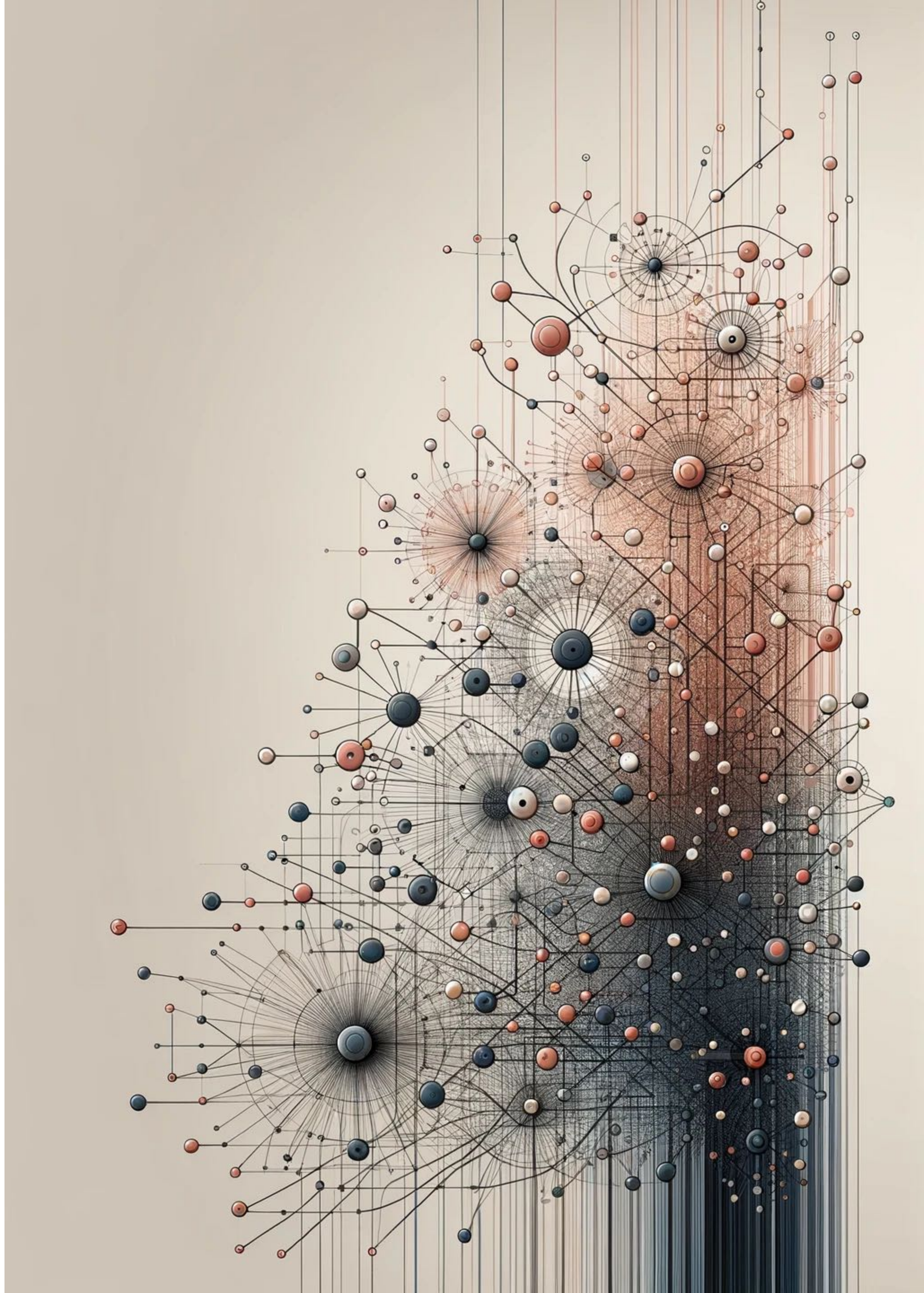
UNIVERSITÀ
DI TORINO

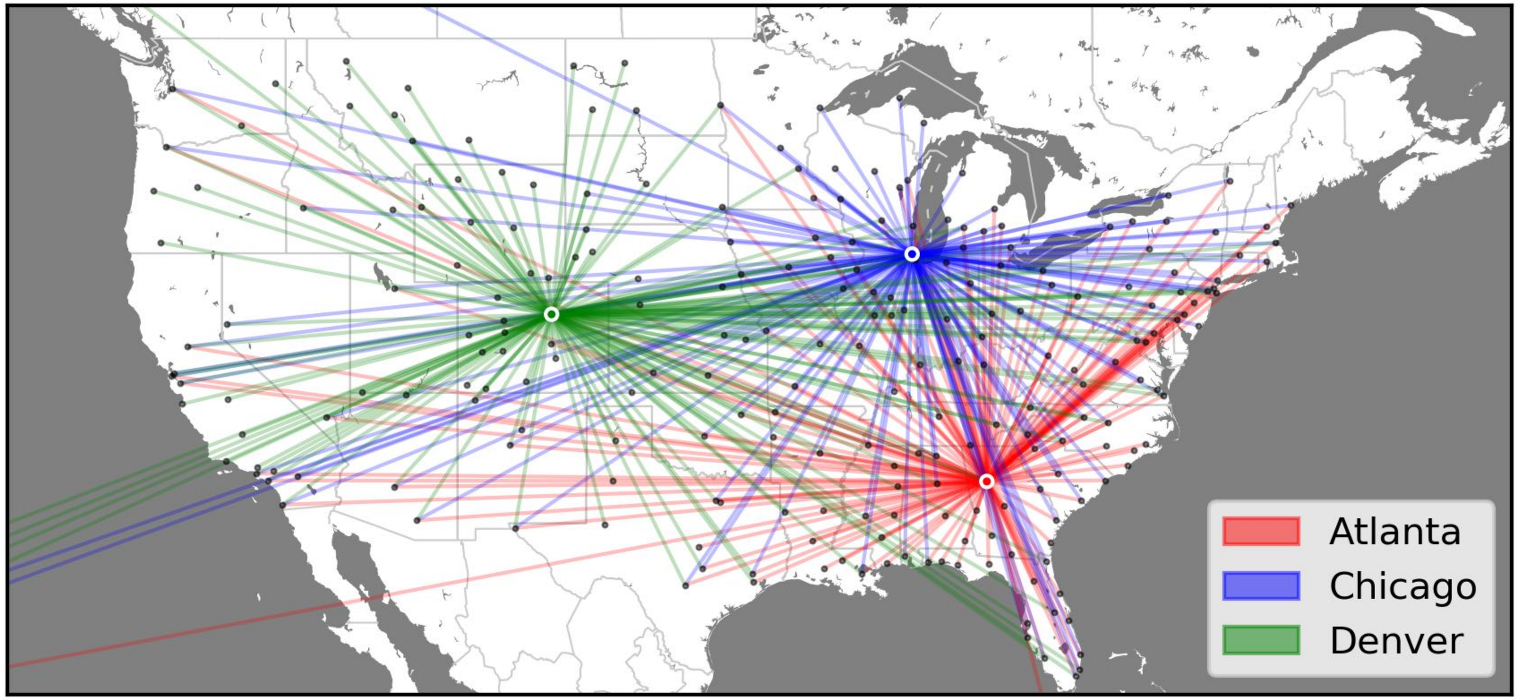
Analisi e Visualizzazione delle Reti Complesse

NS06 - Hubs, Centralities, Robustness

Prof. Rossano Schifanella



Real networks are heterogeneous



Some nodes (and links) are much more important (**central**) than others!

Centrality measures

- **Centrality**: it measures the **importance** of a node in a network
- It can be quantified by different **measures**:
    - **Degree**: The number of direct connections a node has. High-degree nodes are often referred to as **hubs**.
    - **Eigenvector**: Considers not just the quantity but the quality of connections. A node is important if it is connected to other important nodes.
    - **Katz**: An extension of eigenvector centrality that counts walks of all lengths between nodes, with longer walks given lower weight.
    - **Closeness**: Measures how close a node is to all other nodes in the network. A node with high closeness centrality can quickly interact with all other nodes.
    - **Betweenness**: Quantifies the number of times a node acts as a bridge along the shortest path between two other nodes. Nodes with high betweenness centrality have influence over the flow of information in the network.

Degree

- The **degree of a node** k_i is the **number of neighbors** of the node i
- High-degree nodes are called **hubs**
- **Average degree** of an undirected network:

$$\langle k \rangle = \frac{\sum_i k_i}{N} = \frac{2L}{N}$$

In NetworkX:

```
G.degree(2) # returns the degree of node 2

G.degree()   # returns a dict with the degree of all nodes of G
```

Applications

Social Networks

- **Influencers Identification**: In social media platforms like Twitter or Facebook, users with a high degree centrality are often influencers who can spread information quickly.

Biology

- **Protein-Protein Interaction Networks**: In biological networks, proteins with high degree centrality are often essential for the survival of an organism.

Transportation

- **Airport Networks**: Airports with high degree centrality are major hubs that connect many flights, playing a crucial role in the efficiency of air travel.

However, degree centrality is a purely local measure that only considers direct connections, potentially overlooking nodes that serve as critical bridges between different parts of the

Eigenvector

- With degree centrality, we treat all neighbors equally regardless of their importance
- In many contexts, **a node is more important if it connects to other important nodes**
- Eigenvector centrality captures this recursive notion of importance

For a node i, its eigenvector centrality x_i is defined as:

$$x_i = \frac{1}{\kappa} \sum_{j=1}^{n} A_{ij} x_j$$

Where:

- x_j is the centrality of neighboring node j
- A_{ij} is the adjacency matrix entry (1 if connected, 0 if not)
- κ is a proportionality constant

Matrix Formulation

The equation can be written in matrix form:

$$\mathbf{x} = \frac{1}{\kappa} A\mathbf{x}$$

Which rearranges to the standard eigenvalue equation:

$$A\mathbf{x} = \kappa\mathbf{x}$$

$\mathbf{x}$ is an eigenvector of the adjacency matrix A

Moreover:

1. since A is a non-negative matrix
2. we want $\mathbf{x}$ to be non-negative

because of the **Perron–Frobenius theorem $\mathbf{x}$** is the **leading eigenvector**

(Perron–Frobenius theorem states that for a matrix with all elements non-negative, like the adjacency matrix, there is only one eigenvector that also has all elements non-negative, and that is the leading eigenvector)

- **Then the eigenvector centrality x_i node i is the i^{th} element of the leading eigenvector of the adjacency matrix**
- **κ must be equal to the largest eigenvalue**
- Normalization could be applied, e.g., centralities sum to n (which ensures that average centrality stays constant as the network gets larger)
- Technically, this result is only true for connected networks, i.e., networks with only one component.
    - If a network has more than one component, then there is one eigenvector with nonnegative elements for each component.

```
nx.eigenvector_centrality(G)   # returns the eigenvector centrality for the graph G
```

- This works for only for undirected network

Why the Largest Eigenvalue?

- For eigenvector centrality, we use the eigenvector corresponding to the largest eigenvalue (principal eigenvector) because:
    - It represents the most stable solution to the recursive definition of importance
    - It's the only eigenvector guaranteed to have all non-negative values (by the Perron-Frobenius theorem)
    - It captures the most dominant pattern of connectivity in the network
- **Simple explanation:** The principal eigenvector identifies nodes that are:
    i. Connected to many other nodes, or
    ii. Connected to a few very important nodes, or
    iii. Some combination of both
- This provides a natural way to identify influential nodes based not just on how many connections they have, but on the quality of those connections.

Why Eigenvector Centrality Fails for Directed Networks

Problem: Source Nodes

- In our example network:

```
A → B → C
↑       ↓
D ← E ← F
```

- Recall the eigenvector centrality equation: $x_i = \frac{1}{\lambda} \sum_j A_{ji} x_j$
  - $A_{ji} = 1$ if there's an edge from j to i
  - $A_{ji} = 0$ otherwise
- For node A:
  - No incoming edges means all $A_{ji} = 0$
  - Therefore $x_A = 0$ (zero centrality)
- This zero propagates: if $x_A = 0$, then x_B becomes zero, then x_C, etc.

Katz Centrality

- **Extends eigenvector centrality** to solve problems with directed networks
- **Problem with eigenvector centrality in directed networks:**
  - Nodes with no incoming edges receive zero centrality
  - This creates a "zero centrality problem" that cascades through the network
- **Katz's solution:** Add a baseline importance to every node

$$x_i = \alpha \sum_{j=1}^{n} A_{ij} x_j + \beta$$

Where:

- x_i is the centrality of node i
- α is an attenuation factor (smaller than the reciprocal of largest eigenvalue)
- β is the baseline centrality given to each node (typically 1)
- A_{ij} is the adjacency matrix element (1 if there's an edge from j to i)
- In matrix form: $\mathbf{x} = \alpha A \mathbf{x} + \beta \mathbf{1}$
- Solution: $\mathbf{x} = \beta (I - \alpha A)^{-1} \mathbf{1}$

Katz Centrality

- **Walks-based interpretation**: Katz centrality can be interpreted as **counting walks of all lengths** from a node, but with longer walks attenuated by powers of α
- **Mathematical interpretation**: When we solve $\mathbf{x} = \alpha A\mathbf{x} + \beta\mathbf{1}$ to get $\mathbf{x} = \beta(I - \alpha A)^{-1}\mathbf{1}$, this expands to:

  $$\mathbf{x} = \beta(I + \alpha A + \alpha^2 A^2 + \alpha^3 A^3 + ...)$$

  Where:

  - A^k counts walks of length k between nodes
  - α^k reduces the importance of longer walks

- **Parameter significance**:
  - $\alpha < \frac{1}{\lambda_{max}}$ ensures convergence (where λ_{max} is the largest eigenvalue of A)
  - Smaller α values prioritize shorter walks
  - Larger α values (while still below the threshold) give more weight to longer walks
- **Relationship to other measures**:
  - When $\alpha \to 0$: Approaches degree centrality (only direct connections matter)
  - When α approaches $\frac{1}{\lambda_{max}}$: Approaches eigenvector centrality

In networkX:

```
nx.katz_centrality(G, alpha=0.1, beta=1.0)
```

Katz vs. Eigenvector Centrality: When to Use

Use Katz centrality when:

1. You're working with **directed networks**
2. Your network contains **source nodes** (with no incoming edges)
3. Your network has **multiple components**
4. You want to **control the extent of influence propagation** using α

```
# Computing Katz centrality for a directed graph
G = nx.DiGraph([(0,1), (1,2), (2,3), (3,0)])
centrality = nx.katz_centrality(G, alpha=0.1)
```

Closeness

Idea: **A node is more central the closer it is to the other nodes, on average**

$$g_i = \frac{1}{\sum_{j \neq i} l_{ij}}$$

where l_{ij} is the distance between nodes i and j

In a normalized form (discounting the graph size):

$$g_i = \frac{N-1}{\sum_{j \neq i} l_{ij}} = \frac{1}{\frac{\sum_{j \neq i} l_{ij}}{N-1}}$$

That is the inverse of the average distance from the focal node i to the rest of the network

```
nx.closeness_centrality(G, node)   # returns the closeness centrality of node i
```

Closeness Centrality: Key Applications

Public Health

- **Disease Control**: Optimal locations for vaccination centers to minimize epidemic response time
- **Healthcare Facility Placement**: Positioning medical facilities to reduce average patient travel time

Infrastructure & Emergency Services

- **Facility Location**: Optimal positioning for fire stations, distribution centers
- **Network Design**: Identifying critical infrastructure expansion points

Social Networks

- **Information Diffusion**: Finding individuals who can quickly spread information
- **Influence Maximization**: Identifying efficient "influencers" with broad reach

When Most Valuable

- When **speed of dissemination** is critical
- When **minimizing time delays or costs** across a system
- In **connected networks** where all nodes can reach each other

Limitations

- Less meaningful in **disconnected networks** (infinite distances between components)
- Sensitive to **network boundaries** and size
- Computationally expensive for very large networks

Betweenness

Idea: **A node is more central the more often it is crossed by shortest paths**

$$
\begin{aligned}
b_i &= \sum_{h \neq j \neq i} \frac{\sigma_{hj}(i)}{\sigma_{hj}} \\
\sigma_{hj} &= \text{number of shortest paths from } h \text{ to } j \\
\sigma_{hj}(i) &= \text{number of shortest paths from } h \text{ to } j \text{ running through } i
\end{aligned}
$$

This measure **depends on the size of the network**.

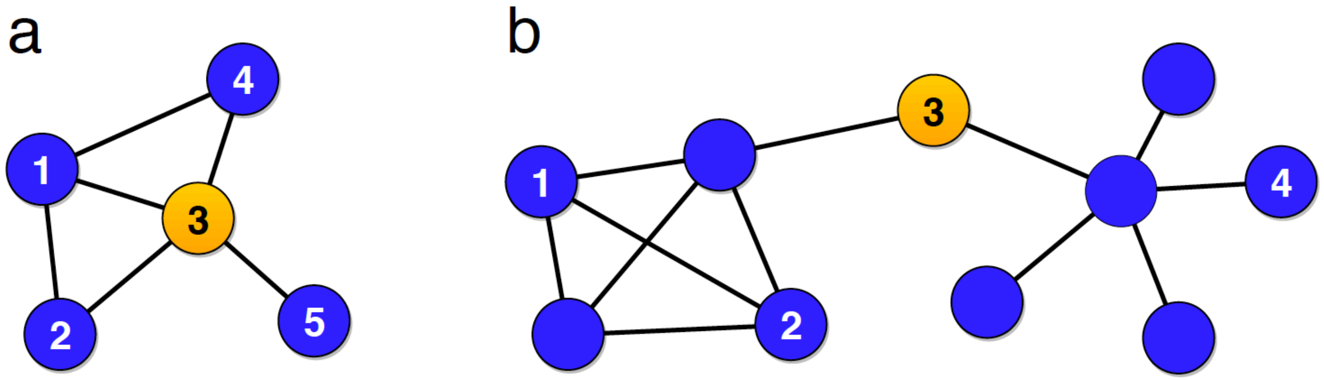
We can normalize the formula by the **maximum number of paths that could go through node** i, e.g., the number of pairs of nodes excluding i:

$$
\frac{b_i}{\binom{N-1}{2}} = \sum_{h \neq j \neq i} \frac{2\sigma_{hj}(i)}{(N-1)(N-2)\sigma_{hj}}
$$

Betweenness

Hubs usually have a **high betweenness** (a).

There can be **nodes with high betweenness that are not hubs** (b).



Edge Betweenness

Betweenness can be easily extended to edges.

Idea: **The fraction of shortest paths among all possible node pairs that pass through the link**

```
nx.betweenness_centrality(G)       # dict nodes: betweenness centrality

nx.edge_betweenness_centrality(G)  # dict links: betweenness centrality
```

Betweenness vs. Closeness Centrality

Measures how often a node acts as a bridge	Measures how easily a node can reach others
Focus on **control of information flow**	Focus on **efficient communication**
Identifies critical mediators and bottlenecks	Identifies nodes with quick access to the network
Based on **paths going through** a node	Based on **paths coming from** a node
High for bridge nodes between communities	High for nodes at the geometric center

Comparison of Centrality Measures

Degree	$k_i = \sum_j A_{ij}$	Direct influence/popularity	Local importance matters	Ignores network structure beyond direct connections
Eigenvector	$x_i = \kappa^{-1} \sum_j A_{ij} x_j$	Importance via important connections	Quality of connections matters	Issues with directed networks; requires connected graphs
Katz	$x_i = \alpha \sum_j A_{ij} x_j + \beta$	Extended influence accounting for all paths	Analyzing directed networks	Parameter tuning needed (α, β)
Closeness	$g_i = \frac{N-1}{\sum_{j \neq i} l_{ij}}$	Efficiency in reaching others	Speed of distribution matters	Less meaningful in disconnected networks
Betweenness	$b_i = \sum_{h \neq j \neq i} \frac{\sigma_{hj}(i)}{\sigma_{hj}}$	Control over information flow	Identifying critical bridges/bottlenecks	Computationally expensive

When to Choose Each Measure

- **Degree**: When direct connections and local influence are most important
- **Eigenvector/Katz**: When importance depends on the importance of neighbors
- **Closeness**: When efficient information diffusion or access to the network matters
- **Betweenness**: When identifying control points or bridges between communities

Centrality distributions

On small networks, it makes sense to ask which nodes or links are most important.

On large networks, it does not.

Solution: **A statistical approach**

> Instead of focusing on individual nodes and links, we consider classes of nodes and links with similar properties.

Understanding Network Centrality Distributions

When analyzing network centrality measures, we often examine their distributions:

- **Count (n_k)**: Number of nodes with degree k
- **Frequency (f_k)**:

  $$f_k = \frac{n_k}{N}$$

  The fraction of nodes having degree k in a network of size N
- **Probability (p_k)**:
  As network size grows large ($N \to \infty$), the frequency f_k converges to p_k, the probability that a randomly selected node has degree k

Example: In a 1,000-node network, if 100 nodes have degree 3, then $f_3 = 0.1$ or 10% of nodes have degree 3.

Probability Distributions in Network Analysis (PDF)

- **Probability distribution**: Shows how a variable's values are distributed
- For **degree distribution** $p(k)$:
  - $p(k)$ = probability that a randomly selected node has degree k
  - $p(k)$ = fraction of nodes having degree k (as $N \to \infty$)
- **How to plot**: Graph probability $p(k)$ on y-axis versus degree k on x-axis
- **Interpretation**: Shows relative frequency of nodes with different degrees
  - Reveals if network has many low-degree nodes and few hubs
  - Helps identify scale-free properties

Cumulative Distribution Function (CDF)

The **Cumulative Distribution Function (CDF)** provides the probability that a random variable (X) takes a value **less than or equal** to a given value (x).

Formally, the CDF (P(x)) is defined as:

$$P(x) = \sum_{v \leq x} p(v)$$

- $p(v)$ is the probability of observing value v.

Interpretation:

- The CDF accumulates probabilities from the lowest value up to x.
- It starts at 0 and increases monotonically to 1 as x increases.

Complementary Cumulative Distribution Function (CCDF)

- **Complementary Cumulative Distribution Function $P(x)$:**
  probability that the variable takes values larger than x as a function of x
- **How to compute it:**
  by summing the frequencies of the variable inside the intervals to the right of x

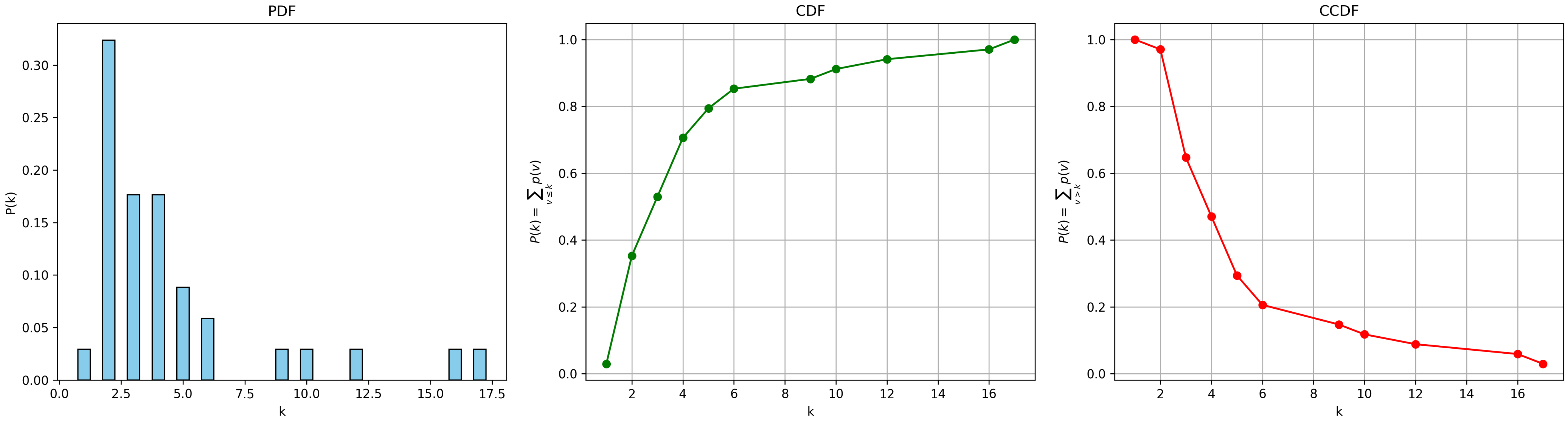
$$P(x) = \sum_{v>x} f_v$$

- The CCDF is the complement of the Cumulative Distribution Function, i.e., in a not-too-formal notation $CCDF(x) = P(X > x) = 1 - CDF(x) = 1 - P(X \leq x)$

Why use CCDF?

- Smooths out fluctuations in the tail
- Better reveals power-law behavior
- Reduces visual noise in sparse regions

Case of the Zacahary's Karate Club network



Logarithmic scale

Question:

How do you plot a probability distribution if the variable spans a large range of values, from small to (very) large?

Answer:

Use a **logarithmic scale**

How to do it:

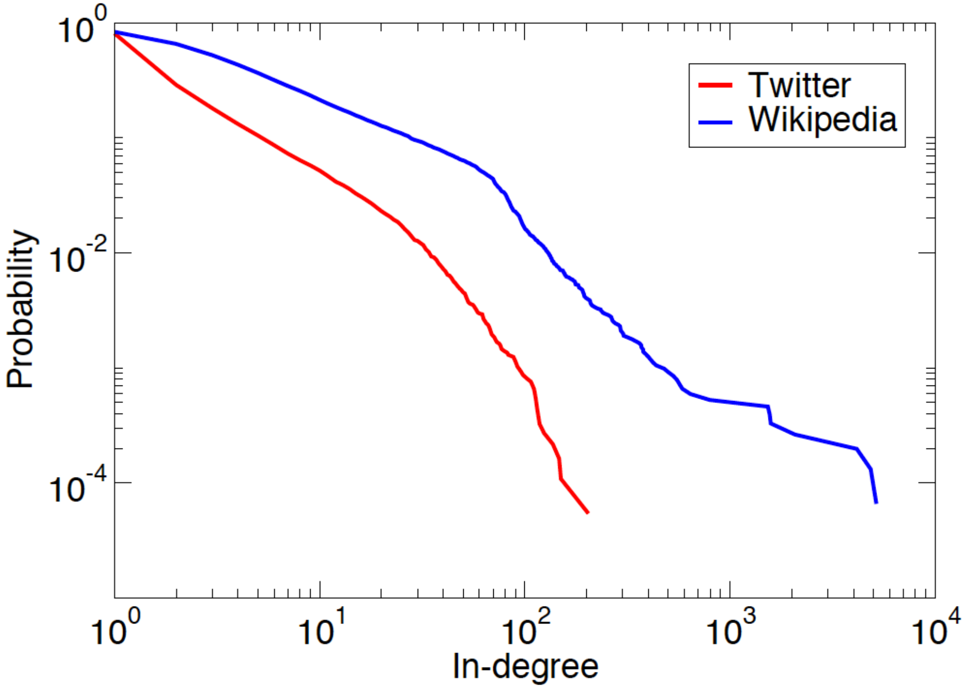
Report the logarithms of the values on the x- and y-axes.

$$\log_{10} 10 = 1$$

$$\log_{10} 1,000 = \log_{10} 10^3 = 3$$

$$\log_{10} 1,000,000 = \log_{10} 10^6 = 6$$

CCDF In-degree



Heavy-tail distributions: the variable goes from small to large values

Heterogeneity Parameter κ

The **heterogeneity parameter** measures the **breadth** of the degree distribution, comparing the variability of the degree across nodes to the average degree

Let us define the **average squared degree** $\langle k^2 \rangle$ as:

$$\langle k^2 \rangle = \frac{k_1^2 + k_2^2 + \cdots + k_{N-1}^2 + k_N^2}{N} = \frac{\sum_i k_i^2}{N}$$

$$\langle k \rangle = \frac{\sum_i k_i}{N} = \frac{2L}{N}$$

$$\kappa = \frac{\langle k^2 \rangle}{\langle k \rangle^2}$$

If most degrees have the same value, say k_0:

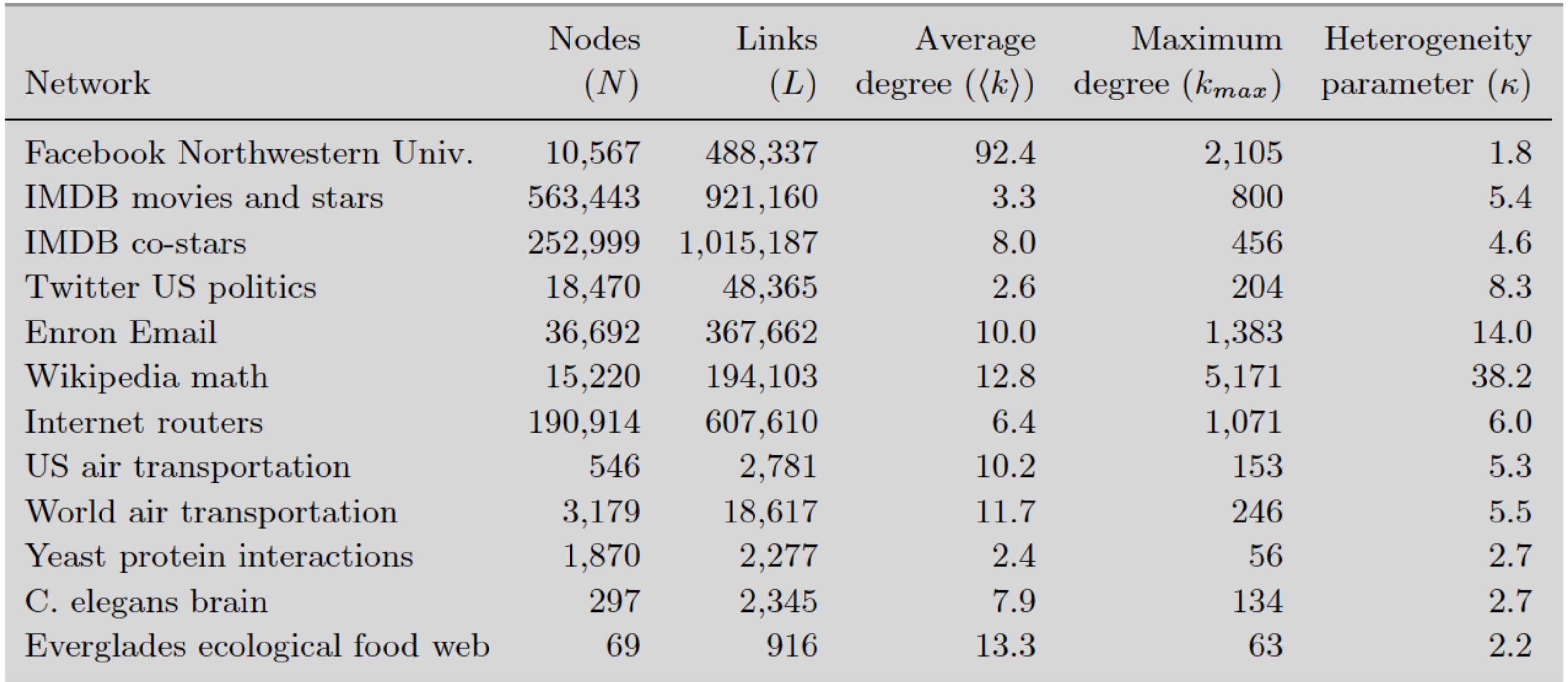
$$\langle k \rangle \approx k_0,\ \langle k^2 \rangle \approx k_0^2 \implies \kappa \approx 1$$

If the distribution is very heterogeneous:

$$\kappa \gg 1$$

If a network is directed, we have to consider two distributions, the **in-degree** and **out-degree** distributions, defined as the probability that a randomly chosen vertex has a given in- or out-degree, respectively.

Degree centrality of real-world networks

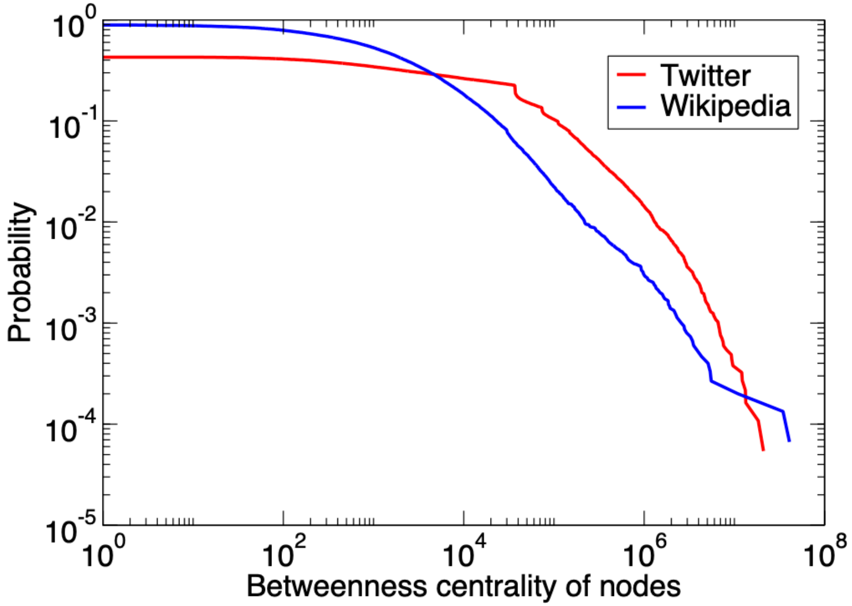


Network	Nodes (N)	Links (L)	Average degree ($\langle k \rangle$)	Maximum degree (k_{max})	Heterogeneity parameter (κ)
Facebook Northwestern Univ.	10,567	488,337	92.4	2,105	1.8
IMDB movies and stars	563,443	921,160	3.3	800	5.4
IMDB co-stars	252,999	1,015,187	8.0	456	4.6
Twitter US politics	18,470	48,365	2.6	204	8.3
Enron Email	36,692	367,662	10.0	1,383	14.0
Wikipedia math	15,220	194,103	12.8	5,171	38.2
Internet routers	190,914	607,610	6.4	1,071	6.0
US air transportation	546	2,781	10.2	153	5.3
World air transportation	3,179	18,617	11.7	246	5.5
Yeast protein interactions	1,870	2,277	2.4	56	2.7
C. elegans brain	297	2,345	7.9	134	2.7
Everglades ecological food web	69	916	13.3	63	2.2

UNIVERSITÀ DI TORINO

Betweenness distributions

- One can analyze the distributions of other properties besides the degree, e.g., betweenness
- It turns out that the **degree is usually correlated with other centrality measures**



- **Heavy-tail distributions:** the variable goes from small to large values

Handling Continuous Values with Binning

- Not all centrality measures produce discrete values:
  - **Discrete values**: Degree centrality (whole numbers)
  - **Continuous values**: Betweenness, closeness, eigenvector centrality
- For continuous values, we need a **binning approach**:
  i. Divide the range of values into intervals (bins)
  ii. Count observations in each bin
  iii. Normalize to get distribution
- **Why binning matters**:
  - Makes continuous data interpretable
  - Reveals patterns in distribution
  - Enables statistical analysis

Common Binning Approaches:

1. **Equal-width binning**:
    - Divide range into equal-size intervals
    - Example: [0-0.1), [0.1-0.2), [0.2-0.3), etc.
    - Simple but may create empty bins in skewed distributions
2. **Equal-frequency binning**:
    - Each bin contains approximately same number of observations
    - Adapts to data distribution
    - Better for highly skewed centrality measures
3. **Logarithmic binning**:
    - Bin widths increase exponentially
    - Ideal for heavy-tailed distributions
    - Example: [0.01-0.1), [0.1-1), [1-10), etc.

Creating Effective Bins: Step-by-Step

When analyzing centrality distributions with continuous values:

1. **Determine range**: Find min and max values in your data

```
min_val = min(centrality_values)
max_val = max(centrality_values)
```

2. **Choose bin count**: Rule of thumb: $\sqrt{n}$ or Sturges' formula $(1 + 3.322 \log_{10} n)$

```
n_bins = int(1 + 3.322 * np.log10(len(centrality_values)))
```

3. **Create bins**: Divide range into intervals

```
bins = np.linspace(min_val, max_val, n_bins + 1)
```

4. **Count values in bins**: Use histogram function

```
hist, bin_edges = np.histogram(centrality_values, bins=bins)
```

Visualizing Binned Distributions

Once you've created bins, effective visualization is key:

Basic histogram:

```
plt.hist(centrality_values, bins=10, alpha=0.7)
plt.xlabel('Centrality Value')
plt.ylabel('Frequency')
```

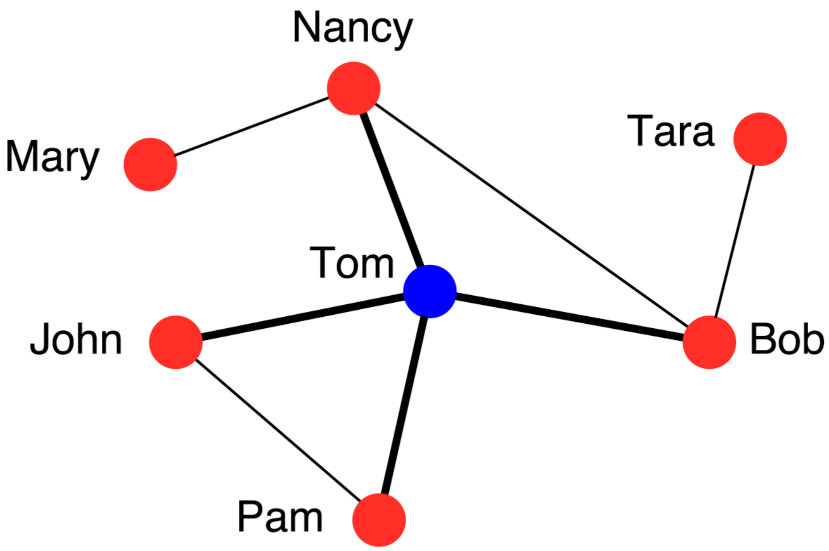
Probability density function (PDF):

```
hist, bin_edges = np.histogram(
    centrality_values, bins=10, density=True
)
bin_centers = (bin_edges[:-1] + bin_edges[1:]) / 2
plt.bar(bin_centers, hist, width=bin_edges[1]-bin_edges[0])
```

Log-log scale (for heavy-tailed distributions):

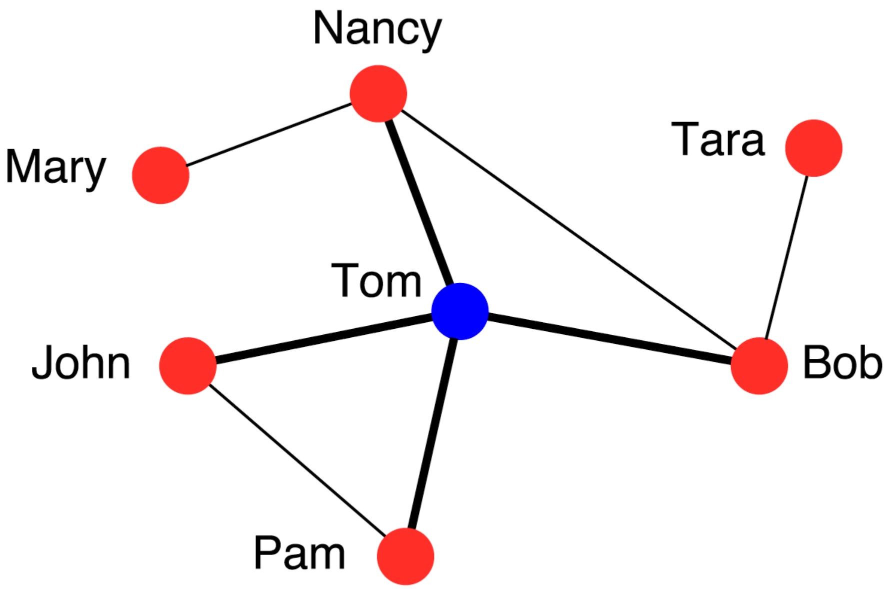
```
plt.loglog(bin_centers, hist, 'o-')
```

Friendship paradox



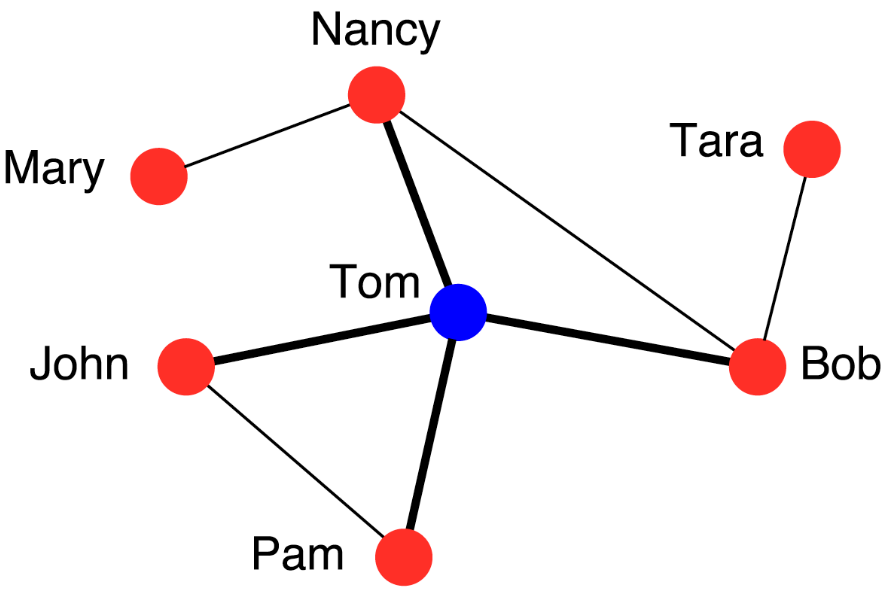
- By choosing a node at random, Tom has **the same chance** to be picked as everybody else
- By choosing at random a friend of a random individual (same as choosing a link at random), Tom has **a higher chance** of being picked than everybody else

Friendship paradox



By following links, **the chance to hit a hub increases.**

Friendship paradox



The average degree of a node = **2.29**

The average degree of the neighbors of a node = **2.83 > 2.29**

Our friends have more friends than we do, on average **(friendship paradox)**

Friendship paradox

Question:

Where does the friendship paradox come from?

Answer:

1. **By averaging the degree of the nodes, we pick them at random**
2. **By averaging the degree of the neighbors, we choose them by following links**: nodes with degree k will be counted k times, which inflates the average

- In other words, the Friendship Paradox is thus due to **sampling**. The two averages are computed by sampling the node degrees differently:
    - **uniformly** for average degree,
    - **proportionally to the degree** for the neighbors' average degree.

The more hubs, the stronger the effect.

Mathematical Explanation of the Friendship Paradox

Let's provide a clearer proof of why your friends have more friends than you do:

Definitions:

- μ_f = average number of friends (average degree)
- μ_{fof} = average number of friends-of-friends (neighbors' average degree)

Step-by-Step Proof

1. The **average degree** of the network:

$$\mu_f = \frac{\sum_i k_i}{N}$$

where k_i is the degree of node i and N is the total number of nodes.

2. The **average degree of neighbors** is calculated by:

$$\mu_{fof} = \frac{\sum_i k_i \cdot k_i}{\sum_i k_i} = \frac{\sum_i k_i^2}{\sum_i k_i}$$

Each node i contributes k_i times to the calculation (once for each of its neighbors).

3. From statistics, we know that the variance σ^2 of degree distribution is:

$$\sigma^2 = \frac{\sum_i k_i^2}{N} - \mu_f^2$$

1. Rearranging to isolate $\sum_i k_i^2$:

$$\sum_i k_i^2 = N(\mu_f^2 + \sigma^2)$$

1. Substituting this into our expression for μ_{fof}:

$$\mu_{fof} = \frac{N(\mu_f^2 + \sigma^2)}{N\mu_f} = \mu_f + \frac{\sigma^2}{\mu_f}$$

1. Since $\sigma^2 > 0$ (unless all nodes have exactly the same degree), we get:

$$\mu_{fof} > \mu_f$$

Why Is This True? A Simple Explanation

The friendship paradox occurs because **popular people are overrepresented in friendship counts**:

1. **Sampling bias**: When you count "friends of randomly selected people," you're more likely to include popular people in your sample
2. **Weighted average**: Popular people appear as friends of many others, so they're counted multiple times in the neighbors' average
3. **Mathematical consequence**: The difference between μ_{fof} and μ_f increases with the **variance** in the degree distribution

The more heterogeneous your network (some very popular people, some with few connections), the stronger the paradox becomes!

Real-World Implications

The friendship paradox has practical applications in:

- **Epidemic monitoring**: Monitoring friends of random people detects outbreaks earlier
- **Social influence**: People tend to feel less popular than they actually are
- **Marketing**: Friends of random users may be better targets for campaigns
- **Vaccination strategies**: Immunizing friends of random individuals can be more effective than random vaccination

This paradox reminds us that our perception of "normal" is often skewed by sampling biases in our social environments.

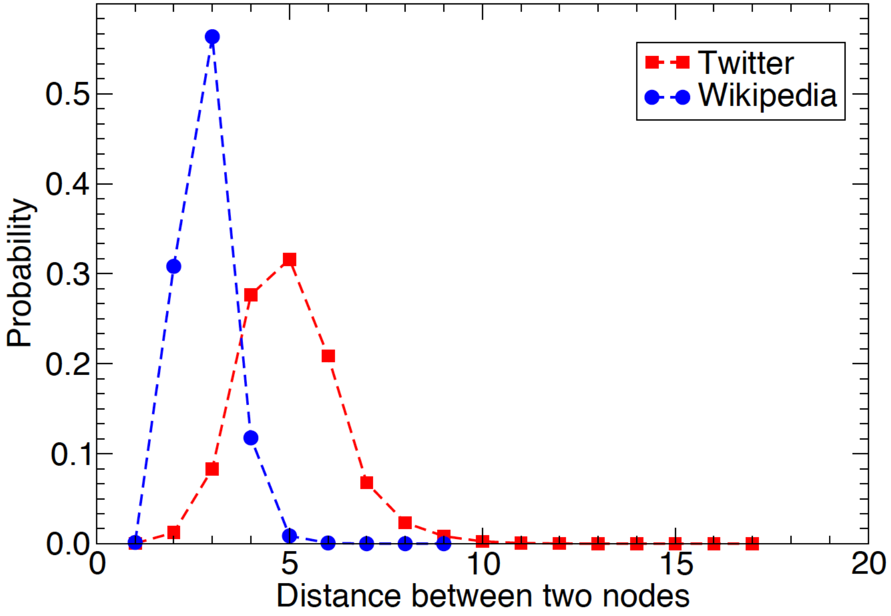
Ultra-small worlds

In real networks, many of the shortest paths go through hubs.

- Example: air transportation
- There may be no routes between airports A and B (if they are small), but it may be possible to go from A to B via a hub airport C

The small-world property is typical of most networks of interest

With **hubs, paths are ultra-short** (ultra-small world)



Robustness

A system is **robust** if the **failure of some of its components does not affect its function**.

Question:

How can we define the robustness of a network?

Answer:

We remove nodes and/or links and see what happens to its structure

Key point:

Connectedness

If the Internet were not connected, transmitting signals (e.g., emails) between routers in different components would be impossible.

Robustness

Robustness test:

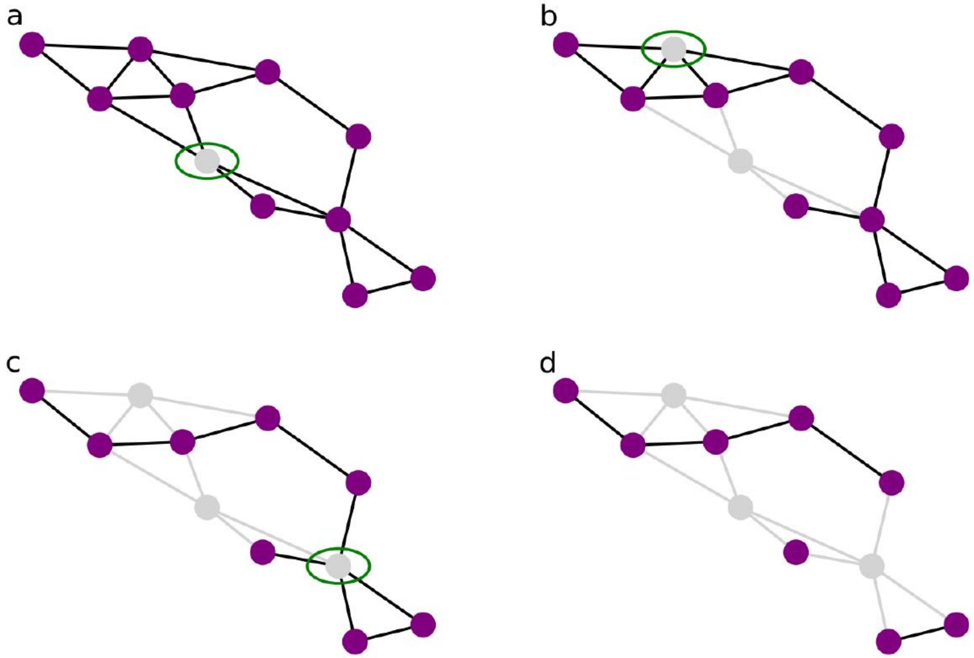
Checking how the connectedness of the network is affected as nodes are removed.

How to do it:

Plot the relative size S of the largest connected component as a function of the fraction of removed nodes.

- We suppose that the network is initially connected
  - there is only one component, and $S = 1$
- As more and more nodes (and their links) are removed, the network is progressively broken up into components, and S goes down.

Robustness



Robustness

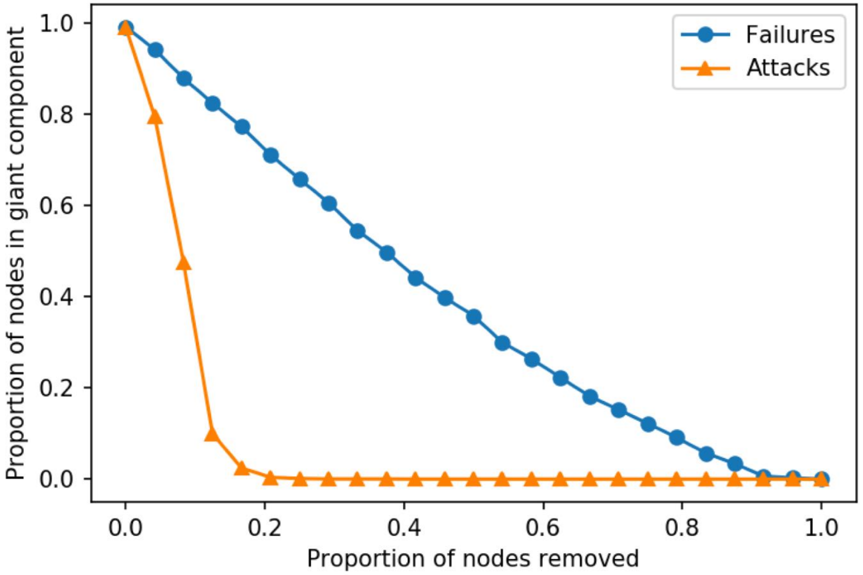
Two strategies:

1. **Random failures:** nodes break down randomly, so they are all chosen with the **same probability**
2. **Attacks:** hubs are deliberately targeted — the **larger** the **degree**, the higher the probability of removing the node

In the first approach: we remove a fraction f of **randomly** chosen nodes.

In the second approach: we remove the fraction f of **nodes with the largest degree**.

Robustness



Real networks are:

1. **Robust against random failures**
2. **Fragile against targeted attacks**

Additional readings

- Linton C. Freeman, Centrality in social networks. A conceptual clarification, Social networks, vol. 1, no 3, 1979, p. 215–239
- Ulrik Brandes (2001) A faster algorithm for betweenness centrality, The Journal of Mathematical Sociology, 25:2, 163-177
- Newman, Mark EJ, and Michelle Girvan. "Finding and evaluating community structure in networks." Physical review E 69.2 (2004): 026113

Reading material

References

[ns1] **Chapter 3**

Q&A 🙋