STAT 5630, Fall 2019

Intro to Network Analysis and Graphical Model

Xiwei Tang, Ph.D. <xt4yj@virginia.edu>

University of Virginia
October 31, 2019

Outline

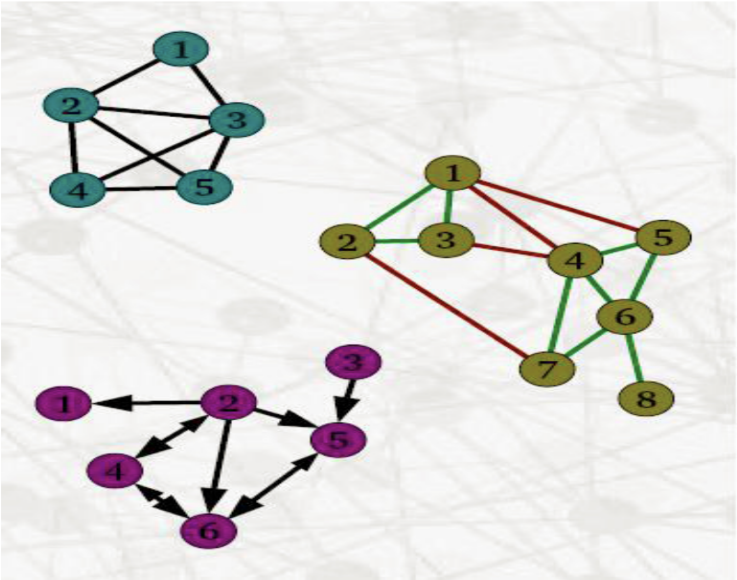
- Basic Concepts
- Graphical Models
- Computation

Networks are Relational Data

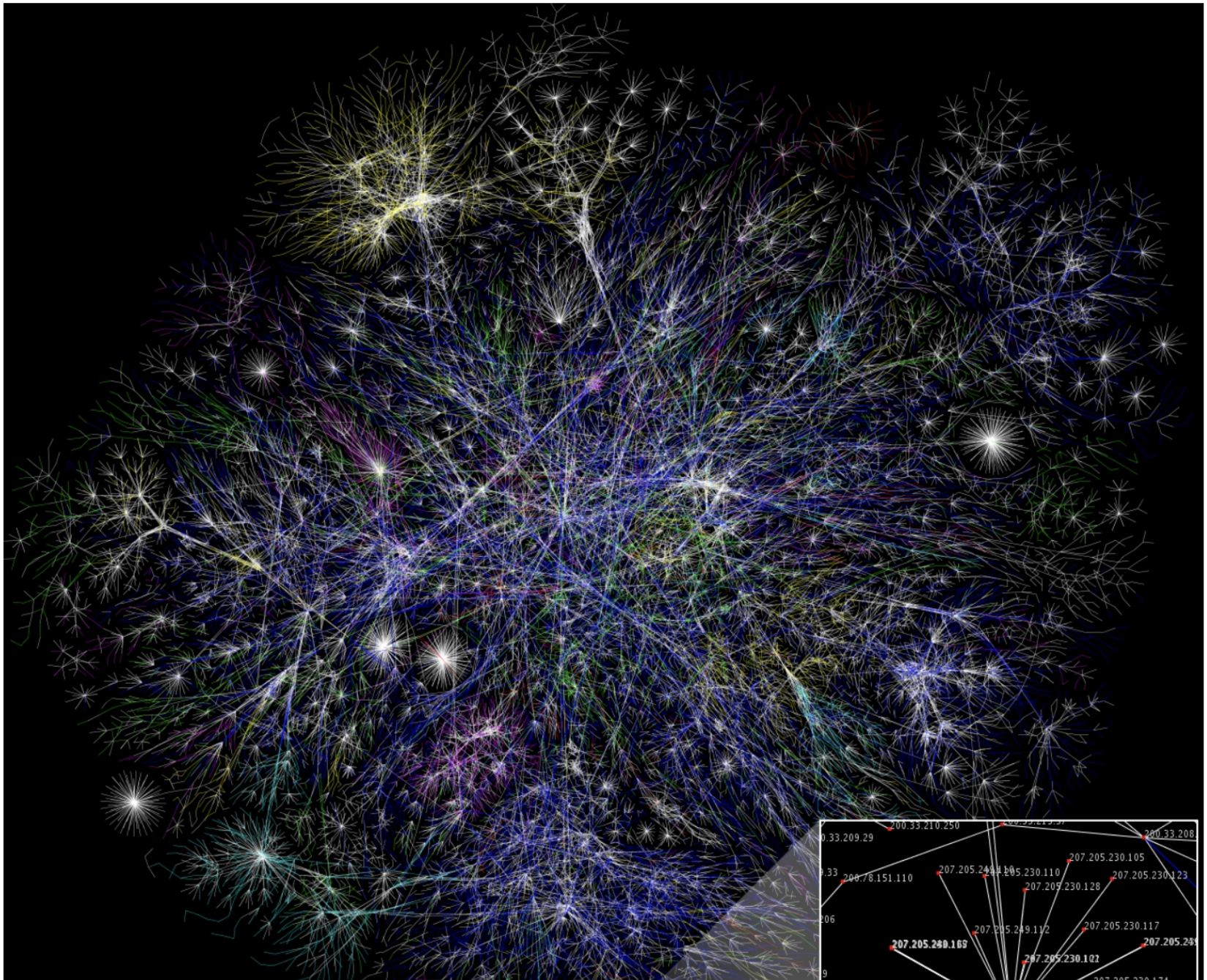
- Relationship: an irreducible property of two or more entities
- Contrast to properties of entities alone (”attributes”)
    - Entities: people, animals, groups, locations, organizations, regions, etc.
    - Relationships: communication, acquaintanceship, sexual contact, trade, migration rate, alliance/conflict, etc.

Basic Concepts

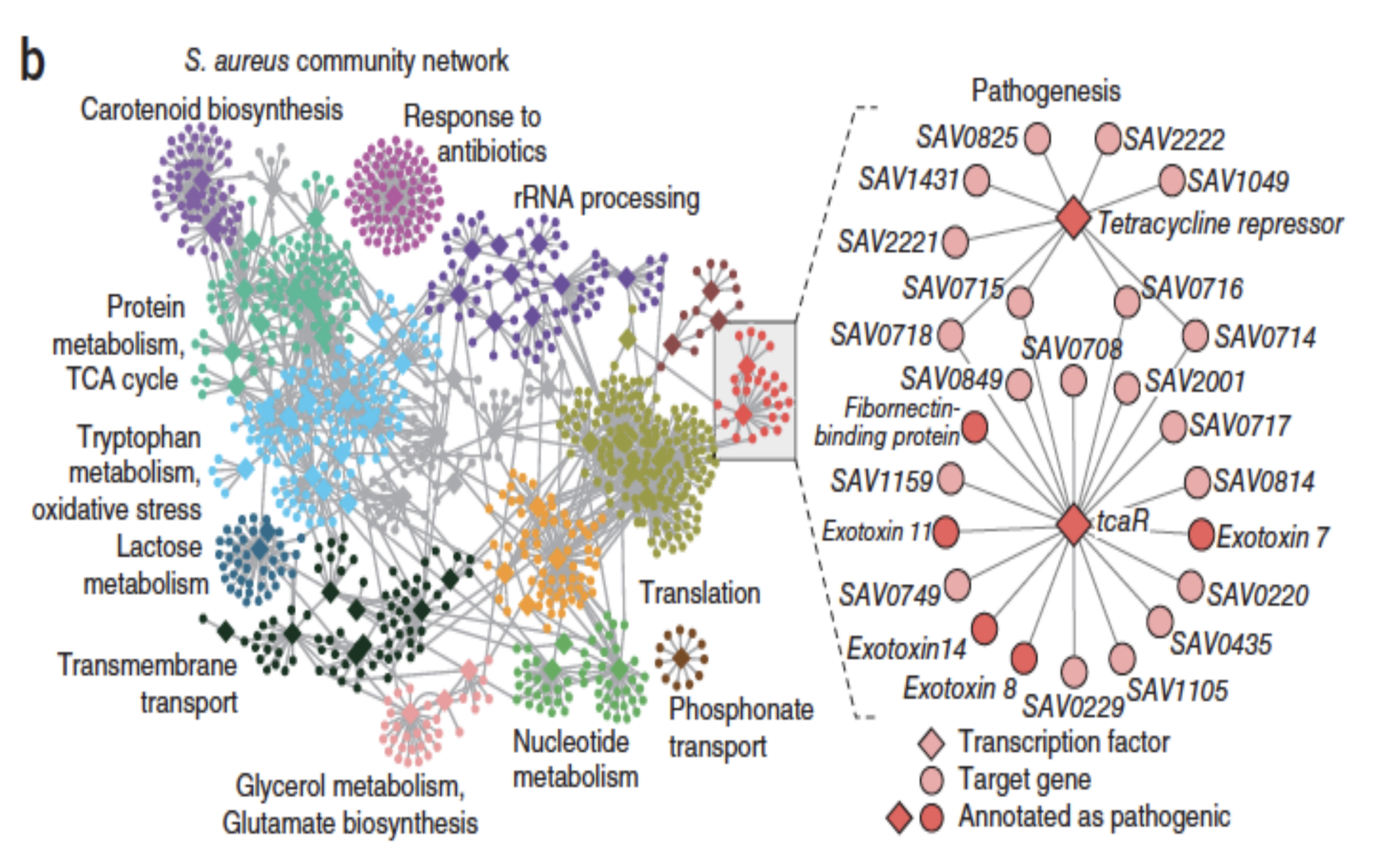
- Network data: A collection of entities and a set of measured relations between them
  - Entities: actors, nodes, vertices
  - Relations: ties, links, edges
- Relations can be
  - Directed or undirected
  - Signed or valued



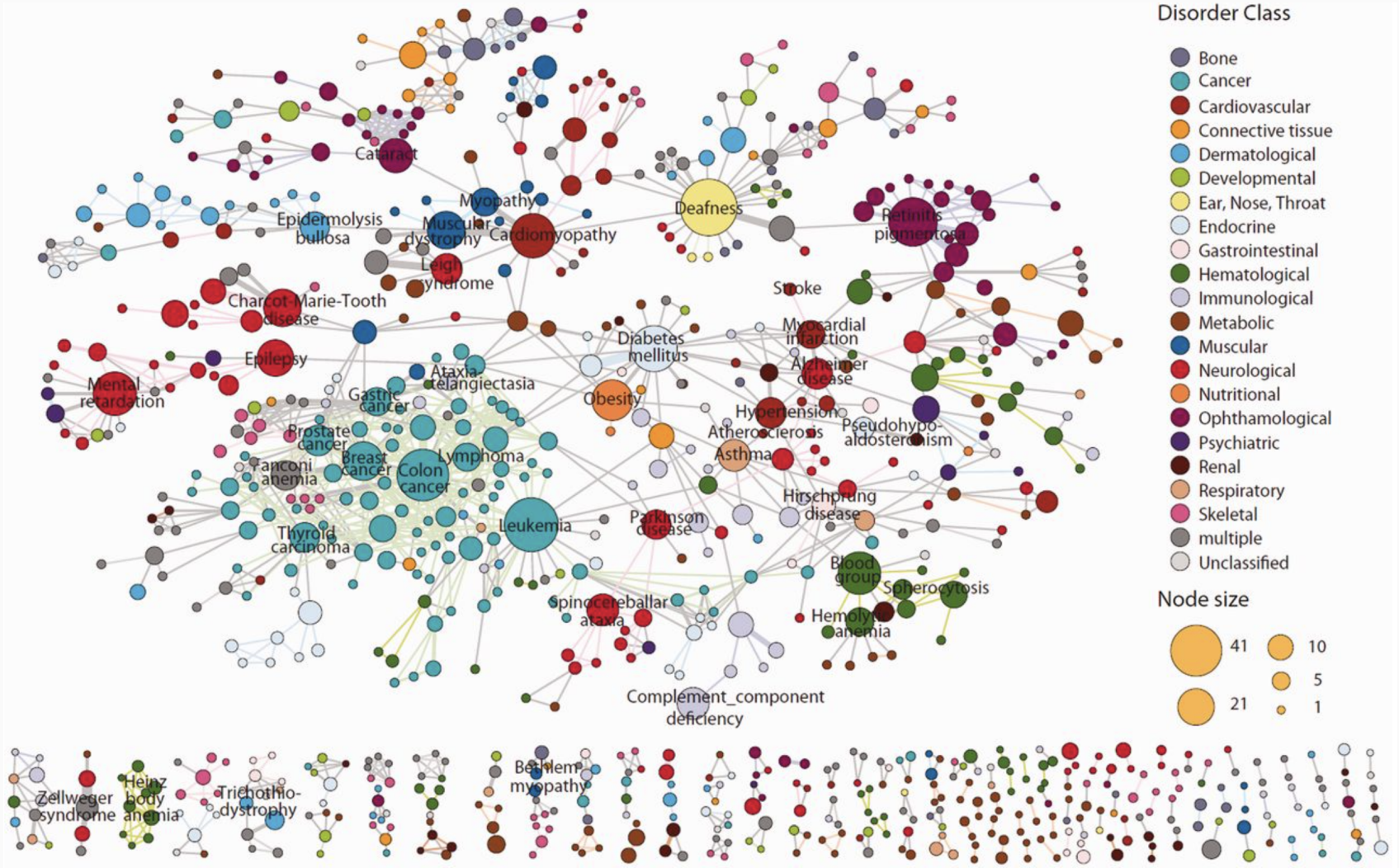
Map of the Internet



S. aureus Network

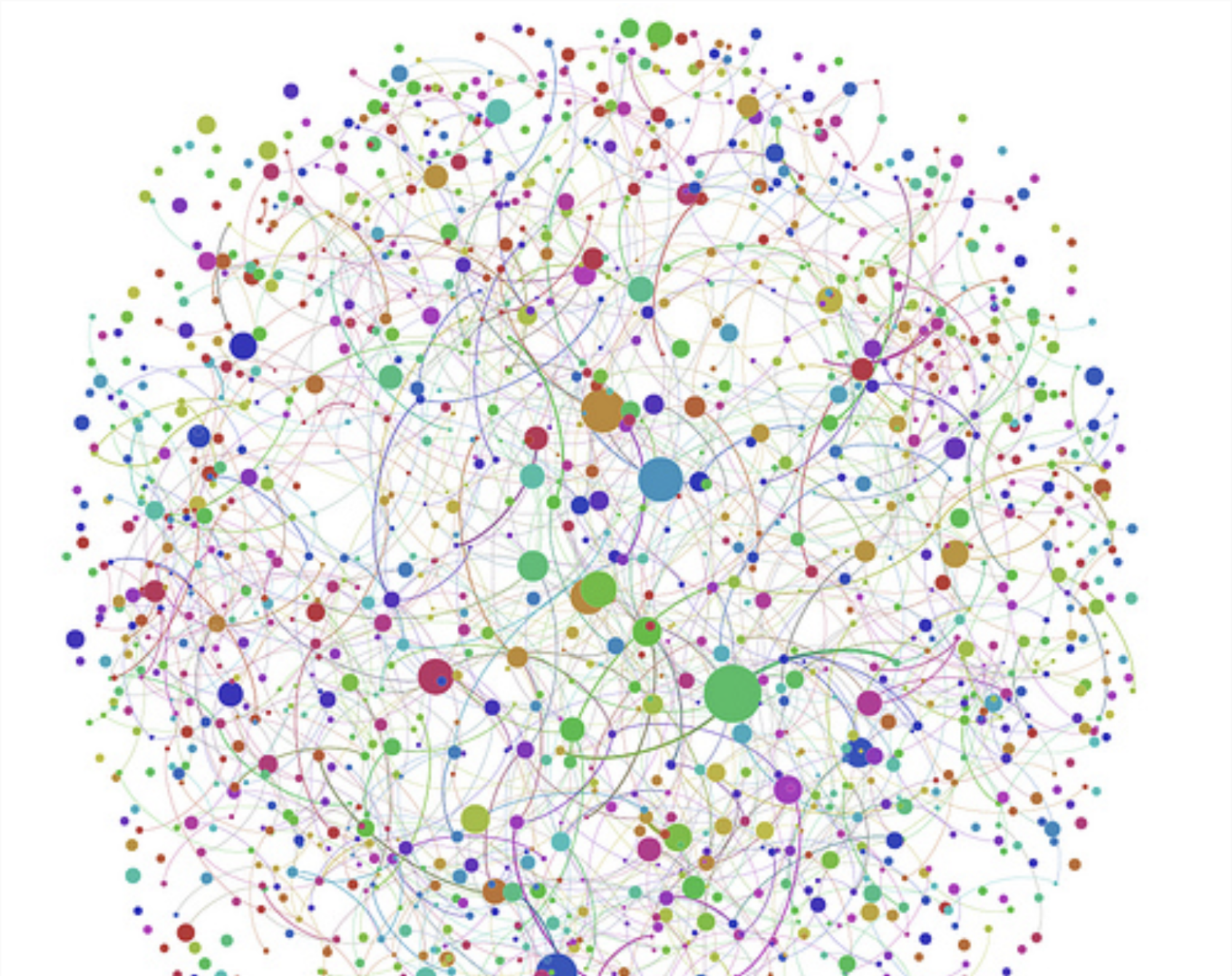


Human disease network

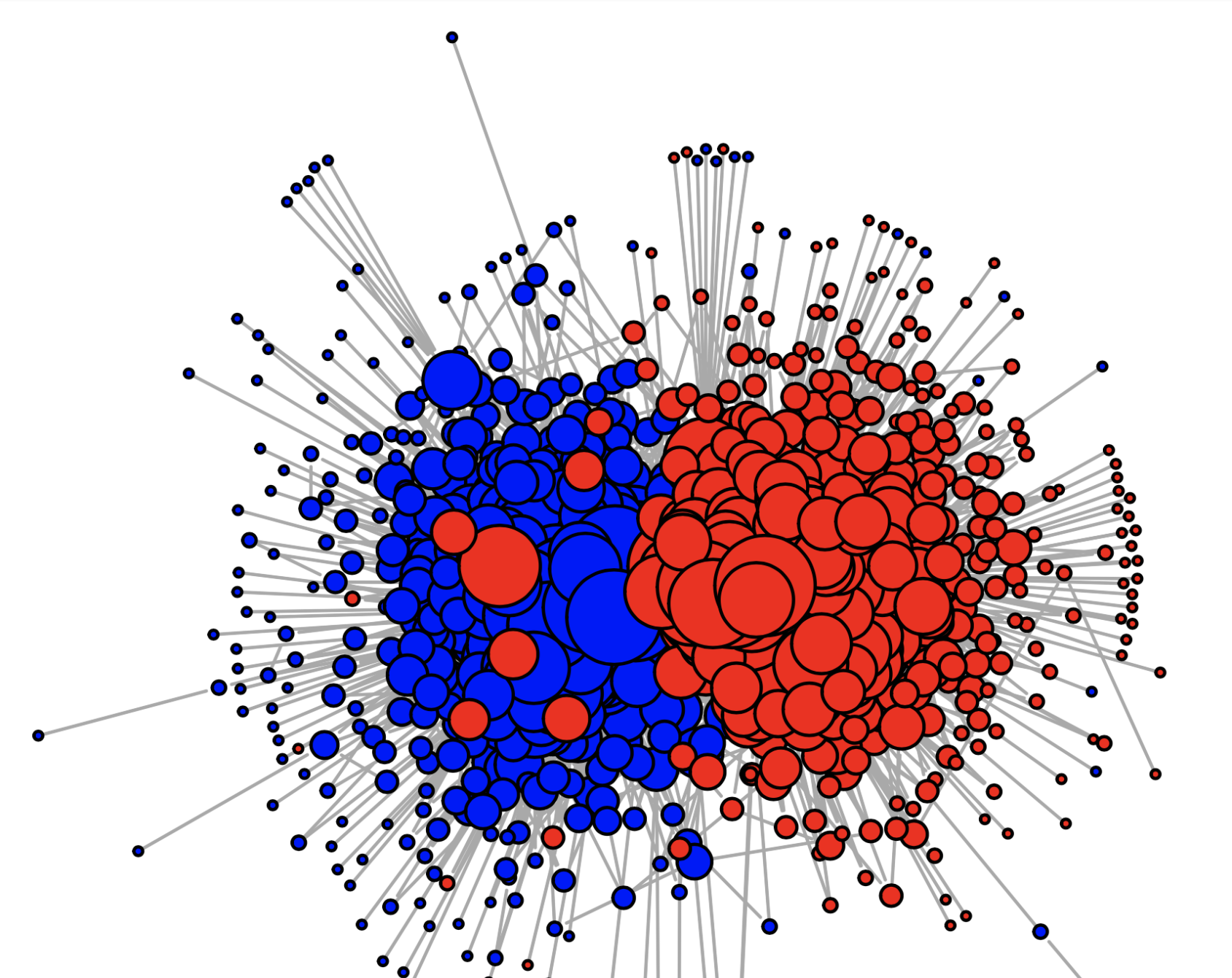


Facebook

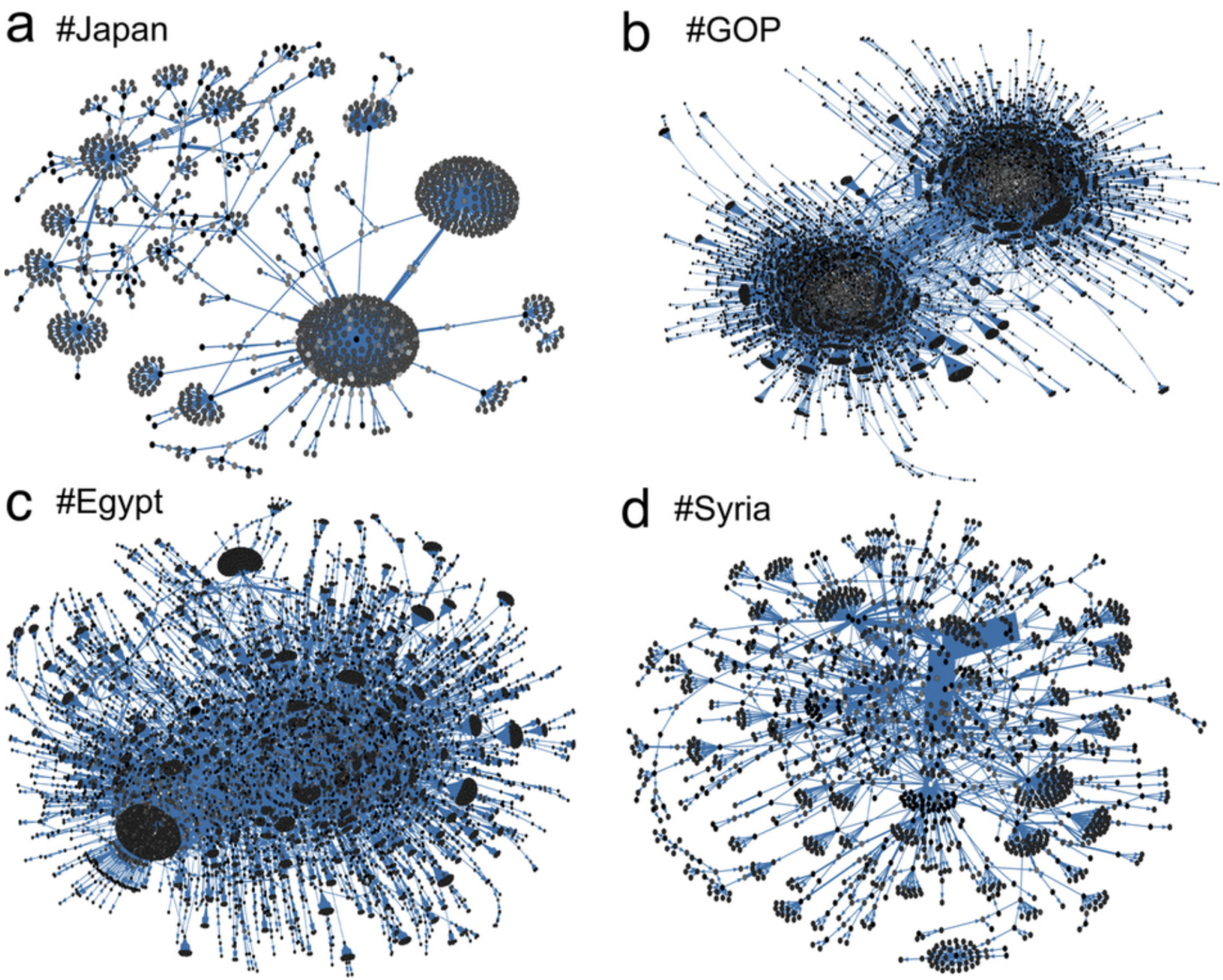
Coauthor network



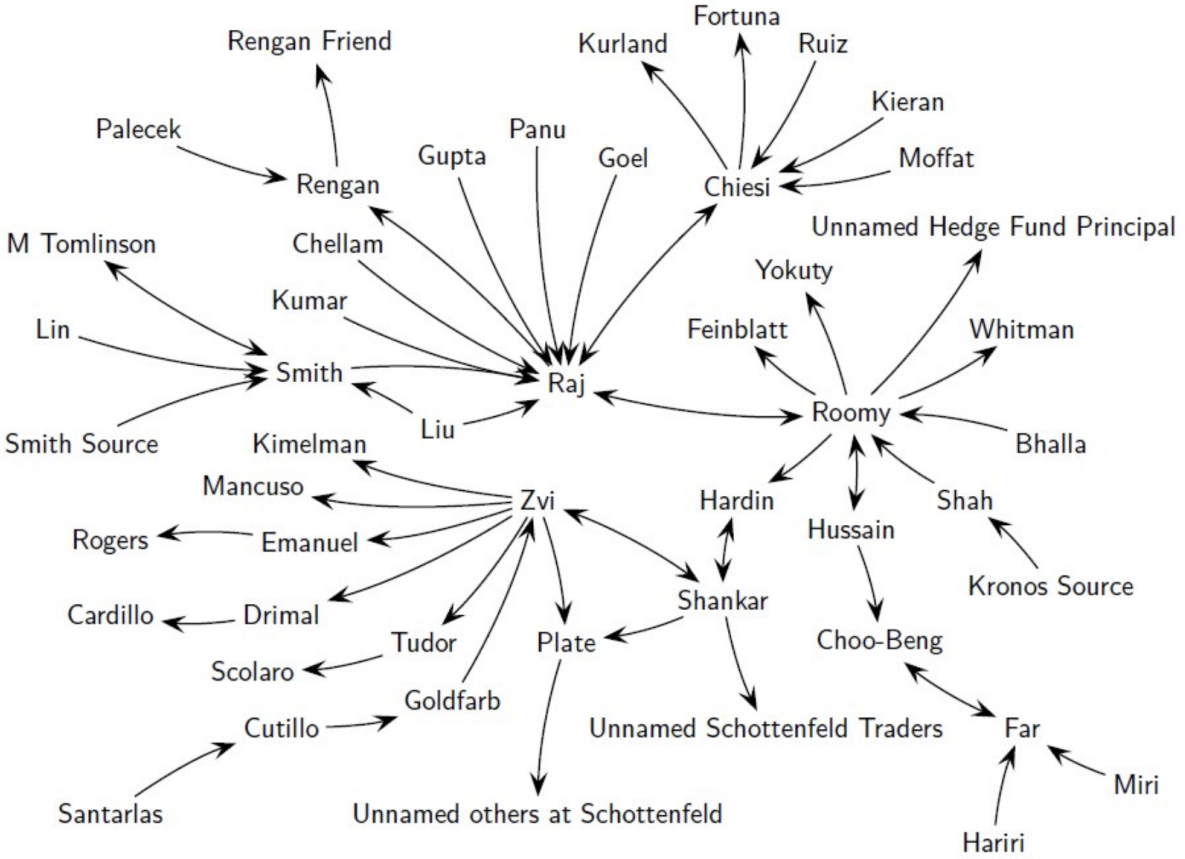
Polotical blogs



Twitter



Insider trading network



INFORMATION NETWORKS

Figure 1
Raj Rajaratnam Network
This figure represents the illegal insider trading network centered on Raj Rajaratnam. Arrows represent the direction of information flow. Data are from SEC and DOJ case documents, plus additional source documents to identify individuals not named in the SEC and DOJ documents.

Nodes and Edges

- A graph consists of
  - A set of nodes $\mathcal{N} = \{1, \ldots, n\}$
  - A set of edges or links between nodes $\mathcal{E} = \{e_1, \ldots, e_m\}$
- The graph is denoted $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. Each edge $e \in \mathcal{E}$ is expressed in terms of the pair of nodes the line connects.
  - Undirected graph: The edges have no direction, and the edge $\{i, j\}$ is the same as the edge $\{j, i\}$, i.e. each edge is an unordered pair of nodes.
  - Directed graph: The edges have direction, and the edge (i, j) is not the same as the edge (j, i), i.e. each edge is an ordered pair of nodes.

Matrix Representations

- The data can be represented by an $n \times n$ matrix $A = \{A_{ij} : i, j \in \mathcal{N}\}$, where

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in \mathcal{E} \\ 0 & \text{if } (i,j) \notin \mathcal{E} \end{cases}$$

- This matrix is called the adjacency matrix of the graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$.
  - The adjacency matrix of every graph is a square, binary matrix.
  - Every adjacent matrix corresponds to a graph.

Adjacency Matrices

- For an undirected binary relation, $\{i,j\} = \{j,i\}$ and so $A_{ij} = A_{ji}$ by design
  - The representing graph is an undirected graph
  - The representing adjacency matrix is symmetric
- For a directed binary relation, $(i,j) \neq (j,i)$ and it is possible that $A_{ij} \neq A_{ji}$.
  - The representing graph is a directed graph
  - The representing adjacency matrix is possibly asymmetric

Summary Statistics

- Even in the simplest case of an undirected binary relational data, an adjacency matrix can be a complicated and opaque object.
  - Statistic: A statistic $t(A)$ is any function of the data
  - Descriptive data analysis: A representation of the main features of a dataset via a set of statistics $t_1(A), \ldots, t_m(A)$
- Many important statistics can be computed from the adjacency matrix using basic matrix algebra.

Density

- The most basic statistic of a relational dataset is the *mean* or *density*.
- Density: the proportion of edges present in a graph, i.e. (the number of edges)/(the maximum possible number of edges)
- The number of edges observed is $|\mathscr{E}|$.
- The number of possible edges is
  - $n(n-1)/2$ in an undirected graph
  - $n(n-1)$ in a directed graph

Density as an Average

- Let A_{ij} be the binary indicator of an edge from i to j.
- The density from the adjacency matrix:

$$\bar{A} = \frac{\sum_{i,j:i \neq j} A_{ij}}{n(n-1)} = \text{average value of } A_{ij} \text{ among non-diagonal entries}$$

Nodal Degree

- Nodes typically vary in their involvement in the network. For binary relations, this heterogeneity can be summarized by the nodal degree.
  - Undirected relation:
    - The degree of a node is the node's number of ties.
  - Directed relation:
    - The outdegree of a node is the node's number of outgoing ties.
    - The indegree of a node is the node's number of incoming ties.

Nodal Degree

- The degrees are easy to calculate from the adjacency matrix A:

$$d_i^o = \sum_{j:j\neq i} A_{ij}$$
$$d_i^i = \sum_{j:j\neq i} A_{ji}$$

- This calculation works for both directed and undirected relations. Specifically, for an undirected relation,

$$d_i^o = d_i^i = d_i$$

Univariate Summaries of Degrees

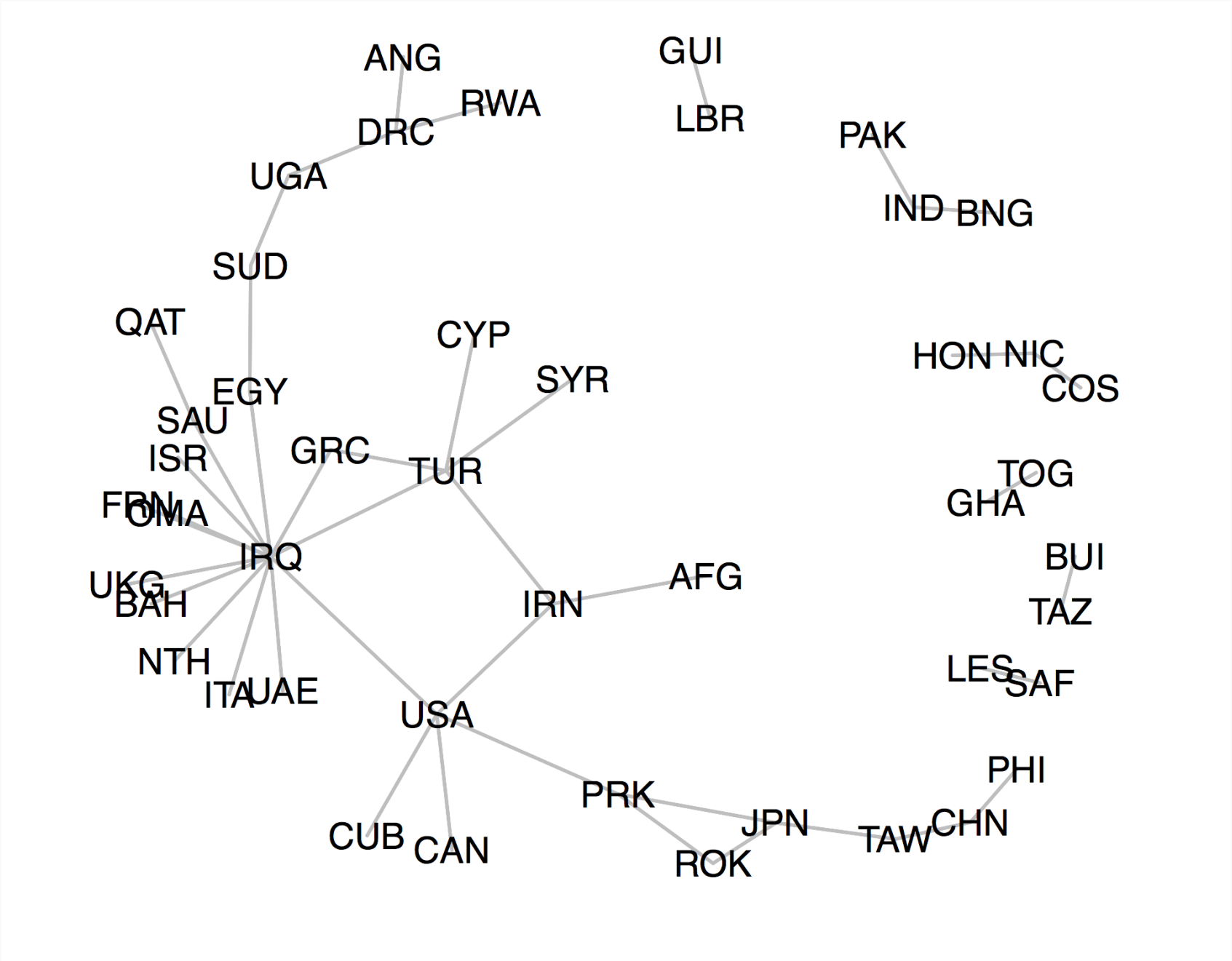
- Let $d = \{d_1, \ldots, d_n\}$ be a set of nodal degrees (either outdegrees, indegrees, or undirected degrees).
- The entries of d are often summarized further with
  - Mean: $\bar{d} = \sum d_i / n = (n-1)\bar{A}$
  - Variance: $s_d^2 = \sum (d_i - \bar{d})^2 / (n-1)$
  - Degree distribution

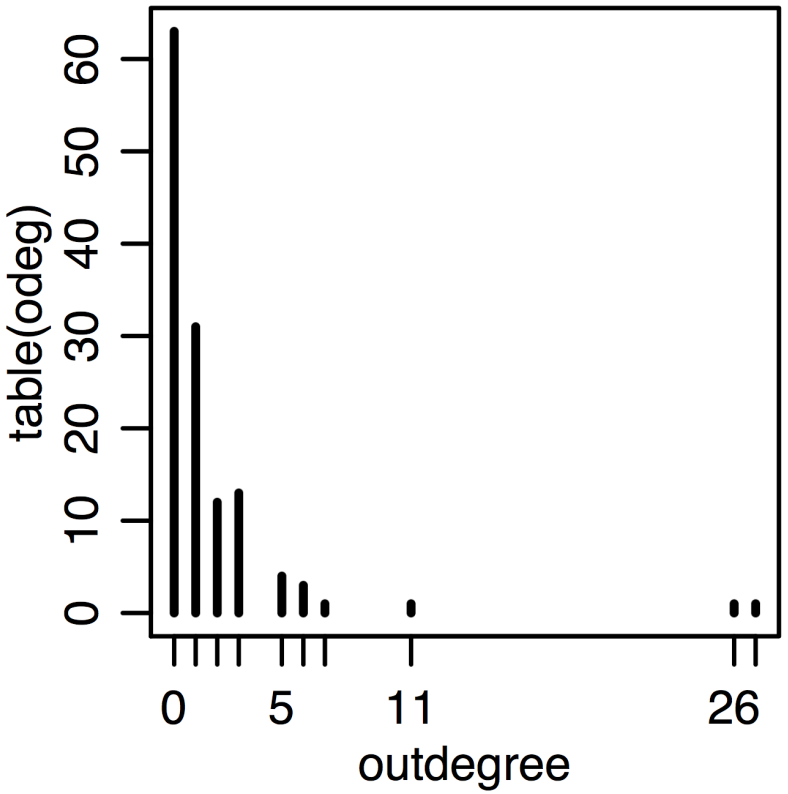
Degree Distribution

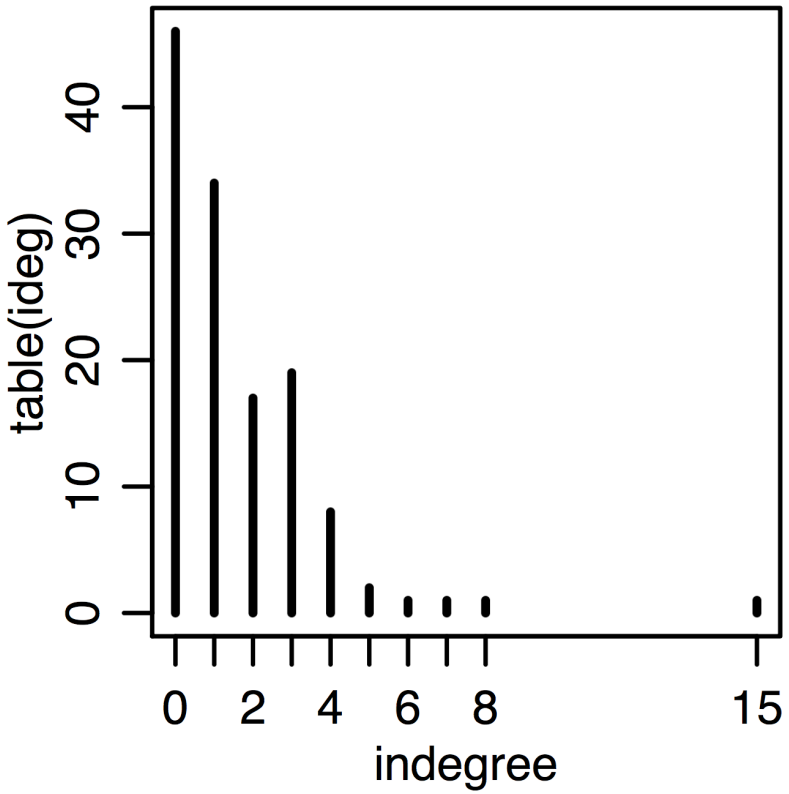
The degree distribution is a set of counts $\{f_0, \ldots, f_{n-1}\}$ where

$$f_k = \#\{d_i = k\} = \text{ number of nodes with degree equal to } k$$

Example: 1990-2000 International Conflict




table(odeg)
0
10
20
30
40
50
60
0
5
11
26
outdegree


table(ideg)
0
10
20
30
40
0
2
4
6
8
15
indegree

Degree Variation

- For the conflict network:
  - Most nodes have small degrees
  - Few nodes have large degrees
- Recall the degree distribution $f = \{f(k), k = 0, \ldots, n-1\}$, where

$$f(k) = f_k = \#\{d_i = k\}$$

- For the conflict network, the degree distribution $f(k)$ is roughly a decreasing function of k.

Power Law Behavior

- Some researchers propose an explicit form for $f(k)$:

$$f(k) = ak^{-b},\ a > 0, b > 0,$$

  a distribution is said to follow a power law.
- A network model with degree distribution following a power law is scale free.

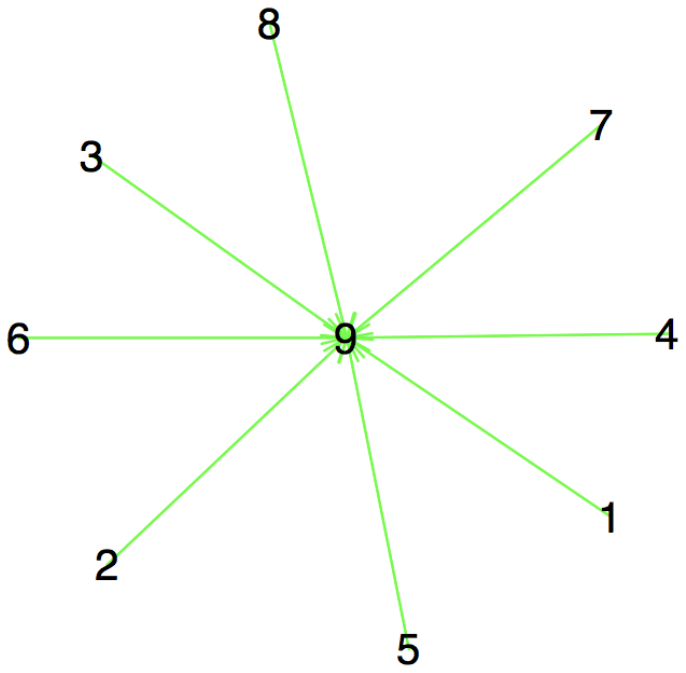
- For such a degree distribution,

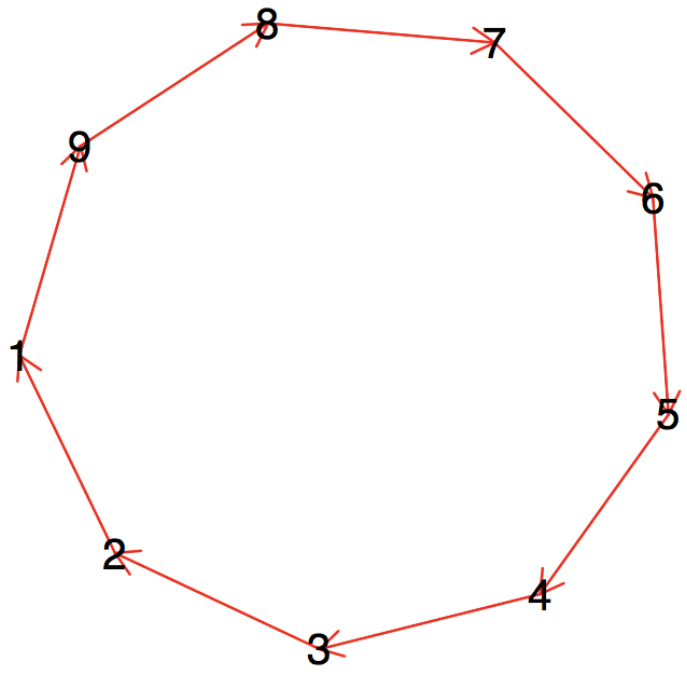
$$\log f(k) = \log a - b \log k,$$

- The log value of $f(k)$ should be linearly decreasing in $\log k$.
- This can be checked empirically by plotting the log degree distribution versus k.

Network Connectivity

- Density (or average degree) is a very coarse description of a graph.
- Compare the n-star graph to the n-circle graph below: The two graphs have roughly the same density, but the structure is very different.



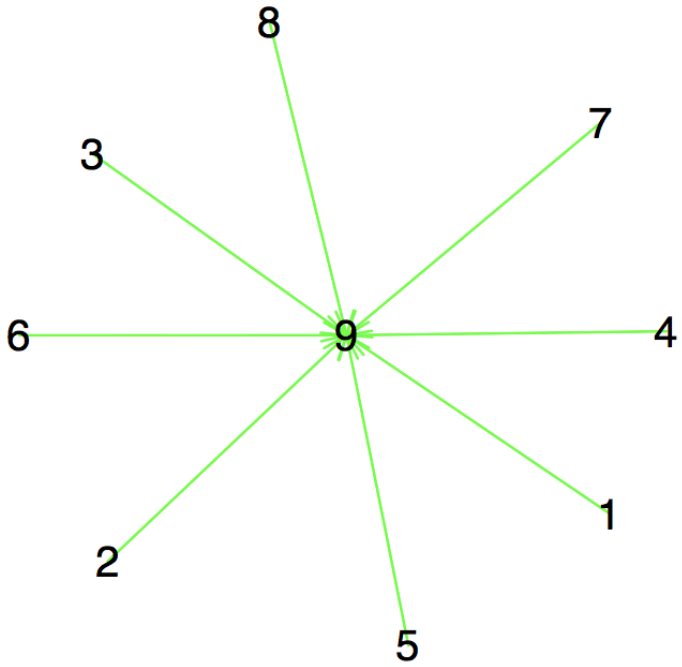


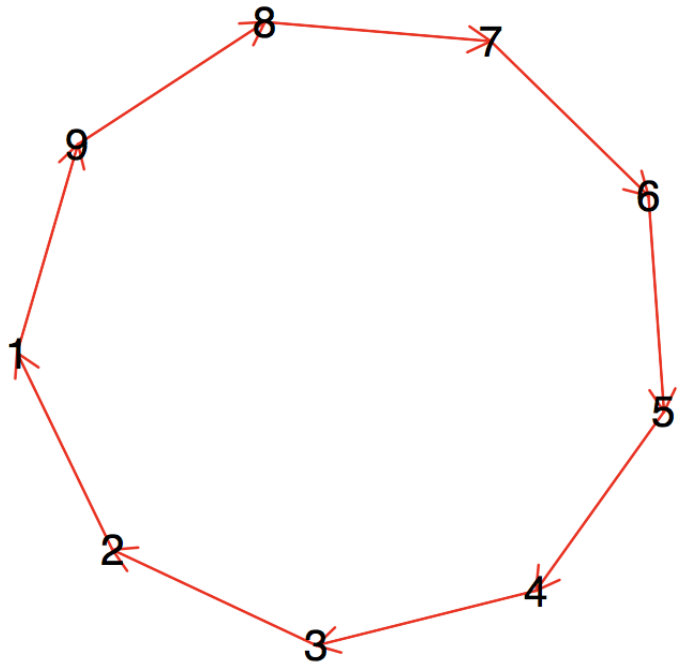
Evaluating Connectivity

- Which graph seems more “connected”?
  - The star graph?
    - Each node is within at most two links of every other node.
    - Transmitting information in this network is easier than in the circle graph.
  - The circle graph?
    - Removal of one node can completely disconnect the star graph.
- Intuitively, a “highly connected” graph is one in which nodes can reach each other via connections, or a “path.”

Node Connectivity

- Node connectivity: The node connectivity of a graph $k(\mathscr{G})$ is the minimum number of nodes that must be removed to disconnect the graph.





Average Connectivity

- Node connectivity is based on a “worst case scenario.”
- A more representative measure might be some sort of average connectivity.
- Connected nodes: Nodes i, j are connected if there is a path between them.
- Dyadic connectivity: $k(i,j)$ = minimum number of removed nodes required to disconnect i, j.
- Average connectivity: $\bar{k} = \sum_{i<j} k(i,j) / \binom{n}{2}$

Centrality

- A common goal in network analysis is to identify “central” nodes in a network.
- What does “central” mean?
  - Active?
  - Important?
  - Non-redundant?

Common Centrality Measures

- Four commonly used centrality measures
  - Degree centrality (based on degree)
  - Closeness centrality (based on average distances)
  - Betweeness centrality (based on geodesics)
  - Eigenvector centrality (recursive: similar to Google page rank methods)

Degree Centrality

- Idea: A central actor is one with many connections.
- This motivates the measure of degree centrality
  - Undirected degree centrality: $c_i^d = \sum_{j:j \neq i} A_{ij}$
  - Outdegree centrality: $c_i^o = \sum_{j:j \neq i} A_{ij}$
  - Indegree centrality: $c_i^i = \sum_{j:j \neq i} A_{ji}$

Closeness Centrality

- Idea: A central node is one that is close, on average, to other nodes.
- This motivates the idea of closeness centrality
  - (Geodesic) distance: d_{ij} is the minimal path length from i to j
  - Closeness centrality: $c_i = 1/\sum_{j:j\neq i} d_{ij} = 1/[(n-1)\bar{d}_i]$

Betweeness Centrality

- Idea: A central actor is one that acts as a bridge, broker or gatekeeper
  - Interaction between unlinked nodes goes through the shortest path (geodesic)
  - A "central" node is one that lies on many geodesic
- This motivates the idea of betweenness centrality
  - $g_{jk} =$ number of geodesics between nodes j and k
  - $g_{jk}(i)$ = number of geodesics between nodes j and k going through i
  - $c_i = \sum_{j<k} g_{jk}(i)/g_{jk}$

Betweeness Centrality

- Interpretation: $g_{jk}(i)/g_{jk}$ is the probability a "message" from j to k goes through i
  - j and k have g_{jk} routes of communication
  - i is on $g_{jk}(i)$ of these routes
  - A randomly selected route contains i with probability $g_{jk}(i)/g_{jk}$

Eigenvector Centrality

- Idea: A central actor is connected to other central actors.
- This definition is recursive.
- Eigenvector centrality: The centrality of each vertex is proportional to the sum of the centralities of its neighbors
  - Formula: $c_i \propto \sum_{j:j \neq i} A_{ji} c_j$
  - Central vertices are those with many central neighbors
  - A variant of eigenvector centrality is used by Google to rank Web pages

Eigenvector Centrality

- Using matrix algebra, such a vector of centralities satisfies

$$Ac = \lambda c$$

  where the diagonal of A are zeros.

- A vector c satisfying the above equation is an eigenvector of A.

- There are generally multiple eigenvectors. The centrality is taken to be the one corresponding to the largest value of λ
  - This corresponds with the best rank-1 approximation to A
  - Nodes with large c_i's have "strong activity" in the "primary dimension" of A

Google PageRank

- PageRank relies on the uniquely democratic nature of the web by using its vast link structure as an indicator of an individual page's value.
- In essence, Google interprets a link from page A to page B as a vote, by page A, for page B.
- But, Google looks at more than the sheer volume of votes, or links a page receives.
- It also analyzes the page that casts the vote.
- Votes cast by pages that are themselves "important" weigh more heavily and help to make other pages "important."

Network Centralization

- Node-level indices:
  Let $c_1, \ldots, c_n$ be node-level centrality measures

  $c_i =$ centrality of node i by some metric (e.g. degree centrality)

- Network-level indices: How centralized is the network?

- Let $c^* = \max\{c_1, \ldots, c_n\}$

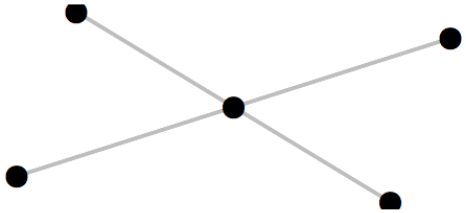
- Freeman's (1979, Social Network) general formula for centralization

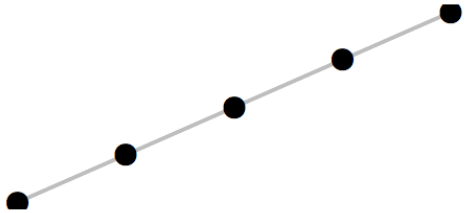
$$C_D = \frac{\sum_i [c^* - c_i]}{[(n-1)(n-2)]}$$

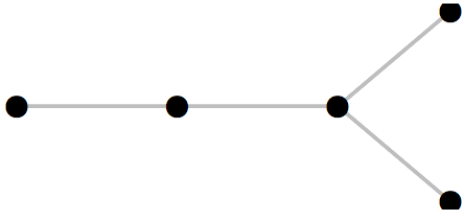
  - $C_D = 0$ if all nodes are equally central
  - C_D is large if one node is most central

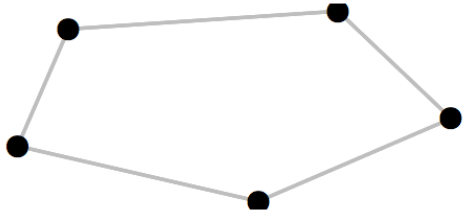
Networks for Comparison

- Compare the following graphs:
- These are the star graph, line graph, y-graph, and the circle graph
- Which do you feel is most “centralized”?









Network Centrality

- Compute the degree centralization for the four $n = 5$ graphs
- Network centrality, C_D:
  - Star graph: 1.0
  - Line graph: 0.58
  - Y-graph: 0.17
  - Circle graph: 0.0

Statistical Network Models: Simple Random Graph

- The simple random graph (SRG) model assumes
  - All ties are formed independently of each other;
  - Each tie exists with some probability p, common across all ties.
- The entries of A are independent and identically distributed (Erdös-Renyi graph, 1959):

$$A_{12}, \ldots, A_{n-1,n} \sim \text{i.i.d. Bernoulli}(p)$$

- The simple random graph (SRG) model:

$$Pr(A_{ij} = 1) = p = \frac{e^{\mu}}{1 + e^{\mu}}$$

Row and Column Effects (RCE) model

- The simple random graph model often does not provide a good representation of A in terms of its degrees
- Row and Column Effects (RCE) model: $\{A_{ij}, i \neq j\}$ are independent with

$$Pr(A_{ij} = 1) = \frac{e^{\mu + a_i + b_j}}{1 + e^{\mu + a_i + b_j}}$$

- The differences among the a_i's represent heterogeneity among nodes in terms of the probability of sending a tie
- And similarly for b_j .

Exponential Random Graph Model

- Both SRG and RCE are special cases of the exponential random graph model (ERGM).
- An ERGM is of the form

$$Pr(A) = g(\theta)e^{\theta^{\mathsf{T}} t(A)}$$

where

- $t(A) = (t_1(A), \ldots, t_m(A))$ is a vector of statistics
- $\theta = (\theta_1, \ldots, \theta_m)$ is a vector of parameters
- $\theta^{\mathsf{T}} t(A) = \sum t_j(A)\theta_j$

RCE as ERGM

- Can the RCE model be expressed as an ERGM?

$$\begin{aligned} Pr(A) &= \prod_{i \neq j} \frac{e^{(\mu + a_i + b_j)A_{ij}}}{1 + e^{\mu + a_i + b_j}} \\ &= e^{\theta^{\mathsf{T}} t(A)} g(\theta) \end{aligned}$$

where

$$\begin{aligned} t(A) &= (A_{\cdot\cdot}, A_{1\cdot}, \ldots, A_{n\cdot}, A_{\cdot 1}, \ldots, A_{\cdot n}) \\ \theta &= (\mu, a_1, \ldots, a_n, b_1, \ldots, b_n) \\ g(\theta) &= \prod_{i \neq j} \{1 + e^{\mu + a_i + b_j}\}^{-1} \end{aligned}$$

- The sufficient statistics that generate the model are the out and indegrees

Latent Variable Models - Stochastic Block Model

- Each node i is independently assigned a (latent) community label $c_i \in \{1, \ldots, K\}$, multinomial with parameter $\{\pi_1, \ldots, \pi_K\}$.
- Given node labels $c_1, \ldots, c_n$, the edges A_{ij} are independent Bernoulli random variables with

$$Pr(A_{ij} = 1) = P_{c_i c_j},$$

  where $P = [P_{ab}]$ is a $K \times K$ symmetric matrix.
- Stochastic equivalence: all nodes within the same block are stochastically equivalent. If $c_i = c_k = c$, then $Pr(A_{ij} = 1) = Pr(A_{kj} = 1) = P_{c_i c_j}$.
- The basic model can be extended in various ways, e.g. including covariates

Latent Factor Models (LFM)

- Matrix form of the block model

$$P_{c_i c_j} = u_i^{\mathsf{T}} P u_j$$

where u_i is a $K \times 1$ vector of all 0's except $u_{i,c_i} = 1$ and P is the $K \times K$ matrix of between class probabilities.

- Generalizing the blockmodel

$$A_{ij} \sim u_i^{\mathsf{T}} D v_j$$

where

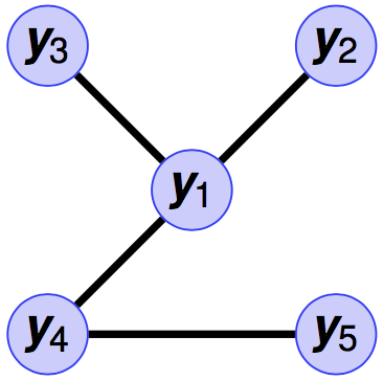
  - u_i is a vector of latent factors describing i as a sender of ties
  - v_j is a vector of latent factors describing j as a receiver of ties
  - D is a diagonal matrix of factor weights

Graphical Models for Undirected Graphs

- Pairwise relations
  - Set of p variables $\Leftrightarrow \boldsymbol{Y} = \left(\boldsymbol{Y}_1, \cdots, \boldsymbol{Y}_p\right)$.
  - Interactions $\Leftrightarrow$ conditional dependencies.
  - Graph:

$$\mathscr{G} = \left(V, E\right), \; V = \{1, \cdots, p\}$$
$$(j,k) \in E \quad \text{if} \quad \boldsymbol{Y}_j \not\perp \boldsymbol{Y}_k \mid \boldsymbol{Y}_{\setminus\{j,k\}}$$

  - Example:



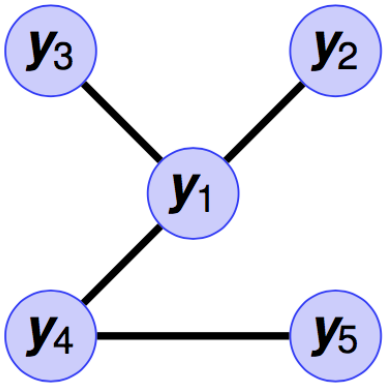
- $\boldsymbol{y}_1 \not\perp \boldsymbol{y}_3 \mid \boldsymbol{y}_2, \boldsymbol{y}_4, \boldsymbol{y}_5$
- $\boldsymbol{y}_1 \perp \boldsymbol{y}_5 \mid \boldsymbol{y}_2, \boldsymbol{y}_3, \boldsymbol{y}_4$

- Goal: reconstruct $\mathscr{G}$ based on n i.i.d. observations from Y.

Gaussian Graphical Model for Undirected Graph

- Model: $\boldsymbol{Y} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$.
- Precision matrix: $\boldsymbol{\Omega} = (\omega_{jk})_{p \times p} = \boldsymbol{\Sigma}^{-1}$
- Conditional independence:

$$\boldsymbol{Y}_j \perp \boldsymbol{Y}_k \mid \boldsymbol{Y}_{\setminus\{j,k\}} \Leftrightarrow \omega_{jk} = 0$$



$$\leftrightarrow \boldsymbol{\Omega} = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} & \omega_{14} & 0 \\ \omega_{21} & \omega_{22} & 0 & 0 & 0 \\ \omega_{31} & 0 & \omega_{33} & 0 & 0 \\ \omega_{41} & 0 & 0 & \omega_{44} & \omega_{45} \\ 0 & 0 & 0 & \omega_{54} & \omega_{55} \end{bmatrix}$$

- Graph connectivity $\Longleftrightarrow$ non-zero off-diagonals of $\boldsymbol{\Omega}$

Conditional Independence

- $(Y_1, \cdots, Y_p)^T \sim N(0, \Sigma)$ with $\Sigma = \Omega^{-1}$.
- Density of $\boldsymbol{Y}$: $f(\boldsymbol{y}) = \frac{1}{\sqrt{(2\pi)^p det(\Sigma)}} \exp(-\frac{1}{2}\boldsymbol{y}^T \Omega \boldsymbol{y})$.
- Let $Z = (Y_3, \cdots, Y_p)$ and $Y = (Y_1, Y_2)$.

$$Y|Z \sim N(\underbrace{\mu_Y + (Z - \mu_Z)^T \Sigma_{ZZ}^{-1} \Sigma_{ZY}}_{\mu_{Y|Z=0}}, \underbrace{\Sigma_{YY} - \Sigma_{ZY}^T \Sigma_{ZZ}^{-1} \Sigma_{ZY}}_{\boldsymbol{V}ar(Y|Z)}).$$

$$\Sigma = \begin{pmatrix} \Sigma_{ZZ} & \Sigma_{ZY} \\ \Sigma_{ZY}^T & \Sigma_{YY} \end{pmatrix}, \quad \Omega = \begin{pmatrix} \Omega_{ZZ} & \Omega_{ZY} \\ \Omega_{ZY}^T & \Omega_{YY} \end{pmatrix}.$$

- Block Inverse: $\Omega_{YY} = \Sigma_{YY} - \Sigma_{ZY}^T \Sigma_{ZZ}^{-1} \Sigma_{ZY}$.
- Conditional density of Y_1, Y_2 given rest is

$$f(y_1, y_2 | y_3, \cdots, y_p) \propto f_1(y_1) f_2(y_2) \exp(-y_1 \omega_{12} y_2),$$

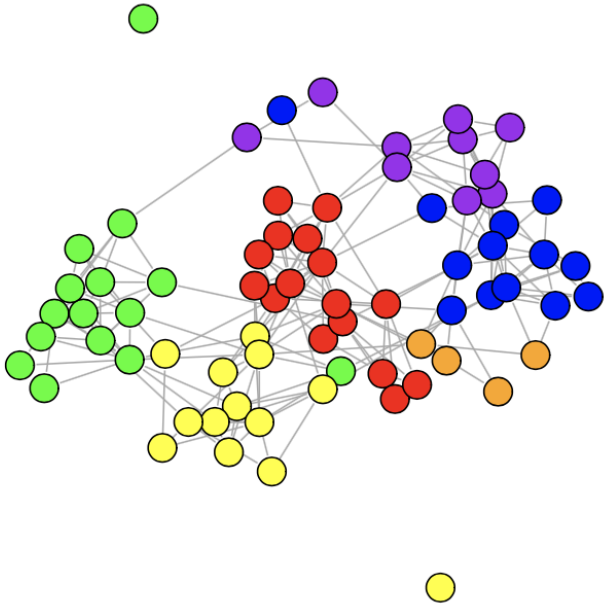
  where ω_{12} is the $(1,2)$-element of Ω_{YY}.
- Conditional independence of Y_1, Y_2 given rest, iff $\omega_{12} = 0$.

Community detection

- One of the most studied problems in network analysis
- Communities are cohesive groups of nodes
- Most common interpretation: many links within and few links between
- The community detection problem is typically formulated as finding a disjoint partition $V = V_1 \cup \cdots \cup V_K$

Example: a school friendship network

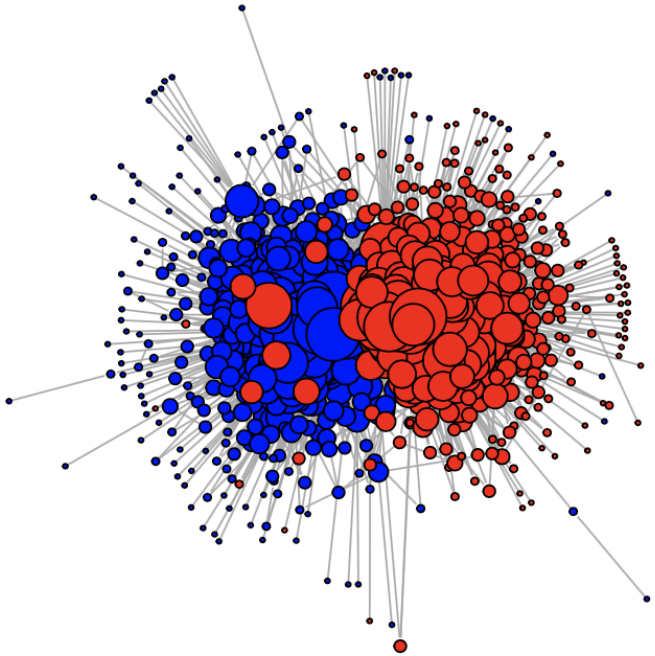
Colors represent grades



Example: a political blogs network

Adamic & Glance (2005): nodes are $\approx$ 1200 political blogs

- Manually labelled liberal or conservative
- Edges are web links (ignoring direction)
- Node size in the plot is proportional to log degree



Community detection methods

- Optimizing a global criterion over all partitions: graph cuts, modularity (Newman & Girvan, 2004), and many others, often with spectral approximations
- Modularity is a measure of the network structure, also called groups, clusters or communities
- Networks with high modularity have dense connections between the nodes within modules but sparse connections between nodes in different modules
- Fitting a model for a network with communities: stochastic block models and extensions, e.g. degree-corrected (Karrer & Newman, 2010) and mixed membership (Airoldi et al, 2008), latent class models (Hoff, Handcock et al, 2002-08)

The stochastic block model

Holland et al (1983)

- Each node i is independently assigned a community label $c_i \in \{1, \ldots, K\}$, multinomial with parameter $\pi = (\pi_1, \ldots, \pi_K)^T$.
- Given node labels $c = (c_1, \ldots, c_n)$, the edges A_{ij} are independent Bernoulli random variables with

$$E(A_{ij}) = P_{c_i c_j} \ ,$$

  where $P = [P_{ab}]$ is a $K \times K$ symmetric matrix.
- The "null" model ($K = 1$): the Erdos-Renyi graph (all edges form independently with probability p)

Degree-corrected stochastic block model (DCBM)

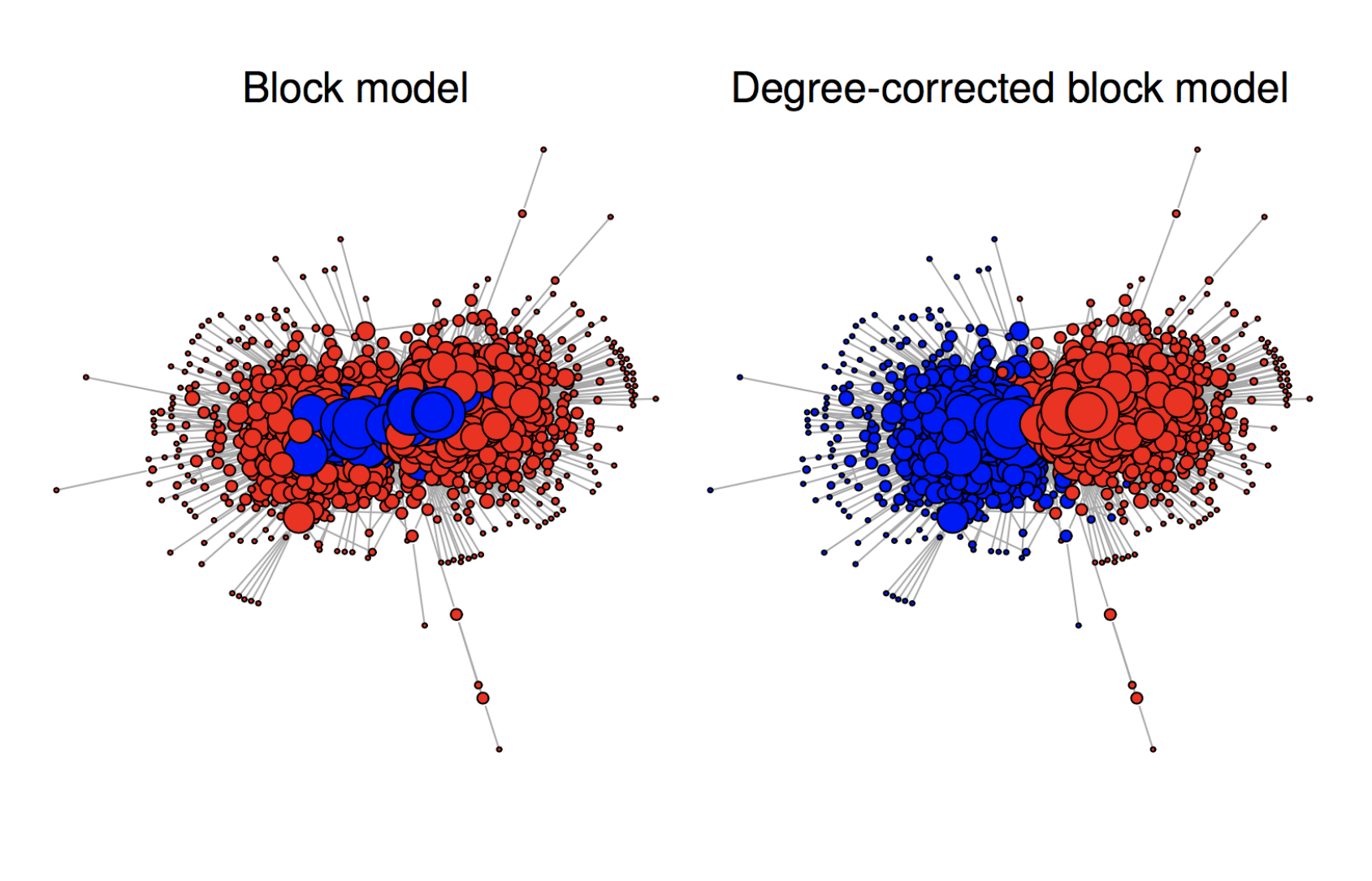
Karrer & Newman (2010)

- Limitation of the block model: expected degree within one community is the same for all nodes, which does not allow for "hubs"
- DCBM: Each node is associated with a degree parameter θ_i, and

$$E(A_{ij}) = \theta_i \theta_j P_{c_i c_j} \ .$$

- The "null" model ($K = 1$): the expected degree random graph, a.k.a. configuration model or the Chung-Lu model. All edges form independently with $P(A_{ij} = 1) \propto \theta_i \theta_j$.

The political blogs example



Modularity

Newman & Girvan (2004)

- A very popular criterion-based approach to community detection
- Maximize observed number of edges within communities minus expected under a null model ($K = 1$), over all label assignments e:

$$Q(e) = \sum_{ij} [A_{ij} - E(A_{ij})] 1(e_i = e_j)$$

  where $E(A_{ij})$ is the (estimated) expectation under the null model.
- Maximization can be approximated by an eigenvalue problem (Newman 2006)

Notation

For any community label assignment $e = \{e_1, ..., e_n\}$, $e_i \in \{1, \ldots, K\}$, define

$$O_{kl} = \sum_{ij} A_{ij} 1\{e_i = k, e_j = l\}, \text{ \# edges between communities } k \text{ and } l$$

$$O_k = \sum_{l} O_{kl}, \text{ \# edges from community } k$$

$$L = \sum_{kl} O_{kl}, \text{ total \# edges}$$

$$n_k = \sum_{i} 1\{e_i = k\}, \text{ \# nodes in community } k$$

The $K \times K$ matrix $O(e) = [O_{kl}(e)]$ depends only on e and the data, not the model.

Null models for modularity

- Block model $\Rightarrow$ the null model is the Erdos-Renyi graph. The "ER modularity" is

$$Q_{ER}(e) = \sum_k \left(O_{kk} - \frac{n_k^2}{n^2} L \right)$$

- Degree-corrected block model $\Rightarrow$ the null model is the configuration model, giving the original Newman-Girvan modularity

$$Q_{NG}(e) = \sum_k \left(O_{kk} - \frac{O_k^2}{L^2} L \right)$$

Summary of community detection criteria

	Block model	Degree-corrected
Profile likelihood	$\sum_{kl} O_{kl} \log \frac{O_{kl}}{n_k n_l}$	$\sum_{kl} O_{kl} \log \frac{O_{kl}}{O_k O_l}$
Modularity	$\sum_k (O_{kk} - \frac{n_k^2}{n^2} L)$	$\sum_k (O_{kk} - \frac{O_k^2}{L^2} L)$

- The block model "weighs" communities by the number of nodes, and DCBM by the number of edges
- Modularity looks for communities with more links within than between

Community detection: Cluster Louvain

- Cluster Louvain algorithm in igraph (Tom Gregorovic):
  - Based on the modularity measure and a hierarchical approach
  - Initially each vertex is assigned to a community on its own.
  - In every step, vertices are re-assigned to communities in a local, greedy way: each vertex is moved to the community with which it achieves the highest contribution to modularity.
  - When no vertices can be reassigned, each community is considered a vertex on its own, and the process starts again with the merged communities.
  - The process stops when there is only a single vertex left or when the modularity cannot be increased any more in a step.

Network Analysis in R

List of some R packages

- General Networks: **igraph**, network, sna
- Visualization: visNetwork, ndtv, d3network
- Statistical Modeling: ggm, statnet, ergm, siena
- More on graphical models:
  `https://cran.r-project.org/web/views/gR.html`

Create a network

Any formats describing structure of a network

- Adjacency matrix
- Edge lists
- Adjacency lists
- Some other formats: GML, GraphML, Pajek, etc.

Adjacency matrix

- A square matrix
- Function: `graph_from_adjacency_matrix`

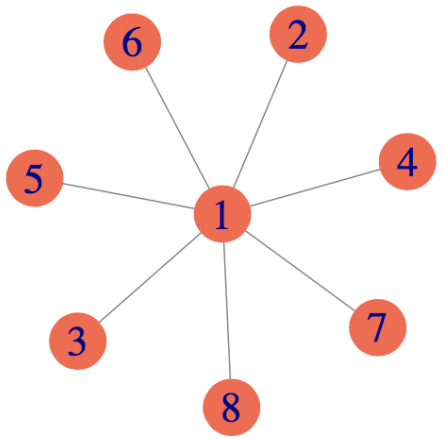
	Node 1	Node 2	Node 3	Node 4	Node 5
Node 1	1	1	1	1	1
Node 2	1	1	0	0	0
Node 3	1	0	1	0	0
Node 4	1	0	0	1	0
Node 5	1	0	0	0	1

A toy example: Star network

```
library(igraph)
nnode <- 8; A <- diag(nnode); A[1,] = A[,1] <- 1

net <- graph_from_adjacency_matrix(A, mode = "undirected", diag = FALSE)

plot(net, vertex.size = 30, vertex.label.cex = 2, edge.color = "gray50",
     vertex.color = "tomato", vertex.frame.color = "tomato")
```



Edge lists

- A two column matrix, character or numeric
- Function: `graph_from_edgelist`

MonkeyKing	Master
Bajie	Master
Wujing	Master
Horse	Master
Beauty	Bajie

GML files

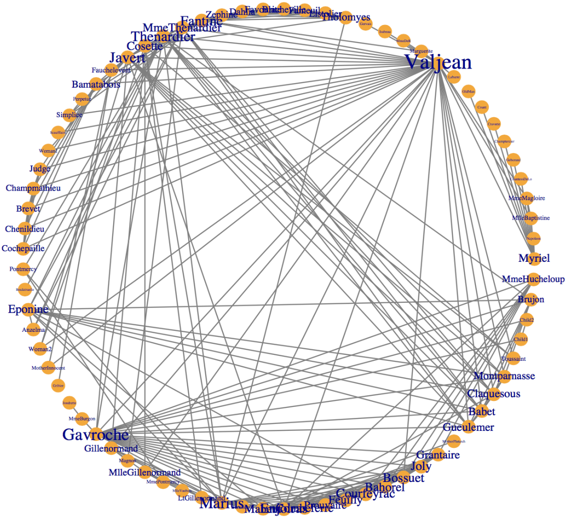
- Graph Modeling Language: a hierarchical ASCII-based file format describing graphs
- Function: `read_graph`
- Les Misérables data

```
graph
[
   node
   [
      id 0
      label "Myriel"
   ]
   node
   [
      id 1
      label "Napoleon"
   ]
.
.
.
edge
   [
      source 1
      target 0
      value 1
   ]
   edge
   [
      source 2
      target 0
      value 8
   ]
]
```

Import data from GML files

```
lesmis <- read_graph("lesmis.gml", format = c("gml"))

plot(lesmis, vertex.size = 5, edge.color = "gray50",
      vertex.color = "orange", vertex.frame.color = "orange",
     vertex.label.cex = sqrt(degree(lesmis, mode="all"))/5,
     layout = layout.circle)
```


Valjean
Tholomyes
Zephine
Dahlia
Fantine
MmeThenardier
Thenardier
Cosette
Javert
Fauchelevent
Bamatabois
Perpetue
Simplice
Woman1
Judge
Champmathieu
Brevet
Chenildieu
Cochepaille
Pontmercy
Eponine
Anzelma
Woman2
MotherInnocent
MmeBurgon
Gavroche
Gillenormand
Magnon
MlleGillenormand
MmePontmercy
LtGillenormand
Marius
Feuilly
Courfeyrac
Bahorel
Bossuet
Joly
Grantaire
Gueulemer
Babet
Claquesous
Montparnasse
Toussaint
Child1
Child2
Brujon
MmeHucheloup
Myriel
MlleBaptistine
MmeMagloire
Marguerite
Labarre
OldMan
Count

Import data from data frames

- Two data frames: nodes and edges with more attributes
- Function: `graph_from_data_frame`
- Media network of hyperlinks and mentions among news sources

```
nodes <- read.csv("Media-NODES.csv", header=T, as.is=T)
links <- read.csv("Media-EDGES.csv", header=T, as.is=T)

nodes[c(1, 7), ]
```

```
##    id    media media.type type.label audience.size
## 1 s01 NY Times          1  Newspaper            20
## 7 s07      CNN          2         TV            56
```

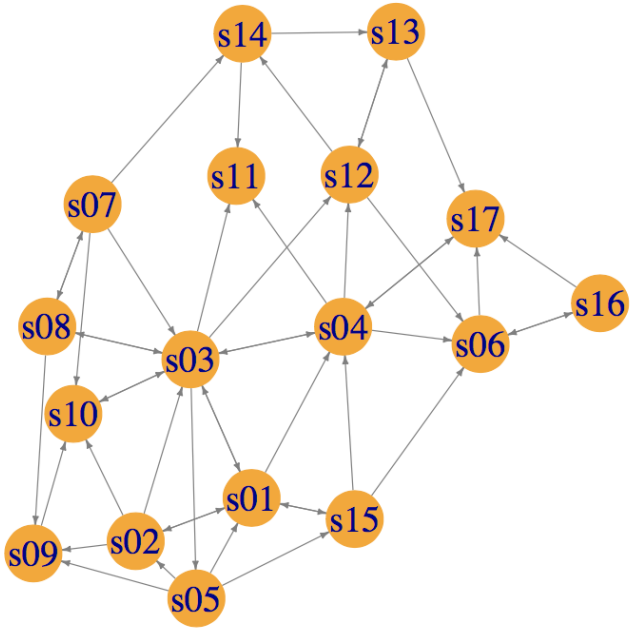
```
links[c(1, 13), ]
```

```
##    from  to      type weight
## 1   s01 s02 hyperlink     22
## 13  s03 s10   mention      2
```

Data source: POLNET 2016 Workshop, Katherine Ognyanova, Rutgers University

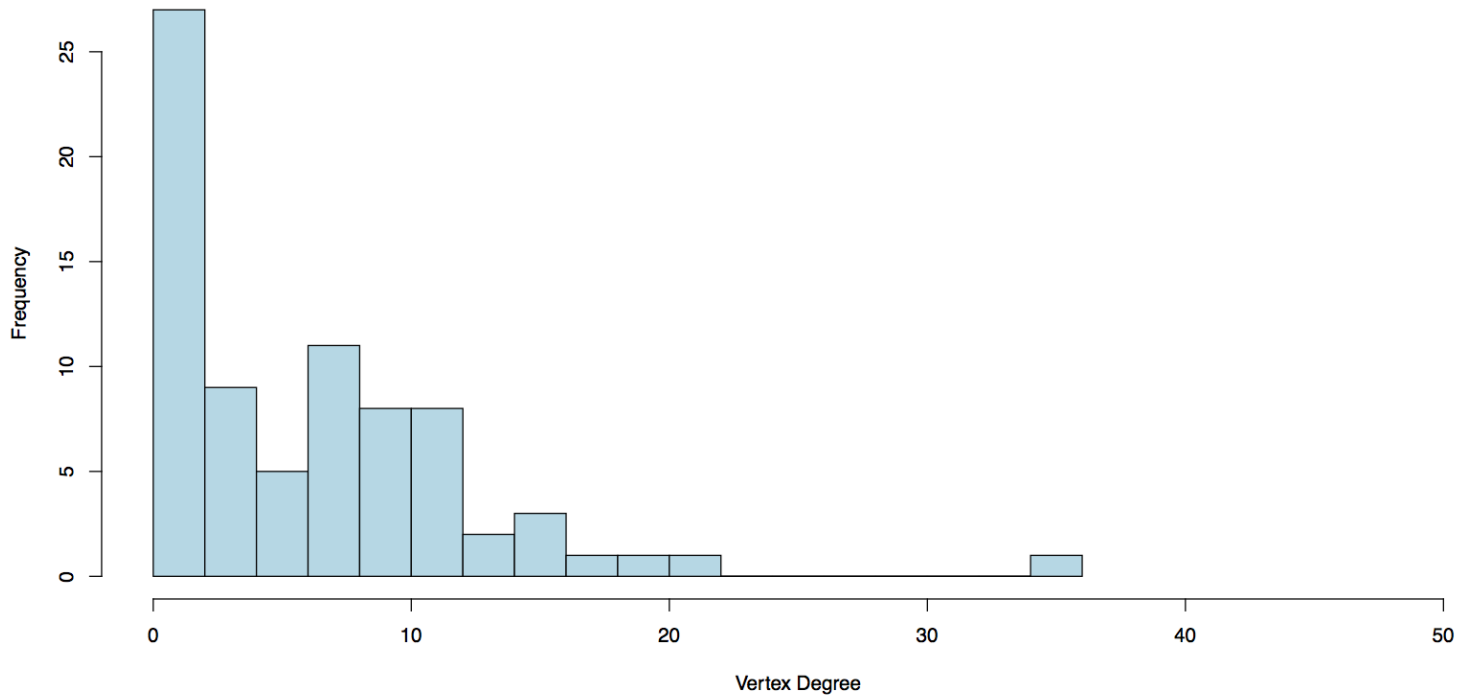
```r
net <- graph_from_data_frame(d = links, vertices = nodes, directed = T)

plot(net, vertex.size = 20, vertex.label.cex = 2, edge.arrow.size = .4,
     edge.color = "gray50", vertex.color = "orange",
     vertex.frame.color = "orange")
```



Histogram of degree for Les Mis?erables network

```
hist(degree(lesmis), breaks = 20, col="lightblue", xlim=c(0, 50),
     xlab="Vertex Degree", ylab="Frequency", main="")
```



Community detection: Network of Thrones

- A Storm of Swords: 3^{rd} book of A Song of Ice and Fire
- 107 characters (vertices)
- 353 weighted edges: names appear within 15 words
- `cluster_louvain`: multi-level modularity optimization algorithm
- Other community detection algorithms:
  `http://igraph.org/c/doc/igraph-Community.html`

Data source: A. Beveridge and J. Shan, "Network of Thrones," Math Horizons Magazine , Vol. 23, No. 4 (2016), pp. 18-22.

Bran

Robert

Cersei

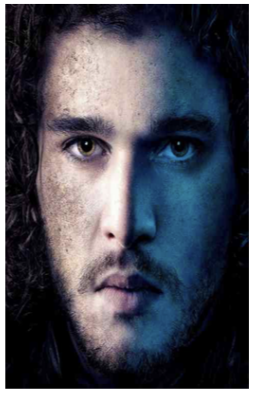
Jaime

Arya

Catelyn

Robb

Tyrion



Jon

Daenerys

Sansa

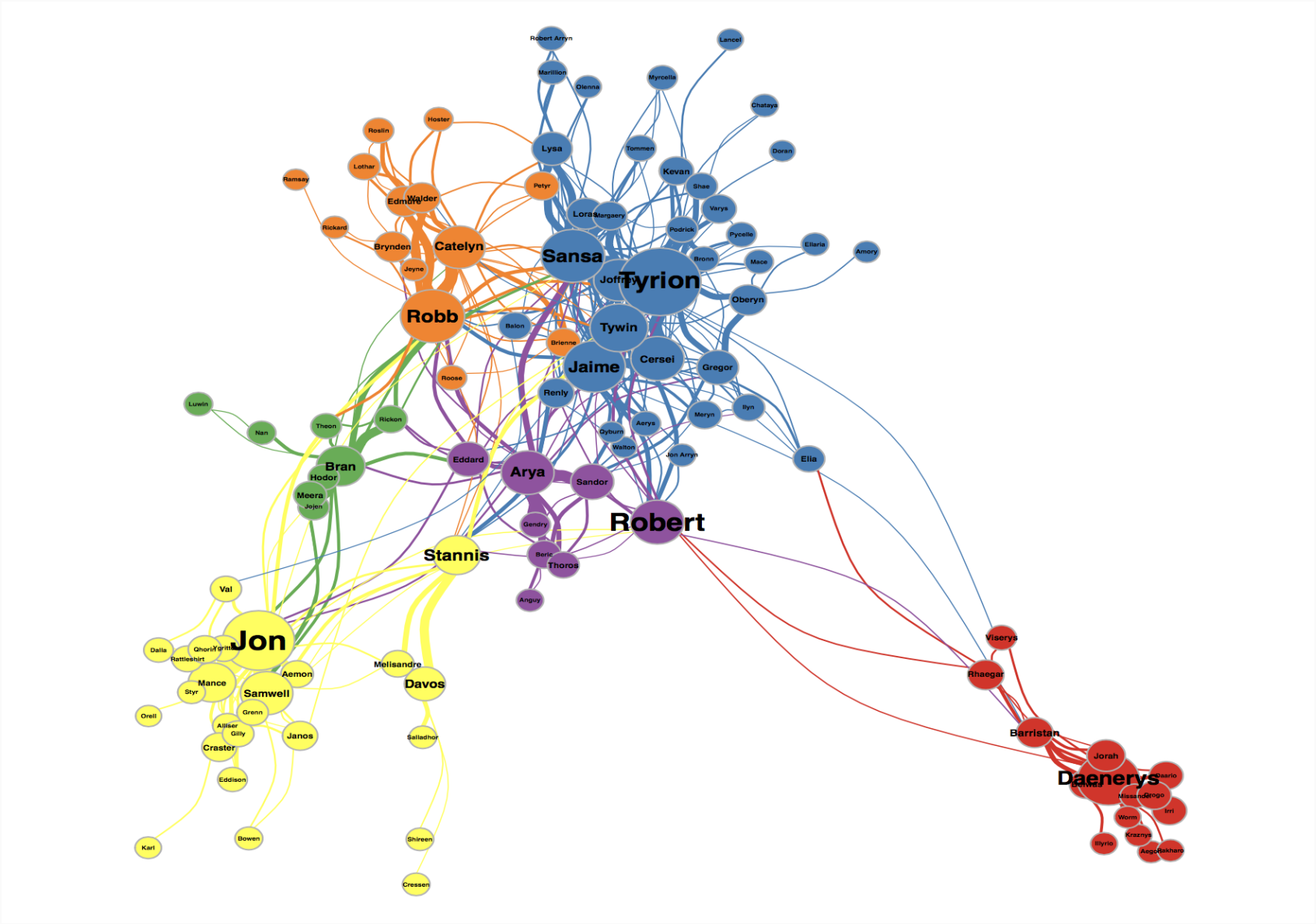
Stannis

```
source("http://michael.hahsler.net/SMU/ScientificCompR/code/map.R")
library(RColorBrewer); colors <- brewer.pal(8, "Set1")

thrones <- read.csv("stormofswords.csv", header = T, as.is = T)
thrones_net <- graph_from_data_frame(d = thrones, directed = F)

library(ForceAtlas2)
layout <- layout.forceatlas2(thrones_net, k = 800, gravity = 1,
                             iterations=4000)

community <- cluster_louvain(thrones_net)
pr <- page.rank(thrones_net)$vector
btw <- betweenness(thrones_net)
vertex.col <- colors[community$membership]
edge.start <- ends(thrones_net, es=E(thrones_net), names = F)[,1]
el.col <- vertex.col[edge.start]
igraph.options(plot.layout = layout, vertex.label.family = "Helvetica",
               vertex.label.font = 2, vertex.frame.color =  "gray70",
               vertex.label.cex = map(btw,c(1,4))/3.5,
               vertex.size = map(pr, c(5,16)), vertex.label.color = "black",
               vertex.color = vertex.col, edge.width = E(thrones_net)$Weight/6,
               edge.curved = 0.4, edge.color= el.col)
plot(thrones_net)
```


Robert Arryn
Lancel
Marillion
Myrcella
Olenna
Chataya
Hoster
Roslin
Lysa
Tommen
Doran
Lothar
Kevan
Ramsay
Petyr
Shae
Walder
Edmure
Loras
Margaery
Varys
Rickard
Podrick
Pycelle
Brynden
Catelyn
Sansa
Ellaria
Amory
Bronn
Mace
Jeyne
Joffrey
Tyrion
Oberyn
Robb
Balon
Tywin
Brienne
Jaime
Cersei
Gregor
Roose
Renly
Luwin
Meryn
Illyn
Nan
Theon
Rickon
Aerys
Gyburn
Walton
Jon Arryn
Eddard
Arya
Sandor
Ella
Bran
Hodor
Meera
Jojen
Robert
Gendry
Stannis
Beric
Thoros
Val
Anguy
Jon
Viserys
Dalla
Qhorin
Rattleshirt
Mellisandre
Aemon
Rhaegar
Mance
Styr
Davos
Samwell
Grenn
Orell
Barristan
Allister
Gilly
Janos
Salladhor
Craster
Jorah
Daenerys
Eddison
Drogo
Irri
Worm
Karl
Bowen
Shireen
Illyrio
Aegon
Cressen