

A complex network diagram with numerous nodes and edges. Nodes are represented by circles of various sizes and colors, including yellow, green, blue, orange, and pink. Some nodes are highlighted with larger, semi-transparent circles. The edges are thin lines connecting the nodes, forming a dense web. The background is white, and the network is spread across the entire slide, with a central white box containing the title.

Lecture 2 · Graph Theory I

Networks, Crowds and Markets

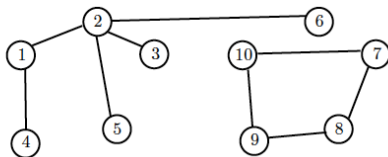
Today's Lecture

- Special graphs: complete, null, path, cycle.
- Degree. Handshaking Lemma. Degree sequence. Average degree. Degree distribution.
- Isomorphic graphs
- Subgraphs
- Adjacency matrix. Powers of adjacency matrix.
- Variations: Multigraphs, directed graphs, weighted graphs

Graph

Definition

A **graph** is a pair $G = (V, E)$ where $V = [N]$ is the set of N nodes (or vertices) and $E \subseteq \binom{V}{2}$ is a set of unordered pairs of distinct vertices, called edges (or links).



$$V = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

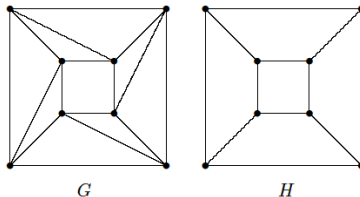
$$E = \{\overline{12}, \overline{14}, \overline{23}, \overline{25}, \overline{26}, \overline{78}, \overline{89}, \overline{910}\}$$

We often write $\{i, j\}$ or simply ij for an edge rather than \overline{ij} .

Subgraphs

Definition

$H = (V', E')$ is a **subgraph** of $G = (V, E)$ if all the nodes and edges of H belong to G ; $V' \subseteq V$, $E' \subseteq E$. We write $H \subseteq G$.



Here G and H have the same vertex sets but this is not the case generally for a subgraph.

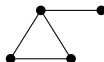
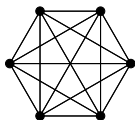
Special graphs

Special Graphs: The Complete Graph K_N

Definition

Graph is **complete** if every pair of vertices is joined by an edge.

A complete graph has $\binom{N}{2} = \frac{N(N-1)}{2}$ edges.



If $H \subseteq G$ and H is complete then H is called a **clique** of G .

In NetworkX:

```
import networkx as nx    - load NetworkX
import matplotlib.pyplot as plt

G = nx.complete_graph(N)  - the complete graph with N nodes.

nx.draw(G, with_labels=True, node_color="lightblue", node_size=600)
plt.show()
```

Special Graphs: The Null Graph (aka the empty graph)

Definition

Graph is **null** if it does not contain any edge, therefore $E = \emptyset$.

The **complement** \overline{G} of a graph $G = (V, E)$ has the same set of vertices and an edge is in \overline{G} if and only if it is not present in G .

The null graph is the complement of the complete graph.

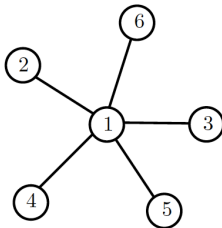
In NetworkX:

`G = nx.empty_graph(N)` – the empty graph with N nodes

Special Graphs: The Star Graph S_N

Definition

A **star graph** S_N has one central vertex connected to all other $N - 1$ vertices, and no other edges. The central node has degree $N - 1$, while all other nodes have degree 1.



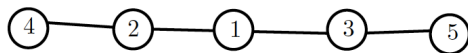
In NetworkX:

`G = nx.star_graph(N)` the star graph with $N + 1$ nodes

The Path Graph P_N

Definition

A **Path Graph** is a connected graph $G = (V, E)$ whose nodes can be listed in order (v_1, \dots, v_N) and the edges are $v_i v_{i+1}$. The central nodes have degree 2, the ones in the terminal vertices have degree 1.



Here $v_1 = 4$, $v_2 = 2$, $v_3 = 1$, $v_4 = 3$, $v_5 = 5$.

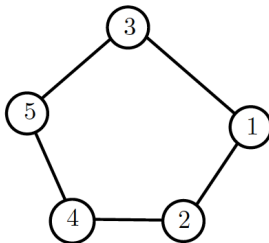
In NetworkX:

`G = nx.path_graph(N)` the path graph with N nodes

Special Graphs: The Cycle Graph C_N

Definition

A **cycle graph** C_N is obtained by connecting N vertices in a closed chain: $E = \{v_1 v_2, v_2 v_3, \dots, v_{N-1} v_N, v_N v_1\}$.



In NetworkX:

`G = nx.cycle_graph(N)` the path graph with N nodes

Special Graphs: Bipartite graphs

Definition

A graph $G = (V, E)$ is **bipartite** if the set of vertices can be partitioned into two parts $V = V_1 \cup V_2$ such that all edges have one end in V_1 and one end in V_2

- The star S_N
- The complete bipartite graph $K_{N,N}$
- Cycle C_N for even N .

Theorem

G is bipartite if and only if G has no odd cycles as subgraphs.

In NetworkX:

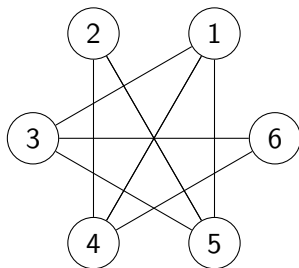
```
from networkx.algorithms import bipartite  
nx.is_bipartite(G)
```

Degree and degree distribution

Degree

Definition

The **degree** of a node $v \in V$ in an undirected graph, represented by $\deg(v)$, is the number of edges incident to it.



$$\deg(1) = 3$$

$$\deg(2) = 2$$

$$\deg(3) = 3$$

$$\deg(4) = 3$$

$$\deg(5) = 3$$

$$\deg(6) = 2$$

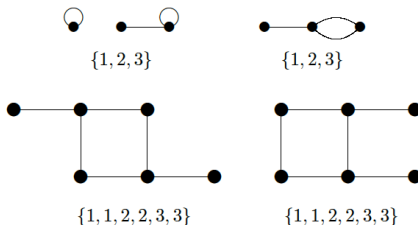
In NetworkX:

`print(G.degree(v))` – print the degree of node v

Degree sequence

Definition

The **degree sequence** of a graph is obtained by ordering, in an increasing way, the degrees of its nodes.



In NetworkX:

```
deg_seq = sorted([d for n, d in G.degree()])  
print(deg_seq_sorted)    – print the degree sequence of G
```

Degrees

$G = (V, E)$ a graph, $N = |V|$, $L = |E|$

- $\delta(G) = \min_{v \in V} \deg(v)$ **minimum degree** of G
- $\Delta(G) = \max_{v \in V} \deg(v)$ **maximum degree** of G
- $\overline{\deg}(G) = \frac{1}{N} \sum_{v \in V} \deg(v)$ **average degree** of G . ([B] uses $\langle k \rangle$)

Theorem (Handshaking Lemma)

The sum of the degrees of the vertices of a graph equals twice the number of edges:

$$\sum_{v \in V} \deg(v) = 2L.$$

In particular, the number of vertices of odd degree is always even.

(each edge contributes +1 to the degree of two nodes)

Exercise

Let $G = (V, E)$ be a graph with $N = 10$ nodes with the following degree sequence $\{0, 0, 1, 1, 1, 2, 2, 2, 3, 6\}$.

- a) Find the number of edges without drawing the graph.
- b) Find the average degree of the nodes in G .
- c) Draw a possible graph that satisfies the conditions.

Degree Distribution

Definition

The **degree distribution**, $p = (p_k)_{k \in \mathbb{N}}$, is a discrete function that provides the relative ratio of the vertices with a given degree k .

(In particular, $p_k = 0$ for $k \geq n$.)

Degree Distribution

Definition

The **degree distribution**, $p = (p_k)_{k \in \mathbb{N}}$, is a discrete function that provides the relative ratio of the vertices with a given degree k .

(In particular, $p_k = 0$ for $k \geq n$.)

Note that:

- $p_k = \frac{N_k}{N}$, (N_k = number of vertices of degree k)
- $\sum_{k=0}^{\infty} p_k = 1$ Normalization

Degree Distribution

Definition

The **degree distribution**, $p = (p_k)_{k \in \mathbb{N}}$, is a discrete function that provides the relative ratio of the vertices with a given degree k .

(In particular, $p_k = 0$ for $k \geq n$.)

Note that:

- $p_k = \frac{N_k}{N}$, (N_k = number of vertices of degree k)

- $\sum_{k=0}^{\infty} p_k = 1$ Normalization

- $\overline{\deg}(G) = \frac{1}{N} \sum_{v \in V} \deg(v) = \frac{1}{N} \sum_{k=0}^{\infty} k \cdot N_k = \sum_{k=0}^{\infty} k \cdot p_k$

Probabilistic viewpoint

This notation and terminology alludes to probability.

Consider a **probability distribution** $q = (q_k)$ on $\mathbb{N}_0 := \{0, 1, 2, \dots\}$.

If $X \sim q$ then $\mathbb{E}X = \sum_{k \geq 0} k \cdot q_k$.

Consider now a random sample $X_1, \dots, X_N \sim q$.

The statistics N_k counts the number of times we observed $X_i = k$.

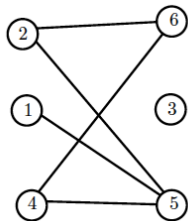
Sample distribution: $p = (p_k)$ with $p_k = \frac{1}{N} N_k$ for $k \geq 0$.

The sample average is

$$\bar{X}_N = \frac{1}{N} \sum_{i=1}^N X_i = \frac{1}{N} \sum_{k \geq 0} k \cdot N_k = \sum_{k \geq 0} k \cdot p_k.$$

If N is large $p \approx q$.

Example



$$p_k = \begin{cases} \frac{1}{6} & \text{if } k = 0 \\ \frac{1}{6} & \text{if } k = 1 \\ \frac{1}{2} & \text{if } k = 2 \\ \frac{1}{6} & \text{if } k = 3 \end{cases}$$

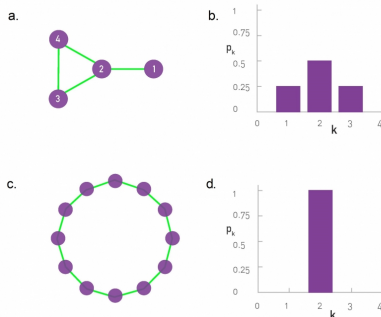
$$\overline{\deg}(G) = \sum_{k=0}^{\infty} k \cdot p_k = \frac{1}{6} \cdot 0 + \frac{1}{6} \cdot 1 + \frac{1}{2} \cdot 2 + \frac{1}{6} \cdot 3 = \frac{5}{3}$$

In NetworkX:

```
import numpy as np

values, counts = np.unique(deg_seq, return_counts=True)
distribution = counts / counts.sum()
```

Plotting the degree distribution

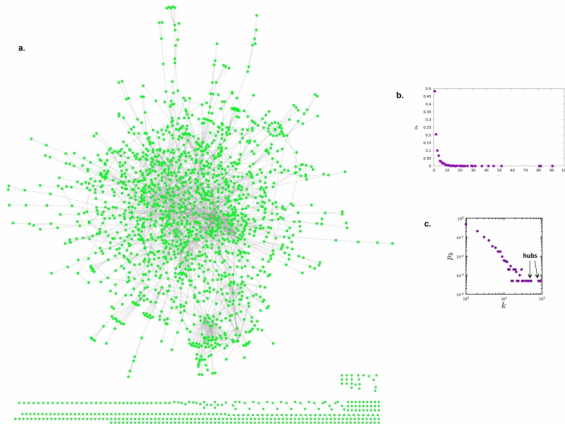


In NetworkX:

```
plt.hist(deg_seq, bins=range(max(deg_seq)+2), align='left', rwidth=0.8)
plt.xlabel("Degree")
plt.show()
```

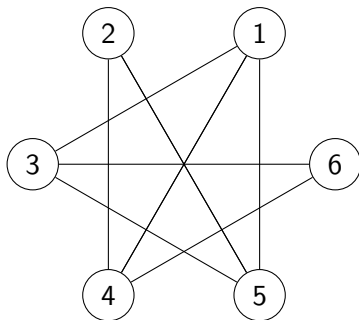
Power law

Many real networks have degree distributions that follow a **power law**, meaning the probability of high-degree nodes decays as a power of their degree, creating rare but influential hubs.



Exercise

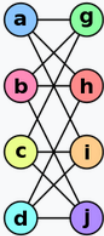
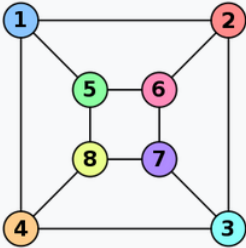
Given the following graph:



- a) Determine the average degree of the graph.
- b) Determine the degree distribution.

Isomorphic graphs

Isomorphic Graphs

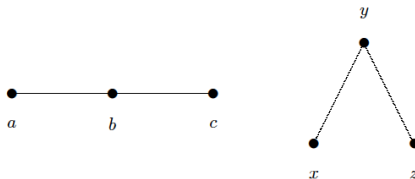
Graph G	Graph H	An isomorphism between G and H
		$f(a) = 1$ $f(b) = 6$ $f(c) = 8$ $f(d) = 3$ $f(g) = 5$ $f(h) = 2$ $f(i) = 4$ $f(j) = 7$

Isomorphic Graphs

Definition

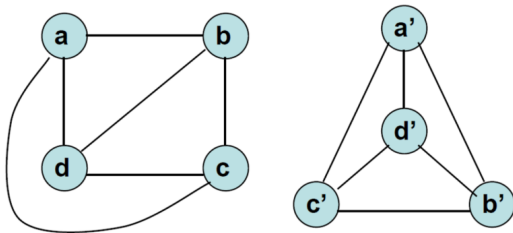
Two graphs G and H are **isomorphic** if there exists a bijection φ between their set of nodes that preserves the edges. Alternatively:

$$\exists \varphi : V_G \rightarrow V_H : \overline{xy} \in E_G \iff \overline{\varphi(x)\varphi(y)} \in E_H$$



Isomorphic Graphs

Example: $V_H = \{a, b, c, d\}$, $V_G = \{a', b', c', d'\}$.



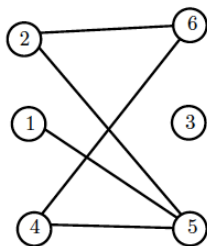
$\varphi(a) = a'$, $\varphi(b) = b'$, $\varphi(c) = c'$, $\varphi(d) = d'$.

Adjacency matrix

Adjacency Matrix

Definition

The **Adjacency Matrix** of an undirected graph is a square $N \times N$ matrix A in which the position (i,j) equals 1 if there is an edge between nodes i and j , otherwise, we type a 0.

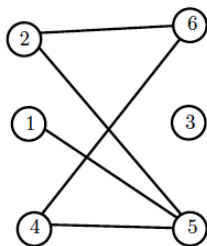


$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Adjacency Matrix

Definition

The **Adjacency Matrix** of an undirected graph is a square $N \times N$ matrix A in which the position (i,j) equals 1 if there is an edge between nodes i and j , otherwise, we type a 0.



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

In NetworkX:

`nx.to_numpy_array(G, nodelist=sorted(G.nodes()))` – G to A

`G = nx.from_numpy_array(A)` – A to G

NumPy provides fast, memory-efficient arrays and mathematical tools for scientific computing.

Adjacency Matrix

Some elementary properties

- For an undirected graph, the adjacency matrix A is symmetric.
- The sum of entries in row i is $\deg(i)$.
- Since there are no self-loops, the diagonal entries are zero.
- The entry $(A^2)_{ij}$ counts the number of walks of length 2 from i to j . In particular, $\text{tr}(A^2) = 2L$.
- More generally, $(A^m)_{ij}$ counts the walks of length m from i to j . In particular, $\text{tr}(A^3) = 6 \times (\text{number of triangles})$.

Note

Matrix powers count walks — this will reappear later in centrality measures and diffusion.

More complicated types of graphs

Multigraphs and loops

Note

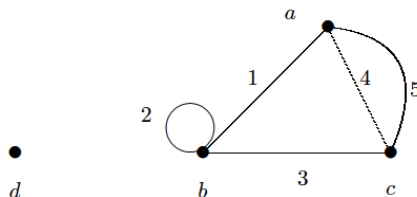
The graphs we have seen so far are called **simple** undirected graphs

- If two or more edges join the same pair of nodes, the graph is called a **multigraph**.
- If one edge joins a node with itself, we call it a **loop**.

In applications more complicated types of graphs may appear.

Undirected multigraphs

Example of a **multigraph with a loop**:



$$V = \{a, b, c, d\} \quad E = \{1, 2, 3, 4, 5\}$$

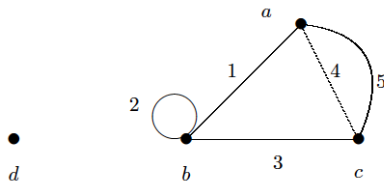
Remark: In this case, the edges have their own labels.

In NetworkX:

```
G = nx.MultiGraph()
G.add_nodes_from([1, 2, 3])
G.add_edge(1, 2)
G.add_edge(1, 2)    # parallel edge between 1 and 2
G.add_edge(2, 3)
```

Adjacency matrix of a multigraph

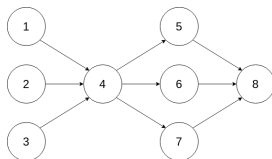
The entry ij is the number of edges joining i and j



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

- **Remark:** Adjacency matrices are symmetric ($a_{ij} = a_{ji}$) as they are undirected graphs.

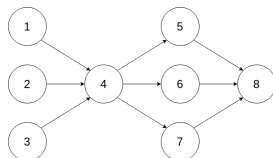
Directed Graph



Definition

A **directed graph** (digraph) is a pair $G = (V, E)$ where $V = [N]$ is the set of vertices and $E \subseteq V \times V$ is a set of ordered pairs of distinct vertices, called *directed edges* (or arcs).

Directed Graph



Definition

A **directed graph** (digraph) is a pair $G = (V, E)$ where $V = [N]$ is the set of vertices and $E \subseteq V \times V$ is a set of ordered pairs of distinct vertices, called *directed edges* (or arcs).

```
G = nx.DiGraph()
G.add_nodes_from(range(1, 9))
edges = [(1,4), (2,4), (3,4), (4,5), (4,6), (4,7), (5,8), (6,8), (7,8)]
G.add_edges_from(edges)
```

Examples of Directed Graphs

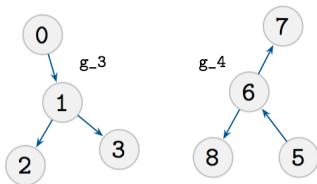
- **World Wide Web (WWW)** Vertices are webpages, edges are hyperlinks. Directed edges capture the one-way nature of links.
- **X/Twitter** Vertices are accounts, edges are “follow” relationships. Edges are directed (A follows B does not imply B follows A).
- **Academic Citation Network** Vertices are papers, edges are citations. Naturally directed in time: newer papers cite older ones.
- **SWIFT Network** Vertices are banks or institutions, edges represent international money transfers. Direction indicates sender \rightarrow receiver of funds.
- **Credit Networks** Vertices are individuals or institutions, edges indicate lending/borrowing. Direction shows the obligation flow (debtor \rightarrow creditor).

Isomorphism of directed graphs

Definition

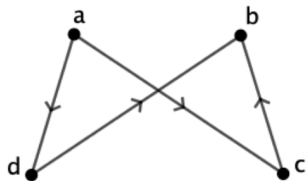
Two directed graphs G and H are **isomorphic** if there exists a bijection φ between their set of nodes that preserves the edges with its direction.

$$\exists \varphi : V_G \rightarrow V_H \text{ such that } (x, y) \in E_G \iff (\varphi(x), \varphi(y)) \in E_H$$

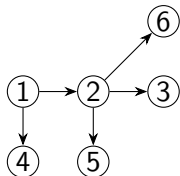


Quick question

Are they isomorphic?



In-degree and out-degree



Definition

For a directed graph $G = (V, E)$ and $v \in V$:

- The **in-degree** $d^-(v)$ is the number of edges arriving at v .
- The **out-degree** $d^+(v)$ is the number of edges leaving v .

Theorem (Handshaking Lemma for directed graphs)

$$\sum_{v \in V} d^+(v) = \sum_{v \in V} d^-(v) = |E|.$$

Average Degree

If A is the adjacency matrix, then the vector $A\mathbf{1}$ gives the degrees.

Thus $\frac{1}{N}\mathbf{1}^\top A\mathbf{1}$ gives the **average degree**.

a. Undirected

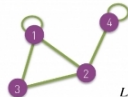


$$A_{ij} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$A_{ij} = 0 \quad A_{ij} = A_{ji}$$

$$L = \frac{1}{2} \sum_{i,j=1}^N A_{ij} \quad \langle k \rangle = \frac{2L}{N}$$

b. Self-loops



$$A_{ij} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

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

$$L = \frac{1}{2} \sum_{i,j=1, i \neq j}^N A_{ij} + \sum_{i=1}^N A_{ii} \quad ?$$

c. Multigraph
(undirected)

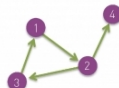


$$A_{ij} = \begin{pmatrix} 0 & 2 & 1 & 0 \\ 2 & 0 & 1 & 3 \\ 1 & 1 & 0 & 0 \\ 0 & 3 & 0 & 0 \end{pmatrix}$$

$$A_{ii} = 0 \quad A_{ij} = A_{ji}$$

$$L = \frac{1}{2} \sum_{i,j=1}^N A_{ij} \quad \langle k \rangle = \frac{2L}{N}$$

d. Directed



$$A_{ij} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$A_{ij} \neq A_{ji}$$

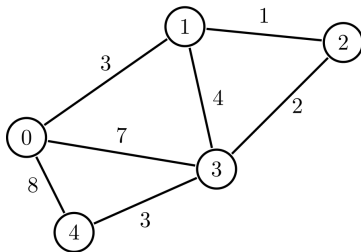
$$L = \sum_{i,j=1}^N A_{ij} \quad \langle k \rangle = \frac{L}{N}$$

Barabási denotes the average degree by $\langle k \rangle$. We do not follow this convention.

Weighted Graphs

Definition

A **weighted graph** is a pair (G, w) where $G = (V, E)$ is a simple graph and $w : E \rightarrow \mathbb{R}$ assigns a weight to each edge. Weights may represent distance, travel time, cost, or interaction strength (often nonnegative).



In NetworkX we can create a weighted graph from the (weighted) adjacency matrix – replace each 1 with the corresponding weight.

Examples of Weighted Graphs

Road Map V = intersections or cities, E = roads. *Weights*: distance, travel time, traffic congestion, or cost of tolls. Used in shortest-path algorithms (GPS navigation).

Social Networks V = individuals, E = relationships. *Weights*: frequency of interaction, strength of friendship, number of shared posts/messages. Captures tie strength rather than just “yes/no” connection.

E-mail Network V = people, E = e-mail communication. *Weights*: number of messages exchanged, total size of correspondence, or recency-weighted activity. Useful to detect communities or hubs of communication.

Food Chain Network V = species, E = “who eats whom”. *Weights*: proportion of diet, biomass transfer, or energy flow between species. Central in ecology to understand stability of ecosystems.

Exercise

Suppose $G = (V, E)$ s.t. $L = 800$ edges and $N = 1000$ nodes. The average degree is

$$\overline{\deg}(G) = \frac{2L}{N} = 1.6.$$

For large graphs it is often reasonable to approximate the degree of a randomly chosen node by a Poisson distribution with mean $\lambda := \overline{\deg}(G)$:

$$\mathbb{P}(\deg(v) = k) \approx \frac{\lambda^k}{k!} e^{-\lambda}, \quad k = 0, 1, 2, \dots$$

- a) Determine the average degree of the graph.
- b) What is the probability that a node has no edges attached?
- c) How many nodes of degree 3 do we expect to find?

(later we discuss more in detail when such Poisson approximations make sense)

Exercise

Let B be the adjacency matrix of a directed graph G with no loops and A be the adjacency matrix of the undirected version of G . Which of the following properties are true? Give evidence of your answer.

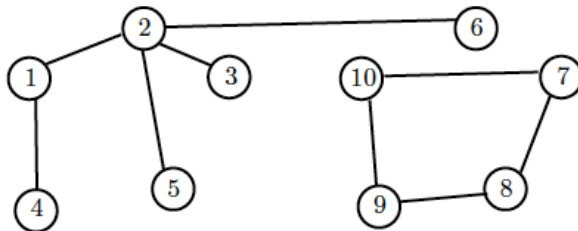
a) $a_{ij} + a_{ji} \geq 1$

b) $b_{ij} \leq a_{ij}$

c) $b_{ij} + b_{ji} \leq 1$

Exercise

Given the following graph:



- a) List the set of vertices and edges
- b) Find the degree for the 3 first nodes
- c) Find the average degree of all the nodes in the graph