



Analisi e Visualizzazione delle Reti Complesse

NS08 - Network Models

Prof. Rossano Schifanella



Outline

- Random networks
- Small-world networks
- The configuration model
- Preferential attachment
- Other preferential models



Recap on structural characteristics of real networks

Features of real networks

- **small-world property**
 - Most real-world networks have **short paths**
- **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**
 - distribution of node degrees follows a power-law distribution
 - the degree of a node goes from small to very large values often spanning several orders of magnitude (heavy-tailed)
 - **presence of hubs**
 - nodes with high degree
 - **high heterogeneity parameter**
 - a measure of how broad the degree distribution is $\kappa = \frac{\langle k^2 \rangle}{\langle k \rangle^2}$



Models

Model:

A set of instructions to build networks

Goal:

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



Random networks

Random networks

Simple idea

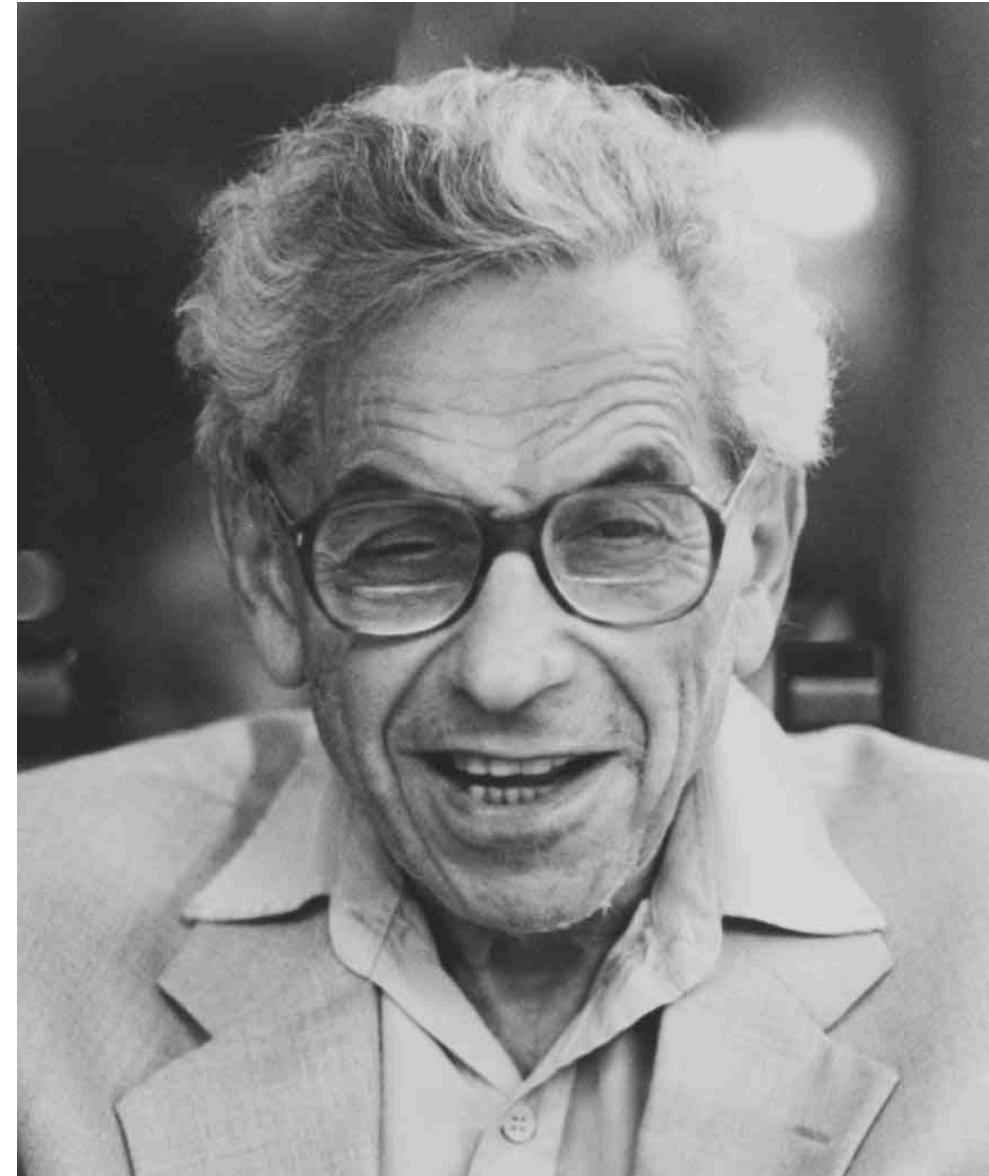
Placing links at random between pairs of nodes

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

Albert 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 $n = \binom{N}{2}$; 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 $p^m(1 - p)^{n-m}$

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:

Number of heads with probability of yielding heads equal to p and the number of trials t equal to the number of potential neighbors of a node:

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

Alternatively:

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

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?

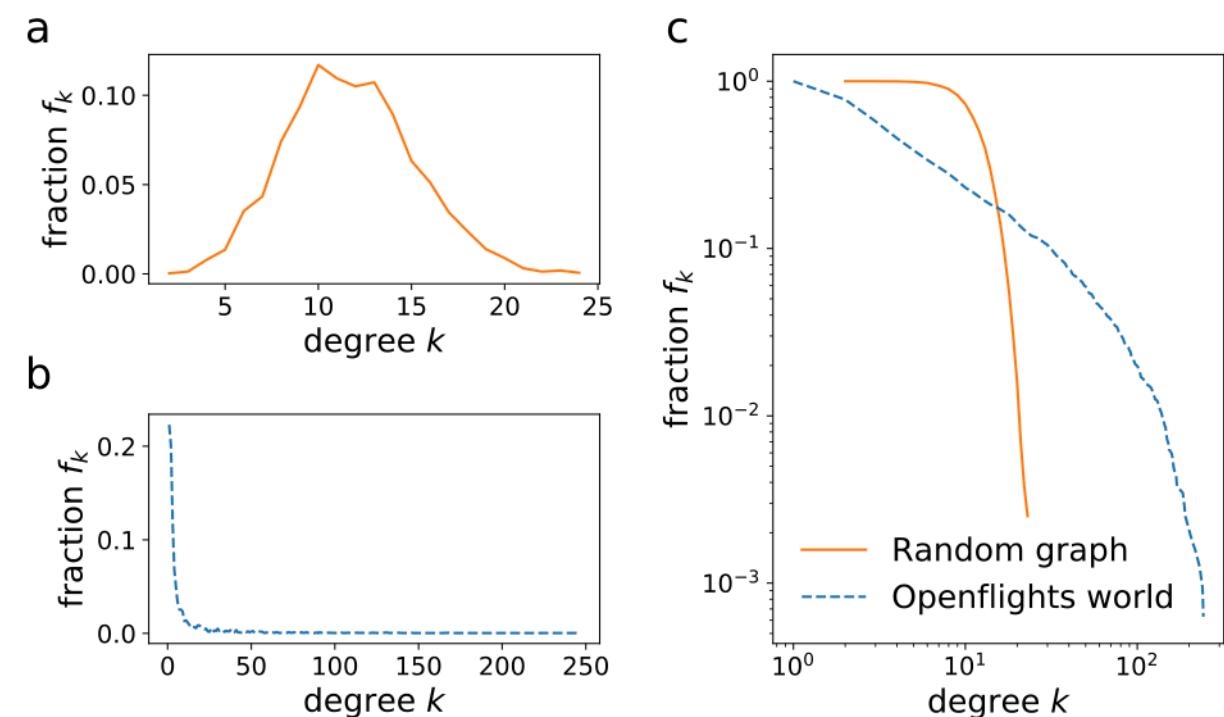
Binomial distribution:

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

For small p and large N the binomial distribution is well approximated by a bell-shaped curve
⇒ **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:

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

Random networks: small-world property

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

$$N \sim k^{d_{max}}$$

$$\log(N) \sim d_{max} \log(k)$$

$$d_{max} \sim \frac{\log(N)}{\log(k)}$$

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
```



Small-world networks

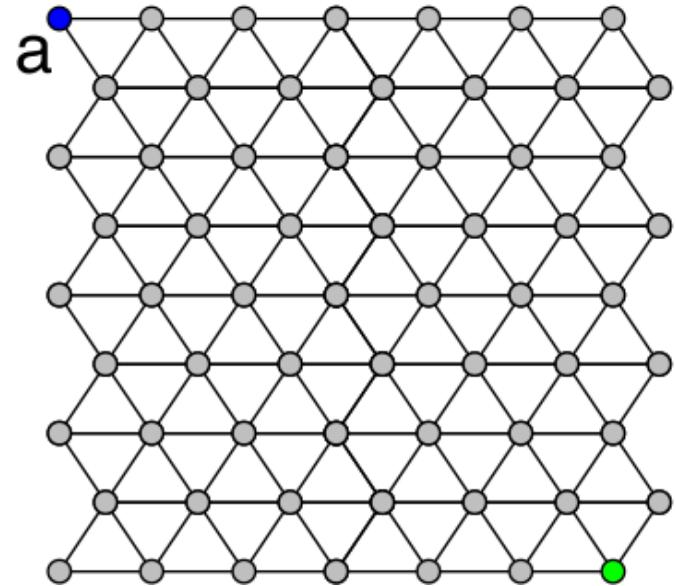
Small-world networks

Goal: building networks with the small-world property and high clustering coefficient

Solution: interpolating between a regular lattice (high clustering) and a random network (small-world property)

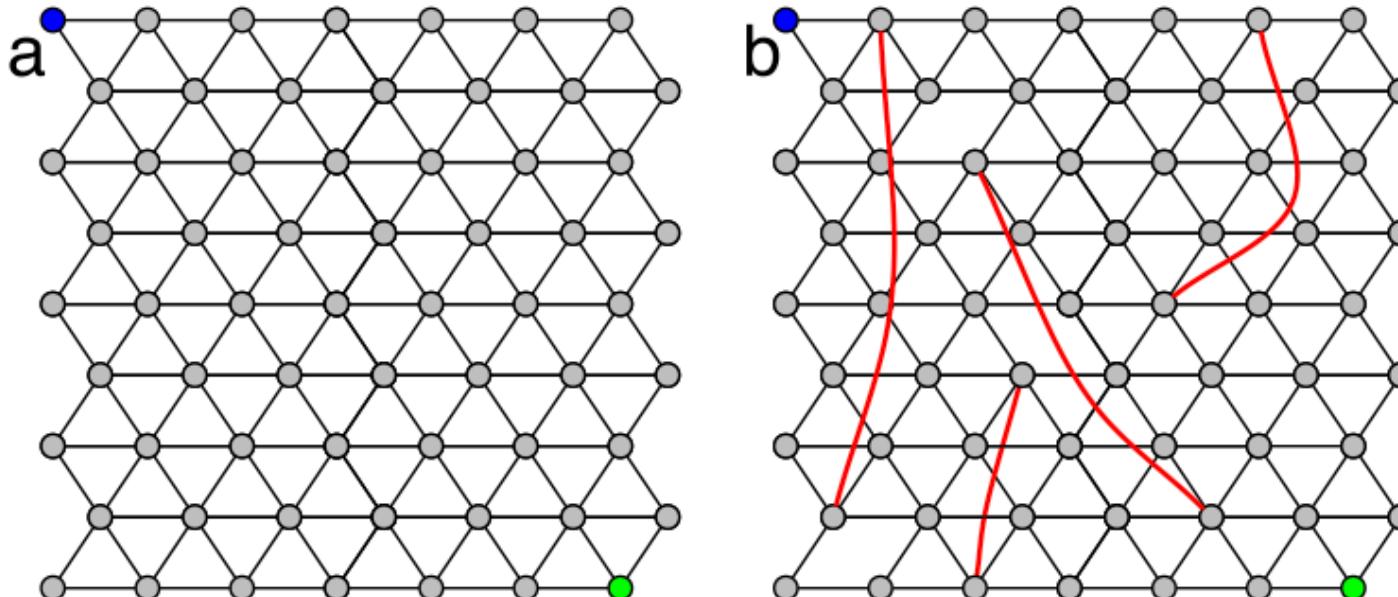
The clustering coefficient of a lattice is high:

- The internal nodes have $k = 6$ neighbors, 6 pairs of which are connected
- $C = \frac{6}{\frac{(6*5)}{2}} = \frac{6}{15} = \frac{2}{5} = 0.4$
- Most nodes are internal, so the average clustering coefficient of the network is close to 0.4!



Small-world networks

Large average shortest path length: going from one node to another can take a large number of steps, which grows rapidly with the size of the network

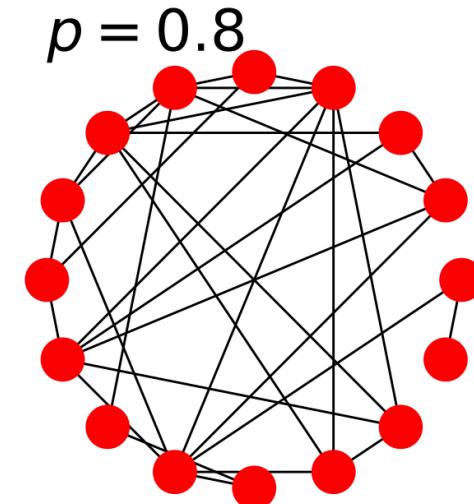
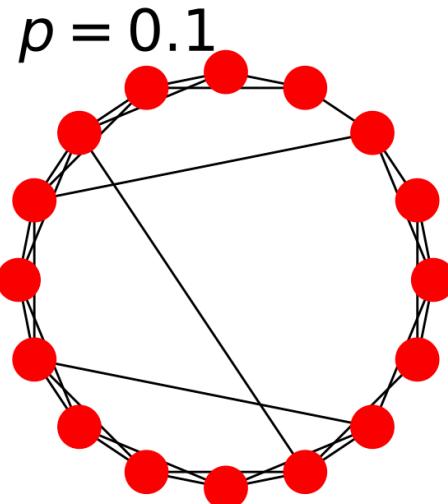
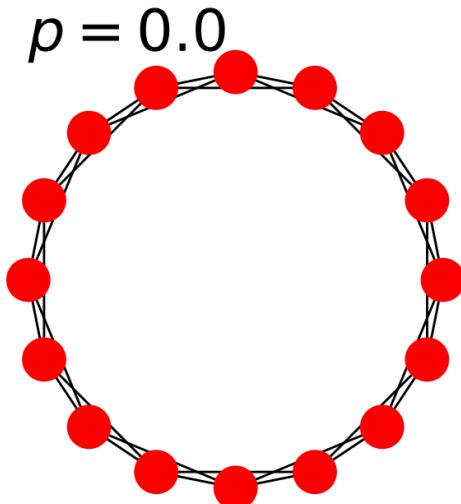


Solution: shortcuts!

The Watts-Strogatz model

N nodes form a **regular ring lattice** with degree k

With probability p , each link is rewired randomly



[Collective dynamics of 'small-world' networks]

The Watts-Strogatz model

The expected number of rewired links is $pL = pN\frac{k}{2}$

- If $p = 0$, no links are rewired: no change
- If p is small, few links are rewired: the average clustering coefficient stays approximately the same because very few triangles are destroyed, but distances shrink considerably
- If $p = 1$, all links are rewired: the network becomes a random network



The Watts-Strogatz model

Distances become short already for low values of p ;

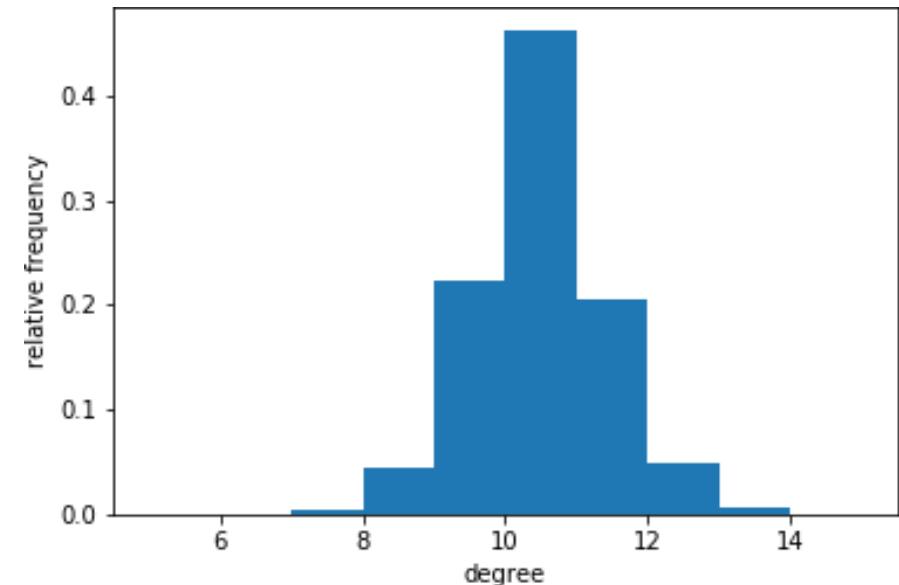
The average clustering coefficient stays high up to large values of p .

There is a range of values of p where the **average path length is short and the clustering coefficient is high!**

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

The Watts-Strogatz model: degree distribution

- The degree distribution is peaked as most nodes have the same degree: **no hubs!**
- The Watts-Strogatz model fails to reproduce the broad degree distributions observed in many real-world networks
 - Example if Figure: $N = 10000$, $k = 10$, $p = 0.1$



The Watts-Strogatz model: summary

- A regular lattice whose links are randomly rewired, with some probability p
- There is a range of values of the rewiring probability p for which distances between pairs of nodes are short (**small-world property**) and the **average clustering coefficient** is high: **good!**
- The nodes have approximately the same degree, **there are no hubs**: **bad!**

In NetworkX:

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



The Configuration Model

The Configuration Model

Problem: is it possible to build networks with a predefinite degree distribution?

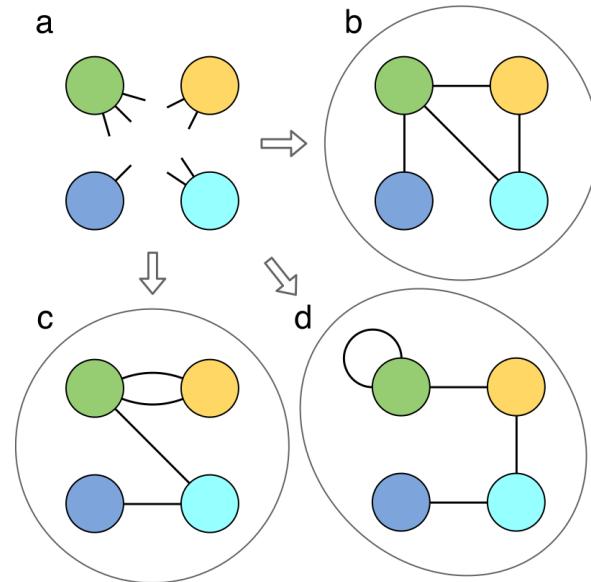
Solution: the configuration model

Focus: build networks with a predefinite degree sequence

Degree sequence: $(k_1, k_2, \dots, k_{N-1}, k_N)$

Caveat: many degree sequences can be extracted from the same distribution

Principle: assign a degree to each node (e.g., from the desired distribution or a real network), place as many **stubs** on each node as the degree of the node, and **attach pairs of stubs at random**



The Configuration Model

Degree-preserving randomization: generate randomized versions of a given network with the same degree sequence, using the configuration model

Why: useful to see whether **a specific property** of the original network **is determined by its degree distribution alone**

- If the property is maintained in the randomized configurations, then the degree distribution is the main driver
- If the property is lost in the randomized configurations, other factors must be responsible

In NetworkX:

```
G = nx.configuration_model(D)      # network with degree sequence D
```



Preferential models

Network growth

Note: real-world networks are **dynamic!**

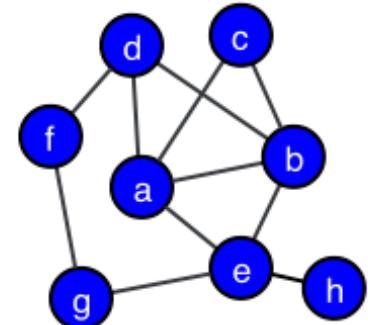
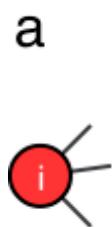
Examples:

- The Web in 1991 had a single node, today there are trillions
- Citation networks of scientific articles and collaboration networks of scientists keep growing due to the publication of new papers
- The collaboration network of actors keeps growing due to the release of new movies
- The protein interaction network has been growing for over 4 billion years, from a few genes to over 20,000

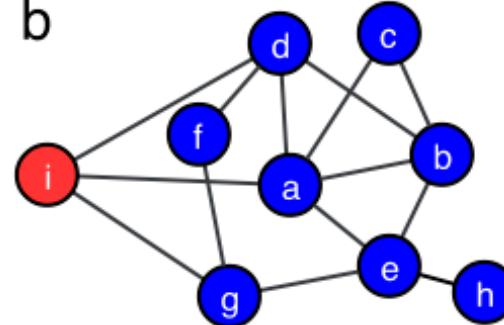
Network growth

General procedure:

1. A new node comes with a given number of **stubs**, indicating the number of future neighbors of the node (degree)
2. The **stubs are attached** to some of the old nodes, according to some rule



b



Preferential attachment

Note: Nodes prefer to link to the more connected nodes

Examples:

- Our knowledge of the Web is biased towards **popular pages**, which are highly linked, so it is more likely that our website points to highly linked Web sites
- Scientists are more familiar with **highly cited papers** (which are often the most important ones), so they will tend to cite them more often than poorly cited ones in their own papers
- The more movies an actor makes, the more popular they get and the higher the chances of being cast in a new movie

Which model?

Our network model should have the following features:

- **Growth: The number of nodes grows** in time after adding new nodes.
 - The models considered so far are static.
- **Preferential attachment: new nodes tend to be connected to the more connected nodes.**
 - The models considered so far set links among pairs of random nodes, regardless of their degree

"For to every one who has will more be given, and he will have abundance; but from him who has not, even what he has will be taken away" — Gospel of Matthew 25:29

Take-home message: the rich get richer, and the poor get poorer

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

Polya's urn model

Start: an urn contains X white and Y black balls

Process: a ball is drawn from the urn and put back in with another ball of the same color

Example: if we first pick a white ball, there will be $X + 1$ white and Y black balls in the urn; white will become more likely to be picked than black in the future

- Preferential attachment used to explain heavy-tail distributions of many quantities:
 - the number of species per genus of flowering plants,
 - the number of (distinct) words in a text,
 - the populations of cities,
 - individual wealth,
 - scientific production,
 - citation statistics, firm size, and many others!

The Barabási-Albert model

Procedure:

- Start with a group of m_0 nodes, usually fully connected (clique)
- At each step a new node i is added to the system, and sets m links with older nodes ($m \leq m_0$)
- The probability that the new node i chooses an older node j as a neighbor is **proportional to the degree k_j of j :**

$$\prod(i \leftrightarrow j) = \frac{k_j}{\sum_l k_l}$$

- The procedure ends when the given number N of nodes is reached

The Barabási-Albert model

Example: if t has to choose between node i , with degree 6, and node j , with degree 3, the probability of choosing i is twice the probability of choosing j



The Barabási-Albert model

- **Rich-gets-richer phenomenon:** due to preferential attachment, the more connected nodes have higher chances to acquire new links, which gives them a bigger and bigger advantage over the other nodes in the future
- **This is how hubs are generated**

In NetworkX;

```
G = nx.barabasi_albert_graph(N, m) # BA model network
```



The Barabási-Albert model

Hubs are the oldest nodes: they get the initial links and acquire an advantage over the other nodes, which increases via preferential attachment

Question: if old nodes have an advantage over newer nodes anyway, do we need preferential attachment at all? Can we explain the existence of hubs just because of growth?

Alternative model: each new node chooses its neighbors at random, not with probability proportional to their degree

The Barabási-Albert model

Conclusion: growth + random attachment does not generate hubs.

Preferential attachment is necessary!



Other preferential models

- The Barabási-Albert model uses **linear preferential attachment**: the linking probability is proportional to the degree
- **Question:** what happens if the linking probability is proportional to a power of the degree?
- **Non-linear preferential attachment!**

Tutorial on network models:

[04-network-models.ipynb](#) in the course GitHub repo.

Picking nodes in Python

Question: how do we pick nodes with a given probability?

Answer: use the random module

Example: picking list elements (e.g., nodes) with the same probability, i.e., at random:

```
import random
nodes  = [1 ,2, 3, 4]
selected_node = random.choice(nodes)
```

Picking nodes in Python

Question: what if nodes have to be picked with different probabilities?

Answer: we need to provide a list whose elements are the weights associated with the nodes

Note: weights are used to calculate probabilities, they do not have to be integers or add up to one

Example: picking nodes with probability proportional to their degrees, as in preferential attachment:

```
import random
nodes = [1, 2, 3, 4]
degrees = [3, 1, 2, 2]
selected_node = random.choice(nodes, degrees)
```



Final remarks and a case study

Final Remarks

- Watts-Strogatz model explains **short paths** and **high clustering**, but fails to get how individuals actually find short paths from only a local perspective
 - Kleinberg's decentralized search (coming soon)
- Barabasi-Albert model generates **scale-free networks**, i.e., degree distribution following power law with exponent $2 \leq \alpha \leq 3$
 - analytical analysis of rich get richer process (coming soon)
- Are random network models realistic? Why do we use them?



Other preferential models

Non-linear preferential attachment

Procedure:

- Start with a group of m_0 nodes, usually fully connected (clique)
- At each step a new node i is added to the system, and sets m links with older nodes ($m \leq m_0$)
- The probability that the new node i chooses an older node j as neighbor is **proportional to the power α of the degree k_j of j :**

$$\prod(i \leftrightarrow j) = \frac{k_j^\alpha}{\sum_l k_l^\alpha}$$

- The procedure ends when the given number N of nodes is reached

Non-linear preferential attachment

$$\prod(i \leftrightarrow j) = \frac{k_j^\alpha}{\sum_l k_l^\alpha}$$

For $\alpha = 1$ we recover the linear preferential attachment (BA model)

Question: what happens when $\alpha \neq 1$?

Answer: it depends on whether $\alpha > 1$ or $\alpha < 1$

Non-linear preferential attachment

- For $\alpha < 1$, the link probability does not grow fast enough with degree, so the advantage of high-degree nodes over the others is not as big.
 - As a result, the degree distribution does not have a heavy tail: **the hubs disappear!**
- If $\alpha > 1$, high-degree nodes accumulate new links much faster than low-degree nodes.
 - As a consequence, one of the nodes will end up being connected to a fraction of all other nodes.
 - For $\alpha > 2$, a single node may be connected to all other nodes (**winner-takes-all effect**), all other nodes having low degree
- **Conclusion: non-linear preferential attachment fails to generate hubs**
 - Linear preferential attachment is the only way to go
- **Problem: strict proportionality of linking probability to degree appears unrealistic**

Limits of preferential attachment

- It yields a **fixed pattern for the degree distribution**: the slope is the same for any choice of the model parameters.
 - Degree distributions in real-world networks could decay faster or more slowly
- **The hubs are the oldest nodes**: new nodes cannot overcome their degree
- **It does not create many triangles**: the average clustering coefficient is much lower than in many real-world networks
- **Nodes and links are only added**: in real networks they can also be deleted
- Since each node is attached to older nodes, **the network consists of a single connected component**.
 - Many real-world networks have **multiple components**

Extensions of the BA model: Attractiveness model

- **Pitfall of preferential attachment:** What happens if a node has no neighbors (degree zero)?
 - It will never get connections from other nodes!
- **No problem for standard initial condition:** the initial subgraph is complete (clique), so every node has nonzero degree
- **What if the network is directed and the linking probability is proportional to the in-degree?**
 - Bad, as each new node has in-degree zero, so it will never be linked by future nodes!

Extensions of the BA model: Attractiveness model

Procedure:

- Start with a group of m_0 nodes, usually fully connected (clique)
- At each step a new node i is added to the system, and sets m links with some of the older nodes ($m < m_0$)
- The probability that the new node i chooses an older node j as neighbor is proportional to the sum of the degree k_j of j and an attractiveness A , indicating the intrinsic appeal of j :

$$\prod(i \leftrightarrow j) = \frac{A + k_j}{\sum_l(A + k_l)}$$

Extensions of the BA model: Attractiveness model

$$\prod(i \leftrightarrow j) = \frac{A + k_j}{\sum_l(A + k_l)}$$

- For $A = 0$ we recover the BA model
- For every value of A we get networks with heavy-tailed degree distributions
- The pattern of the distribution **changes with A** , so it is possible to match distributions of real-world networks, **unlike the BA model**

Extensions of the BA model: Fitness model

Pitfall of preferential attachment: the hubs are the oldest nodes. **Unrealistic!**

Examples:

- In the Web, new pages can overrun old pages (e.g., Google!)
- In science, new papers can be more successful than (many) old papers

Reason: each node has its own individual appeal

Extensions of the BA model: Fitness model

Procedure:

- Start with a group of m_0 nodes, usually fully connected (clique)
- At each step a new node i is added to the system, and sets m links with some of the older nodes ($m \leq m_0$)
- The probability that the new node i chooses an older node j as neighbor is proportional to the product of the degree k_j of j with a fitness η_j , indicating the intrinsic appeal of j :

$$\prod(i \leftrightarrow j) = \frac{\eta_j \cdot k_j}{\sum_l(\eta_l \cdot k_l)}$$

Extensions of the BA model: Fitness model

The fitness values are extracted from a distribution $\rho(\eta)$ and assigned to each new node

Difference with attractiveness model

- The fitness enters as a **factor** in the link probability, not as a summand
- The fitness is **characteristic of each node**, it is not a constant

Extensions of the BA model: Fitness model

Results:

- If the fitness distribution $\rho(\eta)$ has finite support, i.e., the fitness is distributed over a finite range of values, the degree distribution of the network is heavy-tailed
- If the fitness distribution $\rho(\eta)$ has infinite support, i.e., the fitness is distributed over an infinite range of values, the node with the largest fitness attracts a fraction of all links (**monopoly**)
- Nodes with large fitness can acquire a large degree even if they are introduced late in the system (**good!**)

Extensions of the BA model: Random walk model

- **Pitfall of preferential attachment:** the BA model does not generate many triangles. Why?
- To close a triangle we need to set a link between two neighboring nodes, whereas in the BA model links are set based on degree, regardless of whether the future neighbors have common neighbors

Solution: introduce a mechanism for triadic closure in the model!

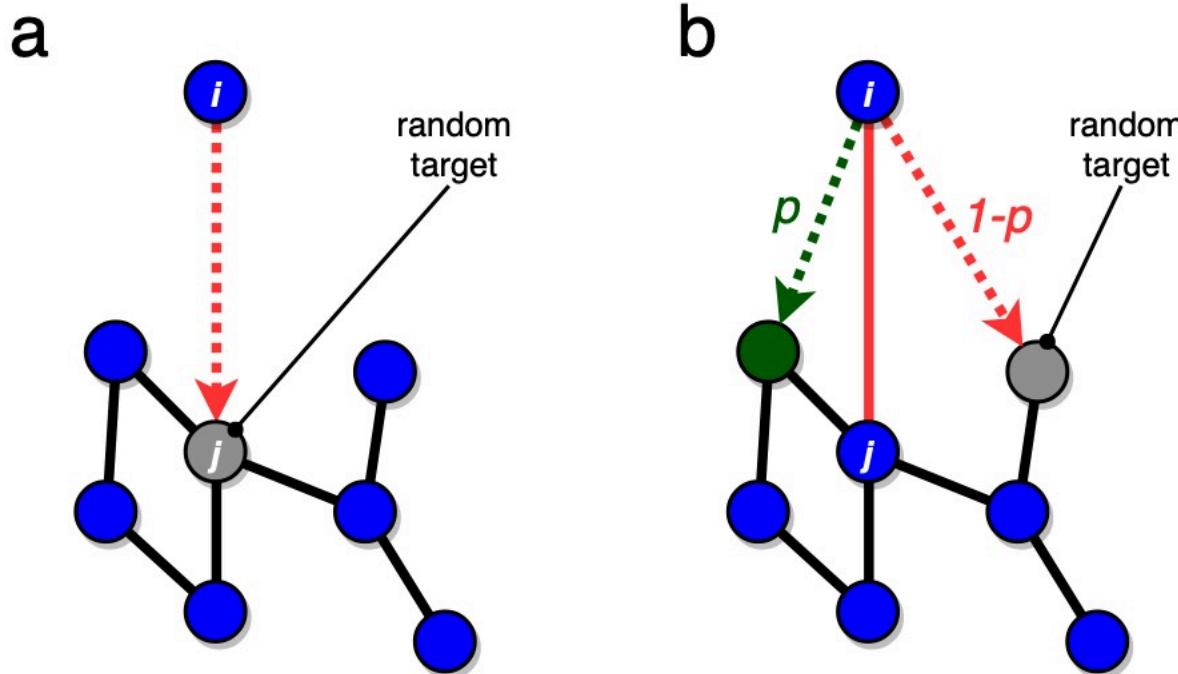


Extensions of the BA model: Random walk model

Procedure:

- Start with a group of m_0 nodes, usually fully connected (clique)
- At each step a new node i is added to the system, and sets m links with some of the older nodes ($1 < m \leq m_0$)
- The first link targets a randomly chosen node j
- From the second link onwards:
 - With probability p the link is set with a neighbor of j , chosen at random
 - With probability $1-p$ the link is set with a randomly chosen node

Extensions of the BA model: Random walk model



Extensions of the BA model: Random walk model

Results:

- The **degree distribution is heavy-tailed**
- The **average clustering coefficient is much higher than in BA networks**
 - the larger, the greater the probability p of triadic closure
- When the triadic closure probability p is sufficiently high that many triangles are formed ($p \approx 1$) the network has **community structure**
 - i.e., it is made of cohesive groups of nodes, loosely connected to each other

Extensions of the BA model: Random walk model

Question:

- If links are set at random, as it seems, how can the model generate hubs?

Answer:

- Choosing a random node and a random neighbor of the node amounts to choosing a link at random
- The probability that the endpoint(s) of a randomly selected link have a given degree is proportional to the degree

Extensions of the BA model: Random walk model

Conclusion:

The triadic closure mechanism of the random walk model **induces effective preferential attachment**

Take-home messages:

- Preferential attachment can be induced by simple mechanisms based on random choices;
- **it is not necessary to require the knowledge of the degree of the nodes, nor a strict expression of the link probability**

Extensions of the BA model: Copy model

Motivation: the authors of a new web page tend to copy the hyperlinks of other pages on the same topic

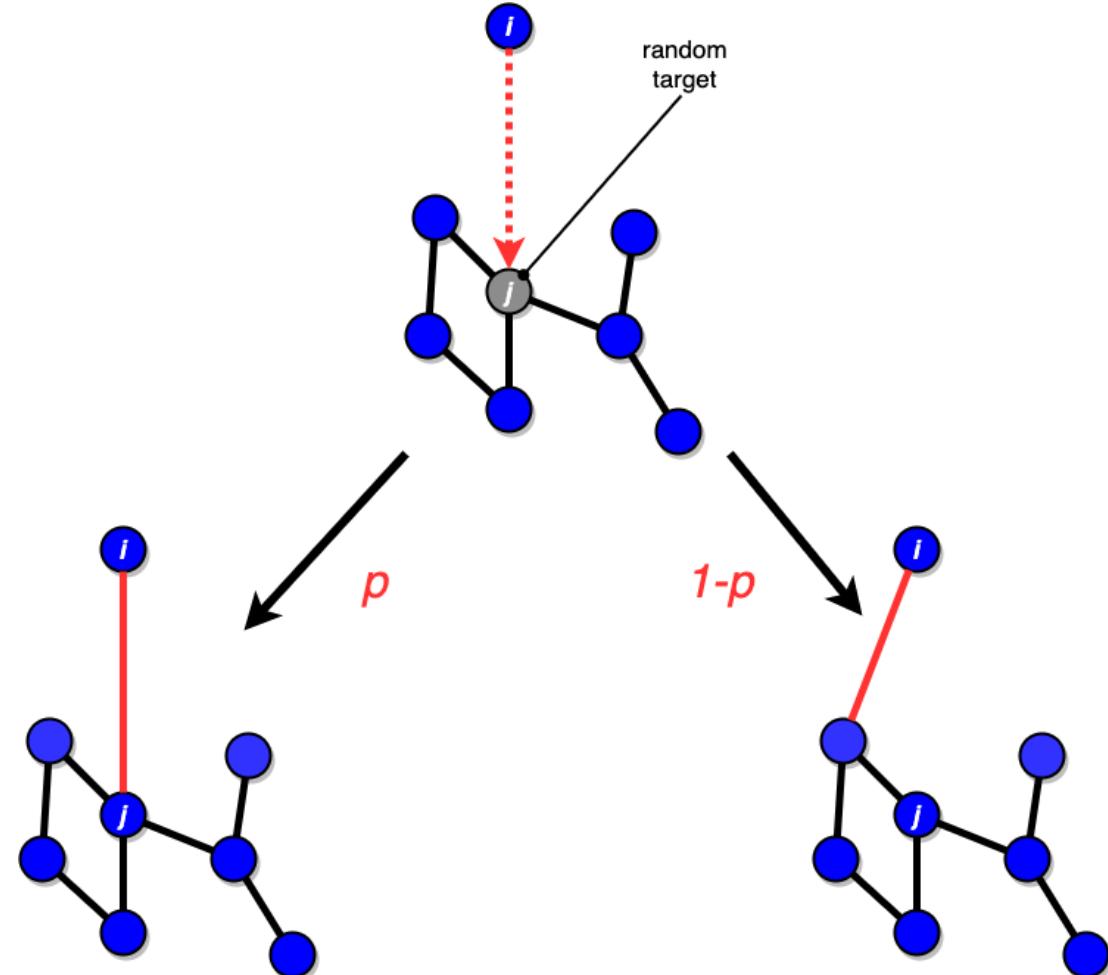
Steps:

- **Growth:** at each time step a new node i is added to the network
- **Target selection:** a node j is selected at random
- **Random connection:** with probability p the new node is connected to j
- **Link copying:** with probability $1-p$ the new node is connected to a neighbor of j , chosen at random

Copy model

Difference from random walk model:

here we do not link to the target and to its neighbor (no triadic closure)!



Extensions of the BA model: Copy model

Relevance to real systems:

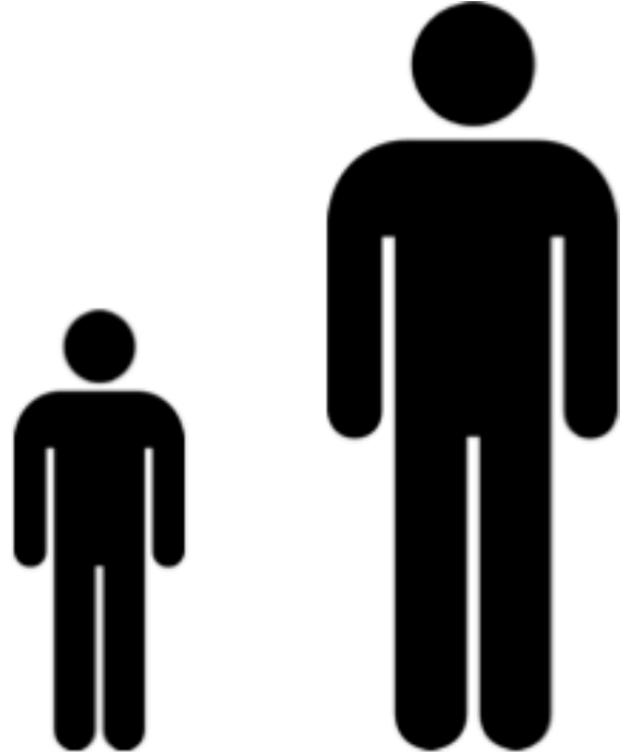
- **Social Networks:** The more acquaintances an individual has, the higher is the chance that she will be introduced to new individuals by her existing acquaintances (we "copy" the friends of our friends)
- **Citation Networks:** Authors decide what to read and cite by "copying" references from the papers they have read. Consequently papers with more citations are more likely to be studied and cited again
- **Gene duplication:** Responsible for the emergence of new genes in a cell, can be mapped into the copying model

Extensions of the BA model: Rank model

Pitfall of preferential attachment: BA model implies that nodes have a perception of how important other nodes are, i.e., how large is their degree

Objection: in the real world there is no such perception of the absolute value of things, it is far easier to perceive the relative value

Solution: ranking



Extensions of the BA model: Rank model

Procedure:

- Nodes are ranked based on a property of interest (e.g., age, degree). The rank of node i is R_i
- Start with a group of m_0 nodes, usually fully connected (clique)
- At each step a new node i is added to the system, and sets m links with some of the older nodes ($m \leq m_0$)
- The probability that the new node i chooses an older node j as neighbor is **proportional to a power of the rank of j** :

$$\prod (i \leftrightarrow j) = \frac{R_j^{-\alpha}}{\sum_l R_l^{-\alpha}}$$

Extensions of the BA model: Rank model

$$\prod(i \leftrightarrow j) = \frac{R_j^{-\alpha}}{\sum_l R_l^{-\alpha}}$$

Remark: highly-ranked nodes (those with low values of R) have high probabilities of being linked, much higher than poorly-ranked nodes

Result: the model generates networks with hubs for **any value** of the exponent α and **any property** used to rank the nodes



Reading material

References

[ns1] Chapter 5 (Network Models)



Q & A

