Lecture 6:
Application Paper: The air Transportation Network
Guimera et al. PNAS (2005)

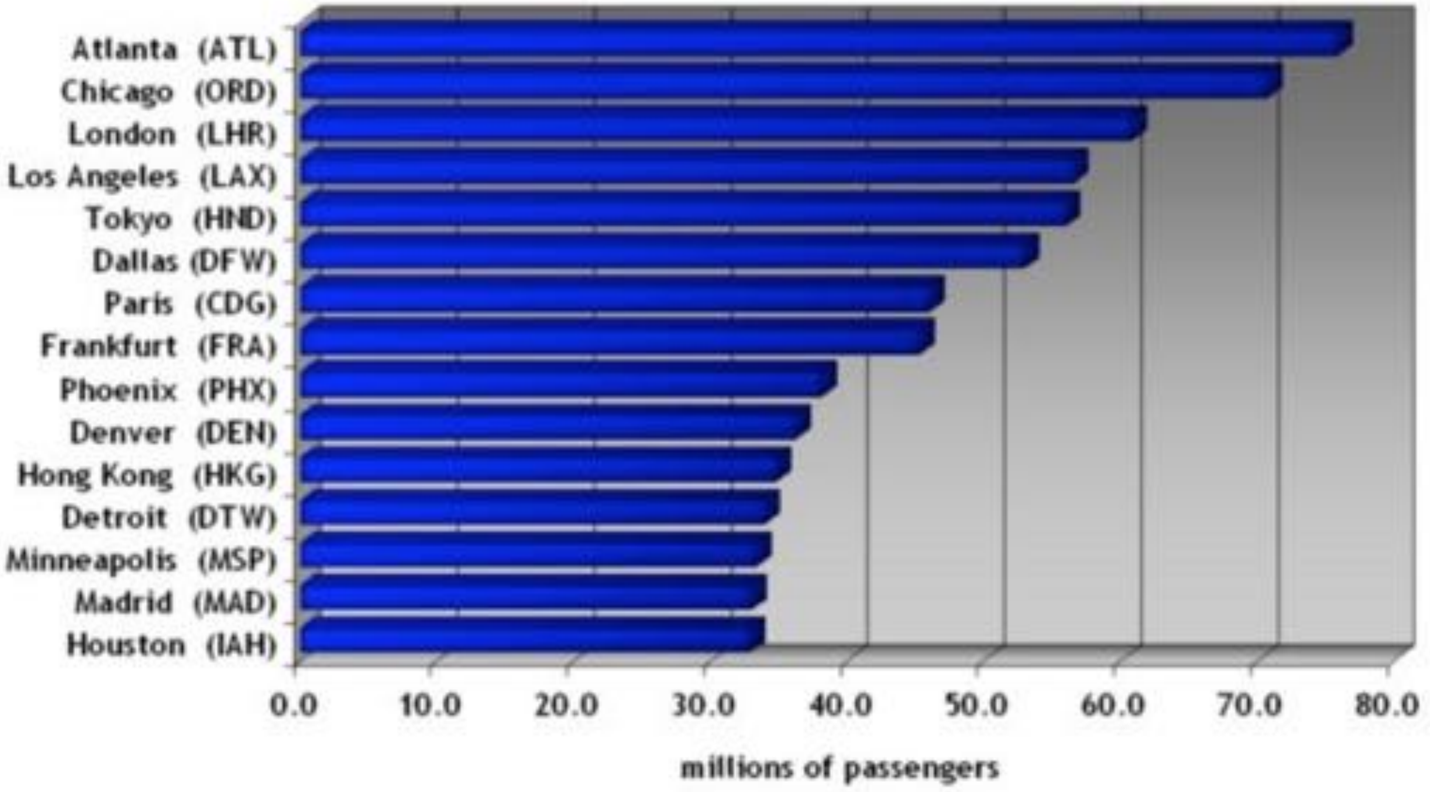
Summary of Unit: Centrality and Community Detection

The worldwide air transportation network: Anomalous centrality, community structure, and cities' global roles

R. Guimerà*, S. Mossa†, A. Turtschi‡, and L. A. N. Amaral*§

WORLD AIRPORTS

ANNUAL TRAFFIC *(IATA 2002)*



A metric of influential spreading during contagion dynamics through the air transportation network.
C. Nicolaides, L. Cueto-Felgueroso, M. C. Gonzalez and R. Juanes,
PLoS ONE, 7(7), e40961 (2012)

Passangers per year world wide?

700 million passengers each year

Percentage of US flights in/out Chicago?

Chicago (10% of the total commercial flights)

How many?

~2,700 daily

blue-sky day ----- landing rate 100 per hour

low-clouds days ----- landing rate 72 landing per hour

Flight delay costs in 1999 --- 150 to 200 billion Euro

Impact in economies

Disease Spreading (SARS or influenza)

Much research has been done in optimal network design

System level analysis >> Network Analysis

3,883 locations (villages towns and cities) (OAG max database)
Nov. 1 200 to Oct 31 2001

>800 of the world’s airlines

531,574 unique non-stop passenger flights

27,051 city pairs have non stop connections

Symmetric network

Giant component 3,663 with d=4.4

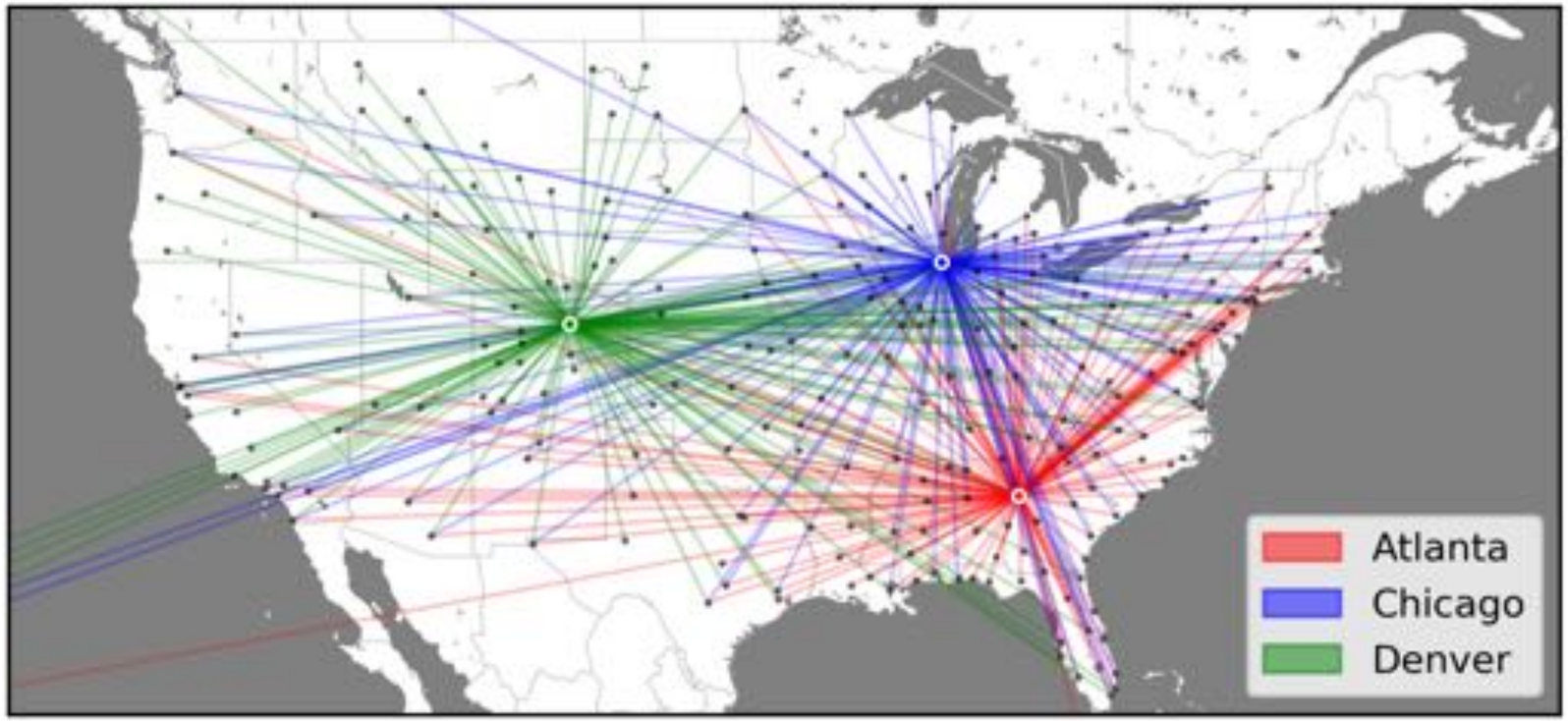
56% of the cities are connected by 4 steps or less

C=0.62, whereas its randomized version C=0.049

Worldwide air transportation network is Small World

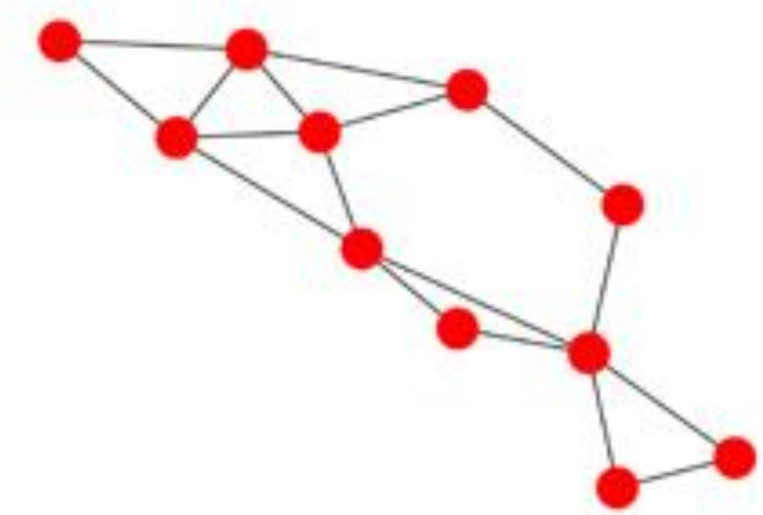
The farthest cities are Mount Pleasant in the Falkand Islands
And Wasu, Papau, New Guinea (15)

Real networks are heterogeneous

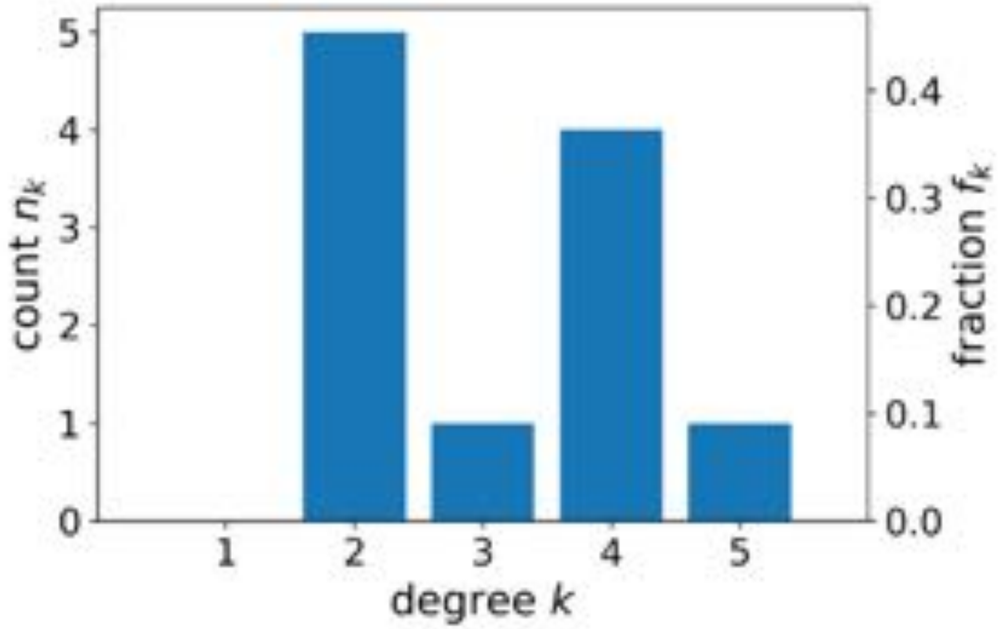


Some nodes (and links) are much more important (**central**) than others!

Centrality distributions

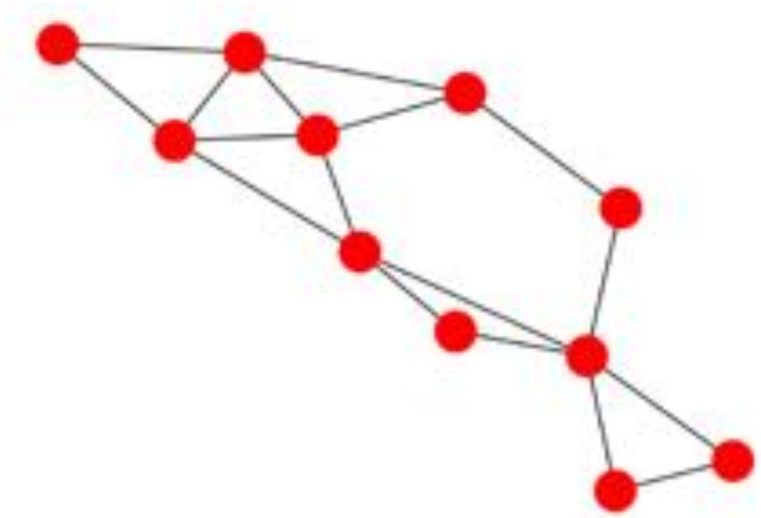


Histogram

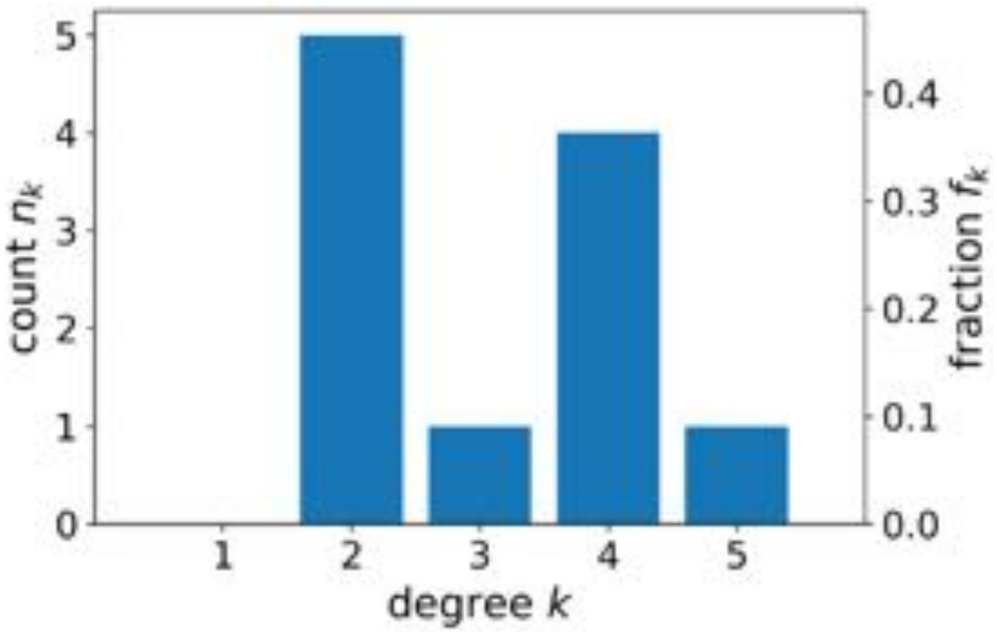


- $n_k =$ **number of nodes with degree** k
- $f_k = \frac{n_k}{N} =$ **frequency of degree** k

Centrality distributions



Histogram



- When $N \rightarrow \infty$, f_k becomes the **probability** p_k of having degree k
- p_k versus k is the **probability distribution** of node degree

Cumulative distributions

- 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

- **Cumulative distribution *P(x)***: probability that the variable takes values *larger* than *x* as a function of *x*

- **How to compute it:** by summing the frequencies of the variable inside the intervals to the right of *x*

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

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

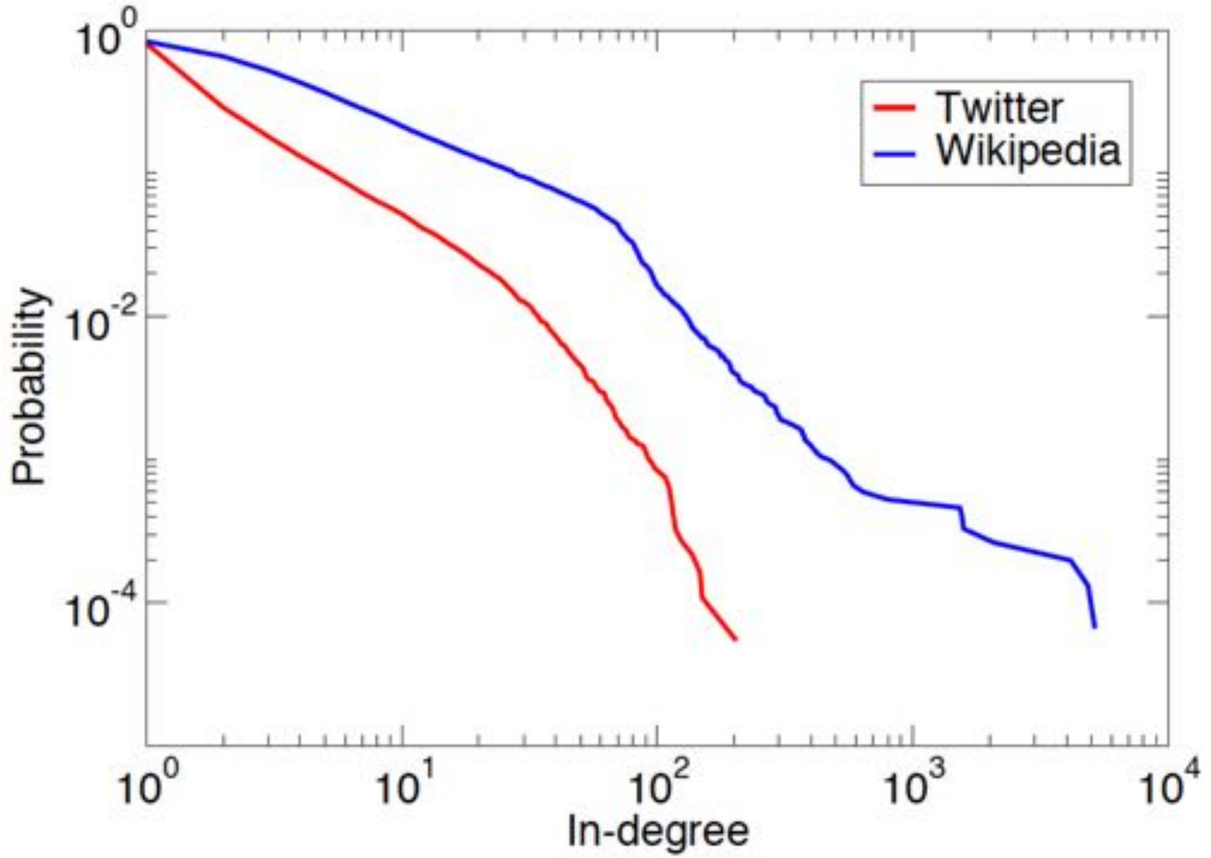
- **How to do it**: report the logarithms of the values on the x- and y-axes

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

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

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

Degree distributions



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

Degree distributions

- The **heterogeneity parameter $\boldsymbol{\kappa}$** says how broad the distribution is:

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

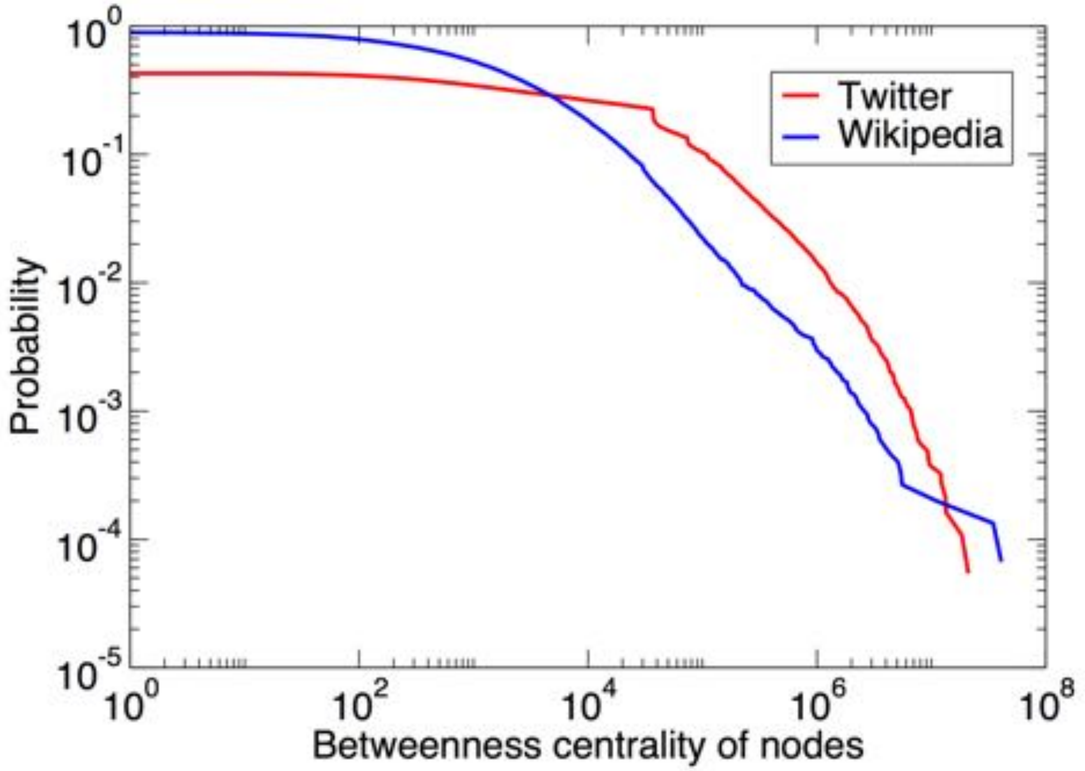
$$\langle k \rangle = \frac{\sum_i k_i}{N} = \frac{2L}{N}; \langle k^2 \rangle = \frac{\sum_i k_i^2}{N}$$

- If most degrees have the same value, say k_0 :

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

- If the distribution is very heterogeneous: $\kappa \gg 1$

Betweenness distributions

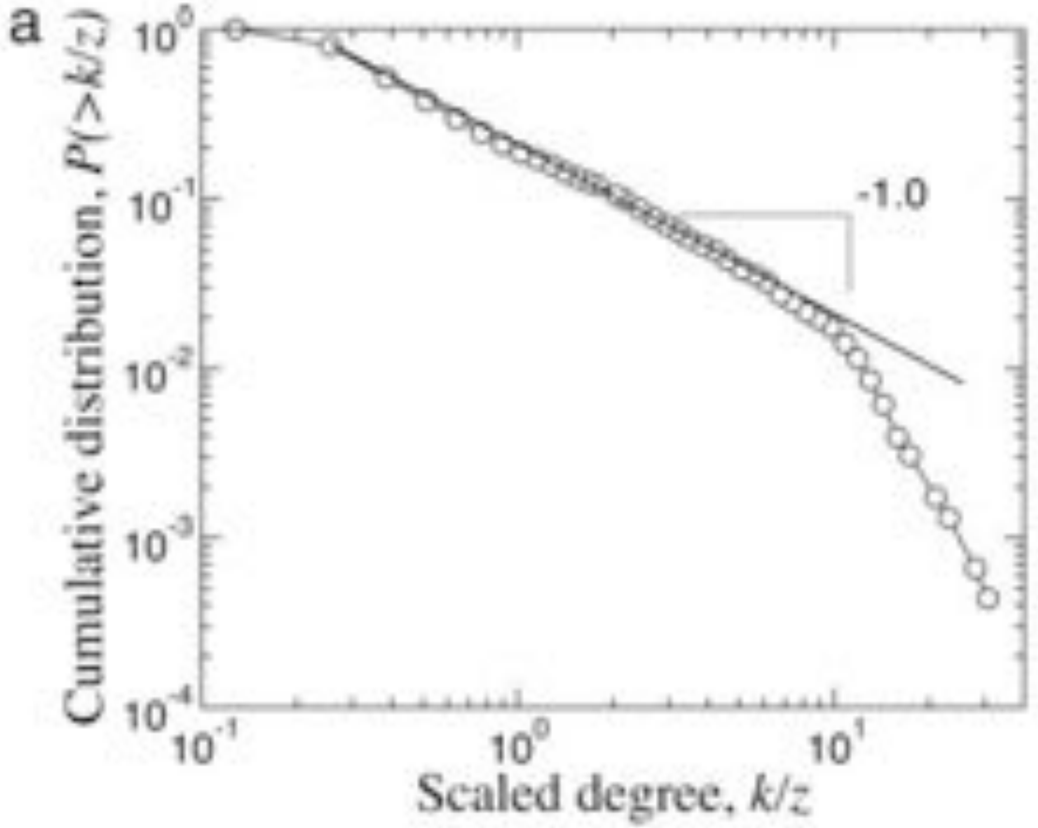


Heavy-tail distribution: the variable goes from small to large values

Degree centrality

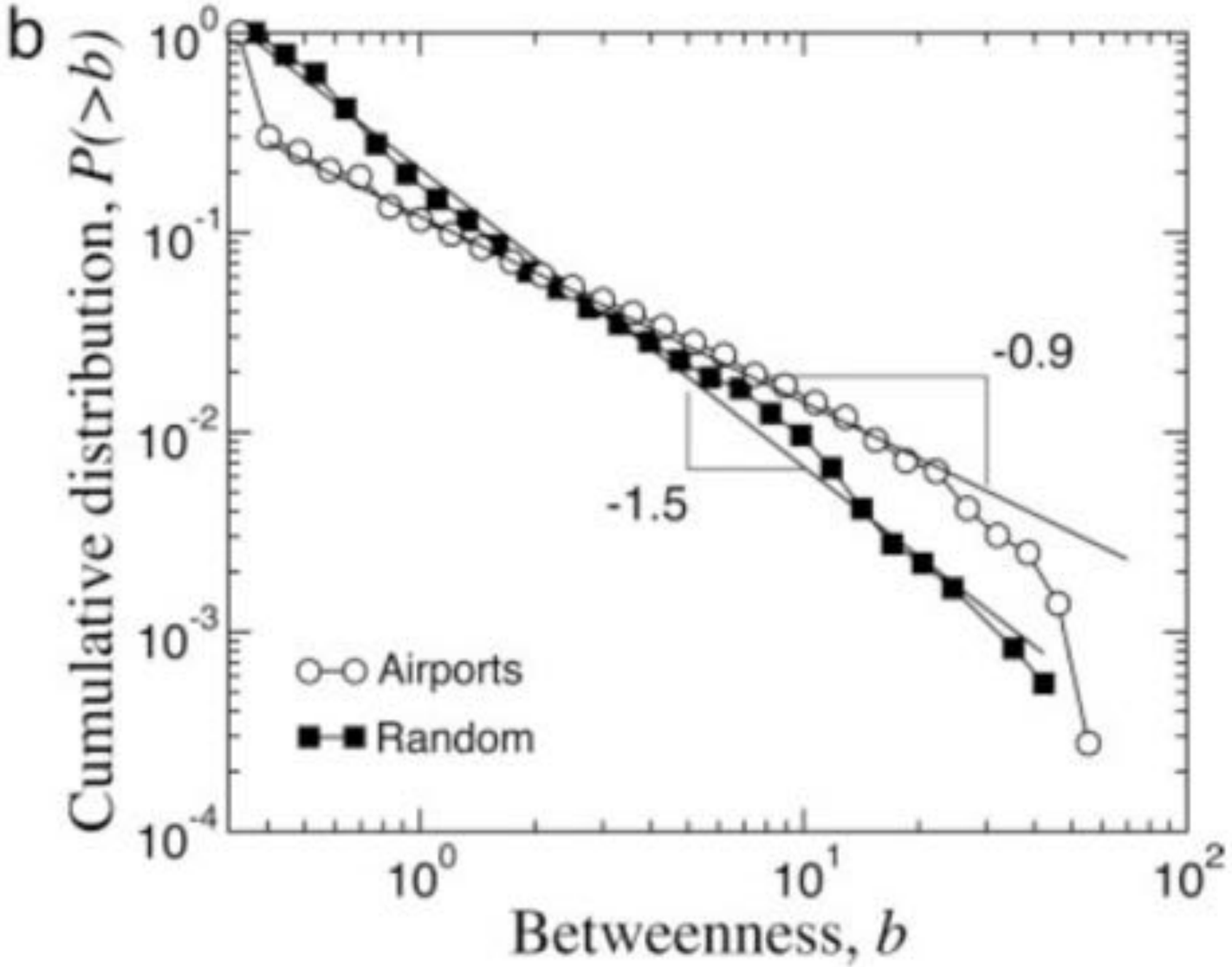
Network	Nodes (N)	Links (L)	Average degree ($\langle k \rangle$)	Maximum degree (k_{max})	Heterogeneity parameter (κ)
Facebook Northwestern Univ.	10,567	488,337	92.4	2,105	1.8
IMDB movies and stars	563,443	921,160	3.3	800	5.4
IMDB co-stars	252,999	1,015,187	8.0	456	4.6
Twitter US politics	18,470	48,365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

Cummulative Degree distribution



$$P(>k) \propto k^{-\alpha} f(k/k_{\times}) \qquad \alpha = 1.0 \pm 0.1$$

aral, L. A. N., Scala, A., Barthelemy, M. & Stanley, H. E. (2000) Proc. Natl. Acad. Sci. USA 97, 11149–11152.

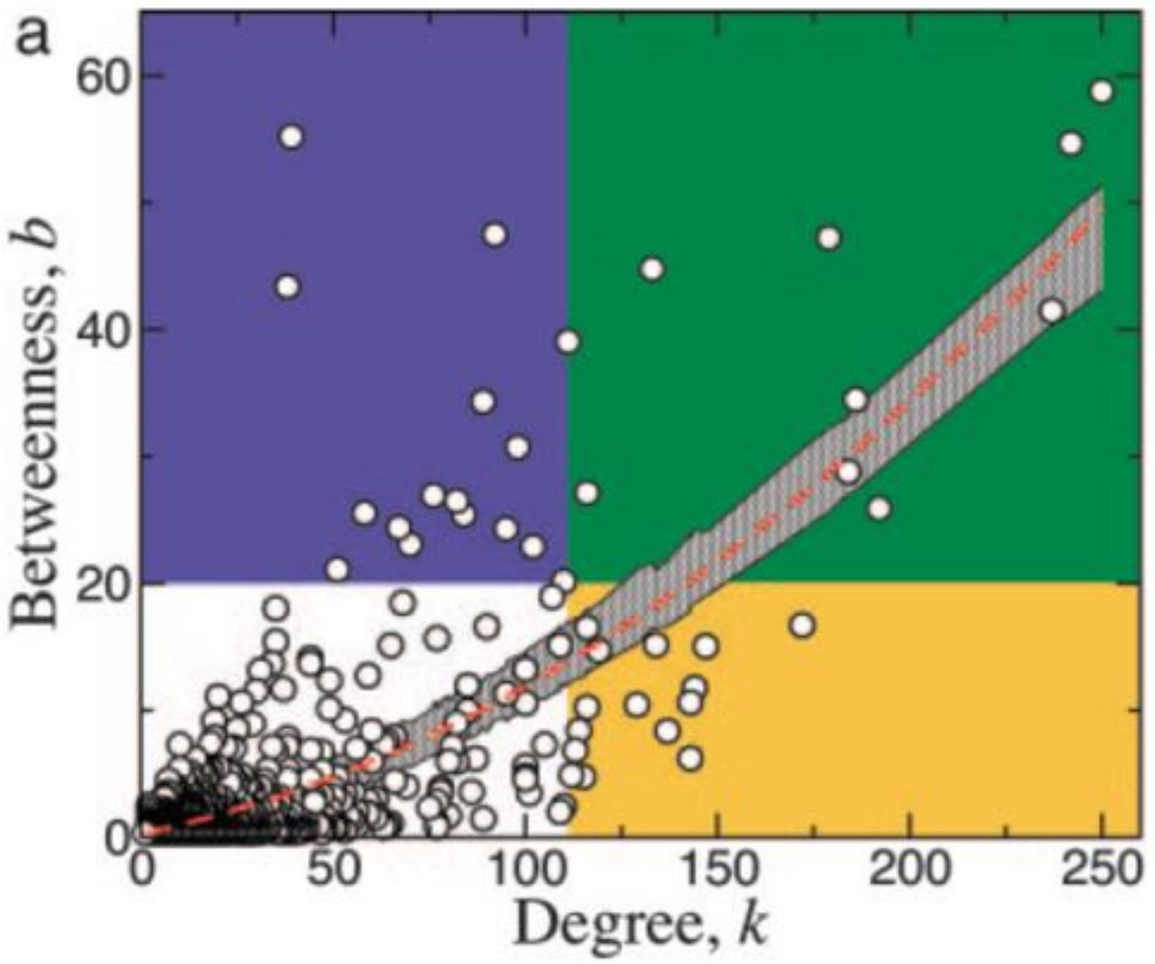


$$P(>b) \propto b^{-\nu} g(b/b_{\times})$$

$$\nu = 0.9 \pm 0.1$$

***bi*= *Bi*/<B>, where <B> represents theaverage betweenness for the network.**

Are the most connected cities also the most central?



Most Connected Cities

Most Central Cities

Table 2. The 25 most central cities in the worldwide air transportation network

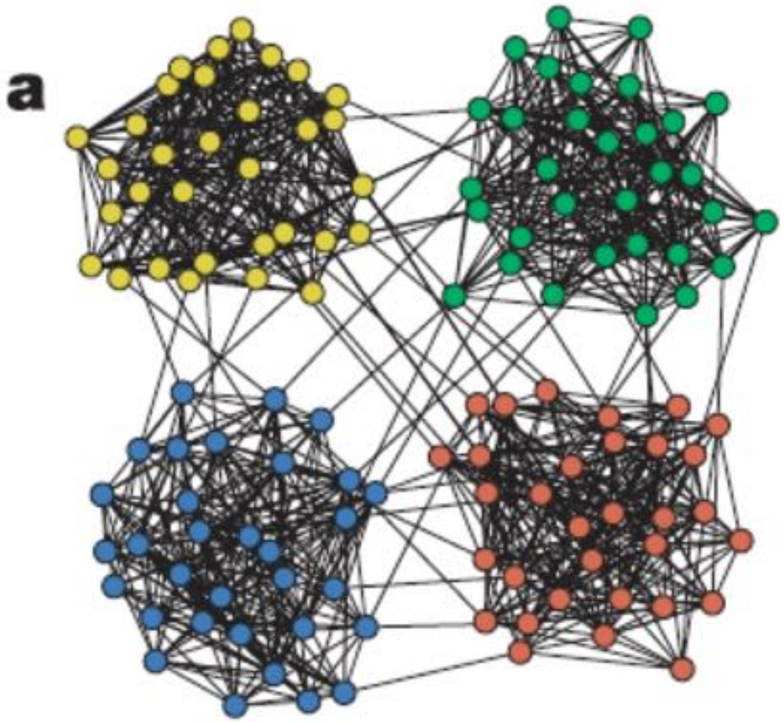
Rank	City	b	b/b_{ran}	Degree
1	Paris	58.8	1.2	250
2	Anchorage*	55.2	16.7	39
3	London	54.7	1.2	242
4	Singapore*	47.5	4.3	92
5	New York	47.2	1.6	179
6	Los Angeles	44.8	2.3	133
7	Port Moresby*	43.4	13.6	38
8	Frankfurt	41.5	0.9	237
9	Tokyo	39.1	2.7	111
10	Moscow	34.5	1.1	186
11	Seattle*	34.3	3.3	89
12	Hong Kong*	30.8	2.6	98
13	Chicago	28.8	1.0	184
14	Toronto	27.1	1.8	116
15	Buenos Aires*	26.9	3.2	76
16	São Paulo*	26.5	2.8	82
17	Amsterdam	25.9	0.8	192
18	Melbourne*	25.5	4.5	58
19	Johannesburg*	25.4	2.6	84
20	Manila*	24.4	3.5	67
21	Seoul*	24.3	2.1	95
22	Sydney*	23.1	3.2	70
23	Bangkok*	22.9	1.8	102
24	Honolulu*	21.1	4.4	51
25	Miami*	20.1	1.4	110

Cities are ordered according to their normalized betweenness. We also show the ratio of the actual betweenness of the cities to the betweenness that they have after randomizing the network.

*These cities are not among the 25 most connected.

Table 1. Number of locations with airports by major geographic region

Region	No. of locations
Africa	364
Asia and Middle East	719
Europe	691
Latin America	523
North America	1,064
Oceania	522



Finding and evaluating community structure in networks,
M. E. J. Newman and M. Girvan,
Phys. Rev. E 69, 026113 (2004).

http://arxiv.org/abs/condmat/0308217

1. Calculate betweenness scores for all edges in the network.
2. Find the edge with the highest score and remove it from the network.
3. Recalculate betweenness for all remaining edges.
4. Repeat from step 2.

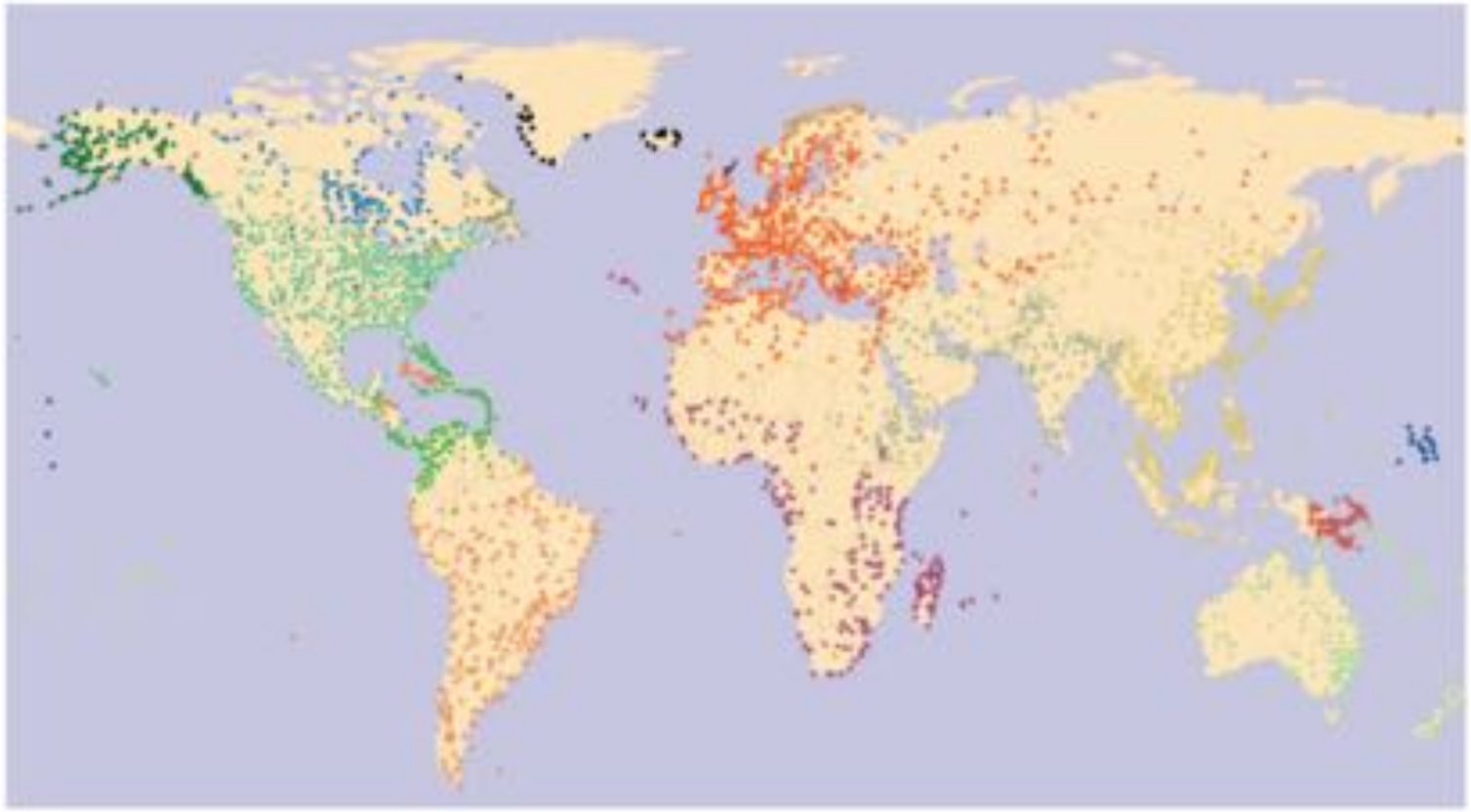


Fig. 3. Communities in the giant component of the worldwide air transportation network. Each node represents a location, and each color corresponds to a community.

Z-score

$$z_i = \frac{\kappa_i - \bar{\kappa}_{s_i}}{\sigma_{\kappa_{s_i}}},$$

κ_i the within community degree of a node minus the average divided by the standard deviation

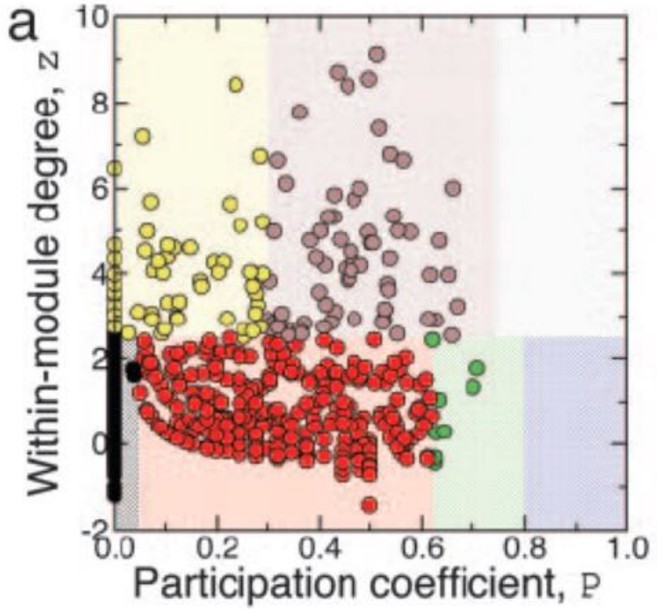
$\bar{\kappa}_{s_i}$ is the average of κ over all of the nodes in s_i

Participation

$$P_i = 1 - \sum_{s=1}^{N_M} \left(\frac{\kappa_{is}}{k_i} \right)^2,$$

The participation coefficient of a node is therefore close to one if its links are uniformly distributed among all of the communities and zero if all its links are within its own community.

κ_{iS} is the number of links of node i into community s



$$z_i = \frac{\kappa_i - \bar{\kappa}_{s_i}}{\sigma_{\kappa_{s_i}}},$$

$$P_i = 1 - \sum_{s=1}^{N_M} \left(\frac{\kappa_{is}}{k_i} \right)^2,$$

Nonhubs

- R1 ultraperipheral
- R2 peripheral
- R3 nonhub connectors
- R4 nonhub kinless nodes

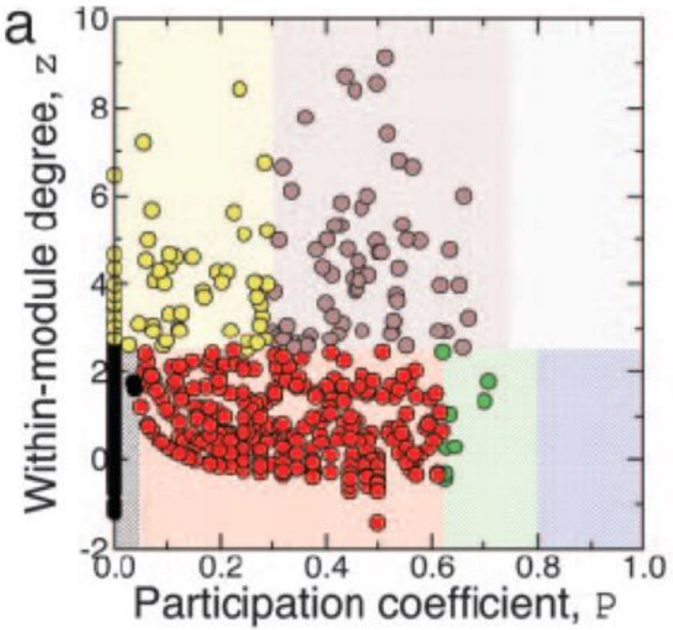
Hubs

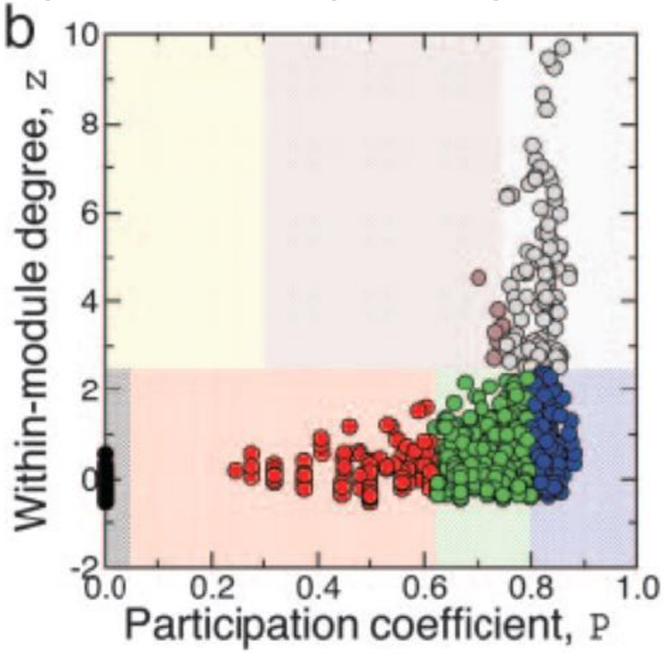
- R5 provincial hubs
- R6 connector hubs
- R7 kinless hubs

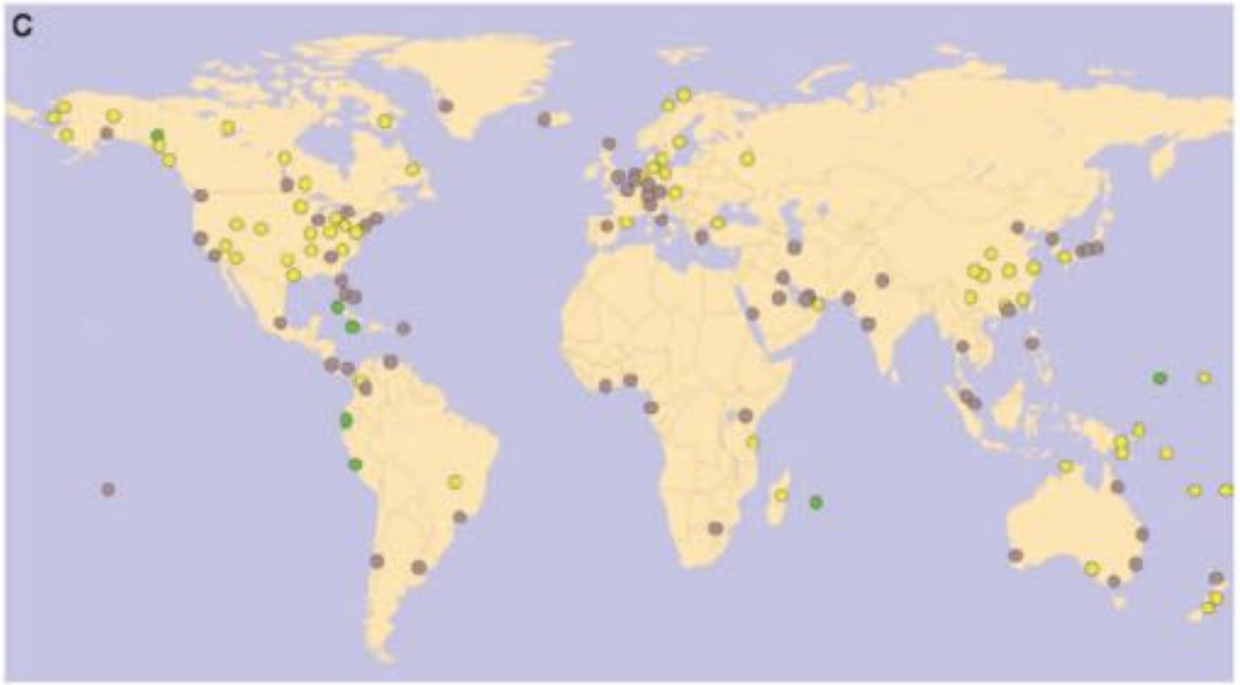
We divided nonhub nodes into four different roles as follows: (R1) "ultraperipheral nodes," i.e., nodes with all their links within their module ($P \leq 0.05$); (R2) "peripheral nodes," i.e., nodes with most links within their module ($0.05 < P \leq 0.62$); (R3) "nonhub connector nodes," i.e., nodes with many links to other modules ($0.62 < P \leq 0.80$); and (R4) "nonhub kinless nodes," i.e., nodes with links homogeneously distributed among all modules ($P > 0.80$).

We divided hub nodes into three different roles as follows: (R5) "provincial hubs," i.e., hub nodes with the vast majority of links within their module ($P \leq 0.30$); (R6) "connector hubs," i.e., hubs with many links to most of the other modules ($0.30 < P \leq 0.75$); and (R7) "kinless hubs," i.e., hubs with links homogeneously distributed among all modules ($P > 0.75$).

Randomly selecting Links keeping the degree of airports







nonhub connectors

provincial hubs

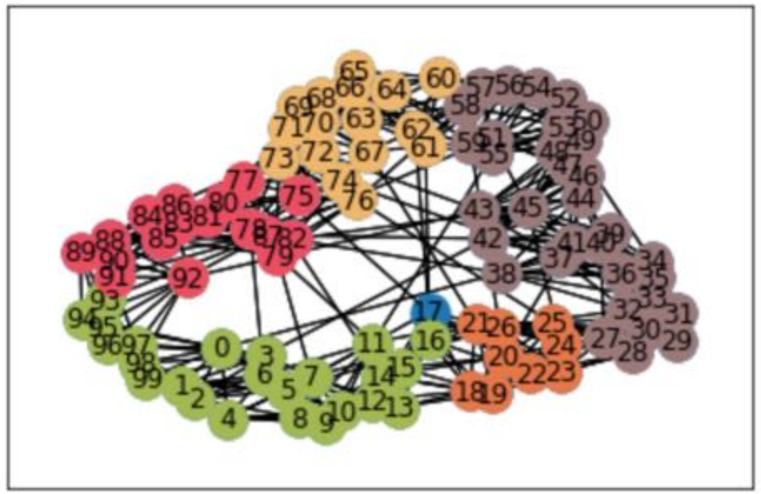
connector hubs

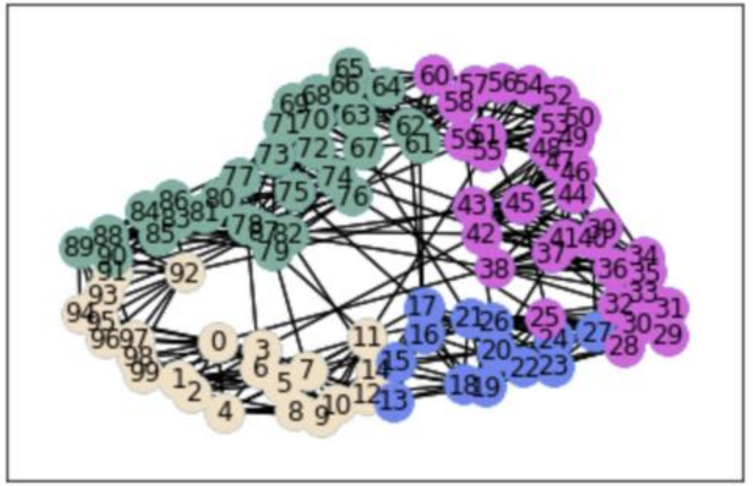
For a Review of Many Applications for your Project see:

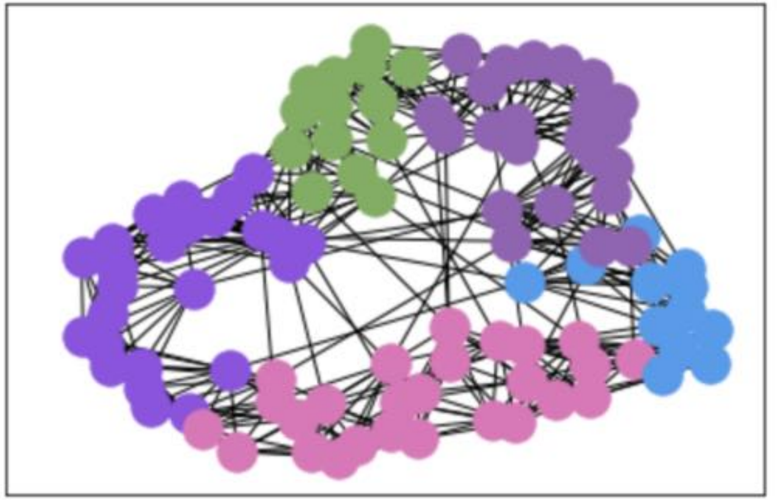
Marc Barthelemy, **Spatial Networks**.
Physics Reports, **499**, 1–101, (2011).

Networksand Space
Empirical Observations
Models
Processes on Networks

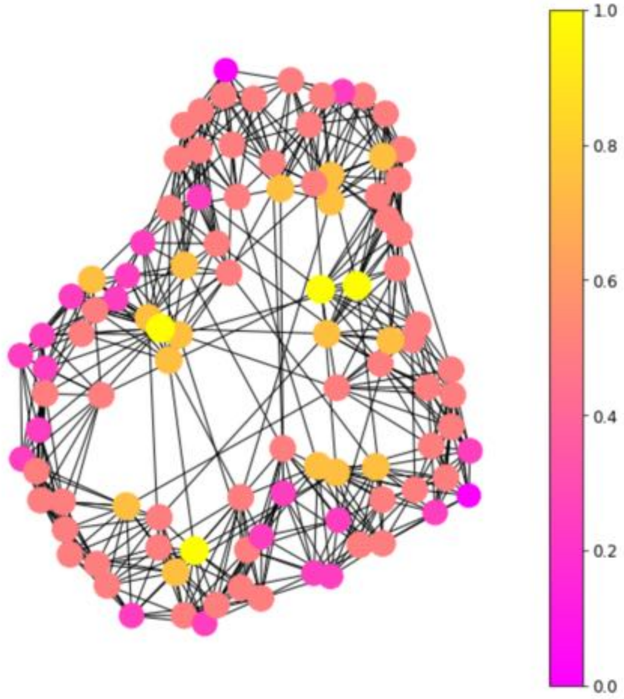
See Assignment2_exercise.ipynb



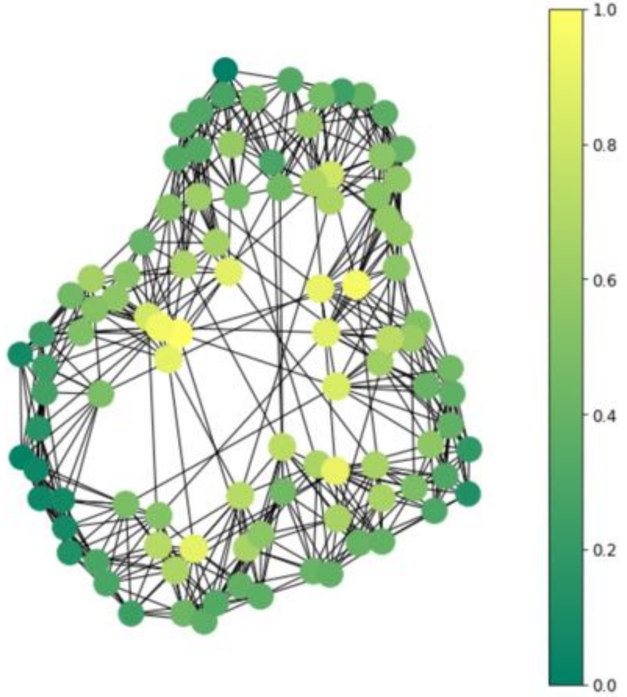




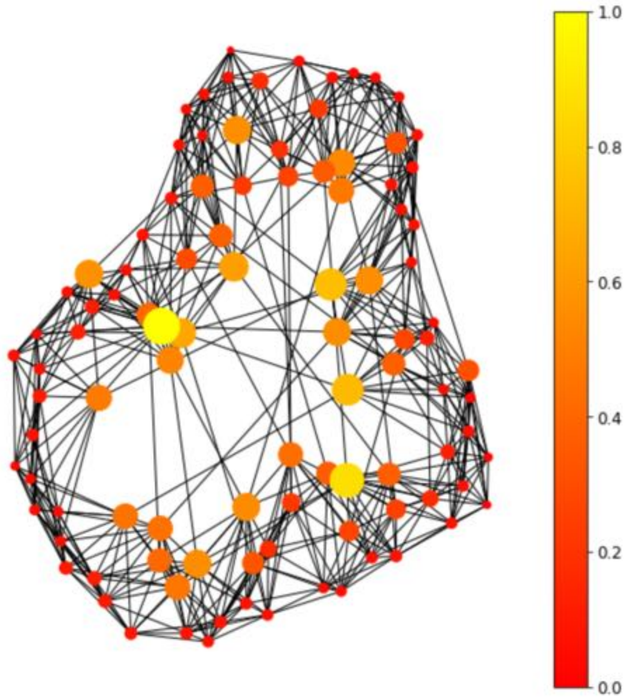
Closeness



Degree Centrality

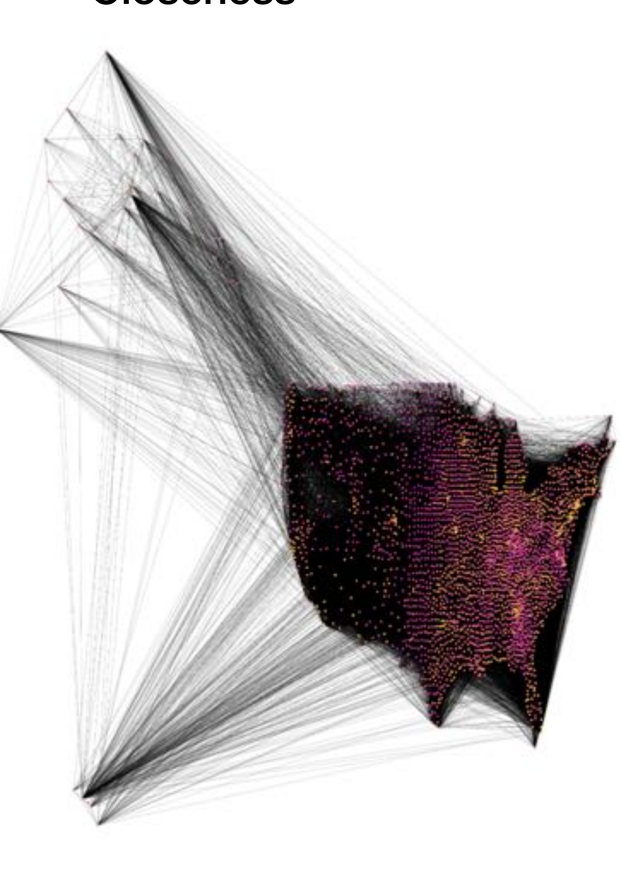


Betweenness Centrality

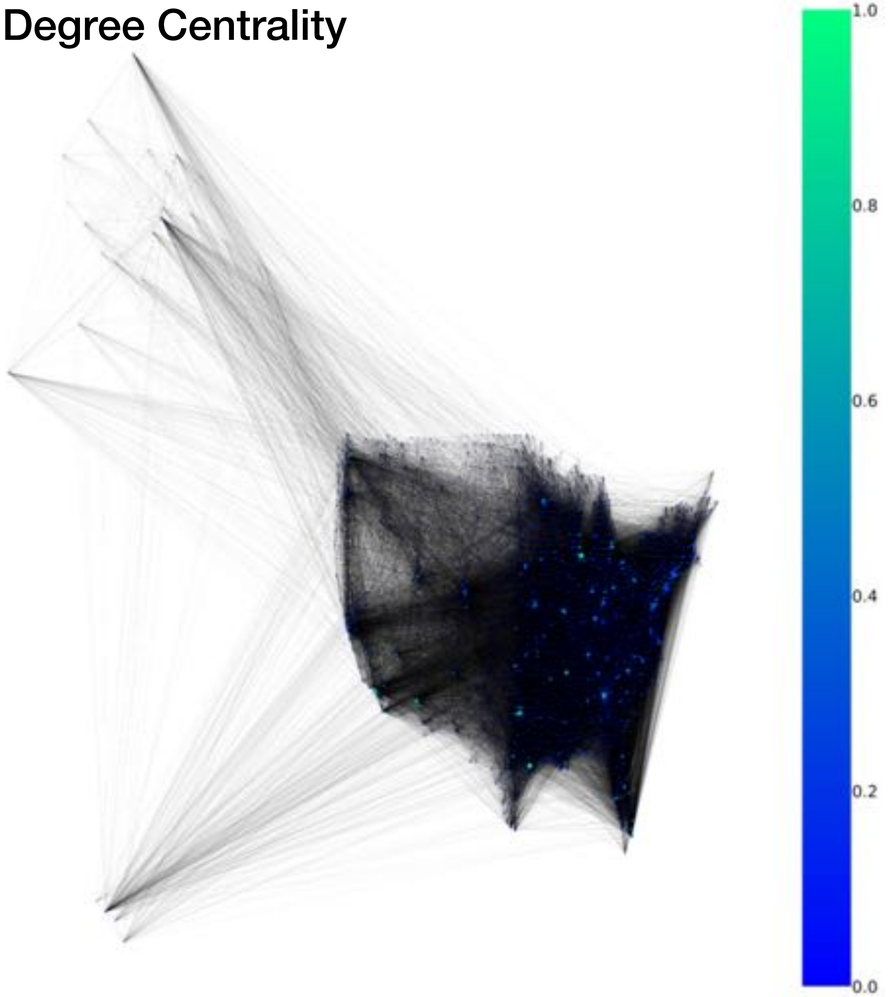


Modify it for the calculations of the Assignment 2

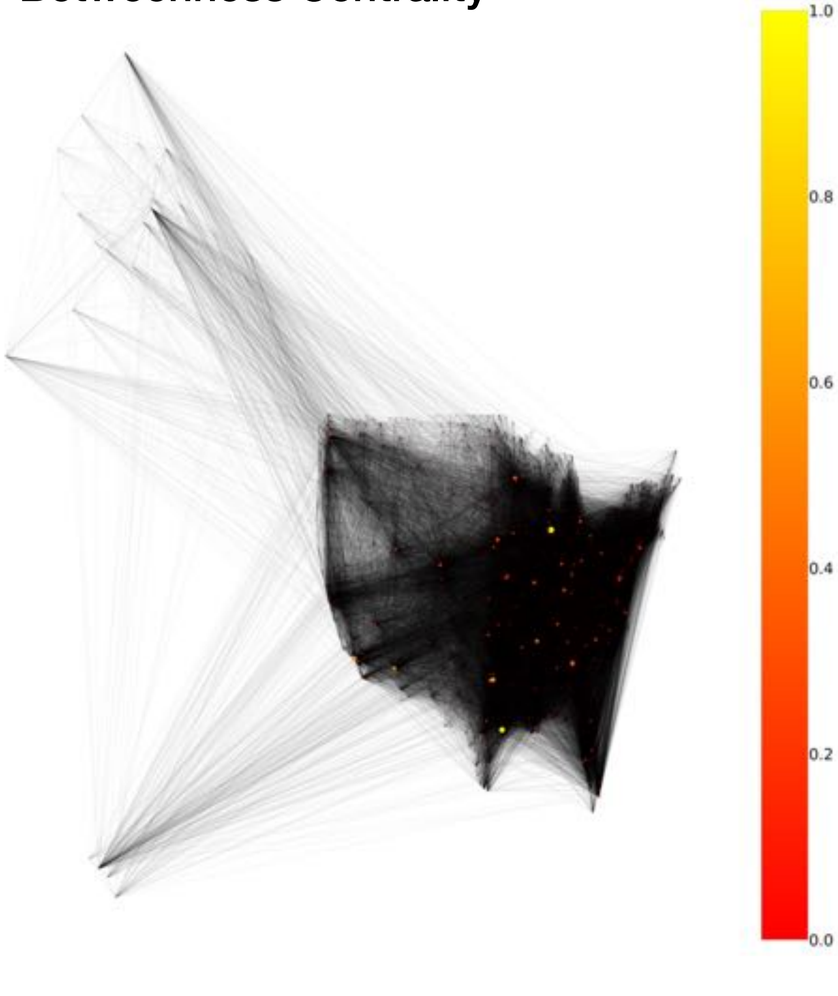
Closeness



Degree Centrality



Betweenness Centrality



Summary:
Community Detection

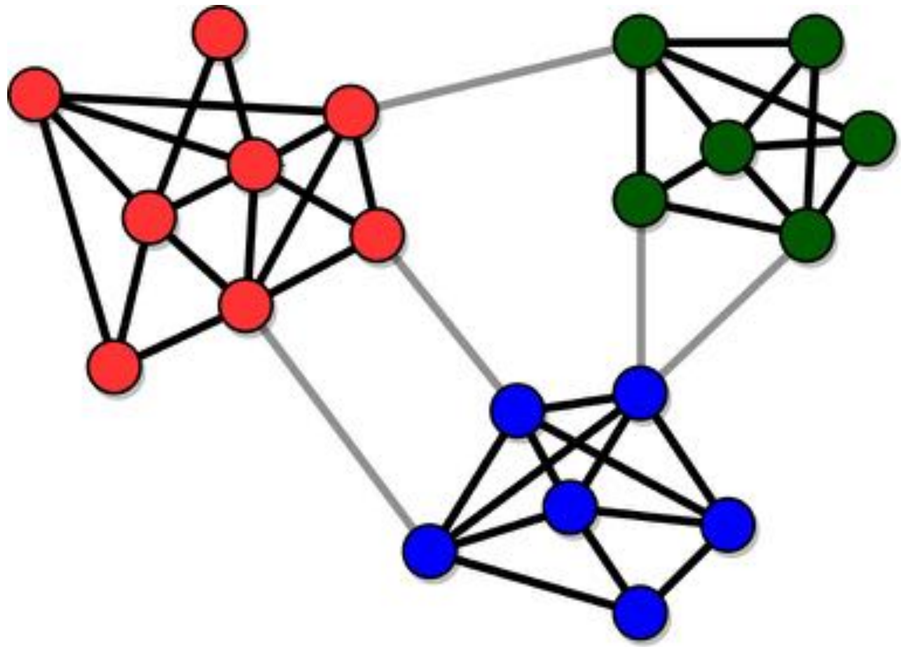
For a working exercise see: CommunityDetection.ipynb from Lecture 5

Outline

- Basic definitions
- Related problems
- Community detection
- Method evaluation

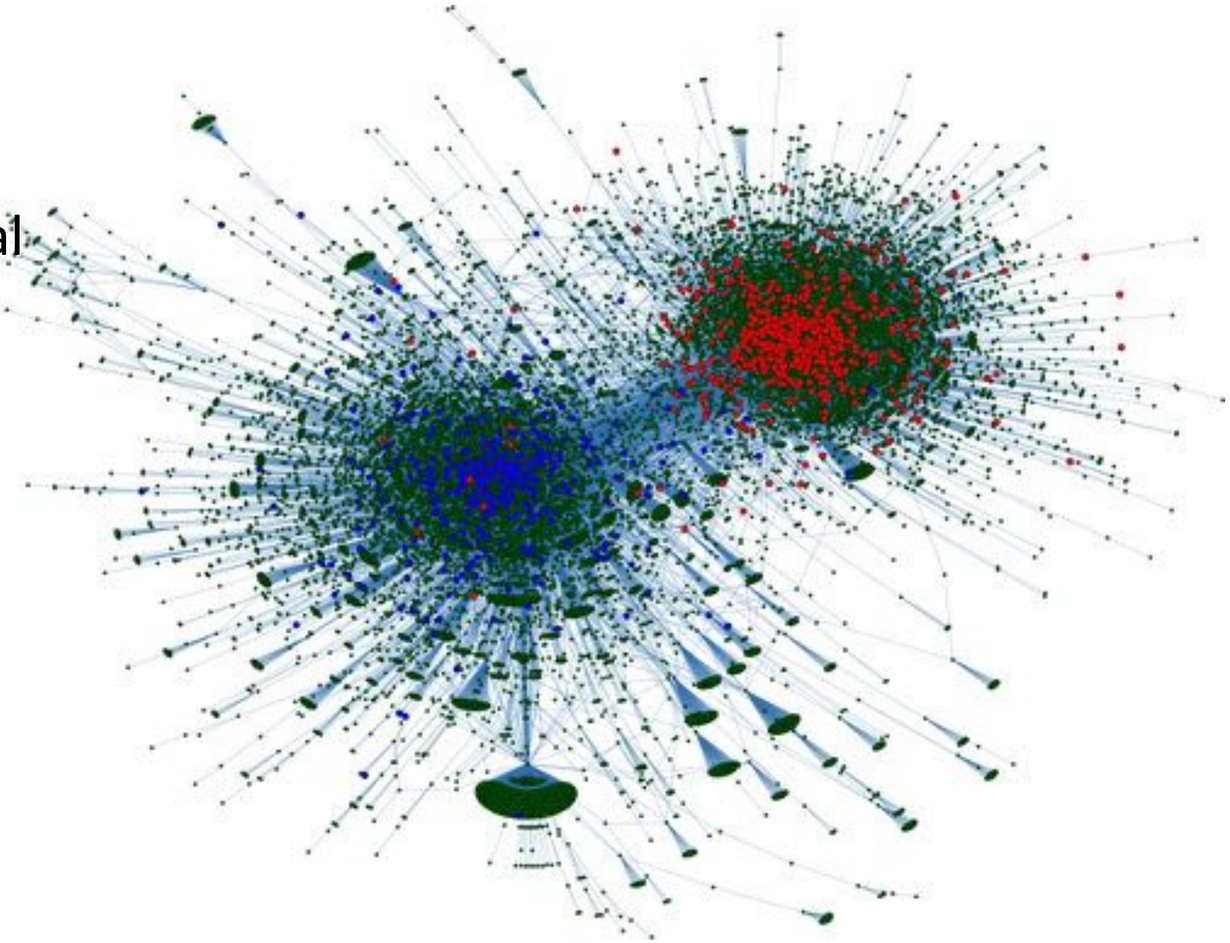
Community structure

Communities (or clusters): sets of tightly connected nodes



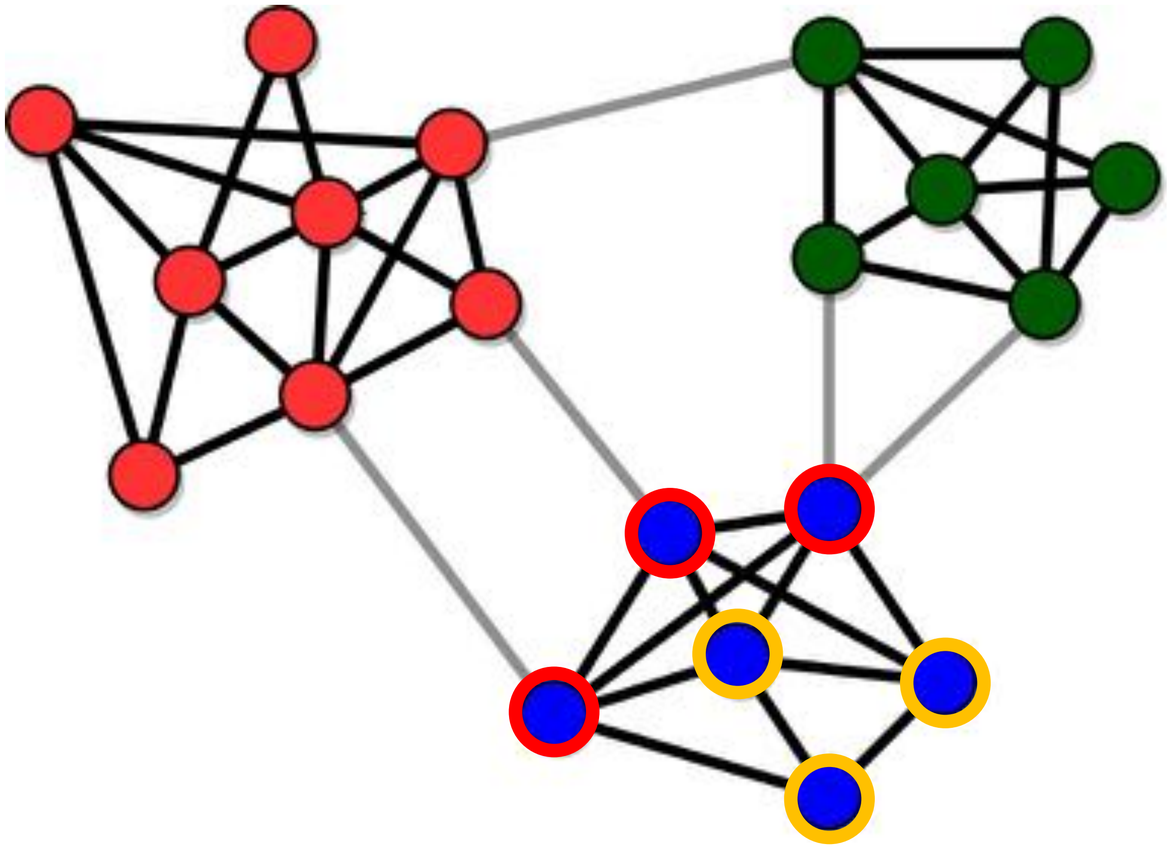
Community structure

- **Example:** Twitter users with strong political preferences tend to follow those aligned with them and not to follow users with different political orientation
- **Other examples:** social circles in social networks, functional modules in protein interaction networks, groups of pages about the same topic on the Web, etc.



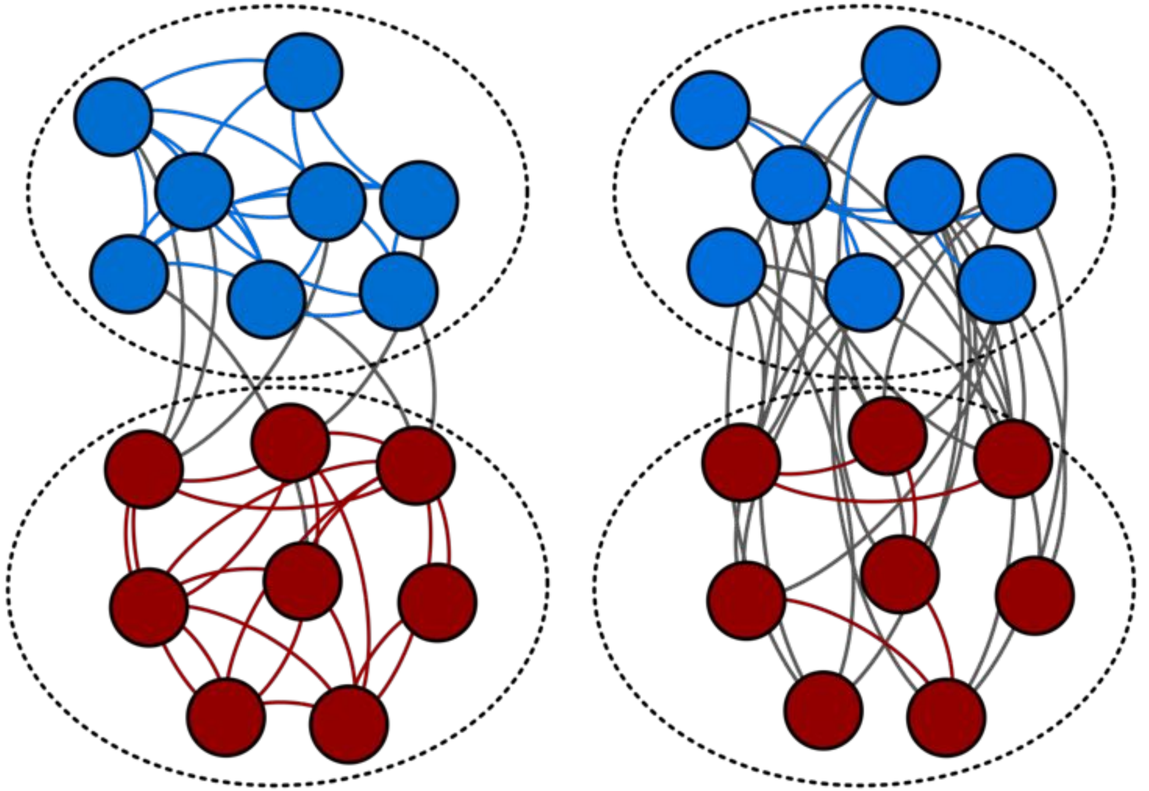
Why study communities?

- Uncover the organization of the network
- Identify features of the nodes
- Classify the nodes based on their position in the clusters
- Find missing links



Modularity

- Let's explain the origin of $k^2_C / 4L$
- Random links are formed by matching pairs of **stubs** (half-links) chosen at random
- The total number of stubs attached to C is k_C
- The probability to select one of those stubs at random is $k_C/2L$ because $2L$ is the total number of stubs of the network (each link yields two stubs)



Original network **Randomized network**

Modularity

$$Q = \frac{1}{L} \sum_{C} \left(L_C - \frac{k_C^2}{4L} \right)$$

- L = number of links in the network
- L_C = number of internal links in community C
- k_C = degree of community C
- $k^2_C/4L$ = expected number of internal links in community C

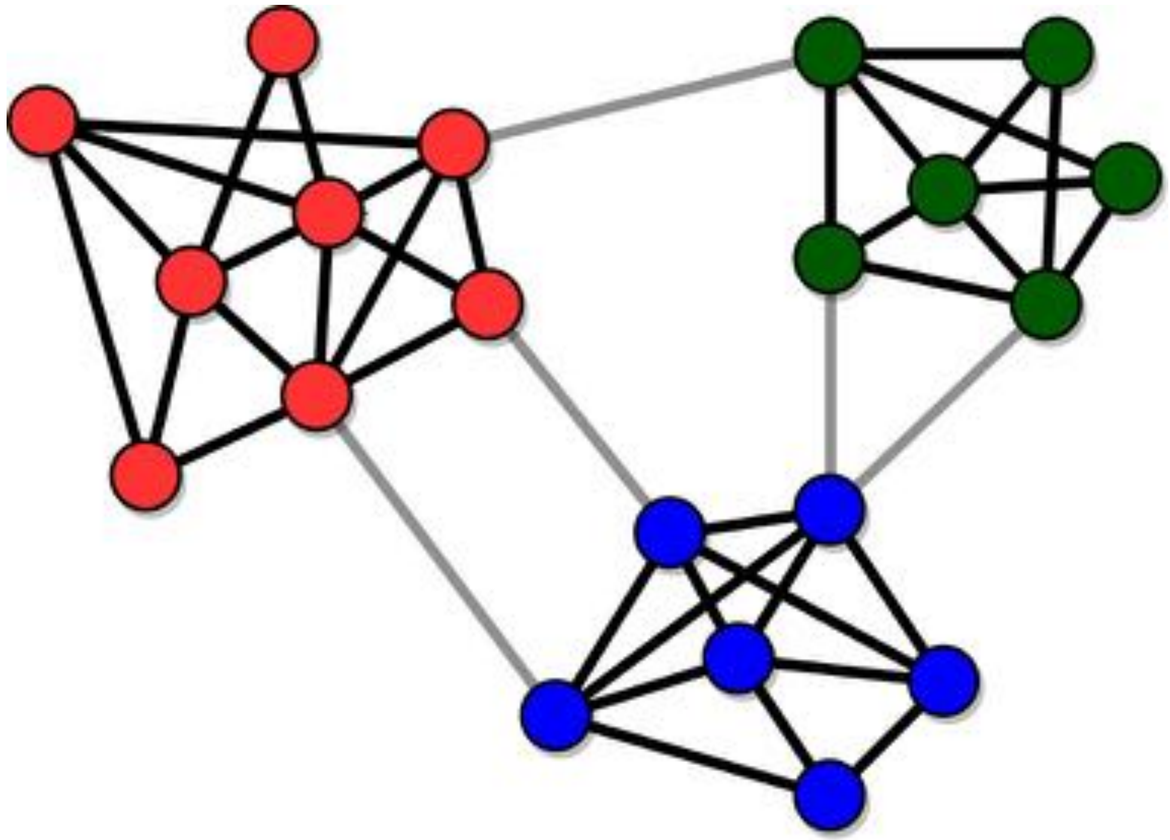
Modularity: features

- $Q < 1$ for every partition of any network
- $Q = 0$ for the partition in which the whole graph is one community
- Q can be negative (e.g., partition in N groups of one node each)
- For most networks, Q has a non-trivial maximum between 0 and 1

```
# returns the modularity of the input partition
modularity = nx.community.quality.modularity(G,partition)
```

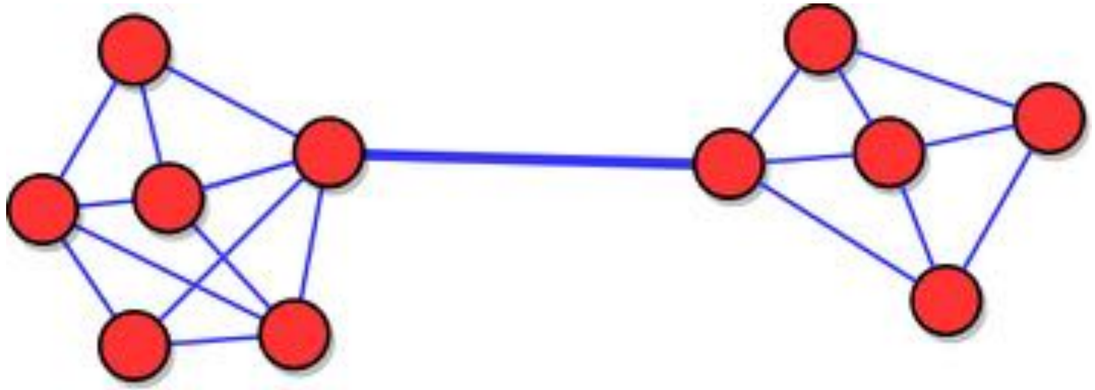
Community detection

- Many different methods
- We discuss these techniques:
  - **Bridge removal**
  - **Modularity optimization**
  - **K-clique**



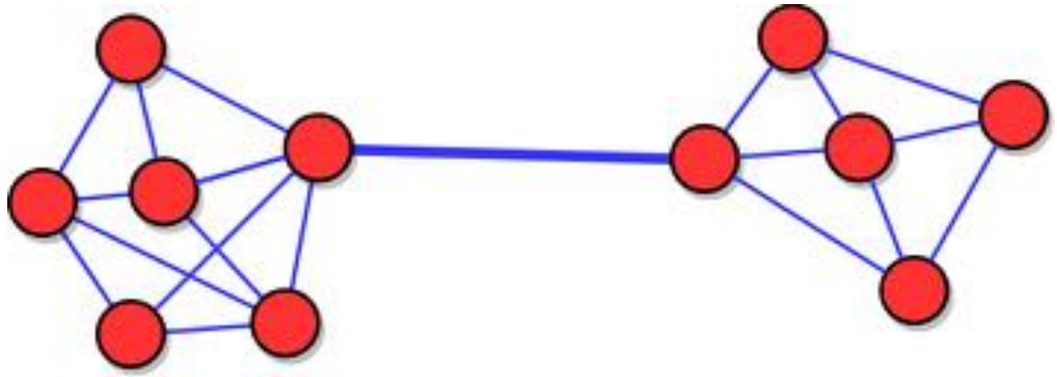
Bridge removal

- **Divisive method:** one removes the links that connect the clusters to each other (**bridges**), until they are isolated
- The problem would then be solved by finding the connected components of the resulting disconnected graph (trivial!)
- Bridges are identified via measures that take larger (smaller) values on those links than on internal links



Girvan-Newman algorithm

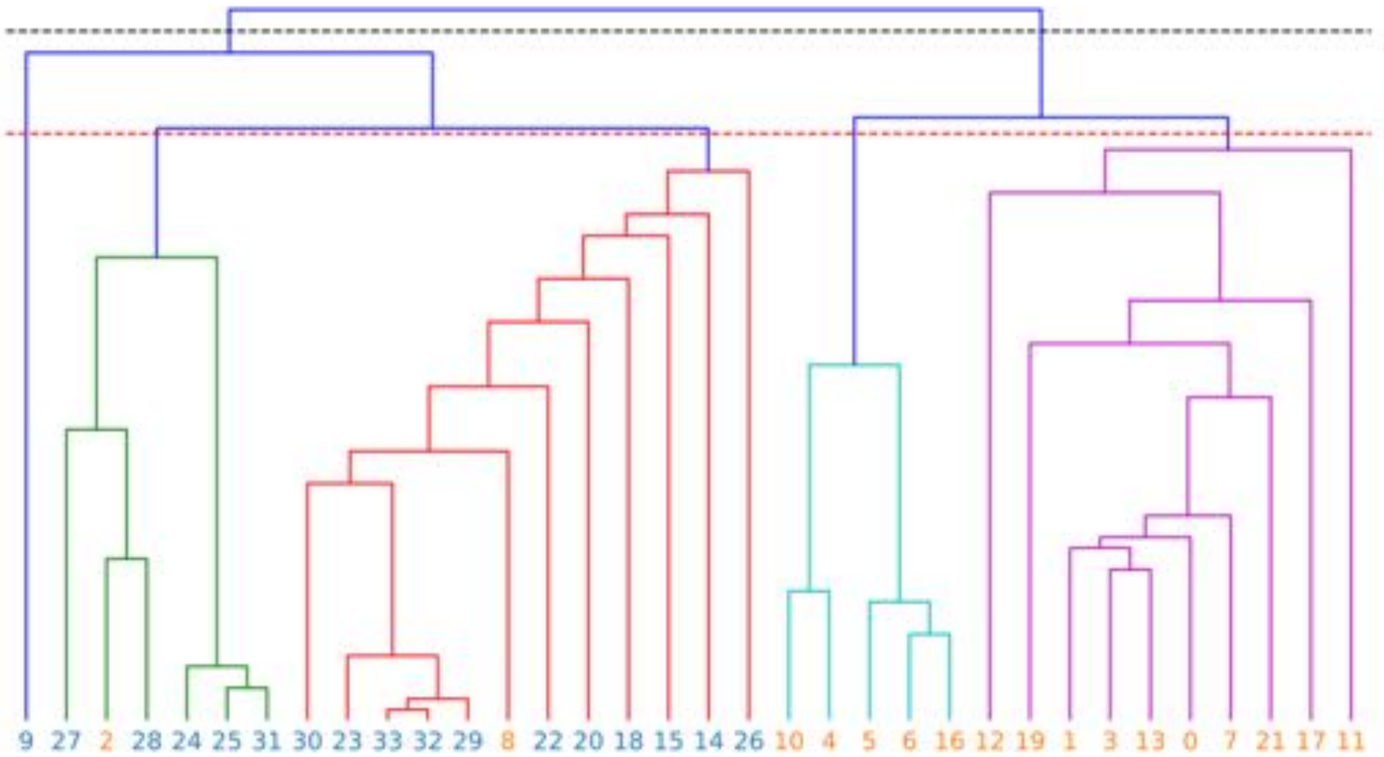
- Measure for bridge identification: **link betweenness**
- Bridges are expected to have large betweenness values because shortest paths between nodes in different communities run through bridges
- Instead, internal links are expected to have comparatively lower betweenness values, because there are many alternative routes going from one node of the community to another, due to the high density of links inside the cluster(s)



Girvan-Newman algorithm

- The Girvan-Newman algorithm a **divisive hierarchical clusterin** method, as it delivers hierarchical partitions by breaking clusters until only **singletons** (isolated nodes) remain

- Output: **dendrogram!**



Girvan-Newman algorithm: limits

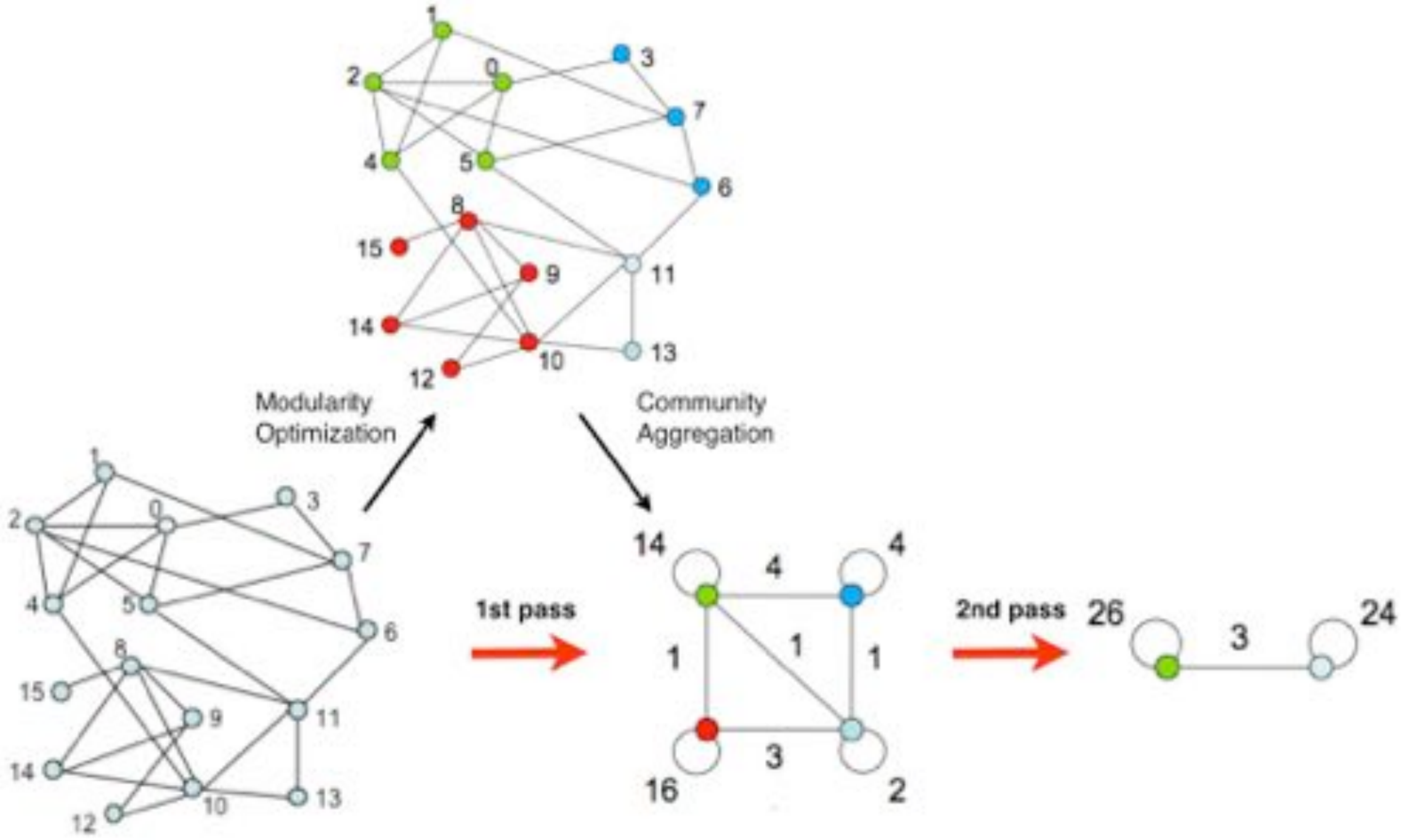
- It is quite slow, it is not practical for large networks with, say, more than 10,000 nodes. The bottleneck is the recalculation of the link betweenness

- Faster variants have been proposed, for instance computing approximations of the link betweenness scores by using only a sample of randomly selected pairs of nodes, or adopting alternative measures to identify bridges, which are quicker to compute

- The method delivers a full hierarchy of *N* partitions: which ones are meaningful and how can they be selected?

```
# returns a list of hierarchical partitions
partitions = nx.community.girvan_newman(G)
```

Louvain algorithm



Louvain algorithm

There currently is no function in NetworkX for the Louvain algorithm. However, it is possible to implement the algorithm using the `community` module

```
# download community module at
# github.com/taynaud/python-louvain
import community

# returns the partition with largest modularity
partition_dict = community.best_partition(G)
```

Louvain algorithm

Limits:

- It is a greedy method, in that it tries to find the best modularity partition at each level of agglomeration. Therefore it yields solutions with sub-optimal modularity
- The final partition depends on the order in which nodes are visited

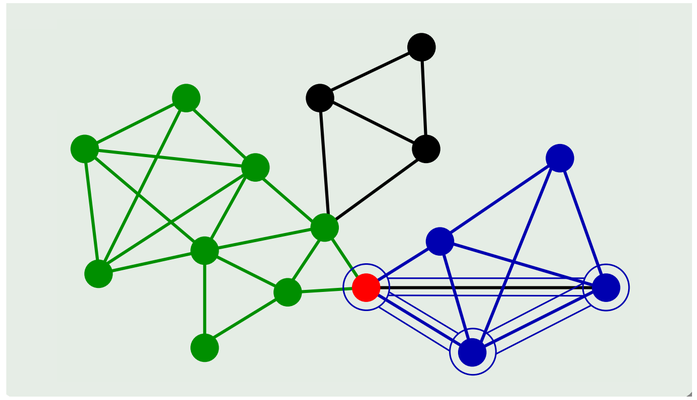
Big plus:

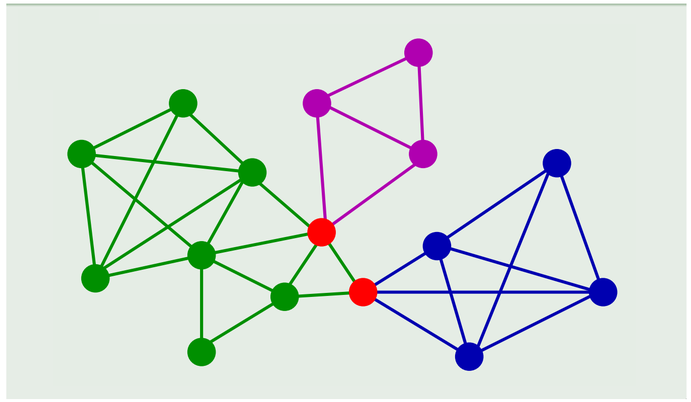
- The algorithm is very fast because, after the first iteration, the successive transformations shrink the network very quickly and typically only a handful of partitions are generated. It can detect communities in networks with millions of nodes and links

K-clique community Detection

- Two *k*-cliques (complete subgraphs of *k* nodes) are considered adjacent if they share *k-1* nodes

- A *k*-clique community is the largest connected subgraph obtained by the union of all adjacent *k*–cliques





In this example: k=3

K-clique Communities

There currently is no function in NetworkX for the Louvain algorithm. However, it is possible to implement the algorithm using the `community` module

```
from networkx.algorithms.community import
k_clique_communities

#select the k value
k=3;

partition = list(k_clique_communities(G,k))
```

Modularity optimization: limits

- The maximum modularity tends to be larger on larger networks, so the measure cannot be used to compare the quality of partitions across networks

- The maximum modularity does not necessarily correspond to the best partition. Communities smaller than a certain size may not be detected (**resolution limit**)

Method evaluation

Problem: How can we tell how good a clustering algorithm is?

Normalized mutual information

- Two partitions: *X* and *Y*
- Probability that a randomly chosen node belongs to cluster *x* (with size N_x) of partition *X*: $P(x) = N_x / N$
- Probability that a randomly chosen node belongs to cluster *x* of partition *X* and to cluster *y* of partition *Y*: $P(x,y) = N_{xy} / N$, where N_{xy} = number of nodes shared by *x* and *y*
- **Shannon entropy of *X*:**

$$H(X) = -\sum_x P(x) \log P(x)$$

- **Conditional entropy of *X* given *Y*:**

$$H(X|Y) = \sum_{x,y} P(x,y) \log[P(y)/P(x,y)]$$

Normalized mutual information

$$\mathrm{NMI}(X,Y) = \frac{2H(X) - 2H(X|Y)}{H(X) + H(Y)}$$

- *NMI* = 1 if and only if the partitions are identical
- *NMI* has an expected value of zero if the partitions are independent, as for example when two random partitions are compared
- **Problem:** Detected partitions with more clusters may yield larger values of the *NMI* even though they are not necessarily closer to the benchmark partition

Centrality measures

- **Centrality**: measure of importance of a node
- **Measures**:
  1. Degree
  2. Closeness
  3. Betweenness

Degree

- **Degree of a node:** number of neighbors of the node

$$k_i = \text{number of neighbors of node } i$$

- High-degree nodes are called **hubs**

- **Average degree of the network:**

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

```
G.degree(2) # returns the degree of node 2
G.degree()  # dict with the degree of all nodes of G
```

Closeness

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

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

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

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

Betweenness

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

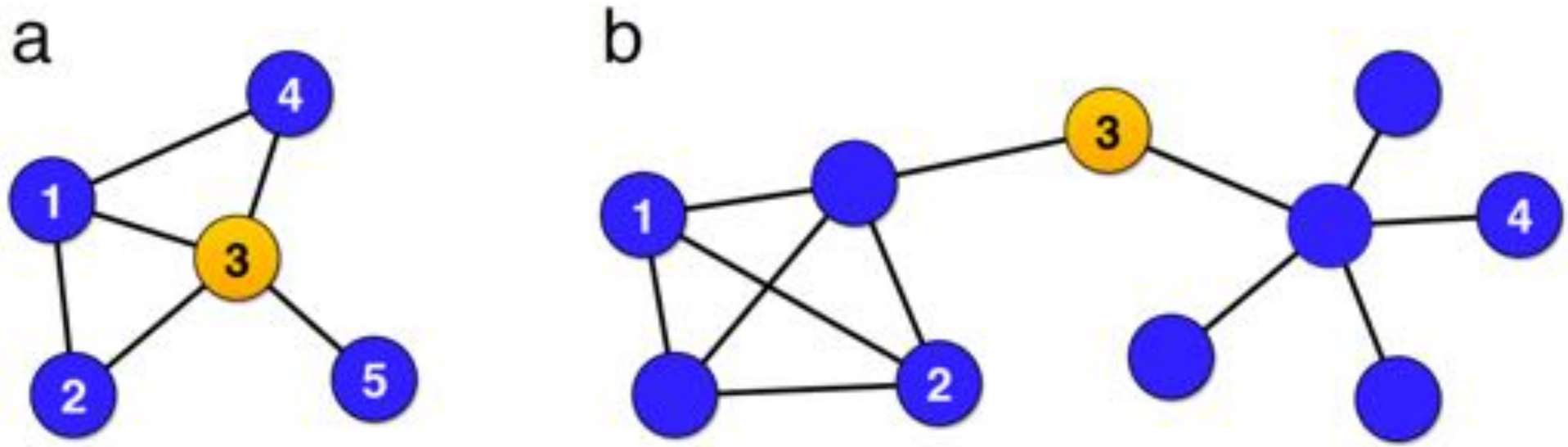
$$b_i = \sum_{h \neq j \neq i} \frac{\sigma_{hj}(i)}{\sigma_{hj}}$$

$\sigma_{hj} =$ number of shortest paths from h to j

$\sigma_{hj}(i) =$ number of shortest paths from h to j running through i

Betweenness

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



Betweenness

- Betweenness can be easily extended to links
- **Link betweenness**: fraction of shortest paths among all possible node pairs that pass through the link

```
nx.betweenness_centrality(G)       # dict nodes ->
                                   # betweenness centrality
nx.edge_betweenness_centrality(G) # dict links ->
                                   # betweenness centrality
```

Centrality distributions

- On small networks it makes sense to ask which nodes or links are most important
- On large networks **it does not**
- **Solution:** statistical approach
- Instead of focusing on individual nodes and links, we consider **classes** of nodes and links with similar properties