- “Donat el caràcter i la finalitat exclusivament docent i eminentment il·lustrativa de les explicacions a classe d'aquesta presentació, l’autor s’acull a l’article 32 de la Llei de propietat intel·lectual vigent respecte de l'ús parcial d'obres alienes com ara imatges, gràfics o altre material contingudes en les diferents diapositives”
- “Dado el carácter y la finalidad exclusivamente docente y eminentemente ilustrativa de las explicaciones en clase de esta presentación, el autor se acoge al artículo 32 de la Ley de Propiedad Intelectual vigente respecto al uso parcial de obras ajenas como imágenes, gráficos u otro material contenidos en las diferentes diapositivas”.

1

UNIVERSITAT DE BARCELONA

Complex Networks

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

CLabB complexity lab barcelona

COMPLEXITAT

2

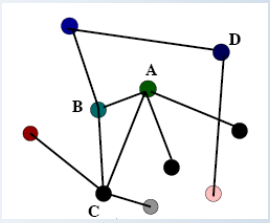
Macroscale

- Shortest-paths
- Clustering (of the network)
- Distributions (degree, betweenness, …)
- Statistical properties: clustering
- Correlations

3

Distance between two nodes

- Number of links that make up the shortest-path between two nodes



- Centrality: nodes that are "close" to many other nodes in the network.
- Global centrality: defined as the sum of minimum distances to any other nodes in the networks

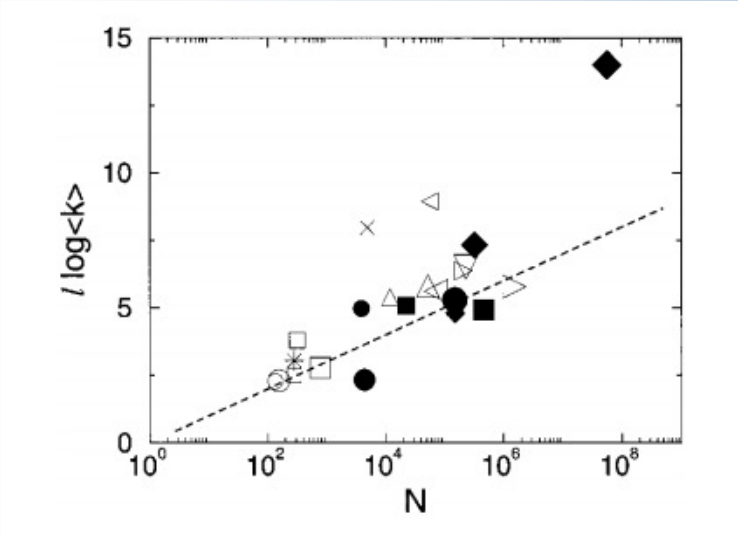
4

Global centrality of the whole network?

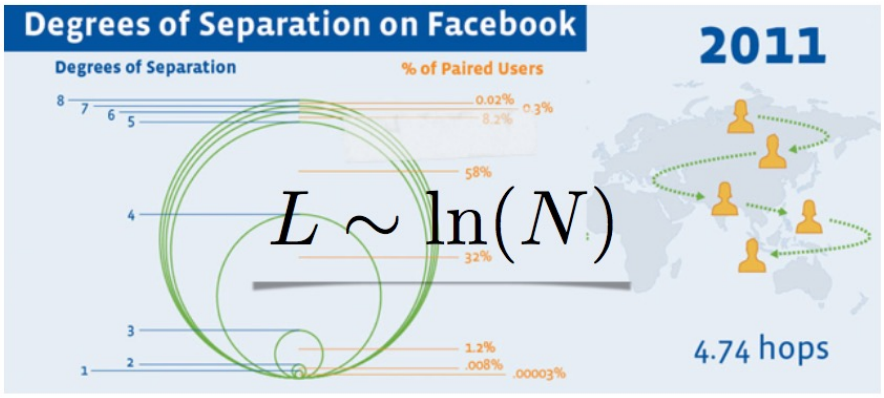
- Mean shortest path = average over all pairs of nodes in the network
- Diameter: largest distance between a pair of nodes in the network

5

What do we find?

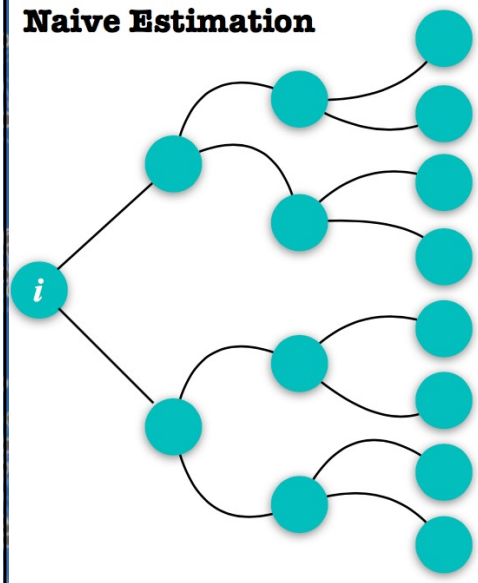


6



7

Naive Estimation



The number of nodes at distance l from node i is:

$$\langle k \rangle^l$$

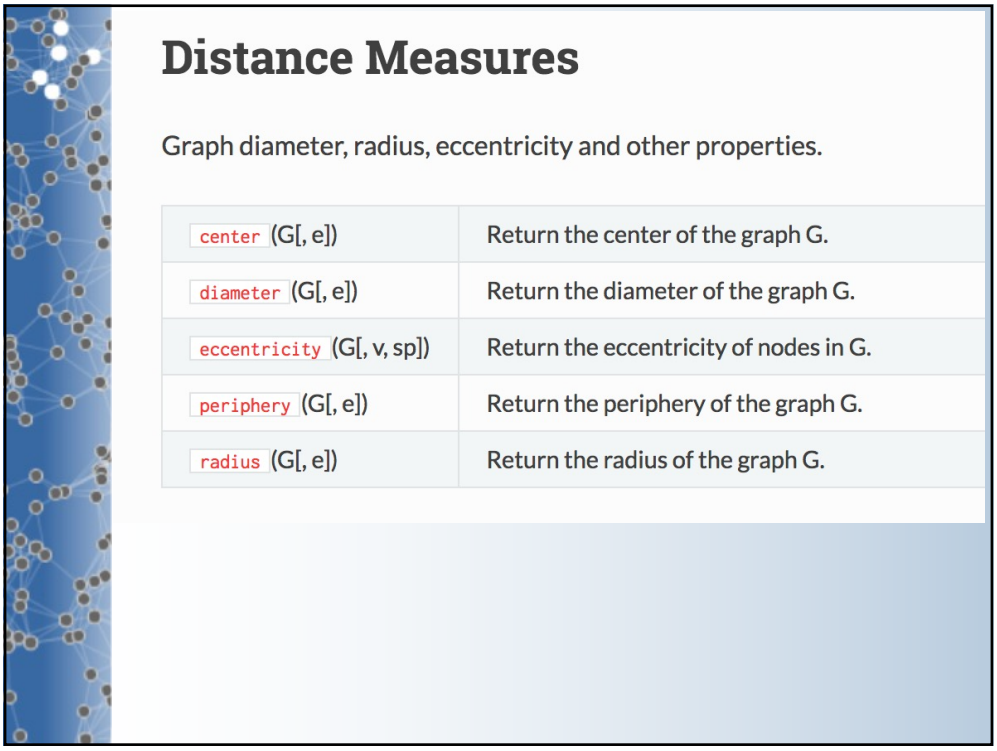
To reach all the nodes:

$$N = \sum_{l=0}^{L_{\max}} \langle k \rangle^l \geq \langle k \rangle^{L_{\max}} \geq \langle k \rangle^L$$

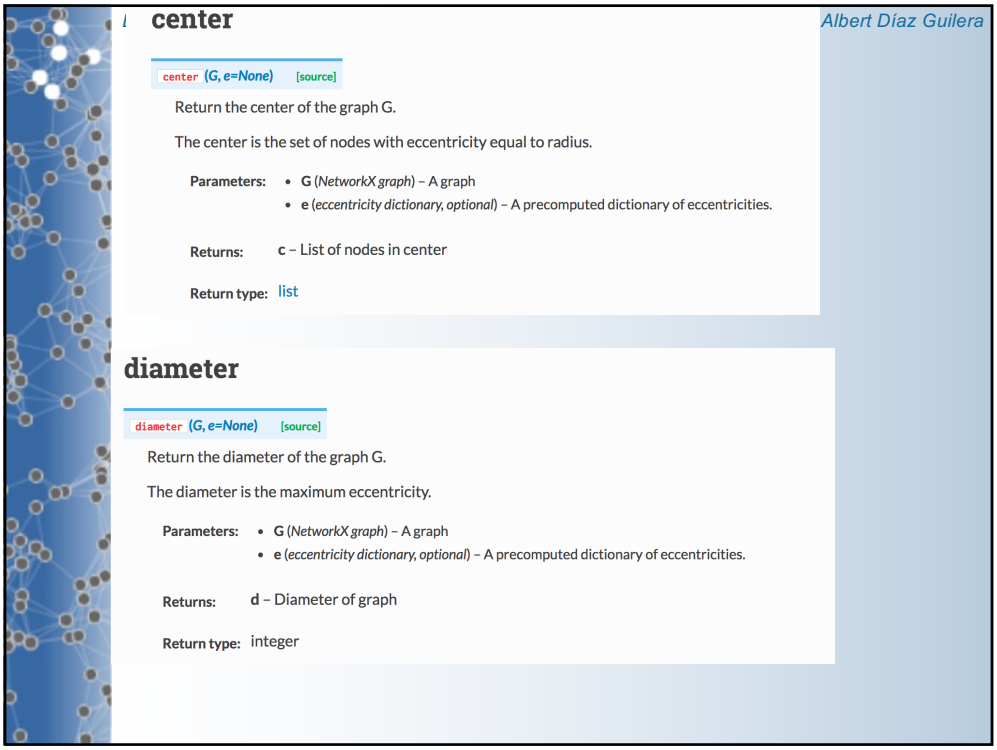
$$\log(N) \geq L \log(\langle k \rangle)$$

$$L \leq \frac{\log(N)}{\log(\langle k \rangle)}$$

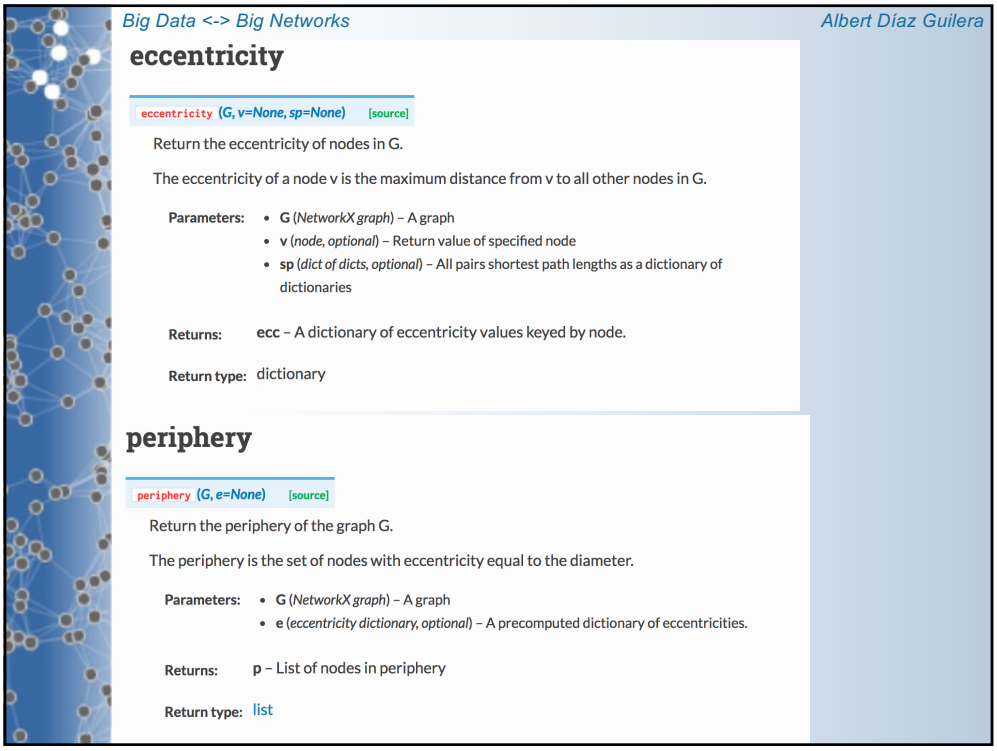
8


Distance Measures
Graph diameter, radius, eccentricity and other properties.
center (G[, e]) Return the center of the graph G.
diameter (G[, e]) Return the diameter of the graph G.
eccentricity (G[, v, sp]) Return the eccentricity of nodes in G.
periphery (G[, e]) Return the periphery of the graph G.
radius (G[, e]) Return the radius of the graph G.

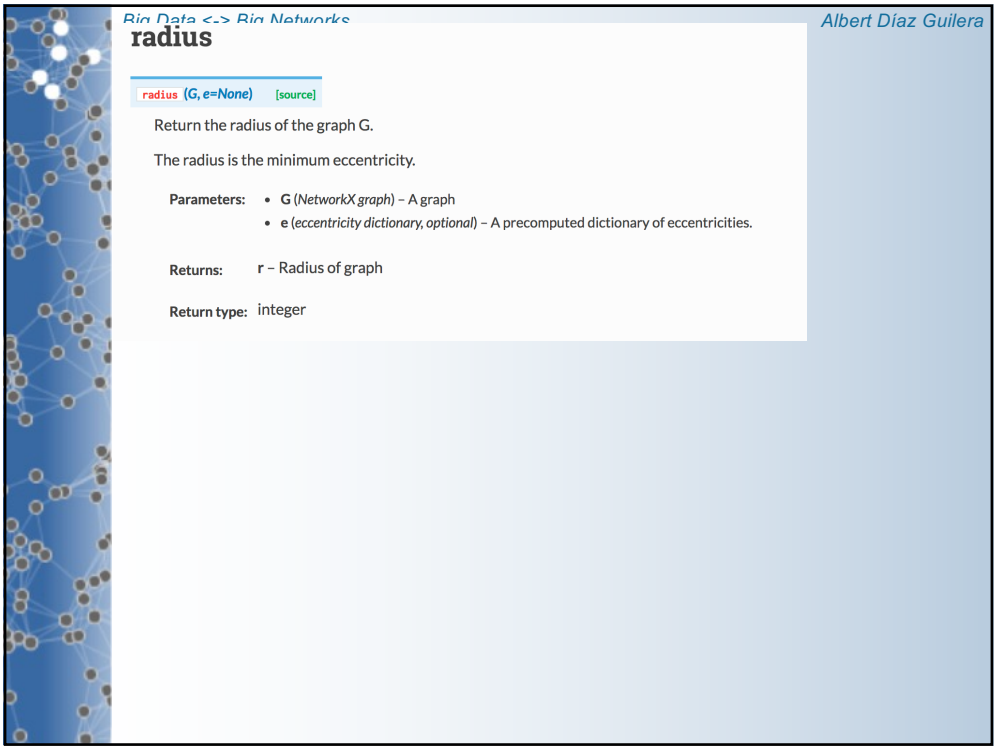
9


center
Albert Díaz Guilera
center (G, e=None) [source]
Return the center of the graph G.
The center is the set of nodes with eccentricity equal to radius.
Parameters: • G (NetworkX graph) – A graph
• e (eccentricity dictionary, optional) – A precomputed dictionary of eccentricities.
Returns: c – List of nodes in center
Return type: list
diameter
diameter (G, e=None) [source]
Return the diameter of the graph G.
The diameter is the maximum eccentricity.
Parameters: • G (NetworkX graph) – A graph
• e (eccentricity dictionary, optional) – A precomputed dictionary of eccentricities.
Returns: d – Diameter of graph
Return type: integer

10


Big Data <-> Big Networks
Albert Díaz Guilera
eccentricity
eccentricity (G, v=None, sp=None) [source]
Return the eccentricity of nodes in G.
The eccentricity of a node v is the maximum distance from v to all other nodes in G.
Parameters: G (NetworkX graph) – A graph
v (node, optional) – Return value of specified node
sp (dict of dicts, optional) – All pairs shortest path lengths as a dictionary of dictionaries
Returns: ecc – A dictionary of eccentricity values keyed by node.
Return type: dictionary
periphery
periphery (G, e=None) [source]
Return the periphery of the graph G.
The periphery is the set of nodes with eccentricity equal to the diameter.
Parameters: G (NetworkX graph) – A graph
e (eccentricity dictionary, optional) – A precomputed dictionary of eccentricities.
Returns: p – List of nodes in periphery
Return type: list

11


Albert Díaz Guilera
radius
radius (G, e=None) [source]
Return the radius of the graph G.
The radius is the minimum eccentricity.
Parameters: G (NetworkX graph) – A graph
e (eccentricity dictionary, optional) – A precomputed dictionary of eccentricities.
Returns: r – Radius of graph
Return type: integer

12

dijkstra_path

`dijkstra_path` **(G, source, target, weight='weight')** [source]

Returns the shortest path from source to target in a weighted graph G.

Parameters:
- **G** (*NetworkX graph*) –
- **source** (*node*) – Starting node
- **target** (*node*) – Ending node
- **weight** (*string, optional (default='weight')*) – Edge data key corresponding to the edge weight

Returns: **path** – List of nodes in a shortest path.

Return type: list

14

Clustering

- Cycles in social network analysis language
- Circles of friends in which every member knows each other

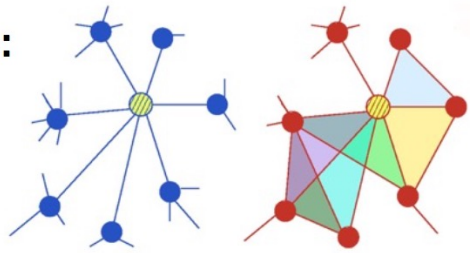
15

Clustering of a node:

$$C_i = \frac{\#\text{triangles connected to } i}{\#\text{possible triangles connected to } i} = \frac{2E_i}{k_i(k_i-1)}$$

Clustering of the Network:

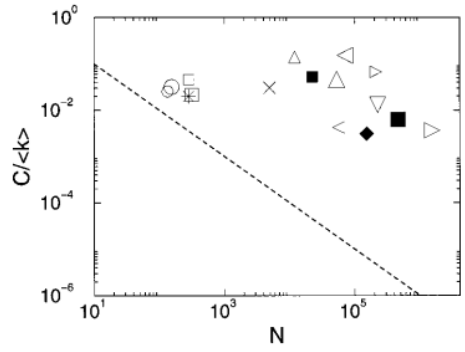
$$C = \frac{1}{N}\sum_{i=1}^{N} C_i$$



- Alternative definition: ratio between total number of triangles and possible

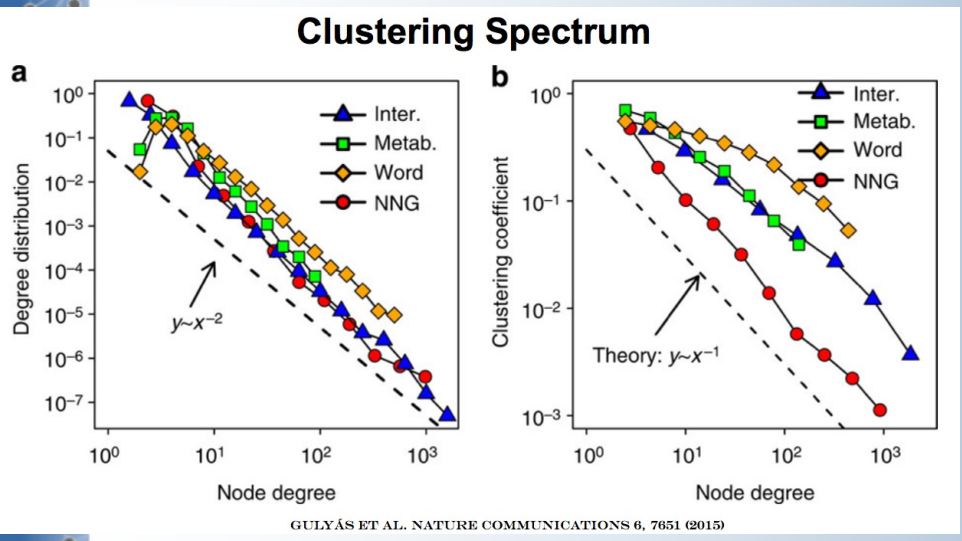
16

What happens in real networks?



- The clustering coefficient is much larger than it is in an equivalent random network

17



18

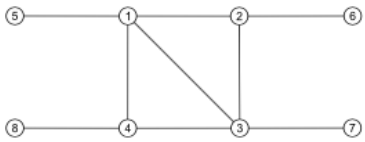


Figure 10.1: A network whose clustering coefficient can be calculated

Example 10.2.1

- Consider the network illustrated in Figure 10.1

Nodes 1 and 3 are equivalent. They both take part in two triangles and their degrees is 4. Thus,

$$C_1 = C_3 = \frac{2 \cdot (2)}{4 \cdot 3} = \frac{1}{3}. \tag{10.4}$$

In a similar way we obtain

$$C_2 = C_4 = \frac{1}{3}. \tag{10.5}$$

Notice that because nodes 5–8 are not involved in any triangle we have that $C_{i \geq 5} = 0$. Consequently,

$$\overline{C} = \frac{1}{8}\left(\frac{4}{3}\right) = \frac{1}{6}. \tag{10.6}$$

19

10.3 The Newman Clustering Coefficient

Another way of quantifying the global clustering of a network is by means of the Newman clustering coefficient, also known as the **transitivity index** of the network. Let $t = |C_3|$ be the total number of triangles, and let $|P_2|$ be the number of paths of length 2 in the network (representing all potential three-way relationships). Then,

$$C = \frac{3t}{|P_2|} = \frac{3|C_3|}{|P_2|}. \tag{10.7}$$

- Consider again the network illustrated in Figure 10.1. We can obtain the number of triangles in that network by using the spectral properties of the adjacency matrix. That is,

$$t = \frac{1}{6}\,\mathrm{tr}(A^3) = 2. \tag{10.8}$$

The number of paths of length 2 in the network can be obtained using the following formula (which we will justify later).

$$|P_2| = \sum_{i=1}^{n} \binom{k_1}{2} = \sum_{i=1}^{n} \frac{k_i(k_i-1)}{2} = 18. \tag{10.9}$$

Thus,

$$C = \frac{3 \times 2}{18} = \frac{1}{3}. \tag{10.10}$$

20

In general, the Watts–Strogatz index quantifies how clustered a network is locally, while Newman one indicates how clustered the network is as a whole. In general there is a good correlation between both indices for real-world networks as illustrated in Figure 10.4.

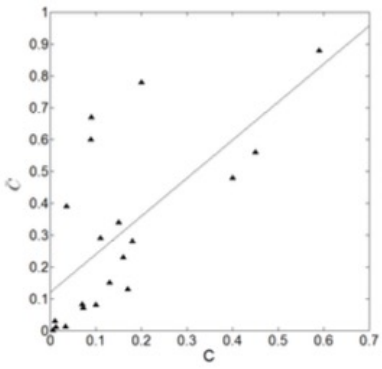


Figure 10.4: Correlation between Watts–Strogatz ($\bar{C}$) and Newman (C) clustering coefficients for 20 real-world networks.

21

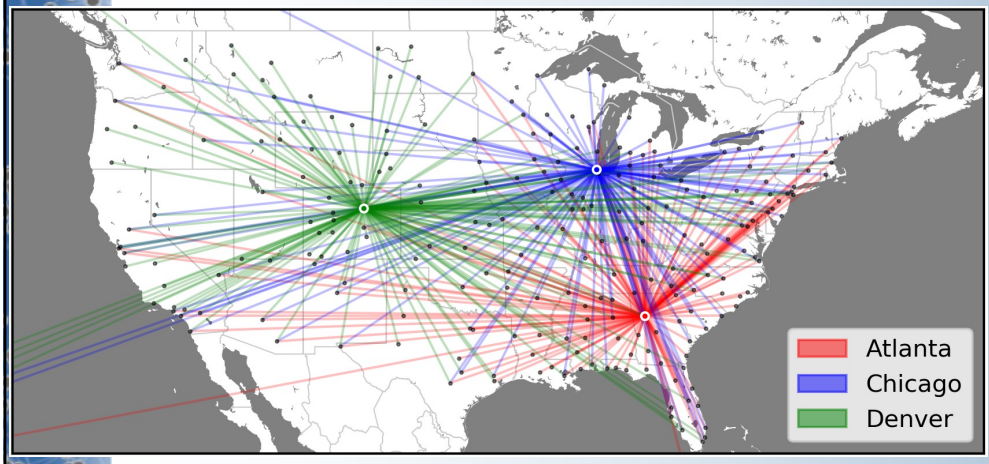
Clustering

Algorithms to characterize the number of triangles in a graph.

`triangles` (G[, nodes])	Compute the number of triangles.
`transitivity` (G)	Compute graph transitivity, the fraction of all possible triangles present in G.
`clustering` (G[, nodes, weight])	Compute the clustering coefficient for nodes.
`average_clustering` (G[, nodes, weight, ...])	Compute the average clustering coefficient for the graph G.
`square_clustering` (G[, nodes])	Compute the squares clustering coefficient for nodes.

23

Degree: real networks are heterogeneous



Some nodes are more important tan others

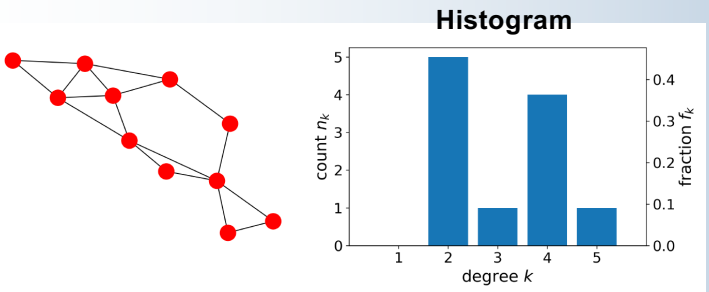
24

Degree distribution (macroscopic scale)

- Gives an idea of the spread in the number of links the nodes have
- P(k) is the probability that a randomly selected node has k links
- Statistical sense

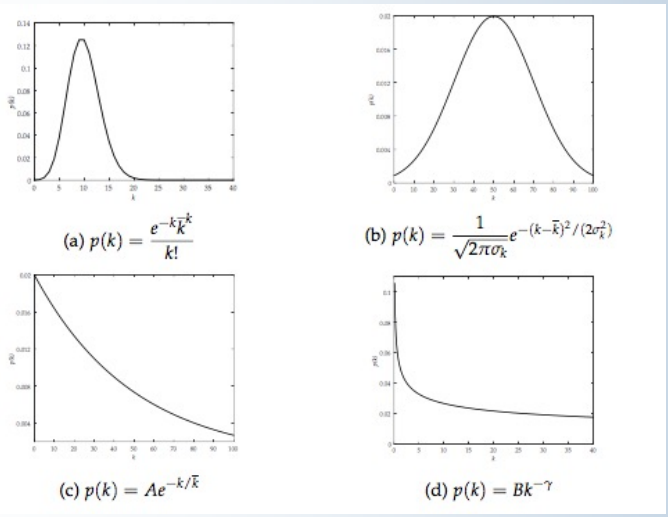
25

Degree distributions

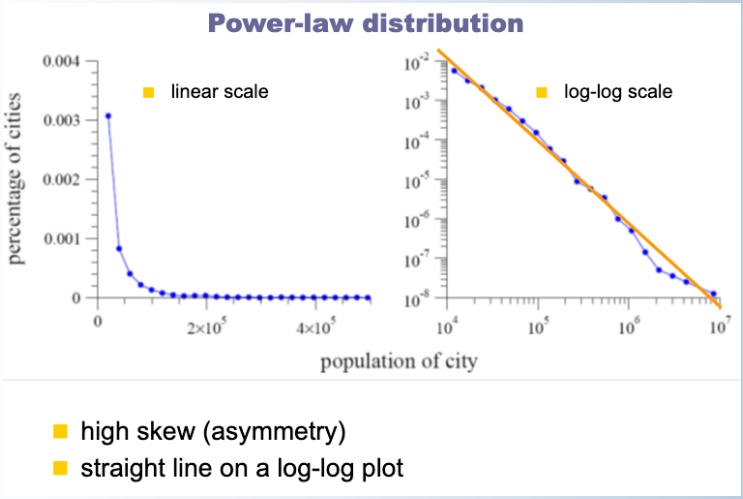


- n_k = **number of nodes with degree** k
- $f_k = \frac{n_k}{N}$ = **frequency of degree** k

26

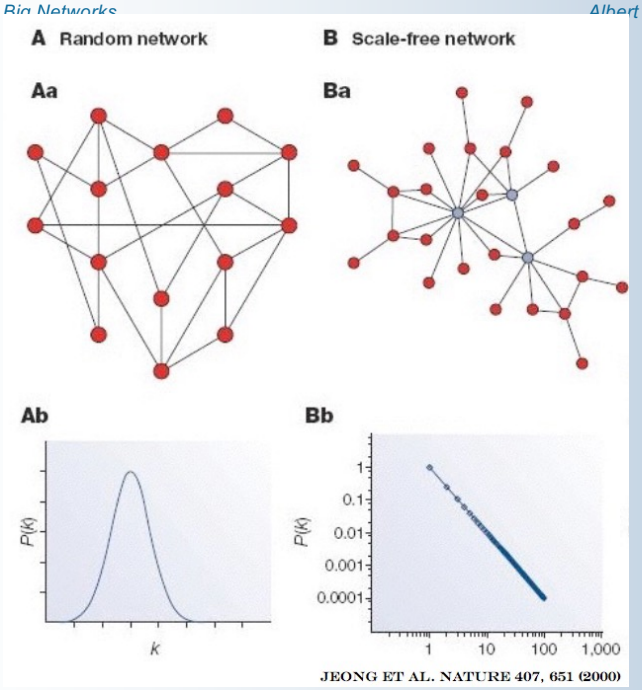


27



- high skew (asymmetry)
- straight line on a log-log plot

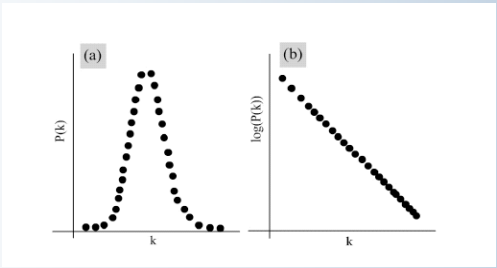
28



29

What should we expect?

- In regular lattices all nodes are identical
- In random networks the majority of nodes have approximately the same degree



- Real-world networks: this distribution has a power tail

$$P(k) \approx k^{-\gamma}$$

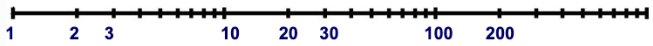
"scale-free" networks

30

Logarithmic scale

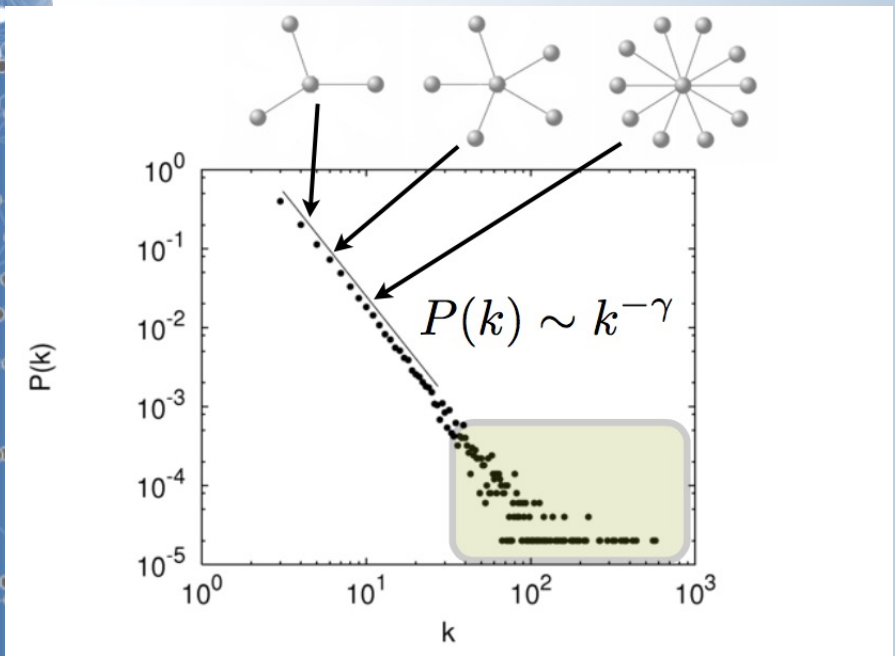
- **Question**: how to plot a probability distribution if the variable spans a large range of values, from small to (very) large?
- **Answer**: use the **logarithmic** scale
- **How to do it**: report the logarithms of the values on the x- and y-axes

powers of a number will be uniformly spaced



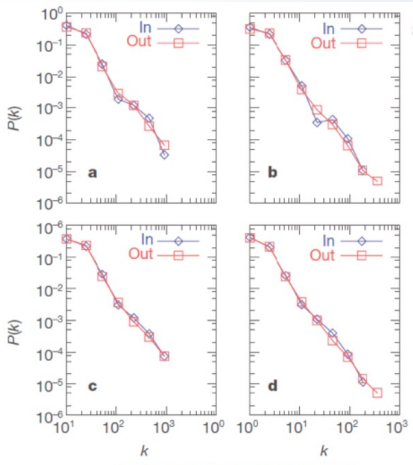
$2^0=1, 2^1=2, 2^2=4, 2^3=8, 2^4=16, 2^5=32, 2^6=64,....$

31



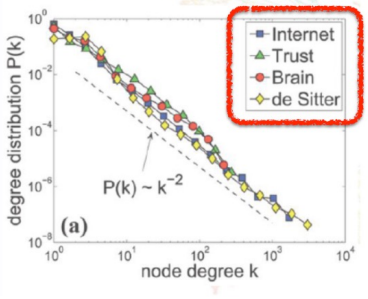
32

Examples of "scale-free" networks



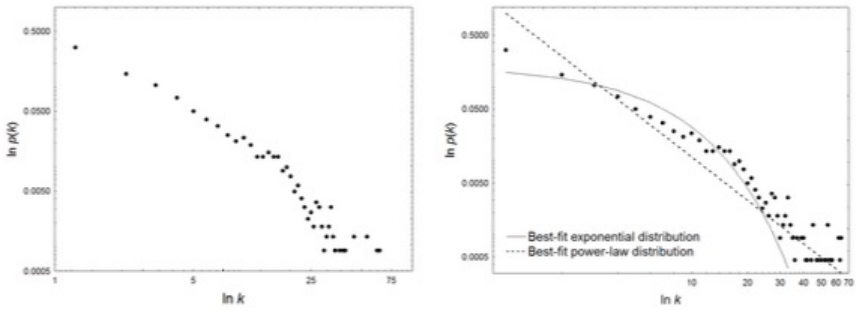
JEONG ET AL. NATURE 407, 651 (2000)

a) A. fulgidus (archaea)
b) E. coli (bacterium)
c) C. elegans (eukaryote)
d) average 43 organisms



KRIOUKOV ET AL. SCIENTIFIC REPORTS 2, 793 (2012)

33



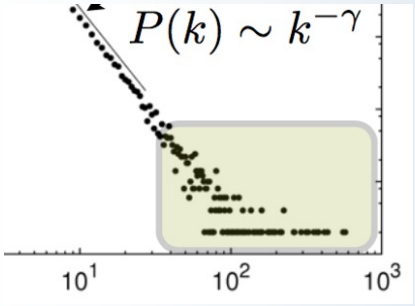
(a) Probability degree distribution (b) Fittings of the distribution

Figure 9.2: Degree distribution of a real-world network

The degree distributions real-world networks, however do not look so smooth as the ones illustrated in Figure 9.1 and there are a number of difficulties in fitting the best model to describe the experimental data. For example, the relationship between $p(k)$ and k can be erratic (see Figure 9.2a); there may not be sufficient data to fit a statistically significant model; or there may be many statistical distributions to which the same dataset can be fitted (see Figure 9.2b).

34

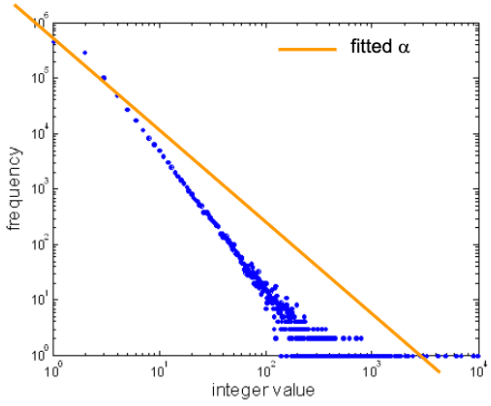
- If the variable is *not integer* (e.g., betweenness), the range of the variable is divided into intervals (bins) and we count how many values fall in each interval



35

Fitting power-laws

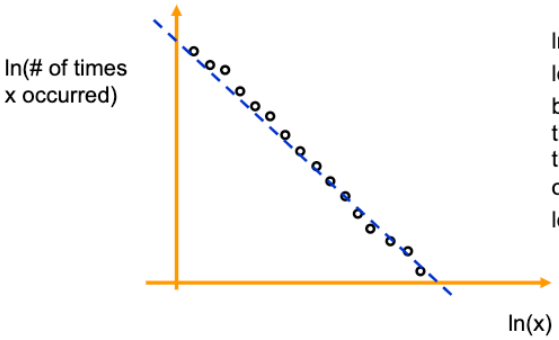
- Fitting a straight line to it via least squares regression will give values of the exponent α that are too low



36

Logarithmic binning

- Most common and not very accurate method:
  - Bin the different values of x and create a frequency histogram



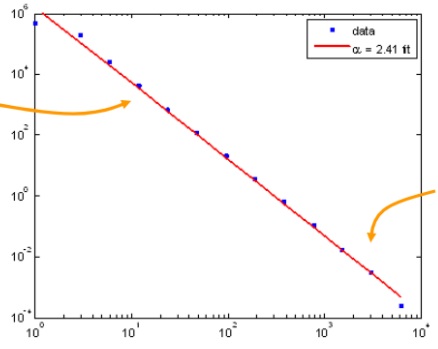
ln(x) is the natural logarithm of x, but any other base of the logarithm will give the same exponent of a because $\log_{10}(x) = \ln(x)/\ln(10)$

37

Logarithmic binning

- bin data into exponentially wider bins:
  - 1, 2, 4, 8, 16, 32, …
- normalize by the width of the bin

evenly spaced datapoints



less noise in the tail of the distribution

38

Cumulative distributions

The second is to consider the cumulative distribution function (CDF) defined as

$$P'(k) = \sum_{k'=1}^{k} p(k'). \tag{9.1}$$

This represents the probability of choosing at random a node with degree smaller than or equal to k. Common CDFs for degree distributions are

$$\text{Poisson}: \quad P'(k) = e^{-\bar{k}} \sum_{i=1}^{\lfloor k \rfloor} \frac{\bar{k}^i}{i}.$$

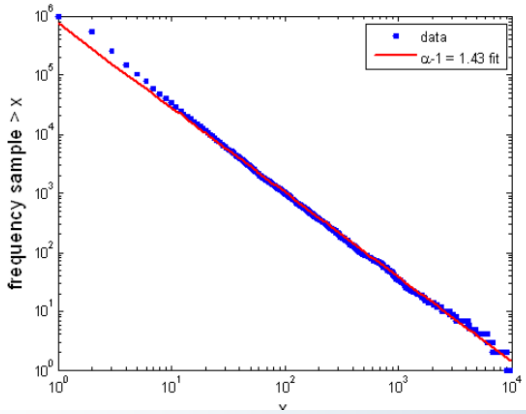
$$\text{Exponential}: \quad P'(k) = 1 - e^{-k/\bar{k}}.$$

$$\text{Powerlaw}: \quad P'(k) = 1 - k^{-\gamma+1}.$$

39

Cumulative distributions

- fitted exponent (2.43) much closer to actual (2.5)



40

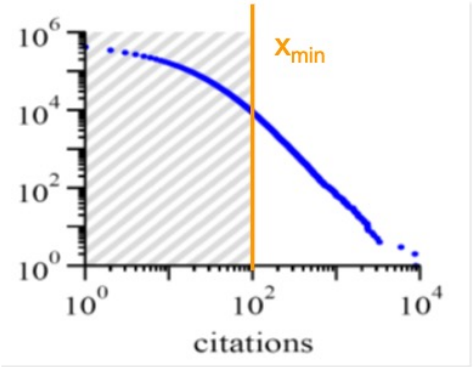
Where to start fitting?

- Some data exhibit a power law only in the tail
- After binning or taking the cumulative distribution you can fit to the tail

41

Example:

- Distribution of citations to papers
- power law is evident only in the tail
  - x_{min} > 100 citations



Source: MEJ Newman, 'Power laws, Pareto distributions and Zipf's law

42

Many real world networks are power law

	exponent α (in/out degree)
film actors	2.3
telephone call graph	2.1
email networks	1.5/2.0
sexual contacts	3.2
WWW	2.3/2.7
internet	2.5
peer-to-peer	2.1
metabolic network	2.2
protein interactions	2.4

43

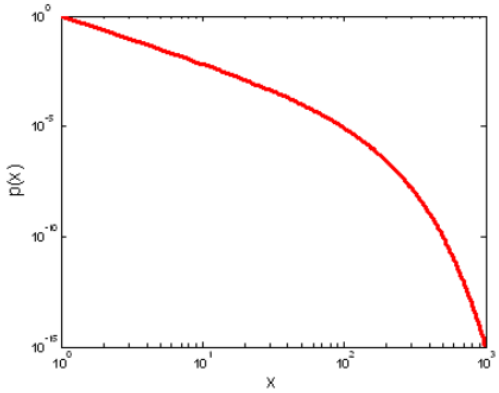
Not every network is power-law distributed

- Email network
- Board directors
- Power grid

44

Another common distribution: power-law with an exponential cutoff

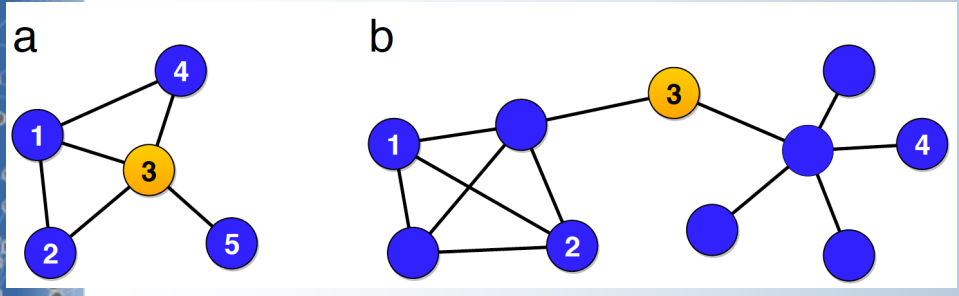
- $p(x) \sim x^{-a} e^{-x/\kappa}$



45

Degree → betweenness

Hubs usually have high betweenness, but there can be nodes with high betweenness **that are not hubs**



46

Universality

- We find the same macroscopic behavior in systems which have completely different origins
- Engineered or self-organized
- Biology, transportation, social, …
- Which are the basic underlying principles?
- Models (simple models) that can explain the basic trends

47

Robustness

OPEN ACCESS Freely available online PLOS ONE

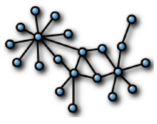
Attack Robustness and Centrality of Complex Networks

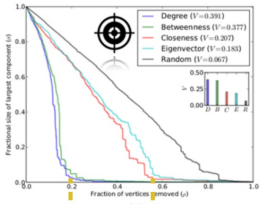
Swami Iyer[1], Timothy Killingback[2]*, Bala Sundaram[3], Zhen Wang[4]

1 Computer Science Department, University of Massachusetts, Boston, Massachusetts, United States of America, 2 Mathematics Department, University of Massachusetts, Boston, Massachusetts, United States of America, 3 Physics Department, University of Massachusetts, Boston, Massachusetts, United States of America, 4 Physics Department, University of Massachusetts, Boston, Massachusetts, United States of America

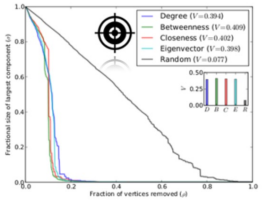
Simultaneous Sequential

Scale-free network



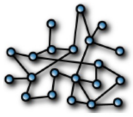


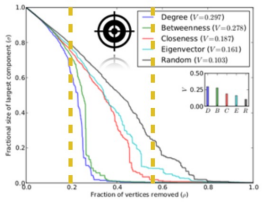
(c)

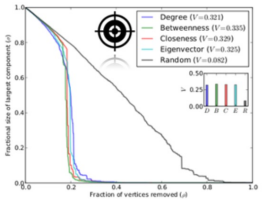


(d)

Random Graph







48

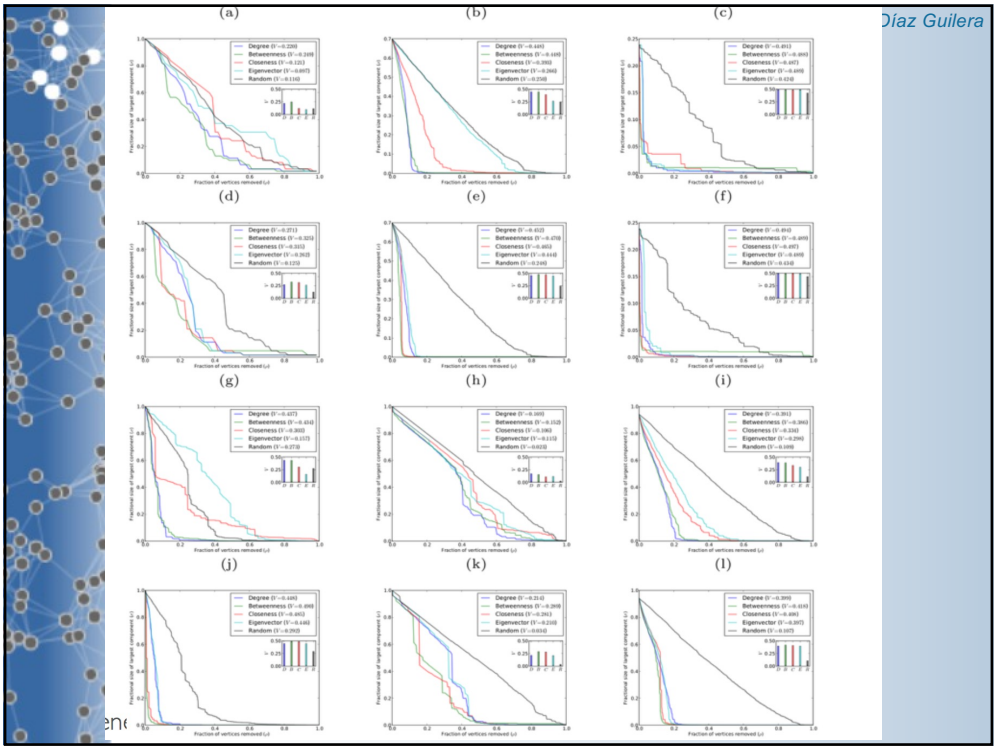
Once a suitable centrality measure has been fixed, we can compute σ as a function of ρ for removing vertices in decreasing order of that centrality measure. The robustness of a network under this type of vertex removal can be quantified by the *R- index*, which is defined by [13]

$$R = \frac{1}{N}\sum_{i=1}^{N}\sigma(i/N).$$

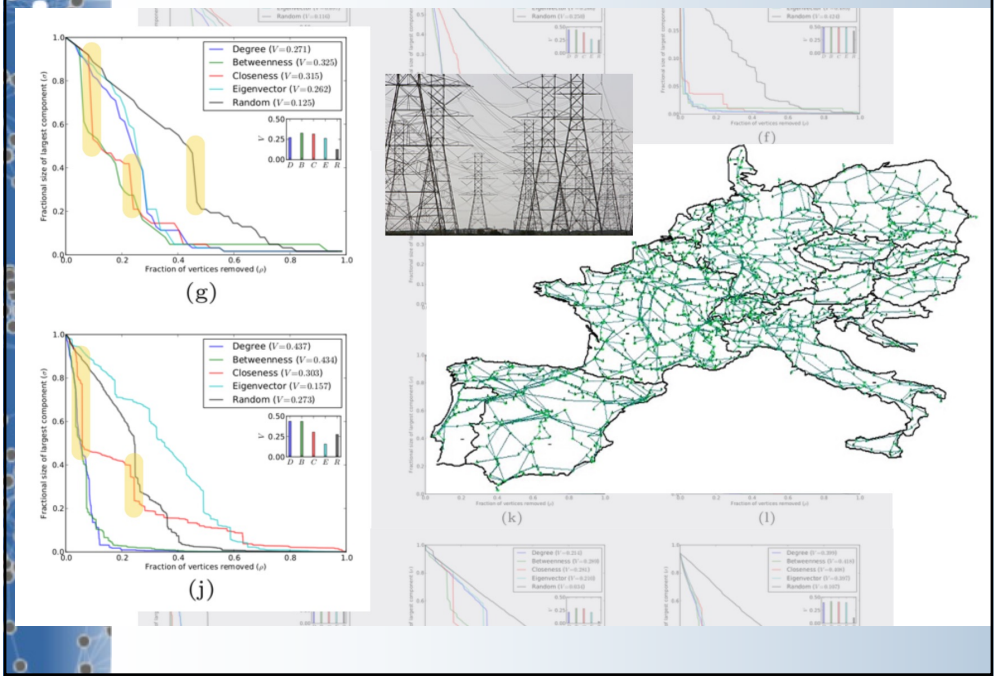
The normalization factor $1/N$ allows the robustness of networks of different sizes to be compared [13]. It is straightforward to show that for any scheme of removing vertices from any network, R attains its minimum value of $1/N$ on the star graph and its maximum value of $\frac{1}{2}(1-1/N)$ on the complete graph. Thus, for any network and method of vertex removal, $R \in [0, \frac{1}{2}]$. Consequently, we define the *V- index* V, which measures the *vulnerability* of a network to a given scheme of vertex removal, to be the complementary quantity to R,

$$V = \frac{1}{2} - R.$$

49



50



51

Robustness codebook

Chapter III - The Internet Network

Giant Component through a Breadth First Search

In [13]:
```
def giant_component_size(G_input):

    G=G_input.copy()

    components=[]

    node_list=G.nodes()

    while len(node_list)!=0:
        root_node=node_list[0]
        component_list=[]
        component_list.append(root_node)
        queue=[]
        queue.append(root_node)
        G.node[root_node]["visited"]=True
        while len(queue):
            working_node=queue.pop(0)
            for n in G.neighbors(working_node):
                #check if any node attribute exists
                if len(G.node[n])==0:
                    G.node[n]["visited"]=True
                    queue.append(n)
                    component_list.append(n)
        components.append((len(component_list),component_list))
        #remove the nodes of the component just discovered
        for i in component_list: node_list.remove(i)
    components.sort(reverse=True)

    GiantComponent=components[0][1]
    SizeGiantComponent=components[0][0]

    return GiantComponent,len(components)

(GCC, num_components)=giant_component_size(G_test)
print "Giant Connected Component:",GCC
print "Number of components:",num_components
```

```
Giant Connected Component: ['A', 'C', 'B', 'E', 'D', 'I', 'G', 'H', 'F']
Number of components: 2
```

52

Correlations

- Degree correlations: expected degree of the neighbors of a node as a function of its degree

$$k_{\mathrm{nn}}(k) = \sum_{k'} k' P(k'|k)$$

53

The correct mathematical way to quantify such a measure is the *conditioned probability* $p(k_1|k_2)$ to have a vertex with degree k_1 at one side of the edge when at the other site of the edge the degree is k_2.

We have two constraints on the conditioned probability. The first one is given by normalization condition

$$\sum_{k_1} p(k_1|k_2) = 1. \qquad (1.10)$$

For non oriented graphs the same quantity obeys the detailed balance distribution (Boguñá and Pastor-Satorras, 2002)

$$k_2 p(k_1|k_2) P(k_2) = k_1 p(k_2|k_1) P(k_1) \qquad (1.11)$$

This balance equation simply states that the number of edges going from vertex k_1 to vertex k_2 must be equal to the number of edges going from vertex k_2 to vertex k_1.

54

$P(k', k)$: Probability that two nodes of degree k and k' are linked

Detailed balance Equation for Networks

$$P(k', k) = kP(k)P(k'|k) = k'P(k')P(k|k')$$

Two ways of measuring

$$k_{nn} = \sum_{k'} k' P(k'|k) = f(k) \qquad r = \frac{\langle k_i k_j \rangle - \langle k \rangle^2}{\langle k^2 \rangle - \langle k \rangle^2}$$

55

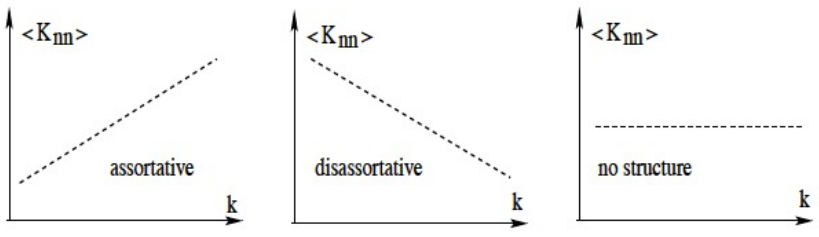
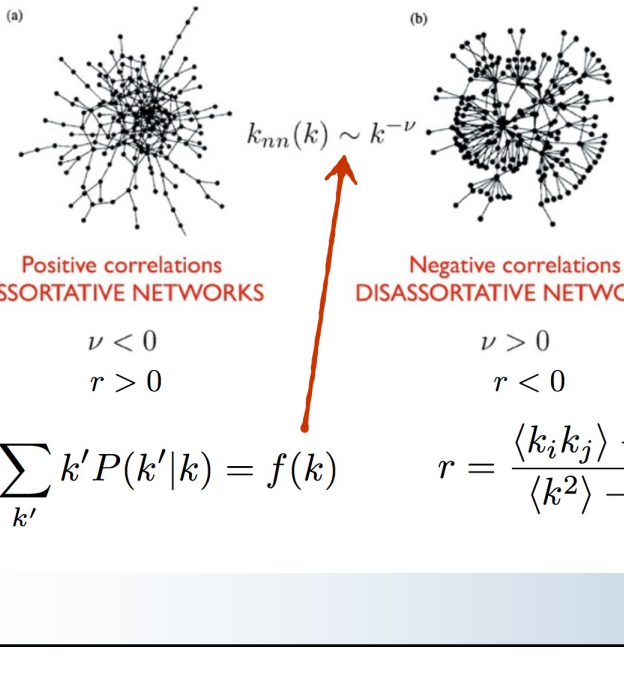
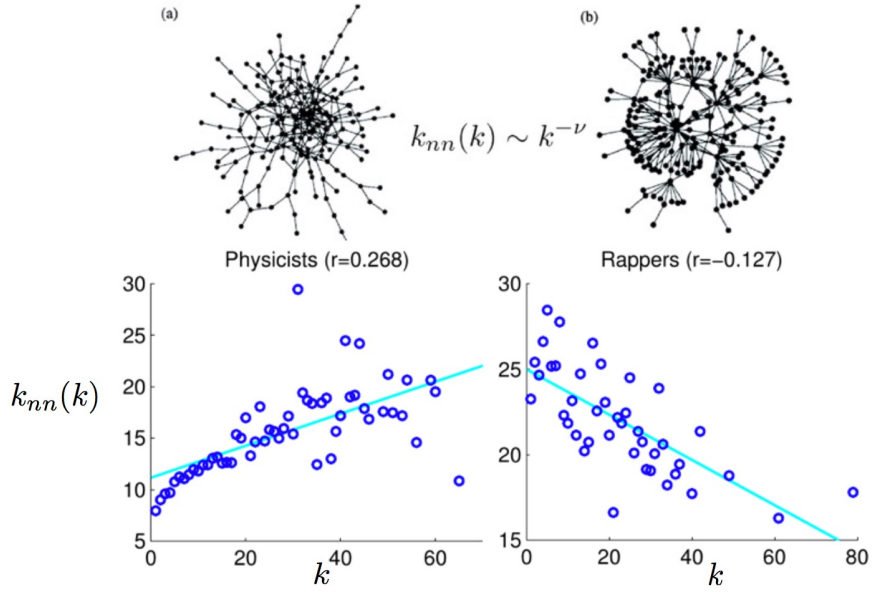


FIG. 1.10. The three possible behaviour of the average degree of the neighbours $\langle K_{nn} \rangle$ versus the degree k of the vertex origin.

56


(a)
(b)
$k_{nn}(k) \sim k^{-\nu}$
Positive correlations
ASSORTATIVE NETWORKS
Negative correlations
DISASSORTATIVE NETWORKS
$\nu < 0$
$r > 0$
$\nu > 0$
$r < 0$
$k_{nn} = \sum_{k'} k' P(k'|k) = f(k)$
$r = \frac{\langle k_i k_j \rangle - \langle k \rangle^2}{\langle k^2 \rangle - \langle k \rangle^2}$

57


(a)
(b)
$k_{nn}(k) \sim k^{-\nu}$
Physicists (r=0.268)
Rappers (r=–0.127)
$k_{nn}(k)$
k
k

58

Network taxonomy

	Network	Type	Nodes	Links	<k>	L		Clustering		Corr. (r)
Social	film actors	undirected	449 913	25 516 482	113.43	3.48	2.3	0.20	0.78	0.208
	company directors	undirected	7 673	55 392	14.44	4.60	–	0.59	0.88	0.276
	math coauthorship	undirected	253 339	496 489	3.92	7.57	–	0.15	0.34	0.120
	physics coauthorship	undirected	52 909	245 300	9.27	6.19	–	0.45	0.56	0.363
	biology coauthorship	undirected	1 520 251	11 803 064	15.53	4.92	–	0.088	0.60	0.127
	telephone call graph	undirected	47 000 000	80 000 000	3.16		2.1			
	email messages	directed	59 912	86 300	1.44	4.95	1.5/2.0		0.16	
	email address books	directed	16 881	57 029	3.38	5.22	–	0.17	0.13	0.092
	student relationships	undirected	573	477	1.66	16.01	–	0.005	0.001	−0.029
	sexual contacts	undirected	2 810				3.2			
Information	WWW nd.edu	directed	269 504	1 497 135	5.55	11.27	2.1/2.4	0.11	0.29	−0.067
	WWW Altavista	directed	203 549 046	2 130 000 000	10.46	16.18	2.1/2.7			
	citation network	directed	783 339	6 716 198	8.57		3.0/–			
	Roget's Thesaurus	directed	1 022	5 103	4.99	4.87	–	0.13	0.15	0.157
	word co-occurrence	undirected	460 902	17 000 000	70.13		2.7		0.44	
Technological	Internet	undirected	10 697	31 992	5.98	3.31	2.5	0.035	0.39	−0.189
	power grid	undirected	4 941	6 594	2.67	18.99	–	0.10	0.080	−0.003
	train routes	undirected	587	19 603	66.79	2.16	–		0.69	−0.033
	software packages	directed	1 439	1 723	1.20	2.42	1.6/1.4	0.070	0.082	−0.016
	software classes	directed	1 377	2 213	1.61	1.51	–	0.033	0.012	−0.119
	electronic circuits	undirected	24 097	53 248	4.34	11.05	3.0	0.010	0.030	−0.154
	peer-to-peer network	undirected	880	1 296	1.47	4.28	2.1	0.012	0.011	−0.366
Biological	metabolic network	undirected	765	3 686	9.64	2.56	2.2	0.090	0.67	−0.240
	protein interactions	undirected	2 115	2 240	2.12	6.80	2.4	0.072	0.071	−0.156
	marine food web	directed	135	598	4.43	2.05	–	0.16	0.23	−0.263
	freshwater food web	directed	92	997	10.84	1.90	–	0.40	0.48	−0.326
	neural network	directed	307	2 359	7.68	3.97	–	0.18	0.28	−0.226

59

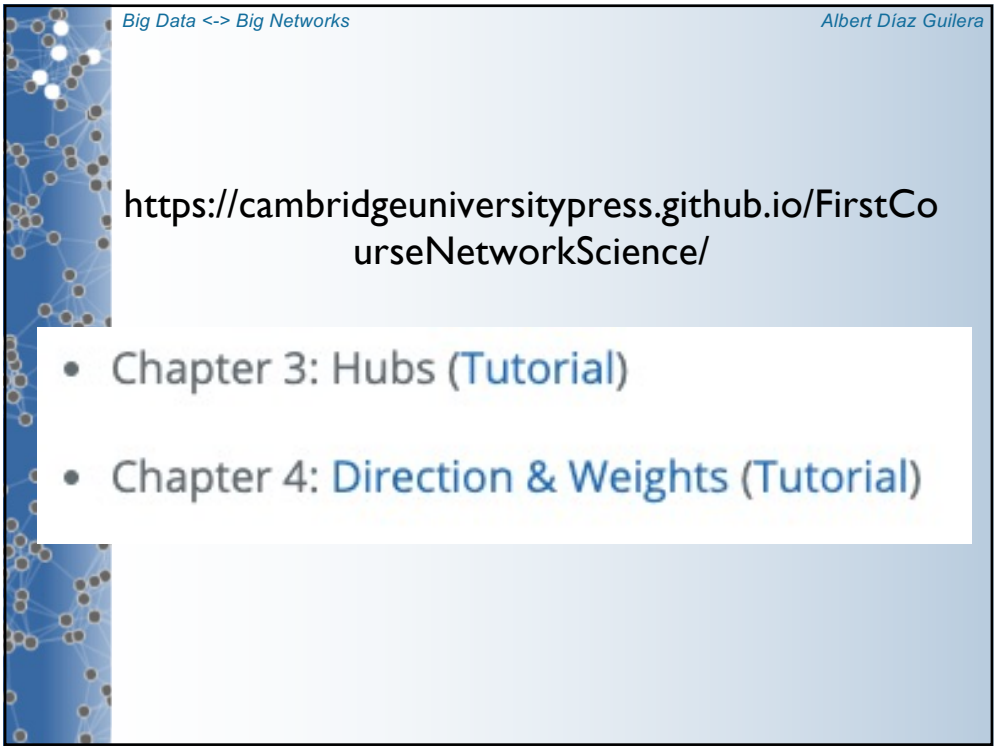
Assortativity

`degree_assortativity_coefficient` (G[, x, y, ...])	Compute degree assortativity of graph.
`attribute_assortativity_coefficient` (G, attribute)	Compute assortativity for node attributes.
`numeric_assortativity_coefficient` (G, attribute)	Compute assortativity for numerical node attributes.
`degree_pearson_correlation_coefficient` (G[, ...])	Compute degree assortativity of graph.

Average neighbor degree

`average_neighbor_degree` (G[, source, target, ...])	Returns the average degree of the neighborhood of each node.

62


Big Data <-> Big Networks
Albert Díaz Guilera
https://cambridgeuniversitypress.github.io/FirstCourseNetworkScience/
• Chapter 3: Hubs (Tutorial)
• Chapter 4: Direction & Weights (Tutorial)

63