

1

UNIVERSITAT DE BARCELONA

Complex Networks

Albert Díaz Guilera

http://diaz-guilera.net

@anduviera

complexity lab barcelona

COMPLEXITAT

2

UNIVERSITAT DE BARCELONA

Models describing simple properties of Complex Networks

Albert Díaz Guilera
http://diaz-guilera.net
@anduviera

complexity lab barcelona

COMPLEXITAT

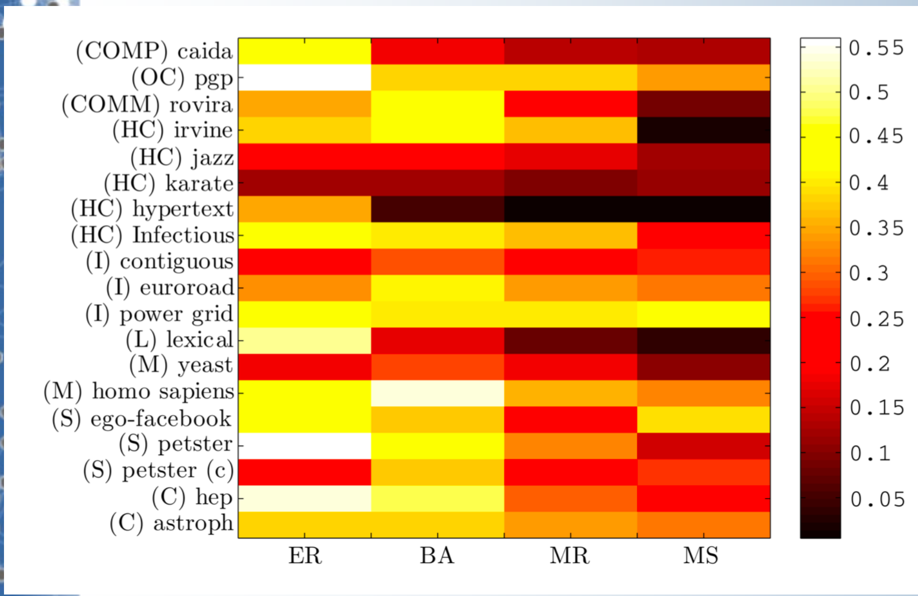
3

Why models?

- Generative models to understand real data
- Looking for vulnerabilities not available in the data

4

 Albert Díaz Guilera



5

 Albert Díaz Guilera

1. Erdos-Renyi: random graph model

- Definition: N labeled nodes connected by n links which are chosen randomly from the N(N-1)/2 possible links
- There are $\binom{N(N-1)/2}{n}$ graphs with N nodes and n links

6

Alternative definition

- Binomial model: start with N nodes, every pair of nodes being connected with probability p
- The expected total number of links, L, is a random variable
  - E(L)=pN(N-1)/2

7

Big Data <-> Big Networks — Albert Díaz Guilera

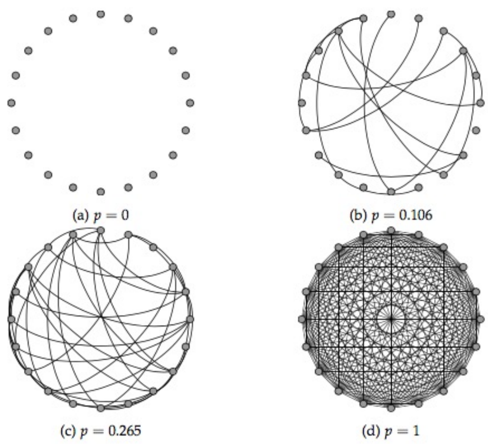


Figure 11.1: Erdös–Rényi random networks for different probabilities p

8

Mean connectivity

- <k>=pN
- If $p \propto N^{-1}$ then <k> is a constant
- If 0< <k> < 1 almost surely all clusters are either trees or clusters containing exactly one cycle
- At <k>=1 the structure changes abruptly. Cycles appear and a giant cluster develops

9

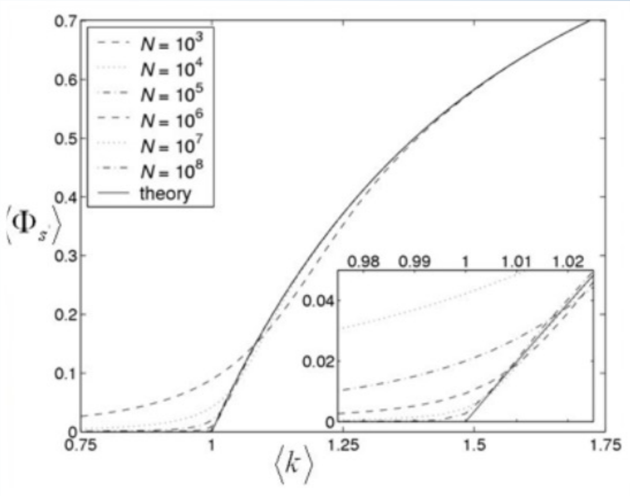
Defining Random Networks

There are two definitions of a random network:

- $G(N, L)$ Model: N labeled nodes are connected with L randomly placed links. Erdős and Rényi used this definition in their string of papers on random networks [2-9]
- $G(N, p)$ Model: Each pair of N labeled nodes is connected with probability p, a model introduced by Gilbert [10].

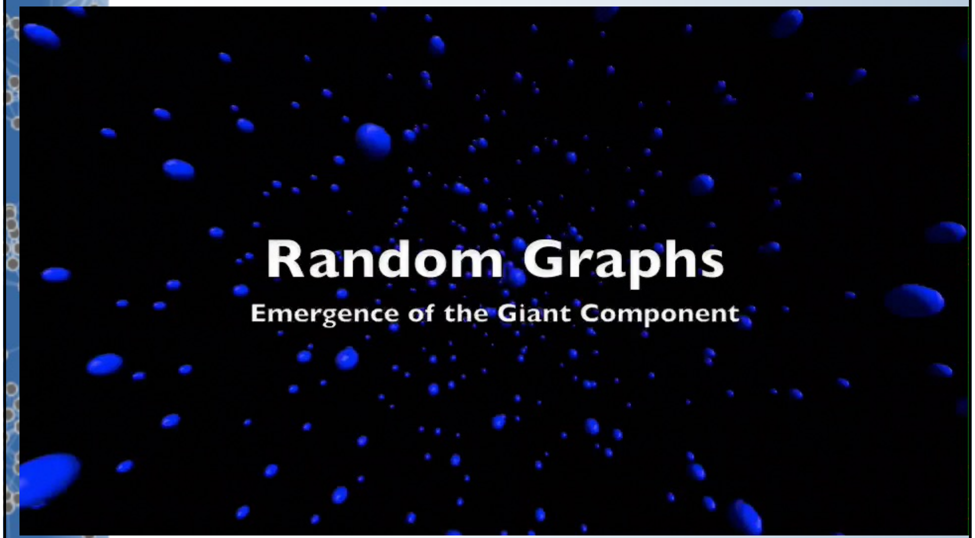
Hence, the $G(N, p)$ model fixes the probability p that two nodes are connected and the $G(N, L)$) model fixes the total number of links L. While in the $G(N, L)$ model the average degree of a node is simply $\langle k\rangle = 2L/N$, other network characteristics are easier to calculate in the $G(N, p)$ model. Throughout this book we will explore the $G(N, p)$ model, not only for the ease that it allows us to calculate key network characteristics, but also because in real networks the number of links rarely stays fixed.

10



11

Big Data <-> Big Networks Albert Díaz Guilera



12

 Albert Díaz Guilera

Degree distribution

- The degree of a node follows a binomial distribution (in a random graph with p)

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

- Probability that a given node has a connectivity k
- For large N, Poisson distribution

$$P(k) \approx e^{-pN} \frac{(pN)^k}{k!} = e^{-<k>} \frac{<k>^k}{k!}$$

13

$$p_k = \binom{N-1}{k} p^k (1-p)^{N-1-k} \qquad (3.22)$$

that characterizes a random graph. We rewrite the first term on the r.h.s. as

$$\binom{N-1}{k} = \frac{(N-1)(N-1-1)(N-1-2)\ldots(N-1-k+1)}{k!} \approx \frac{(N-1)^k}{k!} \qquad (3.23)$$

where in the last term we used that $k \ll N$. The last term of (3.22) can be simplified as

$$\ln\left[(1-p)^{(N-1)-k}\right] = (N-1-k)\ln\left(1 - \frac{\langle k \rangle}{N-1}\right)$$

and using the series expansion

$$\ln(1+x) = \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n} x^n = x - \frac{x^2}{2} + \frac{x^3}{3} - \ldots, \forall |x| \leq 1$$

we obtain

$$\ln\left[(1-p)^{(N-1)-k}\right] \approx (N-1-k)\frac{\langle k \rangle}{N-1} = -\langle k \rangle \left(1 - \frac{k}{N-1}\right) \approx -\langle k \rangle$$

which is valid if $N \gg k$. This represents the *small degree approximation* at the heart of this derivation. Therefore the last term of (3.22) becomes

$$(1-p)^{N-1-k} = e^{-\langle k \rangle} \qquad (3.24)$$

Combining (3.22), (3.23), and (3.24) we obtain the Poisson form of the degree distribution

$$p_k = \binom{N-1}{k} p^k (1-p)^{(N-1)-k} = \frac{(N-1)^k}{k!} p^k e^{-\langle k \rangle} = \frac{(N-1)^k}{k!} \left(\frac{\langle k \rangle}{N-1}\right)^k e^{-\langle k \rangle}$$

or

$$p_k = e^{-\langle k \rangle} \frac{\langle k \rangle^k}{k!} \qquad (3.25)$$

14

Mean short path

- Assume that the graph is homogeneous
- The number of nodes at distance l are $<k>^l$
- How to reach the rest of the nodes?
- l_{rand} to reach all nodes => $k^l=N$

$$l_{\mathrm{rand}} \approx \frac{\ln N}{\ln <k>} \approx \frac{\ln N}{\ln pN}$$

15

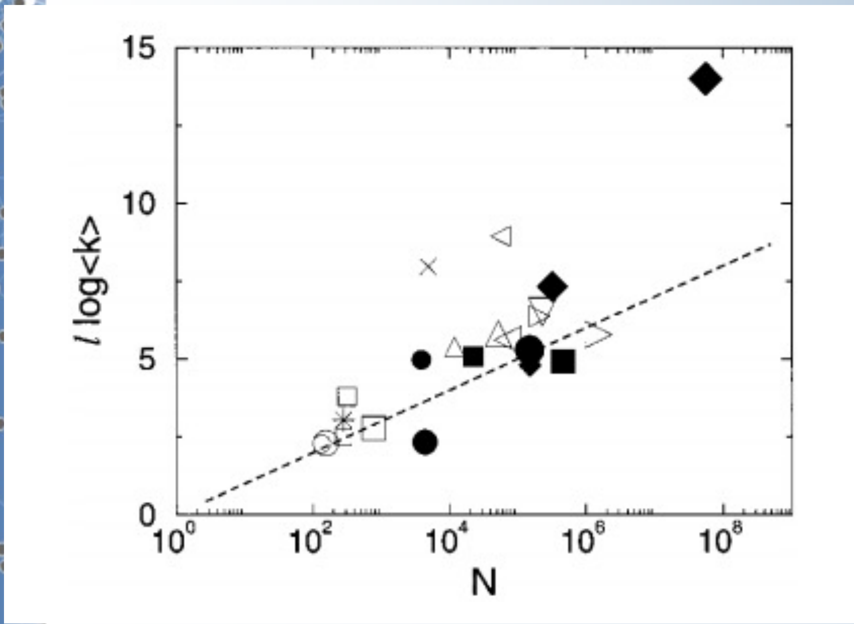
Big Data <-> Big Networks
Albert Díaz Guilera

The average path length for large n is

$$\bar{l}(H) = \frac{\ln n - \gamma}{\ln(pn)} + \frac{1}{2},$$

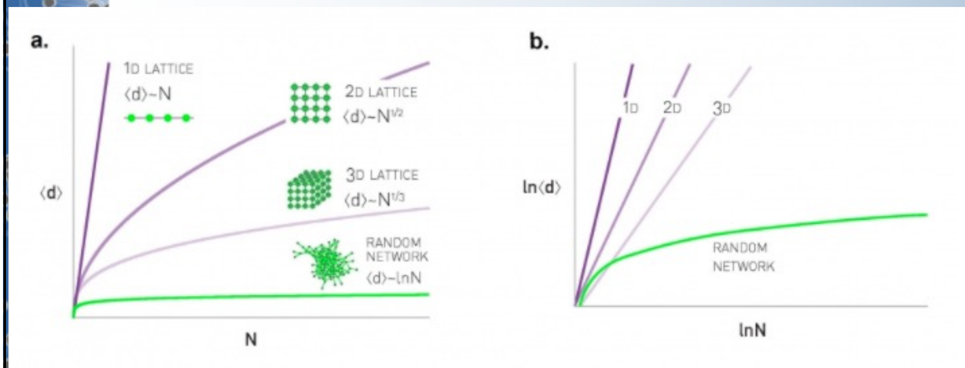
where $\gamma \approx 0.577$ is the Euler–Mascheroni constant.

16



15
10
5
0
l log<k>
10^0
10^2
10^4
10^6
10^8
N

17

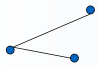


a.
1D LATTICE
⟨d⟩~N
2D LATTICE
⟨d⟩~N^{1/2}
3D LATTICE
⟨d⟩~N^{1/3}
RANDOM NETWORK
⟨d⟩~lnN
⟨d⟩
N
b.
1D
2D
3D
ln⟨d⟩
RANDOM NETWORK
lnN

18

Clustering coefficient

- Probability that two nodes are connected (given that they are connected to a third)?



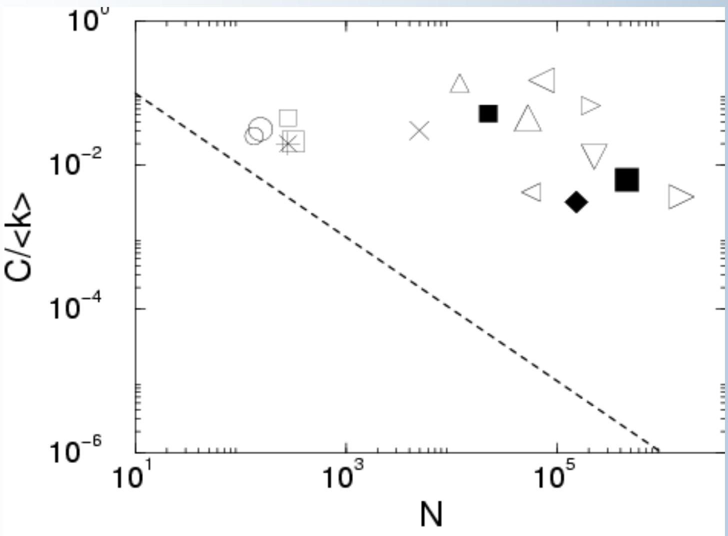
$$C_{rand} = p = \frac{<k>}{N}$$

$$\frac{C_{rand}}{<k>} \approx \frac{1}{N}$$

while it is constant for real networks

19

Big Data <-> Big Networks Albert Díaz Guilera



20

Algorithm 4.17 Erdös and Rényi (ER). Version $G^{ER}_{N,K}$

Input: V set of nodes (of size N), and number of links K .
Output: $\mathcal{K}$ set of links

```
K ← ∅
while size(K) < K do
    Choose at random a pair of nodes i ∈ V and j ∈ V
    if the edge (i, j) ∉ K then
        add link (i, j) to K
    end if
end while
return K
```

Algorithm 4.18 Erdös and Rényi (ER). Version $G^{ER}_{N,p}$

Input: V set of nodes (of size N), and p, the probability to connect any two pair of nodes.
Output: $\mathcal{K}$ set of links

```
size of K = 0
for all Node i ∈ V do
    for all Node j ∈ V, with j > i do
        Generate a random number r
        if r ≤ p then
            add link (i, j) to K
        end if
    end for
end for
return K
```

21

Albert Díaz Guilera

Random Graphs

Generators for random graphs.

fast_gnp_random_graph(n, p[, seed, directed])	Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
gnp_random_graph(n, p[, seed, directed])	Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
dense_gnm_random_graph(n, m[, seed])	Return the random graph G_{n,m}.
gnm_random_graph(n, m[, seed, directed])	Return the random graph G_{n,m}.
erdos_renyi_graph(n, p[, seed, directed])	Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
binomial_graph(n, p[, seed, directed])	Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
newman_watts_strogatz_graph(n, k, p[, seed])	Return a Newman-Watts-Strogatz small world graph.
watts_strogatz_graph(n, k, p[, seed])	Return a Watts-Strogatz small-world graph.
connected_watts_strogatz_graph(n, k, p[, ...])	Return a connected Watts-Strogatz small-world graph.
random_regular_graph(d, n[, seed])	Return a random regular graph of n nodes each with degree d.
barabasi_albert_graph(n, m[, seed])	Return random graph using Barabási-Albert preferential attachment model.
powerlaw_cluster_graph(n, m, p[, seed])	Holme and Kim algorithm for growing graphs with powerlaw
random_lobster(n, p1, p2[, seed])	Return a random lobster.
random_shell_graph(constructor[, seed])	Return a random shell graph for the constructor given.
random_powerlaw_tree(n[, gamma, seed, tries])	Return a tree with a powerlaw degree distribution.
random_powerlaw_tree_sequence(n[, gamma, ...])	Return a degree sequence for a tree with a powerlaw distribution.

22

Generalized random graphs

- Any degree distribution, clustering, …
- Later on ….. configurational model

23

2. Watts-Strogatz: small-world model

- Small world: the average shortest path length in a real network is small
- Six degrees of separation (Milgram, 1967)
- Local neighborhood + long-range friends
- A random graph is a small world

24

Networks in nature (empirical observations)

$$l_{\text{network}} \approx \ln(N)$$

$$C_{\text{network}} >> C_{\text{random graph}}$$

25

Big Data <-> Big Networks
Albert Díaz Guilera

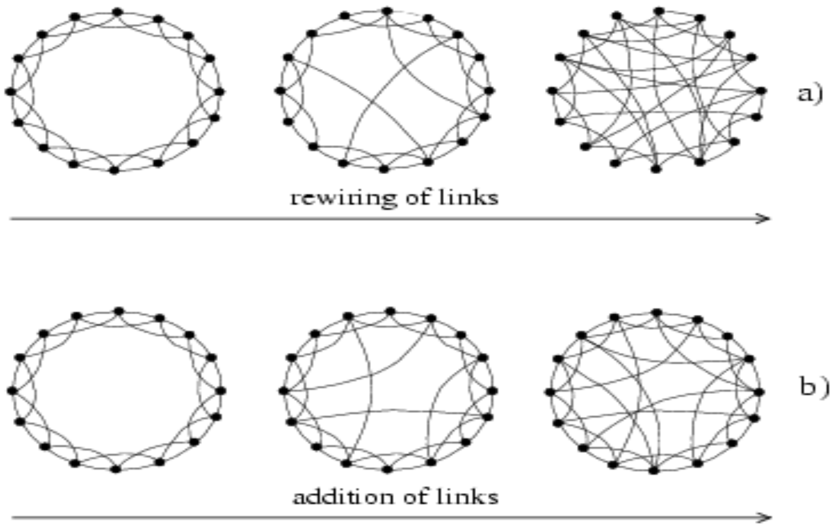
Model proposed

- Crossover from regular lattices to random graphs
- Tunable
- Small world network with (simultaneously):
  - Small average shortest path
  - Large clustering coefficient (not obeyed by RG)

26

Albert Díaz Guilera

Two ways of constructing

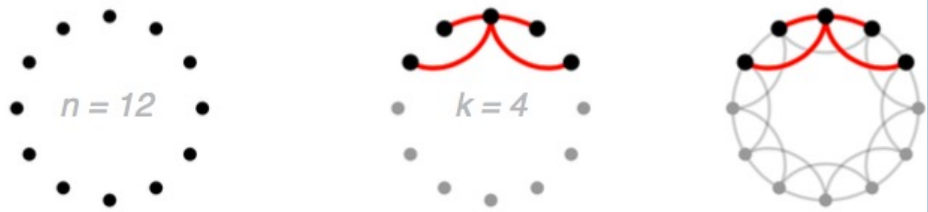


27

We start with a ring of n vertices

where each vertex is connected to its k nearest neighbors

like so.



We choose a vertex, and the edge to its nearest clockwise neighbour.

With probability p, we reconnect this edge to a vertex chosen uniformly at random over the entire ring, with duplicate edges forbidden. Otherwise, we leave the edge in place.

28

 Albert Díaz Guilera

Original model

- Each node has K>=4 nearest neighbors (local)
- Probability p of rewiring to randomly chosen nodes
- p small: regular lattice
- p large: classical random graph

29

Big Data <-> Big Networks Albert Díaz Guilera

30

p=0 Ordered lattice

$$l \approx \frac{N}{2K} >> 1$$

$$C = \frac{3(K-2)}{4(K-1)}$$

31

p=1 Random graph

$$l \approx \frac{\ln N}{\ln K}$$ small

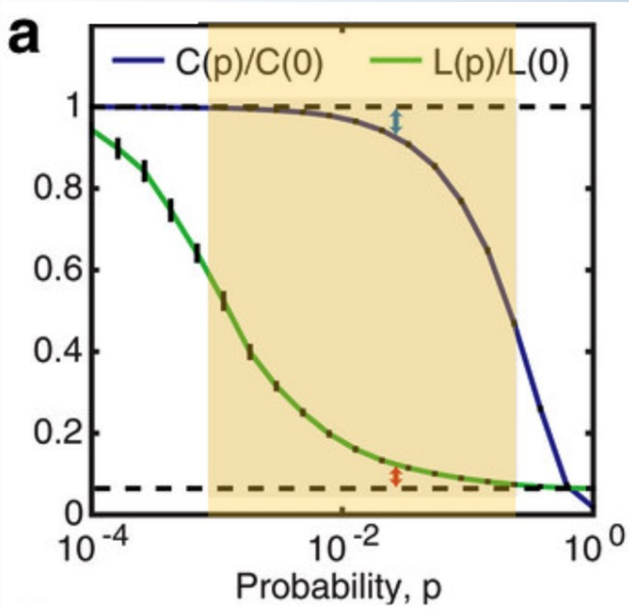
$$C \approx \frac{K}{N}$$ small

32

- Small shortest path means small clustering?
- Large shortest path means large clustering?
- They discovered: there exists a broad region:
  - Fast decrease of mean distance
  - Constant clustering

33

Big Data <-> Big Networks Albert Díaz Guilera



34

Average shortest path

$$l(p \to 0) \approx N$$

$$l(p \to 1) \approx \ln N$$

- Rapid drop of l, due to the appearance of short-cuts between nodes
- It starts to decrease when p>=2/NK (existence of one short cut)

35

Big Data <-> Big Networks Albert Díaz Guilera

- The value of p at which we should expect the transtion depends on N
- There will exist a crossover value of the system size:

$$N < N^* \Rightarrow l \approx N$$

$$N > N^* \Rightarrow l \approx \ln N$$

36

 Albert Díaz Guilera

Degree distribution

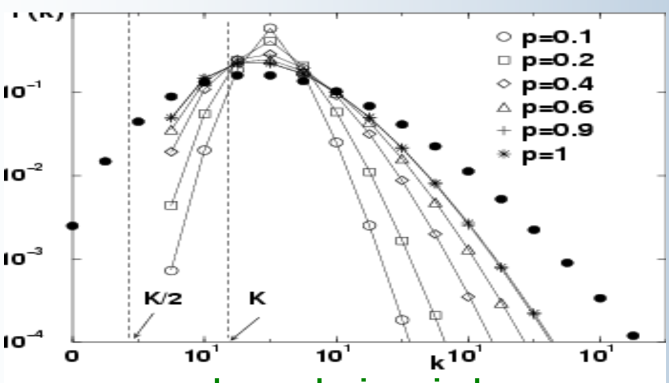
- p=0 delta-function
- p>0 broadens the distribution
- Edges left in place with probability (1-p)
- Edges rewired towards i with probability 1/N

37

Albert Díaz Guilera

$$P(k) = \sum_{n=0}^{f(k,K)} C^{n}_{K/2} (1-p)^n p^{K/2-n} \frac{(pK/2)^{k-K/2-n}}{(k-K/2-n)!} e^{-pK/2}$$

for $k \geq K/2$, where $f(k,K) = \min(k-K/2, K/2)$



only one edge is rewired
exponential decay, all nodes have similar number of links

38

Algorithm 4.19 Watts-Strogatz Small-world Model.

```
Input: V (set of nodes, whose size is N), m (number of connections to
  clockwise neighbors) and p (rewiring probability).
  for all node i ∈ V do
      add edge to the m clockwise nearest neighbors
  end for
  for all number of iterations (this is to assure that the resulting degree
  distribution is stable) do
      for all node i ∈ V do
          for all links to clockwise nearest neighbors of i do
              Generate a random number r
              if r ≤ p then
                  select a random node j ∈ V
                  if link (i, j) does not exist then
                      add the edge (i, j)
                  else
                      go back and select another random node j ∈ V
                  end if
              end if
          end for
      end for
  end for
  return G(V, E)
```

39

Random Graphs

Generators for random graphs.

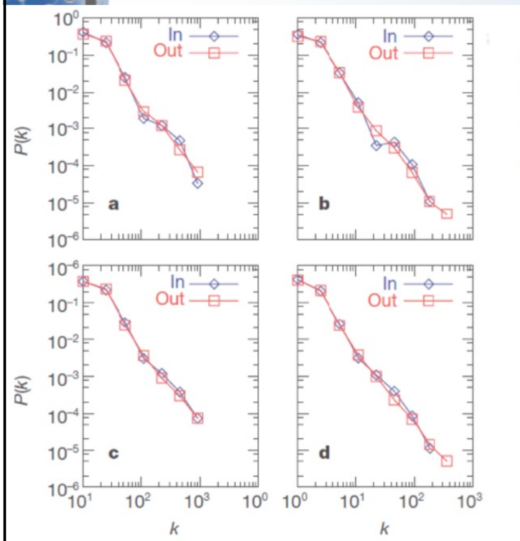
fast_gnp_random_graph(n, p[, seed, directed])	Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
gnp_random_graph(n, p[, seed, directed])	Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
dense_gnm_random_graph(n, m[, seed])	Return the random graph G_{n,m}.
gnm_random_graph(n, m[, seed, directed])	Return the random graph G_{n,m}.
erdos_renyi_graph(n, p[, seed, directed])	Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
binomial_graph(n, p[, seed, directed])	Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
newman_watts_strogatz_graph(n, k, p[, seed])	Return a Newman-Watts-Strogatz small world graph.
watts_strogatz_graph(n, k, p[, seed])	Return a Watts-Strogatz small-world graph.
connected_watts_strogatz_graph(n, k, p[, ...])	Return a connected Watts-Strogatz small-world graph.
random_regular_graph(d, n[, seed])	Return a random regular graph of n nodes each with degree d.
barabasi_albert_graph(n, m[, seed])	Return random graph using Barabási-Albert preferential attachment model.
powerlaw_cluster_graph(n, m, p[, seed])	Holme and Kim algorithm for growing graphs with powerlaw
random_lobster(n, p1, p2[, seed])	Return a random lobster.
random_shell_graph(constructor[, seed])	Return a random shell graph for the constructor given.
random_powerlaw_tree(n[, gamma, seed, tries])	Return a tree with a powerlaw degree distribution.
random_powerlaw_tree_sequence(n[, gamma, ...])	Return a degree sequence for a tree with a powerlaw distribution.

40

Big Data <-> Big Networks

Albert Díaz Guilera

3. Barabasi-Albert: scale-free network



JEONG ET AL. NATURE 407, 651 (2000)

- Many large networks are scale free
- The degree distribution has a power-law behavior for large k (far from a Poisson distribution)
- Random graph theory and the Watts-Strogatz model cannot reproduce this feature

41

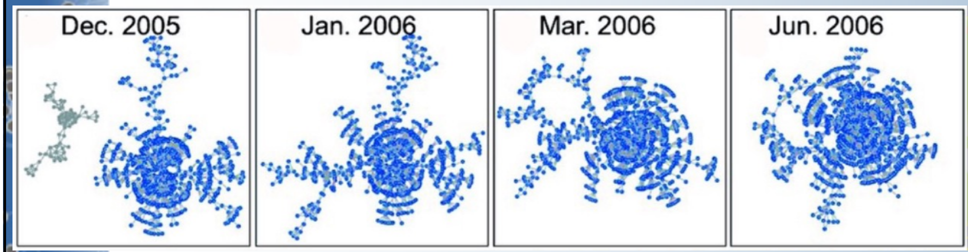
Big Data <-> Big Networks

Albert Díaz Guilera

- We can construct power-law networks by hand
- Which is the mechanism that makes scale-free networks to emerge as they grow?
- Emphasis: network dynamics rather to construct a graph with given topological features

42

Networks evolve



43

- Topology is a result of the dynamics
- But only a random growth?
- In this case the distribution is exponential!

44

Barabasi-Albert model (1999)

- Two generic mechanisms common in many real networks
  - Growth (www, research literature, ...)
  - Preferential attachment (idem): attractiveness of popularity
- The two are necessary

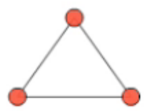
45

Big Data <-> Big Networks Albert Díaz Guilera

Growth

- t=0, m_0 nodes
- Each time step we add a new node with m ($\leq m_0$) edges that link the new node to m different nodes already present in the system

46



47

 Albert Díaz Guilera

Preferential attachment: rich gets richer

- When choosing the nodes to which the new connects, the probability Π that a new node will be connected to node *i* depends on the degree k_i of node *i*

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j}$$

Linear attachment (more general models)
Sum over all existing nodes

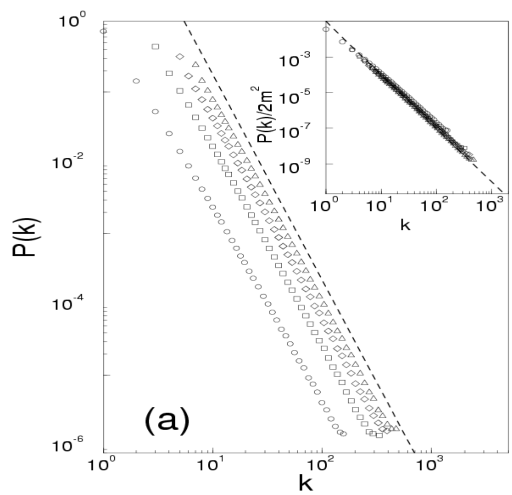
48

Numerical simulations

- Power-law $P(k) \approx k^{-\gamma}$ $\gamma_{SF}=3$
- The exponent does not depend on m (the only parameter of the model)

49

Big Data <-> Big Networks
Albert Díaz Guilera



γ=3. different m's. P(k) changes. γ not

50

Degree distribution

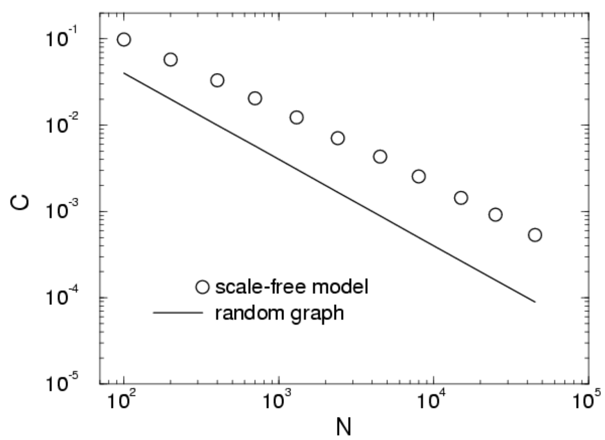
- Analytically

$$P(k) = \frac{2m(m+1)}{k(k+1)(k+2)}$$

51

Big Data <-> Big Networks
Albert Díaz Guilera

Clustering coefficient



5 times larger

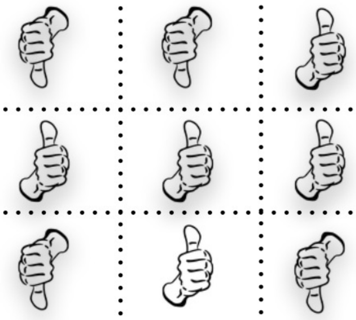
$$C_{\mathrm{SF}} \sim N^{-0.75}$$

$$C_{\mathrm{RG}} = <k> N^{-1}$$

SW: C is independent of N

52

	ER	WS	BA
• Degree distribution			
• Average length			
• Clustering Coefficient			



53

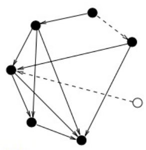
Alternative models

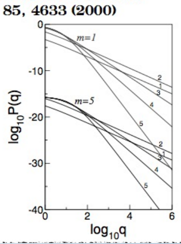
Scale-free with tunable γ

DOROGOVTSEV-MENDES-SAMUKHIN
PHYS. REV. LETT. 85, 4633 (2000)

$$\Pi_i(t) = \frac{k_i + \alpha}{\sum_{j=1}^{m_0+t-1}(k_j(t)+\alpha)}$$

$$\alpha \in (-m, \infty) \longrightarrow \gamma = 3 + \frac{\alpha}{m}$$

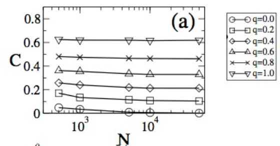


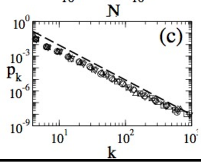


Scale-free with high clustering

HOLME-KIM
PHYS. REV. E 65, 026107 (2002)

- First link: follow usual PA rule
- For each of the $m-1$ links:
  - (i) With probability $(1-q)$: usual PA
  - (ii) With probability q: Attach to one neighbor of the first chosen node





$\gamma = 3$

54

Algorithm 4.20 Barabási-Albert (BA) Scale-Free Model.

Input: N (size of the network), $m \leq m_0$ (number of connections of newly added nodes) and m_0 (number of nodes in the initial core, should be small).

```
for all m_0 initial nodes do
    create a fully connected network of m_0 nodes
end for
for all time step t ≤ N − m_0 do
    add a new node j
    number of newly added links ← 0
    Σ_i^{(m_0+t−1)} k_i = m_0(m_0 − 1) + 2m(t − 1) = S
    while number of newly added links < m do
        add 1 to number of newly added links
        Generate a random number r
        while Σ_{i≠l} k_i < rS do
            add the edge (l, j) (l satisfies that Σ_i^{l−1} k_i < rS ≤ Σ_i^l k_i
            S = S − k_l
        end while
    end while
end for
return G_{N,K}^{BA}
```

55

Other (microscopic) models

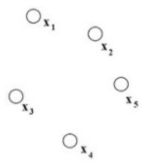
- Fitness model
- Hidden variables
- Social distance
- Popularity
- Gravitational model

56

Big Data <-> Big Networks Albert Díaz Guilera

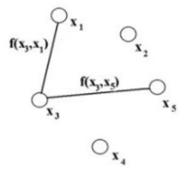
Fitness model

FITNESS model (Caldarelli *et al.* *Phys. Rev. Lett.* 2002)



• Each vertex ***i*** is assigned a *fitness value* $\boldsymbol{x_i}$ drawn from a given distribution $\boldsymbol{\rho(x)}$;

• A link is drawn between each pair of vertices ***i*** and ***j*** with probability $\boldsymbol{f(x_i,x_j)}$ depending on $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$.



Power-law degree distributions are obtained by chosing

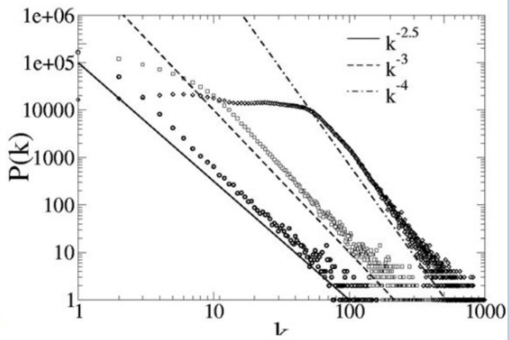
$$\rho(x) \propto x^{-\alpha}$$

$$f(x_i,x_j) \propto x_i x_j$$

or

$$\rho(x) = e^{-x}$$

$$f(x_i,x_j) \propto \theta(x_i+x_j-z)$$



57

Big Data <-> Big Networks Albert Díaz Guilera

Hidden variables

1. Each vertex i is assigned a variable h_i, independently drawn from the probability distribution $\rho(h)$.

2. For each pair of vertices i and j, with respective hidden variables h_i and h_j, an undirected edge is created with probability $r(h_i, h_j)$ (the *connection probability*), where $r(h, h') \geq 0$ is a symmetric function of h and h'.

58

A model based on social distance

- Properties of a nonbipartite network
- Relations are established between individuals that feel "close"
- Assign a position to each individual = a set of values in a number of social dimensions

59

Big Data <-> Big Networks Albert Díaz Guilera

1 social dimension

- Position of individual i in social space h_i
- Define a density $\rho(h)$
- Probability of establishing a relation decreases with social distance

$$r_n(h_i^n, h_j^n) = \frac{1}{1 + \left[b_n^{-1} d_n(h_i^n, h_j^n)\right]^{\alpha_n}}$$

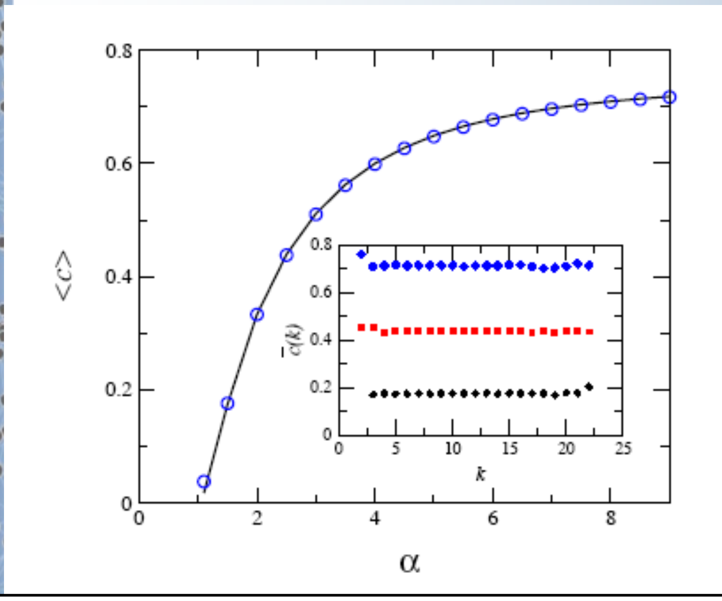
- α measures the degree of homophily

60

Big Data <-> Big Networks

Albert Díaz Guilera

Clustering coefficient

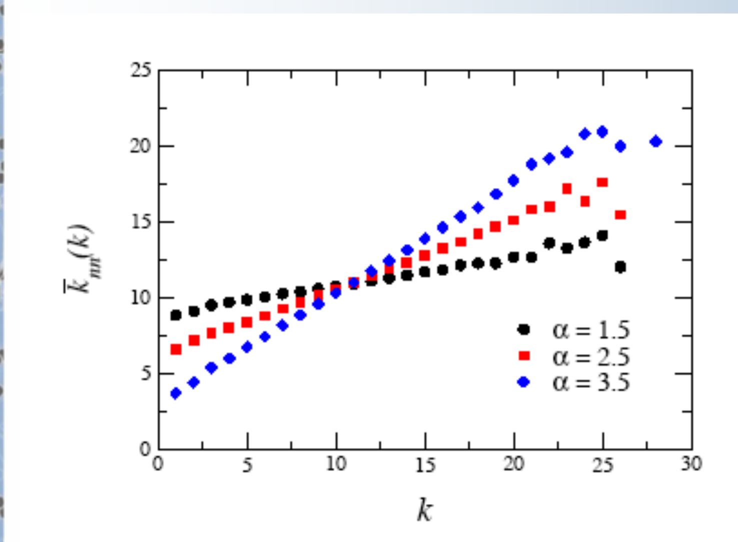


61

Big Data <-> Big Networks

Albert Díaz Guilera

Assortativity: degree correlations

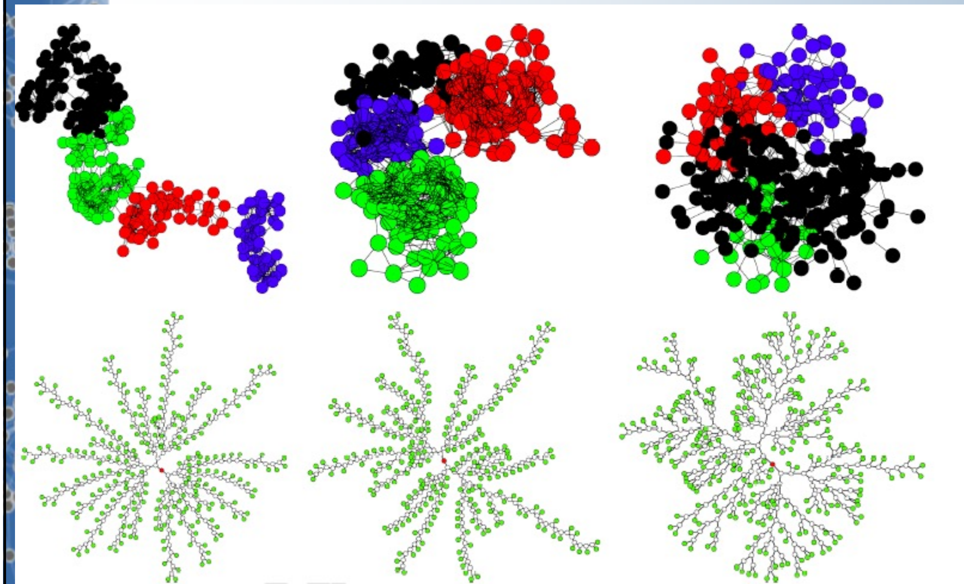


62

Big Data <-> Big Networks

Albert Díaz Guilera

Communities



63

Big Data <-> Big Networks

LETTER

doi:10.1038/nature11459

Popularity versus similarity in growing networks

Fragkiskos Papadopoulos[1], Maksim Kitsak[2], M. Ángeles Serrano[3], Marián Boguñá[3] & Dmitri Krioukov[2]

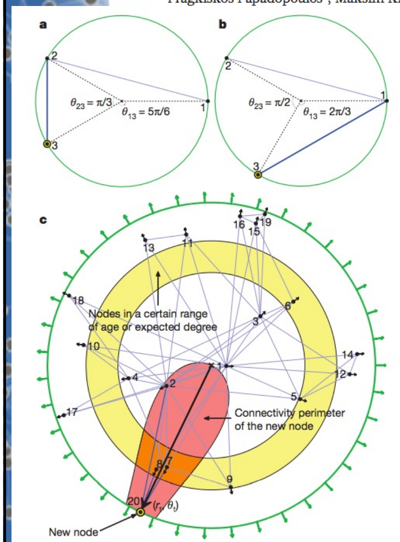


Figure 1 | Geometric interpretation of popularity × similarity optimization. The nodes (dots) are numbered by their birth times, and located at random angular (similarity) coordinates. On its birth, the new circled node t in the yellow annulus connects to m old nodes s minimizing $s\theta_{st}$. The new connections are shown by the thicker blue links. In **a** and **b**, $t = 3$ and $m = 1$. In **a**, node 3 connects to node 2 because $2\theta_{23} = 2\pi/3 < 1\theta_{13} = 5\pi/6$. In **b**, node 3 connects to node 1 because $1\theta_{13} = 2\pi/3 < 2\theta_{23} = \pi$. In **c**, an optimization-driven network with $m = 3$ is simulated for up to 20 nodes. The radial (popularity) coordinate of new node $t = 20$ is $r_t = \ln t$, shown by the long thick arrow. This node connects to the three hyperbolically closest nodes. The red shape marks the set of points located at hyperbolic distances less than r_t from the new node. Arrows on dots show all nodes drifting away from the crossed origin, emulating popularity fading as explained in the text. The drift speed in the network shown corresponds to the degree distribution exponent $\gamma = 2.1$. The outer green circle shows the current network boundary of radius $r_t = \ln t$ expanding with time t as indicated by green arrows.

$$x_{st} = r_s + r_t + \ln(\theta_{st}/2) = \ln(st\theta_{st}/2)$$

64

Gravity model

9.2.1.1 *The family of gravity models of transportation*

A first step towards modelling mobility was based on an analogy with a phyisical law which gave rise to the term *gravity law of transportation* proposed by Zipf in [184]. This model is based on assuming that each location has a mass M_i (which can be related to population, job availability, interest...) and that the number of trips between two locations can be approximated by:

$$\langle t_{ij} \rangle = K M_i M_j f(d_{ij}). \tag{122}$$

Where K is a normalizing constant ensuring that $\sum_{ij} \langle t_{ij} \rangle = \hat{T}$ and $f(d_{ij})$ is an arbitrary (decreasing) function of the cost or distance. Its most common version uses a power law $f(d_{ij}) = d_{ij}^{-\gamma}$ but also exponential versions are present in the literature. In the urban context, we will only consider exponential versions of the deferrence function

65

Configurational model

- Choose P(k)
- Stubs that are connected

66

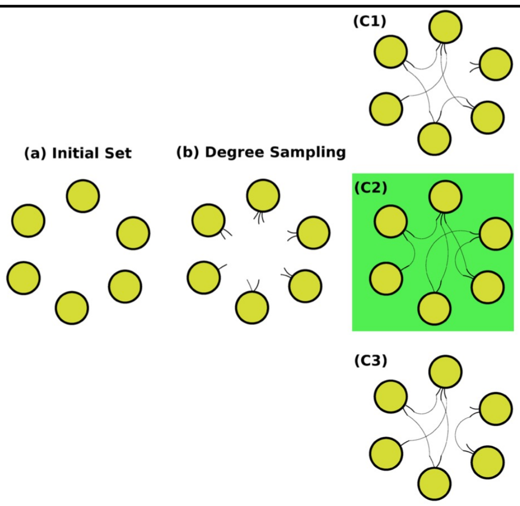


Figure 4.1 Schematic representation of the configurational model. In (a) the set of $N = 6$ nodes is given whereas in (b) the nodes are assigned a given degree (here ranging from $k = 1$ to $k = 3$). Finally in (c) the set of nodes are wired according to their degrees. We show three possibilities in (C1) and (C3) the wiring fails since some nodes cannot accommodate their stubs as it implies to make a self-connection (C1) or because it produces multiple connections between a pair of them (C3). In the central plot (C2) we show a successful wiring in which all the stubs have been matched without any of the former problems.

67

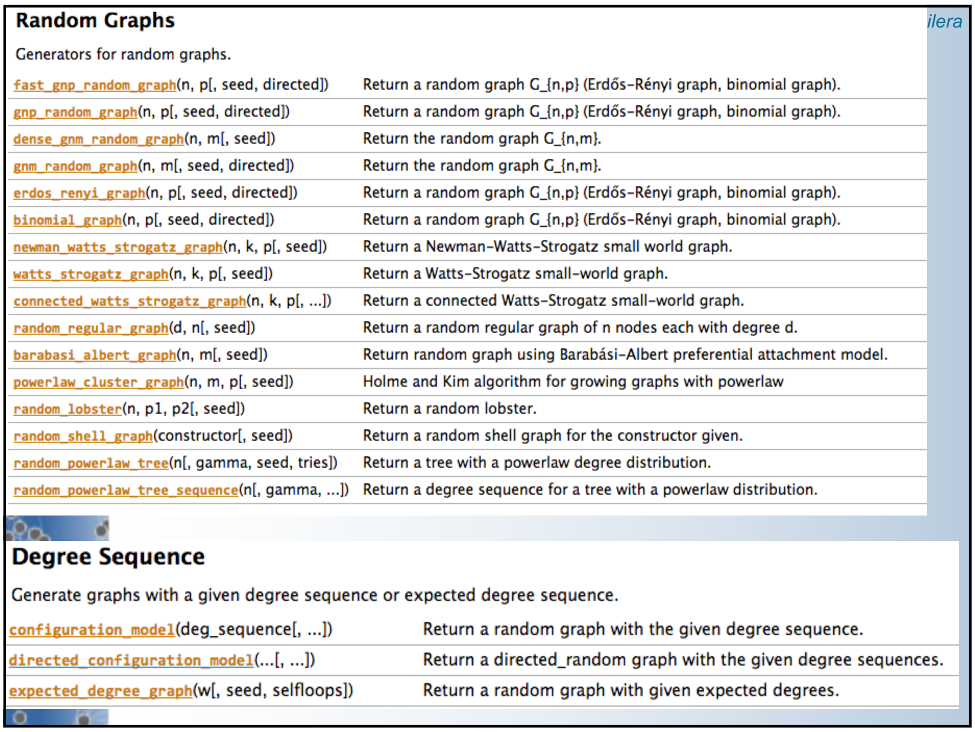
Algorithm 4.21 Configurational model

Input: V set of nodes, k_0 minimum connectivity, k_{max} maximum connectivity, $P(k)$ degree distribution.

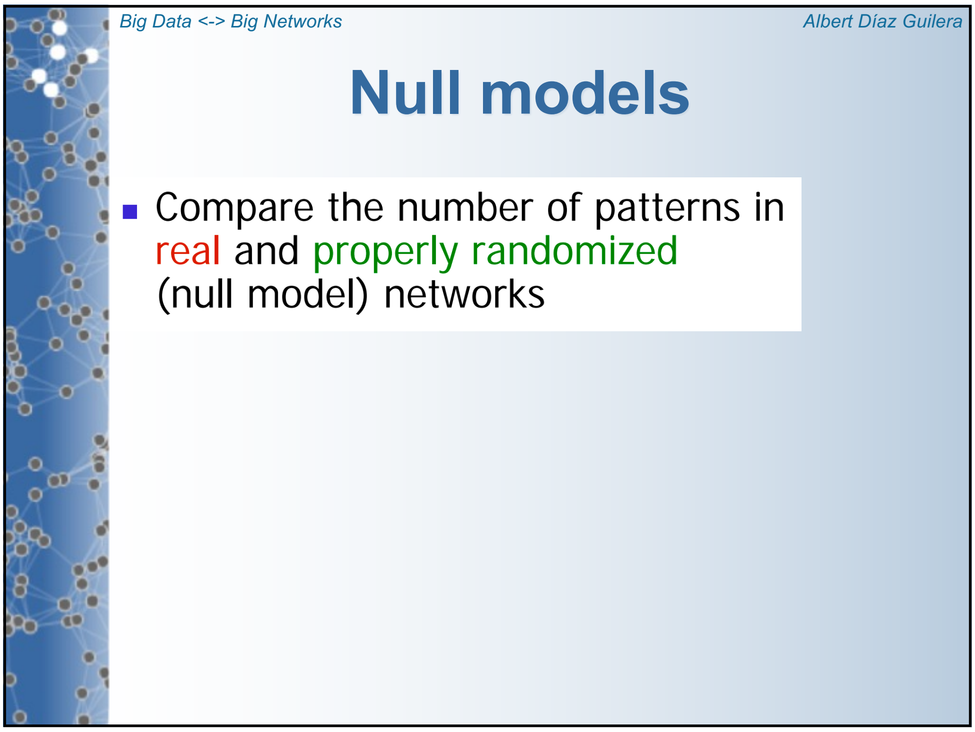
Output: $G(V, E)$ unweighted undirected network.

```
S ← 0
E ← 0
for all node i ∈ V do
    s(i) ← 0
end for
for all node i ∈ V do
    Assign the connectivity k(i) from P(k)
    Store the new k(i) stubs in s(i)
    Store the new k(i) stubs in S
end for
for all nodes i ∈ V do
    if s(i) contains stubs then
        Choose at random one stub m ∈ s(i)
        Choose at random one stub j ∈ S
        Identify the set s(l) so that j ∈ s(l)
        if s(i) ≠ s(l) then
            if edge (i, l) ∉ E then
                Remove stub m from S and s(i)
                Remove stub j from S and s(l)
                Add (i, l) in E
            end if
        end if
    end if
end for
return G(V, E)
```

68

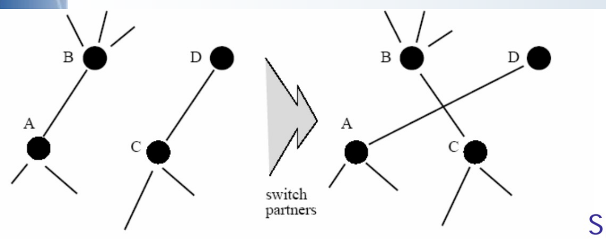

ilera
Random Graphs
Generators for random graphs.
fast_gnp_random_graph(n, p[, seed, directed]) Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
gnp_random_graph(n, p[, seed, directed]) Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
dense_gnm_random_graph(n, m[, seed]) Return the random graph G_{n,m}.
gnm_random_graph(n, m[, seed, directed]) Return the random graph G_{n,m}.
erdos_renyi_graph(n, p[, seed, directed]) Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
binomial_graph(n, p[, seed, directed]) Return a random graph G_{n,p} (Erdős-Rényi graph, binomial graph).
newman_watts_strogatz_graph(n, k, p[, seed]) Return a Newman-Watts-Strogatz small world graph.
watts_strogatz_graph(n, k, p[, seed]) Return a Watts-Strogatz small-world graph.
connected_watts_strogatz_graph(n, k, p[, ...]) Return a connected Watts-Strogatz small-world graph.
random_regular_graph(d, n[, seed]) Return a random regular graph of n nodes each with degree d.
barabasi_albert_graph(n, m[, seed]) Return random graph using Barabási-Albert preferential attachment model.
powerlaw_cluster_graph(n, m, p[, seed]) Holme and Kim algorithm for growing graphs with powerlaw
random_lobster(n, p1, p2[, seed]) Return a random lobster.
random_shell_graph(constructor[, seed]) Return a random shell graph for the constructor given.
random_powerlaw_tree(n[, gamma, seed, tries]) Return a tree with a powerlaw degree distribution.
random_powerlaw_tree_sequence(n[, gamma, ...]) Return a degree sequence for a tree with a powerlaw distribution.
Degree Sequence
Generate graphs with a given degree sequence or expected degree sequence.
configuration_model(deg_sequence[, ...]) Return a random graph with the given degree sequence.
directed_configuration_model(...[, ...]) Return a directed_random graph with the given degree sequences.
expected_degree_graph(w[, seed, selfloops]) Return a random graph with given expected degrees.

69


Big Data <-> Big Networks
Albert Díaz Guilera
Null models
Compare the number of patterns in real and properly randomized (null model) networks

70

Maslov-Sneppen



- Randomly select and rewire two edges
- Repeat many times

S. Maslov,
K. Sneppen,
Science (2002)

R. Kannan,
P. Tetali,
S. Vempala,
Random Structures and Algorithms (1999).

71

Random Network Generator

RandNetGen

Random Network Generator

- http://polcolomer.github.io/RandNetGen/

73