Universitat Pompeu Fabra Barcelona

Seminar 2 Networks, Crowds and Markets

Random Networks

Exercise 1: Betweeness centrality.

We defined the *betweenness centrality* of a vertex x in a graph G as

$$b(v) = \sum_{s,t \neq v} \frac{\sigma_{st}(v)}{\sigma_{st}},$$

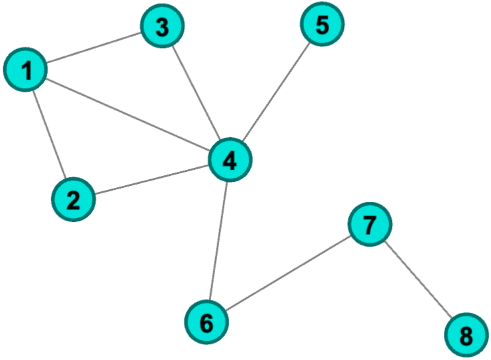
where $\sigma_{st}(v)$ denotes the number of shortest s-t-paths in G that go through v, and σ_{st} denotes the number of shortest s-t-paths.

(a) Consider a path on N vertices. Calculate the betweenness centralities. At which vertex are they maximised?

(b) Consider a tree on n vertices with a vertex v of degree k, so that its removal would divide the tree into k disjoint subtrees compromising $n_1, \ldots, n_k$ vertices. Show that $b(x) = \frac{1}{2}((N-1)^2 - \sum_{i=1}^{k} n_i^2)$.
Hint: each pair of vertices in different subtrees contributes 1 to $b(x)$.

Exercise 2: Centrality Measures

For the network shown below, compute all the centrality measures you know: Degree centrality, Closeness centrality, Betweenness centrality, Eigenvector centrality.



All measures except the eigenvector centrality can be computed by hand. Compare which nodes are most important under each criterion.

Associated code

```
import networkx as nx
import numpy as np

# Define the graph
edges = [
    (1,2), (1,3), (1,4), (2,4), (3,4),
    (4,5), (4,6), (6,7), (7,8)]

G = nx.Graph()
G.add_edges_from(edges)
centrality = nx.eigenvector_centrality_numpy(G)
print("\nEigenvector centrality:")
for node, c in centrality.items():
    print(f"Node {node}: {c:.4f}")
```

Exercise 3: Bounds on $\lambda_{\max}$

Show that for any simple undirected graph with adjacency A, the largest eigenvalue $\lambda_{\max}(A)$ is:

1. at least the average degree of G,
2. at most the maximum degree.

Hint:

- $\lambda_{\max}(A) = \max_{\|\mathbf{x}\|=1} \mathbf{x}^\top A\mathbf{x}$.
- $2x_i x_j \leq x_i^2 + x_j^2$

Exercise 4: Link probability distribution.

Given an ER graph with $N = 15$ and probability $p = 0.1$, determine:

a) What is the distribution of L?

b) What is the probability that $L \geq 15$?
   - Computing this by hand may be hard (see next slide).
   - Check that the Hoeffding bound is not very good in this case.

c) Find smallest $\ell \in \mathbb{N}$ such that $\mathbb{P}(L \geq \ell) \leq 0.05$ (use the code).
   - Use the one sided Hoeffding as an alternative way to construct such interval. What do you observe?

d) Do we expect a Giant Component?

One-sided Hoeffding: $X = \sum_{i=1}^{n} Z_i$ with $Z_i \in [0, 1]$ then

$$\mathbb{P}(X - \mu \geq t) \leq e^{-\frac{2t^2}{n}}.$$

Associated code

```
from scipy.stats import binom

# Parameters
n = 105     # number of trials
p = 0.1     # success probability

# Compute P(X >= 16) = 1 - P(X <= 14)
prob = 1 - binom.cdf(14, n, p)

print(prob)
```

Exercise 5: Degree and average degree in ER graphs

Let $X = \deg(v)$ for a fixed vertex in the $ER(N, p)$ graph:

- Compute the mean and the variance of $\deg(v)$.
- Compute the covariance of $\deg(u)$ and $\deg(v)$ for $u \neq v$.
- Are $\deg(u)$ and $\deg(v)$ independent?

Let $Y = \overline{\deg}(G)$ be the average degree in the $ER(N, p)$ graph.

- Show that $\mathbb{E}[Y] = (N-1)p$.
- What is the distribution of Y?
- Use the previous exercise to compute $\mathbb{P}(Y - (N-1)p \geq 2)$.
- Develop Hoeffding bound for $\mathbb{P}(Y - (N-1)p \geq t)$.

Exercise 6: Connectivity Threshold in $G(N, p)$

Let $\omega(N)$ be any sequence that grows to infinity (however slowly). Examples: $\log\log(N)$, $\sqrt{\log(N)}$, or even $\log\log\log(N)$.

In $G(N, p)$, the expected number of isolated vertices is

$$\mathbb{E}[N_0] = N(1-p)^{N-1} \approx Ne^{-p(N-1)}.$$

Suppose G is the $\mathrm{ER}(N, p)$

(a) Derive the formula above.

(b) Let $p = \frac{\log N - \omega(N)}{N}$ with $\omega(N) \to +\infty$. Show $\mathbb{E}[N_0] \to \infty$ and conclude G is disconnected w.h.p.

(c) Let $p = \frac{\log N + \omega(N)}{N}$ with $\omega(N) \to +\infty$. Show $\mathbb{E}[N_0] \to 0$. Can we conclude G is connected w.h.p.?
(actually $\mathbb{E}[N_k] \to 0$ for any fixed k)

Exercise 7: Triangles in ER models

Let T be the number of triangles in the $\mathrm{ER}(N, p)$ graph.

What is the expected number of T?

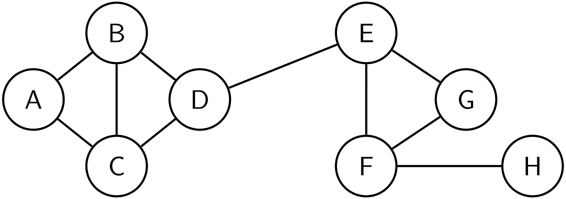
Bonus: Compute $\mathrm{Var}(T)$.

Additional exercises

Exercise: Degree vs. Eigenvector Centrality

Compute the **eigenvector centrality** of all nodes in the undirected graph below (you may use Python/NetworkX). Then compare with **degree centrality**.

- Which nodes are important under each measure?
- Why can these rankings differ?



The associated code

Python (NetworkX):

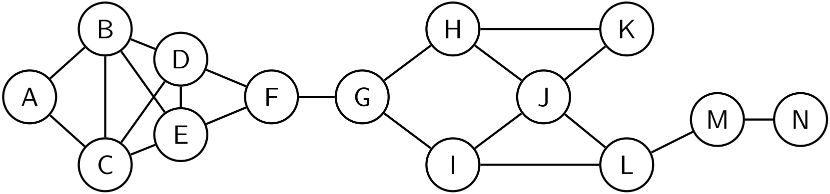
```
import networkx as nx

G = nx.Graph()
edges = [
 ('A','B'),('A','C'),('B','C'),('B','D'),('C','D'),
 ('D','E'),('E','F'),('E','G'),('F','G'),('F','H')
]
G.add_edges_from(edges)
x = nx.eigenvector_centrality(G, max_iter=1000, tol=1e-6)
x_rounded = {node: round(val, 3) for node, val in x.items()}
print(x_rounded)
```

Exercise: Closeness and Betweenness centrality

For the graph shown below, compute for every node:

- **Closeness centrality** $C_{\mathrm{close}}(v)$
- **Betweenness centrality** $C_{\mathrm{betw}}(v)$

Which measure better identifies bridge nodes in this network?



The associated code

```
G = nx.Graph()
edges = [
    ('A','B'),('B','C'),('C','A'),('B','D'),('D','E'),
    ('E','C'),('B','E'),('C','D'),('D','F'),('E','F'),
    ('F','G'),('G','H'),('G','I'),('H','J'),('I','J'),
    ('J','K'),('J','L'),('H','K'),('I','L'),('L','M'),
    ('M','N')]
G.add_edges_from(edges)

# --- Centralities ---
close = nx.closeness_centrality(G)                          # closeness
betw  = nx.betweenness_centrality(G, normalized=True)  # between

print("Node  Closeness   Betweenness")
for v in sorted(G.nodes(), key=lambda x: betw[x], reverse=True):
    print(f"{v:>4}  {close[v]:8.3f}   {betw[v]:10.3f}")
```

Exercise: PageRank (small directed web)

Consider the directed network with adjacency matrix

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

(a) Write down the transition matrix P.

(b) Find the stationary vector satisfying $P^\top \pi = \pi$, $\mathbf{1}^\top \pi = 1$.

(c) With teleportation $\alpha = 0.85$,

$$P_\alpha = \alpha P + (1 - \alpha)\frac{1}{4}\mathbf{1}\mathbf{1}^\top,$$

compute the PageRank vector π (solve $P_\alpha^\top \pi = \pi$, $\mathbf{1}^\top \pi = 1$).

Follow-up: Understanding PageRank qualitatively

The network is

$$1 \to \{2, 4\}, \quad 2 \to 3, \quad 3 \to \{1, 4\}, \quad 4 \to \varnothing.$$

Questions:

(a) Which node acts as a sink in the random walk defined by P? What happens to probability mass over time if there is no teleportation?

(b) After adding teleportation ($\alpha = 0.85$), which nodes PageRank values *increase* the most? Why does this happen?

(c) What is the qualitative effect of changing α?

- As $\alpha \to 1$, what happens to π?
- As $\alpha \to 0$, what does π converge to?

(d) Suppose we add one new edge $4 \to 1$. How would that affect the PageRank scores? (Hint: which node now becomes a stronger hub?)

Exercise: Random Walk Stationary Distribution

Let G be a connected, undirected graph with adjacency A and degree matrix D. The random walk transition is $P = D^{-1}A$.

(a) Show that $\pi_i = \dfrac{\deg(i)}{\sum_j \deg(j)}$ is a stationary distribution: $P^\top \pi = \pi$.

(b) Why is this stationary distribution unique when G is connected?

Exercise: Spectrum of $P = D^{-1}A$

Let G be a connected, undirected graph. Show / verify that the eigenvalues of $P = D^{-1}A$ lie in $[-1, 1]$.

- Hint: Relate P to the symmetric matrix $S = D^{1/2}PD^{-1/2} = D^{-1/2}AD^{-1/2}$.
- Why does $\lambda = 1$ correspond to the stationary distribution? Why simple (multiplicity 1) if G is connected?

(Optional: verify numerically on a medium graph.)

Exercise: Simulating the Giant Component (ER)

Simulate $G(N, p)$ with $N = 500$ and $p = c/N$ for $c \in \{0.5, 1, 1.5, 2, 3, 4\}$.

- For each c, estimate the fraction $\frac{|C_{\max}|}{N}$ of nodes in the largest component.
- Plot $\frac{|C_{\max}|}{N}$ vs. c and mark roughly where the phase transition occurs.

(You may do this as a short *optional homework* using NetworkX.)