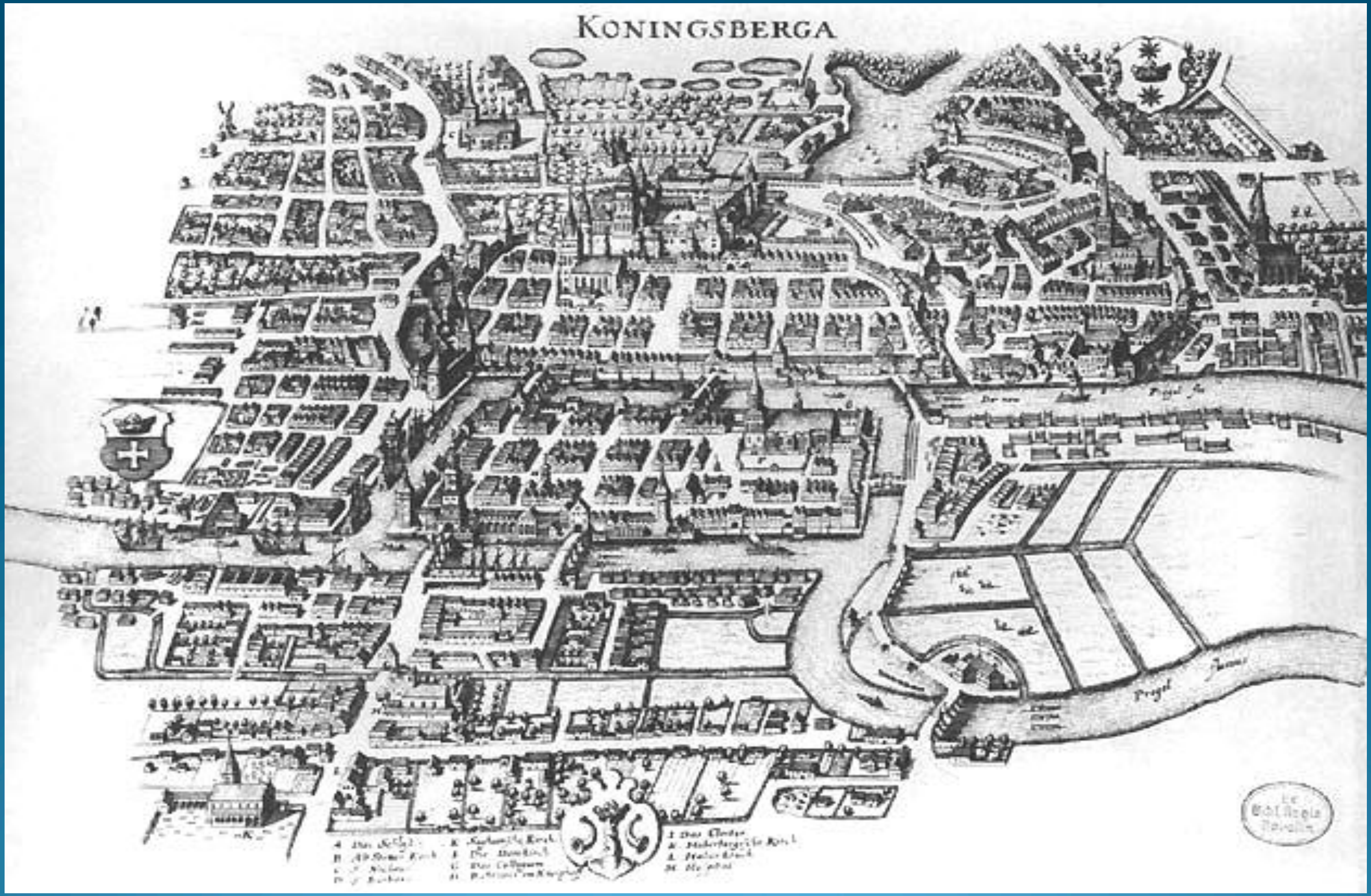
Various Network Programs



@jotis13

Degree (Twitter Example)

- in-degree (# of people tweeting about you)
  - sum of all edges going to a node
- out-degree (# of people you tweet about)
  - sum of all edges going out of a node

- weighted in-degree (# tweets about you)
  - sum of all edges going to a node multiplied by their weights
- weighted out-degree (# tweets about other people)
  - sum of all edges going out from a node multiplied by their weights

- degree (# of people you connect to)
  - sum of all edges
- weighted degree (# times connect to people)
  - sum of all edges multiplied by their weights

@jotis13

Global Network Analysis

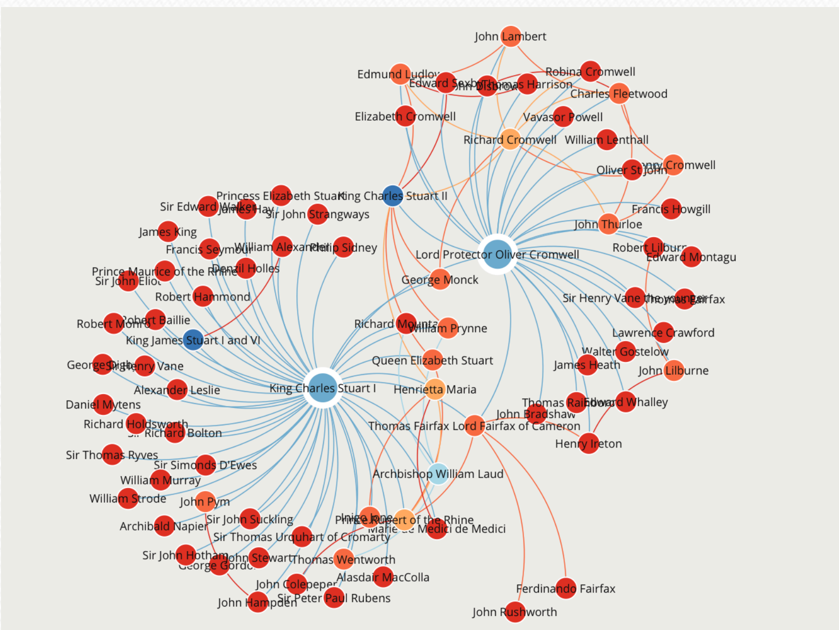
- connected components
  - weakly connected = undirected
    - requires 1 edge connecting two nodes, more are optional
  - strongly connected = directed
    - requires enough edges for effective 2-way travel between two nodes
    - will be more than 1 edge, even if path “back” is not direct
- network density
  - how close is the network to complete? e.g. what percentage of edges are extant?
- network diameter
  - what is the longest shortest path between two nodes?

Node “Centrality” Measures

- betweenness centrality:
  - the number of the shortest paths through the network that go through a node
- closeness centrality:
  - the average distance from a node to all other nodes in the network
- eccentricity:
  - the distance from a node to the farthest node from it in the network
  - the center of the graph is the node(s) with the minimum eccentricity
  - the maximum eccentricity is also the diameter

How Can You Analyze Them?

E.g. in a social network



- who knows the most people?
- who knows the most highly connected people?
- how many disconnected subnetworks are there?
- whose removal severs the network into disconnected subnetworks?
- how many “hops” does it take to get from one person in the network to another?
- who do most of the shortest hops go through?
- are there any smaller cliques or “communities” within the network?

@jotis13

Eigenvector Centrality

- we call our adjacency matrix $A = (a_{v,t})$
  - each $a_{v,t}$ corresponds to the 0 or 1 at the intersection of row v and column t
- eigenvector centrality for a vertex v is

$$x_v = \frac{1}{\lambda} \sum_{t \in M(v)} x_t = \frac{1}{\lambda} \sum_{t \in G} a_{v,t} x_t$$

- note: $\mathbf{Ax} = \lambda \mathbf{x}$

	And	Bil	Car	Dan	Ele	Fra	Gar
Andy		1	0	1	0	0	1
Bill	1		1	0	1	0	0
Carol	1	1		1	1	0	0
Dan	1	1	1		0	0	0
Elena	0	0	0	0		1	0
Frank	0	0	0	0	1		0
Garth	1	1	0	0	0	0	

@jotis13

Some (But Not All!) Network Programs

Web-Based Options

- Palladio – hdlab.stanford.edu/palladio/
- Network Navigator – http://dh-web.hss.cmu.edu/network_navigator/
- NodeGoat – nodegoat.net

Application Options

- Gephi – gephi.org
- Cytoscape – www.cytoscape.org
- Pajek – mrvar.fdv.uni-lj.si/pajek/ (Windows only)
- NodeXL – nodexl.codeplex.com (Excel, Windows only)

Programming Options

- Python's NetworkX library
- R's Tidyverse/Tidygraph libraries
- JavaScript's D3 library (visualization only)