



UNIVERSITÀ
DI TORINO

Analisi e Visualizzazione delle Reti Complesse

NS08 - Network Models

Prof. Rossano Schifanella





Outline

- Random networks
- Small-world networks
- The configuration model



Recap on structural characteristics of real networks

Features of real networks

- **small-world property**

- Most real-world networks have **short paths** between any pair of nodes
- The average shortest path length scales logarithmically with network size: $\langle l \rangle \sim \log N$
- Typically characterized by a "six degrees of separation" phenomenon
- Examples:
 - Social networks: any two people are connected through a short chain of acquaintances
 - Brain networks: information can travel efficiently between distant brain regions
 - Internet: data can reach any destination in a few hops
- Implications:
 - Efficient information/disease spread
 - Fast communication across the network
 - Network is "navigable" with local information
- Co-exists with clustering in real networks (unlike random networks)

Features of real networks

- **high clustering coefficient**

- The clustering coefficient of a node is the fraction of pairs of the node's neighbors that are connected to each other:

$$C(i) = \frac{\tau(i)}{k_i(k_i - 1)/2} = \frac{2\tau(i)}{k_i(k_i - 1)}$$

- where $\tau(i)$ is the number of triangles involving i . Note that in this definition, the clustering coefficient is undefined if $k_i < 2$, i.e., a node must have at least degree 2 to have any triangles.
- NetworkX assumes $C = 0$ if $k = 0$ or $k = 1$
- Many networks have high clustering coefficients
- Other networks, e.g., bipartite and tree-like networks, have low clustering coefficient

Features of real networks

- **scale-free**

- A network is **scale-free** if its **degree distribution** follows a **power-law**: $P(k) \sim k^{-\gamma}$
- Where k is the node degree and γ is the exponent (typically $2 < \gamma < 3$ for real networks)
- A power-law distribution $P(k) \sim k^{-\gamma}$, when plotted on a **log-log scale**, the distribution appears as a **straight line**
 - Mathematically, taking the logarithm of both sides:
 $\log P(k) = -\gamma \log k + \log C$ (where C is a constant)
 - This is equivalent to the equation of a straight line $y = mx + b$ where $y = \log P(k)$, $x = \log k$, $m = -\gamma$ (the slope is the negative exponent) and $b = \log C$ (y-intercept)
- In a power-law distribution:
 - There are **many nodes with few connections**
 - There are **few nodes with many connections** (hubs)
 - The distribution has a **heavy tail**

- The term "scale-free" refers to the **lack of a characteristic scale** in the degree distribution
 - No "typical" node with which to characterize the network
 - Degrees span several orders of magnitude
 - The power-law distribution remains the same at different scales.

Characteristics of scale-free networks

- **Presence of hubs:** nodes with abnormally high degree
 - Hubs can have orders of magnitude more connections than average nodes
 - Hubs often play critical roles in the network's function
- **Heavy-tailed degree distribution**
 - The probability of finding extremely high-degree nodes is not negligible
 - No sharp cutoff in the maximum degree
- **High heterogeneity parameter** $\kappa = \frac{\langle k^2 \rangle}{\langle k \rangle^2}$
 - In scale-free networks, κ can be very large or even diverge
 - In random networks, $\kappa \approx 1 + \frac{1}{\langle k \rangle}$

Properties of scale-free networks

- **Robustness against random failures**
 - Random removal of nodes is unlikely to affect hubs
 - Network connectivity remains largely intact
- **Vulnerability to targeted attacks**
 - Removing hubs can quickly fragment the network
 - Critical for understanding network resilience
- **Ultra-small world property**
 - Average path length scales as $\sim \ln \ln N$ (where N is the number of nodes)
 - Even shorter paths than in random networks ($\sim \ln N$)



Models

Model:

A set of instructions to build networks

Goal:

Find models that generate networks with the same characteristics as real-world networks

Why build models?

To understand the fundamental processes that create real networks

The importance of network models

- **Explanation:** Models help us understand the underlying mechanisms that generate network structures we observe
 - Why do real networks have hubs?
 - Why do social networks form dense communities?
 - What causes the small-world property?
- **Prediction:** Models let us forecast how networks might evolve or respond to changes
 - How will removing certain nodes affect connectivity?
 - How quickly will information spread?
 - How resilient is the network to failures or attacks?
- **Benchmark:** Models provide reference points to compare real networks against
 - How different is our observed network from random?
 - Which features are significant vs. expected by chance?

Scientific value of network models

- **Parsimony:** Good models capture complex network properties using simple rules
 - Example: Preferential attachment explains scale-free degree distributions with a single mechanism
- **Generative understanding:** Models reveal how microscopic rules lead to macroscopic properties
 - Example: Simple rewiring rules in Watts-Strogatz model create small-world networks
- **Counterfactuals:** Models let us explore "what if" scenarios
 - Example: How would the Internet evolve with different connection rules?
- **Abstraction:** Models strip away details to reveal fundamental principles
 - Example: Despite their differences, citation networks and the Web share similar growth patterns



Random networks

Random networks

Simple idea

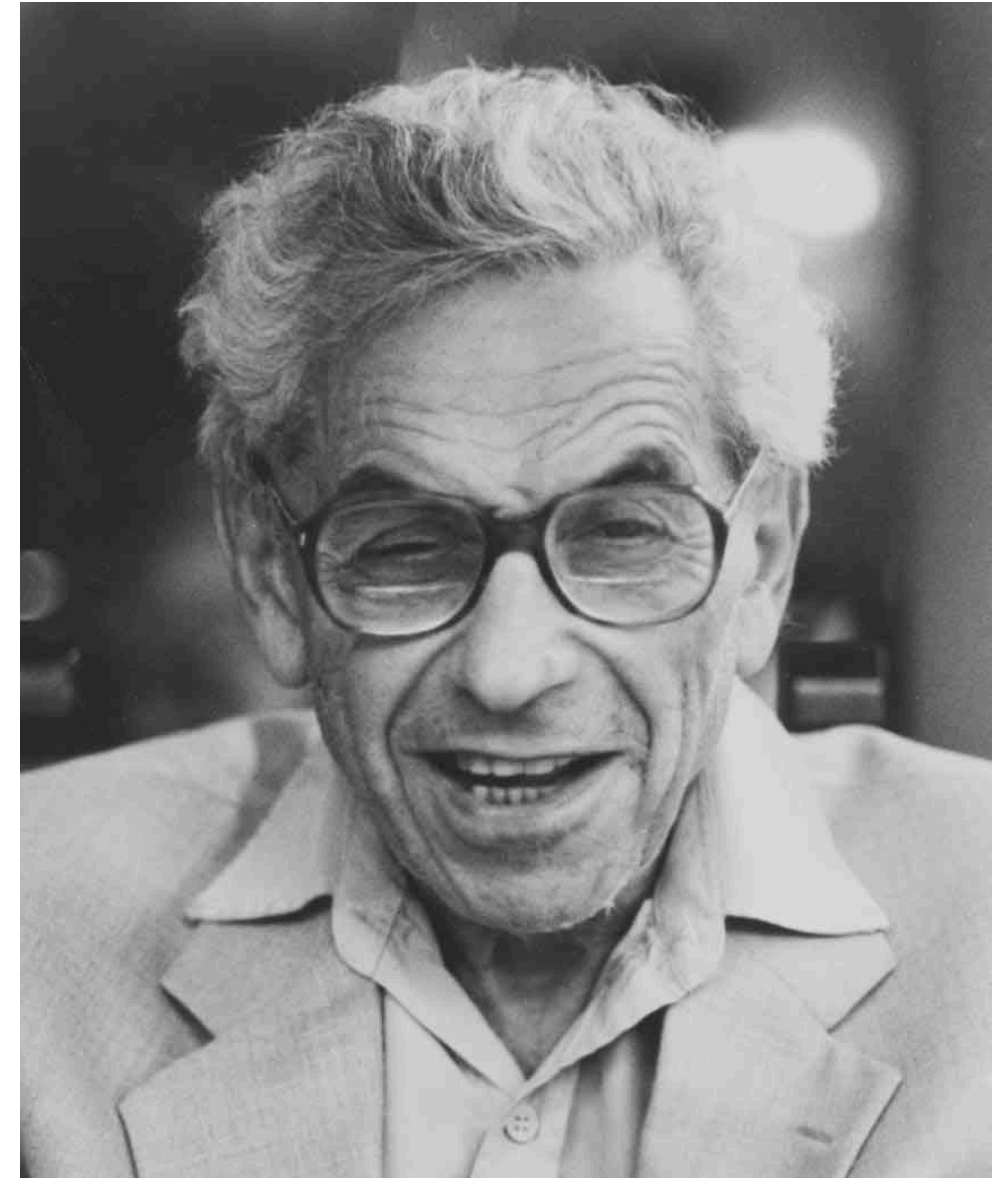
Placing links at random between pairs of nodes

Paul Erdős (1913-1996)

Famous for the Erdős–Rényi random model

We present in these slides an equivalent formulation: the Gilbert's model

Main difference: version by Erdős and Rényi the number of links of the network is fixed, whereas in the model by Gilbert it is variable



Random networks

Gilbert's random network model

$G(n, p)$ where n is the number of nodes and $0 < p < 1$ is the probability that an edge occurs.

Algorithm:

1. Start with n nodes and zero links
2. Go over all pairs of nodes $\binom{n}{2}$; for each pair of nodes i and j , generate a random number r between 0 and 1
 - If $r < p \Rightarrow$ **i and j get connected**
 - If $r > p \Rightarrow$ **i and j remain disconnected**

The probability of obtaining any one particular random graph with m edges is $p^m(1 - p)^{\binom{n}{2} - m}$

Random networks: evolution

Let us focus on the **connected components**

- With $p = 0$ **no links**: N components with one node each
- With $p = 1$ **all links are there**: one component (complete network) with N nodes

Question:

What happens as we add links to the network?

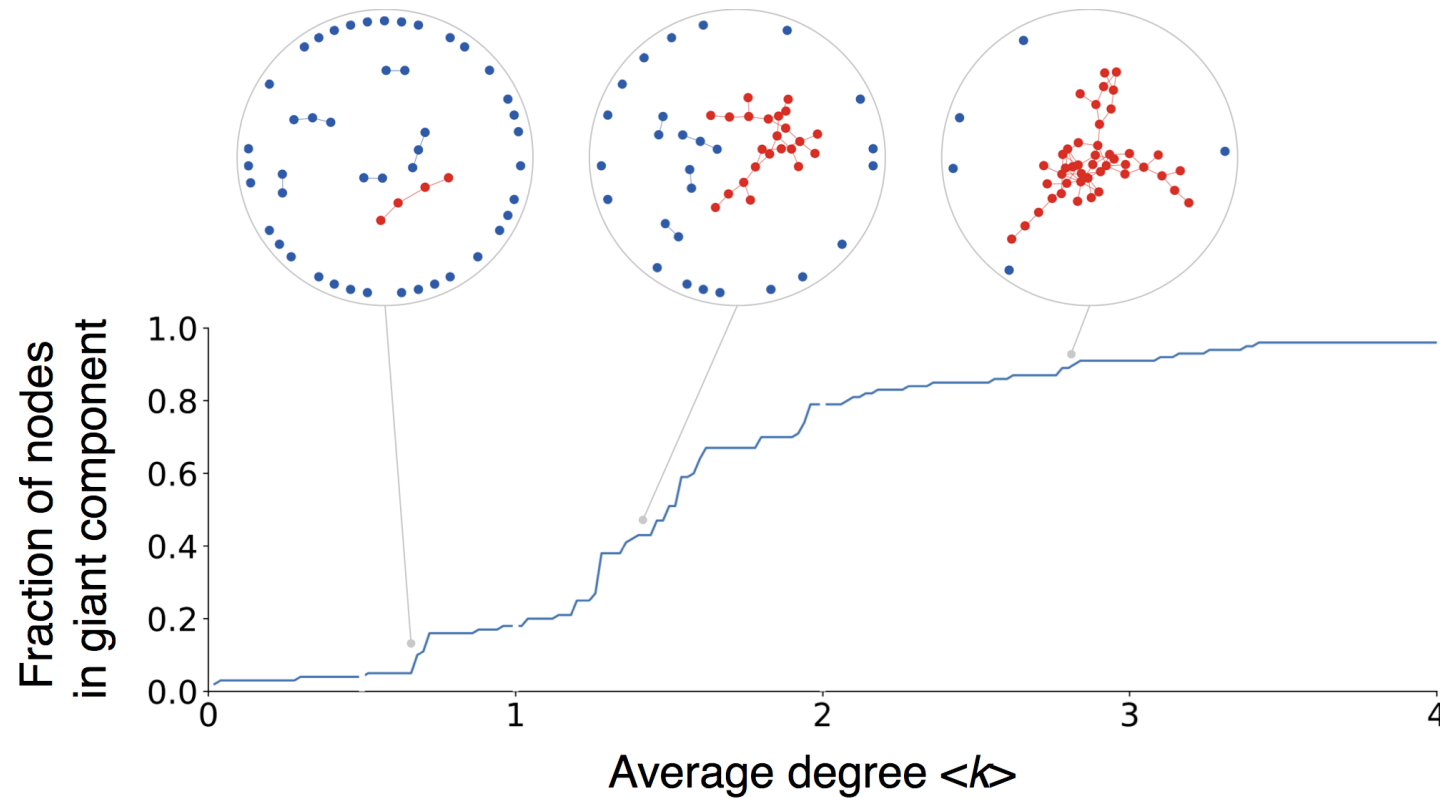
Naïve expectation:

The size of the largest component grows smoothly with the number of links

Wrong expectation:

There is an **abrupt increase** for a given value of the link probability p

Random networks: evolution



Around $\langle k \rangle = 1$ a giant component grows very fast at the expense of the other, smaller components.

[\[Example in NetLogo\]](#)

Random networks: number of links, density, average degree

- Equivalence with the tossing of a biased coin, which yields heads with probability p
- Number of independent trials (tosses): t
- Number of heads after t trials: h
- Special cases:
 - $p = 0$ the coin never yields heads $\Rightarrow h = 0$
 - $p = 1$ the coin always yields heads $\Rightarrow h = t$
 - $p = \frac{1}{2}$ the coin yields heads (about) half of the times $\Rightarrow h \approx \frac{t}{2}$
- General rule:
 - $h \approx p \cdot t$

Random networks: number of links, density, average degree

Expected number of links $\langle L \rangle$ of a random network with N nodes:

Number of heads with probability of yielding heads equal to p and the number of trials t equal to the number of all node pairs of the network:

$$t = \frac{N(N-1)}{2} \rightarrow \langle L \rangle = p \binom{N}{2} = p \frac{N(N-1)}{2}$$

Expected density of links d of a random network with N nodes:

$$d = \frac{\langle L \rangle}{\frac{N(N-1)}{2}} = \frac{p \frac{N(N-1)}{2}}{\frac{N(N-1)}{2}} = p$$

Real-world networks are **sparse**.

For random networks to be better models of real networks, p must be very small.

Random networks: number of links, density, average degree

Expected average degree $\langle k \rangle$ of a random network with N nodes:

Let's derive this step by step:

1. For any node in a random network, it has $N - 1$ potential neighbors (all other nodes)
2. For each potential connection, the probability of a link existing is p
3. Therefore, the expected number of links per node is:

$$\langle k \rangle = p \cdot (N - 1)$$

Alternative derivation:

1. The expected total number of links in the network is $\langle L \rangle = p \frac{N(N-1)}{2}$
2. The average degree is twice the number of links divided by the number of nodes:

$$\langle k \rangle = \frac{2\langle L \rangle}{N} = \frac{2 \cdot p \frac{N(N-1)}{2}}{N} = \underline{p(N-1)}$$

Example: For a random network with $N = 1000$ nodes and link probability $p = 0.01$:

- Expected average degree is $\langle k \rangle = 0.01 \times 999 \approx 10$ links per node

Random networks: degree distribution

Question: What is the probability that a node has k neighbors?

Back to coin tossing problem: What is the probability that a coin that yields heads with probability p results in k heads out of $N-1$ (independent) trials?

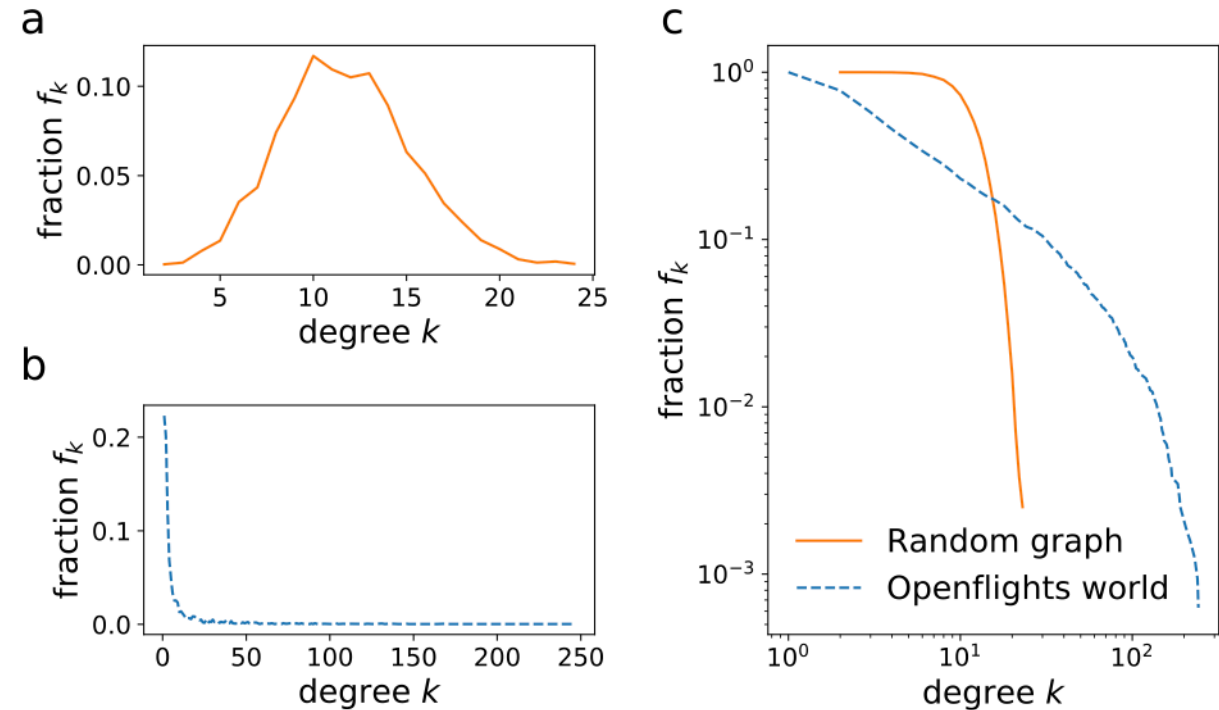
Binomial distribution:

$$P(k) = \binom{N-1}{k} p^k \underline{(1-p)^{N-1-k}}$$

For small p and large N the binomial distribution is well approximated by a bell-shaped curve
 \Rightarrow **Most degree values are concentrated around the peak, so the average degree is a good descriptor of the distribution**

Random networks: degree distribution

The degree distribution of random networks is **very different** from the broad distributions of most real-world networks!



Random networks: small-world property

Question: How many nodes are there (on average) d steps away from any node?

Premise: Since nodes have approximately the same degree, let us assume they all have the same degree k

- At distance $d = 1$ there are k nodes
- At distance $d = 2$ there are $k(k-1)$ nodes
- At distance d there are $k(k-1)^{d-1}$ nodes
- If k is not too small, the total number of nodes within a distance d from a given node is approximately:

$$N_d \sim \underline{k(k-1)^{d-1}} \sim \textcircled{k^d}$$

Random networks: small-world property

Question: how many steps does it take to cover the whole network?

$$\begin{aligned} N &\sim k^{d_{max}} \\ \log(N) &\sim d_{max} \log(k) \\ \underline{d_{max}} &\sim \frac{\log(N)}{\log(k)} \end{aligned}$$

The diameter of the network **grows like the logarithm** of the network size

Example: $N = 7,000,000,000$, $k = 150$ (Dunbar's number)

$$d_{max} = 4.52$$

Random networks: clustering coefficient

The clustering coefficient of a node i can be interpreted as the probability that two neighbors of i are connected

$$C_i = \frac{\text{number of pairs of connected neighbors of } i}{\text{number of pairs of neighbors of } i}$$

Question: what is the probability that two neighbors of a node are connected?

Answer: since links are placed independently of each other, it is the probability p that any two nodes of the graph are connected:

$$C_i = p = \frac{\langle k \rangle}{N-1} \sim \frac{\langle k \rangle}{N}$$

Since $\langle k \rangle$ is a small number, the average clustering coefficient of random networks with realistic values for $\langle k \rangle$ and N is much smaller than the ones observed in real-world networks

Random networks: summary

- Links are placed at **random**, independently of each other
- **Distances between pairs of nodes are short** (small-world property): **good!**
- The **average clustering coefficient is much lower** than on real networks of the same size and average degree: **bad!**
- The nodes have approximately the same degree; there are **no hubs**: **bad!**

Conclusion: the random network is **not a good model** for many real-world networks

In NetworkX:

```
G = nx.gnm_random_graph(N,L) # Erdős–Rényi random graph  
G = nx.gnp_random_graph(N,p) # Gilbert random graph
```

Why random network models are important

Despite their limitations in capturing real network properties, random networks remain valuable:

- **Null models:** Serve as baseline for comparison
 - "Is this network property significant or expected by chance?"
 - Statistical testing against randomized models
- **Mathematical tractability:**
 - Have analytical solutions for many properties
 - Allow rigorous proofs and predictions
- **Historical significance:**
 - First systematic approach to model networks mathematically
 - Foundation for more complex models



Small-world networks

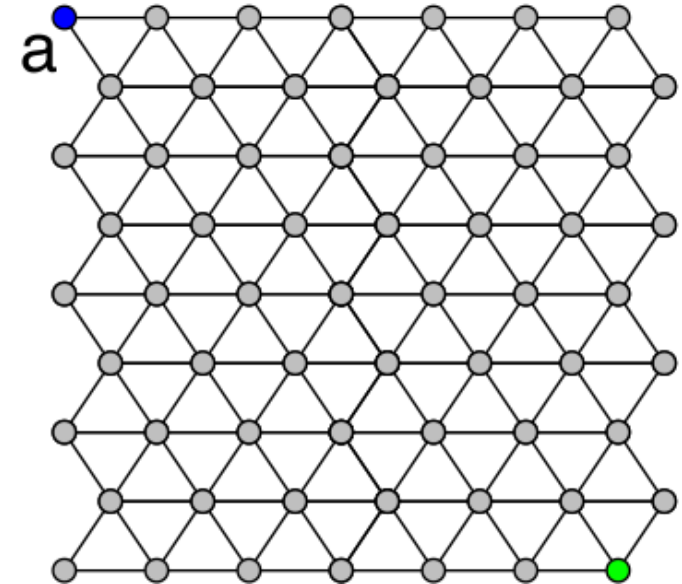
Small-world networks

Problem: Real networks often have two seemingly contradictory properties:

- **High clustering** (many triangles, like in social networks)
- **Short average path lengths** (small-world property)

Goal: Create a model that captures both properties simultaneously

Approach: Start with a highly clustered structure and introduce shortcuts

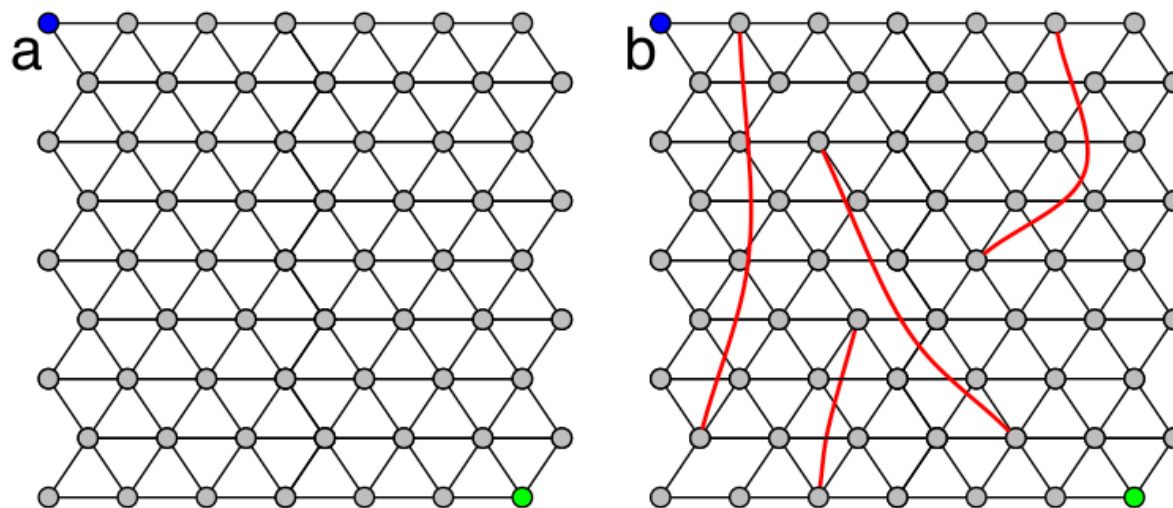


Small-world networks

Regular lattices have high clustering but long paths:

- In a regular lattice, neighbors of a node tend to be connected (high clustering)
- But getting from one side to another requires many steps (long paths)
- This doesn't match real-world networks where paths are much shorter

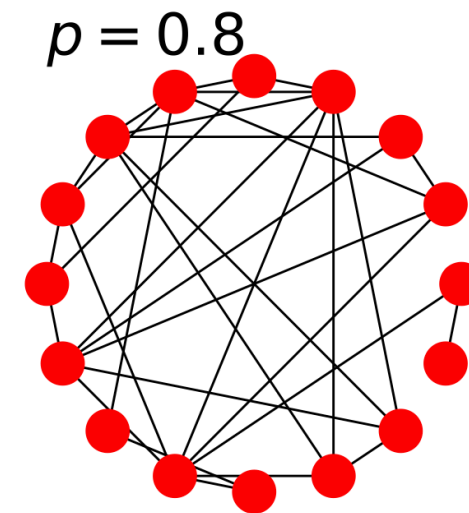
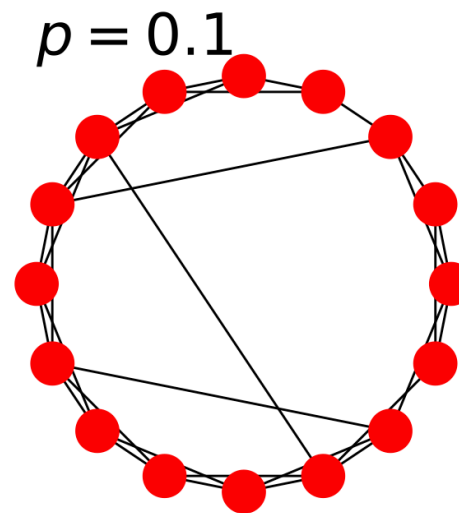
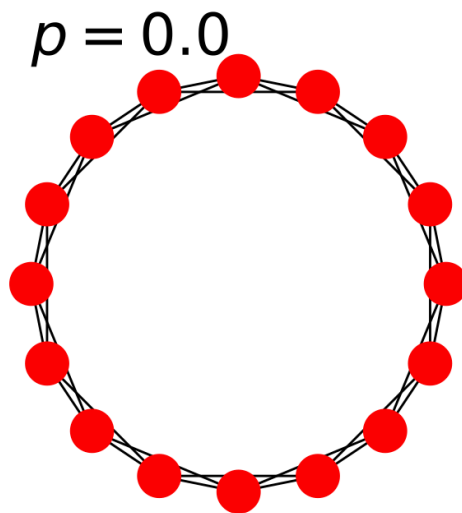
The key insight: Just a few random "shortcuts" can dramatically reduce path lengths while preserving most of the clustering



The Watts-Strogatz model

Model procedure:

1. Start with a **regular ring lattice** of N nodes, each connected to k nearest neighbors
2. For each link, with probability p , **rewire one end to a randomly chosen node**
3. Continue until all links have been considered for rewiring
4. The final network is the Watts-Strogatz model with parameters N , k , and p

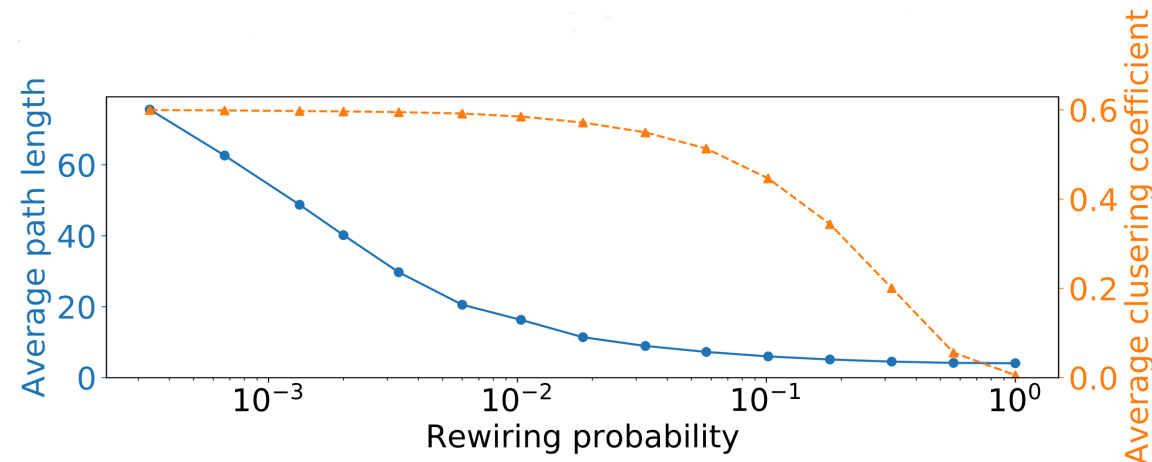


The Watts-Strogatz model: Effect of rewiring probability

The rewiring parameter p controls the network structure:

- $p = 0$: Pure regular lattice (high clustering, long paths)
- $0 < p < 0.1$: Small-world network (high clustering, short paths)
- $p = 1$: Random network (low clustering, short paths)

Key finding: Even a small p (≈ 0.01 - 0.1) creates sufficient shortcuts to drastically reduce path lengths while maintaining most triangles



The Watts-Strogatz model: Small-world region

The "sweet spot" for representing real-world networks:

- For a narrow range of p (typically $0.01 < p < 0.1$):
 - Average path length $L(p) \approx L(1) \ll L(0)$
 - Clustering coefficient $C(p) \approx C(0) \gg C(1)$

This means: Networks in this region have both:

- Short paths like random networks
- High clustering like regular lattices

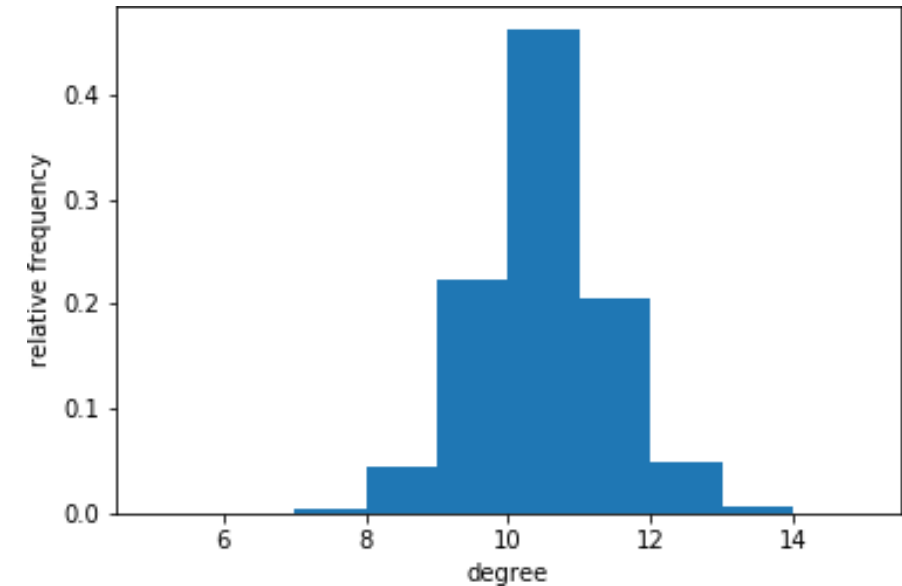
This combination matches many real-world networks like social networks, power grids, and neural networks.

[\[Example in NetLogo\]](#)

The Watts-Strogatz model: degree distribution

Limitations in representing real networks:

- The degree distribution remains narrowly distributed around k
- Most nodes have similar degrees (k or slightly different)
- **No hubs** are generated, unlike many real-world networks
- Example: $N = 10000$, $k = 10$, $p = 0.1$



The Watts-Strogatz model: summary

Strengths:

- Successfully models the coexistence of **high clustering** and **short paths**
- Explains the "small-world phenomenon" observed in many real networks
- Only needs a few random links to create efficient paths

Limitations:

- Fails to generate hubs and scale-free properties
- Starts with an artificial regular structure
- Assumes uniform rewiring across all nodes

In NetworkX:

```
G = nx.watts_strogatz_graph(N, k, p)
```



The Configuration Model

The Configuration Model

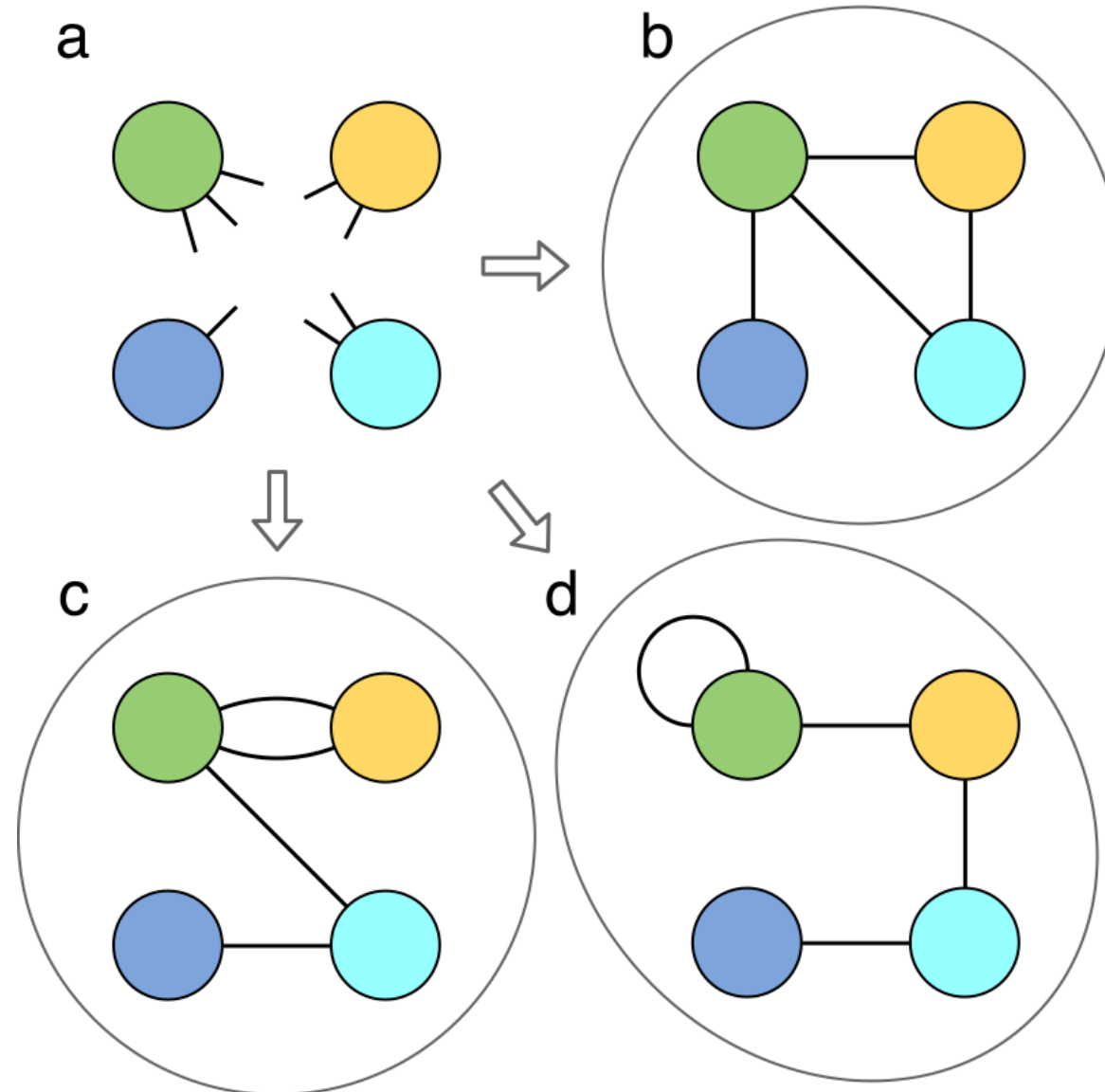
Problem: How can we build networks with a specific degree distribution?

Solution: The **configuration model** allows us to generate random networks with a prescribed degree sequence

Core concept: Instead of building networks with random connections (like Erdős-Rényi) or preferential attachment, we can directly control the degree of each node

Application areas:

- Creating null models to test hypotheses about network structure
- Studying the effect of degree distribution on network properties
- Generating synthetic networks with realistic degree distributions



The Configuration Model: Algorithm

Input: A degree sequence (k_1, k_2, \dots, k_N) where k_i is the desired degree of node i

Procedure:

1. Create N nodes, assign k_i "stubs" (half-edges) to each node i
2. Randomly pair stubs to form complete edges
3. Continue until all stubs are connected

Important constraint: The sum of degrees must be even: $\sum_{i=1}^N k_i = 2m$ (where m is the number of edges)

Note: In practice, self-loops and multi-edges may be created and are typically removed or avoided through rewiring

The Configuration Model: Properties

Random mixing: The model creates random connections while preserving the degree sequence

Clustering coefficient: Generally low (similar to random networks)

Path lengths: Similar to random networks, typically exhibiting the small-world property

Degree correlations: No inherent degree correlations (assortativity ≈ 0)

- However, constraints to avoid self-loops can introduce some disassortativity

In NetworkX:

```
# From a degree sequence
degree_sequence = [3, 3, 3, 3, 4, 4, 5, 5]
G = nx.configuration_model(degree_sequence)

# From an existing graph's degree sequence
H = nx.Graph() # some existing graph
G = nx.configuration_model(dict(H.degree()).values())
```

Degree-Preserving Randomization

Goal: Generate randomized versions of an existing network while preserving its degree sequence

Procedure:

1. Start with the original network
2. Repeatedly select two edges at random (e.g., A-B and C-D)
3. Rewire them (A-D and B-C) if no self-loops or multiple edges would result
4. Repeat many times until the network is sufficiently randomized

Applications (two of many):

- **Statistical testing of network properties:**
 - Compare properties of real network vs. randomized versions
 - If property values differ significantly, they cannot be explained by degree sequence alone
 - Example: If clustering in real network is much higher than in randomized versions, it suggests meaningful triangles beyond what's expected by chance
- **Motif significance analysis:**
 - Generate many randomized networks preserving degree sequence
 - Count occurrences of subgraphs (motifs) in real and randomized networks
 - Calculate z-scores to identify statistically over/under-represented motifs
 - Common in biological networks to identify functional circuit patterns

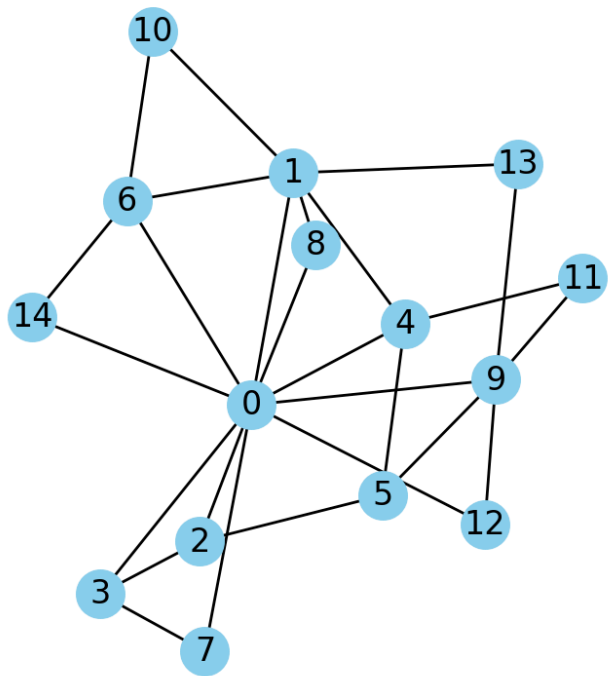
- **Null models for community detection:**

- Provides baseline expectation for modularity-based algorithms
- Helps determine if observed community structure is statistically significant

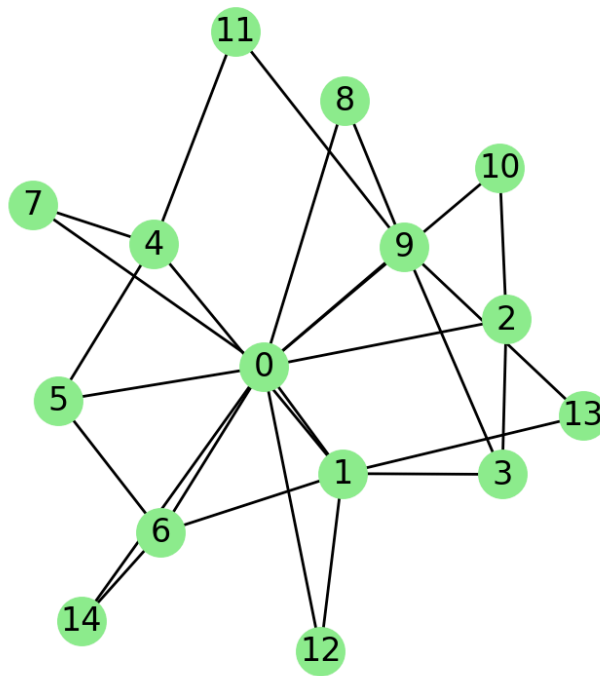
In NetworkX:

```
G_rand = nx.double_edge_swap(G, nswap=1000) # Rewire 1000 edge pairs
```

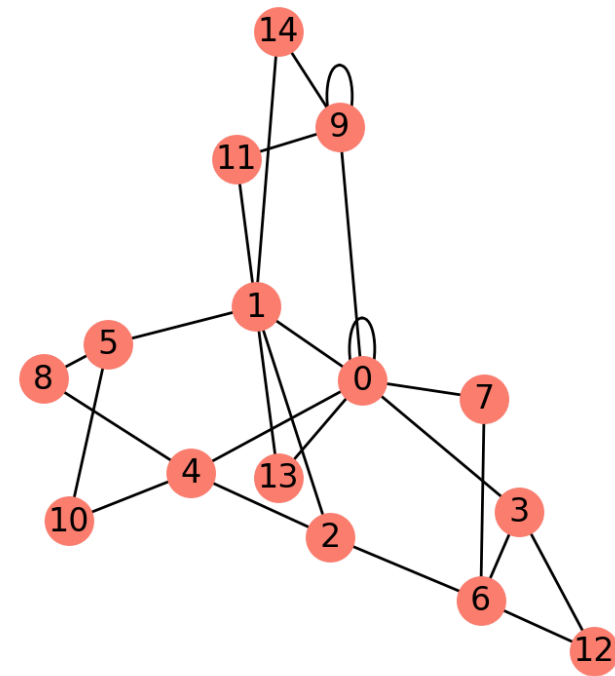
Original Graph



Double Edge Swap



Configuration Model Graph



The Configuration Model: Limitations

Self-loops and multiple edges:

- The basic model can create self-loops (edges connecting a node to itself)
- It can also create multiple edges between the same node pairs

Realizability:

- Not all degree sequences can be realized as simple graphs

Structure:

- No community structure, clustering, or other mesoscale patterns
- Used primarily as a null model against which to compare real networks



Reading material

References

[ns1] **Chapter 5 (Network Models)**

Q&A

