



University of Stuttgart  
Germany

# Complex Network Systems

Structural metrics

Ilche Georgievski

[ilche.georgievski@iaas.uni-stuttgart.de](mailto:ilche.georgievski@iaas.uni-stuttgart.de)

Room: U38 0.353

2019/2020

Winter

# Types

## Graph-level metrics



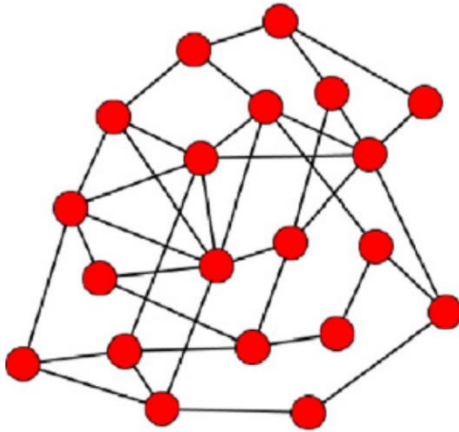
### Size

- Density
- Paths and distances
- Neighbourhoods
- Egocentric network
- Clustering coefficient
- Transitivity
- Cores
- Cliques
- Communities

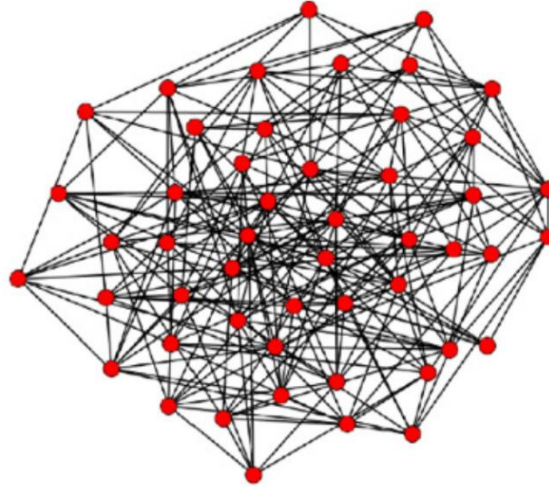
## Node-level metrics

- Closeness centrality
- Betweenness centrality
- Degree centrality
- Eigenvector centrality
- Katz centrality
- PageRank

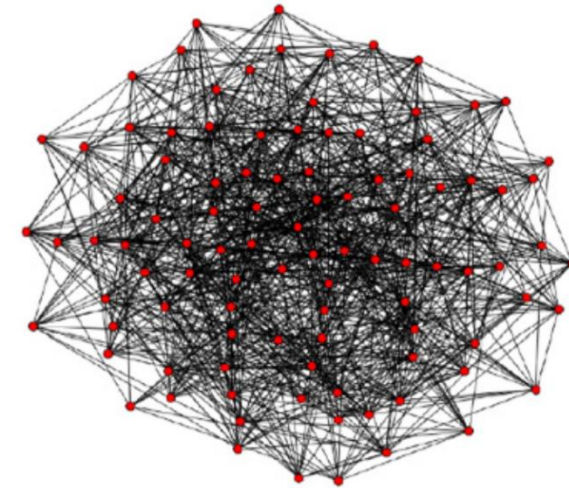
# Density



20 nodes, 38 links  
Density = 0.20  
Links per node = 1.9



50 nodes, 245 links  
Density = 0.20  
Links per node = 4.9



100 nodes, 990 links  
Density = 0.20  
Links per node = 9.9

Hoppe, B. and Reinelt, C. (2010) Social network analysis and the evaluation of leadership networks, *The Leadership Quarterly*, 21(4), pp. 600-619.

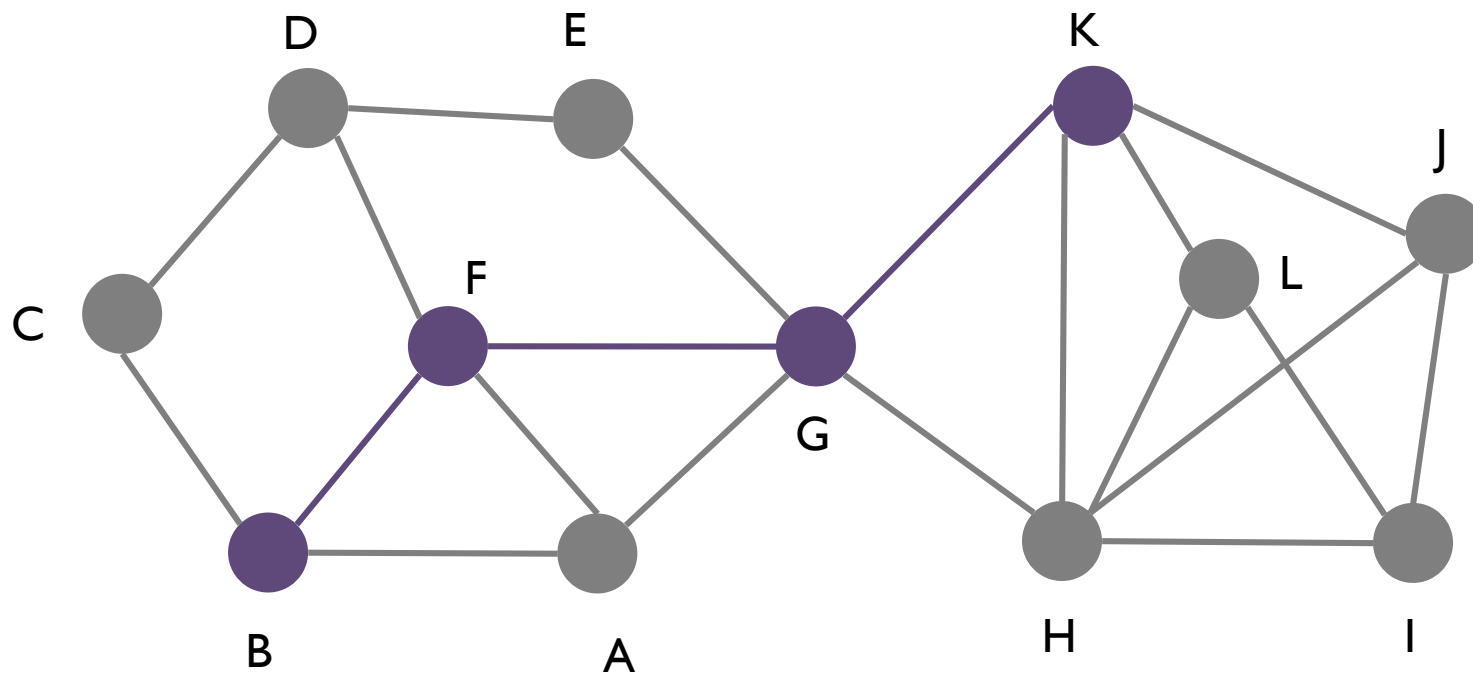
# Density

- Ratio of existing edges to all possible edges
- Gives a sense of how closely knit the network is
- `nx.density(G)`

# Paths

- Path
- Cycle
- Eulerian path
- Hamiltonian path

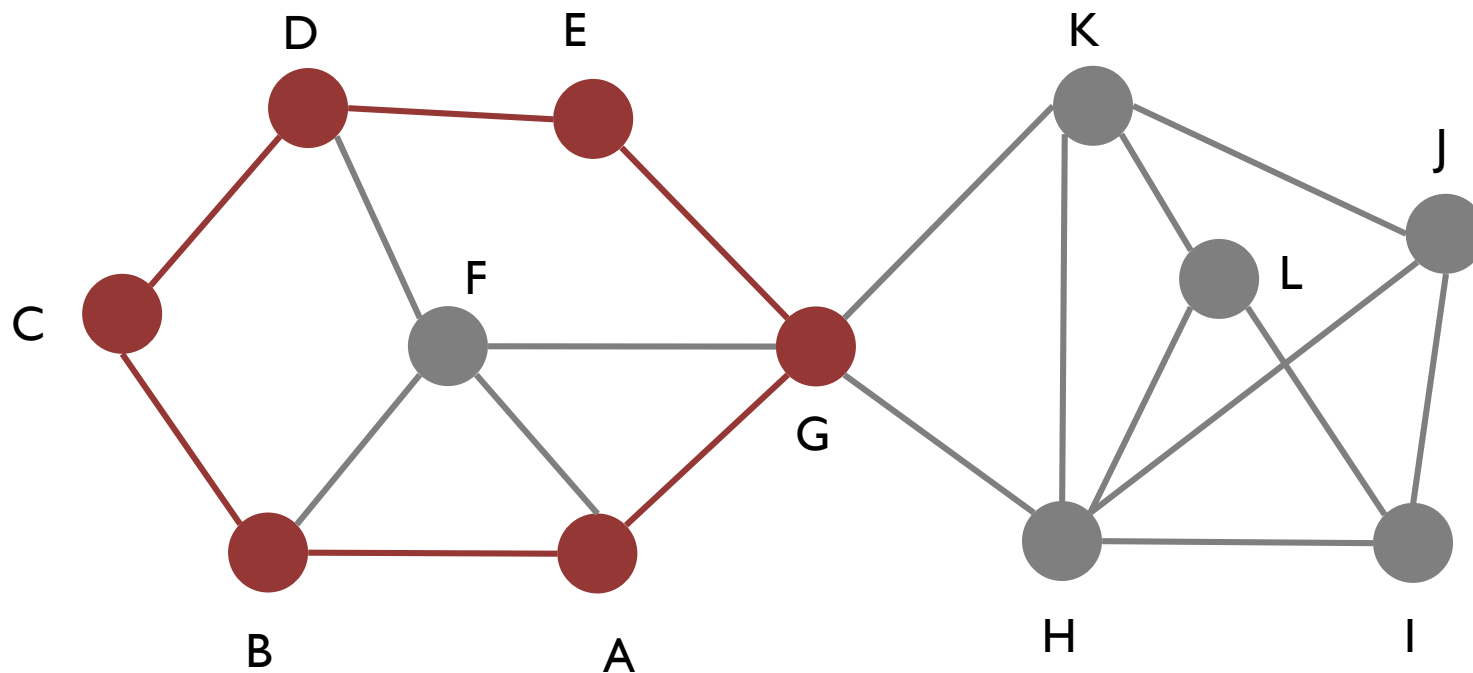
# Path



$$P_{BK} = ?$$

$$P_{BK} = \{(B, F), (F, G), (G, K)\}$$

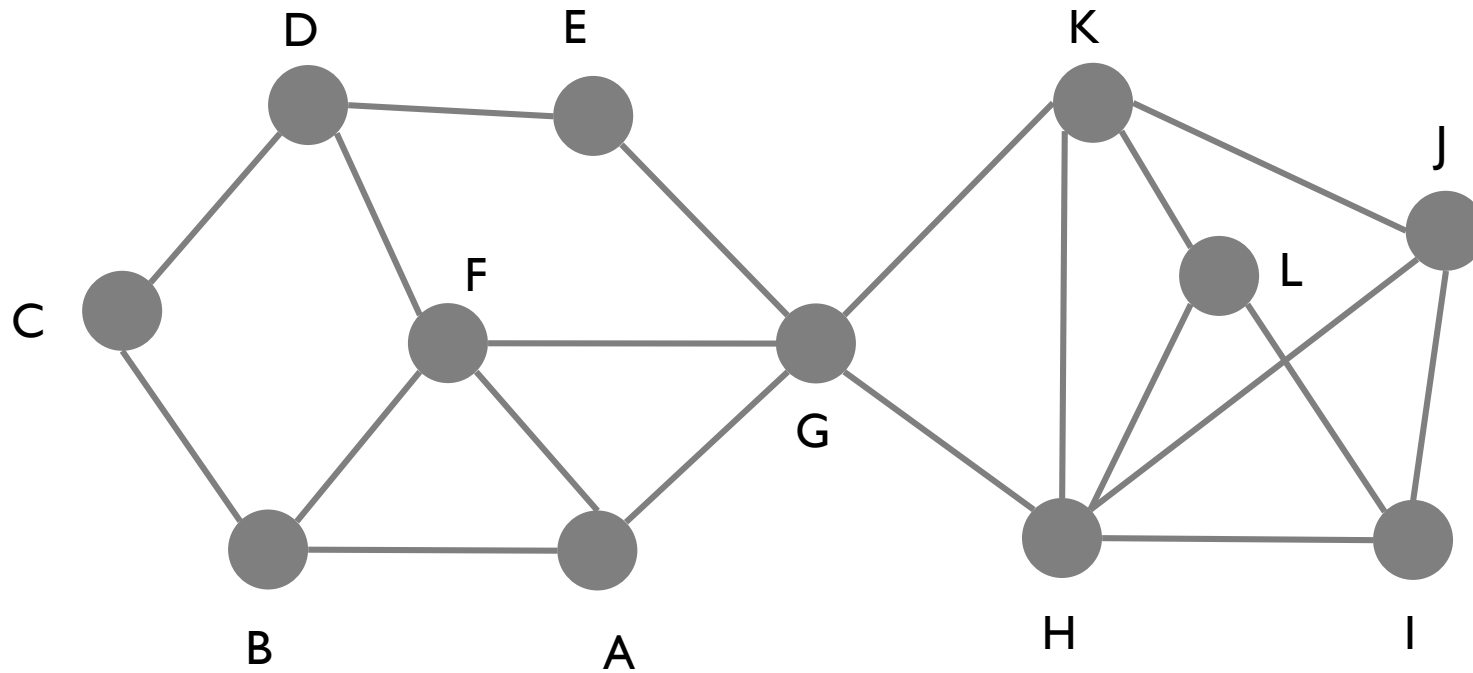
# Cycle



$$P_{BB} = ?$$

$$P_{BB} = \{(B, A), (A, G), (G, E), (E, D), (D, C), (C, B)\}$$

# Eulerian path

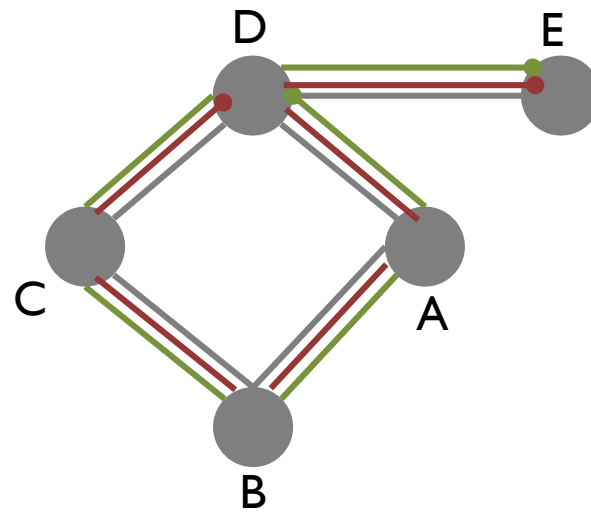


$P_{Eulerian} = ?$



# Eulerian path

An Euler path is a path that passes through every "edge" exactly once.  
If it ends at the initial vertex then it is an Euler cycle.

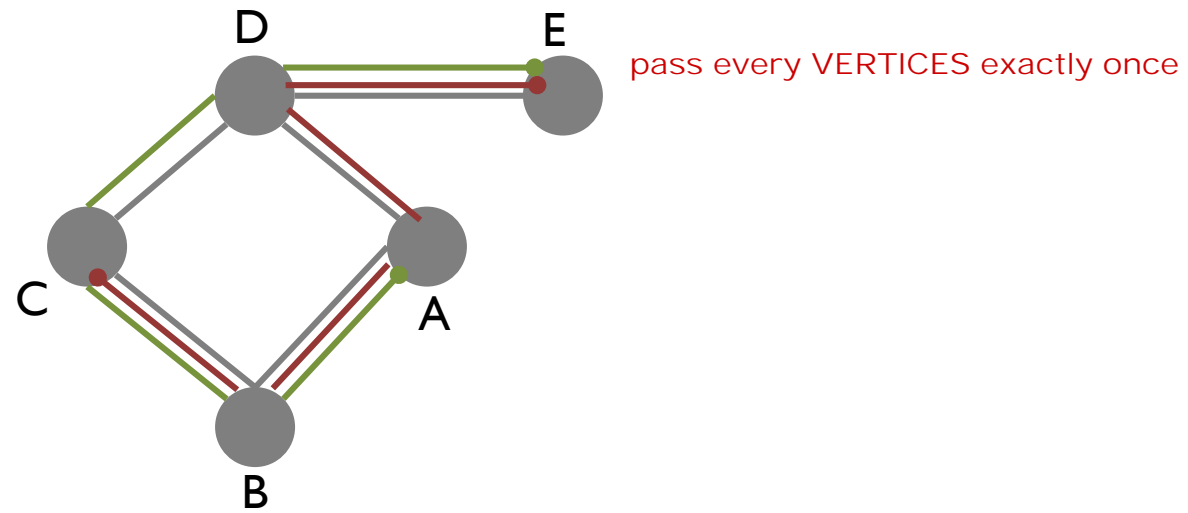


pass every EDGES! exactly once

$P_{Eulerian} = ?$

# Hamiltonian path

A Hamiltonian path, also called a Hamilton path, is a graph path between two vertices of a graph that visits each vertex exactly once.

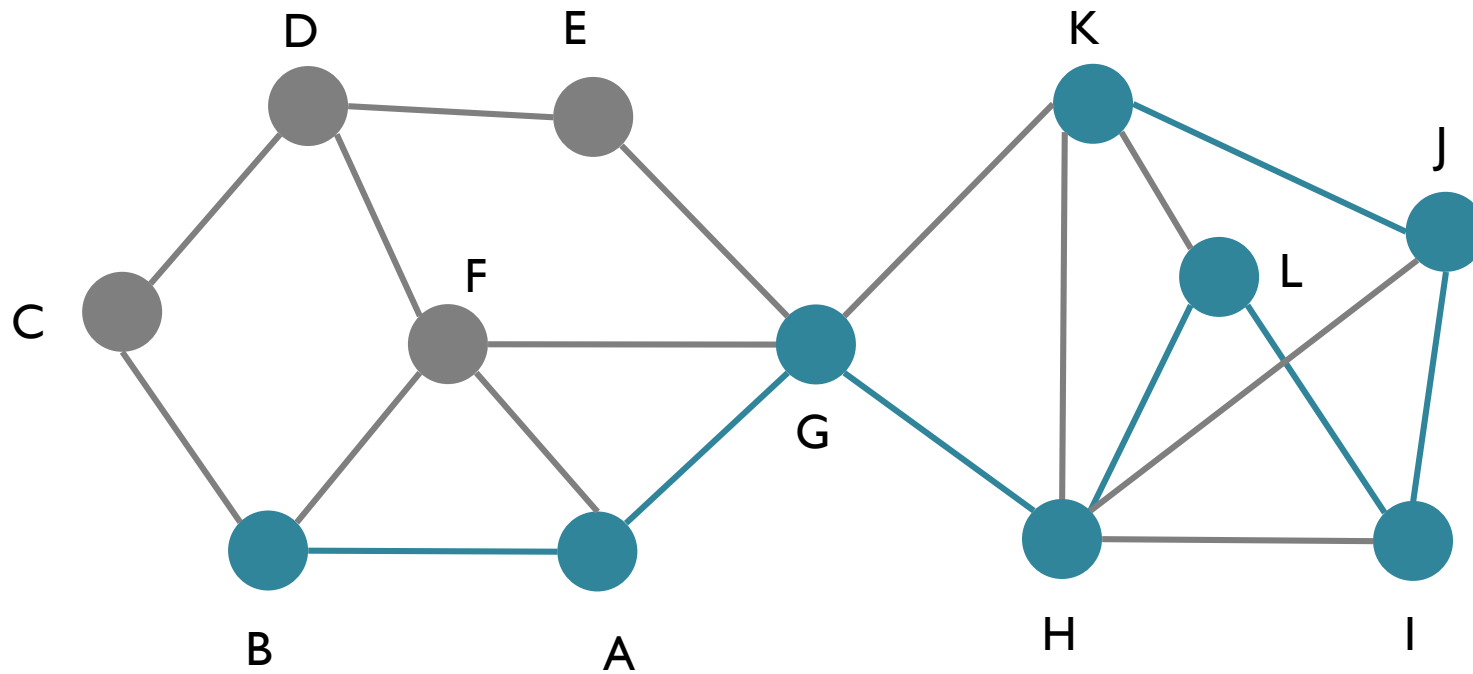


$P_{Hamiltonian} = ?$

# Distance measures

- Path length
- Shortest path
- Average path length
- Eccentricity
- Diameter
- Radius
- Center
- Periphery

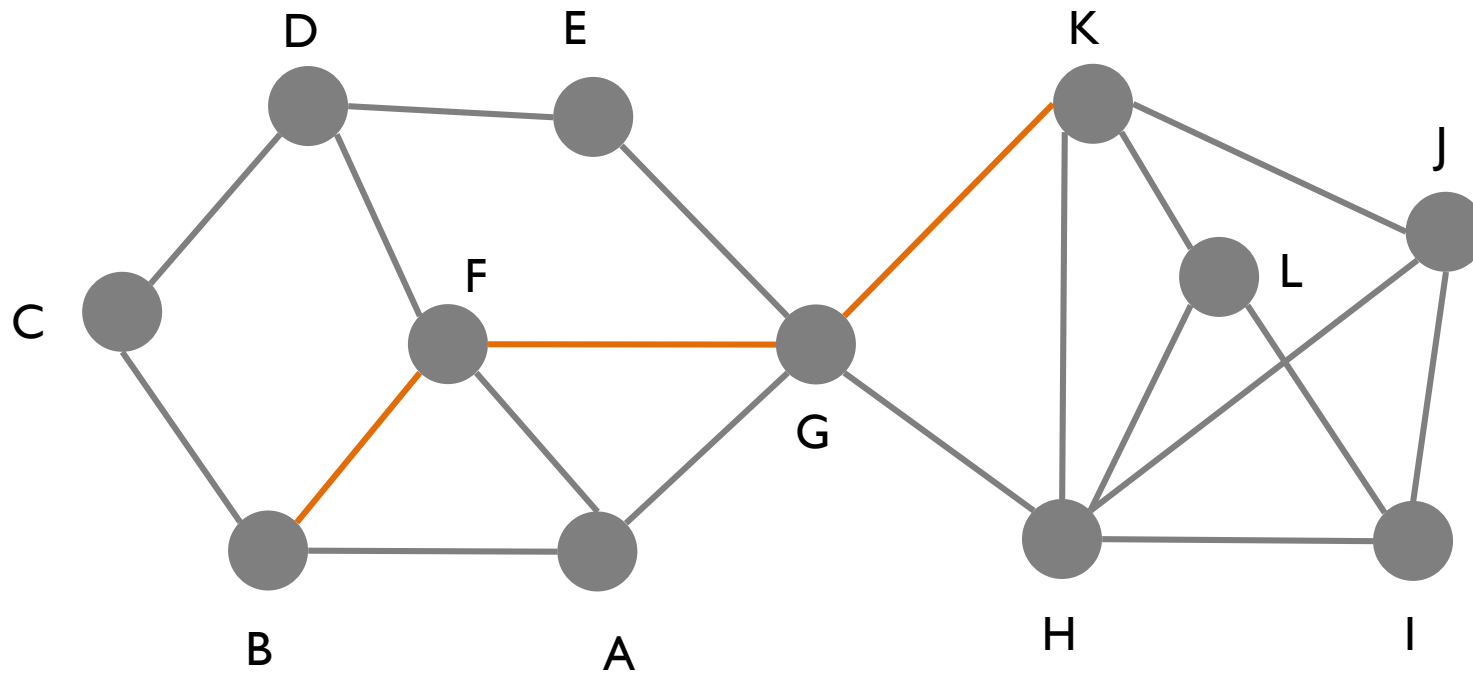
# Path length



$$n_{BK} = ?$$

$$n_{BK} = 7$$

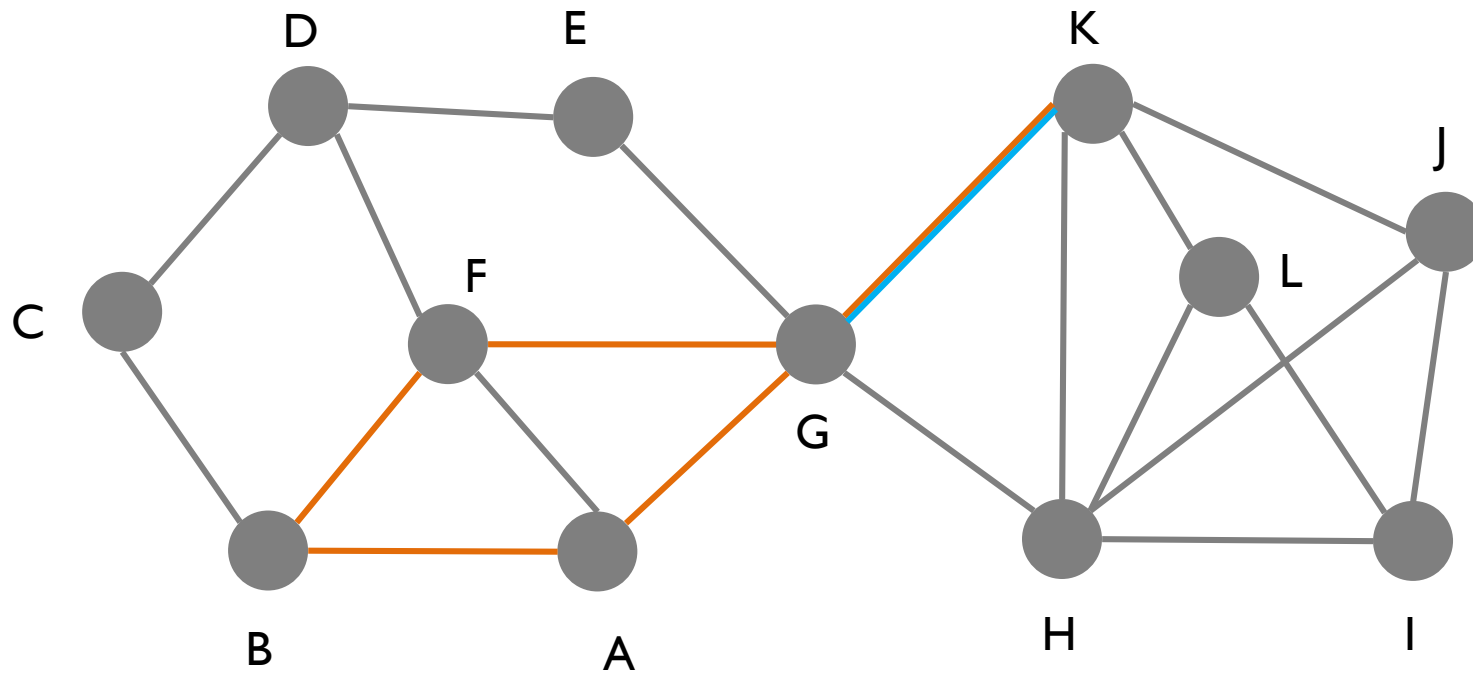
# Shortest path



$$d_{BK} = ?$$

$$d_{BK} = 3$$

# Shortest path



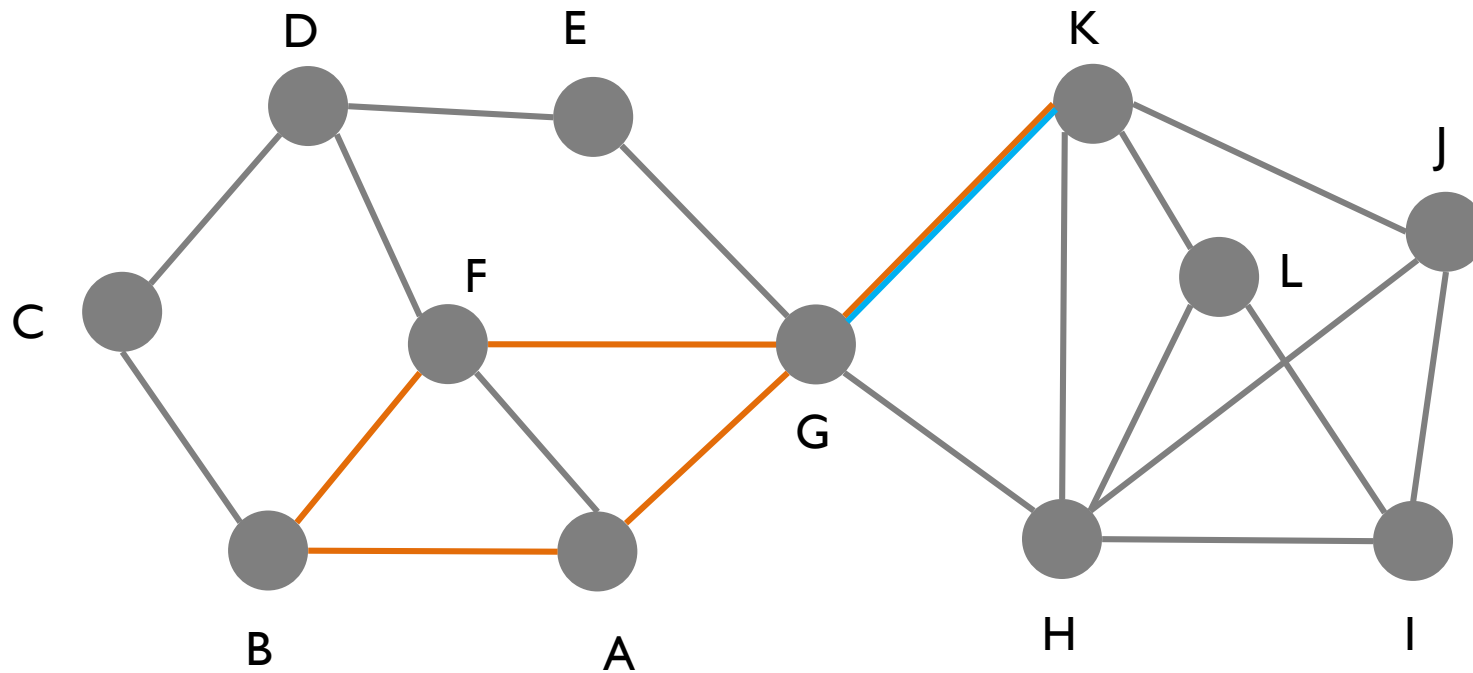
$$d_{BK} = ?$$

$$d_{BK} = 3$$

# Shortest path

Distance

Geodesic

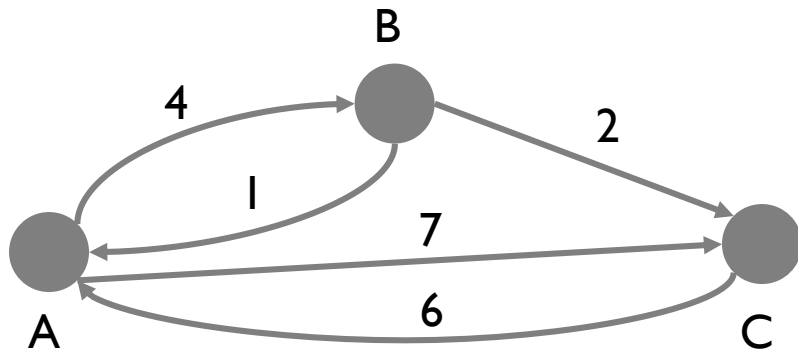


$$d_{BK} = ?$$

$$d_{BK} = 3$$

How to determine the distance between two vertices?





# Floyd-Warshall algorithm

an algorithm for finding shortest paths in a weighted graph with positive or negative edge weights (but with no negative cycles).

$$D^{(0)} = \begin{bmatrix} 0 & 4 & 7 \\ 1 & 0 & 2 \\ 6 & \infty & 0 \end{bmatrix}$$

$$D^{(1)} = \begin{bmatrix} 0 & 4 & 7 \\ 1 & 0 & 2 \\ 6 & 10 & 0 \end{bmatrix}$$

Take vertex A

$$\begin{aligned} D_{CB} &= D_{CA} + D_{AB} \\ D_{CB} &= 6 + 4 = 10 \end{aligned}$$

$$D^{(2)} = \begin{bmatrix} 0 & 4 & 6 \\ 1 & 0 & 2 \\ 6 & 10 & 0 \end{bmatrix}$$

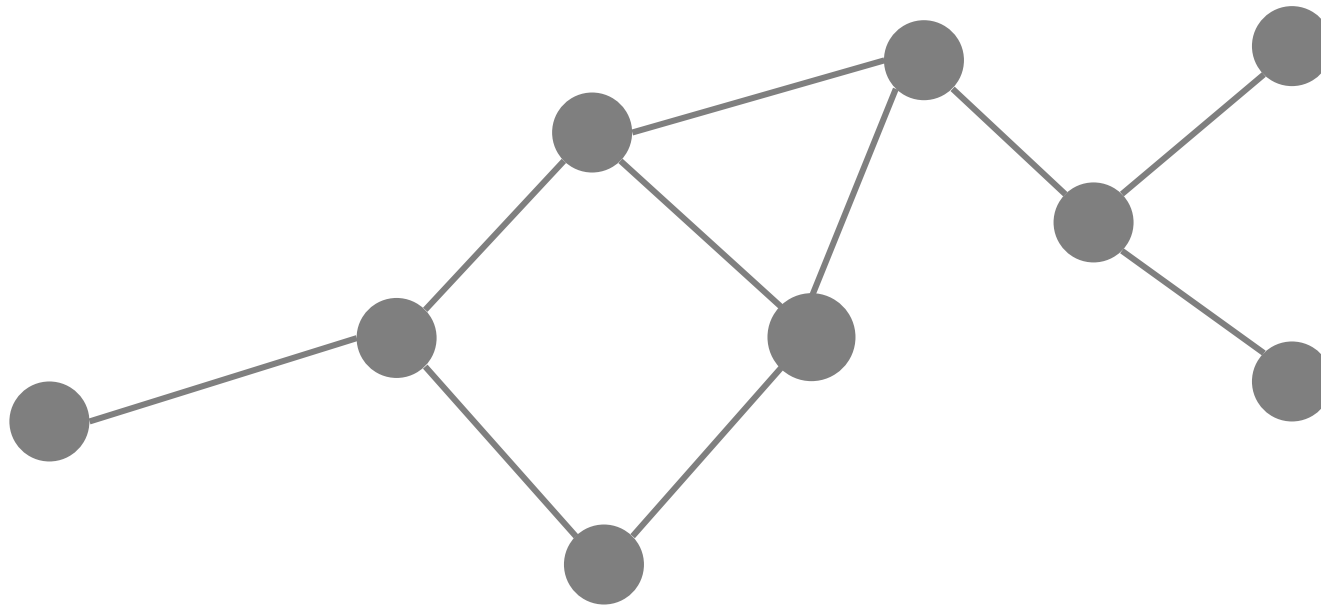
Take vertex B

$$\begin{aligned} D_{AC} &= D_{AB} + D_{BC} \\ D_{AC} &= 4 + 2 = 6 \end{aligned}$$

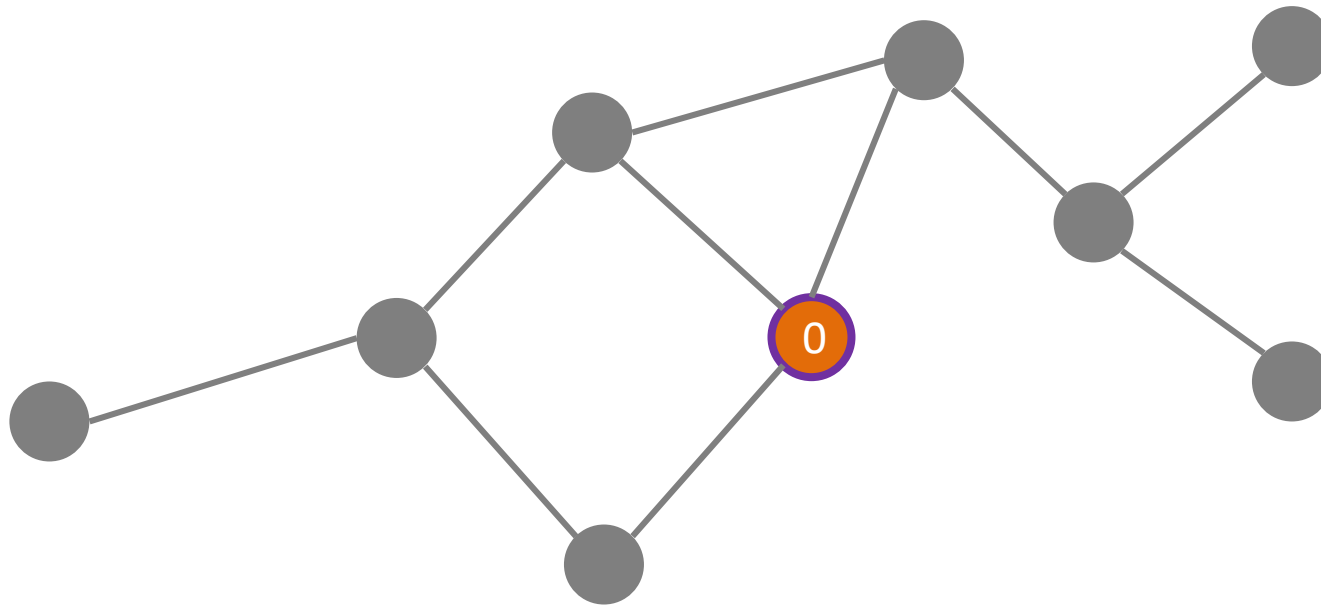
$$D^{(3)} = \begin{bmatrix} 0 & 4 & 6 \\ 1 & 0 & 2 \\ 6 & 10 & 0 \end{bmatrix}$$

Take vertex C

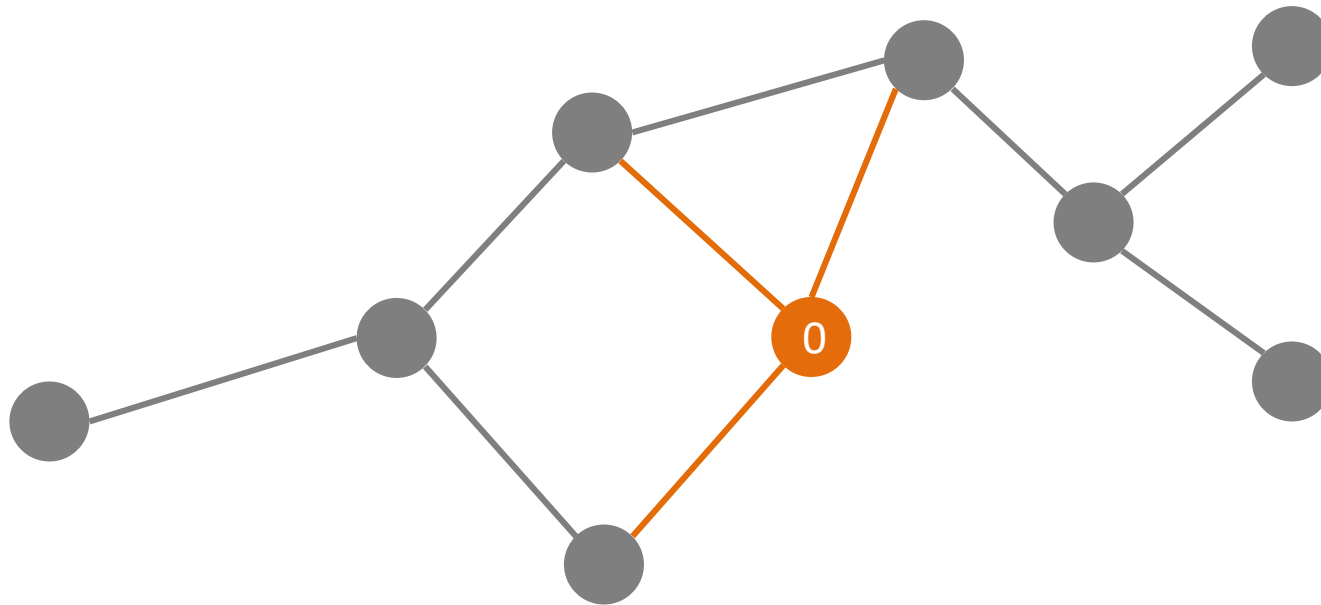
# Breadth-first search



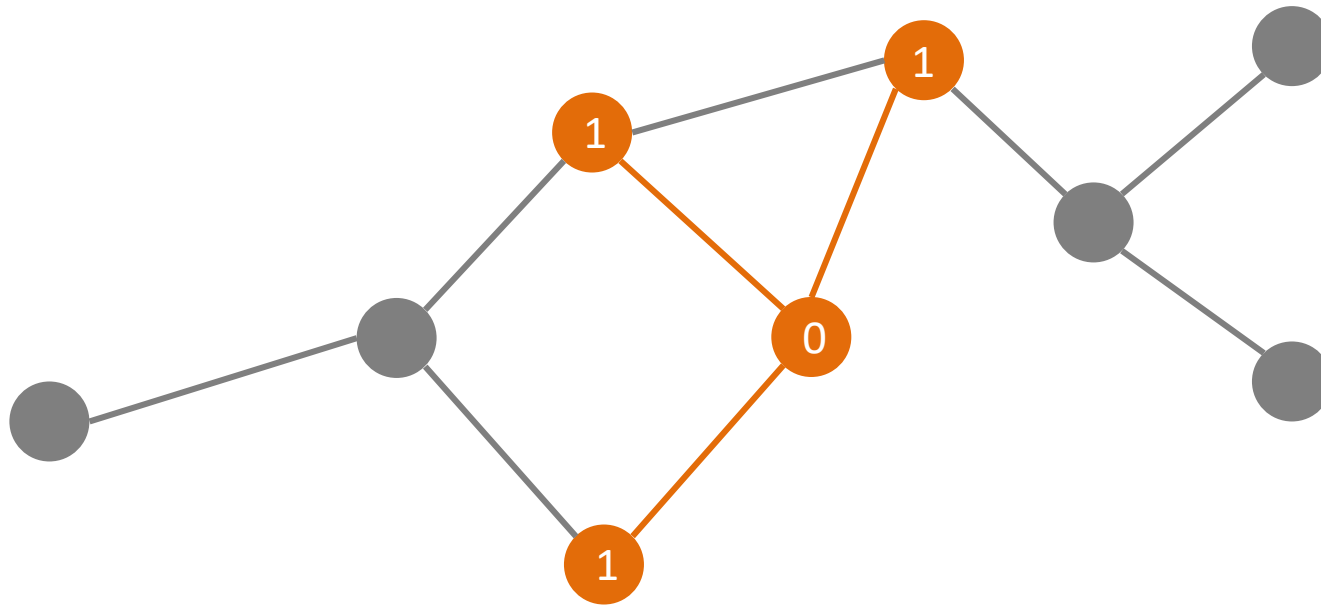
# Breadth-first search



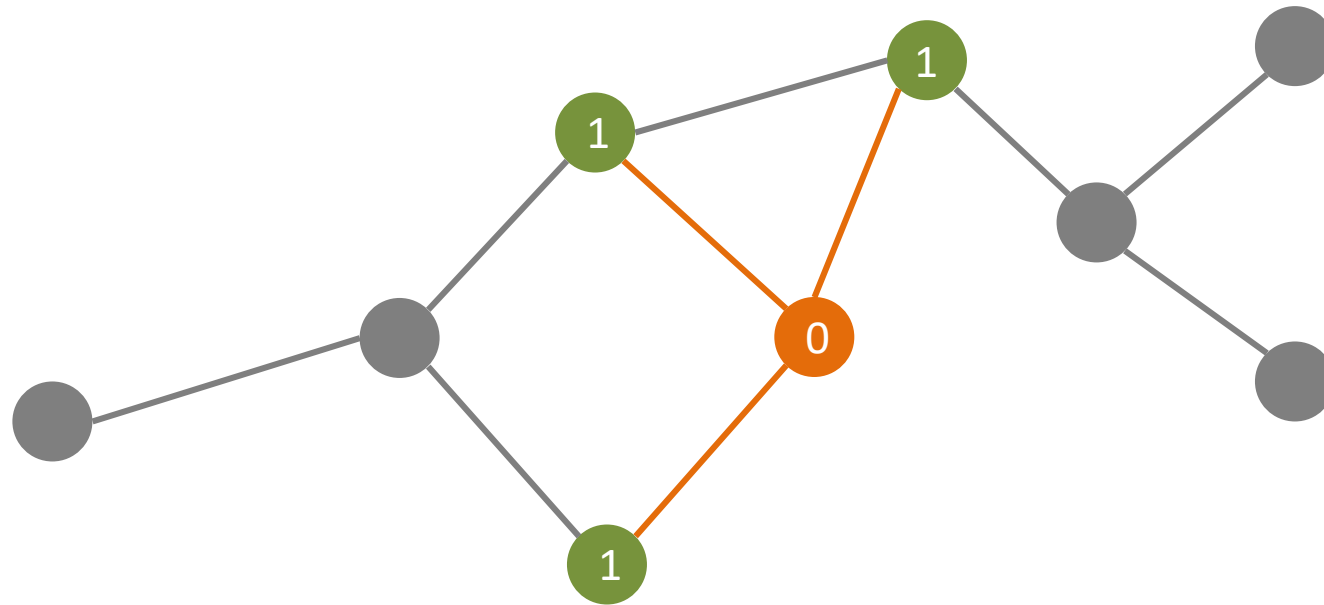
# Breadth-first search



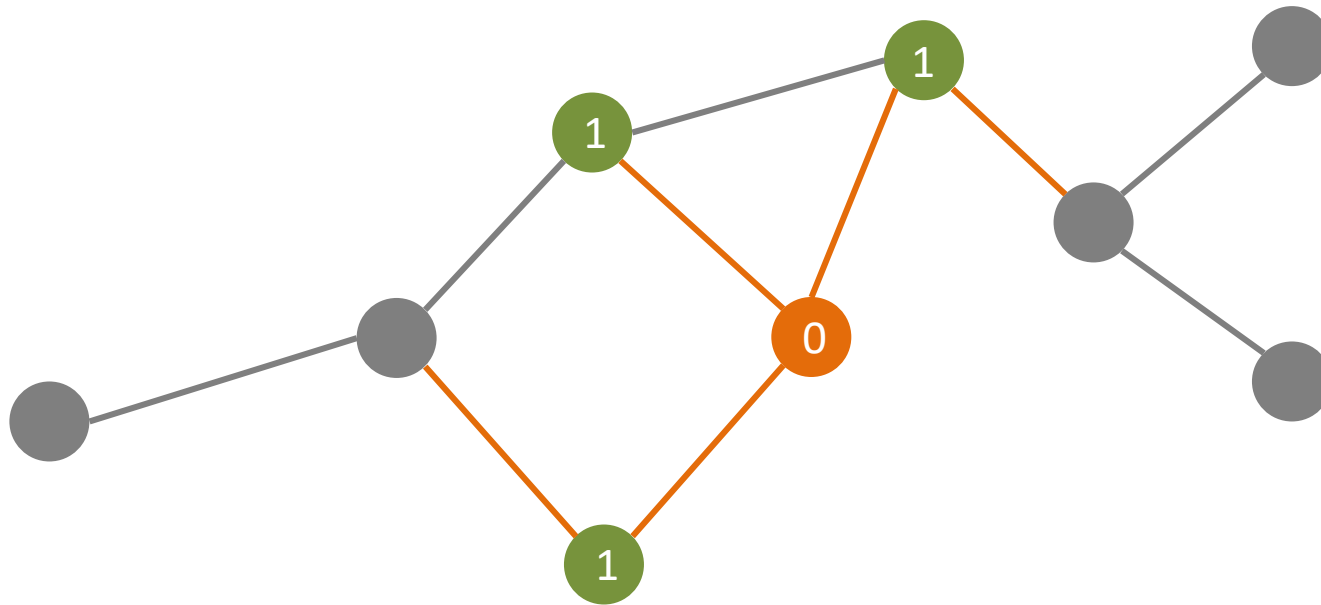
# Breadth-first search



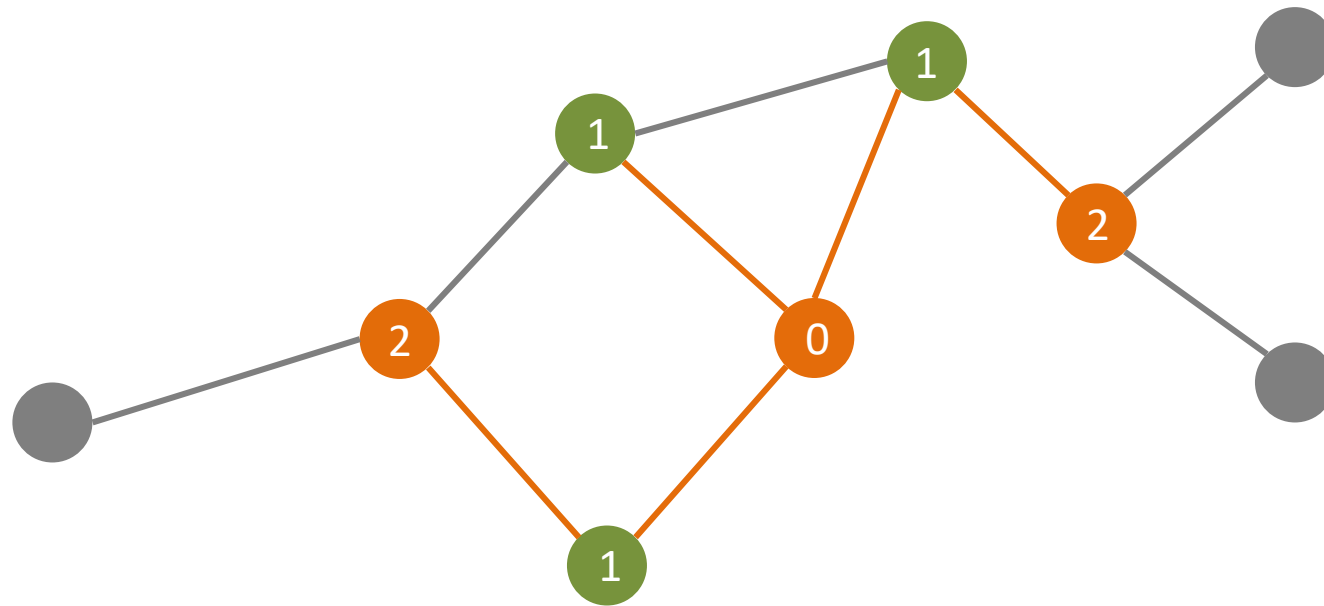
# Breadth-first search



# Breadth-first search

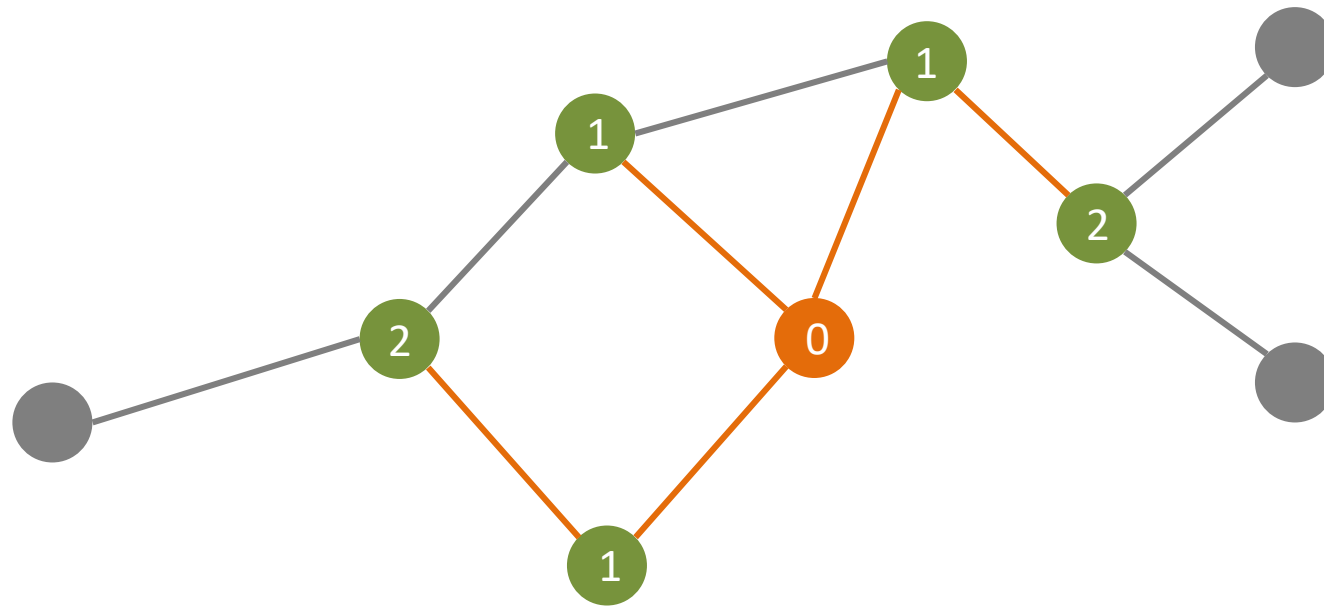


# Breadth-first search

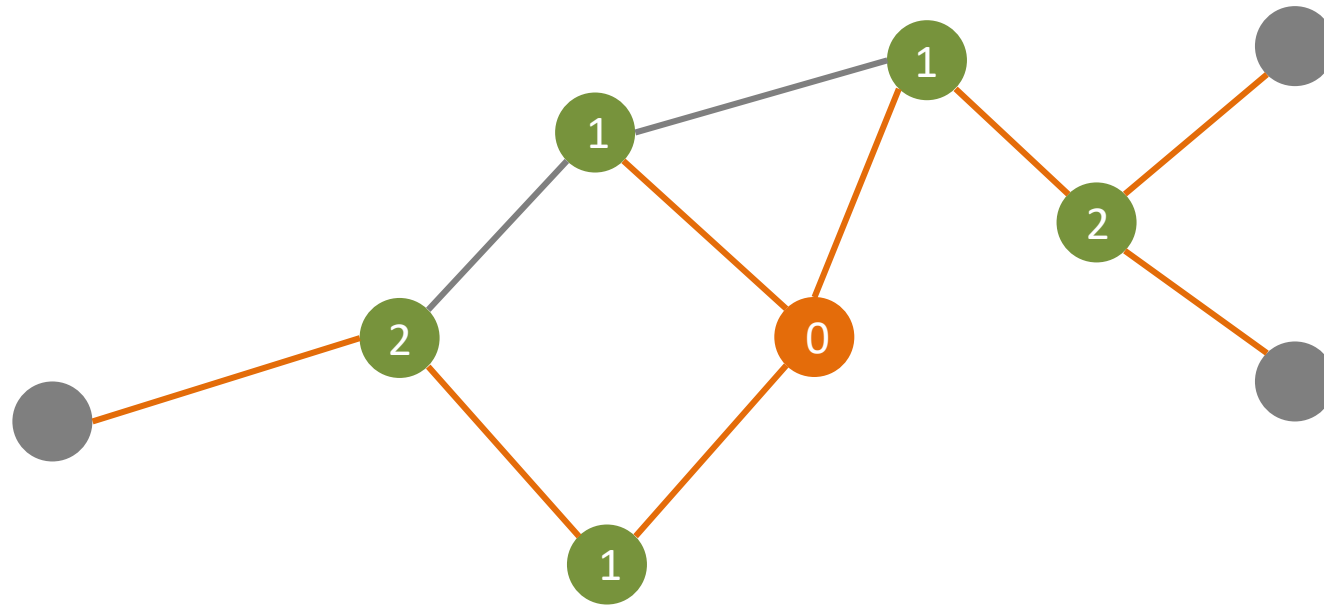




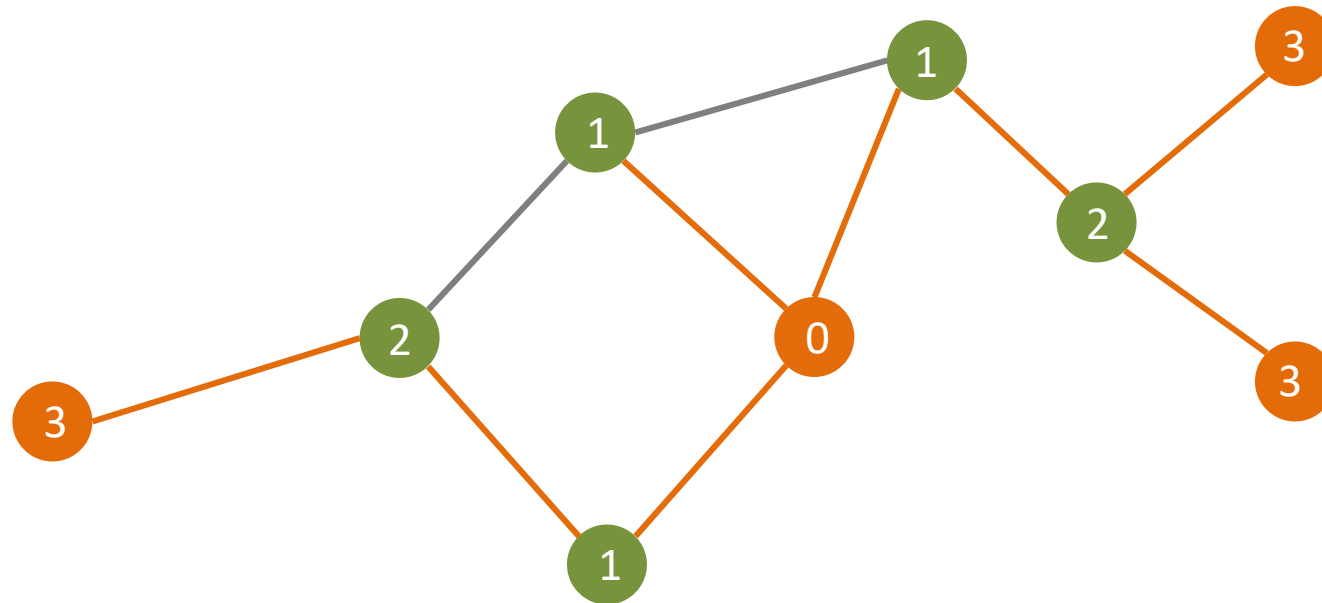
# Breadth-first search



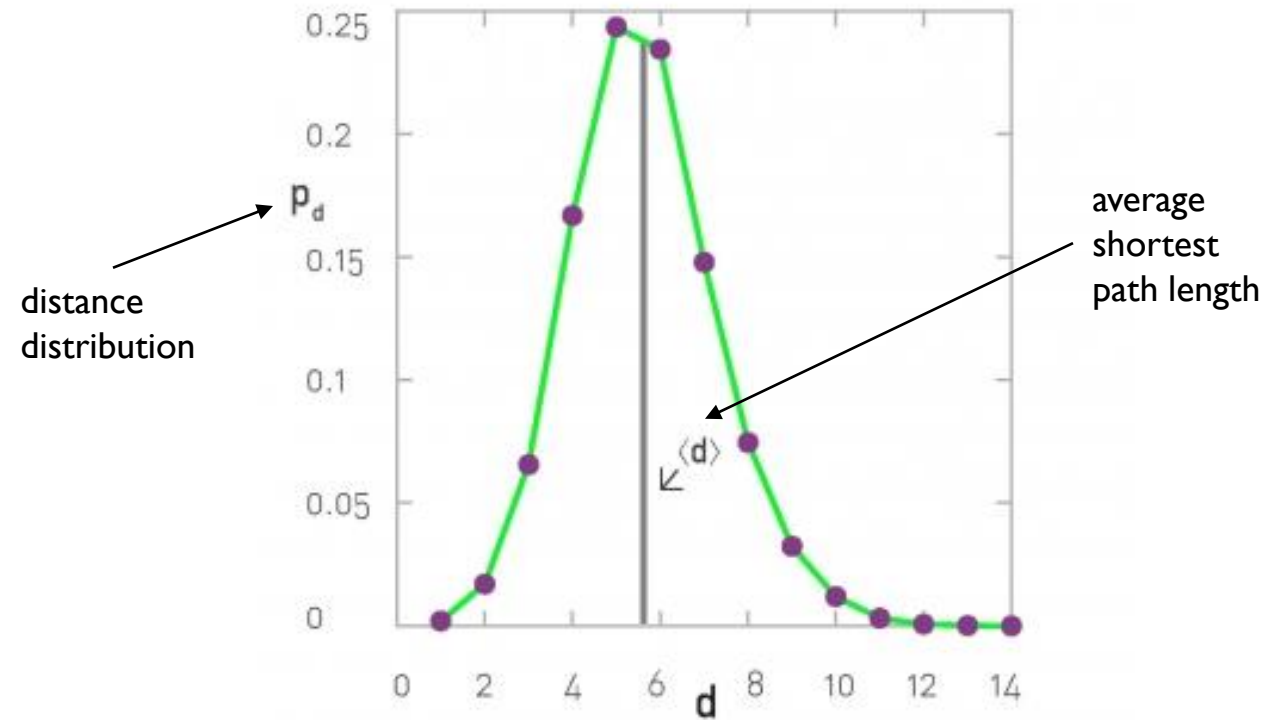
# Breadth-first search



# Breadth-first search



# Protein-protein interaction network of yeast

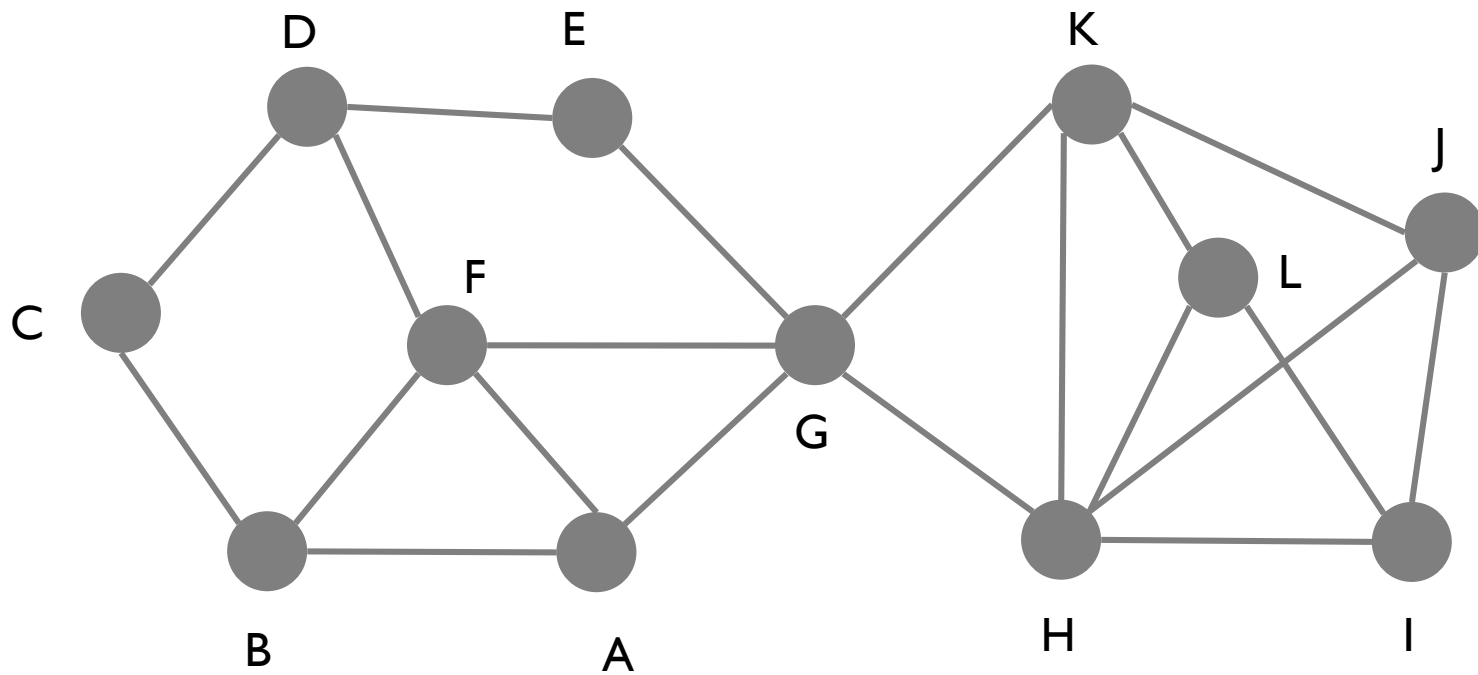


# Shortest path

- Social network
  - finding friends-of-friends
  - six degree of separation
- `nx.shortest_path`
  - computes shortest paths given the whole graph, source node and target node

# Eccentricity

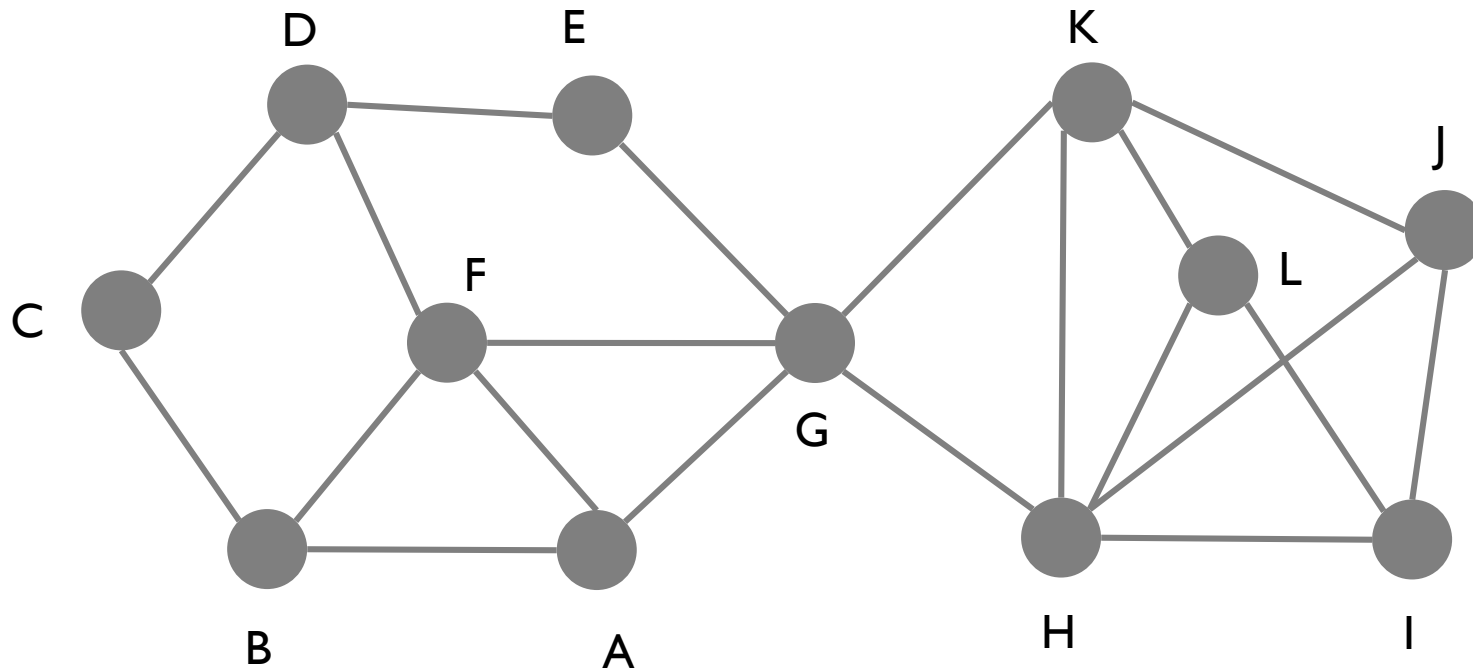
- Maximum distance from a vertex to all other vertices
- Measure of how far from the center a vertex is
- `nx.eccentricity(G,node)`



Eccentricity of node H?

H to I	1
H to J	1
H to K	1
H to L	1
H to G	1
H to A	2
H to B	3
H to C	4
H to D	3
H to E	2
H to F	2

# Diameter



$$d_{max} = ?$$

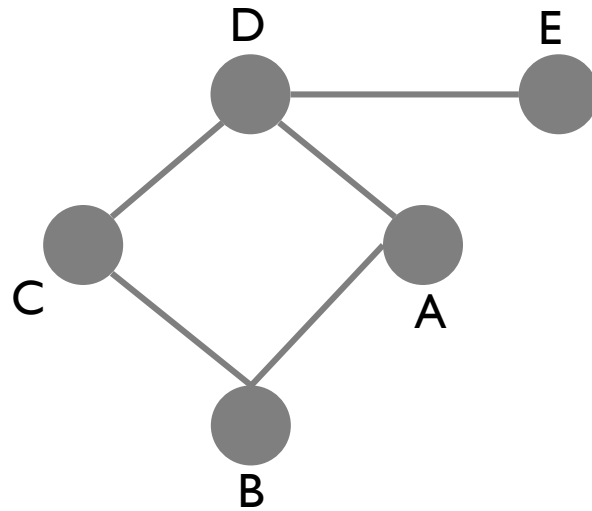
$$d_{max} = 5$$



# Diameter

- Distance from one end of the network to another
  - maximum shortest path
- Gives a sense of the network's overall size
- `nx.diameter(G)`
  - works only if the graph  $G$  is connected

# Average path length



$$\bar{d} = \frac{d_{AB} + d_{AC} + d_{AD} + d_{AE} + d_{BC} + d_{BD} + d_{BE} + d_{CD} + d_{CE} + d_{DE}}{10}$$

$$\bar{d} = \frac{1 + 2 + 1 + 2 + 1 + 2 + 3 + 1 + 2 + 1}{10} = 1.6$$

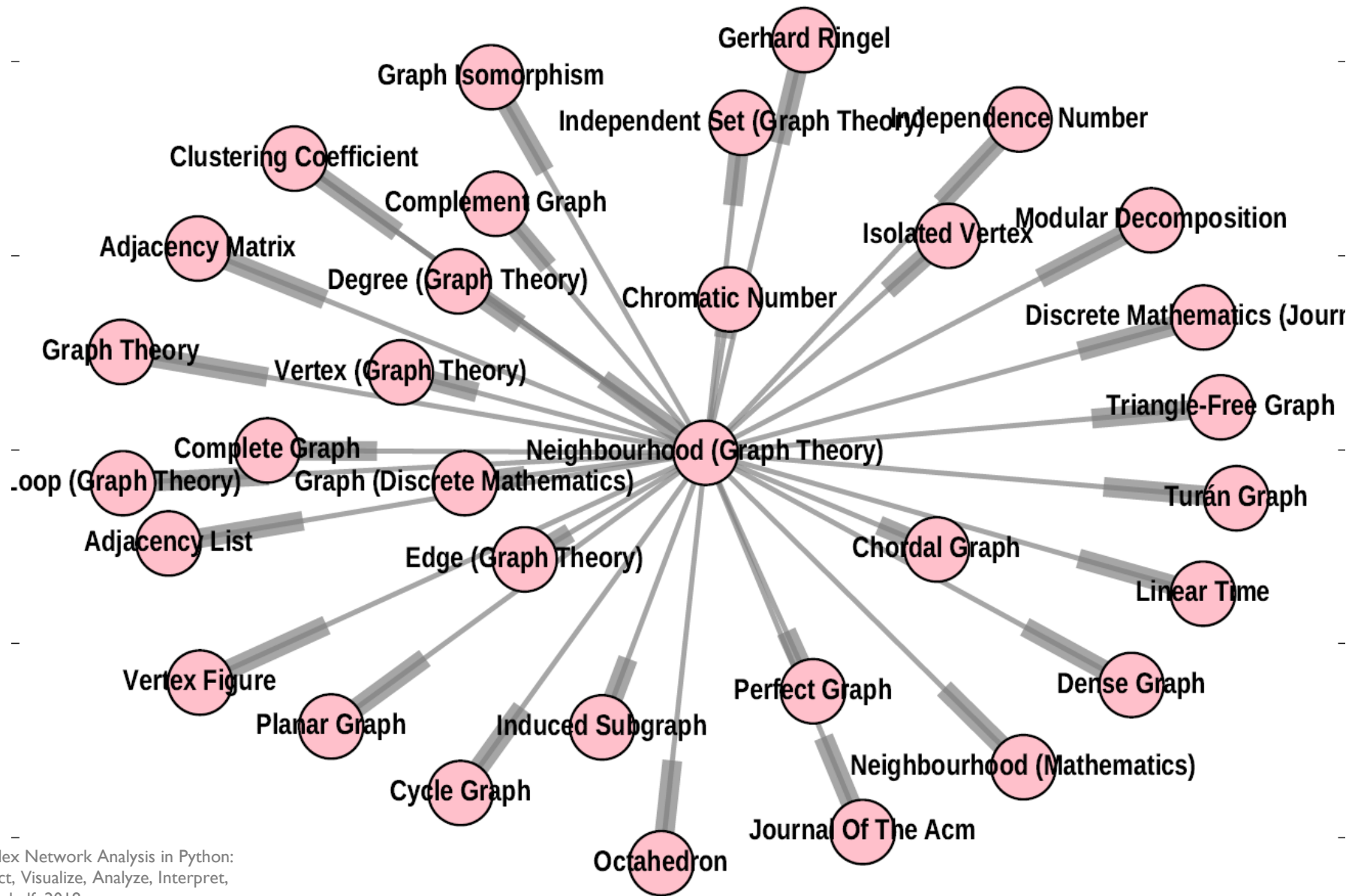
$\bar{d} = ?$

# Other distance measures

- Radius of a network
  - it is not a half of the diameter!
  - `nx.radius(G)` The radius  $\{ \displaystyle r \}r$  of a graph is the minimum eccentricity of any vertex or, in symbols,
- Center of a network
  - all nodes whose eccentricity equals the radius
  - `nx.center(G)`
- Periphery of a network
  - all nodes whose eccentricity equals the diameter
  - `nx.periphery(G)`

# Connected components

- *Component* is a set of vertices reachable from one vertex (in undirected graph)
- If every pair of nodes is connected by some path, then the network is *connected*



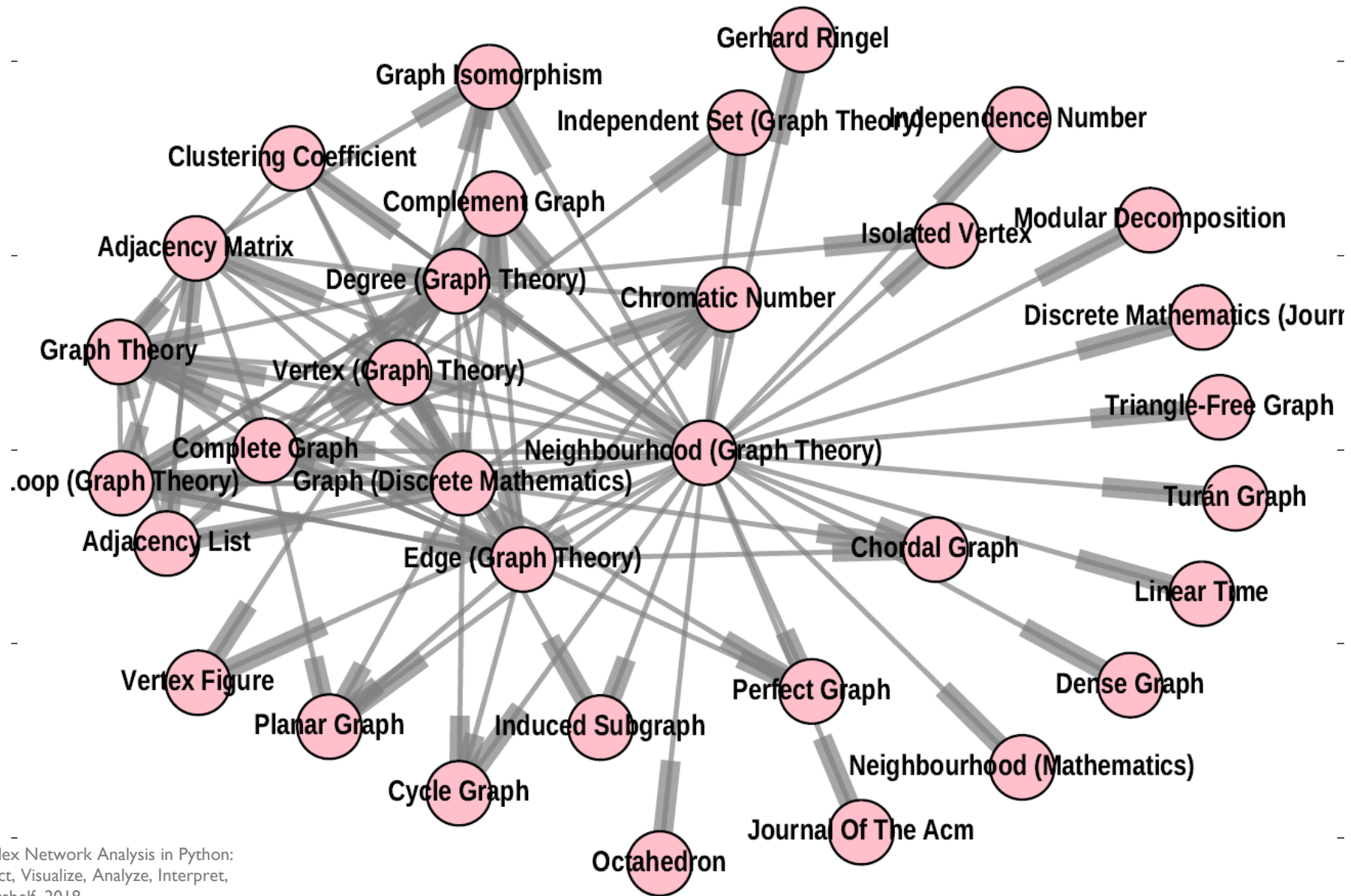
Zinoviev, D.. Complex Network Analysis in Python: Recognize, Construct, Visualize, Analyze, Interpret, The Pragmatic Bookshelf, 2018.

# Neighbourhood

- Neighbourhood of a node is a set of all nodes adjacent to that node
- Localised properties of network graphs
- Usually convey little information

# Neighbourhood

- Social network analysis pays special attention to neighbourhoods
  - relatives
  - close friends
  - colleagues
  - most significant *alters* of the *ego*



Zinoviev, D.. Complex Network Analysis in Python: Recognize, Construct, Visualize, Analyze, Interpret, The Pragmatic Bookshelf, 2018.

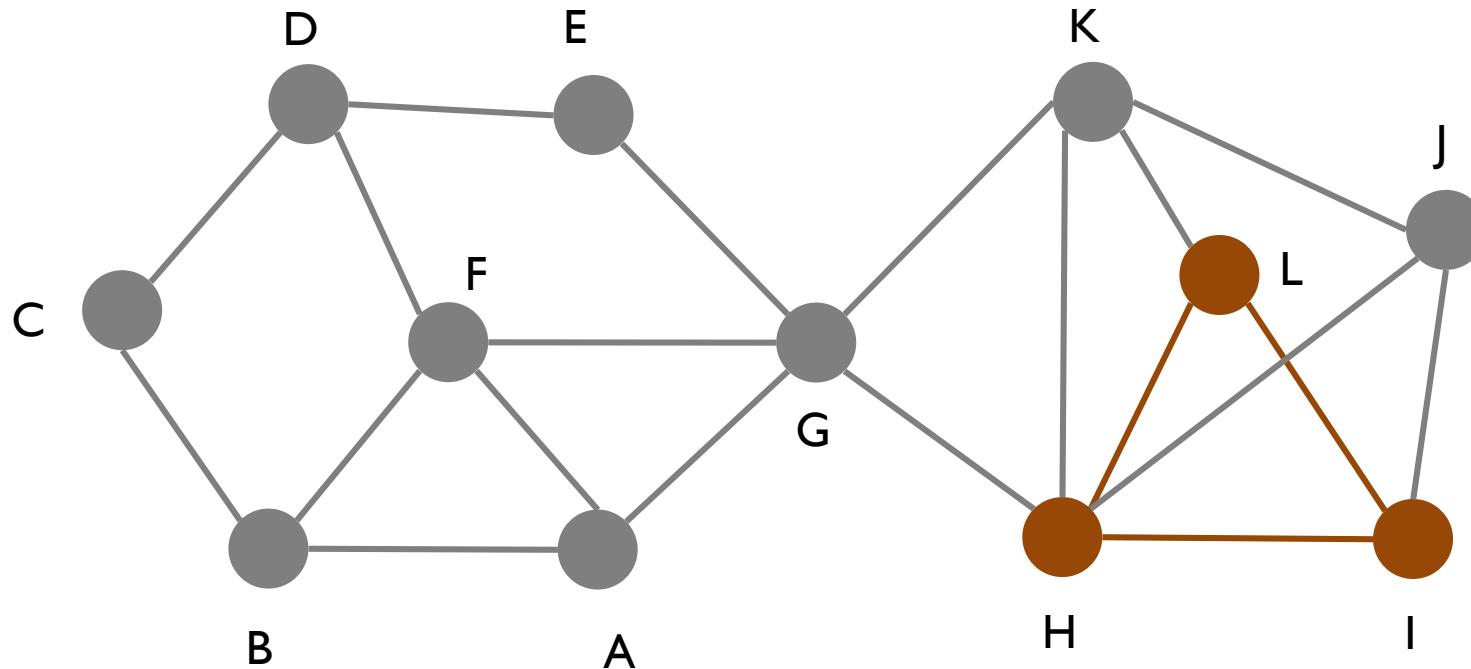


# Egocentric network

- *Ego* is the central node
- *Alters* are all other nodes
- Should be much denser than a neighbourhood
  - some nodes may be connected only to the hub
  - others may form triangles that involve more neighbourhood members
- `nx.ego_graph(G, node)`

Triadic closure is the property among three nodes A, B, and C, such that if a strong tie exists between A-B and A-C, there is a weak or strong tie between B-C.[2] This property is too extreme to hold true across very large, complex networks, but it is a useful simplification of reality that can be used to understand and predict networks.[3]

# Triadic closure

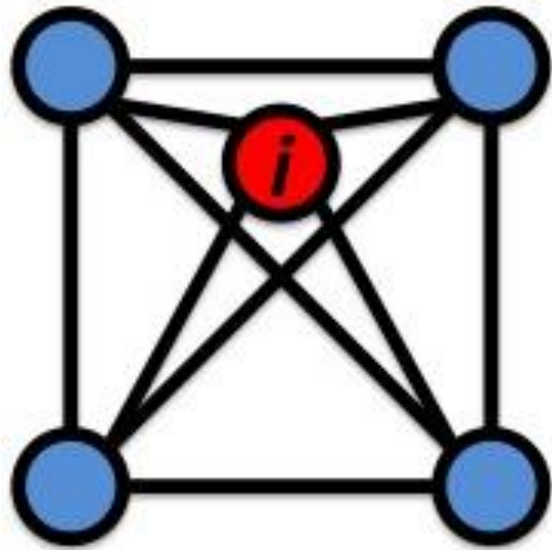


*If X is connected to Y and Y to Z, then X is connected to Z*

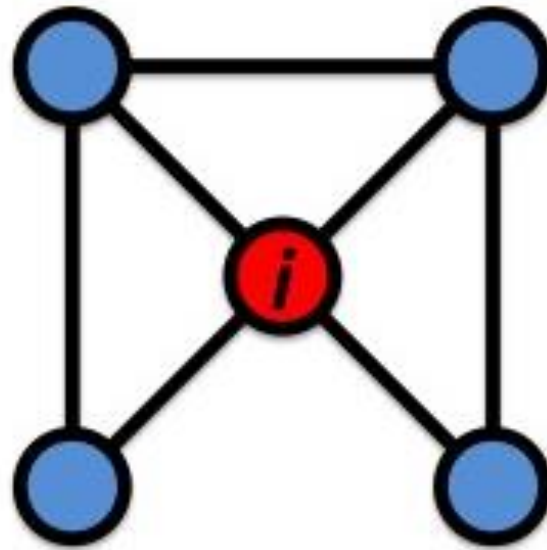
# Triadic closure

- Social network
  - if two people know the same person, they are likely to know each other
  - triads or triangles essential units of social network analysis
  - number of triads can be used to find clusters and communities of individuals that all know each other

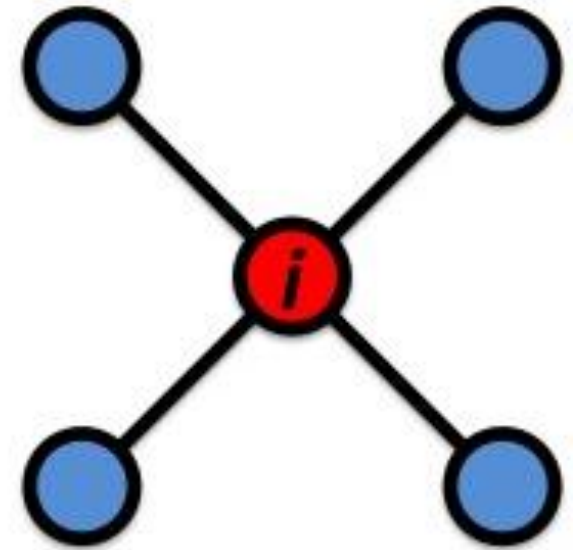
# Clustering coefficient



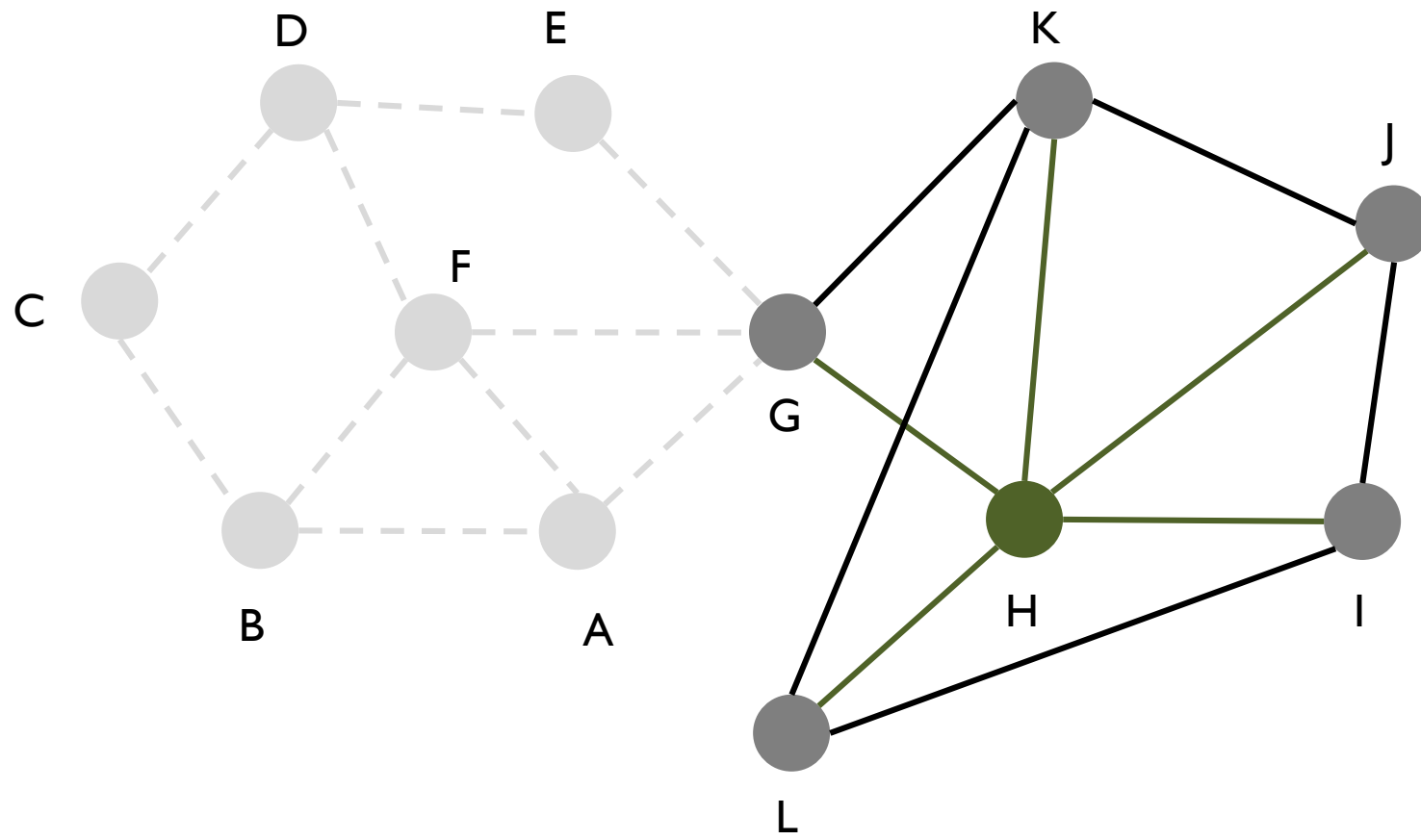
$$C_i = 1$$



$$C_i = 1/2$$



$$C_i = 0$$

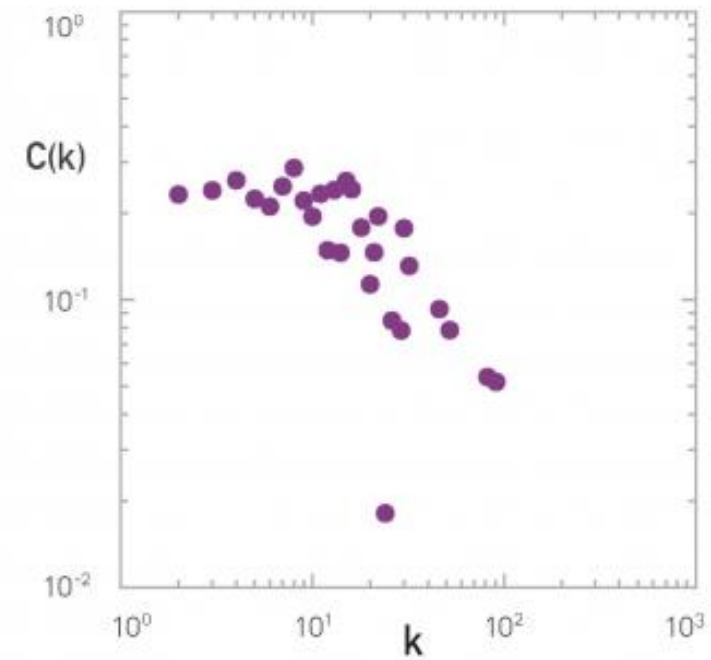


$$C_H = ?$$

# Protein-protein interaction network of yeast



average of the  
clustering coefficient  
of all nodes with the  
same degree  $k$



Dependence of the clustering  
coefficient on the node's degree

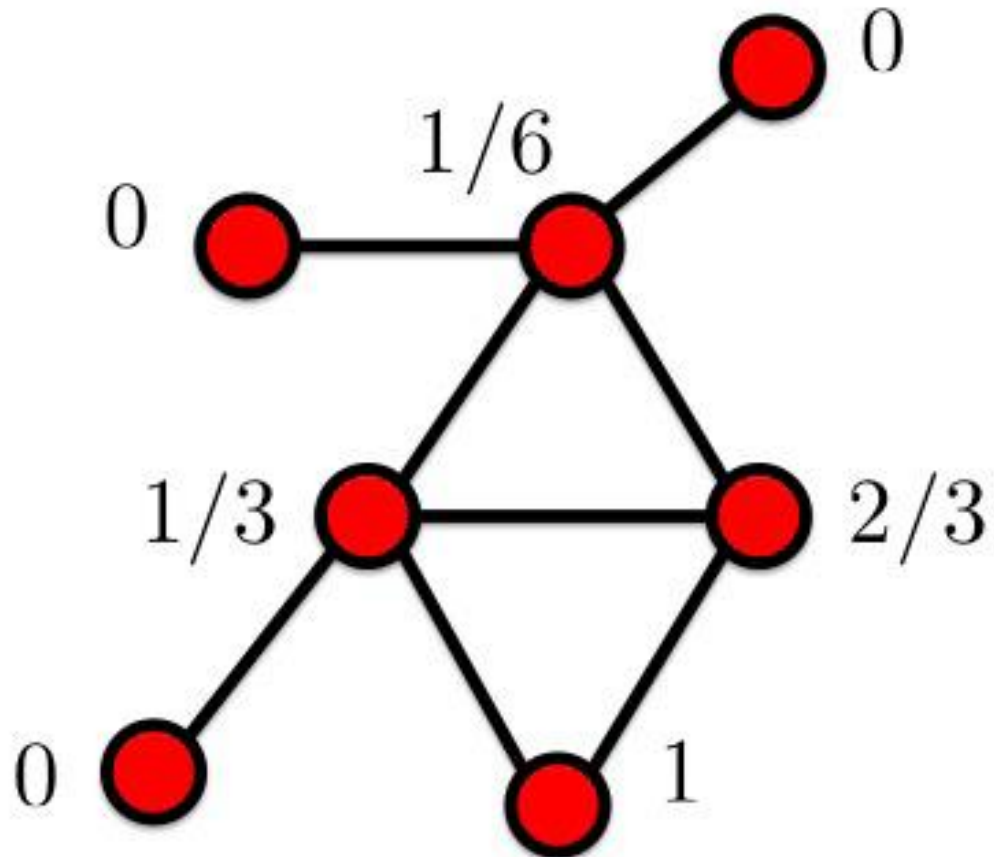
# Clustering coefficient

- Fraction of all possible triangles that contain a given node and exist
- Measure of the prevalence of triangles in a network
- Measure of “stardom”
- `nx.clustering(G, node)`

$C_i = \text{number of pairs of neighbours of } i \text{ that are connected} / \text{number of pairs of neighbours of } i$

$\bar{C} = ?$

$C = ?$



# Transitivity

$$\bar{C} = \frac{13}{42} \approx 0.310$$

$$C = \frac{3}{8} = 0.375$$

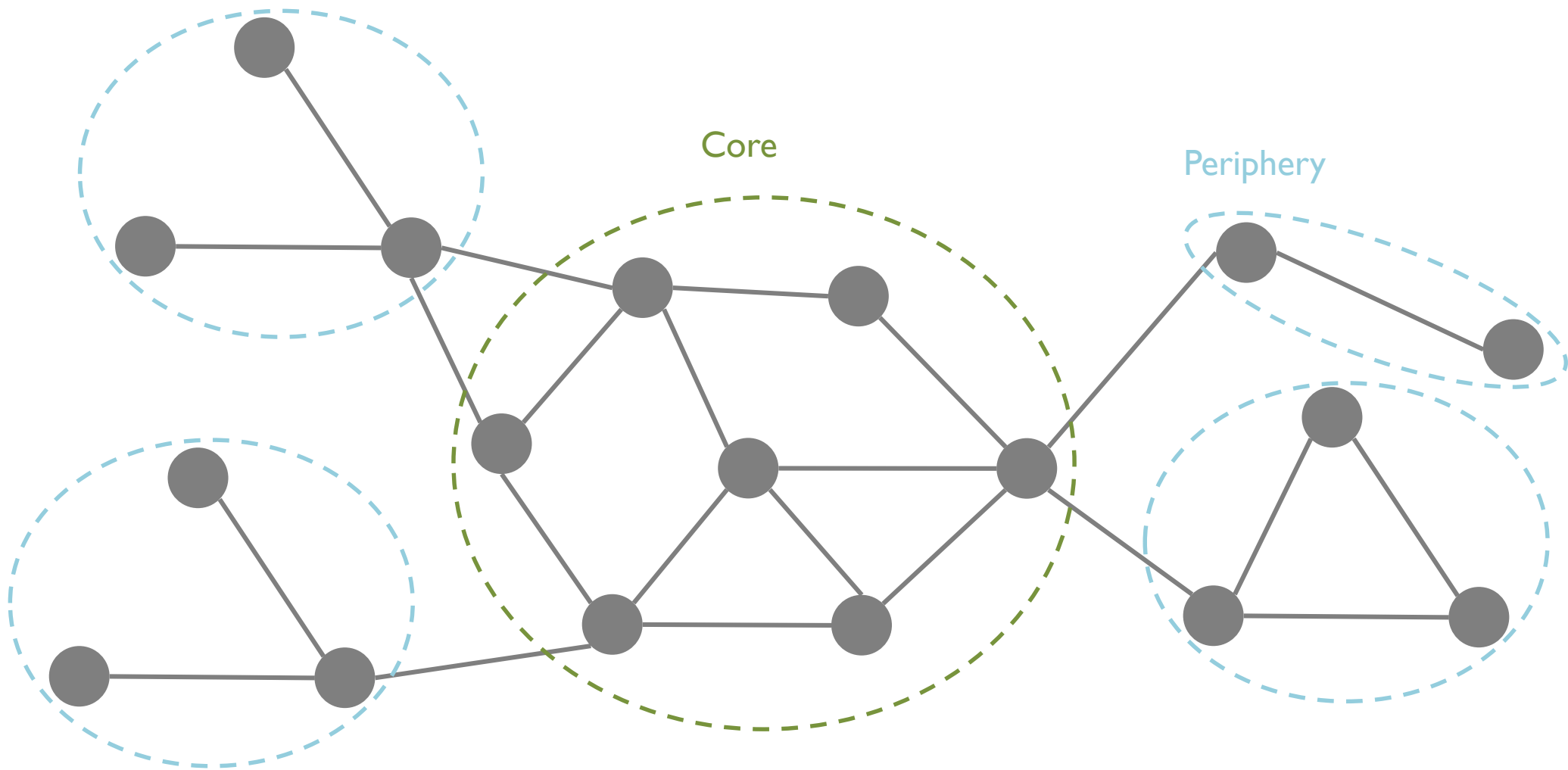


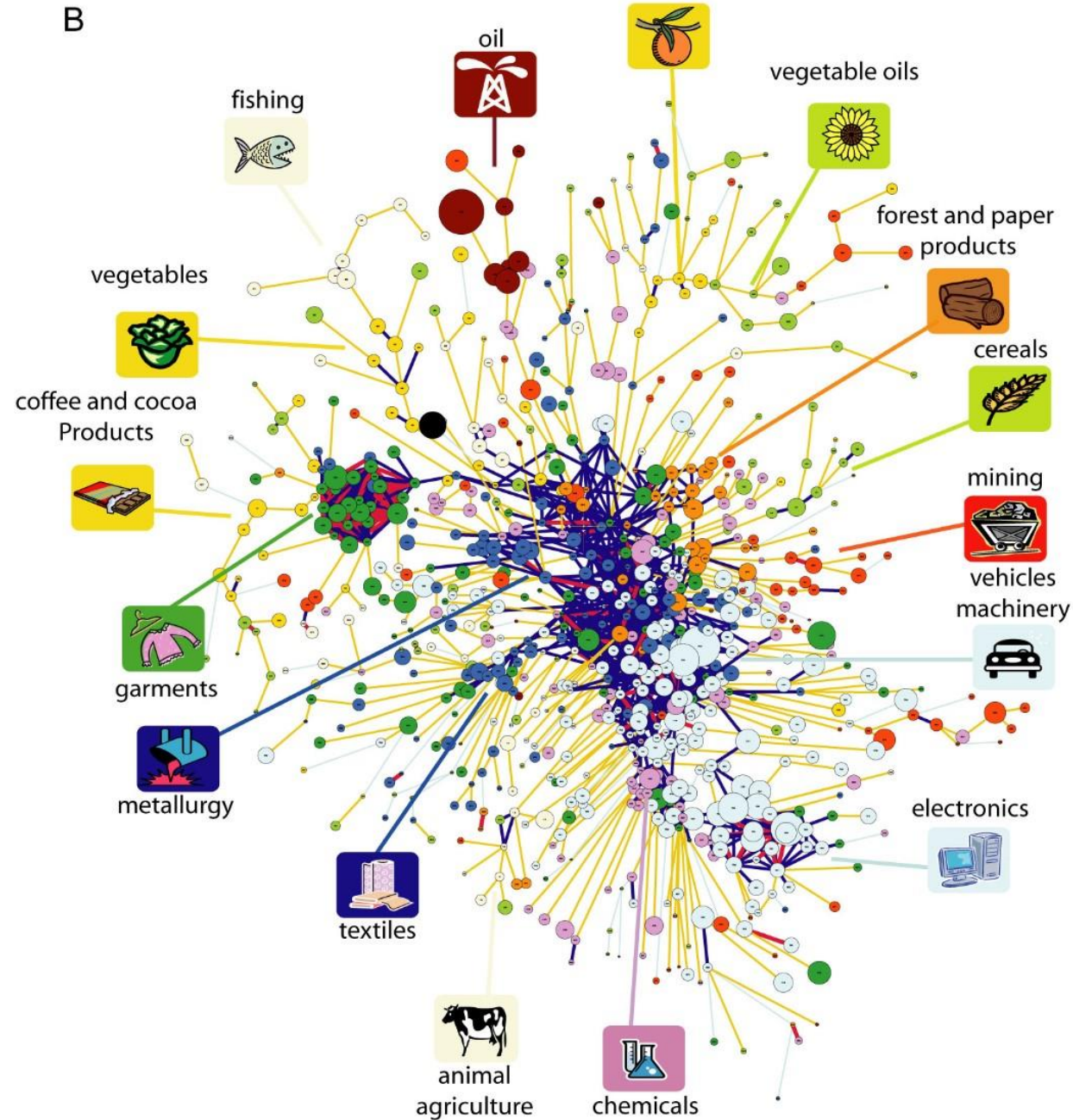
# Transitivity

- Ratio of all triangles to all, closed and “open”, triangles
- Expresses how interconnected a graph is in terms of a ratio of actual over possible connections
- `nx.transitivity(G)`

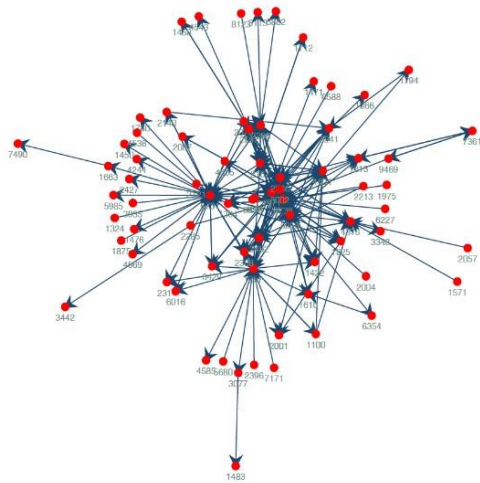
# Core-peripheral analysis

- Network consists of two sets of nodes
  - Core
    - nodes that are more or less tightly interconnected
  - Periphery
    - nodes that are tightly connected to the core, but weakly or not connected at all to the other peripheral nodes
- “Hairy” appearance

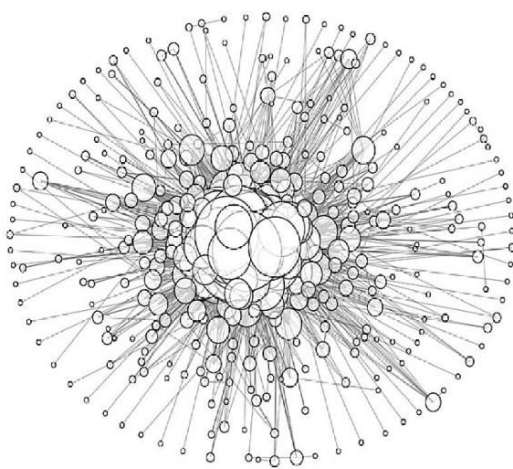




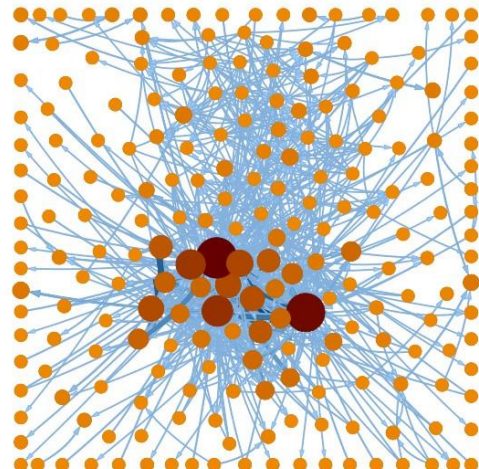
The core is formed by metal products, machinery and chemicals while the periphery is formed by the rest of the product classes.



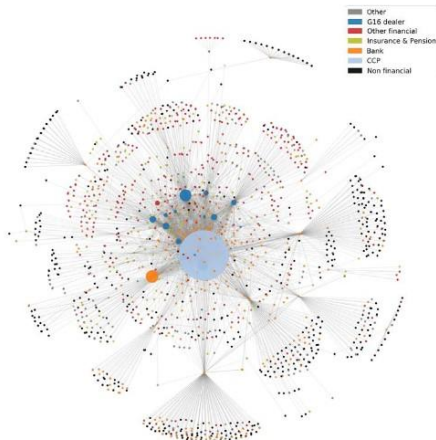
Li and Schürhoff - muni bonds



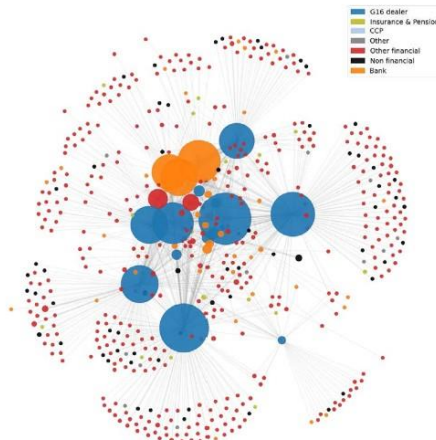
Hollifield, Neklyudov, Spatt - ABS



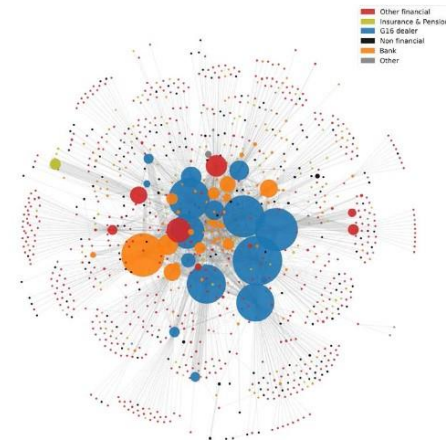
Bech and Atalay - Fed funds



ESRB - Interest rate swaps



ESRB - Credit default swaps

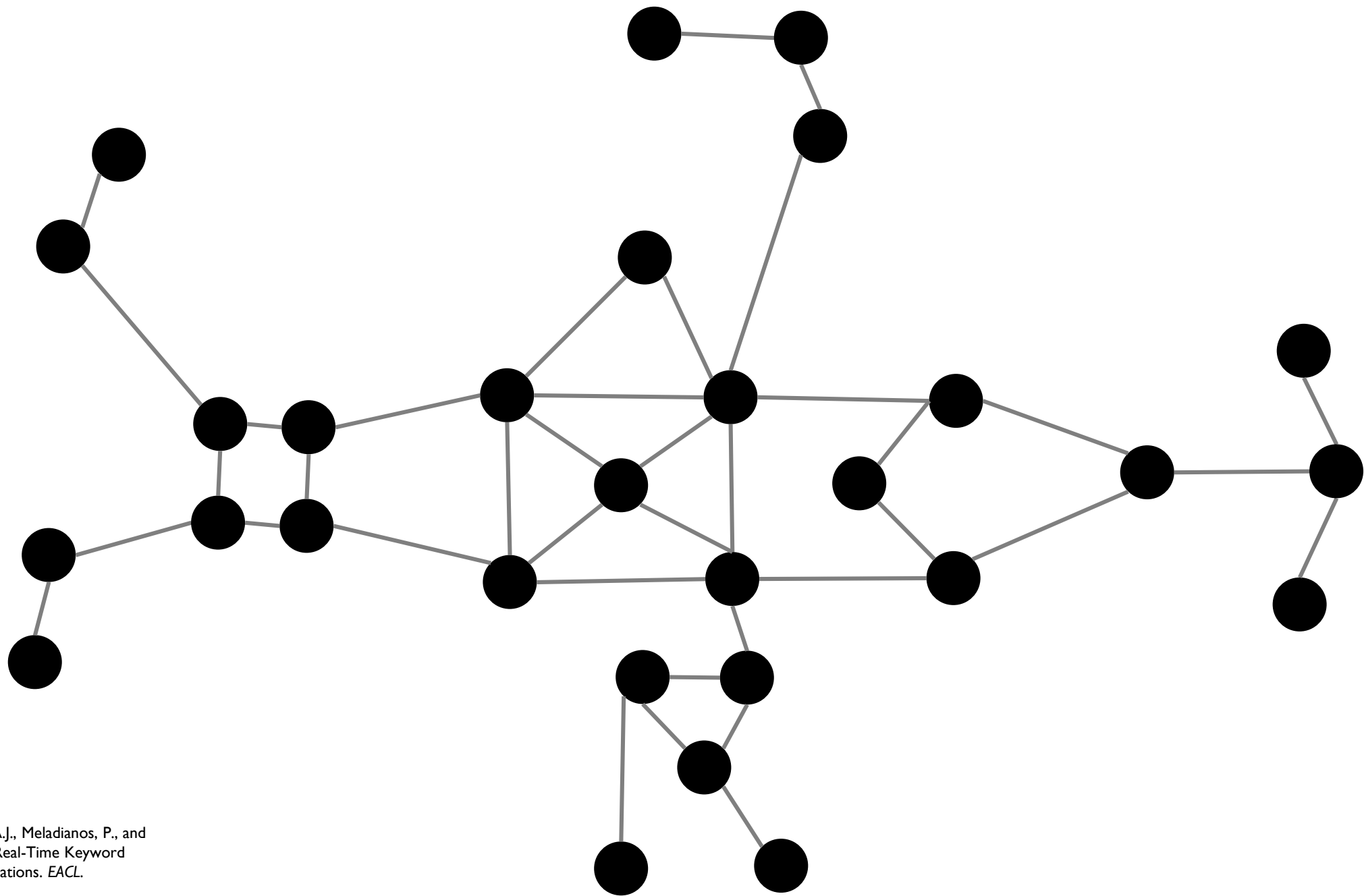


ESRB - FX forwards

Figure 1 – Core-periphery networks in OTC markets

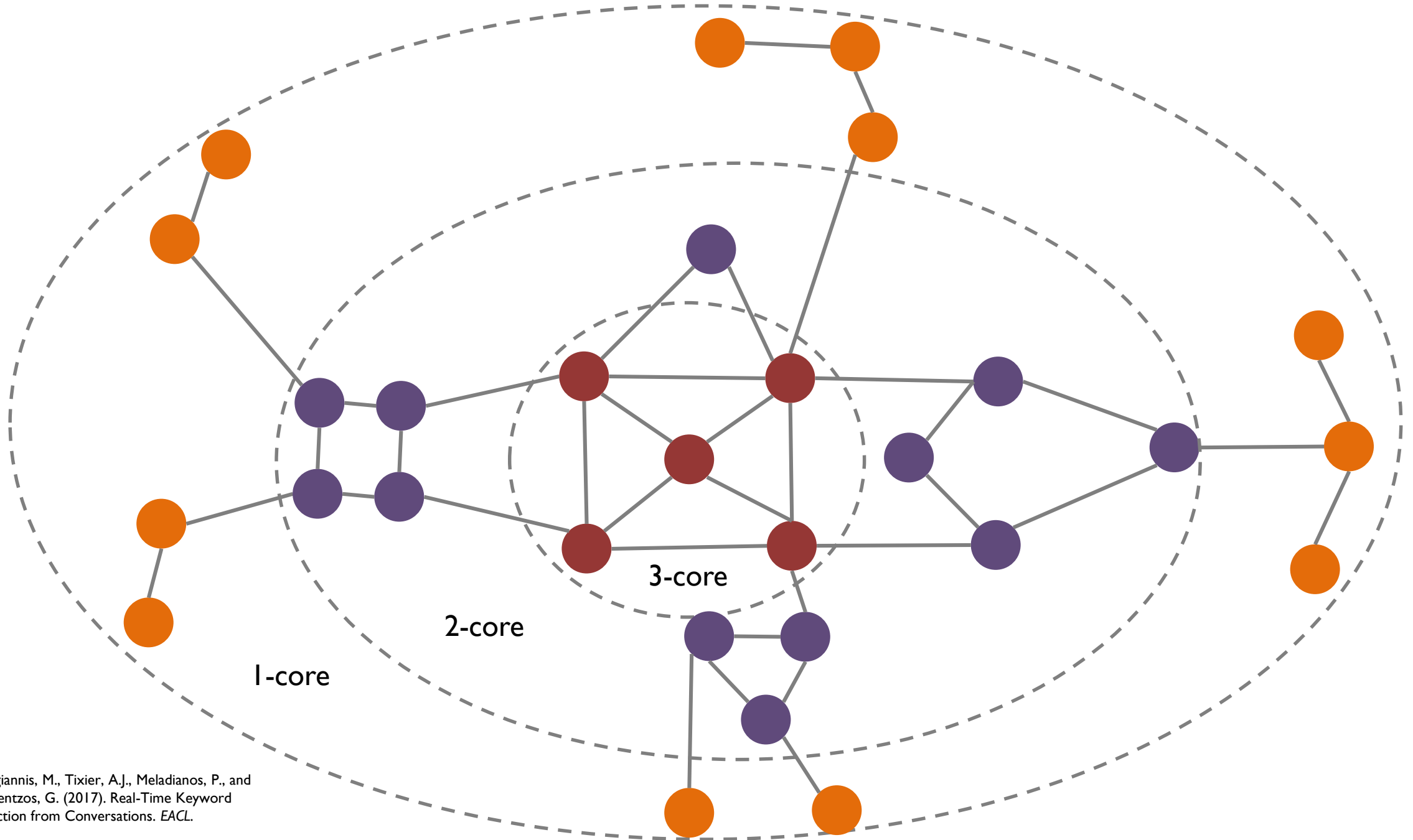
# $k$ -core

- $k$ -core ( $k > 0$ ) is a subgraph of the original network graph such that each node in the subgraph has at least  $k$ -neighbours
- Main core is the one with the largest possible  $k$
- Used to study the clustering structure of social networks, the evolution of random graphs, applied in bioinformatics, network visualisation, Internet structure, spreading of economic crises, etc.
- `nx.k_core(G)`



Vazirgiannis, M., Tixier, A.J., Meladianos, P., and Nikolentzos, G. (2017). Real-Time Keyword Extraction from Conversations. *EACL*.





Vazirgiannis, M., Tixier, A.J., Meladianos, P., and Nikolentzos, G. (2017). Real-Time Keyword Extraction from Conversations. *EACL*.



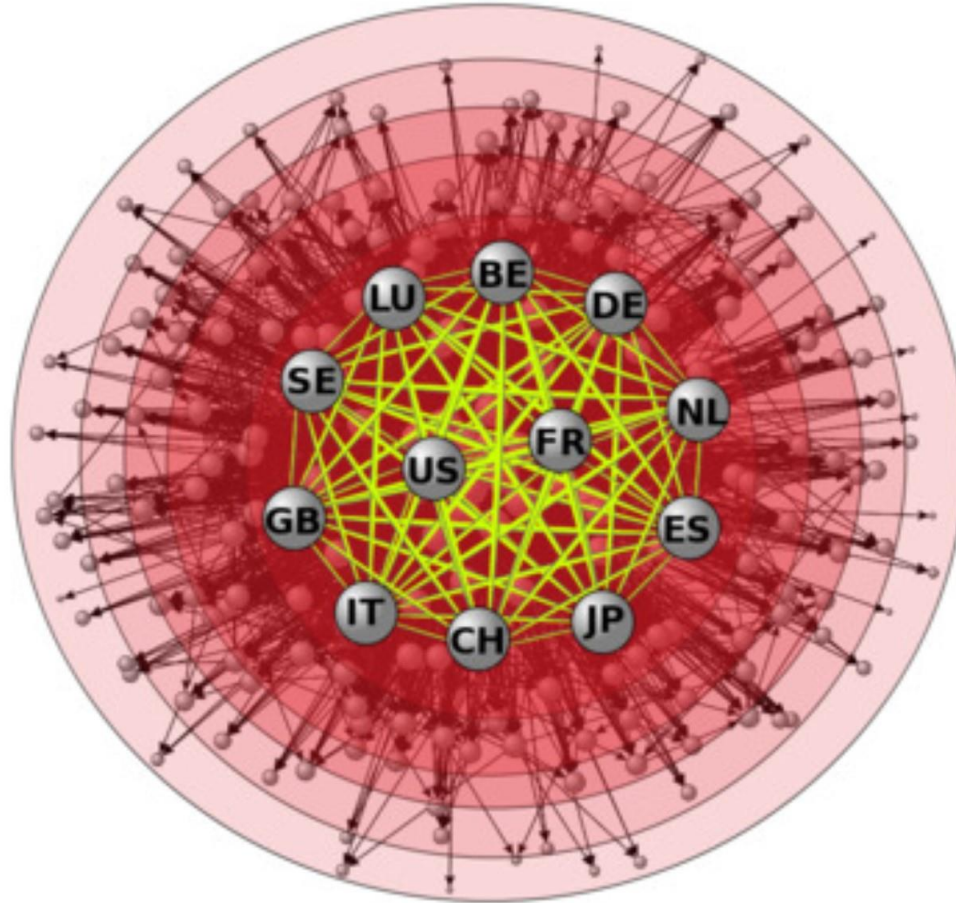


FIG. 1. An illustration of the layered structure of the global economic network of 206 countries of the world using the large corporation subsidiary relations.

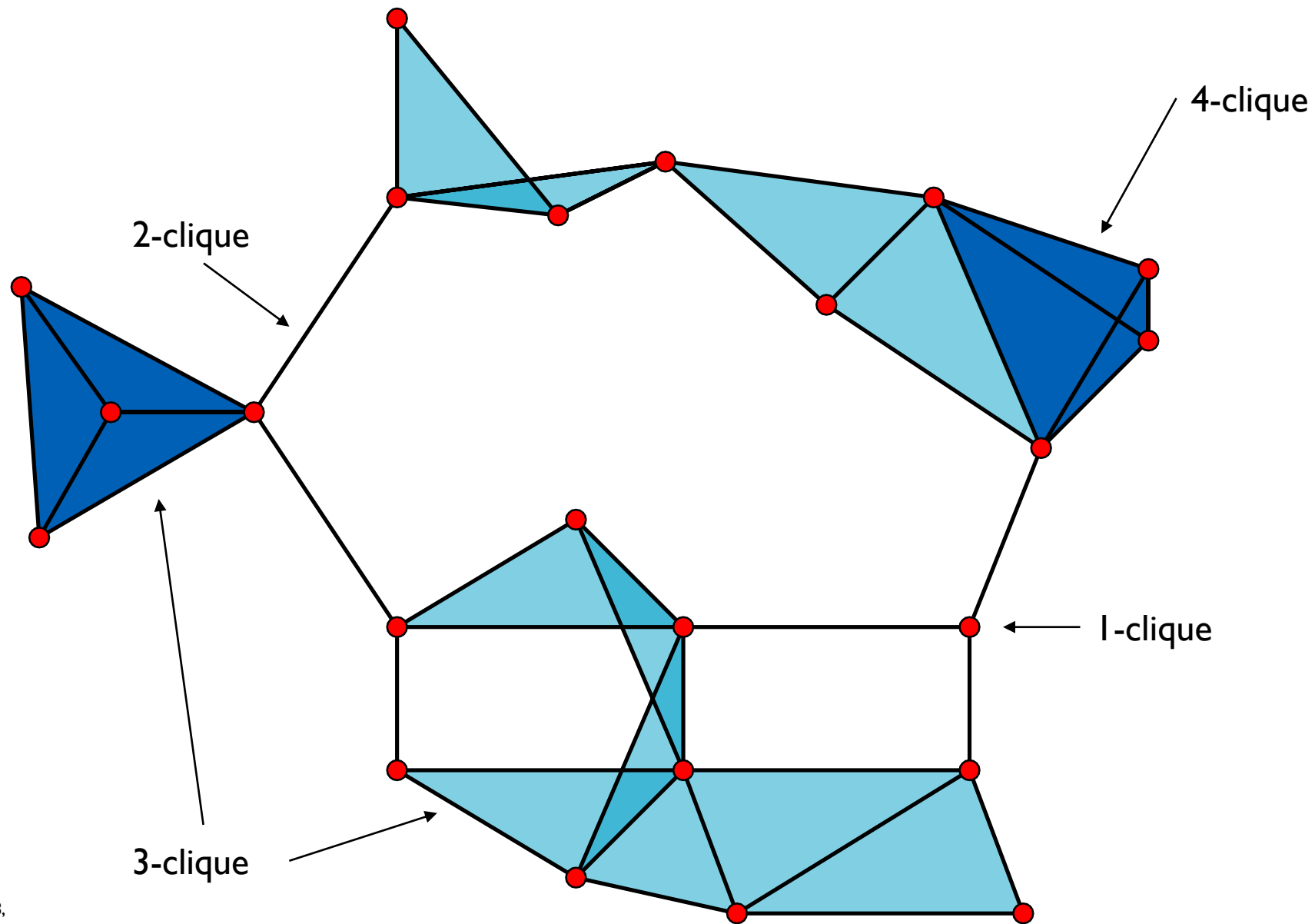
# $k$ -core

- $k$ -core construction process
  - remove all nodes from the original graph that have a degree smaller than  $k$  and all the incident nodes\*
  - remove nodes that have fewer than  $k$  neighbors
  - iterate until no remaining node has fewer than  $k$  neighbors
  - the remaining nodes form the  $k$ -core

\*A node is incident to an edge if it is start or end of the edge. Two nodes are adjacent if they are incident to the same edge.

# Cliques

- Zoom in in a search for smaller network building blocks
- $k$ -clique is a subset of a  $k$  nodes such that each node is directly connected to each other node in the clique
  - complete subgraphs
  - the degree of a node in a  $k$ -clique is at least  $k-1$
- `nx.find_cliques(G)`
- `nx.enumerate_all_cliques(G)`



Wikipedia, Clique (Graph Theory), November 28, 2018.

# Sources

- Zinoviev, D.. Complex Network Analysis in Python: Recognize, Construct, Visualize, Analyze, Interpret, The Pragmatic Bookshelf, 2018.
- Clauset, A. Network Analysis and Modeling, CSCI5352, Santa Fe Institute (2017), <http://tuvalu.santafe.edu/~aaronc/courses/5352/>
- Ladd, J., Otis, J., Warren, C. N., and Weingart, S. Exploring and Analyzing Network Data with Python, The Programming Historian 6 (2017), <https://programminghistorian.org/en/lessons/exploring-and-analyzing-network-data-with-python>.
- Barabási, A. Network Science, <http://networksciencebook.com>