Part III

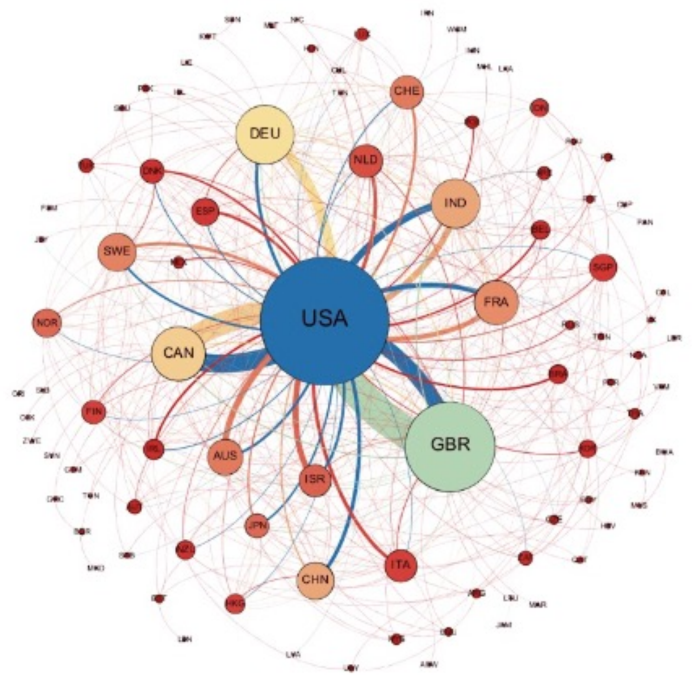
Measurements

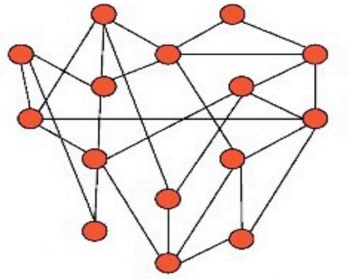
Measures based analyse

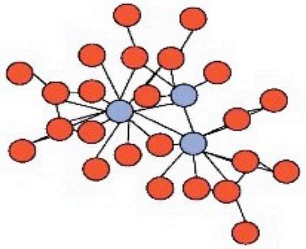
- Analysis based on measures
- Measures of distance, importance, connectivity, and robustness
- Multi-level measures
  - Microscopic: degree, local clustering coefficient
  - Mesoscopic: connected components, community detection
  - Macroscopic: transitivity, diameter, density

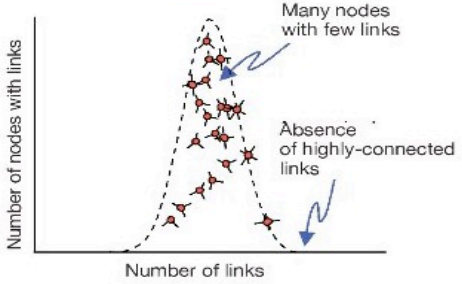
Degree

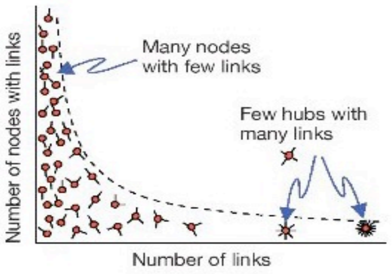
- **Degree**: number of incident connections
  - Weighted graph/multigraph: sum
  - Loop: equivalent to 2 edges
  - Directed graph: in-degree & out-degree
- ∑ degrees (nodes) = 2 x number of edges
- Visual & statistical analysis (distribution)
- Probability distribution of degrees in the network







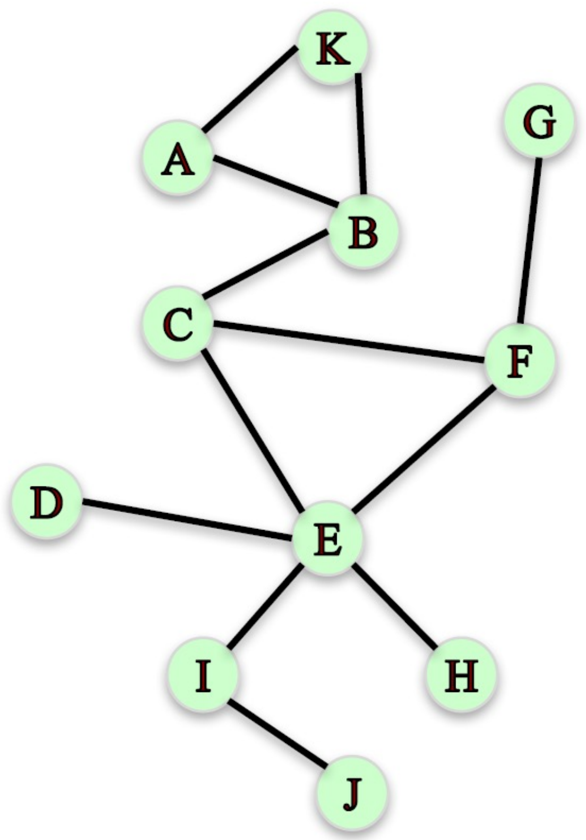




Distance

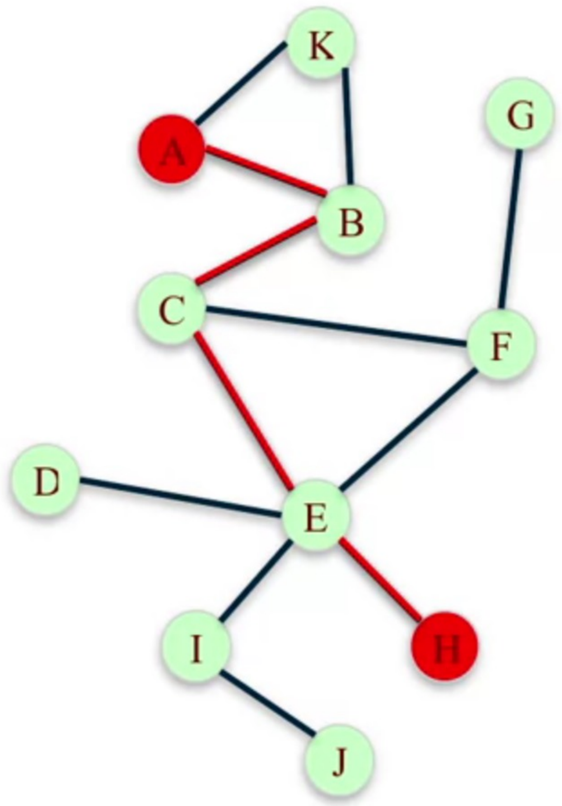
- At what **distance** is node A from node H?
- Are the nodes far from or close to each other in this network?
- Which nodes are the closest and the **farthest** from other nodes?

➢ We need to define a sense of **distance** between the nodes to answer these questions.



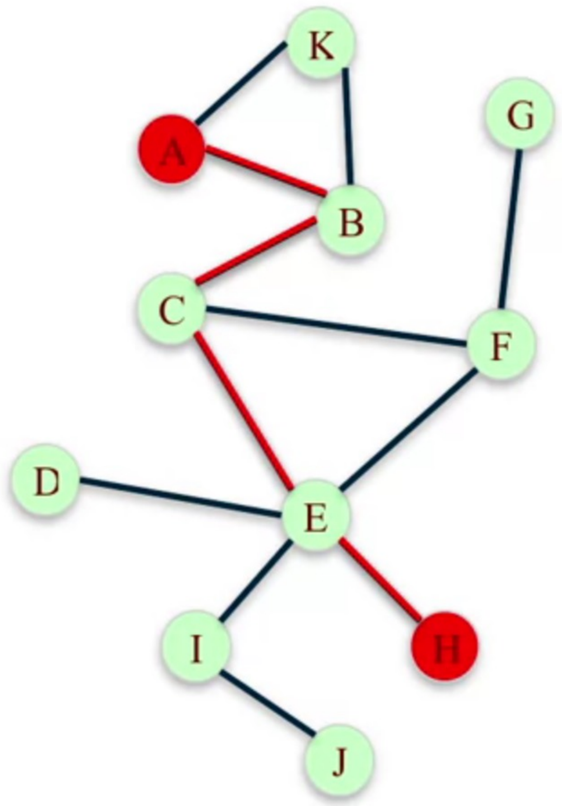
Distance

- How to characterize the distance between all pairs of nodes in a graph?
- The **distance** between two nodes is the length of the shortest path.
  - Distance(A-H) = ?
  - Distance(D-G) = ?
- The **eccentricity** of a node n is the greatest distance between n and all other nodes.
  - Eccentricity(A) = ?
  - Eccentricity(C) = ?
  - Eccentricity(I) = ?



Distance

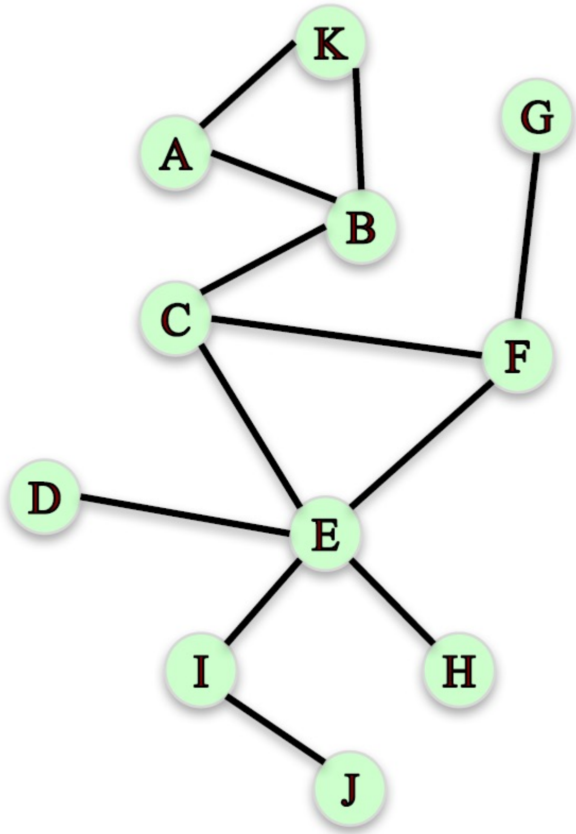
- How to characterize the distance between all pairs of nodes in a graph?
- The **distance** between two nodes is the length of the shortest path.
  - Distance(A-H) = 4
  - Distance(D-G) = 3
- The **eccentricity** of a node n is the greatest distance between n and all other nodes.
  - Eccentricity(A) = 5
  - Eccentricity(C) = 3
  - Eccentricity(I) = 4
  - Eccentricity = {'A': 5, 'B': 4, 'C': 3, 'D': 4, 'E': 3, 'F': 3, 'G': 4, 'H': 4, 'I': 4, 'J': 5, 'K': 5}



Distance

How to summarize the distances between all pairs of nodes in a graph?

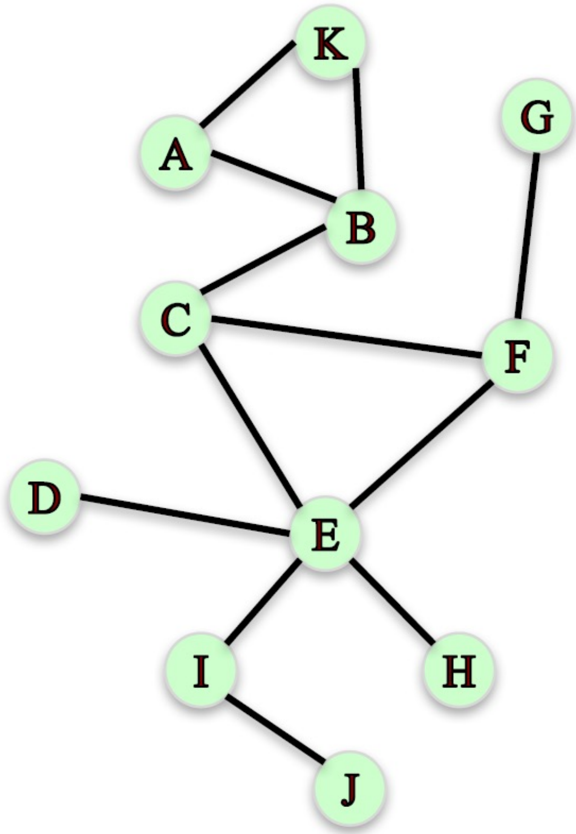
- **Average distance** between each pair of nodes.
  - Average distance = ?
- **Diameter**: maximum distance between any pair of nodes.
  - Diameter = ?
- **The radius** of a graph is the minimum eccentricity.
  - Radius = ?



Distance

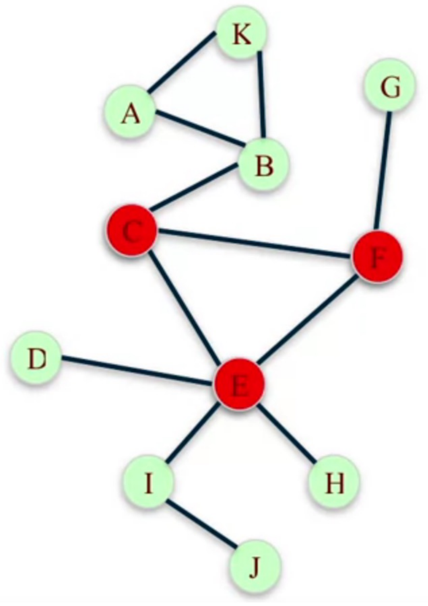
How to summarize the distances between all pairs of nodes in a graph?

- **Average distance** between each pair of nodes.
  - Average distance = 2.53
- **Diameter**: maximum distance between any pair of nodes.
  - Diameter = 5
- **The radius** of a graph is the minimum eccentricity.
  - Radius = 3

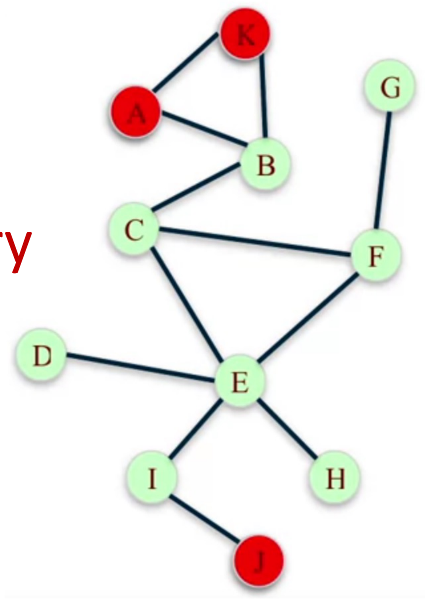


Distance

- The **periphery** of a graph is the set of nodes with eccentricity equal to the diameter.
- The **center** of a graph is the set of nodes with eccentricity equal to the radius.

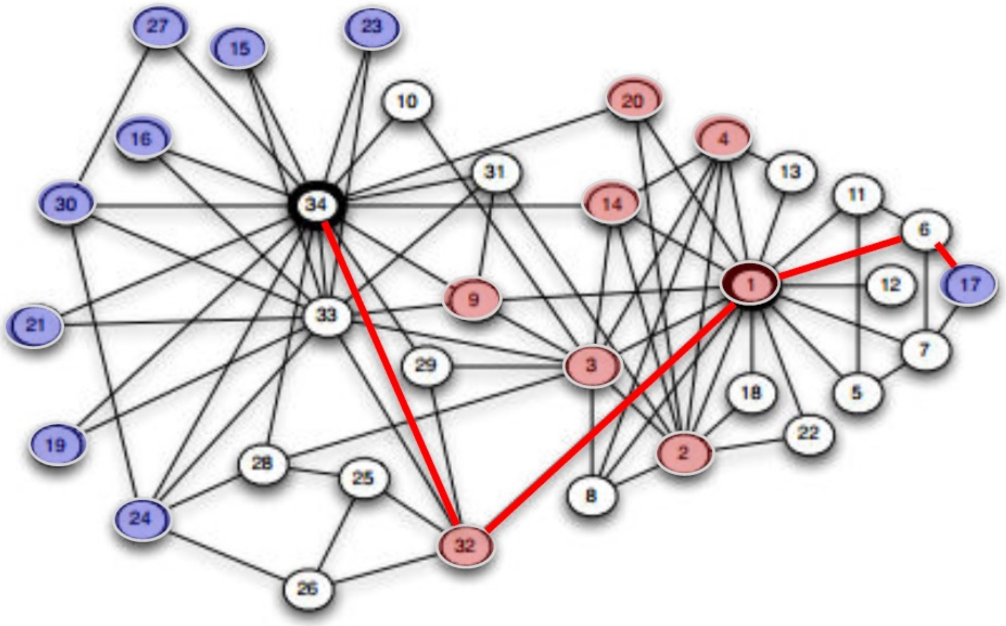


Center & periphery



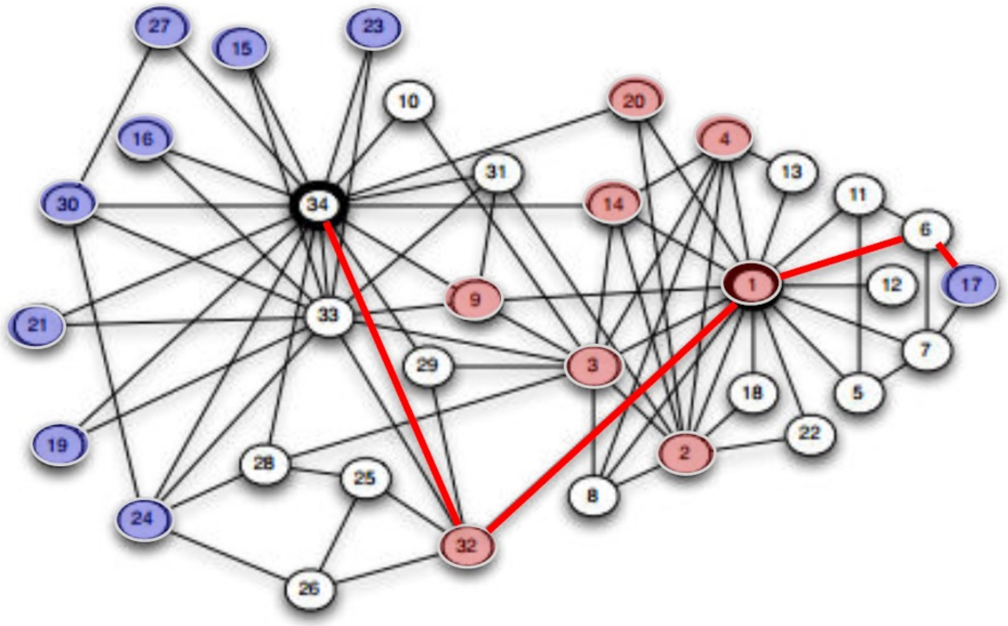
Distance - Example

- Shortest Path = ?
- Radius = ?
- Diameter = ?
- Center = ?
- Periphery : ?



Distance - Example

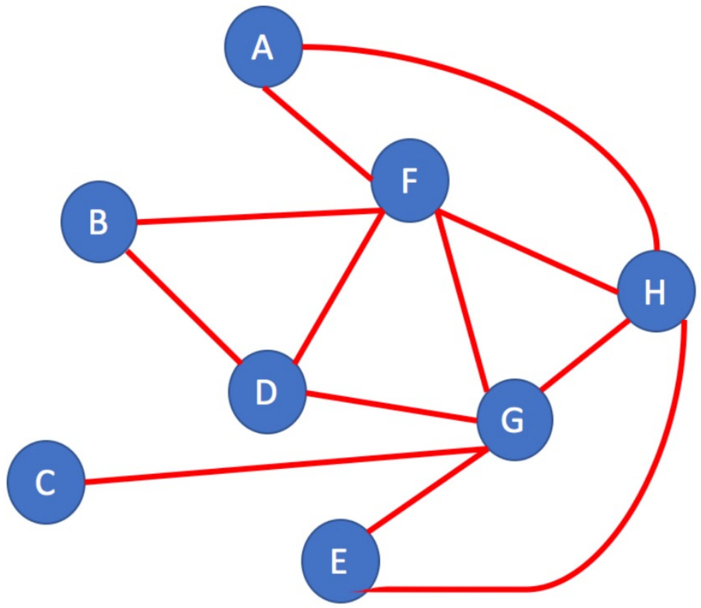
- Shortest Path = 2,41
- Radius = 3
- Diameter = 5
- Center = [1, 2, 3, 4, 9, 14, 20, 32]
- Periphery : [15, 16, 17, 19, 21, 23, 24, 27, 30]
- Node 34 seems quite **central**. However, it has a distance of 4 to node 17.



Distance - Exercice

Calculate the distance measures for the following graph:

Eccentricity = ?
Average distance = ?
Diameter = ?
Radius = ?
Peripheral nodes = ?
Graph Center = ?



Distance - Exercise

Calculate the distance measures for the following graph:

Eccentricity = {'A': 3, 'B': 3, 'C': 3, 'D': 2, 'E': 3, 'F': 2, 'G': 2, 'H': 2}

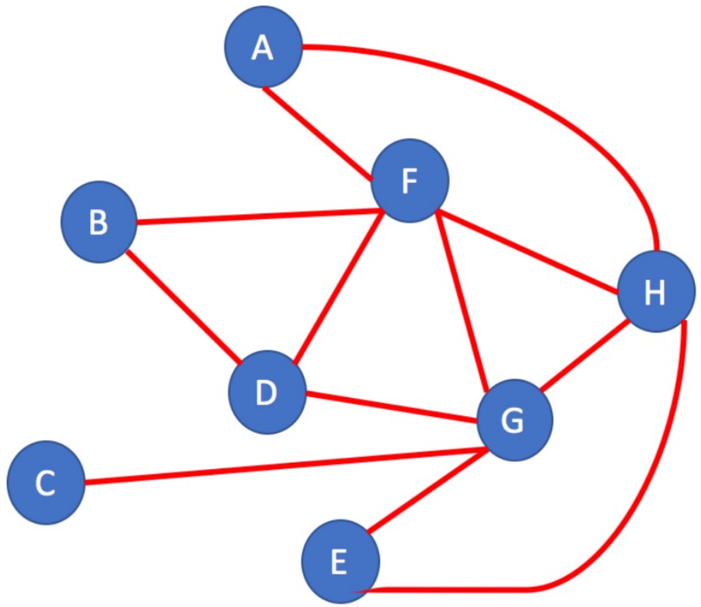
Average Distance = 1.66

Diameter = 3

Radius = 2

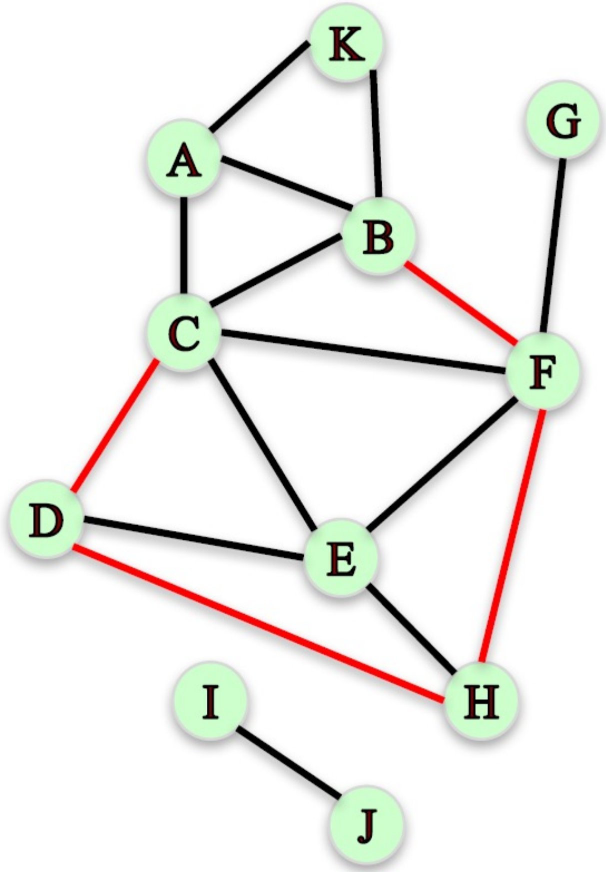
Peripheral Nodes: [A, B, C, E]

Graph Center: [D, F, G, H]



Clustering coefficient

- **Triadic closure** is the tendency of people who share connections in a social network to connect with each other.
- How can we measure the prevalence of triadic closure in a social network?
- **Triadic closure** is a way to measure the density



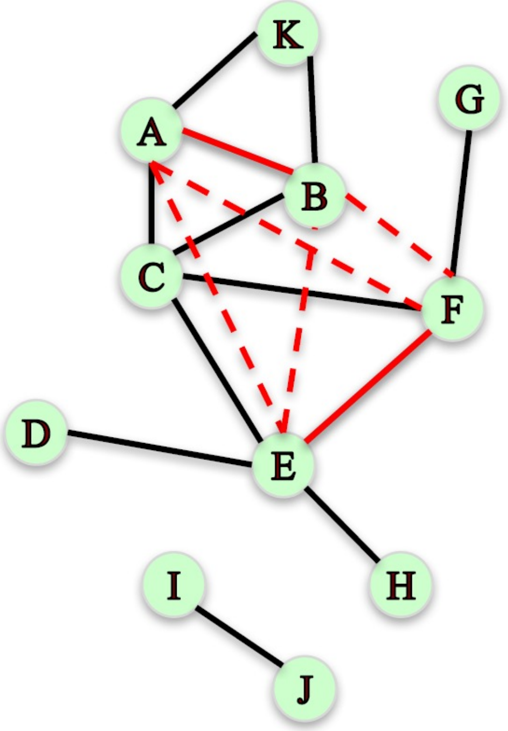
Local Clustering coefficient

- The **local clustering coefficient** (grouping) of a node is the fraction of pairs of the node's friends who are friends with each other

- $\text{LCC} = \dfrac{\text{\# of pairs of friends of a node who are friends}}{\text{\# of pairs of friends of a node}}$

- $\text{\# of pairs of friends of a node} = \dfrac{degree(degree-1)}{2}$

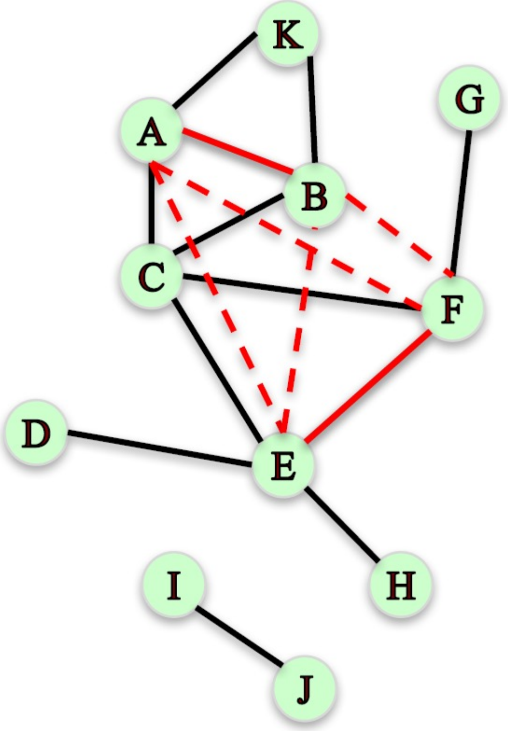
LCC(C) = 2/6 = 1/3
LCC(E) = ?
LCC(F) = ?
LCC(J) = ?



Local Clustering coefficient

- The **local clustering coefficient** (grouping) of a node is the fraction of pairs of the node's friends who are friends with each other
- $\mathrm{LCC} = \dfrac{\#\ \text{of pairs of friends of a node who are friends}}{\#\ \text{of pairs of friends of a node}}$
- $\#\ \text{of pairs of friends of a node} = \dfrac{degree(degree-1)}{2}$

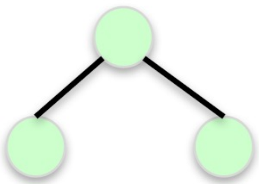
LCC(C) = 2/6 = 1/3
LCC(E) = 1/6
LCC(F) = 1/3
LCC(J) = 0

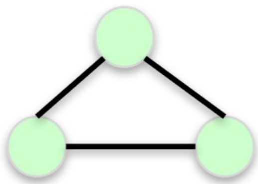


Global Clustering coefficient

- Measuring the clustering coefficient for the entire network:
- **Option I:** calculate the average of all clustering coefficients across all nodes in the graph
- **Option II**: calculate the transitivity of the graph
  - Measures the ratio between the number of triangles and the number of open triads
  - transitivity= $\frac{3 \times \#\ of\ closed\ triads(triangles)}{\#\ of\ open\ triads}$

Open Triad



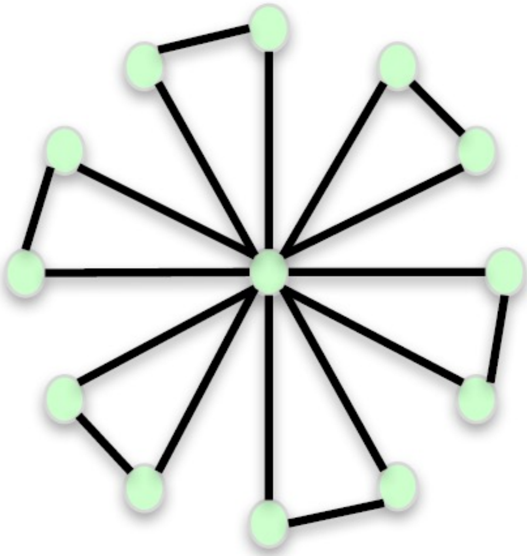


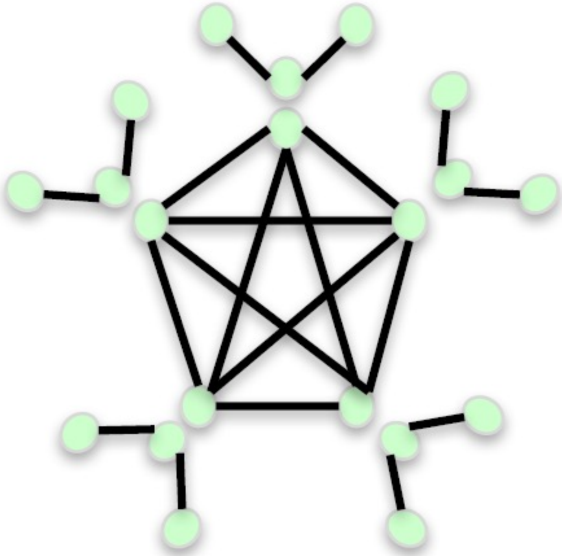
Triangle

Transitivity vs. ACC

- Both measure the tendency of edges to form triangles.
- Transitivity gives more weight to nodes with a higher degree

- Most nodes have a high LCC
- **ACC = 0.93**
- The high-degree node has a low LCC.
- **Transitivity = 0.23**



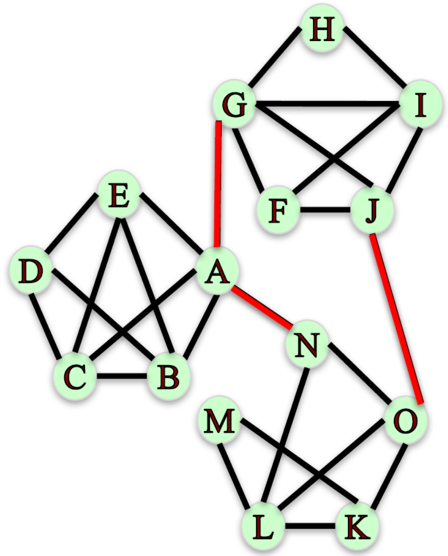


- Most nodes have a low LCC
- **ACC = 0.25**
- The low-degree nodes have a high LCC
- **Transitivity = 0.86**

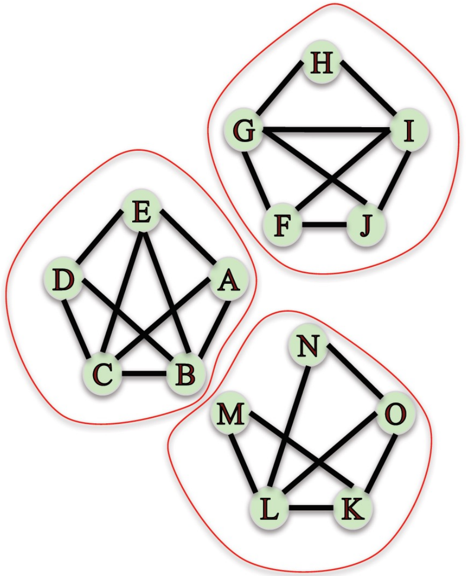
Connectivity – undirected

- An undirected graph is **connected** if there is a chain linking each pair of its nodes.
- G1 is connected, but if the links (A,G) and (A,N) are removed, it will be not.
- In G2, there is no connection between the nodes of the three **communities**.

Graph G1

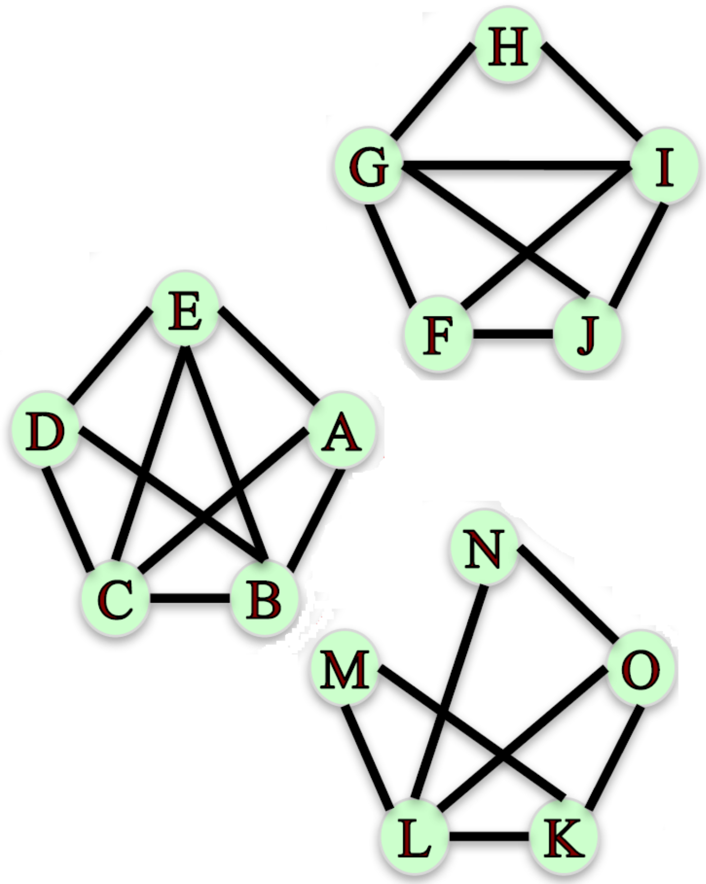


Graph G2



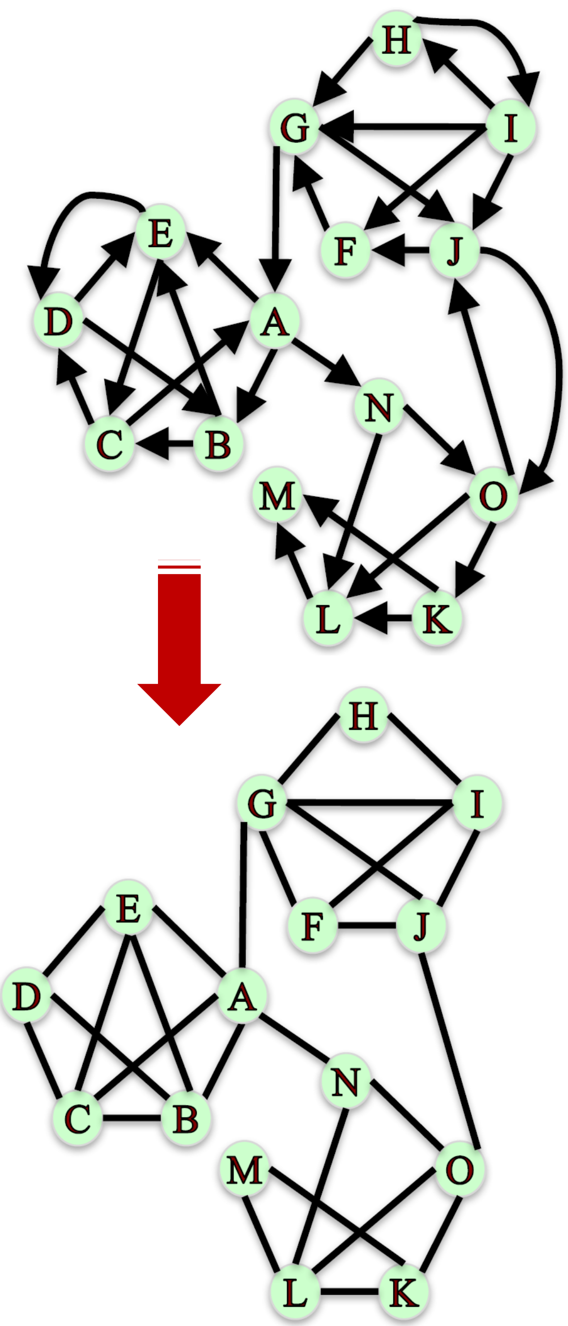
Connectivity – undirected

- A **connected component** is a subgraph such that:
  1. Each pair of nodes has a link connecting them
  2. No external node has a link with a node in the component
- Is {E, A, G, F} a connected component?
- Is {N, O, K} a connected component?
- What are the connected components in this graph?



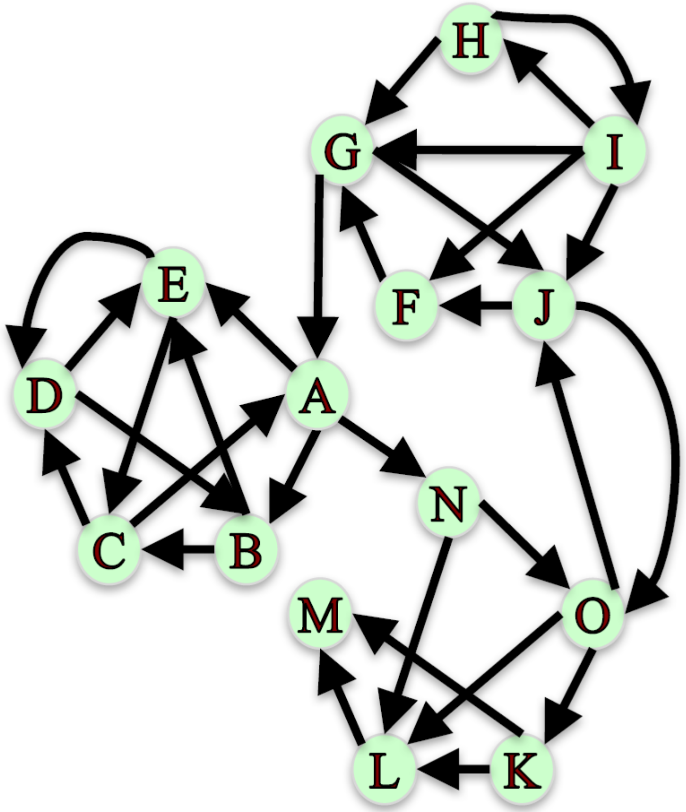
Connectivity – directed

- A directed graph is **strongly connected** if for each pair (u, v) of its nodes, there is a path from u to v and a path from v to u.
  - There is no path from A to H, so this graph is not strongly connected.
- A directed graph is **weakly connected** if it becomes connected after converting all its arcs into edges.



Connectivity – directed

- A **strongly connected component** is a subgraph such that:
  1. Each of its nodes has a path to the other nodes.
  2. No external node has a path to a node in the component.
- A **component** is **weakly connected** if it is a connected component after converting all its arcs into edges.
- What are the strongly and weakly connected components in this graph?



Connectivity – directed

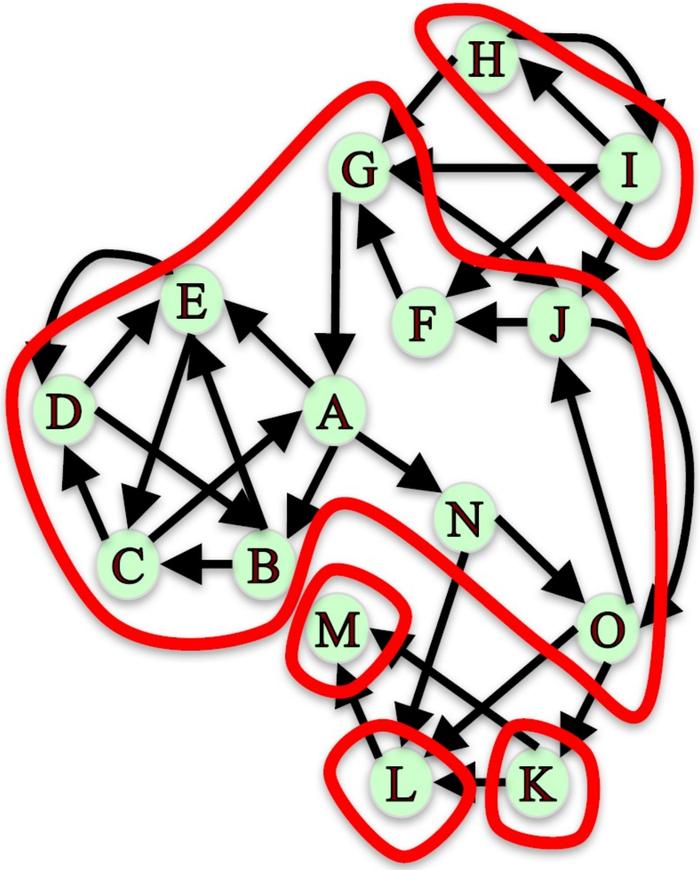
- A **strongly connected component** is a subgraph such that:

  1. Each of its nodes has a path to the other nodes.

  2. No external node has a path to & from a node in the component.

- A **component** is **weakly connected** if it is a connected component after converting all its arcs into edges.

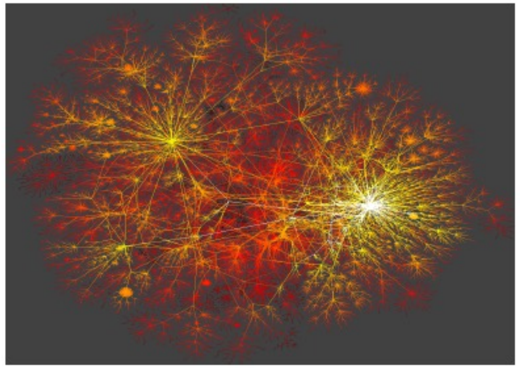
- The graph is weakly connected, so we have a single weakly connected component.



Robustness

- The **robustness** of a network is its ability to maintain its general structural properties when facing failures or attacks.
- **Structural properties**: connectivity.
- Types of **attacks**: removal of nodes or links.
- **Examples**: airport closures or flight cancellations, internet router failures or cable cuts, power line failures.

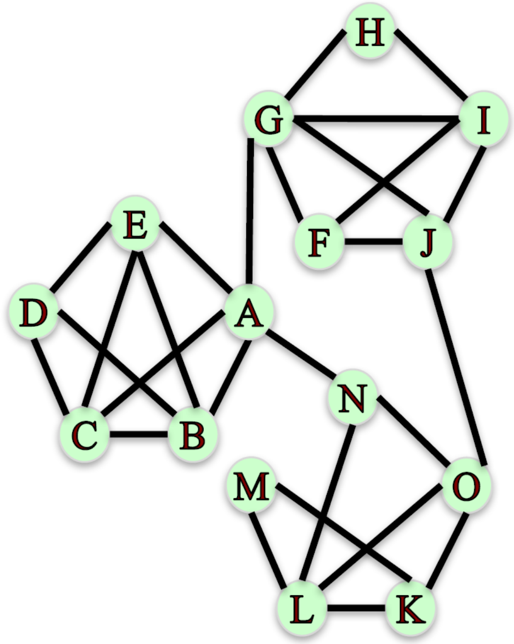
Network of direct flights around the world



Internet Connectivity

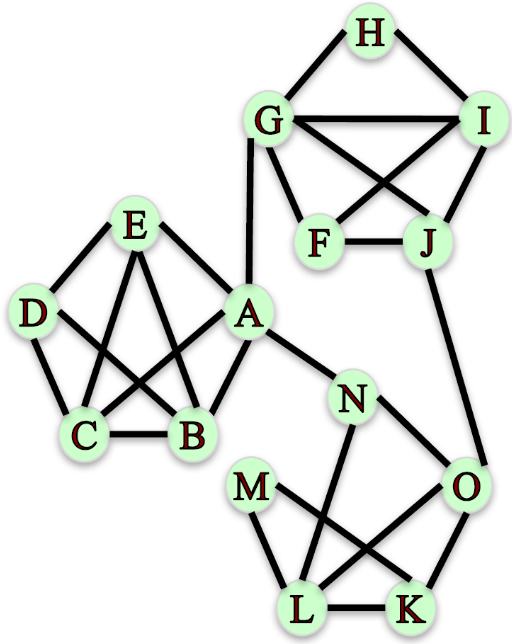
Robustness – nodes

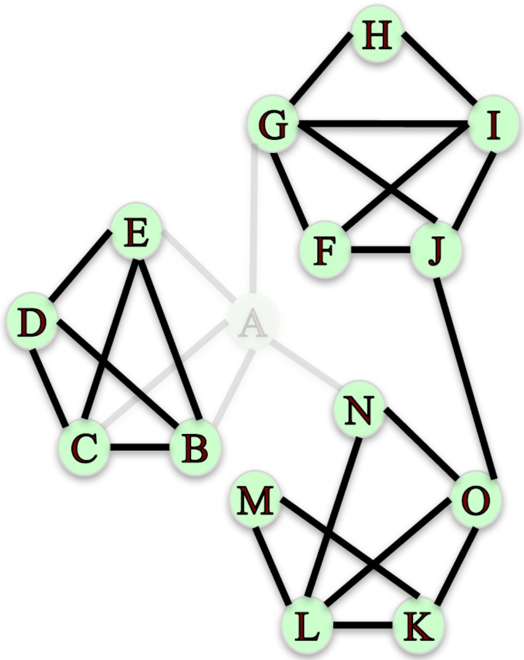
What is the minimum number of nodes that must be removed to **disconnect** this graph? Which nodes are they ?



Robustness – nodes

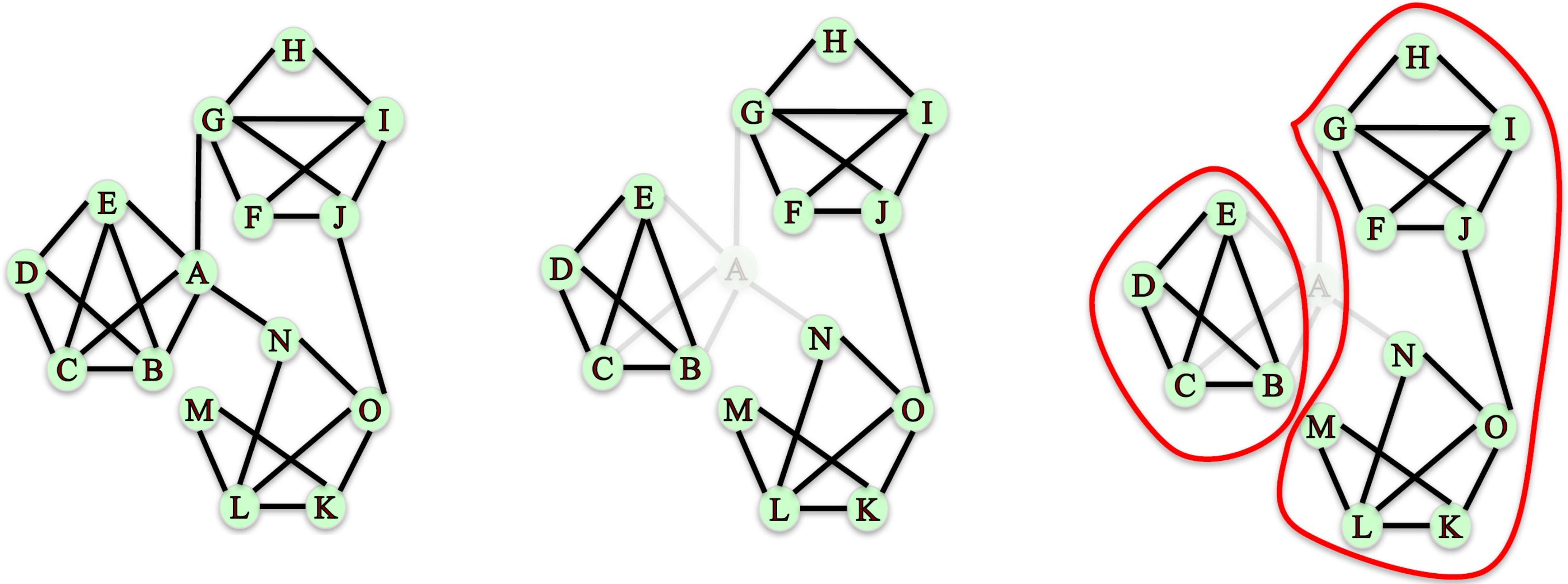
What is the minimum number of nodes that must be removed to **disconnect** this graph? Which nodes are they ?





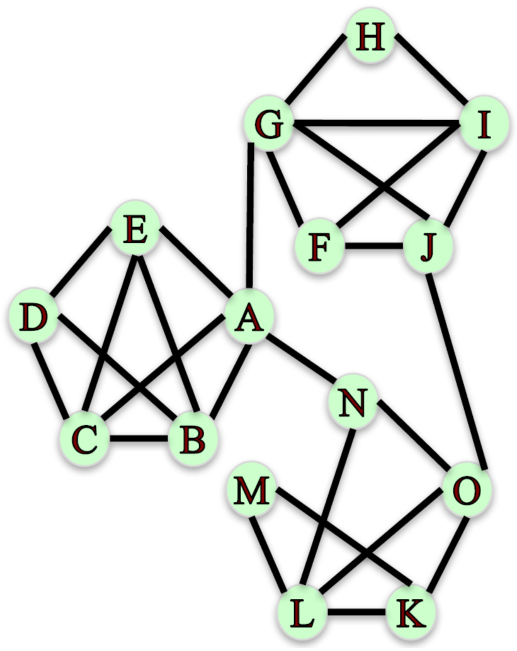
Robustness – nodes

What is the minimum number of nodes that must be removed to **disconnect** this graph? Which nodes are they ?



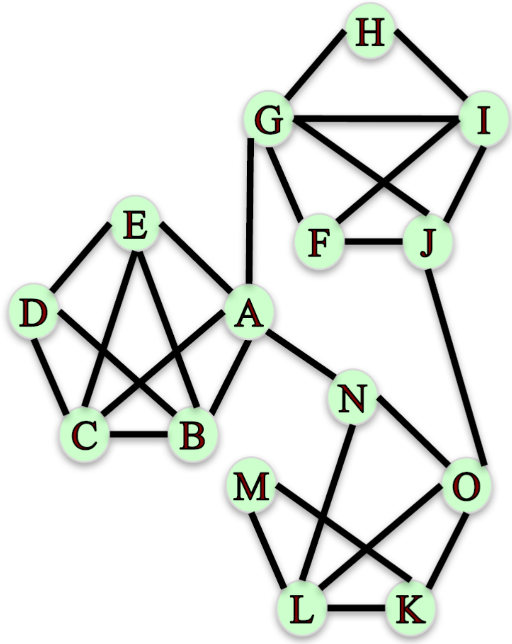
Robustness – edges

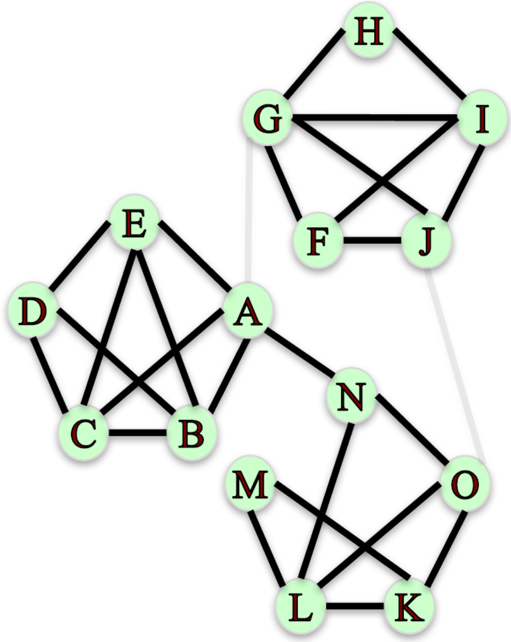
What is the minimum number of edges that must be removed to **disconnect** this graph? Which edges are they ?



Robustness – edges

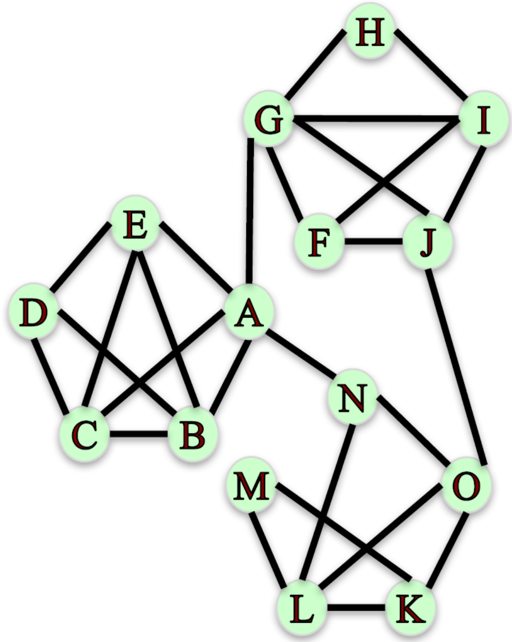
What is the minimum number of edges that must be removed to **disconnect** this graph? Which edges are they ?

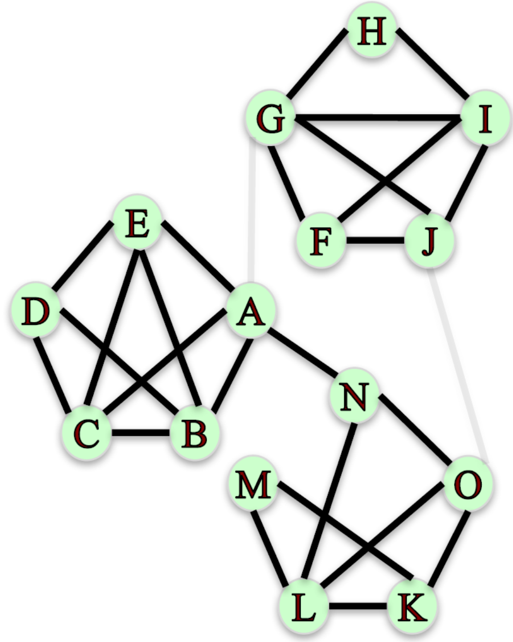


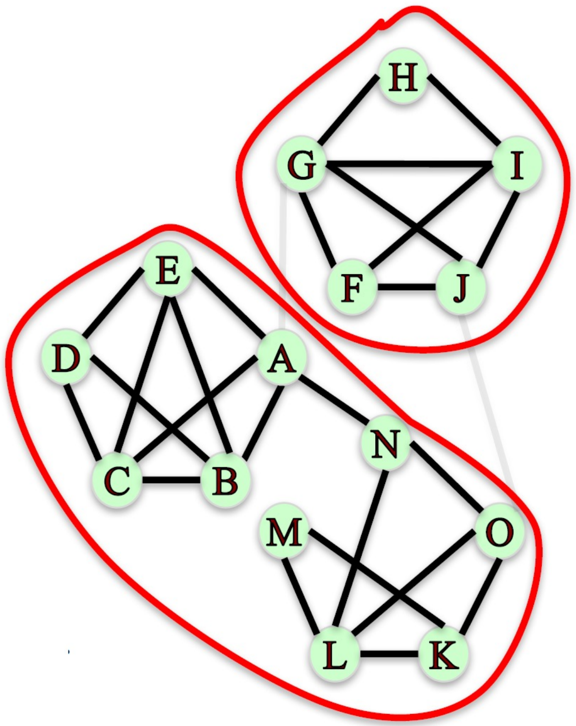


Robustness – edges

What is the minimum number of edges that must be removed to **disconnect** this graph? Which edges are they ?

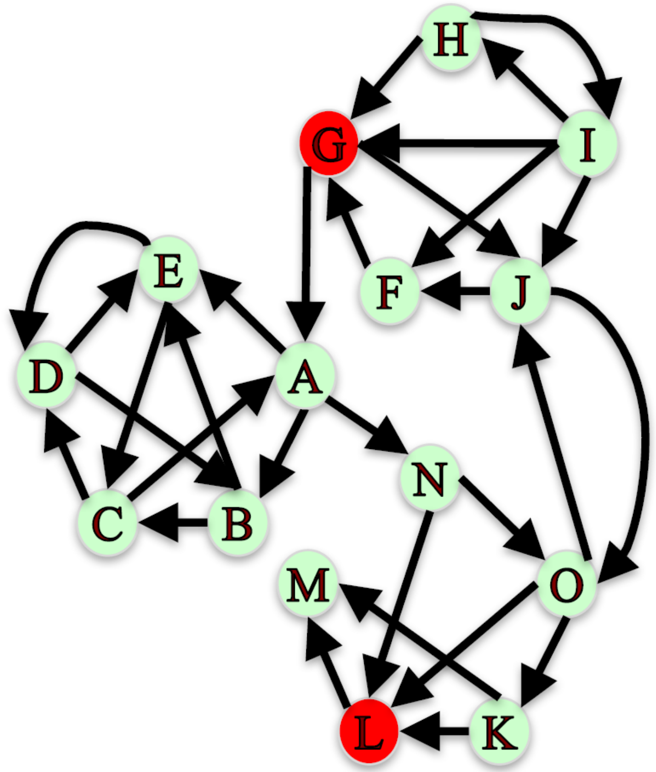






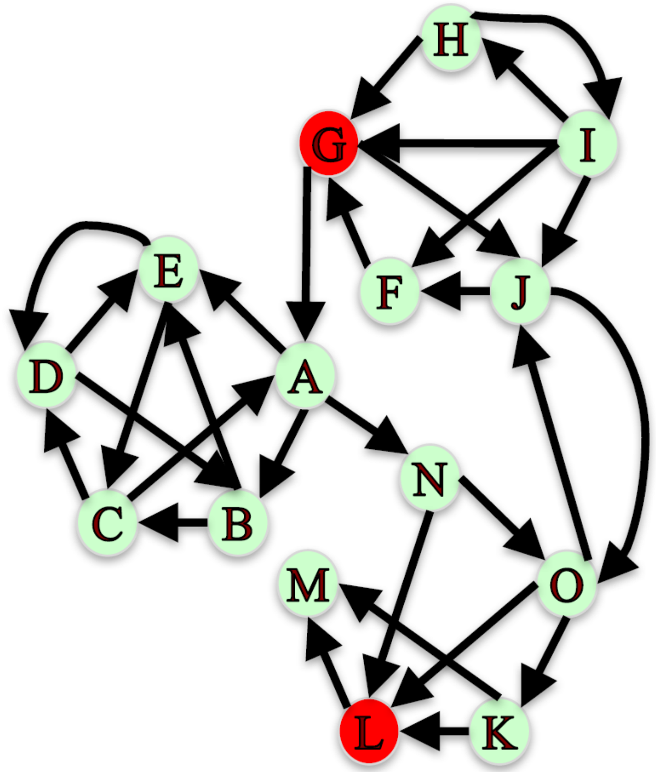
Robustness – Example (directed)

- Imagine that node G wants to send a message to node L by passing it through other nodes in this network.
- What are the possible paths?



Robustness – Example (directed)

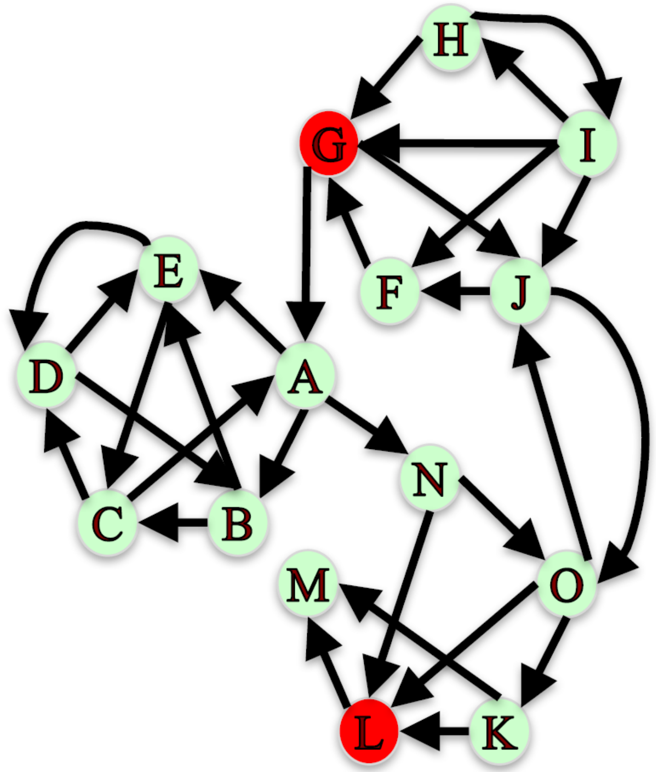
- Imagine that node G wants to send a message to node L by passing it through other nodes in this network.
- What are the possible paths?
  - ['G', 'A', 'N', 'L'],
  - ['G', 'A', 'N', 'O', 'K', 'L'],
  - ['G', 'A', 'N', 'O', 'L'],
  - ['G', 'J', 'O', 'K', 'L'],
  - ['G', 'J', 'O', 'L']



Robustness – Example (directed)

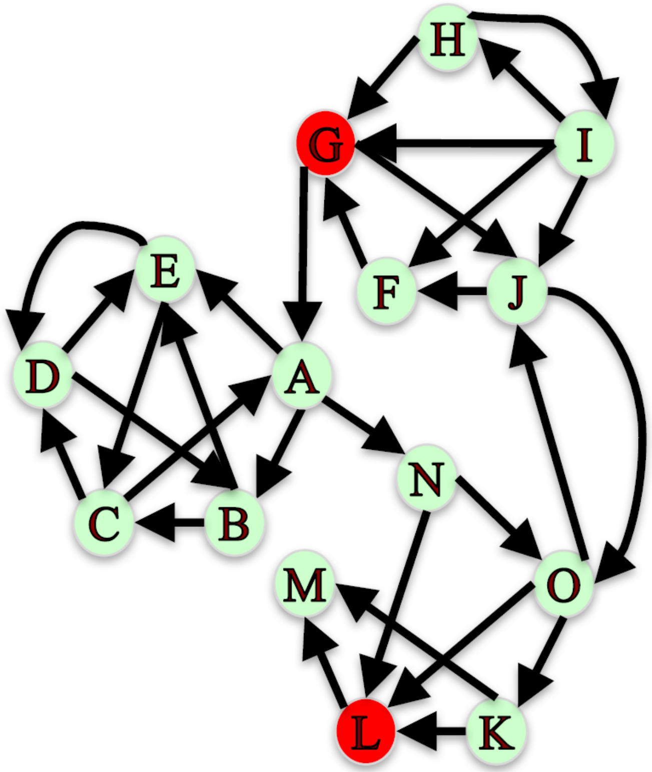
- If we wanted to block the message from G to L by removing nodes from the network, how many nodes would we need to remove? Which ones?

- If we wanted to block the message from G to L by removing connections from the network, how many connections would we need to remove? Which ones?



Robustness – Example (directed)

- If we wanted to block the message from G to L by removing nodes from the network, how many nodes would we need to remove? Which ones?
  - 2 nodes: N & O
  - with N only, we can take the path G→J→O→K→L
  - with O only, we can take the path G→A→N→L
- If we wanted to block the message from G to L by removing connections from the network, how many connections would we need to remove? Which ones?
  - 2 arcs: (A,N), (J,O)
  - We have to delete these 2 arcs to be able to block the message

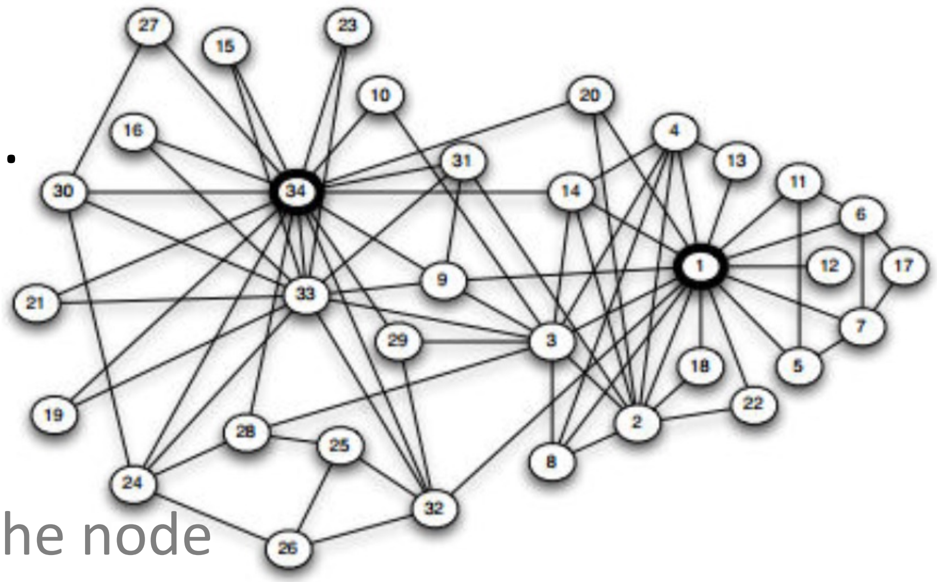


Robustness – synthesis

- A **minimum cut** is the smallest set of nodes/edges that needs to be removed to disconnect a network (or a pair of nodes).
- **Node/edge connectivity** is the minimum number of nodes needed to be removed to disconnect a network (or a pair of nodes).
- **Robust networks** are those with a large minimum cut.
- Networks with high connectivity are more **resilient (Robust)** to node/edge loss (disconnection).

Node importance

- Based on the structure of the Karate club's friendship network, who are the 5 most **important** people in the club?
- Importance is interpreted in different ways.
  - Degree (number of friends)
    - 5 Most important nodes: 34, 1, 33, 3, 2
  - Average proximity to other nodes
    - 5 Most important nodes: 1, 3, 34, 32, 9
  - Fractions of shortest paths that pass through the node
    - 5 Most important nodes: 1, 34, 33, 3, 32



Karate club’s friendship network

Centrality

Centrality measures identify the most **important** nodes in a network:

- Influential actors in a social network.
- Members that spread information to many other members or prevent epidemics.
- Hubs in a transportation network.
- Important pages on the web.
- People that prevent the network from breaking apart.

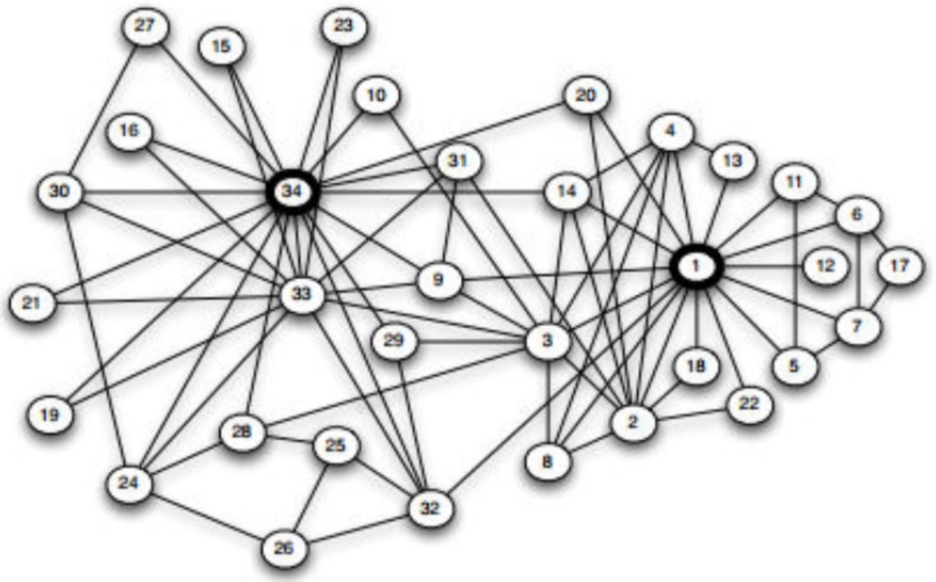
Centrality – measurements

- Degree centrality
- Proximity centrality
- Betweeneness centrality
- Page Rank
- Power centrality
- Prestige centrality
- Katz centrality
- …

Degree centrality – undirected

Hypothesis: important nodes are those with many connections.

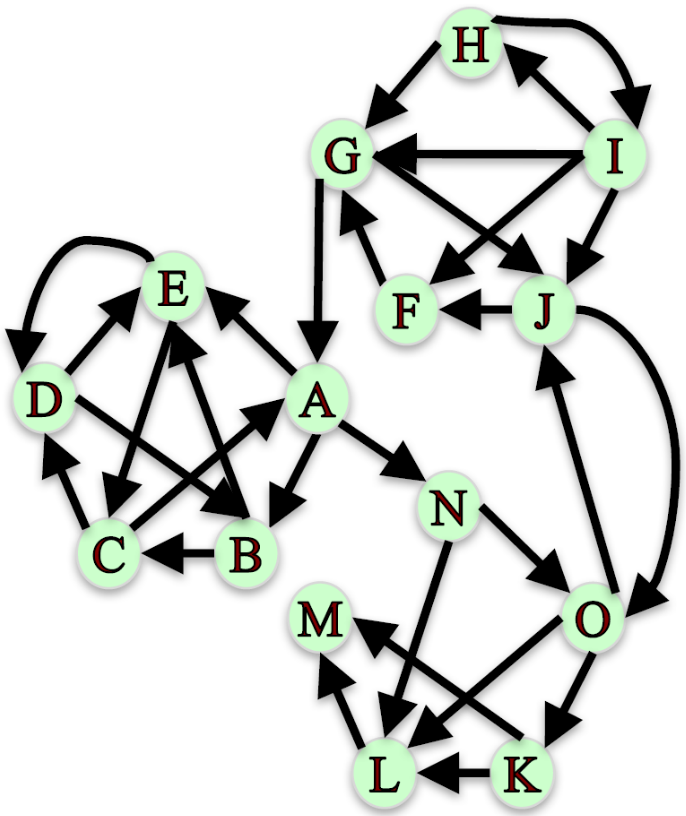
- The simplest measure of centrality: the number of neighbors.
- Undirected networks: use the degree.
  - $\boldsymbol{C_{deg}(v)} = \frac{d_v}{|N|-1}$, N is the nodes set, dv is the v node degree.
    - $C_{deg}(34) = \frac{17}{33} = 0.515$
    - $C_{deg}(1) = ?$
    - $C_{deg}(33) = ?$
    - $C_{deg}(3) = ?$
    - $C_{deg}(2) = ?$



Degree centrality– directed

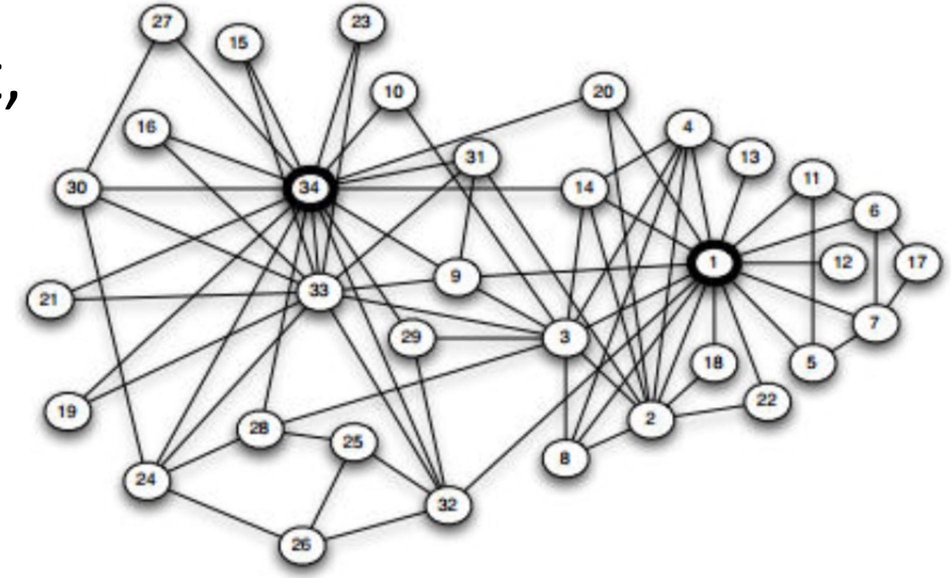
Hypothesis: important nodes are those with many connections.

- The simplest measure of centrality: the number of neighbors.
- Undirected networks: use the in-degree / out-degree.
  - $\boldsymbol{C_{in\_deg}(v)} = \frac{d_v^{in}}{|N|-1}$, $\boldsymbol{C_{out\_deg}(v)} = \frac{d_v^{out}}{|N|-1}$, N is the nodes set, dv in/out are respectively in/out degrees
    - $C_{in\_deg}(G) = C_{in\_deg}(E) = C_{in\_deg}(L) = \frac{3}{14} = 0.214$
    - $C_{out\_deg}(I) = \frac{4}{14} = 0.285$



Closeness centrality

- **Hypothesis**: Important nodes are those closest to other nodes.
- $\boldsymbol{C_{close}(v)} = \frac{|N|-1}{\sum_{u \in N \setminus \{v\}} d(v,u)}$, N is the node set, d(v,u) is distance between nodes v and u
  - $C_{close}(32) = \frac{33}{61} = 0.541$
  - $C_{close}(1) = ?$
  - $C_{close}(3) = ?$
  - $C_{close}(34) = ?$
  - $C_{close}(9) = ?$



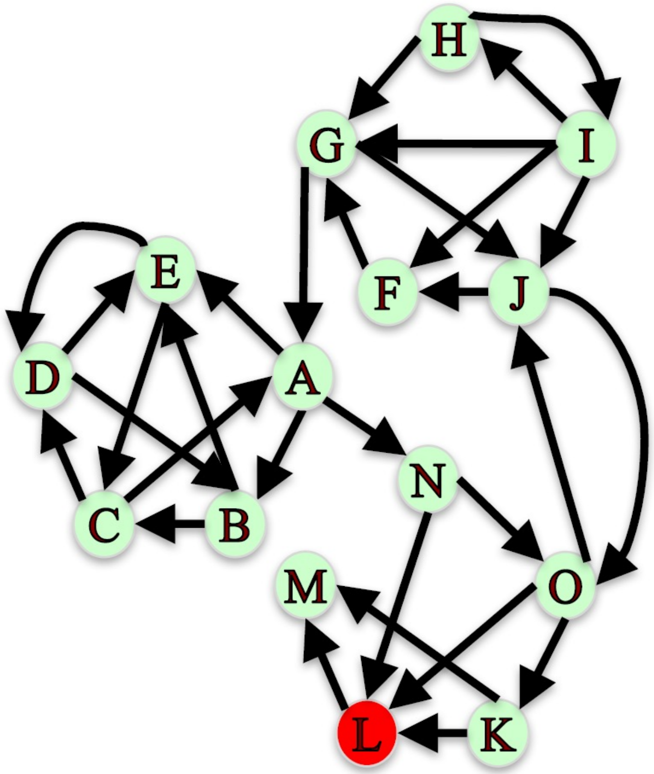
Closeness centrality – not reachable nodes

- What is the closeness centrality of node L?
- How to measure the closeness centrality of a node when it cannot reach all other nodes?

➢ **Option I**: Consider only the nodes reachable by the node

- $C_{close}(v) = \frac{|R(v)|}{\sum_{u \in R(v)} d(v,u)}$ R(v) is the set of nodes reachable by node v.
- $C_{close}(L) = \frac{1}{1} = 1$, only M is reachable.

➢ **Problem**: The centrality of 1 is too high for a node that can only reach one other node!



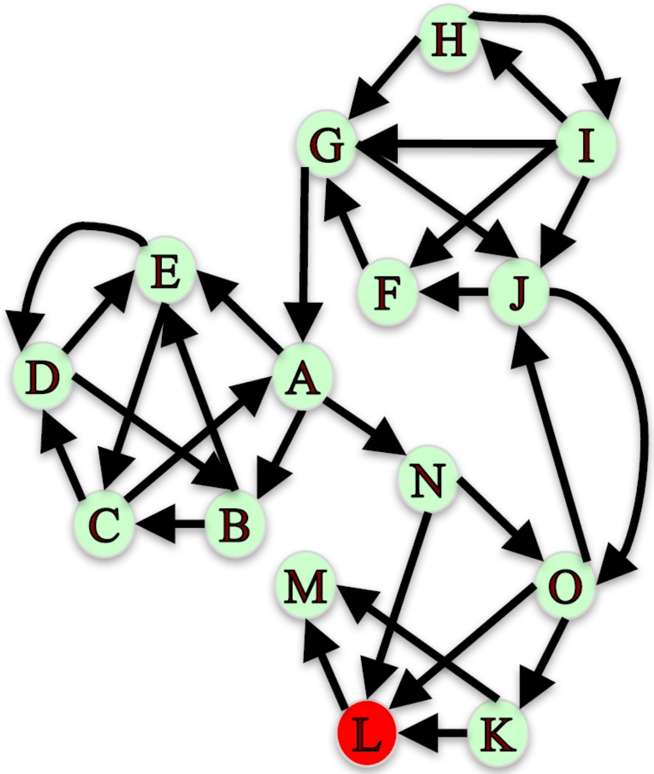
Closeness centrality – not reachable nodes

- What is the closeness centrality of node L?
- How to measure the closeness centrality of a node when it cannot reach all other nodes?

➢ **Option II:** Consider only the nodes reachable by the node and normalize by the fraction of reachable nodes.

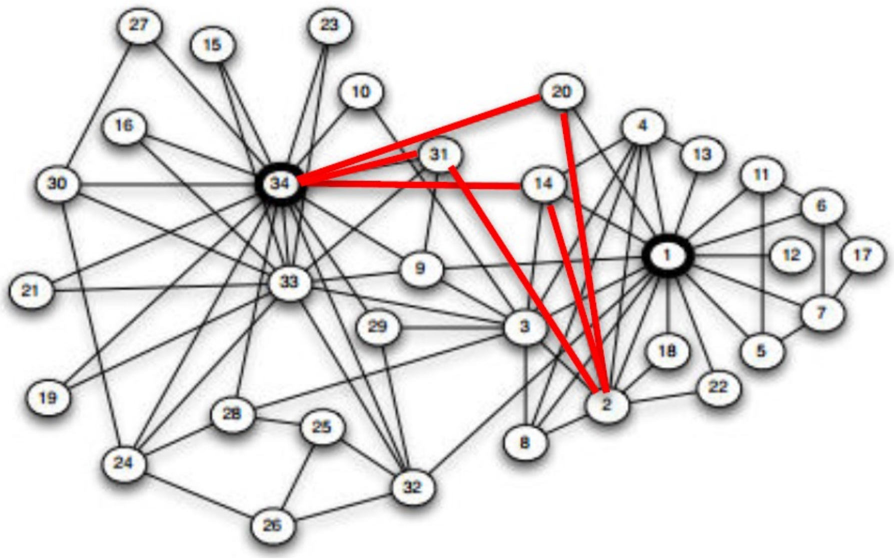
- $C_{close}(v) = \left[\frac{|R(v)|}{|N-1|}\right] \frac{|R(v)|}{\sum_{u \in R(v)} d(v,u)}$
- $C_{close}(L) = \left[\frac{1}{14}\right] \frac{1}{1} = 0.071$

➢ **Note:** this definition corresponds to our definition of closeness centrality when a graph is connected since: R(v)=N−1.



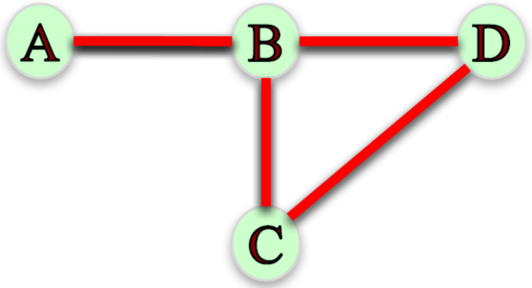
Betweeness centrality

- **Hypothesis**: important nodes are those that connect other nodes.
- **Reminder**: the distance between two nodes is the length of the shortest path between them.
- **Example**: The distance between nodes 34 and 2 is 2, and 3 paths are possible:
    - Path 1 : 34 – 31 – 2
    - Path 2 : 34 – 14 – 2
    - Path 3 : 34 – 20 – 2
  - Nodes 31, 14, and 20 are on the shortest path between nodes 34 and 2.



Betweeness centrality

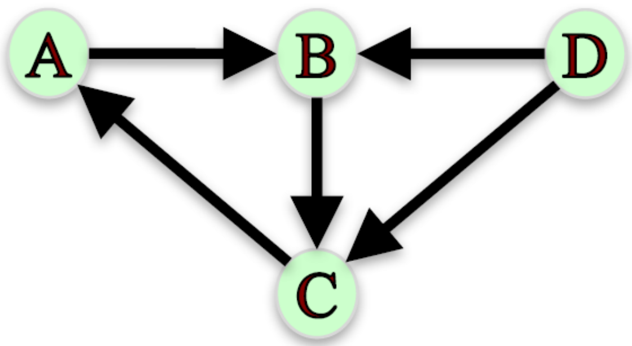
- **Hypothesis**: important nodes are those that connect other nodes.
- $\boldsymbol{C_{btw}(v)} = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$, where
  - $\sigma_{s,t}$ # of shortest paths between nodes s and t.
  - $\sigma_{s,t}(v)$ # of shortest paths between nodes s and t that pass through node v.
  - It is possible to include or exclude node v as node s and t in the calculation.
  - **Example.** If we exclude node v, we will have :



- $C_{btw}(B) = \frac{\sigma_{A,D}(B)}{\sigma_{A,D}} + \frac{\sigma_{A,C}(B)}{\sigma_{A,C}} + \frac{\sigma_{C,D}(B)}{\sigma_{C,D}} = 2$
- $C_{btw}(C) = ?$
- Max(C_{btw}) = ?

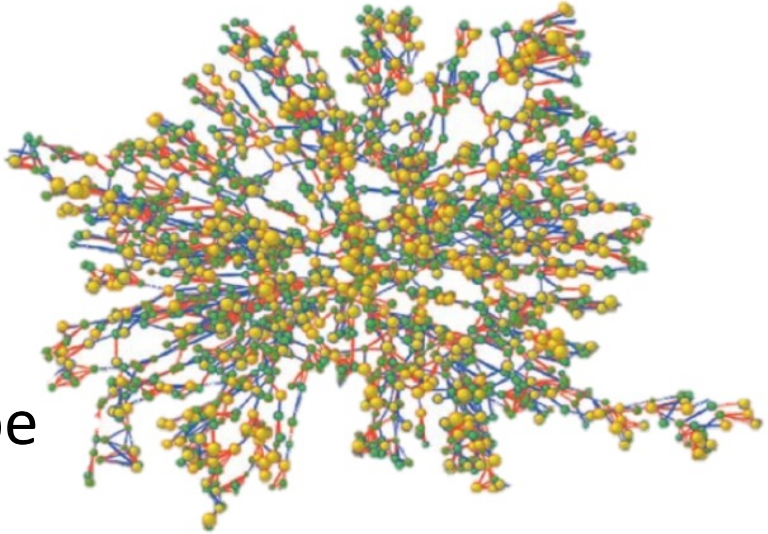
Betweeness centrality

- **Hypothesis**: important nodes are those that connect other nodes.
- $C_{btw}(v) = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$
- What if not all nodes are reachable?
  - ➢ Node D cannot be reached by any other node !
  - ➢ $\sigma_{A,D} = 0$, making the above formula undefined !
- **Solution**: consider only nodes s, t for which there is at least one path between them.
  - $C_{btw}(B) = \frac{\sigma_{A,C}(B)}{\sigma_{A,C}} + \frac{\sigma_{C,A}(B)}{\sigma_{C,A}} + \frac{\sigma_{D,C}(B)}{\sigma_{D,C}} + \frac{\sigma_{D,A}(B)}{\sigma_{D,A}} = 1$
  - $C_{btw}(C) = ?$



Betweeness centrality – normalization

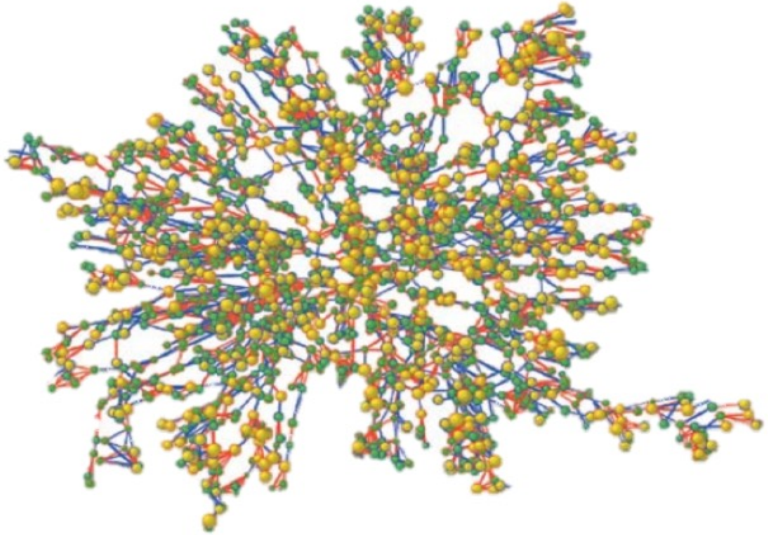
- **Hypothesis**: important nodes are those that connect other nodes.
- $\boldsymbol{C_{btw}(v)} = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$
- **Normalization**: the betweenness centrality value will be larger in larger graphs. To control this, we divide the centrality value by the number of node pairs in the graph (excluding v):
  - $\boldsymbol{C_{btw}(v)} = \left[\frac{1}{\frac{1}{2}(|N|-1)(|N|-2)}\right]\left[\sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}\right]$ (undirected graphs)
  - $\boldsymbol{C_{btw}(v)} = \left[\frac{1}{(|N|-1)(|N|-2)}\right]\left[\sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}\right]$ (directed graphs)



Network of friendship, marital ties, and family connections among 2,200 people

Betweeness centrality – complexity

- Calculating the betweenness centrality of all nodes can be very computationally expensive (time and space).
- Depending on the algorithm used, this calculation can have a complexity of $O(|N|^3)$.
- **Approximation**: instead of calculating the betweenness centrality for all pairs of nodes, we can approximate it by taking a sample of nodes.
  N=2200 nodes → ~4.8 millions of pairs



Network of friendship, marital ties, and family connections among 2,200 people

Betweeness centrality - Edges

- We can use betweenness centrality to find important links instead of nodes:
- $\boldsymbol{C_{btw}(e)} = \sum_{s,t \in N} \frac{\sigma_{s,t}(e)}{\sigma_{s,t}}$, where
  - $\sigma_{s,t}$ # of shortest paths between nodes s and t.
  - $\sigma_{s,t}(e)$ # of shortest paths between nodes s and t that pass through the link e.

Eigenvector centrality

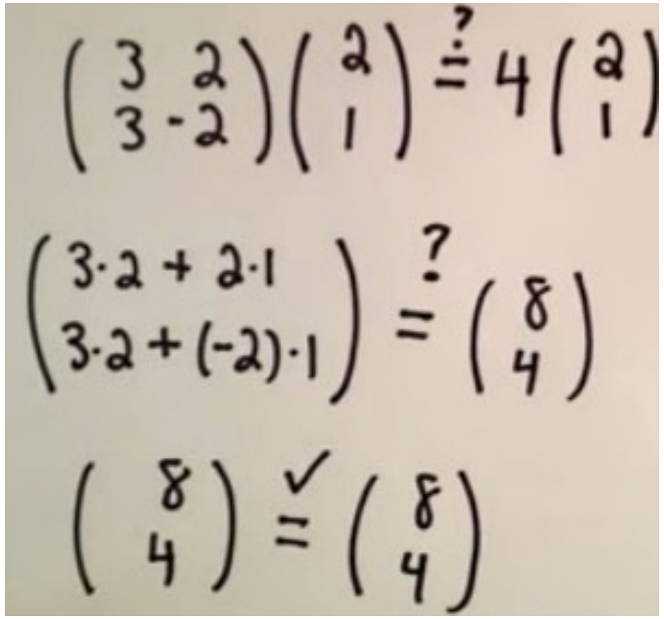
Hypothesis

- This metric measures how well a given actor is connected to other well-connected actors.
- The basic idea is that the importance of an actor is recursively defined by the importance of its neighbors. Thus, the centrality of a given node i is proportional to the sum of the centralities of i's neighbors.
- This score is given by the **principal eigenvector** of the adjacency matrix.

Eigenvector centrality

Eigenvalue & Eigenvector

- Let A be a square matrix of size n×n.
- The scalar λ is called an **eigenvalue** of A if there exists a non-zero vector X such that **AX=λX**.
  This vector X is called the **eigenvector** of A corresponding to the eigenvalue λ.
- It is the vector that retains its direction after being multiplied by the matrix.



Eigenvector centrality- calculation

- **Eigenvalues**: AX= λX ⇔
  AX= λIX ⇔
  AX- λIX = O ⇔
  (A-λI)X=O ➔
  det(A- λI)=0
- **Eigenvectors**: (A-λI)X=O

$$\mathtt{A} = \begin{bmatrix} 1 & 2 & 2 \\ 0 & 2 & 1 \\ -1 & 2 & 2 \end{bmatrix} \qquad \mathtt{A} - \lambda \mathtt{Id} = \begin{bmatrix} 1-\lambda & 2 & 2 \\ 0 & 2-\lambda & 1 \\ -1 & 2 & 2-\lambda \end{bmatrix}$$

$$\det(\mathtt{A} - \lambda \mathtt{Id}) = 4 - 8\lambda + 5\lambda^2 - \lambda^3 \qquad \lambda = 1 \qquad \boxed{\lambda = 2}$$

$$\begin{bmatrix} -1 & 2 & 2 \\ 0 & 0 & 1 \\ -1 & 2 & 0 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \boxed{\begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}}$$

Eigenvector centrality - calculation

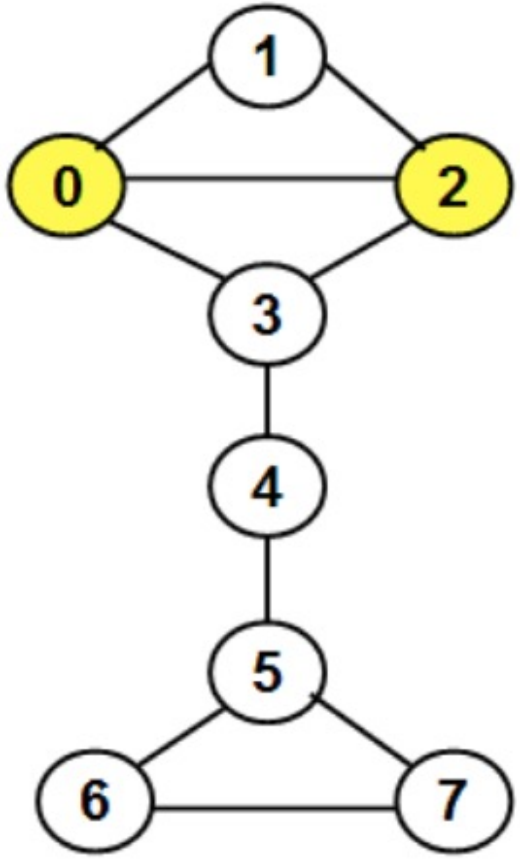
Example

How to calculate the eigenvector centrality for this network?

Iteration 1

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \\ 3 \\ 3 \\ 2 \\ 3 \\ 2 \\ 2 \end{bmatrix} \equiv \begin{bmatrix} 0.416 \\ 0.277 \\ 0.416 \\ 0.416 \\ 0.277 \\ 0.416 \\ 0.277 \\ 0.277 \end{bmatrix}$$

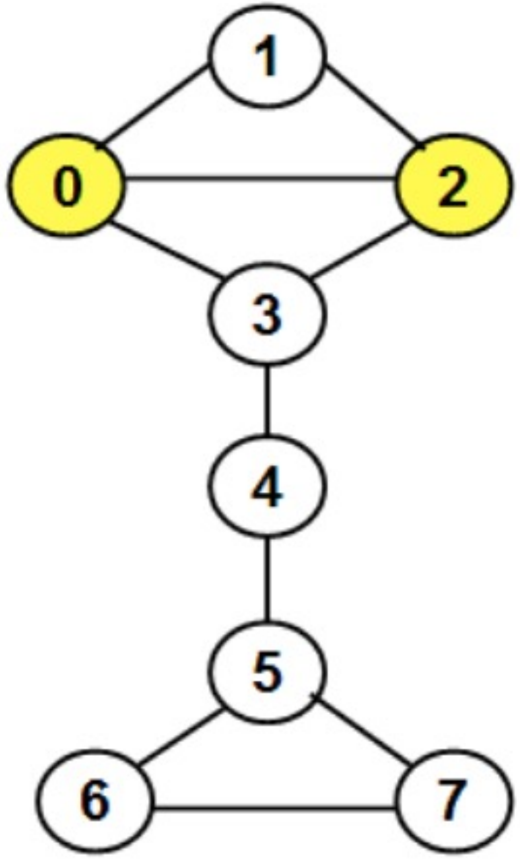
Normalized Value = 7.21



Eigenvector centrality - calculation

Example

How to calculate the eigenvector centrality for this network?



Iteration 2

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0.416 \\ 0.277 \\ 0.416 \\ 0.416 \\ 0.277 \\ 0.416 \\ 0.277 \\ 0.277 \end{bmatrix} = \begin{bmatrix} 1.109 \\ 0.832 \\ 1.109 \\ 1.109 \\ 0.832 \\ 0.831 \\ 0.693 \\ 0.693 \end{bmatrix} \equiv \begin{bmatrix} 0.428 \\ 0.321 \\ 0.428 \\ 0.428 \\ 0.321 \\ 0.321 \\ 0.268 \\ 0.268 \end{bmatrix}$$

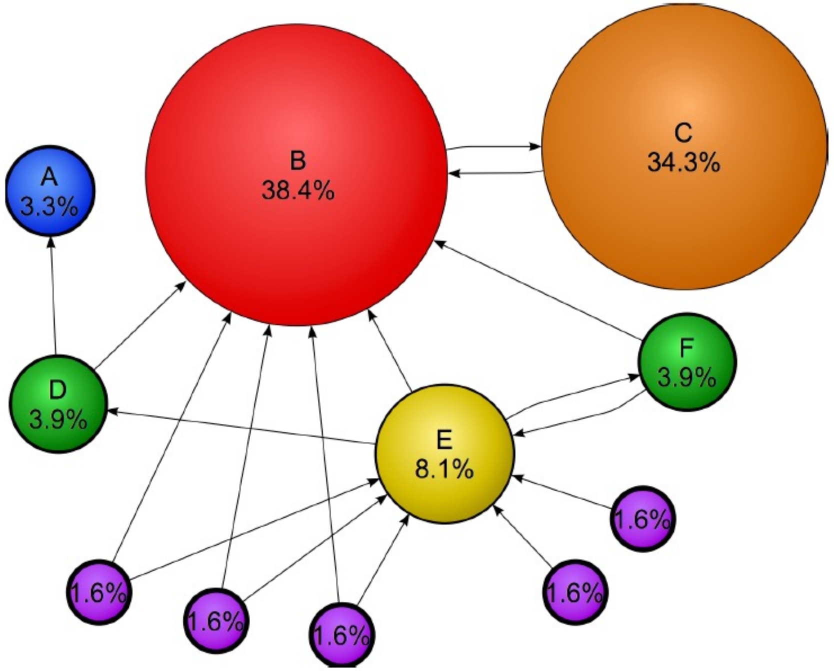
Normalized Value = 2.59

After 7 iterations:

0	1	2	3	4	5	6	7
0.489	0.364	0.489	0.467	0.264	0.232	0.155	0.155

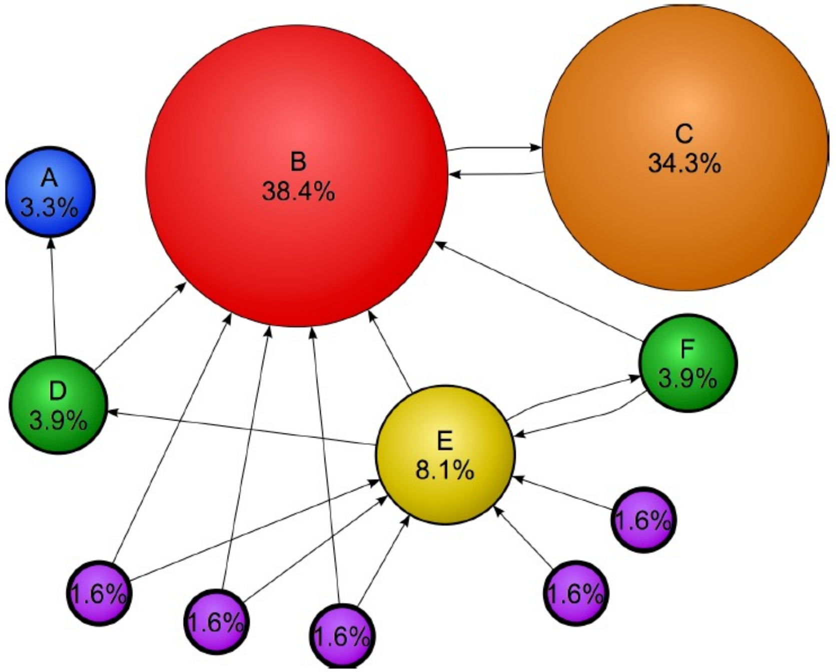
PageRank

- Developed by the founders of Google to measure the importance of web pages based on the structure of hyperlinks in the network.
- **PageRank** assigns an importance score to each node. Important nodes are those with many incoming links from other important pages.
- PageRank can be used for any type of network but is particularly useful for **directed** networks.
- The PageRank of a node depends on the PageRank of other nodes (circular definition).



PageRank – Calculation

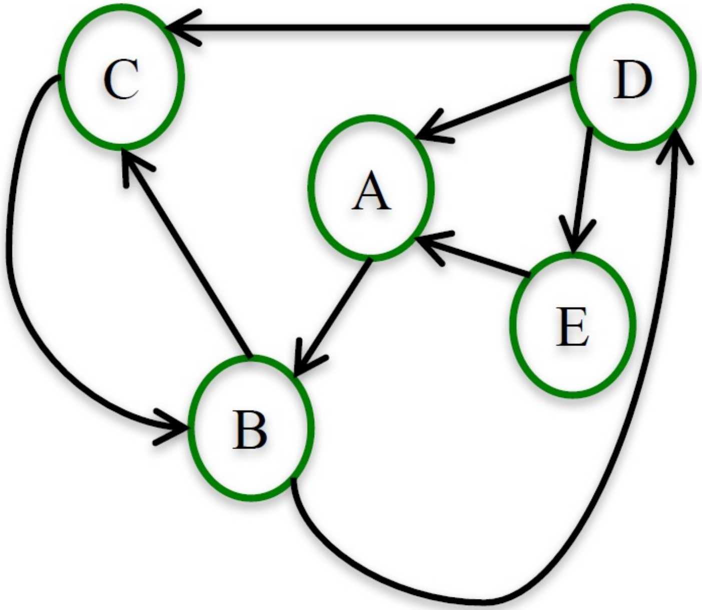
- **Algorithm**:
  1. Assign a PageRank of 1/n to all nodes.
  2. Execute the PageRank update rule k times.
     - n: number of nodes in the network.
     - k: number of iterations.
- **Basic PageRank update rule**:
  Each node distributes an equal share of its current PageRank to all the nodes it links to.
- The new PageRank of each node is the sum of all the PageRank it has received from other nodes.



PageRank – Example

- Which node should be the most **important** in this network?
- Calculate the PageRank of each node after 2 iterations of the algorithm (k=2).

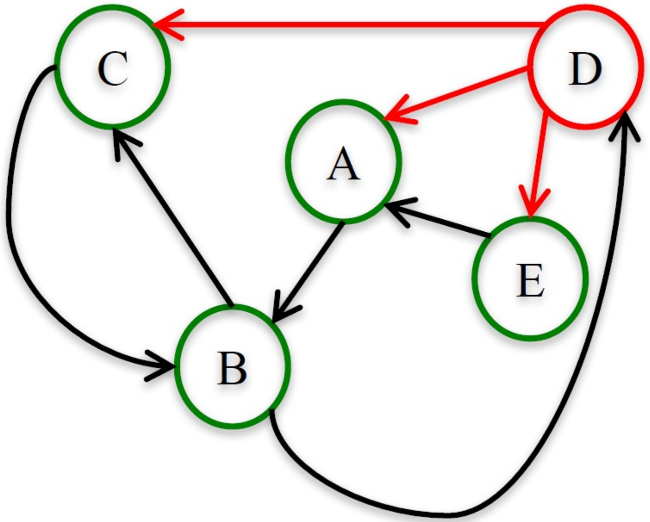
Page Rank					
	A	B	C	D	E
	1/5	1/5	1/5	1/5	1/5

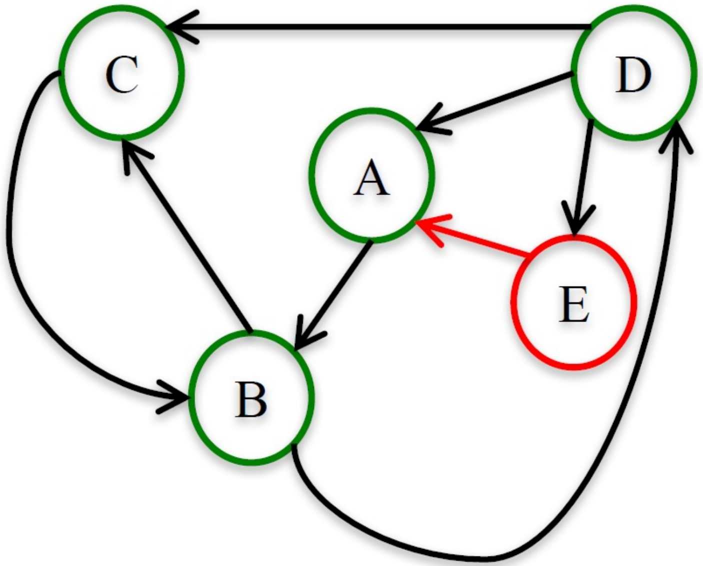


PageRank – Example

Page Rank (k = 1)					
	A	B	C	D	E
Old	1/5	1/5	1/5	1/5	1/5
New					

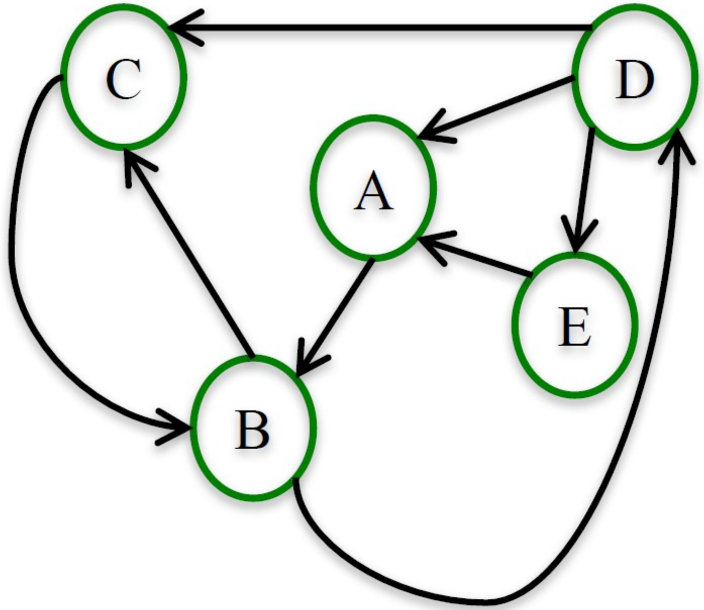
A: (1/3)*(1/5) + 1/5 = 4/15
B: ?
C: ?
D: ?
E: ?





PageRank – Example

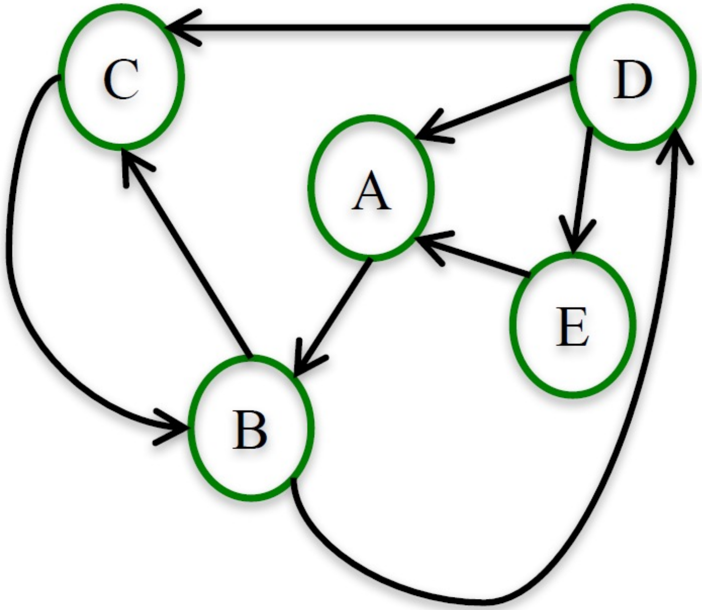
Page Rank (k = 1)					
	A	B	C	D	E
Old	1/5	1/5	1/5	1/5	1/5
New	4/15	2/5	1/6	1/10	1/15



A: (1/3)*(1/5) + 1/5 = 4/15
B: 1/5 + 1/5 = 2/5
C: (1/3)*(1/5) + (1/2)*(1/5) = 5/30 = 1/6
D: (1/2)*(1/5) = 1/10
E: (1/3)*(1/5) = 1/15

PageRank – Example

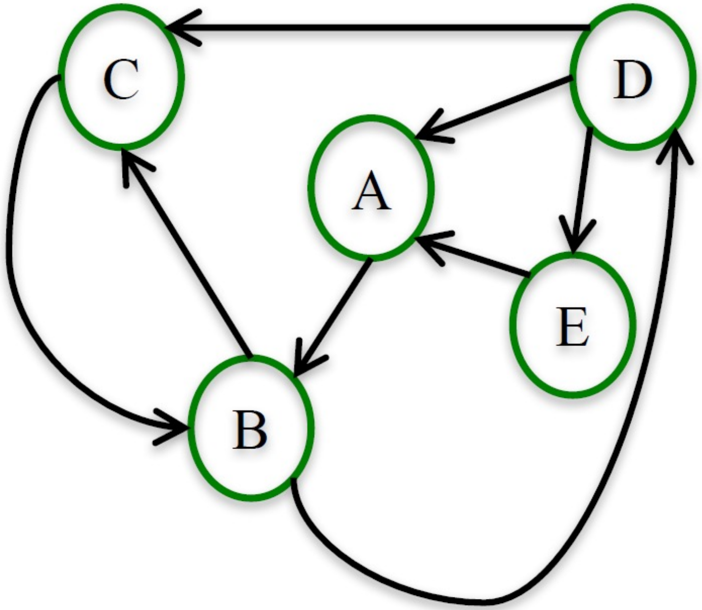
Page Rank (k = 2)					
	A	B	C	D	E
Old	4/15	2/5	1/6	1/10	1/15
New	1/10	13/30	7/30	2/10	1/30



A: (1/3)*(1/10) + 1/15 = 1/10
B: 1/6 + 4/15 = 13/30
C: (1/3)*(1/10) + (1/2)*(2/5) = 7/30
D: (1/2)*(2/5) = 2/10
E: (1/3)*(1/10) = 1/30

PageRank – Example

	Page Rank				
	A	B	C	D	E
k=2	1/10	13/30	7/30	2/10	1/30
k=2	.1	.43	.23	.20	.03
k=3	.1	.33	.28	.22	.06
k=∞	.12	.38	.25	.19	.06

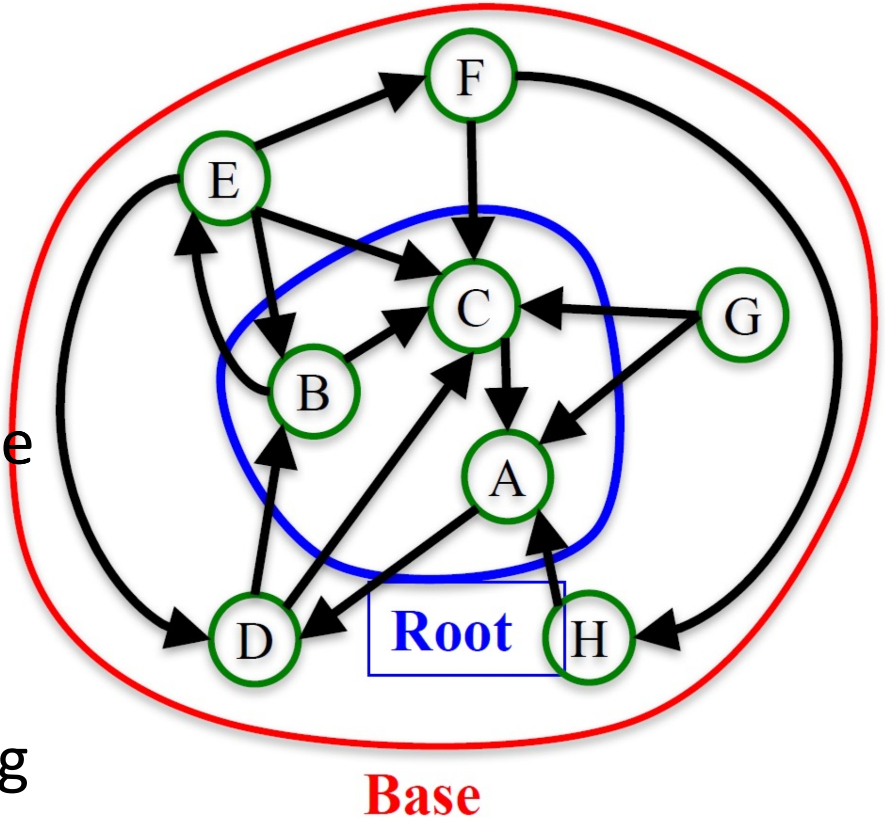


For most networks, the PageRank values converge.

Hubs & Authority

Context: Given a query submitted to a search engine:

- **Root:** A set of highly relevant web pages (e.g., pages containing the query string) – **potential authorities.**
- Find all the pages that lead(link) to a relevant page (Root) – **potential hubs**.
- **Base:** Nodes in the Root and any node linked to a node in the Root.
- **Important pages:** Consider all the links connecting nodes within the base set (a subset of pages).



Hubs & Authority – Calculation

Algorithm (HITS):

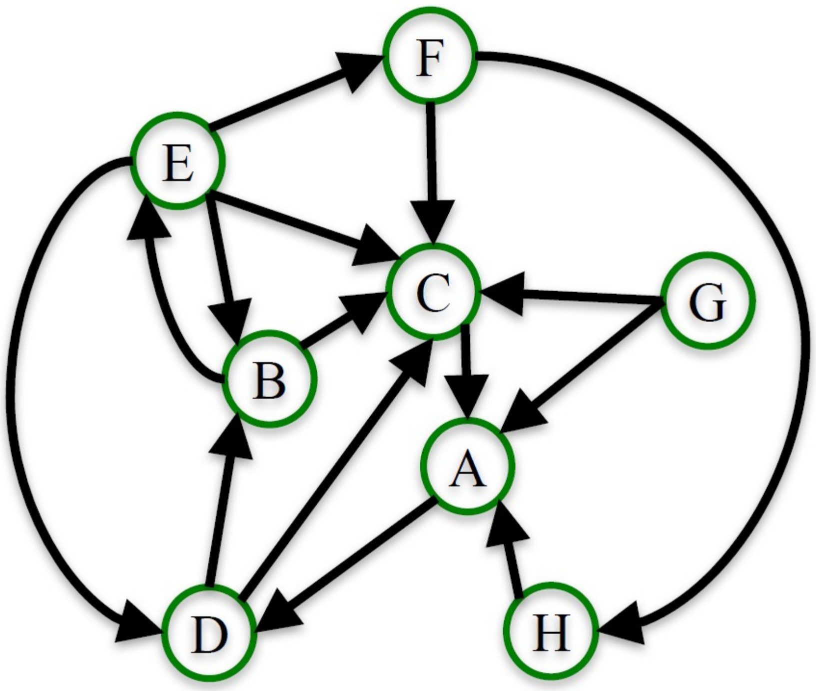
1. Assign each node an **authority score** and a **hub score** equal to 1.
2. Apply the **authority update rule**: the authority score of each node is the sum of the hub scores of all nodes pointing to it.
3. Apply the **hub update rule**: the hub score of each node is the sum of the authority scores of all nodes it points to.
4. **Normalize** the authority and hub scores:

$$Aut(j) = \frac{Aut(j)}{\sum_{i \in N} Aut(i)}, \quad Hub(j) = \frac{Hub(j)}{\sum_{i \in N} Hub(i)}$$

1. Repeat k times.

Hubs & Authority – Example

	Old Auth	Old Hub	New Auth	New Hub
A	1	1	3/15	1/15
B	1	1	2/15	2/15
C	1	1	5/15	1/15
D	1	1	2/15	2/15
E	1	1	1/15	4/15
F	1	1	1/15	2/15
G	1	1	0/15	2/15
H	1	1	1/15	1/15



Normalisation

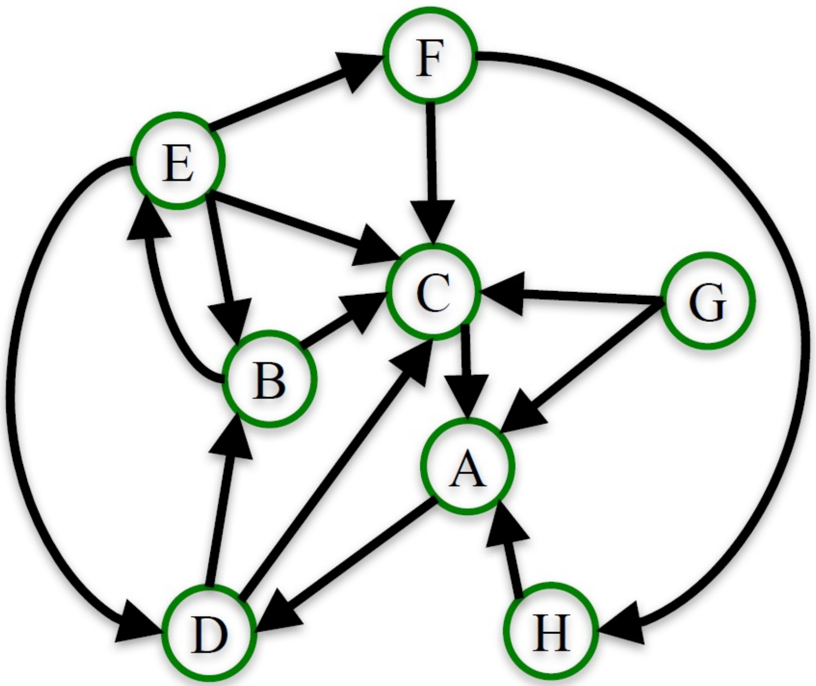
$\sum_{i \in N} Aut(i) = 15, \sum_{i \in N} Hub(i) = 15$

Hubs & Authority – Example

	Old Auth	Old Hub	New Auth	New Hub
A	1/5	1/15	4/35	2/45
B	2/15	2/15	6/35	2/15
C	1/3	1/15	12/35	1/15
D	2/15	2/15	1/7	7/45
E	1/15	4/15	2/35	2/9
F	1/15	2/15	4/35	2/15
G	0	2/15	0	8/45
H	1/15	1/15	2/35	1/15

Normalisation

$\sum_{i\in N} Aut(i) = \frac{35}{15}, \sum_{i\in N} Hub(i) = \frac{45}{15} = 3$



Hubs & Authority – Example

- For most networks, as k increases, the authority and hub scores converge to a unique value.
- When k→∞, the hub and authority scores approach:

	A	B	C	D	E	F	G	H
Auth	.08	.19	.40	.13	.06	.11	0	.06
Hub	.04	.14	.03	.19	.27	.14	.15	.03