LISBON SCHOOL OF ECONOMICS & MANAGEMENT
UNIVERSIDADE DE LISBOA

Carlos J. Costa

SNA (SOCIAL NETWORK ANALYSIS)

SNA

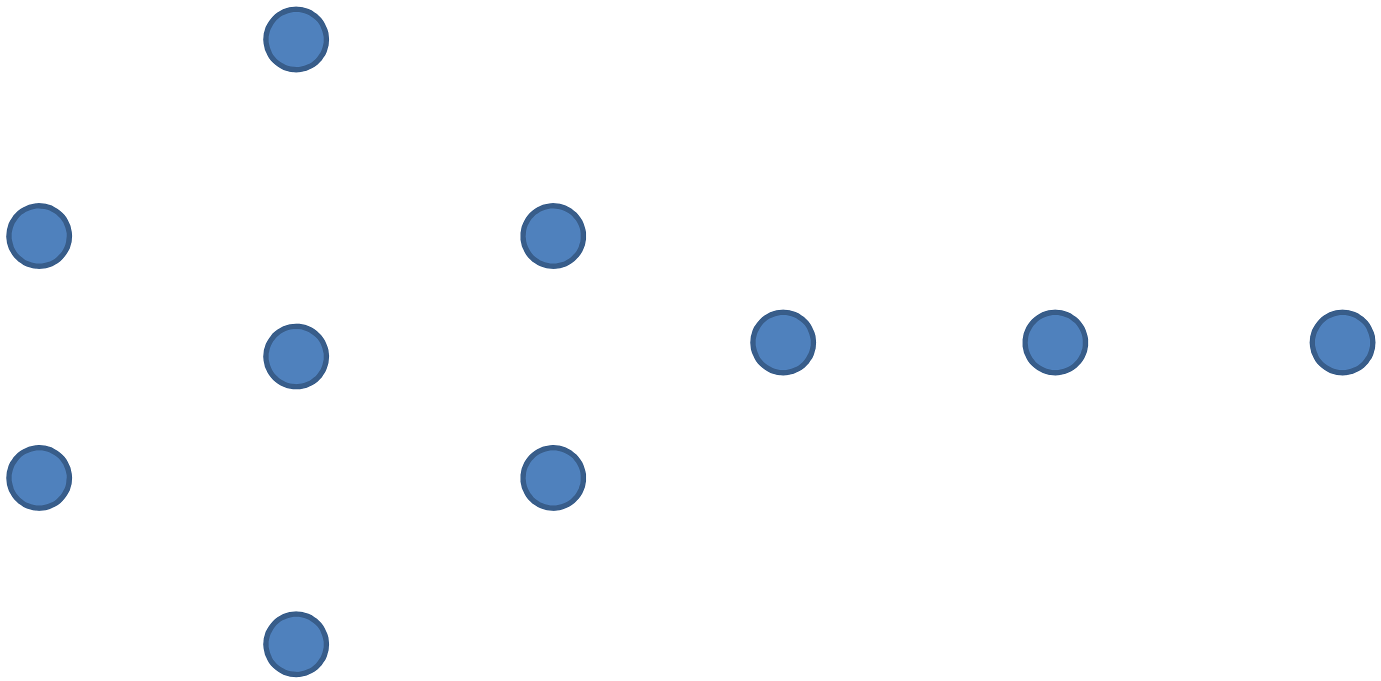
- Social network analysis (SNA) is the
  - mapping and measuring of
  - relationships and flows between
  - people, groups, organizations, computers, URLs, and other connected information/knowledge entities.

Network Elements

- "vertex" and "edge" (Mathematics)
- “nodes” and “connections” (or links) (Computer Science)
- “Actors” (or “agents”) and “relationships” (Sociology)
- “site” and “bond” (Physics)
- “Dot” and “arcs” (or ties)

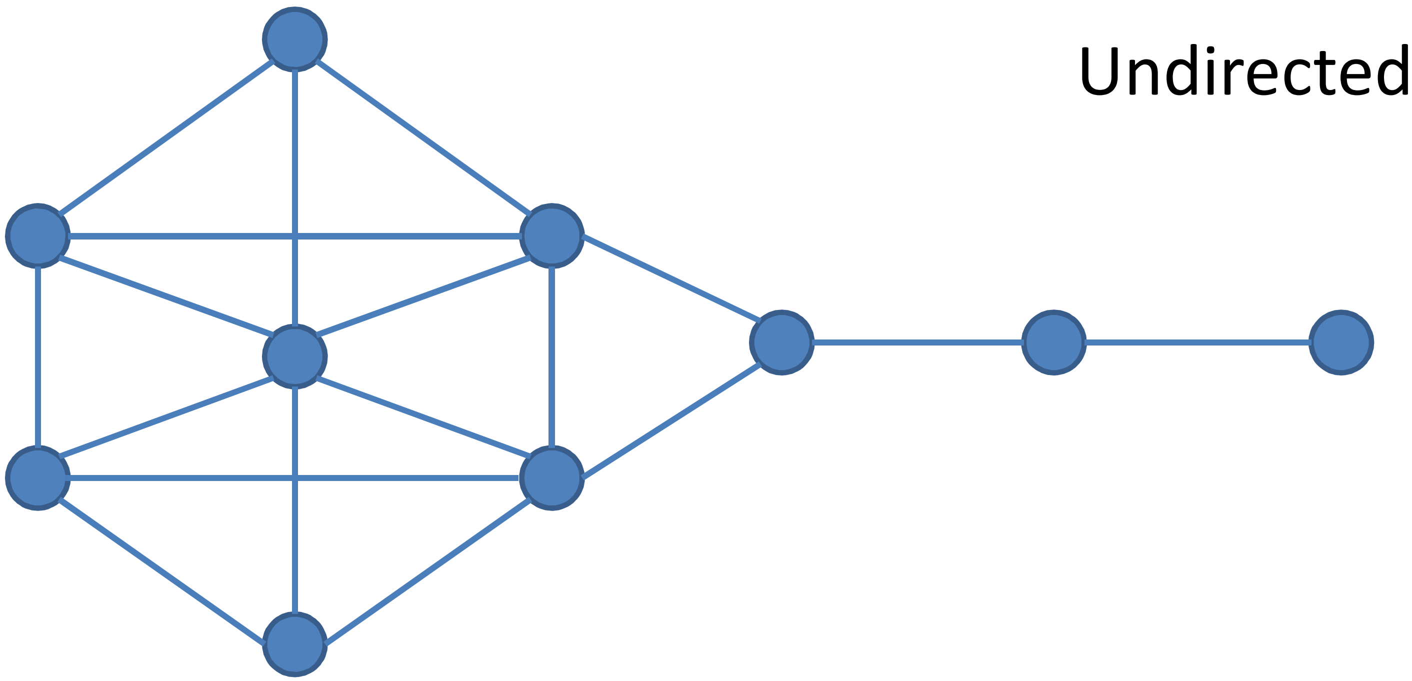
Network Elements

- Vertex, nodes and actors



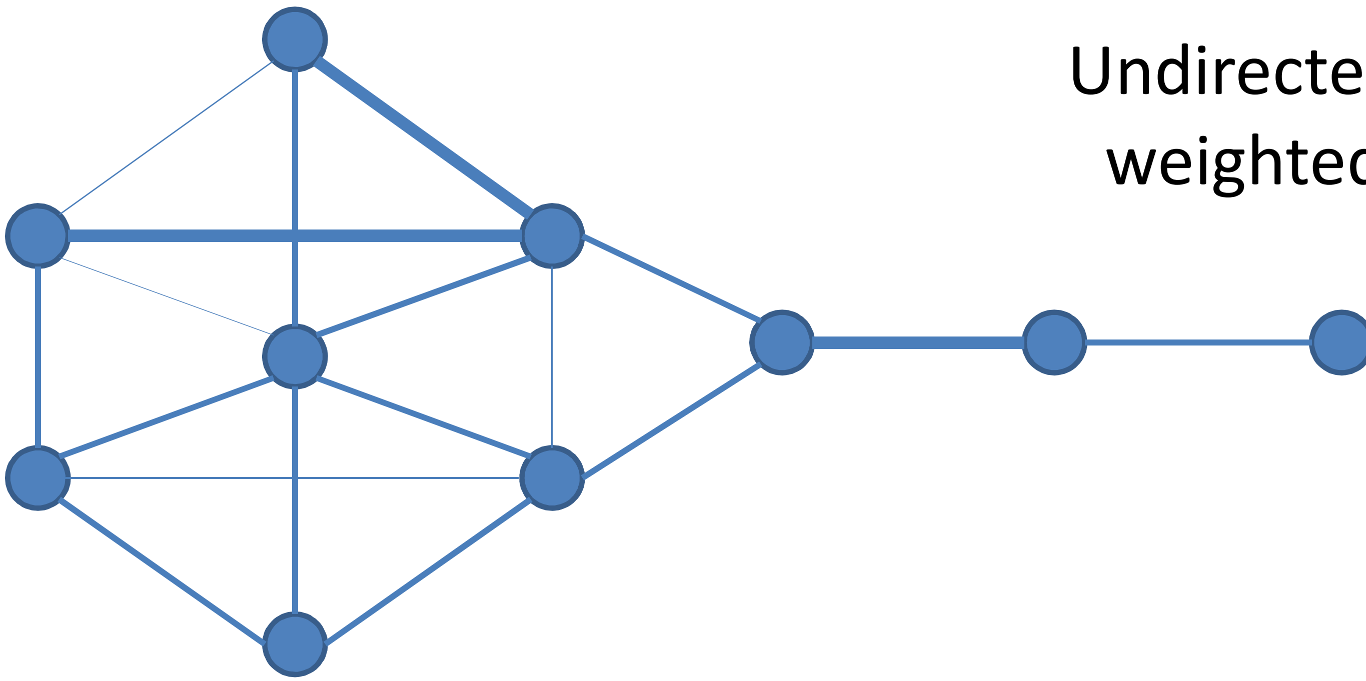
Network Elements

- Edges, arcs, links and relationships



Network Elements

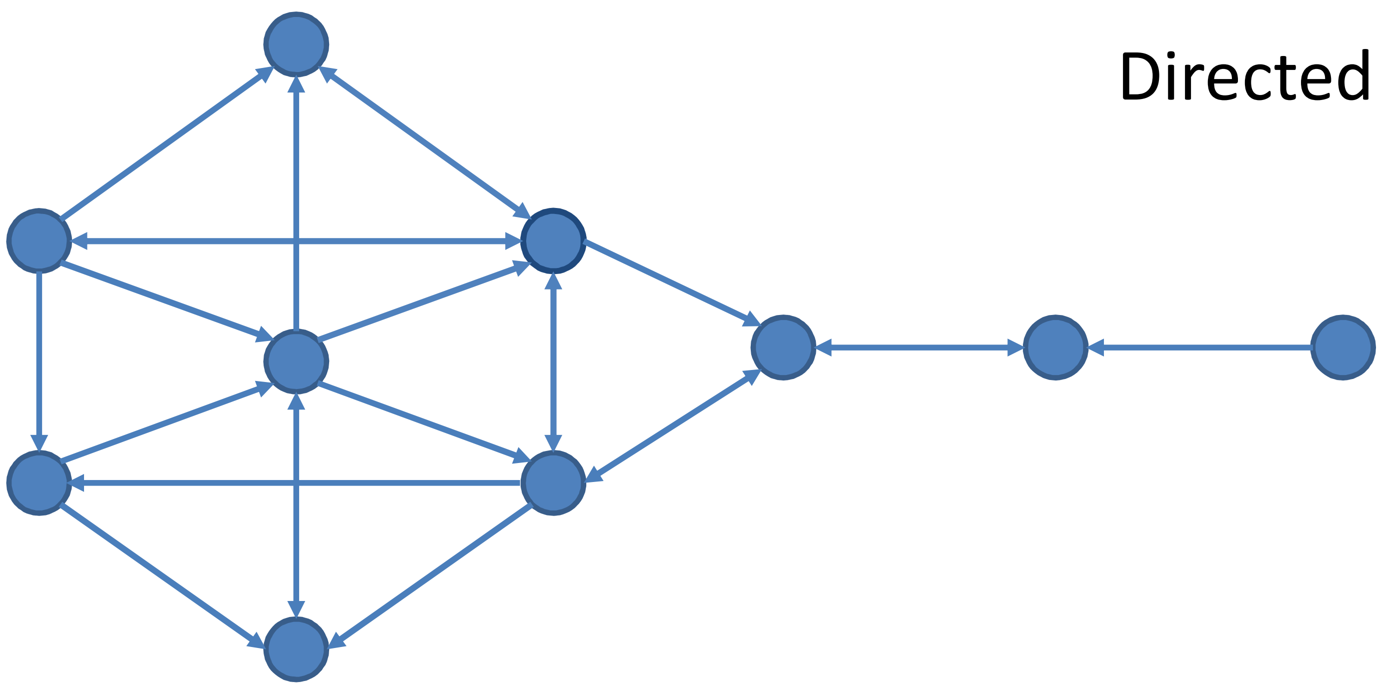
- Edges, arcs, links and relationships



Undirected, weighted

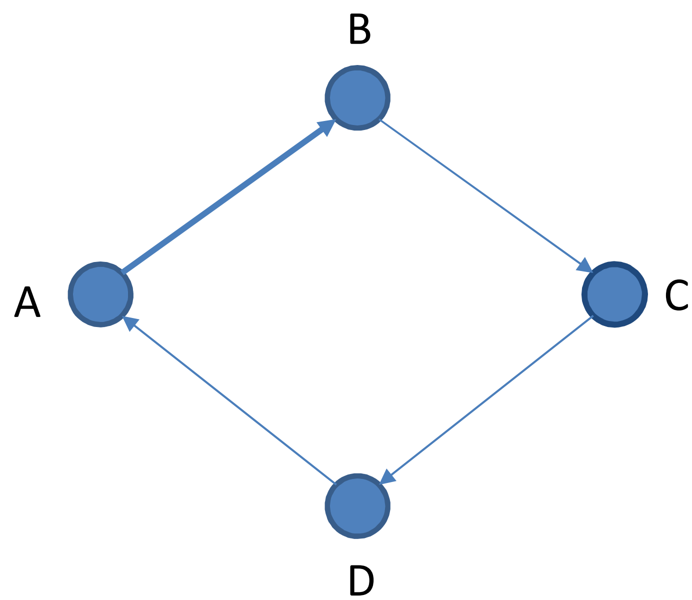
Network Elements

- Edges, arcs, links and relationships



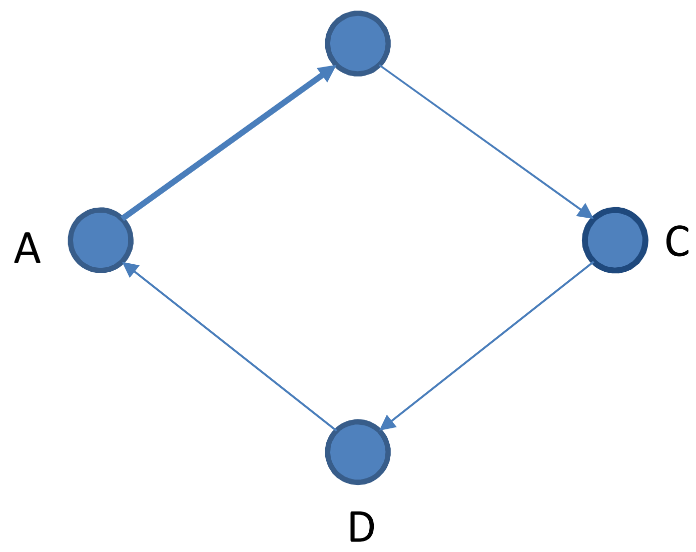
Representation

	A	B	C
1	A		
2	B		
3	C		
4	D		
5	A	B	
6	B	C	
7	A	B	
8	C	D	
9	D	A	



Representation

	A	B	C	D
A	-	2	0	0
B	0	-	1	0
C	0	0	-	1
D	0	0	1	-



Measurement

- Node Degree
- Diameter
- Density
- Degree Centrality
- In-degree centrality
- Out-degree centrality
- Betweenness centrality
- Closeness centrality

Node Degree

- The degree of a node in a network is the number of connections it has to other nodes and
- the degree distribution is the probability distribution of these degrees over the whole network.

Network Overview

- Average Degree
- Average Weighted Degree
- Distance
- Average Distance
- Network Diameter
- Modularity
- Connected Components

Network Overview

- Average Degree - Average number of links per node.

Network Overview

- Average Weighted Degree - Average of sum of weights of the edges of nodes.
-

Network Overview

- Distance - The distance between two nodes is defined as the number of edges along the shortest path connecting them.

Network Overview

- Average Distance - The Average of distance between all pairs of nodes.
-

Network Overview

- Network Diameter
  - The maximum distance between any pair of nodes in the graph.

Network Overview

- Modularity
  - Modularity is one measure of the structure of networks or graphs.
  - It was designed to measure the strength of division of a network into modules (also called groups, clusters or communities).
  - Networks with high modularity have dense connections between the nodes within modules but sparse connections between nodes in different modules.

Network Overview

- Connected Components
  - a connected component (or just component) of an undirected graph is a subgraph in which any two vertices are connected to each other by paths, and which is connected to no additional vertices in the supergraph.

Node Overview

- Clustering Coefficient
- Centrality
- Closeness Centrality
- Betweenness Centrality
- Eigenvector Centrality

Node Overview

- Clustering Coefficient - a clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together.

Node Overview

- Centrality
  - Centrality refers to indicators which identify the most important vertices within a graph.
  - e.g. the most influential person(s) in a social network, key infrastructure nodes in the Internet or urban networks, and super spreaders of disease.
  - In-degree centrality: popularity or prestige
  - Out-degree centrality: gregariousness

Node Overview

- Closeness Centrality
  - In connected graphs there is a natural distance metric between all pairs of nodes, defined by the length of their shortest paths.
  - The farness of a node is defined as the sum of its distances to all other nodes, and its closeness is defined as the reciprocal of the farness.
  - Thus, the more central a node is the lower its total distance to all other nodes.

Node Overview

- Betweenness Centrality
  - Betweenness is a centrality measure of a vertex within a graph (there is also edge betweenness, which is not discussed here).
  - Betweenness centrality quantifies the number of times a node acts as a bridge along the shortest path between two other nodes.
  - Allows to find “network broker(s)”

Node Overview

- Eigenvector Centrality
  - Eigenvector centrality is a measure of the influence of a node in a network.
  - It assigns relative scores to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes.

Python

```
# importar bibliotecas
import numpy as np
import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt
```

Generating a Network

```
# Generating a network
G = nx.barabasi_albert_graph(10, 3)
nx.draw(G, with_labels=True)
```

Degree

```
# degree of each node  link number that each node has
degrees = [deg for node, deg in nx.degree(G)]
print(degrees)

Result:
[4, 6, 1, 7, 6, 5, 4, 3, 3, 3]
```

Degree

```
# kmin - minimum degree
kmin = np.min(degrees)

# kmax - maximum degree
kmax = np.max(degrees)

# kavg - average degree
kavg = np.mean(degrees)
```

Shortest path

```
nx.shortest_path(G,1,2)
```

```
nx.shortest_path(G,1,2, weight=True)
```

Clustering coefficient

```
# triangles
nx.triangles(G)
# clustering coefficient of a node
nx.clustering(G)
# clustering coefficient of all nodes (returns a
dictionary)
nx.clustering(G)
# clustering coefficient of the network
cc = nx.clustering(G)
avg_clust = sum(cc.values()) / len(cc)
print(avg_clust)
```

Centrality

```
# betweenness centrality of network
nx.betweenness_centrality(G)
# closeness centrality of network
nx.closeness_centrality(G)
# eigenvector centrality of network
nx.eigenvector_centrality(G)
# degree centrality
nx.degree_centrality(G)
```

Connected Components

```
# find number of connected components
nx.number_connected_components(G)
# get the nodes in the same component as *n*
nx.node_connected_component(G, 3)
# Assortativity
# Pearson correlation coefficient [-1; 1]
# Social networks are highly assortative (homophily):
high degree
# nodes connect to other high degree nodes
# technological are disassortative: high degree nodes
connect to low
# degree nodes
nx.degree_assortativity_coefficient(G)
```