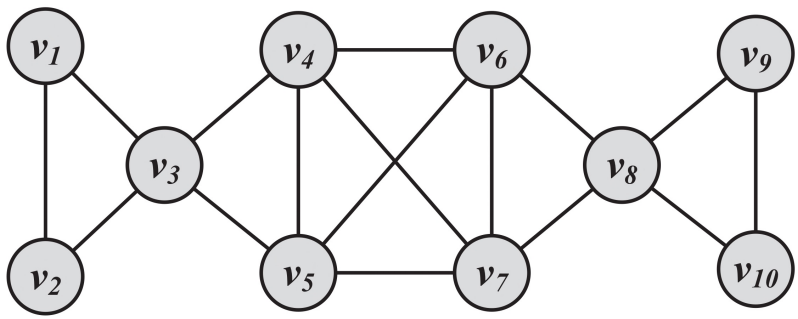
Algorithm

Algorithm 6.2 Clique Percolation Method (CPM)

Require: parameter k

1: **return** Overlapping Communities
2: $Cliques_k$ = find all cliques of size k
3: Construct clique graph $G(V, E)$, where $|V| = |Cliques_k|$
4: $E = \{e_{ij} \mid$ clique i and clique j share $k-1$ nodes$\}$
5: Return all connected components of G (union)

Q: Consider the graph shown. For $k=3$, apply CPM



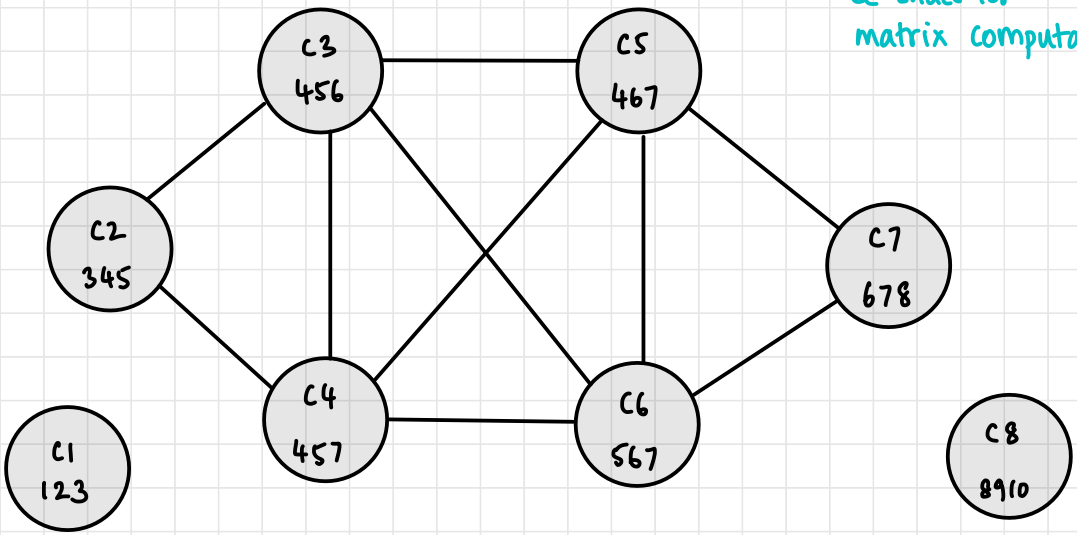
- Cliques of size $k=3$ (fully connected subgraphs of 3 nodes)

C1. $\{v_1, v_2, v_3\}$
C2. $\{v_3, v_4, v_5\}$
C3. $\{v_4, v_5, v_6\}$
C4. $\{v_4, v_5, v_7\}$
C5. $\{v_4, v_6, v_7\}$
C6 $\{v_5, v_6, v_7\}$
C7. $\{v_6, v_7, v_8\}$
C8. $\{v_8, v_9, v_{10}\}$

- Cliques that share $k-1 = 3-1 = 2$ nodes are connected (only considering connections from $i \to j$ where $j > i$ to avoid duplicates)

1. C1 —
2. C2 — C3, C4
3. C3 — C4, C5, C6
4. C4 — C5, C6
5. C5 — C6, C7
6. C6 — C7
7. C7 —
8. C8 —

see slides for matrix computation



- Connected components are returned as communities (here, 3 components)

- Communities:

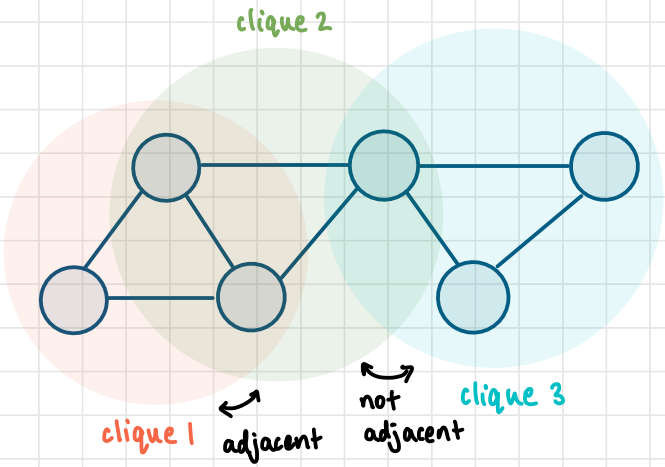
1. $\{v_1, v_2, v_3\}$ (union)
2. $\{v_3, v_4, v_5, v_6, v_7, v_8\}$
3. $\{v_8, v_9, v_{10}\}$

- Nodes v_3 & v_8 belong to 2 communities each (overlapping communities)

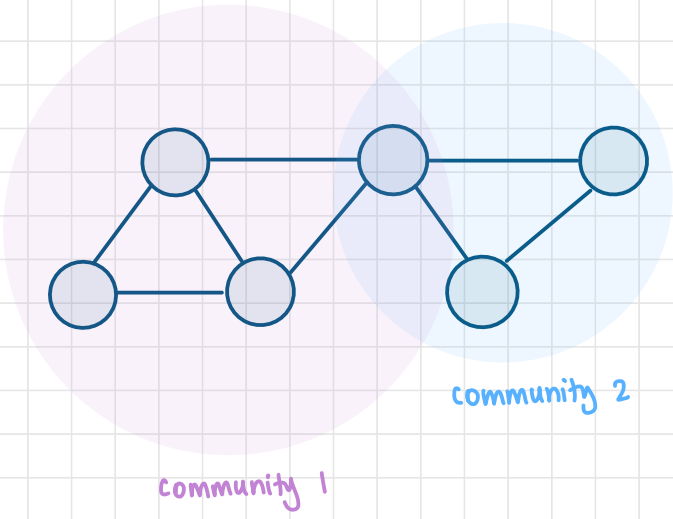
- CPM is computationally intensive (only perform for small k values in practice)

- Adjacent cliques of sizes k : share $k-1$ nodes

- For $k=3$



- Communities



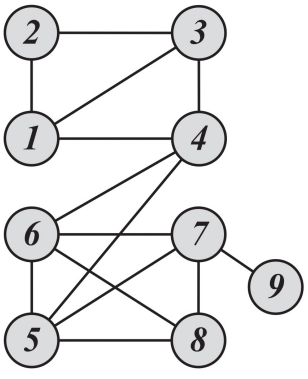
2. GIRVAN-NEWMAN ALGORITHM

- Group-based community detection
- Edge betweenness: no. of shortest paths that include a particular edge
  - measures how bridge-like an edge acts
  - bridge between communities - high betweenness
- Hierarchical communities – divisive clustering performed (assume initially one big cluster and divide)
- Removing edges of high edge betweenness splits up graph into communities

Algorithm

1. Calculate edge betweenness for all edges
2. Remove edge with highest betweenness
3. Recalculate edge betweenness
4. Repeat

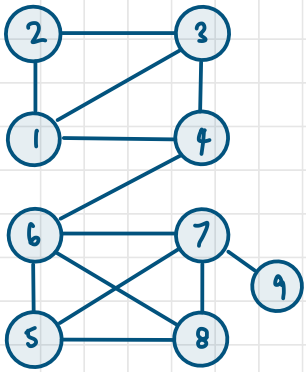
Q: Consider the graph shown. Apply Girvan-Newman algorithm to find 3 communities. Edge betweenness shown below.



	1	2	3	4	5	6	7	8	9
1	0	4	1	9	0	0	0	0	0
2	4	0	4	0	0	0	0	0	0
3	1	4	0	9	0	0	0	0	0
4	9	0	9	0	10	10	0	0	0
5	0	0	0	10	0	1	6	3	0
6	0	0	0	10	1	0	6	3	0
7	0	0	0	0	6	6	0	2	8
8	0	0	0	0	3	3	2	0	0
9	0	0	0	0	0	0	8	0	0

max

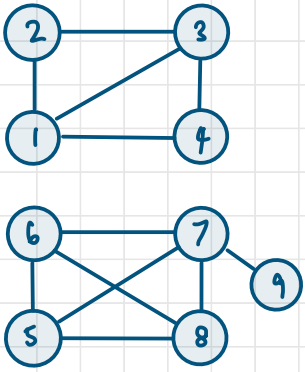
- First edge to remove : e(4,5) (or e(4,6))
- Recompute edge betweenness



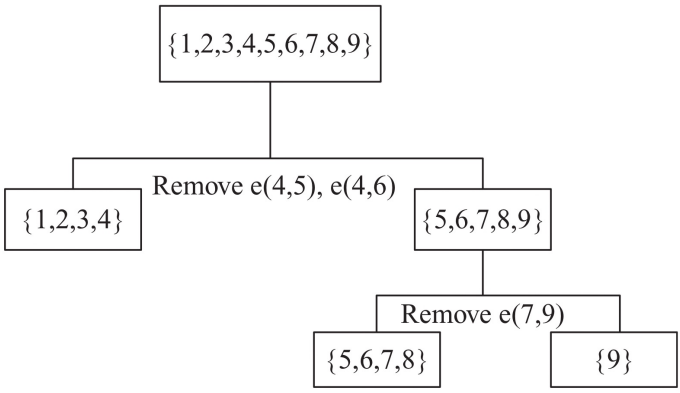
e(4,6) : {1,2,3,4} to {5,6,7,8,9} through e(4,6)

∴ betweenness = 4×5 = 20

e(7,9): betweenness = 4×1
{5,6,7,8} to {9}



- Dendogram



Computing Betweenness Values

1. Perform a breadth-first search of the graph, starting at A.
2. Determine the number of shortest paths from A to each other node.
3. Based on these numbers, determine the amount of flow from A to all other nodes that use each edge.

- Example

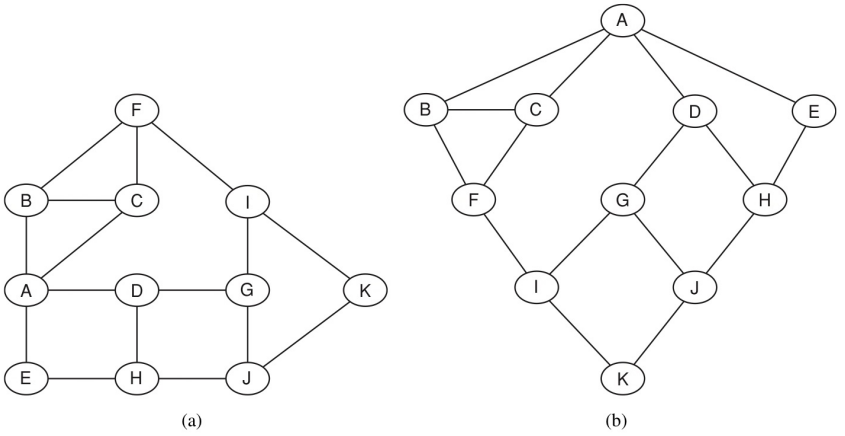
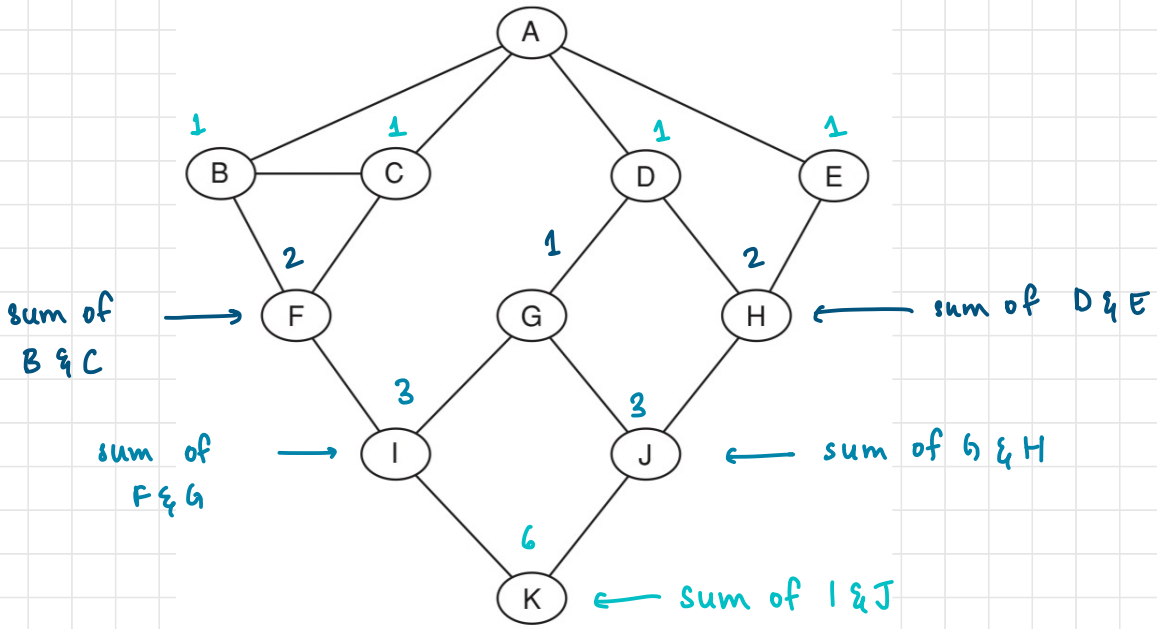
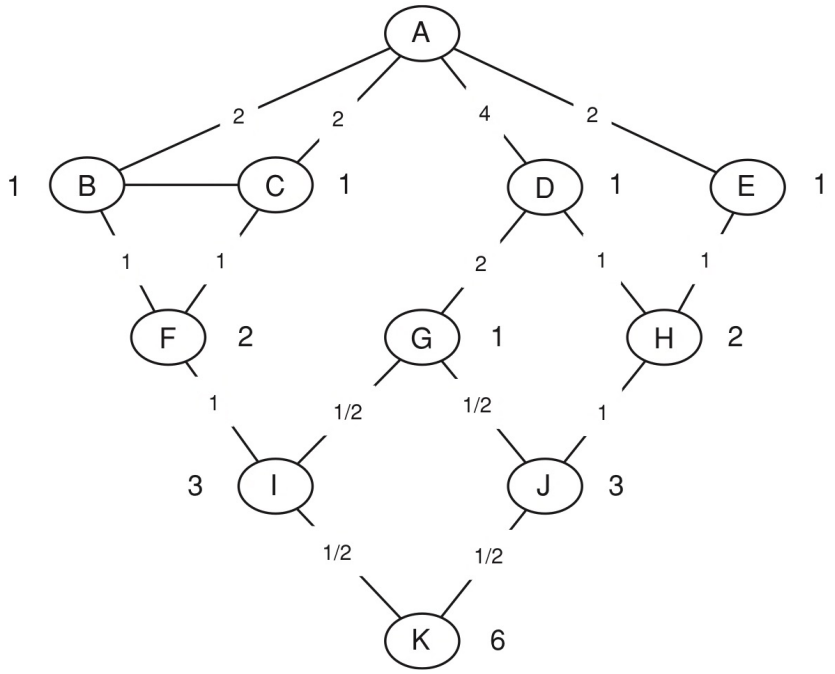


Figure 3.18. The first step in the efficient method for computing betweenness values is to perform a breadth-first search of the network. (a) A sample network and (b) the results of breadth-first search from node A are shown; over the course of the method, breadth-first search is performed from each node in turn.

- For source A



- Assume each node generates 1 unit of information that 'flows' through the graph
- Divide this flow (+ incoming flow) among all edges directly above it (start at bottom)
- node K: 1 unit of flow divided evenly between I and J ⇒ edge (I, K) and edge (J, K) get a flow of 1/2 each
- node I: incoming 1/2 unit + its own 1 unit of flow, 2 parts to F and 1 to G ⇒ edge (F, I) gets 1/2 and edge (G, I) gets 1
- Repeat unit source A



- Repeat for all n source vertices
- Time complexity: $O(mn)$ where m = no. of edges, n = no. of nodes
- Repeat for each edge removed → algorithm complexity $O(m^2n)$

3. LOUVAIN ALGORITHM

- **Modularity**: how likely the community structure was created at random
- Expected no. of edges between 2 nodes v_i & v_j of degrees d_i and d_j (m = no. of edges in graph)

- For any edge out of v_i, probability that it connects to v_j is (probability that it belongs to v_j = fraction of edges that go to v_j)

$$P = \frac{d_j}{\sum_k d_k} = \frac{d_j}{2m}$$

← sum of all degrees = 2 × no. of edges

- Expected no. of edges from v_i to v_j

$$\frac{d_i d_j}{2m}$$

- Actual edges between v_i and v_j = A_{ij}
- Difference between actual and expected = $A_{ij} - \frac{d_i d_j}{2m}$
- Assume k partitions $(P_1, P_2, \ldots, P_k)$ of G
- For each partition, distance

$$\sum_{v_i, v_j \in P_x} A_{ij} - \frac{d_i d_j}{2m}$$

- Over all k partitions

$$\sum_{x=1}^{k} \sum_{v_i, v_j \in P_x} A_{ij} - \frac{d_i d_j}{2m}$$

- Normalized

$$Q = \frac{1}{2m} \sum_{x=1}^{k} \sum_{v_i, v_j \in P_x} A_{ij} - \frac{d_i d_j}{2m}$$

- Define matrix X as indicator / partition matrix $(n \times k)$

$$X_{ij} = 1 \text{ if } v_i \in P_j$$

- Matrix form of equation $\left(B = A - \frac{dd^T}{2m}\right)$

$$Q = \frac{1}{2m} Tr(X^T B X)$$

Algorithm

1. Start with n distinct clusters
2. Join 2 communities that produce the largest increase or smallest decrease in Q

- Agglomerative algorithm

$$Q(G) = \sum_{C \in G} \left[\frac{|E(C)|}{m} - \left(\frac{\sum_{v \in C} deg(v)}{2m} \right)^2 \right]$$

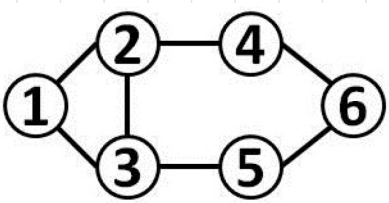
($|E(C)|$: edges in cluster C)

- Greedy optimization $\approx O(n \log n)$

COMMUNITY DETECTION BIG PICTURE

1. Member-based community detection

- Intuition: nodes with same characteristics tend to form a community
- Based on
  1. Node degree - K-clique, K-clan, CPM
  2. Node reachability - K-plex, K-core
  3. Node similarity - Jaccard, cosine
- Let $N(v_i)$ = immediate neighbours of v_i



Jaccard Similarity

$$\sigma_{Jaccard}(v_i, v_j) = \frac{|N(v_i) \cap N(v_j)|}{|N(v_i) \cup N(v_j)|}$$

Eg: $\sigma_{Jaccard}(2,5) = \frac{|\{1,3,4\} \cap \{3,6\}|}{|\{1,3,4,6\}|} = \frac{1}{4} = 0.25$

Cosine Similarity

$$\sigma_{cosine}(v_i, v_j) = \frac{|N(v_i) \cap N(v_j)|}{\sqrt{|N(v_i)|\,|N(v_j)|}}$$

$$\text{Eg: } \sigma_{\text{cosine}}(2,5) = \frac{|\{1,3,4\} \cap \{3,6\}|}{\sqrt{|\{1,3,4\}||\{3,6\}|}} = \frac{1}{\sqrt{6}} = 0.41$$

Pearson Correlation Coefficient

$$\sigma_{\text{Pearson}}(v_i, v_j) = \frac{\sum_k (A_{ik} - \bar{A}_i)(A_{jk} - \bar{A}_j)}{\sqrt{\sum_k (A_{ik} - \bar{A}_i)^2} \sqrt{\sum_k (A_{jk} - \bar{A}_j)^2}}$$

2. Group-based community detection

- Based on group properties
- Methods
  1. Girvan-Newman
  2. Modularity maximization / Louvain algorithm
  3. Spectral clustering

EVALUATION of CLUSTERS

1. With ground truth (test | labelled data)
2. Without ground truth

1. With Ground Truth

- Precision

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

A TP decision assigns 2 nodes to the same cluster iff they are similar

A TN decision assigns 2 nodes to different clusters iff they are dissimilar

- Recall

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

A FP decision assigns 2 nodes to the same cluster when they are dissimilar

A FN decision assigns 2 nodes to diff clusters when they are similar

- F1-score / F-measure

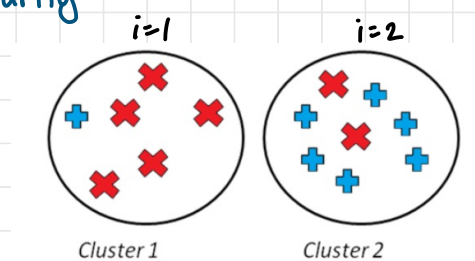
$$F1 = \frac{2 \times P \times R}{P + R}$$

- Purity: fraction of instances that have labels equal to the community's majority label

$$\text{Purity} = \frac{1}{N} \sum_{i=1}^{k} \max_j |C_i \cap L_j|$$

k = no. of communities
N = no. of nodes
L_j = set of instances with label j in all communities
C_i = set of members in community i

Q: Calculate purity

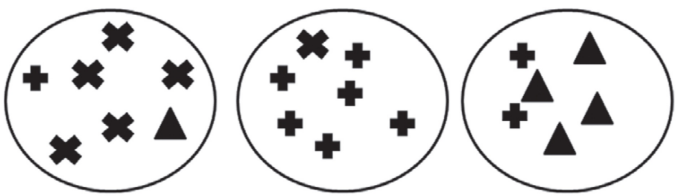


$N = 14$
$k = 2$
$j = 1$: red ×
$j = 2$: blue +

$$\text{Purity} = \frac{1}{14}\left(\max_j |(\text{cluster 1} \cap \text{red}), (\text{cluster 1} \cap \text{blue})| + \max_j |(\text{cluster 2} \cap \text{red}), (\text{cluster 2} \cap \text{blue})| \right)$$

$$= \frac{1}{14}(5+6) \quad = \frac{11}{14} = 0.79$$

Q: Calculate precision, recall, F1-score, purity



Community 1 *Community* 2 *Community* 3

Figure 6.15. Commmunity Evaluation Example. Circles represent communities, and items inside the circles represent members. Each item is represented using a symbol, +, ×, or △, that denotes the item's true label.

$$TP = \underbrace{{}^5C_2}_{C_1} + \underbrace{{}^6C_2}_{C_2} + \underbrace{{}^4C_2}_{C_3} + {}^2C_2 = 32$$

$$TN = \underbrace{(5\times6 + 5\times2 + 5\times4)}_{x_1\ t_2\quad x_1\ t_3\quad x_1\ \Delta_3} + \underbrace{(1\times1 + 1\times4)}_{t_1\ x_1\quad t_1\ \Delta_3} + \underbrace{(1\times1 + 1\times6 + 1\times2)}_{\Delta_1 x_2\quad \Delta_1 t_2\quad \Delta_1 t_3} + \underbrace{(1\times2 + 1\times4 + 6\times4)}_{x_2\ t_3\quad x_2\ \Delta_3\quad t_2\ \Delta_3} = 104$$

$$FN = \underbrace{(5\times1)}_{x} + \underbrace{(1\times6 + 1\times2 + 6\times2)}_{t} + \underbrace{(1\times4)}_{\Delta} = 29$$

$$FP = \underbrace{(5\times1 + 5\times1 + 1\times1)}_{C_1} + \underbrace{(6\times1)}_{C_2} + \underbrace{(4\times2)}_{C_3} = 25$$

$$P = \frac{TP}{TP+FP} = \frac{32}{32+25} = 0.56$$

$$R = \frac{TP}{TP+FN} = \frac{32}{32+29} = 0.52$$

Clustering Quality

- Min intra cluster, max inter cluster distance
- Types of measures

1. External measures – require ground truth
   * Rand index
   * Purity

2. Intrinsic measures - do not require ground truth
   * Silhouette coefficient
   * inter / intra cluster similarity

   - not always reliable

Rand Index

- Let a = no. of times a pair of items belong to same cluster
- Let b = no. of times a pair of items belong to diff clusters

$$\text{Rand index } R = \frac{a+b}{{}^{n}C_{2}}$$

Silhouette Coefficient

- Range : [-1, 1]
  * 1 : clusters well apart
  * 0 : indifferent
  * -1 : clusters assigned in the wrong way

- Let a = avg intra cluster distance
- Let b = avg inter cluster distance

$$\text{Silhouette score} = \frac{b - a}{\max(a, b)}$$