NETWORK ANALYSIS

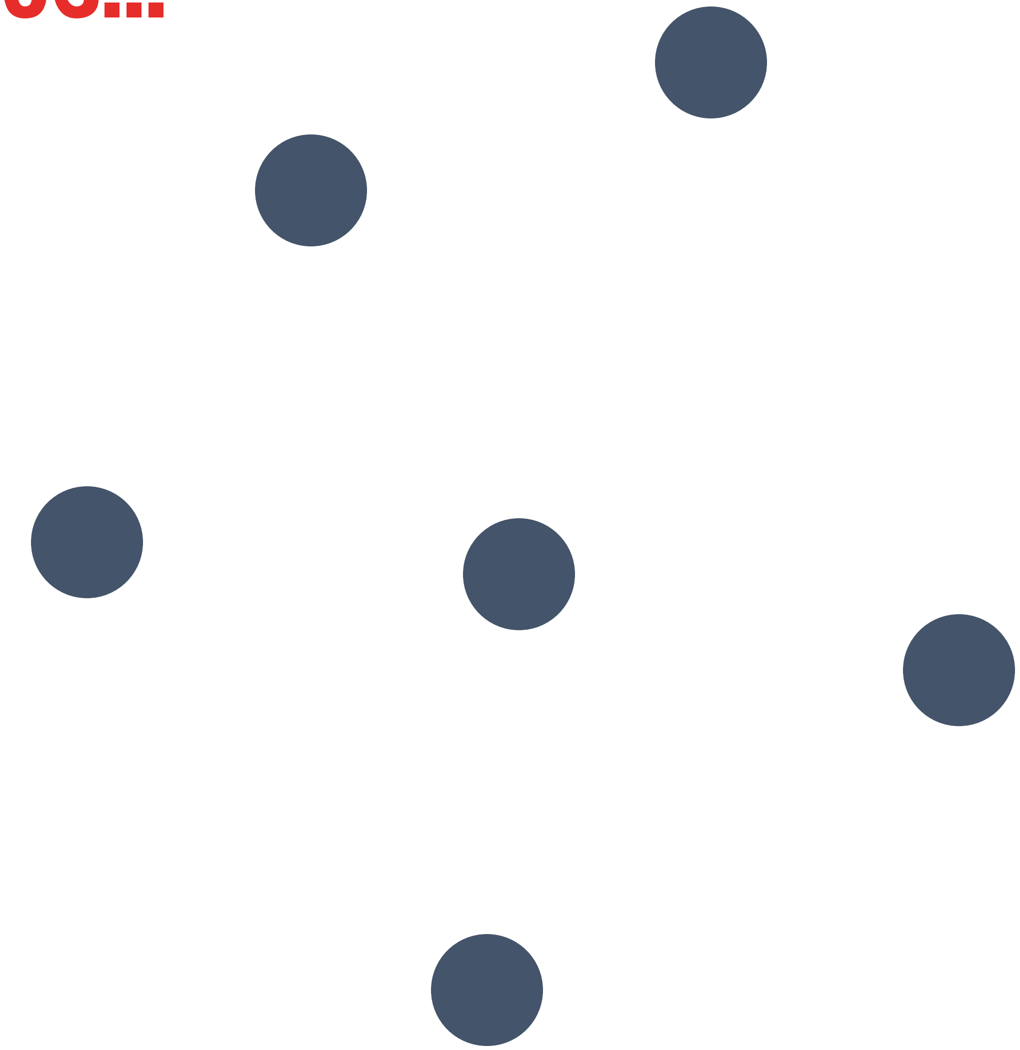
FOR TODAY
DOWNLOAD
“Datathon 5 - Marvel Network.zip”

INSTALL GEPHI
www.gephi.org

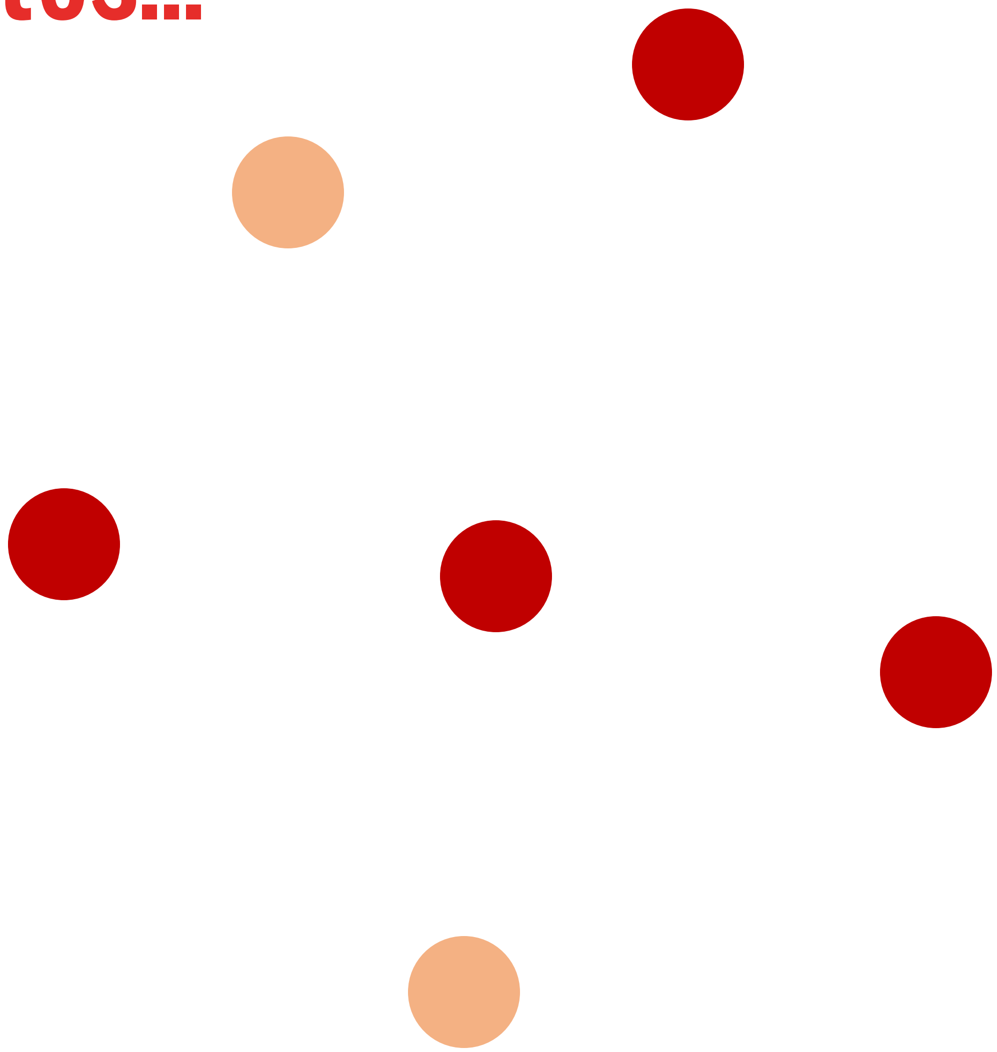
UNIVERSITY OF CALGARY

WHAT ARE NETWORKS / GRAPHS?

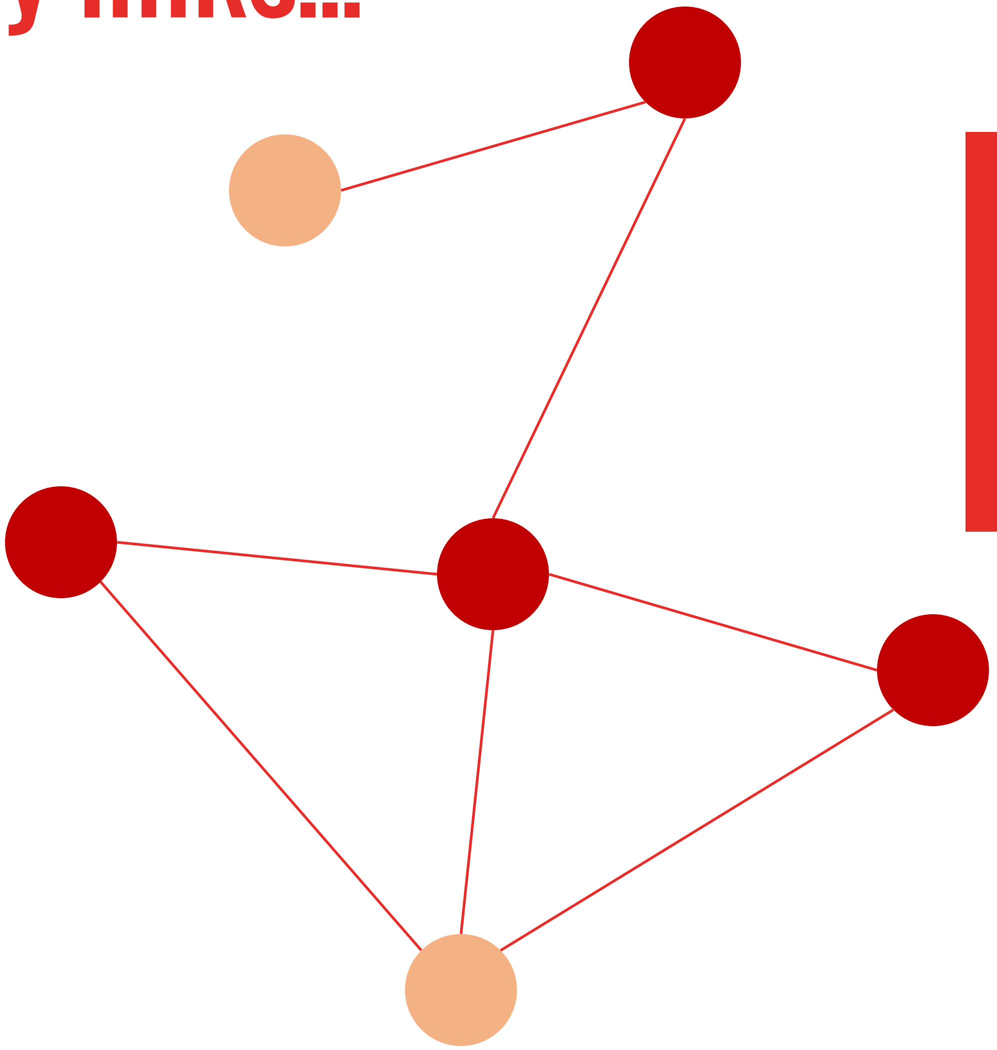
A set of nodes...



With attributes…

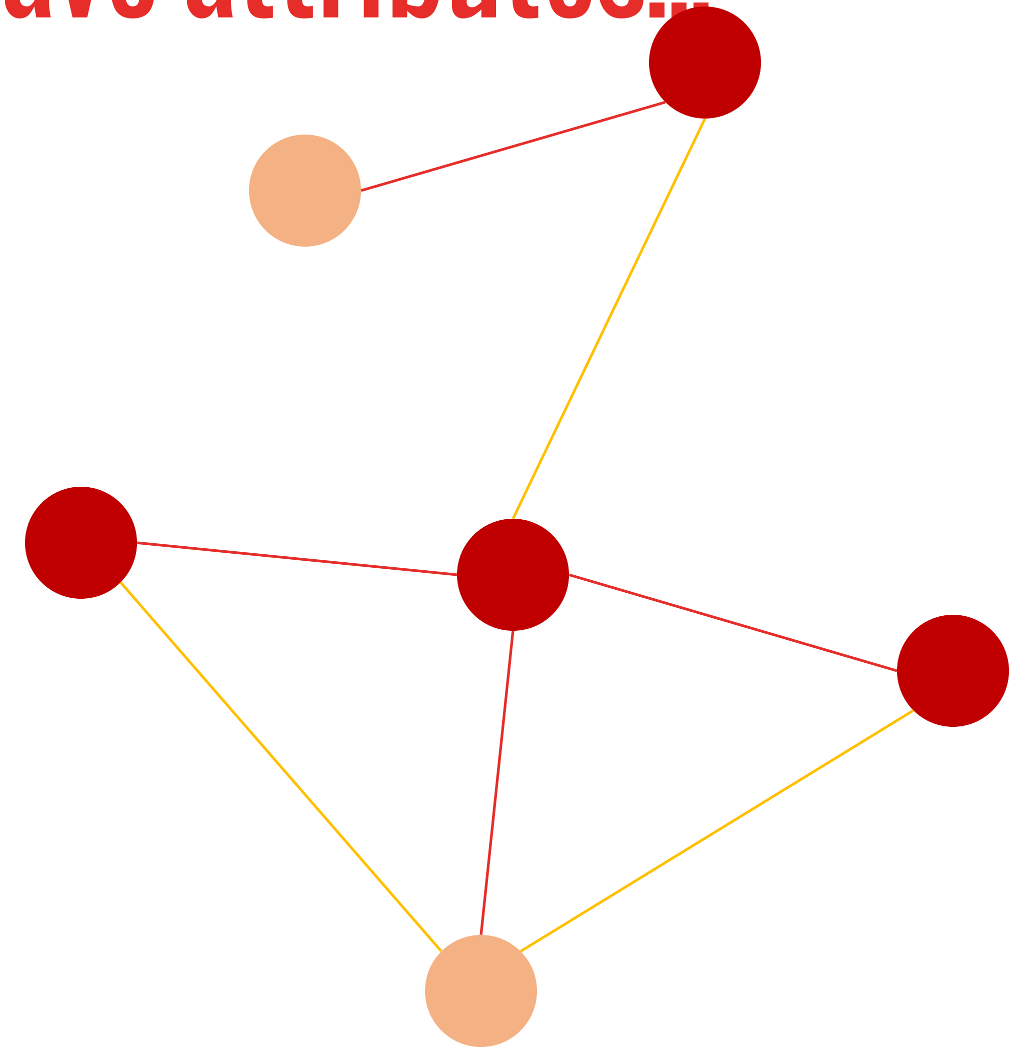


Connected by links...

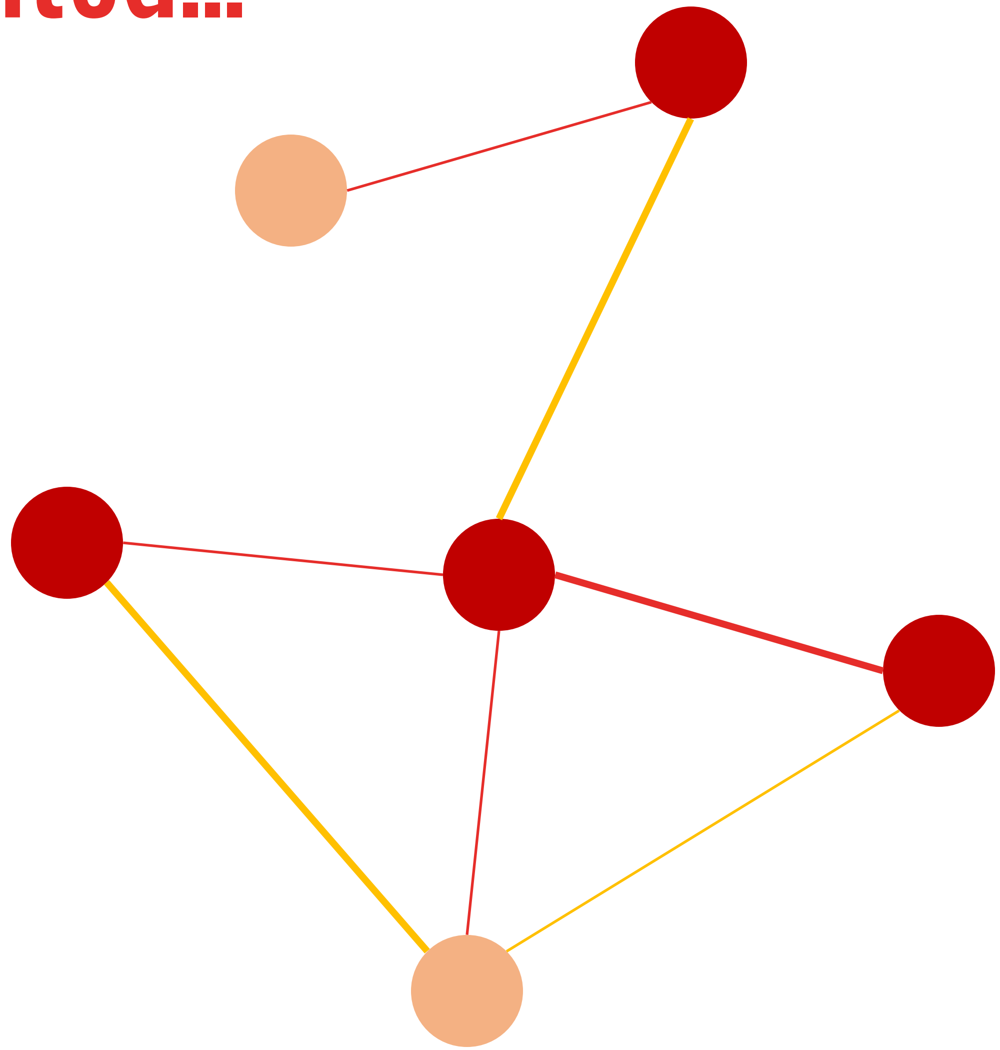


Unlike a tree, a graph **can contain cycles**, and there are often several paths from one node to another

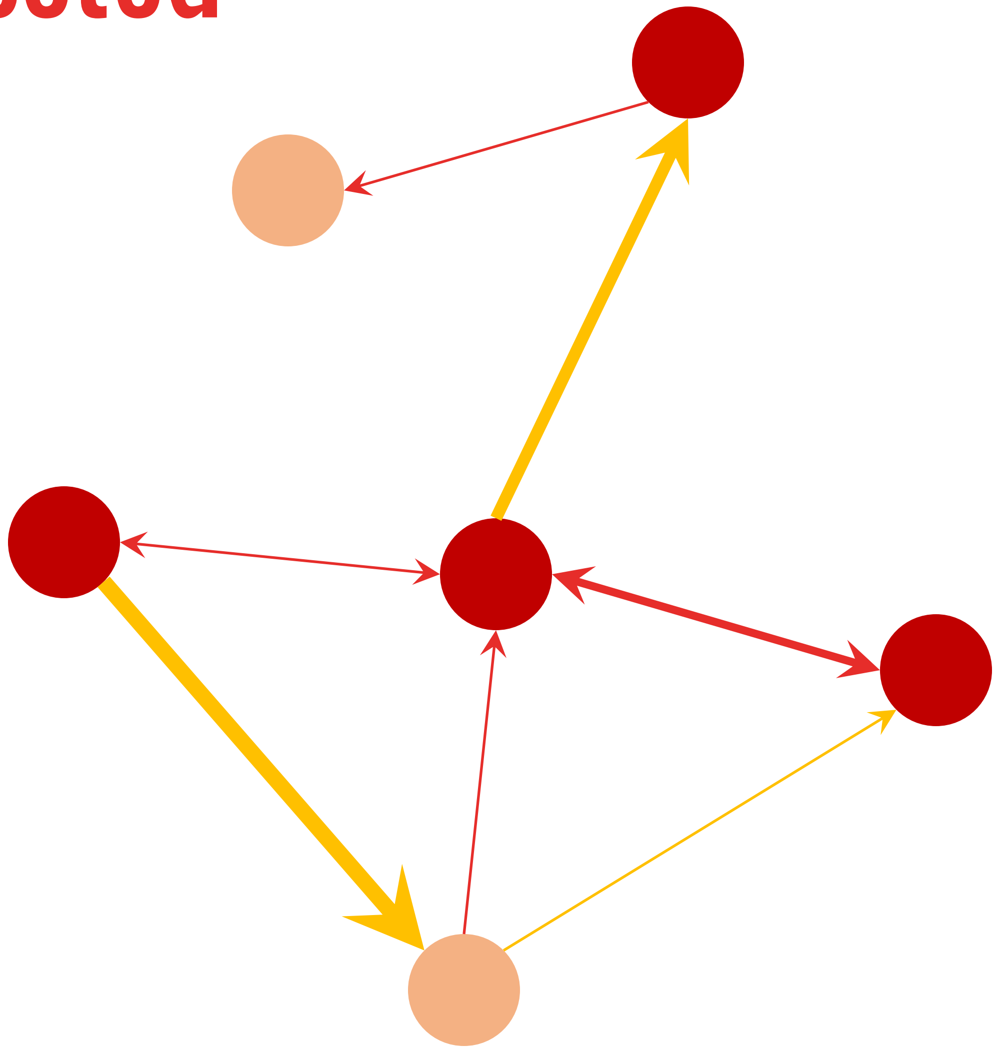
Which can have attributes...



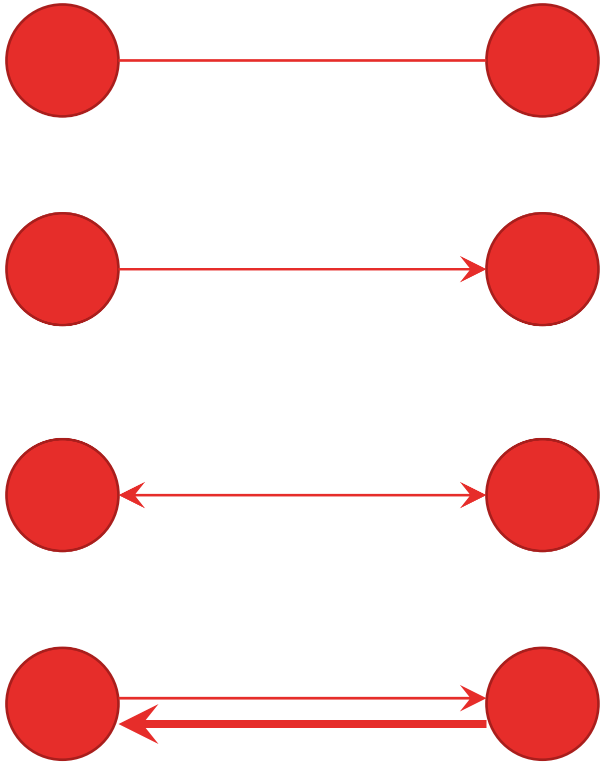
can be weighted...



…and/or directed



PROPERTIES



Undirected edges

Directed edges

Bi-directional symmetrical edges

Bi-directional asymmetrical edges

LOTS OF DIFFERENT KINDS OF NETWORK DATA

Social Networks
- **Explicit** (Facebook friends, Twitter followers, favorite brands, etc.)
- **Implicit** (Co-authorship networks, name co-occurrence in docs, etc.)

Computer Networks
Infrastructure Networks (Roads, electricity, transit, etc.)
Concept Networks
Biological Processes
…

REPRESENTING GRAPHS

Usually two data structures - one for **nodes** and one for **edges**.

NETWORK ELEMENTS: NODES

Usually represented as
data tuples with associated properties.

NETWORK ELEMENTS: EDGES

Directed (also called arcs)

A -> B

A likes B, A gave a gift to B, A is B' s child

Undirected

A <-> B or A – B

A and B like each other

A and B are siblings

A and B are co-authors

Edge attributes

weight (e.g. frequency of communication)

ranking (best friend, second best friend...)

type (friend, relative, co-worker)

...

LIST REPRESENTATIONS

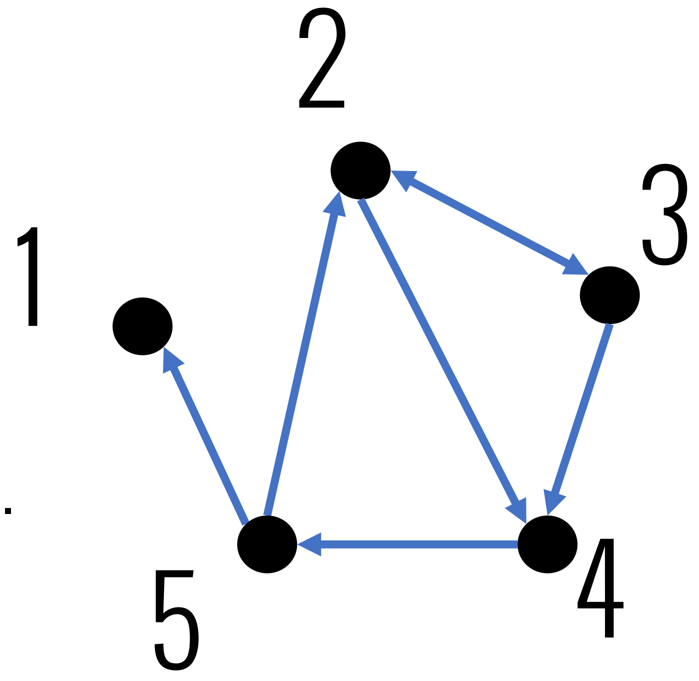
Edge list (simplest)

2 3
2 4
3 2
3 4
4 5
5 2
5 1

Adjacency list

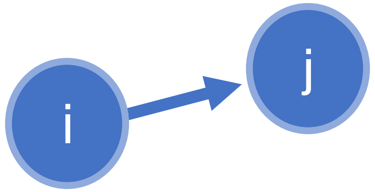
Easier to work with if network is large and sparse.
Can quickly retrieve all neighbors for a node.

1:
2: 3 4
3: 2 4
4: 5
5: 1 2

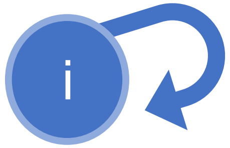


ADJACENCY MATRICES

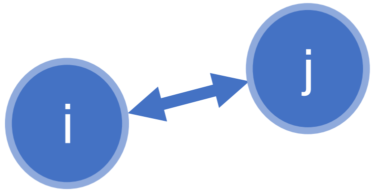
Represent edges as a matrix



A_{ij} = 1 if node i has an edge to node j
= 0 if node i does not have an edge to j

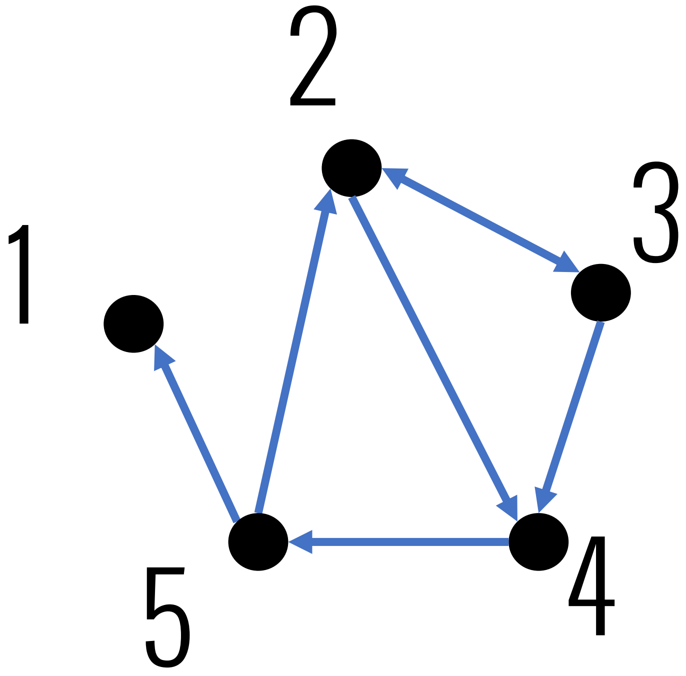


A_{ii} = 0 unless the network has self-loops



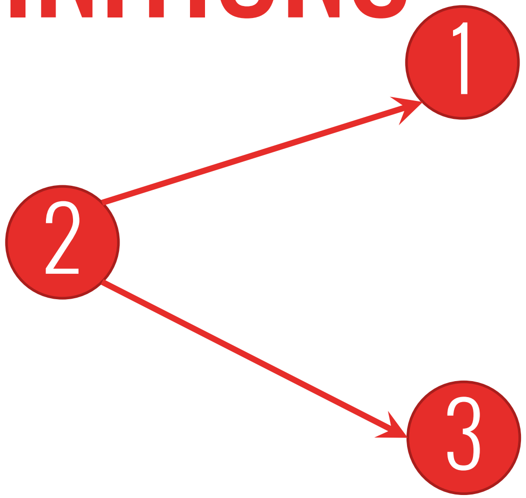
$A_{ij} = A_{ji}$ if the network is undirected,
or if i and j share a reciprocated edge

$$A = \begin{array}{cc} & \text{TARGET} \\ \text{SOURCE} & \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$



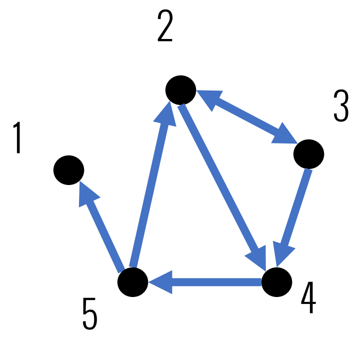
TOPOLOGY / DEFINITIONS

In-degree, out-degree



	1	2	3
In-degree	1	0	1
Out-degree	0	2	0

NODE DEGREE FROM MATRICES



Outdegree = $\sum_{j=1}^{n} A_{ij}$

example: **outdegree** for node 3 is 2, which we obtain by summing the number of non-zero entries in the 3rd row

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$\sum_{j=1}^{n} A_{3j}$

Indegree = $\sum_{i=1}^{n} A_{ij}$

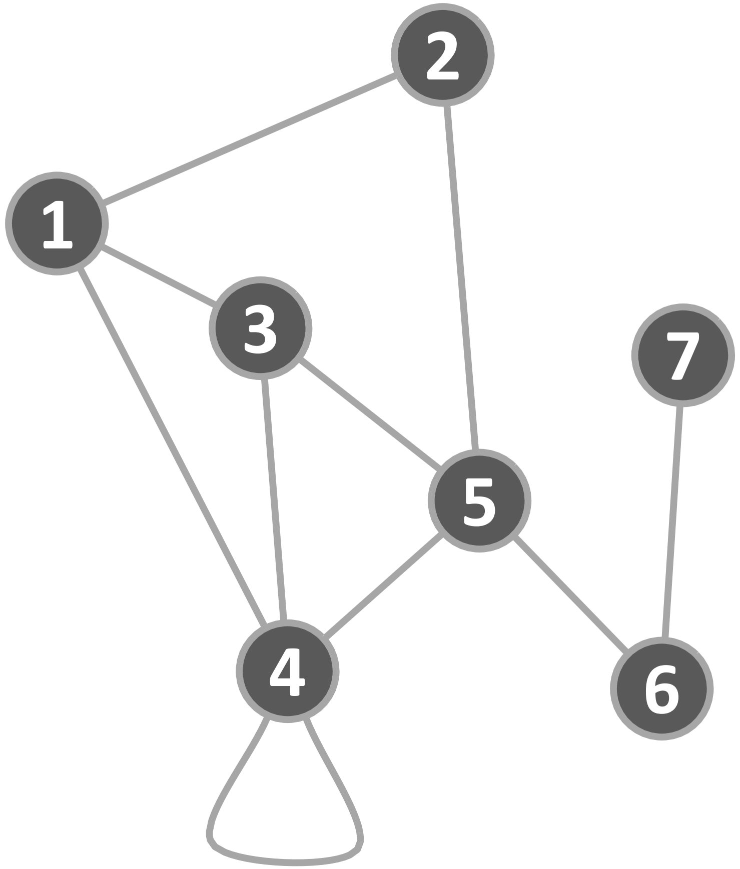
example: the **indegree** for node 3 is 1, which we obtain by summing the number of non-zero entries in the 3rd column

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$\sum_{i=1}^{n} A_{i3}$

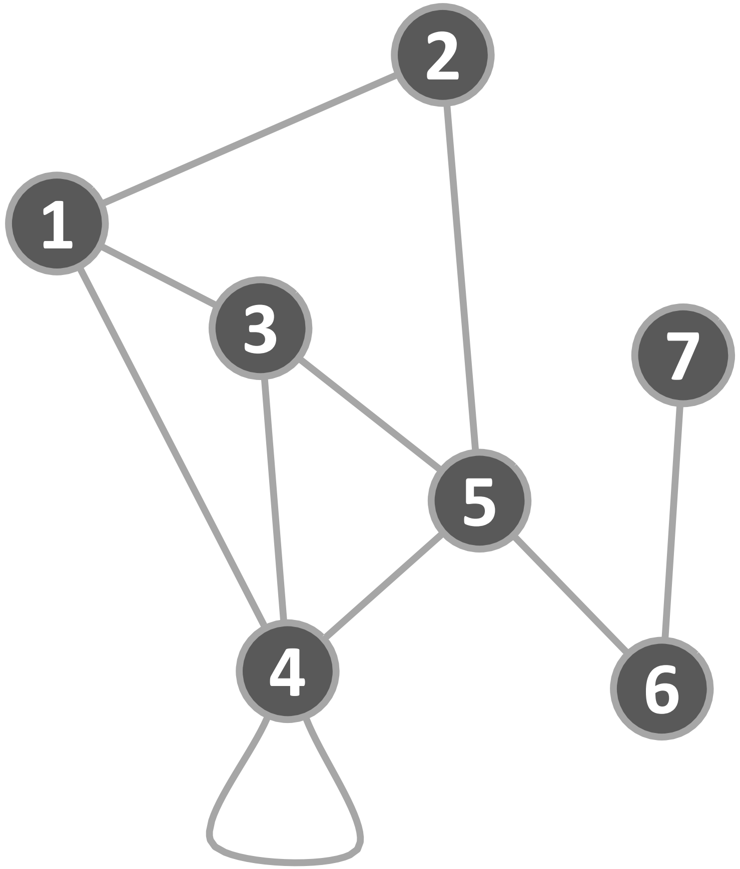
UNDIRECTED GRAPHS
THE MATRIX IS SYMMETRIC

	1	2	3	4	5	6	7
1	0	1	1	1	0	0	0
2	1	0	0	0	1	0	0
3	1	0	0	1	1	0	0
4	1	0	1	1	1	0	0
5	0	1	1	1	0	1	0
6	0	0	0	0	1	0	1
7	0	0	0	0	0	1	0



UNDIRECTED GRAPHS
THE MATRIX IS SYMMETRIC

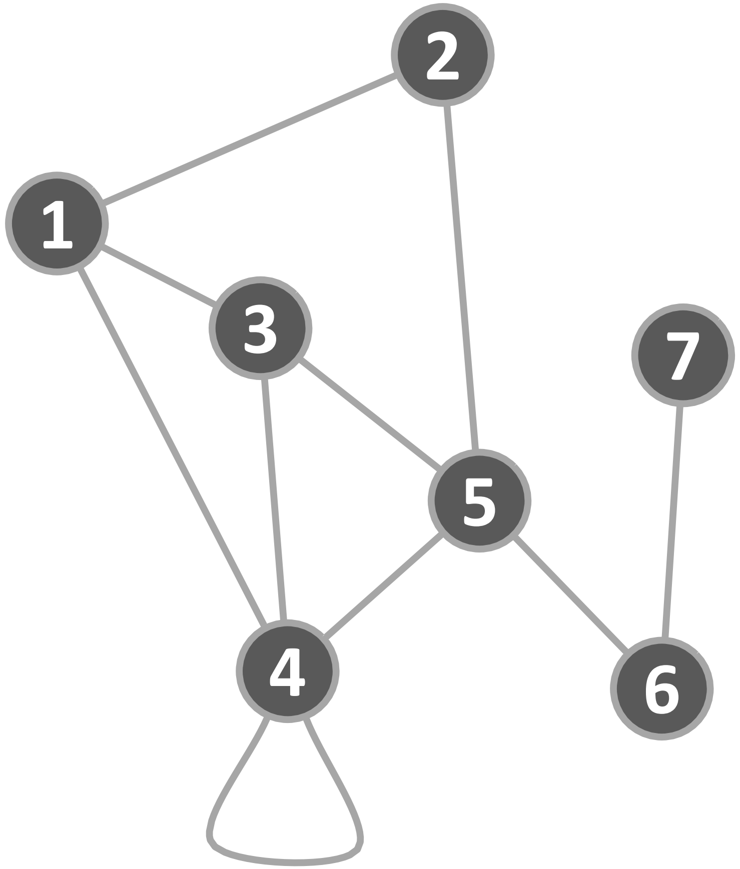
	1	2	3	4	5	6	7
1	0	1	1	1	0	0	0
2	1	0	0	0	1	0	0
3	1	0	0	1	1	0	0
4	1	0	1	1	1	0	0
5	0	1	1	1	0	1	0
6	0	0	0	0	1	0	1
7	0	0	0	0	0	1	0



WEIGHTED EDGES

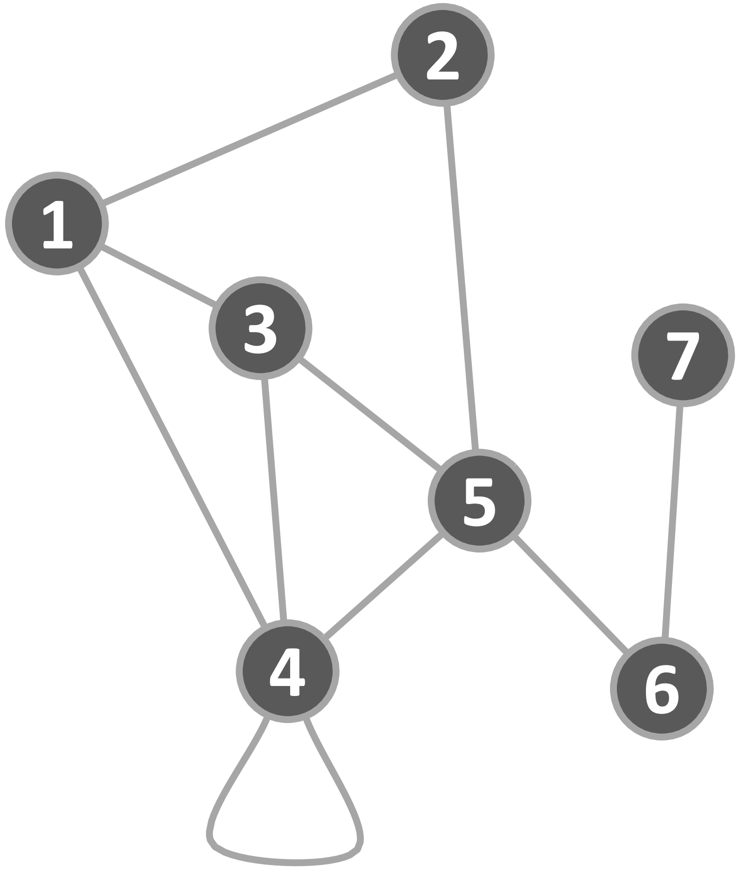
What happens to the node-link graph?

	1	2	3	4	5	6	7
1	0	1	1	1	0	0	0
2	1	0	0	0	2	0	0
3	1	0	0	4	1	0	0
4	1	0	4	1	1	0	0
5	0	2	1	1	0	1	0
6	0	0	0	0	1	0	1
7	0	0	0	0	0	1	0



WEIGHTED EDGES

	1	2	3	4	5	6	7
1	0	1	1	1	0	0	0
2	1	0	0	0	2	0	0
3	1	0	0	4	1	0	0
4	1	0	4	1	1	0	0
5	0	2	1	1	0	1	0
6	0	0	0	0	1	0	1
7	0	0	0	0	0	1	0



WHY ANALYZE NETWORKS?

LOTS OF NATURAL AND CONSTRUCTED PHENOMENA CAN BE DESCRIBED AS NETWORKS

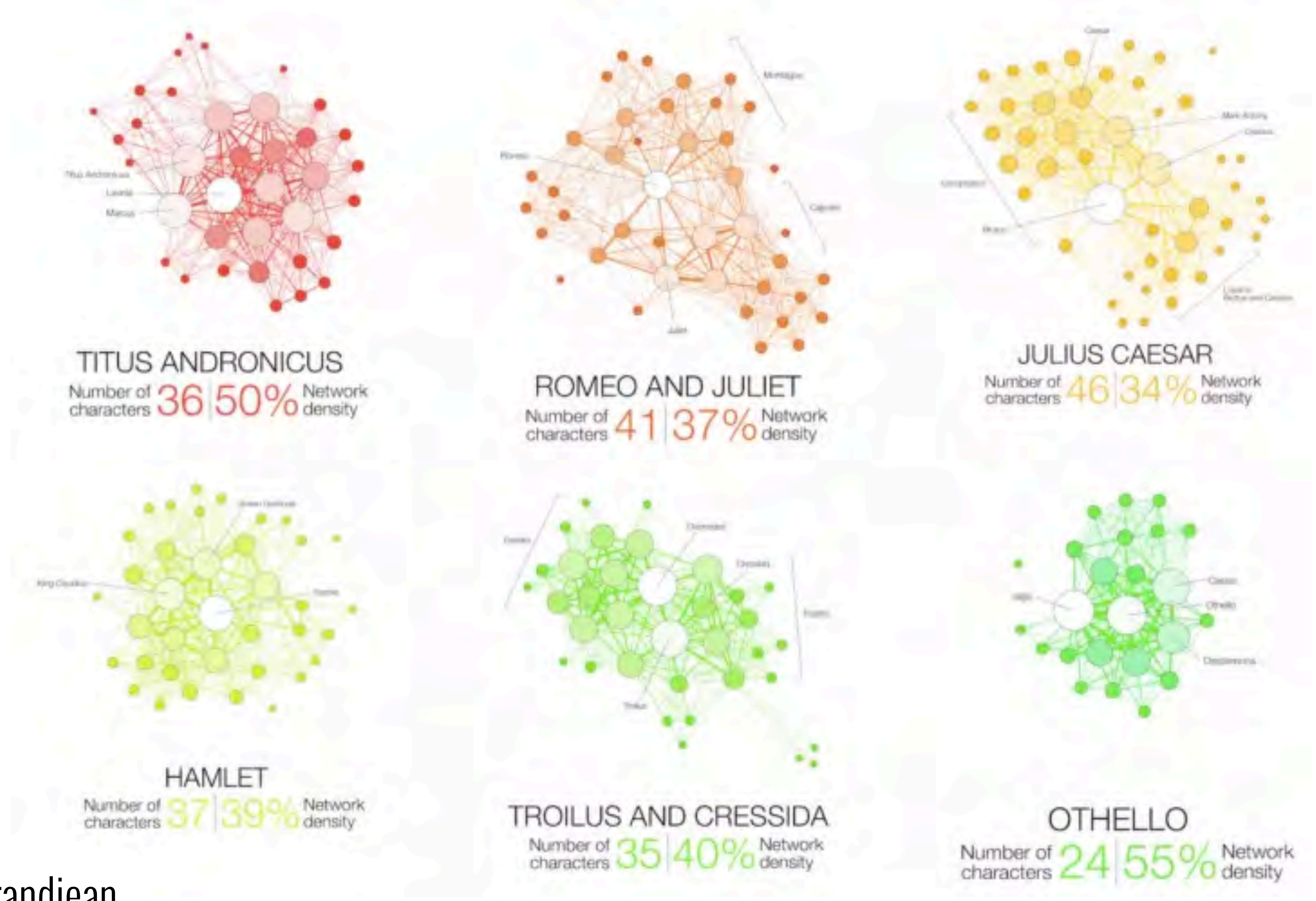
Relationships (Social, Trade, Bureaucratic, Citation, etc.)

Infrastructure (Roads, electricity, data, etc.)

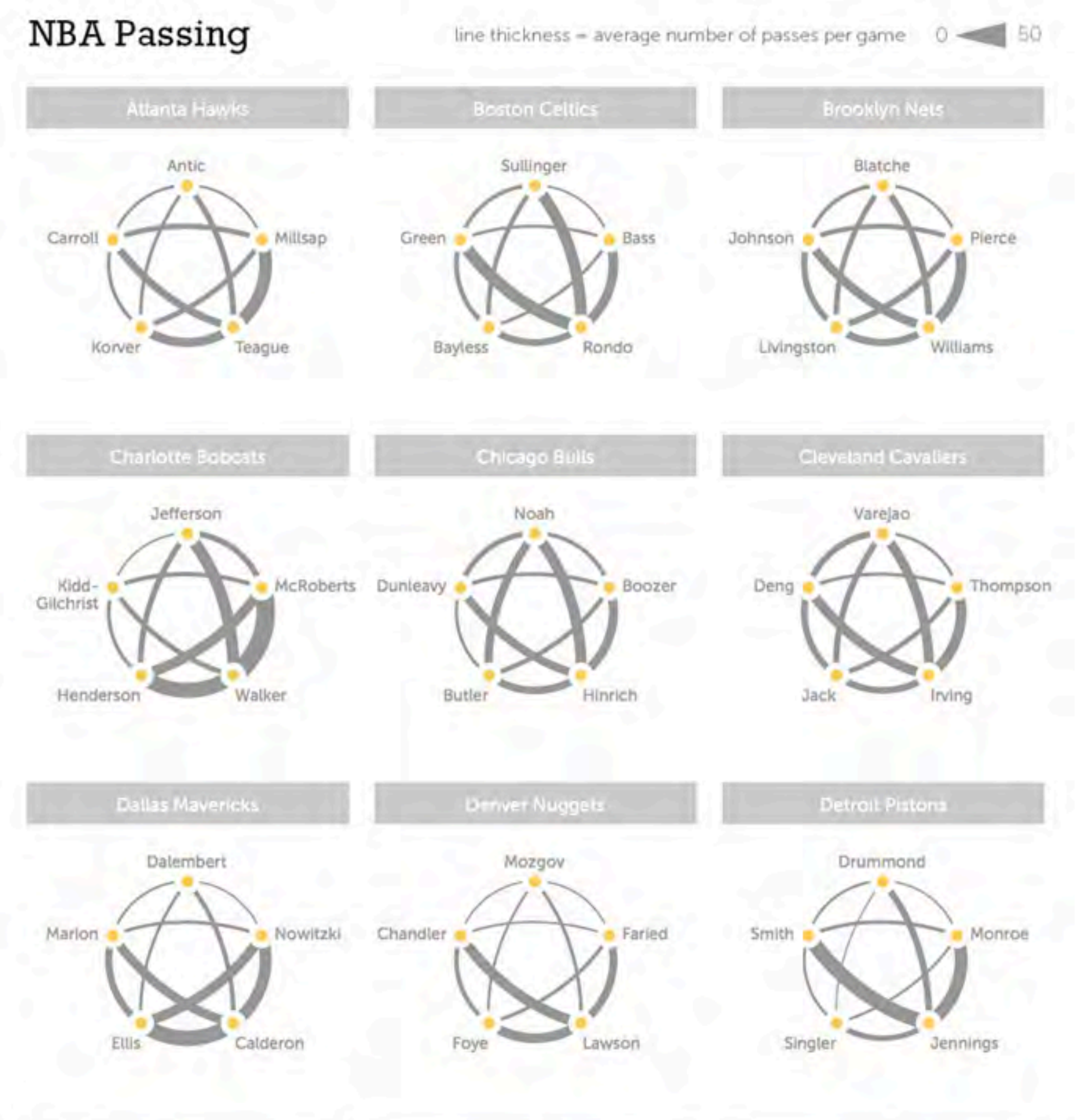
Natural Processes (Metabolic processes, food web, etc.)

Concepts (Language, semantics, etc.)

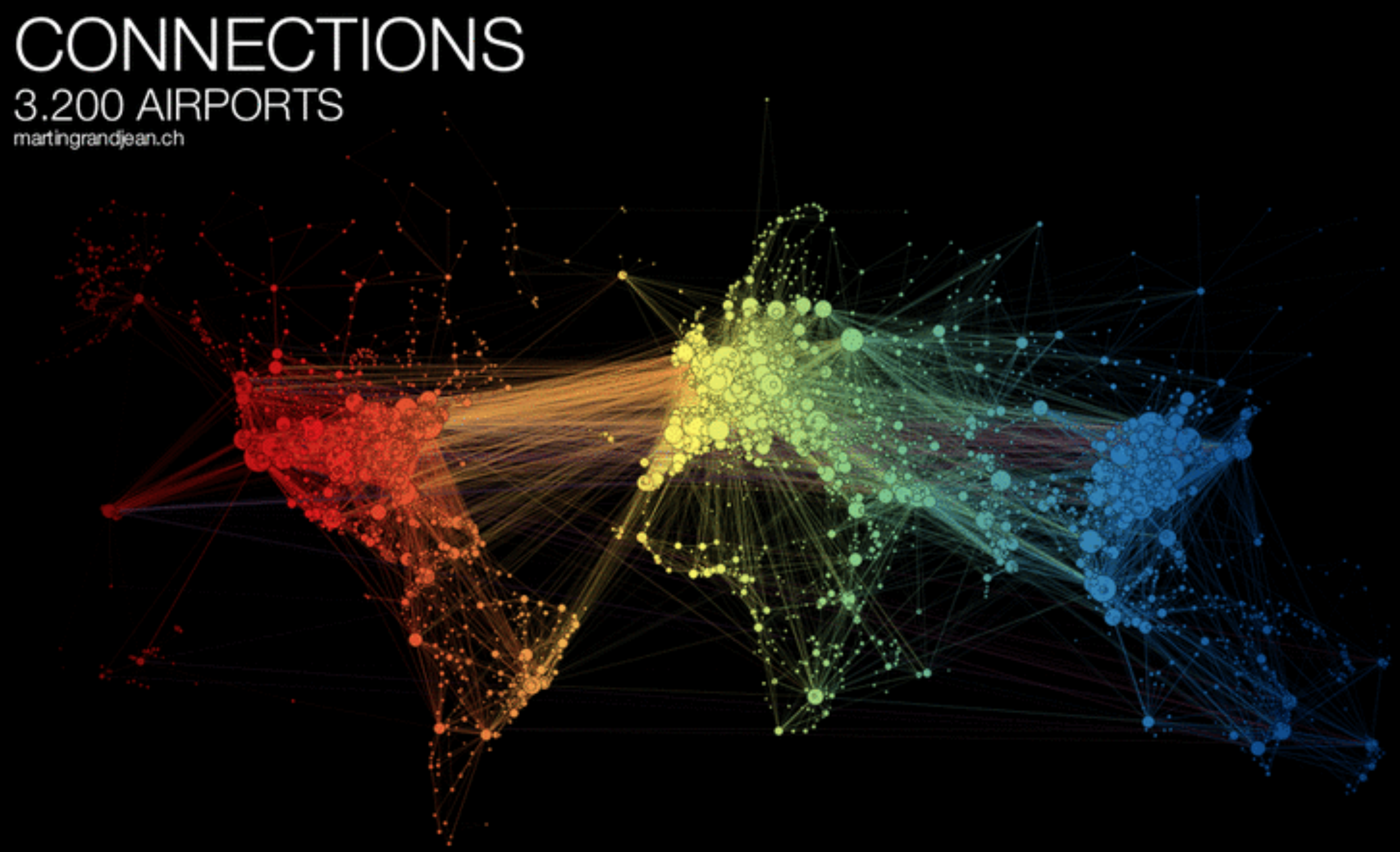
…



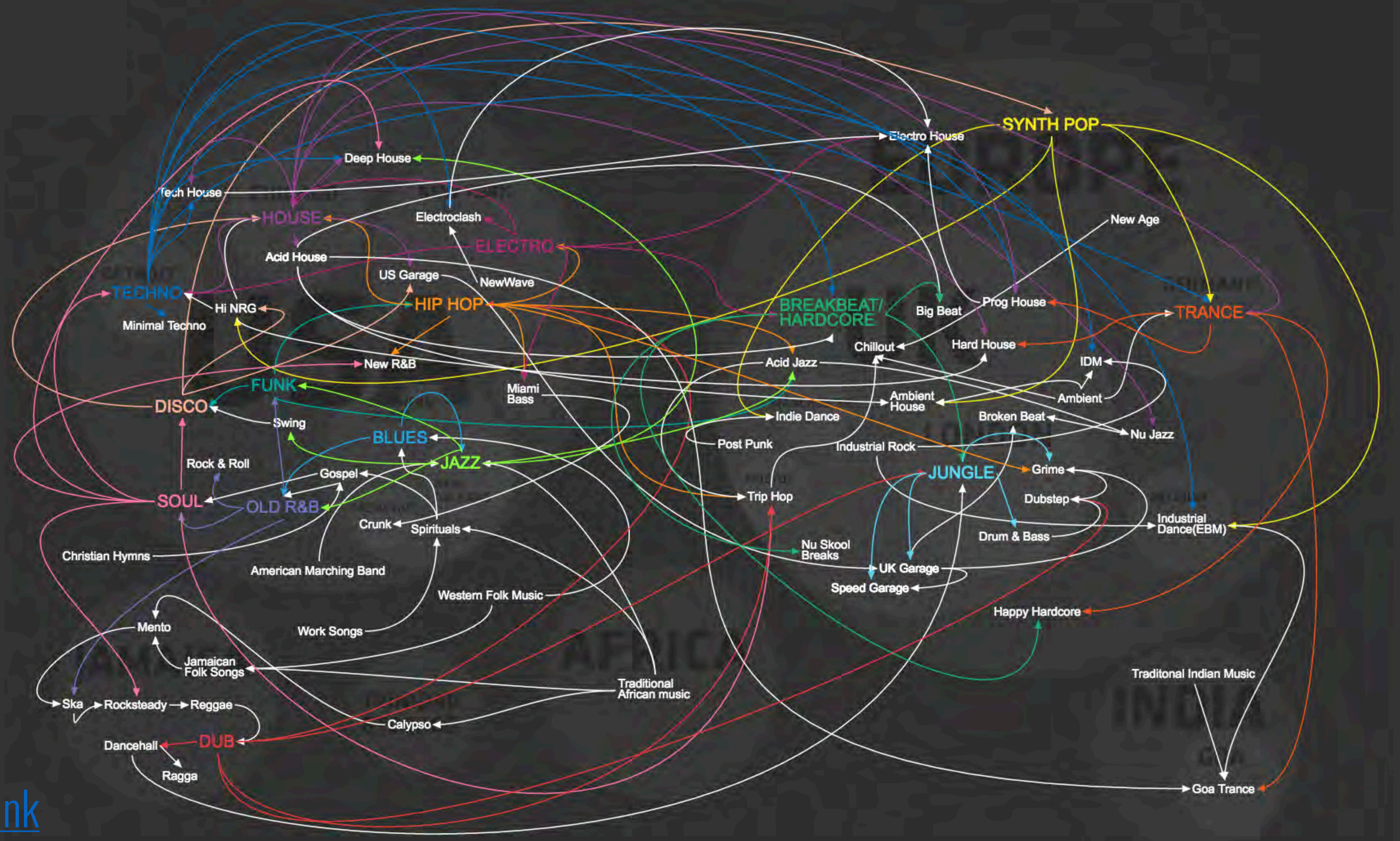
Martin Grandjean



Andrew Bergmann



[Martin Grandjean]


SYNTH POP
Electro House
Deep House
Tech House
HOUSE
Electroclash
ELECTRO
Acid House
US Garage
NewWave
TECHNO
Hi NRG
HIP HOP
Minimal Techno
New R&B
FUNK
DISCO
Miami Bass
Swing
BLUES
JAZZ
Rock & Roll
Gospel
SOUL
OLD R&B
Crunk
Spirituals
Christian Hymns
American Marching Band
Western Folk Music
Work Songs
Mento
Jamaican Folk Songs
Traditional African music
Ska
Rocksteady
Reggae
Calypso
Dancehall
DUB
Ragga
BREAKBEAT/ HARDCORE
Big Beat
Prog House
New Age
TRANCE
Chillout
Hard House
Acid Jazz
IDM
Ambient House
Ambient
Indie Dance
Broken Beat
Nu Jazz
Post Punk
Industrial Rock
JUNGLE
Grime
Trip Hop
Dubstep
Industrial Dance(EBM)
Drum & Bass
Nu Skool Breaks
UK Garage
Speed Garage
Happy Hardcore
Traditonal Indian Music
Goa Trance

Link

TASTE MAP

The Flavor Connection

Julia Child famously said that fat carries flavor, but perhaps instead we should give thanks to 4-methylpentanoic acid. Unique combinations of such chemical compounds give foods their characteristic flavors. Science-minded chefs have gone so far as to suggest that seemingly incongruous ingredients—chocolate and blue cheese, for example—will taste great together as long as they have enough flavor compounds in common. Scientists recently put this hypothesis to the test by creating a flavor map, a variant of which we have reproduced here. Lines connect foods that have components in common; thick lines mean many components are shared. By comparing the flavor network with various recipe databases, the researchers conclude that chefs do tend to pair ingredients with shared flavor compounds—but only in Western cuisine. Dishes from a database of recipes from East Asia tend to combine ingredients with few overlapping flavors.

How to Read This Graphic

Each blue dot is a food. Similar foods are grouped into 14 category columns (listed in alphabetical order).

Food

The size of a dot shows how popular the food is—the frequency with which it appears in a global 56,498-recipe database.

Sturgeon caviar, pelargonium and 14 others

Egg

Least prevalent (in 1 recipe)

Most prevalent (in 20,951 recipes)

A line connecting two dots means the two foods share at least one flavor-related chemical compound. The more flavor compounds they share, the thicker the line. Red lines connect foods in different categories.

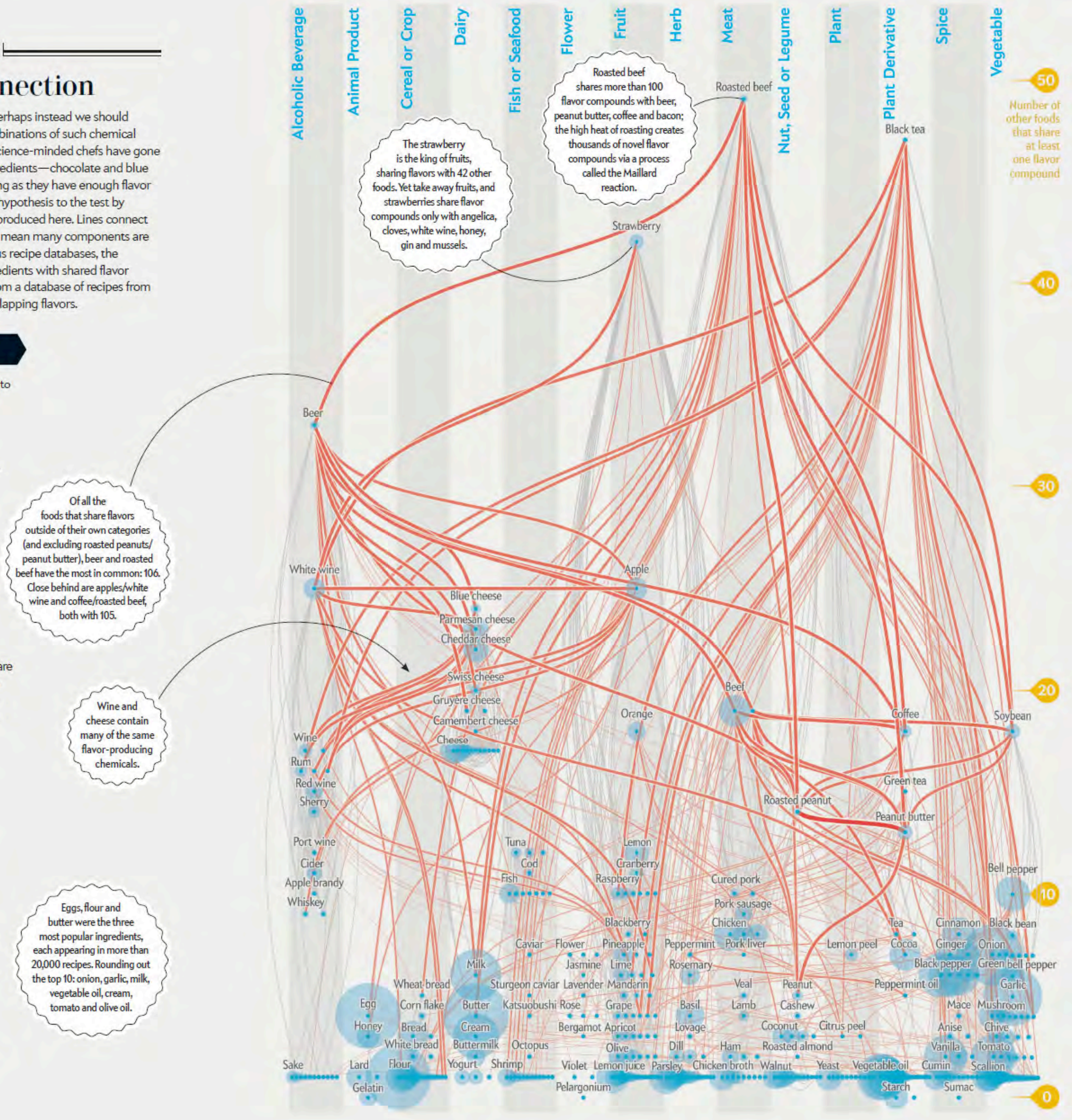
144 shared compounds

One shared compound

Gray lines connect foods in the same category.

A food's vertical position on the page reveals the total number of foods that connect to it. Foods at the top of the page share flavor compounds with many other foods. Foods at the bottom of the page are completely unique—they don't share flavors with any other foods.

Because of space constraints, only the most popular ingredient in a cluster of dots is labeled.



Jan Willem Tulp

ANSWERING QUESTIONS ABOUT THE GRAPH

Which nodes are the most **important/central**?
Which nodes are **close/related**?
Are there distinct **groups** or **communities**?
How far apart/how connected are nodes?
How much **flow** is there between nodes?
Which **paths** are important?
Are there particular **patterns** (“motifs”)?
...

VISUAL ANALYSIS TOOLS FOR NETWORKS

FAR FEWER TOOLS THAN FOR TABULAR DATA

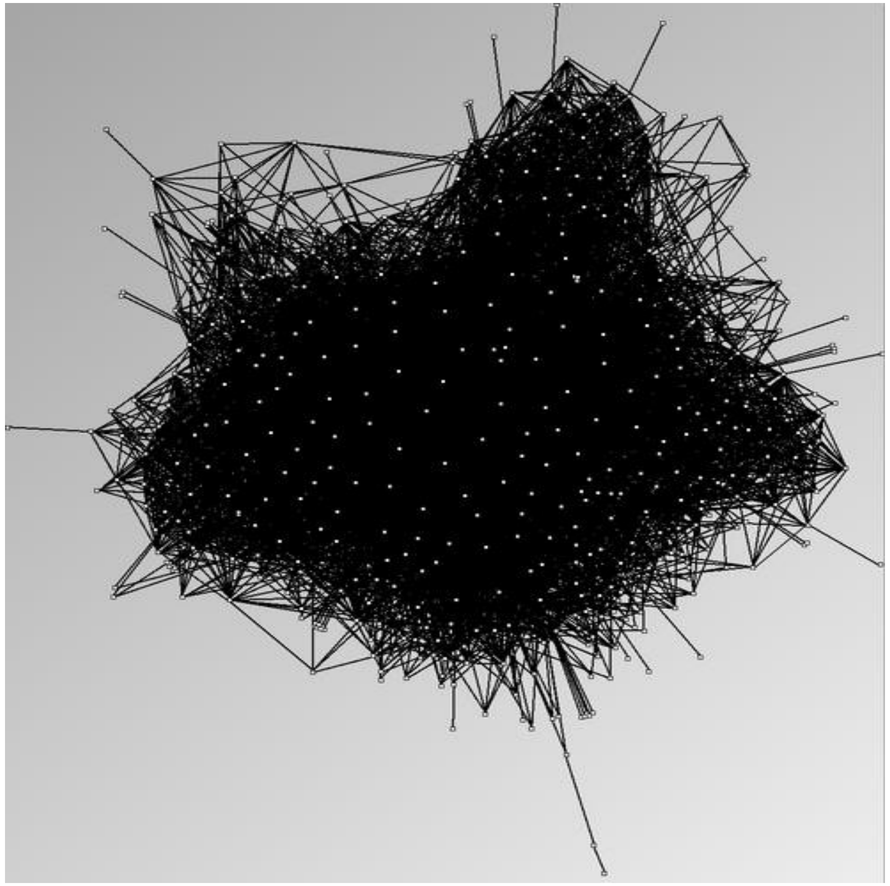
Graphs are **challenging to visualize**

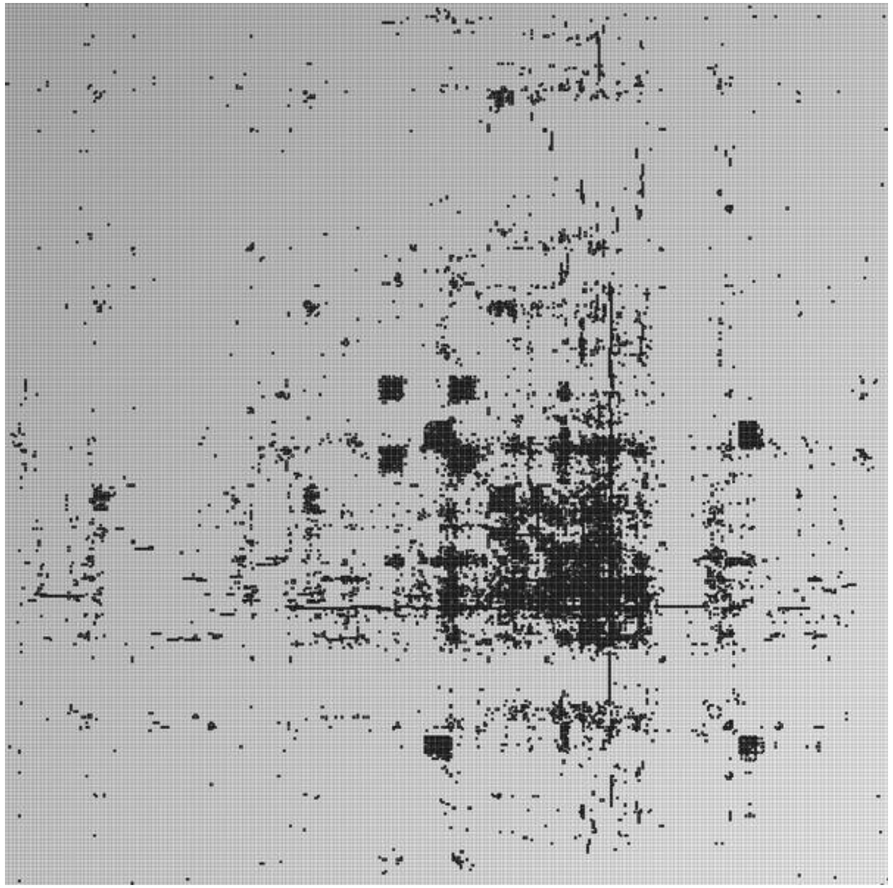
- Overplotting
- Edge crossings
- Ambiguous structures

Scaling up is difficult (both visually and technically)

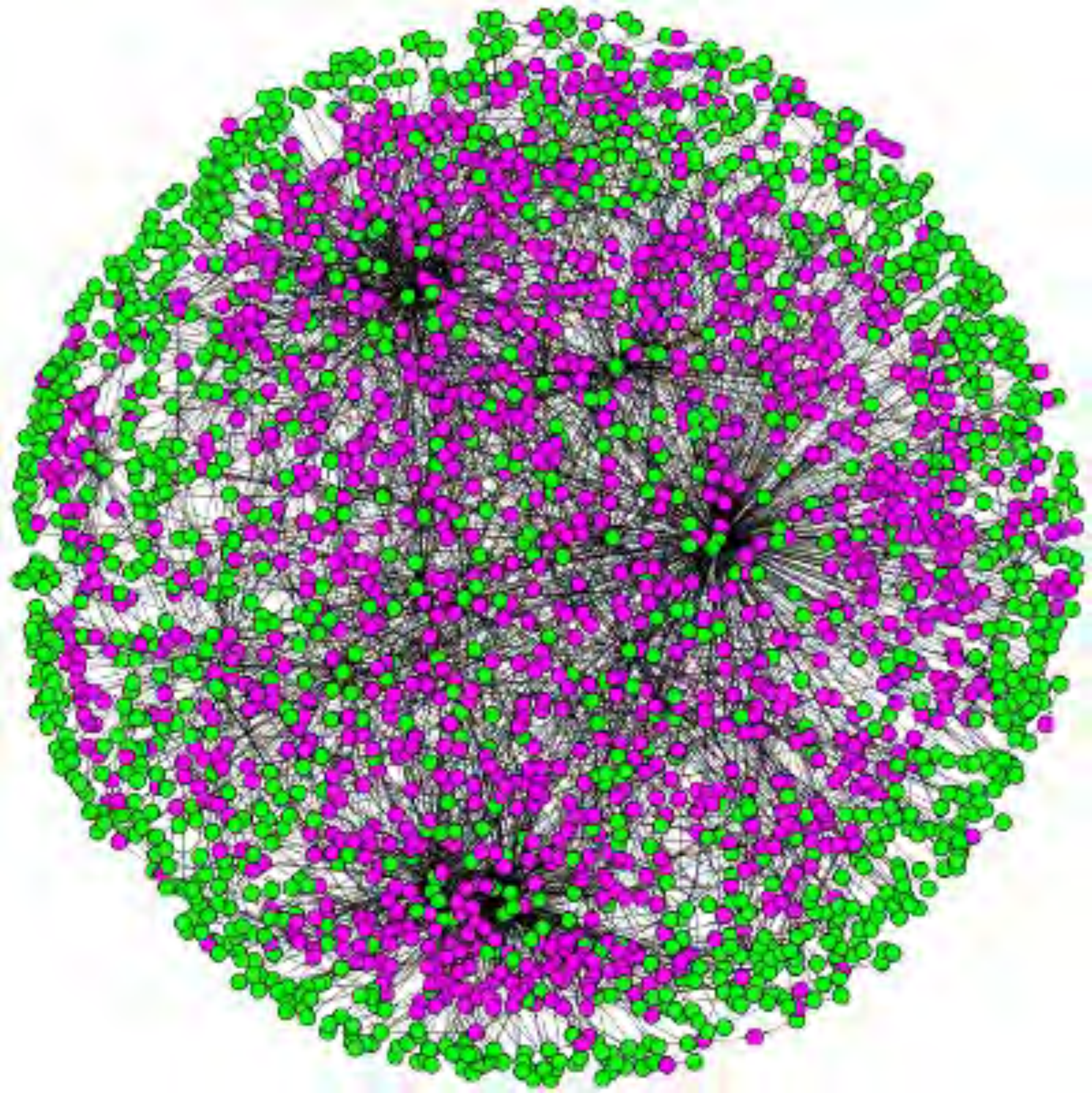
EVEN SMALL NETWORKS CAN BE COMPLICATED

One year of email between ~500 researchers





“HAIRBALLS”



http://eagereyes.org/techniques/graphs-hairball

GRAPH VISUALIZATION CHALLENGES

Graph layout

How do we render the data?

Navigation

How does a user move in the space

Scale

The larger the graph the harder layout and navigation are

SHNEIDERMAN'S CRITERIA

In a perfect world...

1. Every node is **visible**
2. For every node you can **count degree**
3. You can **follow every link** from source to destination
4. Clusters and outliers are **identifiable**

LAYOUT HEURISTICS

Planar

Grid-based

Orthogonal

Curved lines

Hierarchies

Circular

...

http://guess.wikispot.org/Laying_out_Graphs?action=Files&do=view&target=spring.jpg

LAYOUT HEURISTICS

Planar
Grid-based
Orthogonal
Curved lines
Hierarchies
Circular

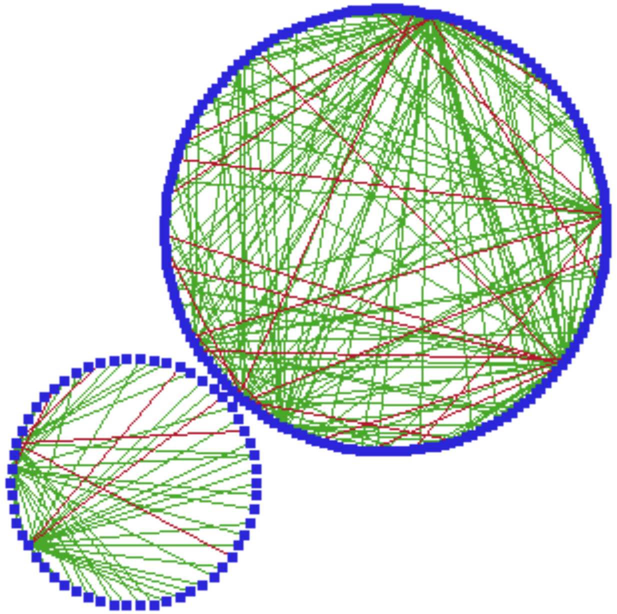
...

http://guess.wikispot.org/Laying_out_Graphs?action=Files&do=view&target=fr.jpg

LAYOUT HEURISTICS

Planar
Grid-based
Orthogonal
Curved lines
Hierarchies
Circular

...



http://support.sas.com/documentation/cdl/en/grnvwug/61307/HTML/default/p0q343kxjyj36jn1e2z6lulkda3j.htm

FORCE DIRECTED LAYOUTS

Treat edges like springs

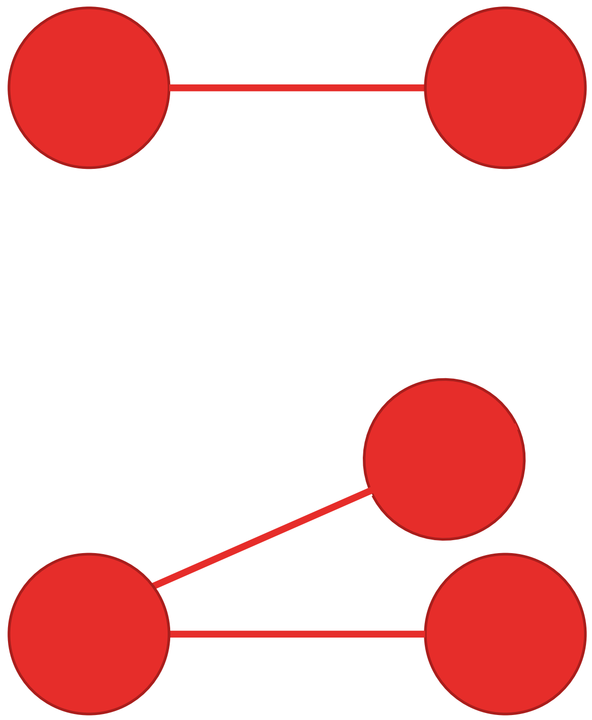
Pull their source and target together

Treat nodes like charged particles

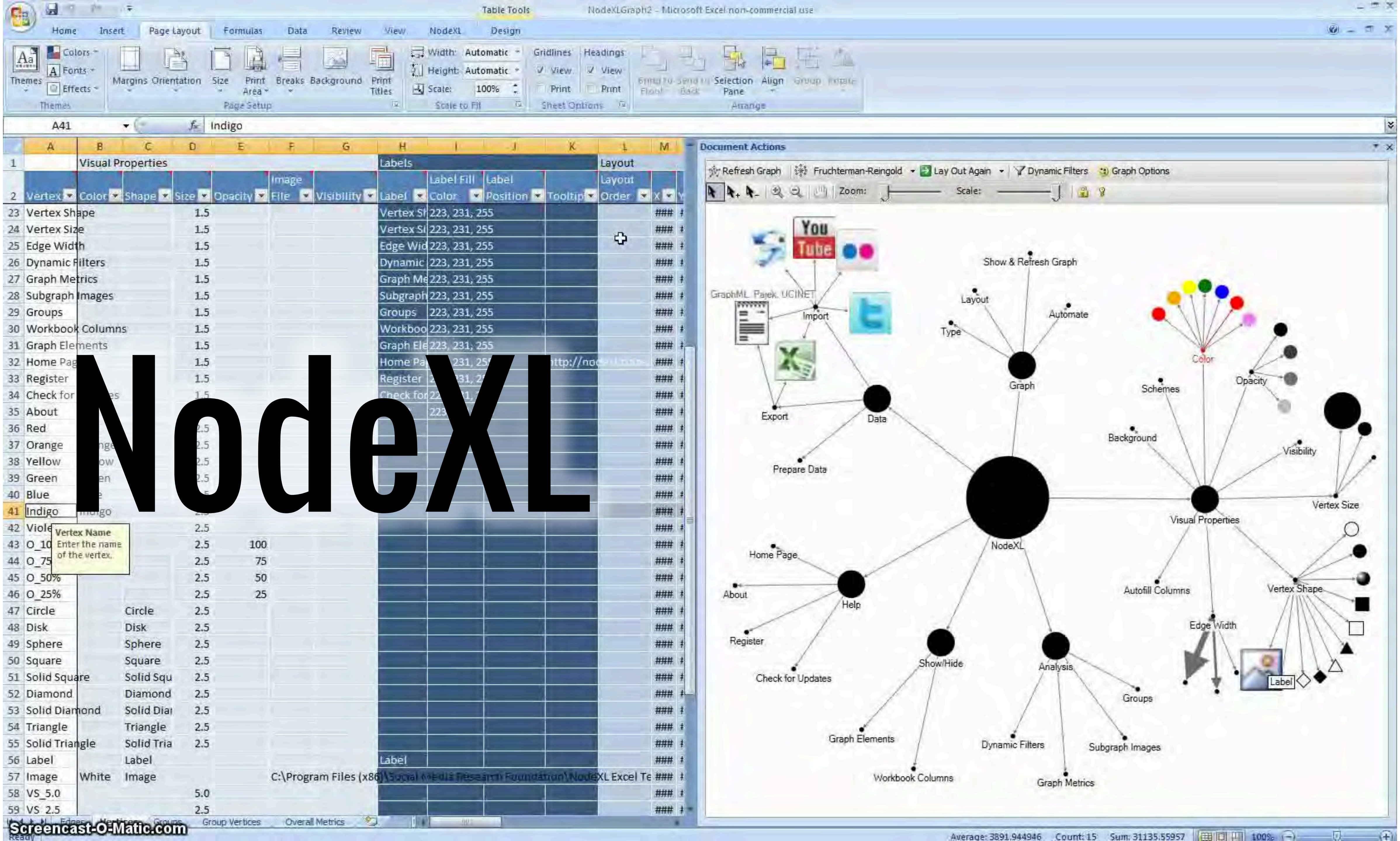
Push one another apart

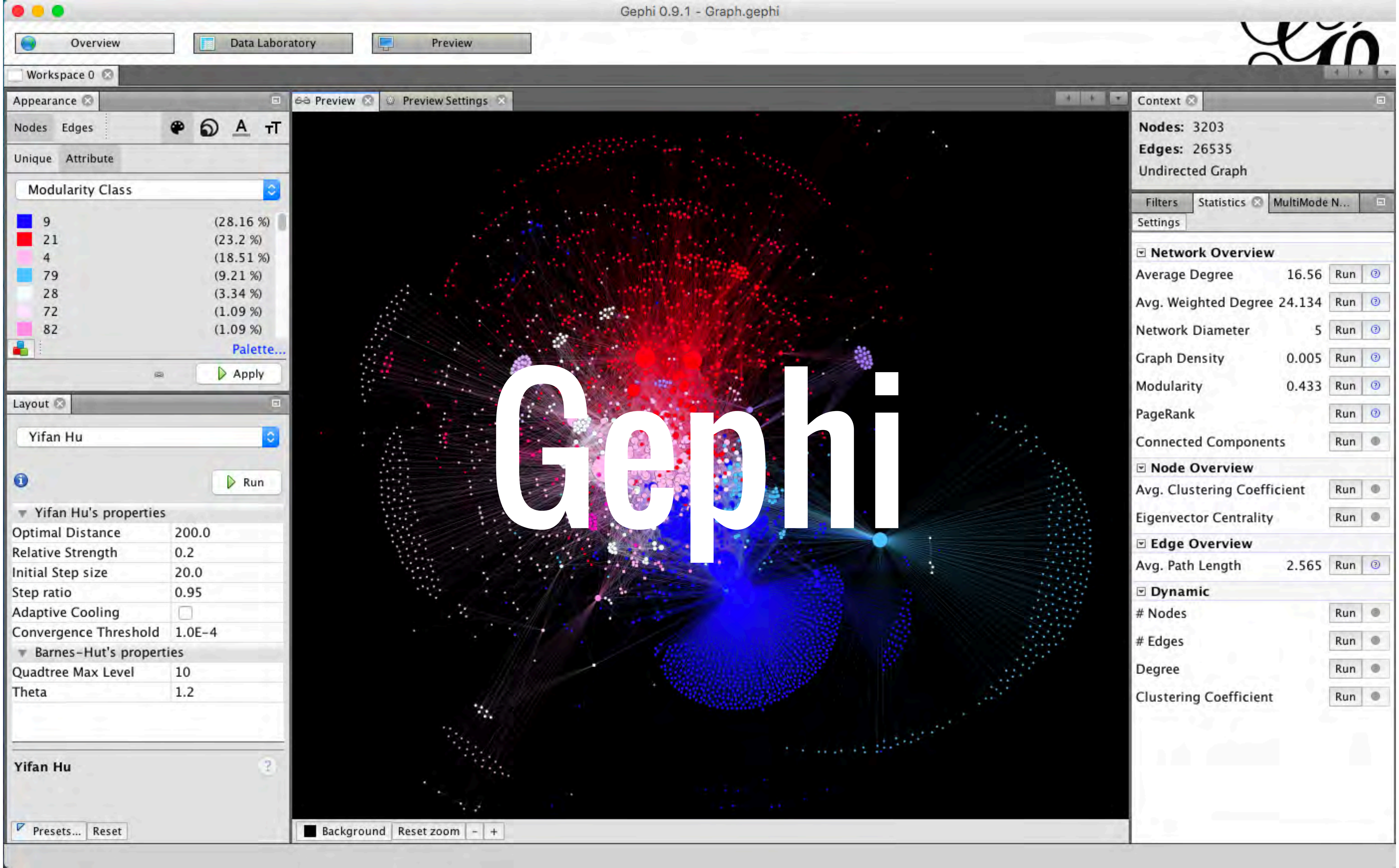
Repeatedly calculate in simulation

Different tensions/gravities
will cause different behaviors



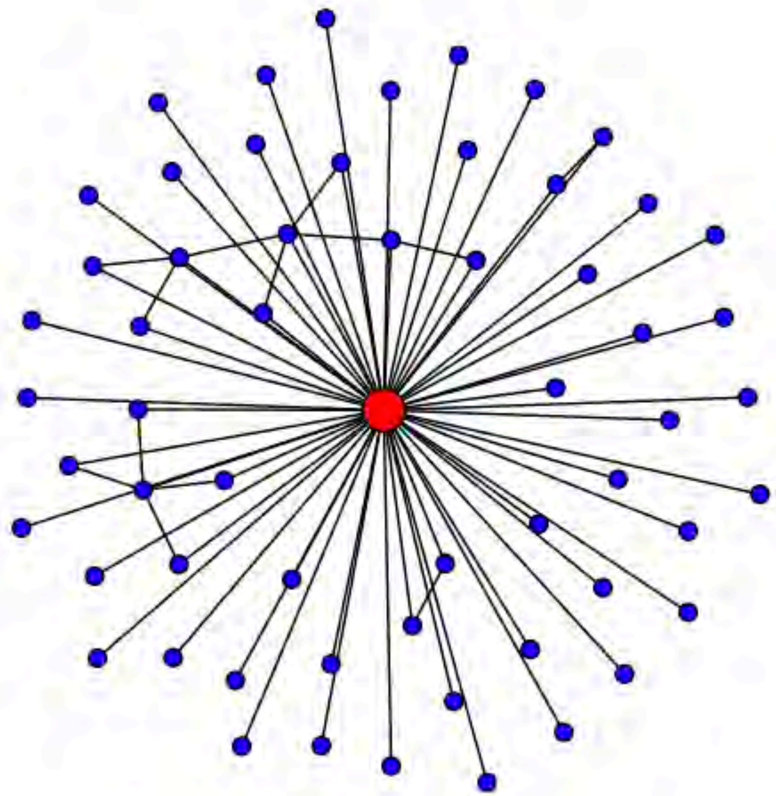
NodeXL


NodeXLGraph2 - Microsoft Excel non-commercial use
Visual Properties
Labels
Layout
Document Actions
Refresh Graph
Fruchterman-Reingold
Lay Out Again
Dynamic Filters
Graph Options
NodeXL
Data
Import
Export
Prepare Data
GraphML, Pajek, UCINET
Graph
Show & Refresh Graph
Layout
Automate
Type
Visual Properties
Color
Opacity
Schemes
Background
Visibility
Vertex Size
Vertex Shape
Autofill Columns
Edge Width
Label
Help
Home Page
About
Register
Check for Updates
Show/Hide
Graph Elements
Workbook Columns
Analysis
Dynamic Filters
Graph Metrics
Subgraph Images
Groups
Screencast-O-Matic.com


Gephi 0.9.1 - Graph.gephi
Overview
Data Laboratory
Preview
Workspace 0
Appearance
Nodes
Edges
Unique
Attribute
Modularity Class
9 (28.16 %)
21 (23.2 %)
4 (18.51 %)
79 (9.21 %)
28 (3.34 %)
72 (1.09 %)
82 (1.09 %)
Palette...
Apply
Layout
Yifan Hu
Run
Yifan Hu's properties
Optimal Distance 200.0
Relative Strength 0.2
Initial Step size 20.0
Step ratio 0.95
Adaptive Cooling
Convergence Threshold 1.0E-4
Barnes-Hut's properties
Quadtree Max Level 10
Theta 1.2
Yifan Hu
Presets...
Reset
Preview
Preview Settings
Gephi
Background
Reset zoom
Context
Nodes: 3203
Edges: 26535
Undirected Graph
Filters
Statistics
MultiMode N...
Settings
Network Overview
Average Degree 16.56 Run
Avg. Weighted Degree 24.134 Run
Network Diameter 5 Run
Graph Density 0.005 Run
Modularity 0.433 Run
PageRank Run
Connected Components Run
Node Overview
Avg. Clustering Coefficient Run
Eigenvector Centrality Run
Edge Overview
Avg. Path Length 2.565 Run
Dynamic
Nodes Run
Edges Run
Degree Run
Clustering Coefficient Run

NetworkX

“A Python library for the **creation**, **manipulation**, and **study** of the **structure**, **dynamics**, and **functions** of complex networks”



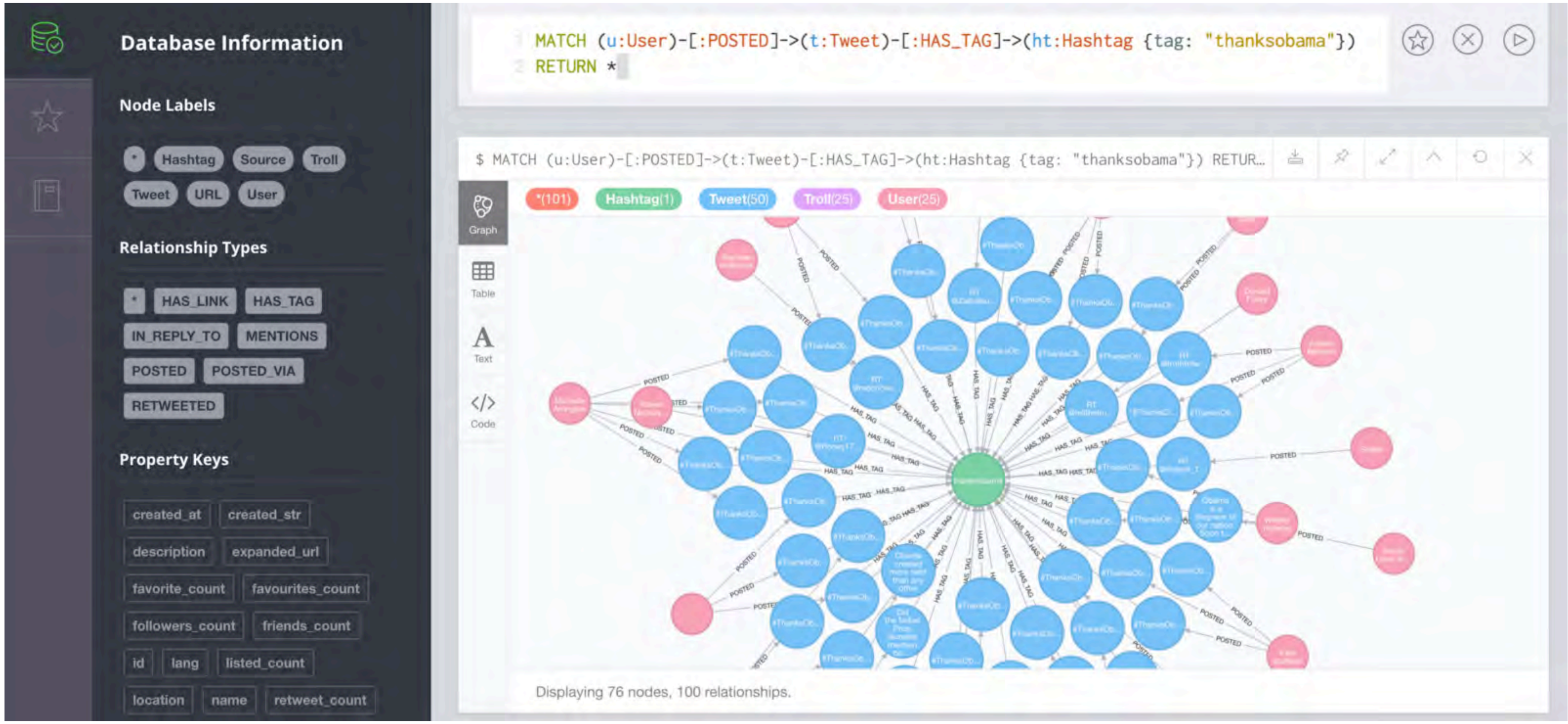
```
from operator import itemgetter
import networkx as nx
import matplotlib.pyplot as plt

if __name__ == '__main__':
    # Create a BA model graph
    n=1000
    m=2
    G=nx.generators.barabasi_albert_graph(n,m)
    # find node with largest degree
    node_and_degree=G.degree()
    (largest_hub,degree)=sorted(node_and_degree.items(),key=itemgetter(1))[-1]
    # Create ego graph of main hub
    hub_ego=nx.ego_graph(G,largest_hub)
    # Draw graph
    pos=nx.spring_layout(hub_ego)
    nx.draw(hub_ego,pos,node_color='b',node_size=50,with_labels=False)
    # Draw ego as large and red
    nx.draw_networkx_nodes(hub_ego,pos,nodelist=[largest_hub],node_size=300,node_color='r')
    plt.savefig('ego_graph.png')
    plt.show()
```

GRAPH DATABASES

Neo4j etc.

Specialized software stacks for graph analytics.

Lots of tools for creating, querying, cleaning, and displaying graphs.



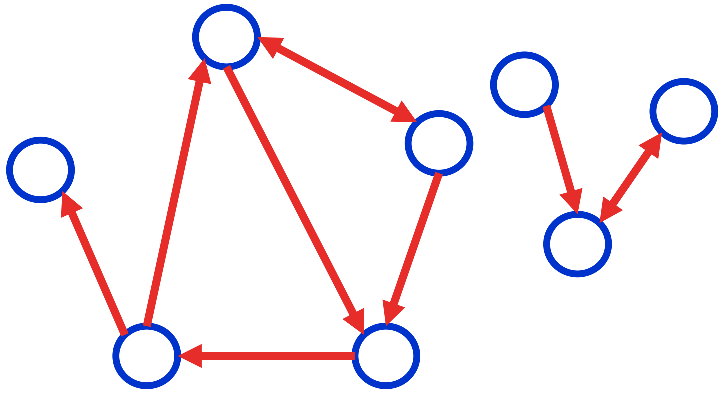
NETWORK ANALYSIS METHODS

BASIC NETWORK METRICS

DEGREE SEQUENCE AND DEGREE DISTRIBUTION

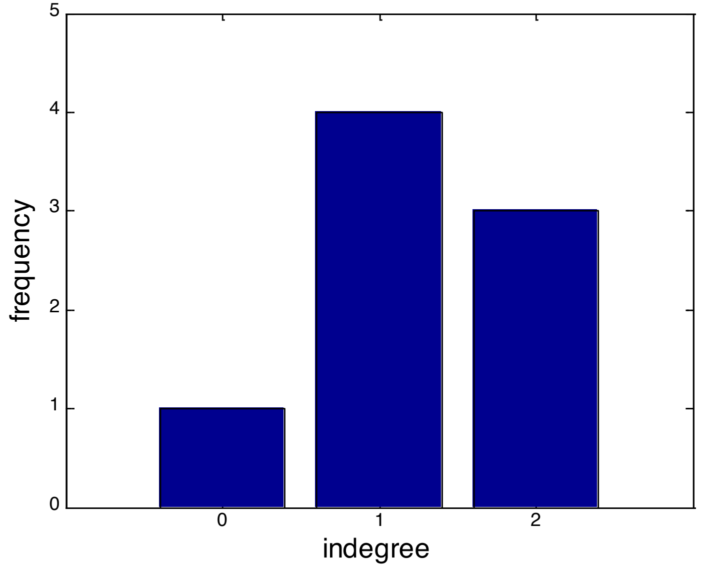
Degree sequence: An ordered list of the (in,out) degree of each node

- In-degree sequence:
  - [2, 2, 2, 1, 1, 1, 1, 0]
- Out-degree sequence:
  - [2, 2, 2, 2, 1, 1, 1, 0]
- (undirected) degree sequence:
  - [3, 3, 3, 2, 2, 1, 1, 1]



Degree distribution: Frequency count for each degree

- In-degree distribution:
  - [(2,3) (1,4) (0,1)]
- Out-degree distribution:
  - [(2,4) (1,3) (0,1)]
- (undirected) distribution:
  - [(3,3) (2,2) (1,3)]



CONNECTED COMPONENTS

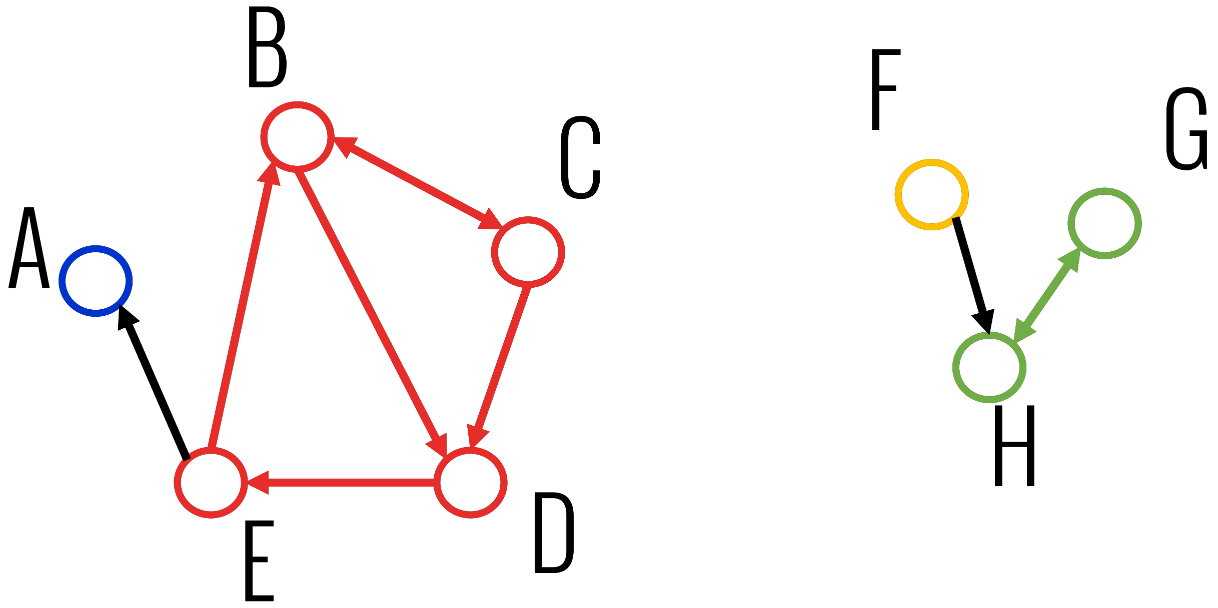
Is the graph actually connected?

Strongly connected components

Each node can be reached from every other node via directed links

How many here?

B C D E
A
G H
F



CONNECTED COMPONENTS

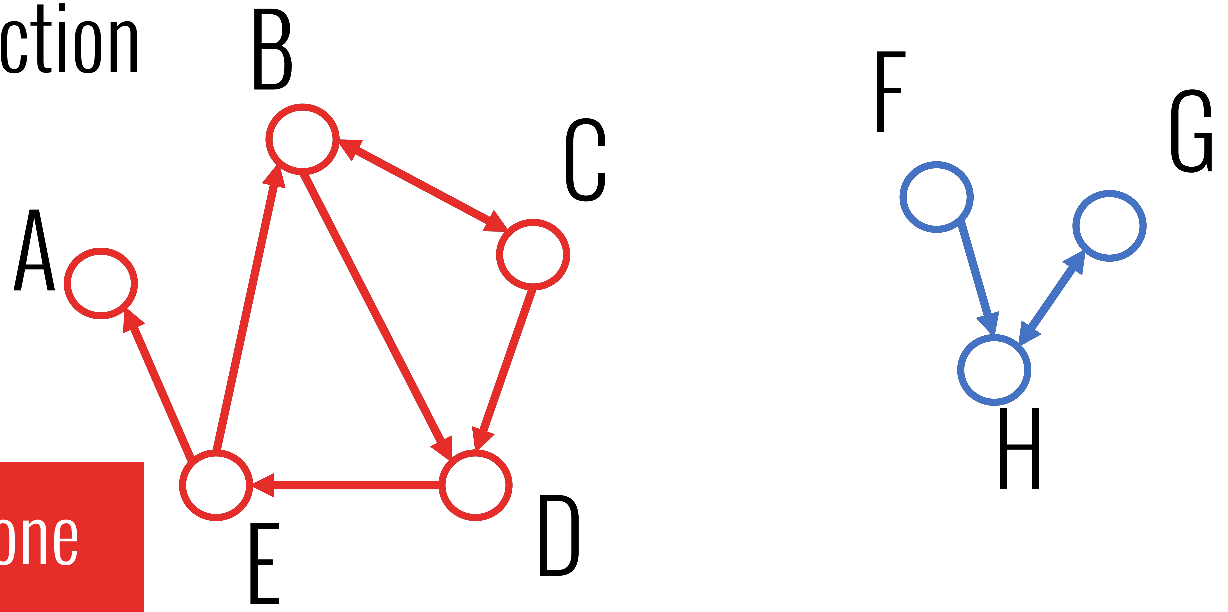
Weakly connected components

Every node can be reached from every other node by following links in either direction

How many here?

A B C D E
G H F

In **undirected** networks one talks simply about 'connected components'



SHORTEST PATHS

Shortest path (also called a geodesic path)

- The shortest sequence of links connecting two nodes
- Not always unique

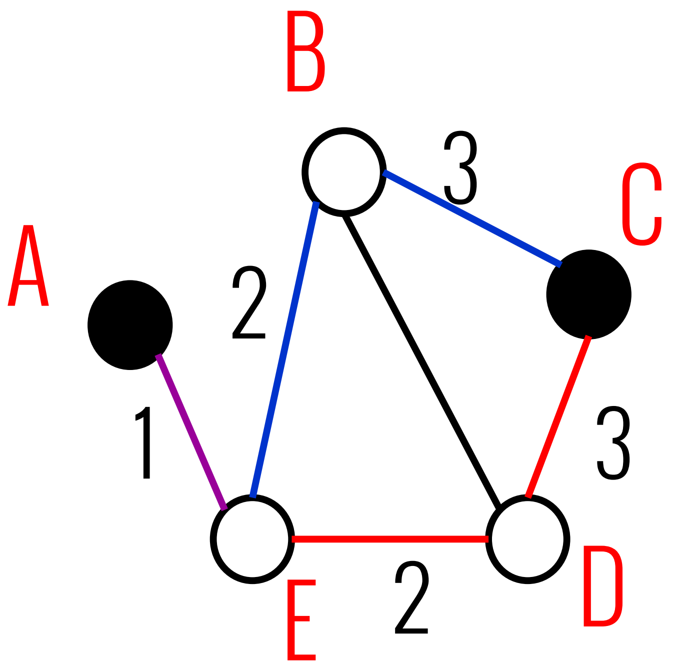
Shortest path from A-C?

A and C are connected by 2 shortest paths

- A – E – B - C
- A – E – D – C

Diameter: the largest geodesic distance in the graph

- Distance between A and C is the maximum for the graph: 3



Caution: some people use the term 'diameter' for the **average** shortest path distance. We will only use it to refer to the **max distance.**

GRAPH DENSITY

% of the edges that *could* exist which actually do

directed graph

$e_{max} = n*(n-1)$

each node can connect to (n-1) other nodes

undirected graph

$e_{max} = n*(n-1)/2$

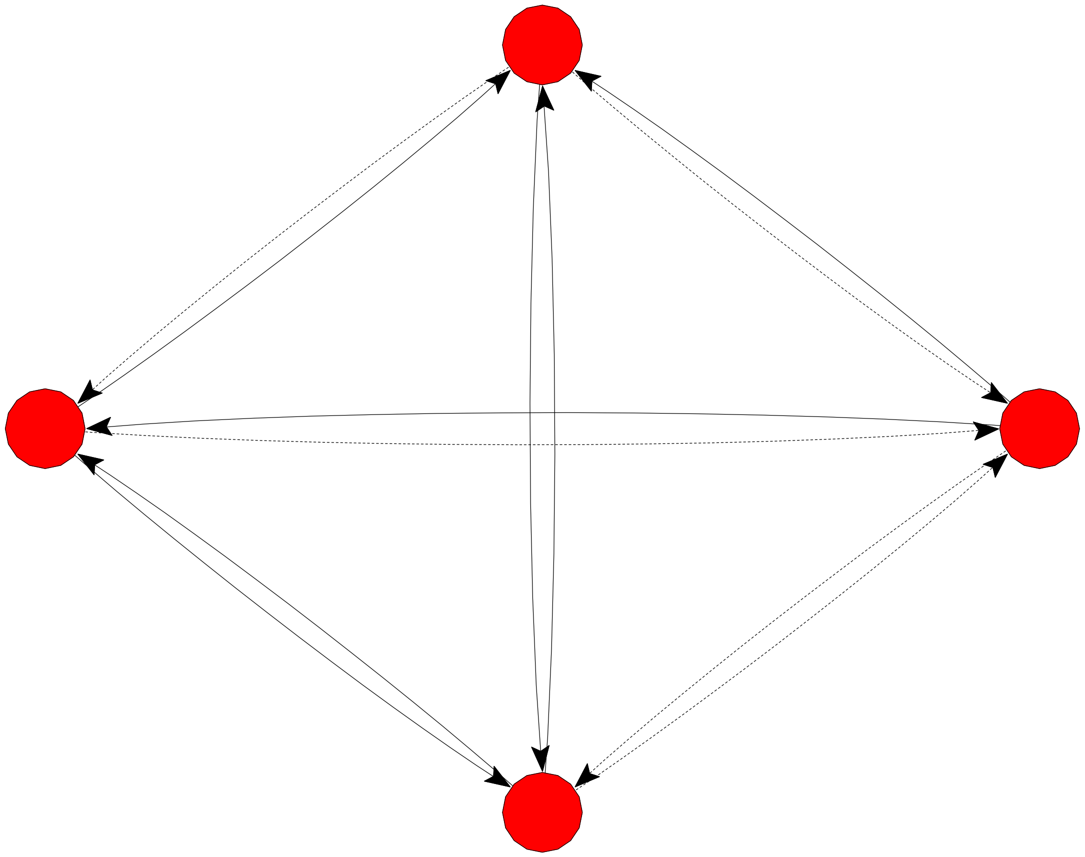
half as many

What fraction are present?

density = e / e_{max}

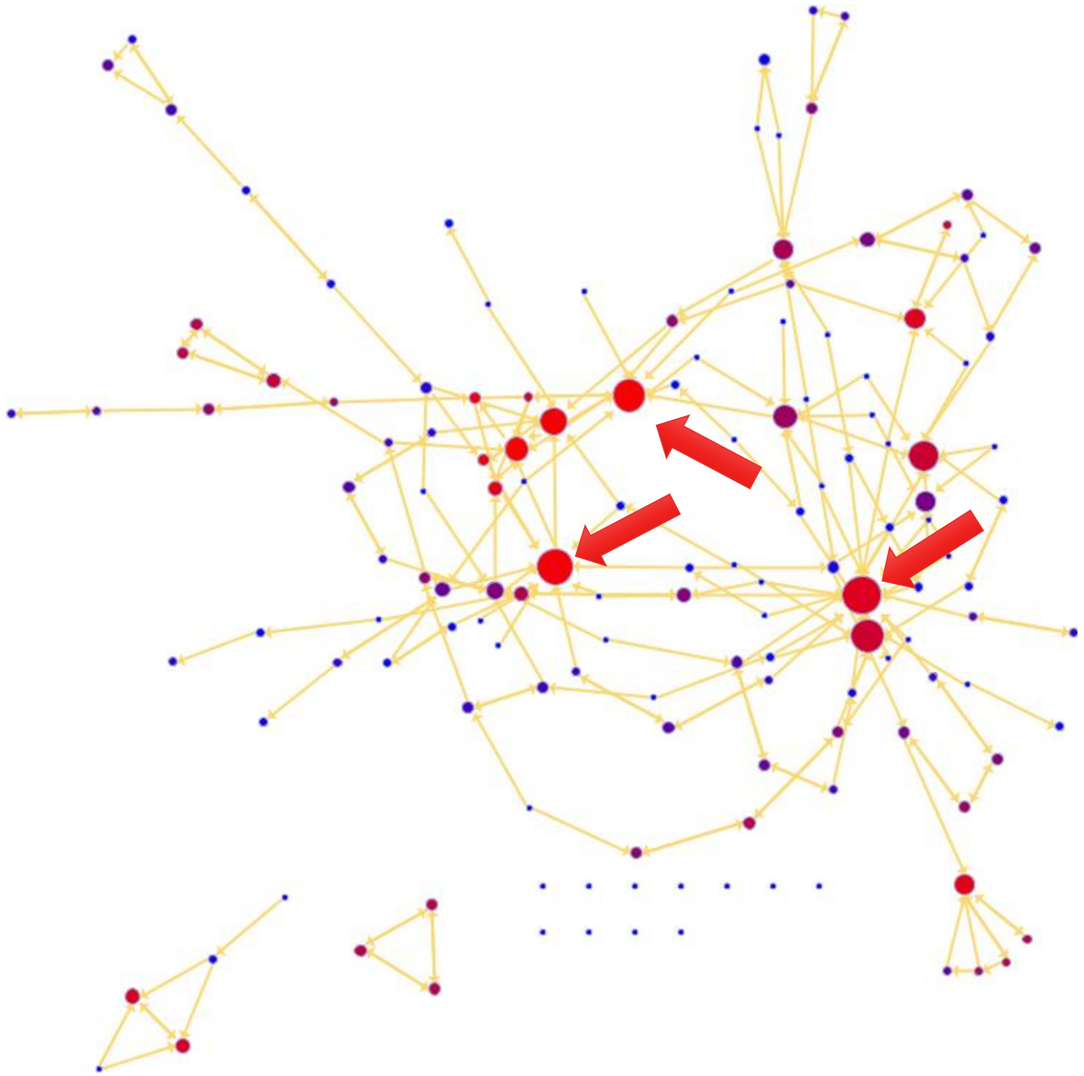
This graph has 7 out of 12 possible

7/12 = **0.583**



CENTRALITY MEASURES

WHO IS MOST IMPORTANT?



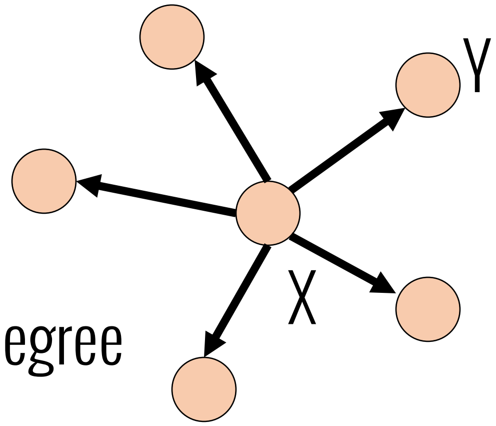
Who is...

... central?

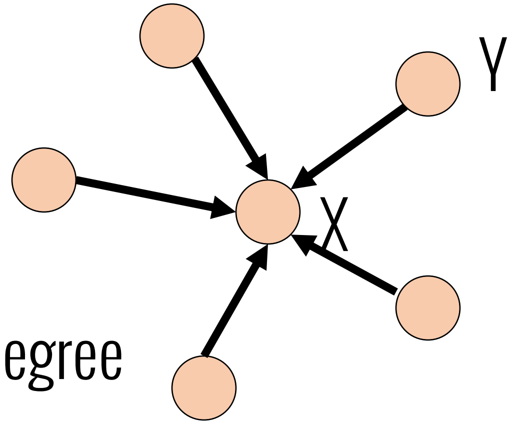
... important?

... prestigious?

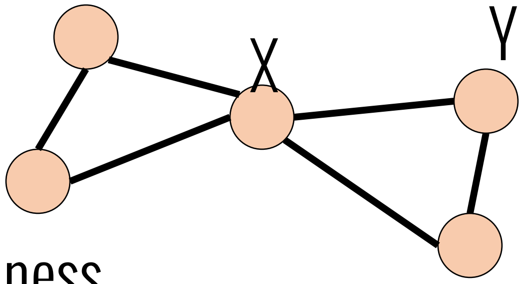
CENTRALITY



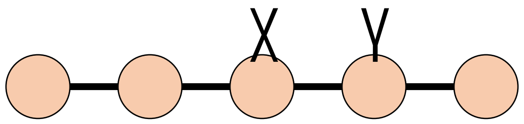
outdegree



indegree

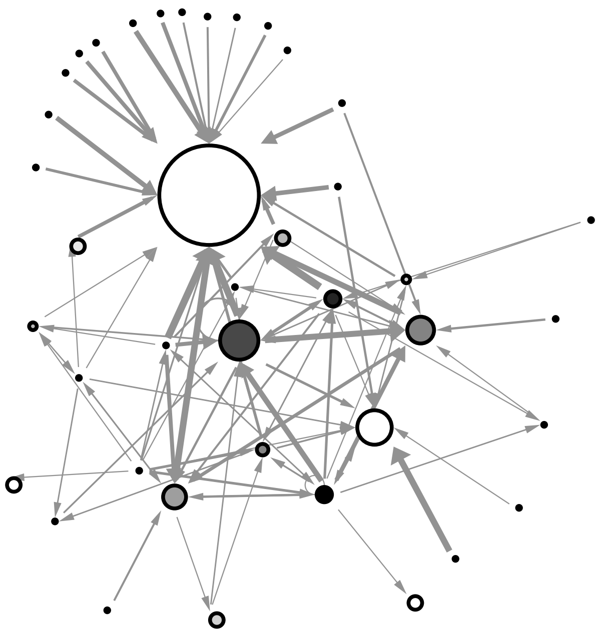


betweenness

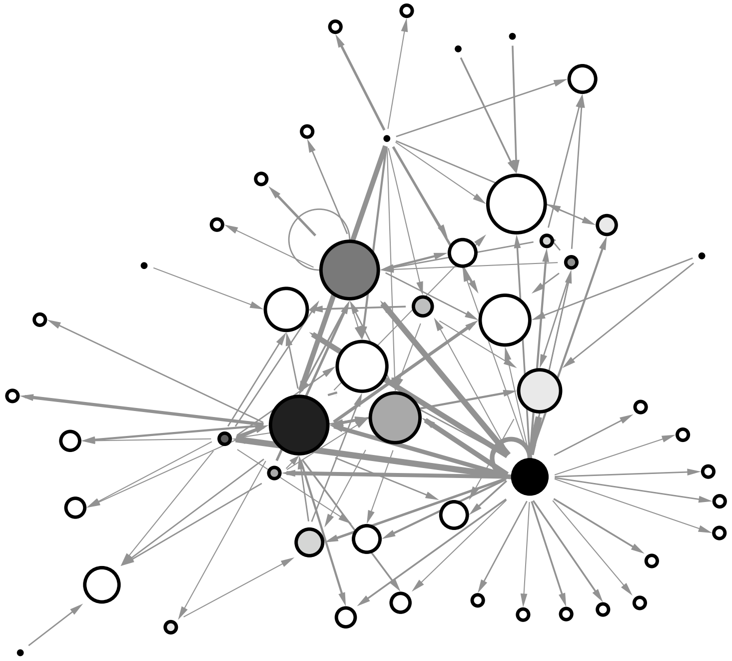


closeness

DEGREE CENTRALITY (UNDIRECTED)



high centralization: one node trading with many others



low centralization: trades are more evenly distributed

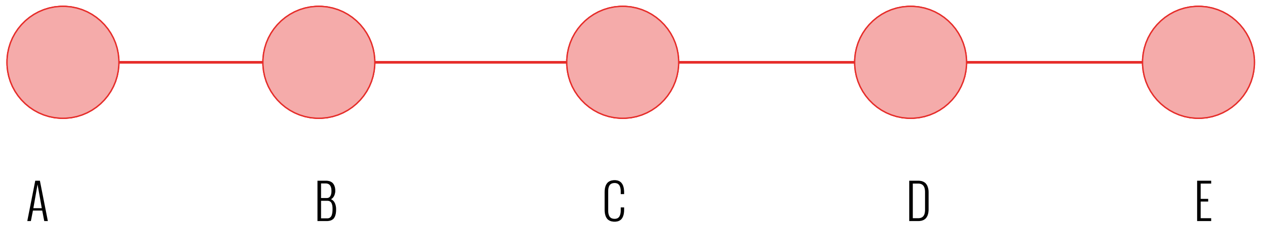
WHERE DEGREE ≠ CENTRALITY

Ability to broker between groups, mediate conflicts, etc.

Likelihood that information originating **anywhere in the network** reaches you quickly.

"BETWEENNESS" CENTRALITY

Intuition: how many paths between others lead through me?

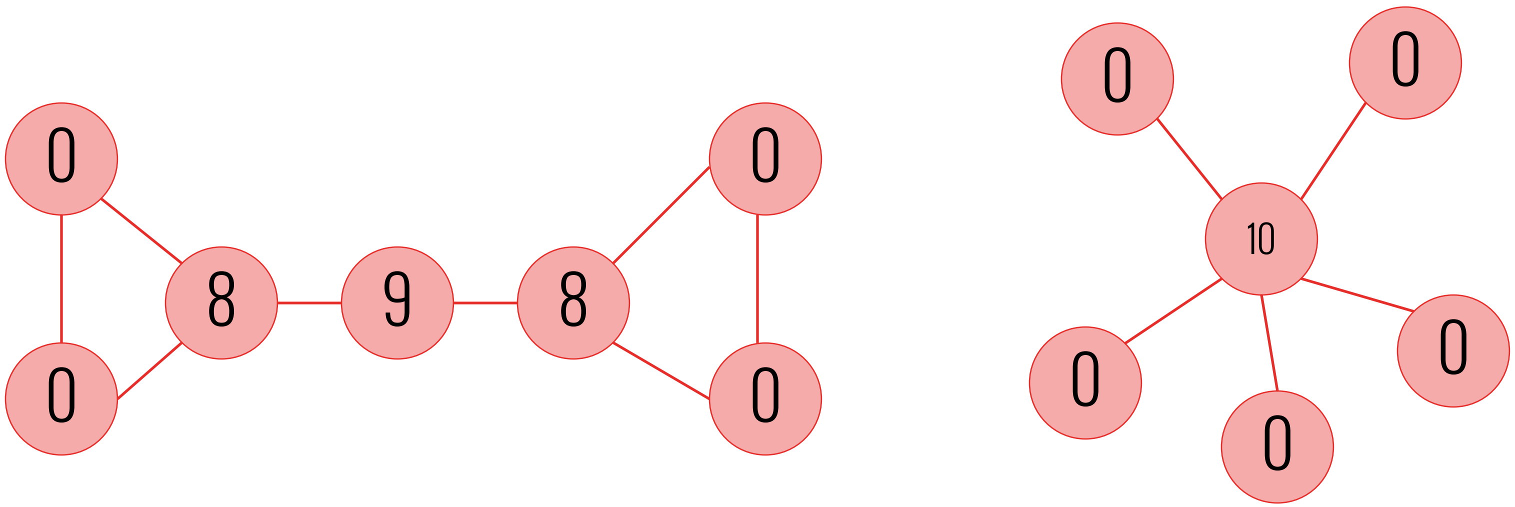


A lies between no two other vertices
B lies between A and 3 other vertices: C, D, and E
C lies between 4 pairs of vertices (A,D),(A,E),(B,D),(B,E)

there are no alternate paths for these pairs to take, so C gets full credit

A FEW MORE EXAMPLES

(non-normalized version)



CLOSENESS CENTRALITY

What if we don't care about the **# of direct friends** (degree) or **number of shortest paths** that flow through (betweenness)?

Closeness measures how close a node is to the middle of things.

"As long as gossip/information can reach me quickly… "

CLOSENESS: DEFINITION

Closeness is based on the length of the average shortest path between a vertex and all vertices in the graph.

$$C_c(i) = \left[\sum_{j=1, j\neq i}^{N} d(i,j) \right]^{-1}$$

All vertices but i

Undirected: shortest path between i and j

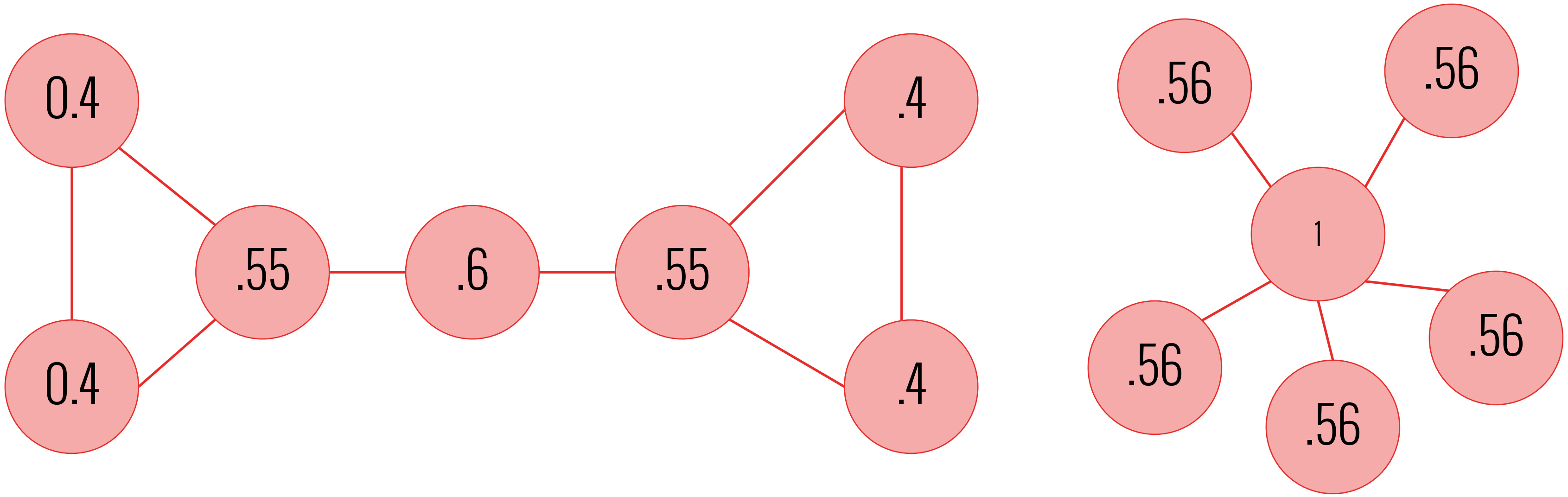
Directed:

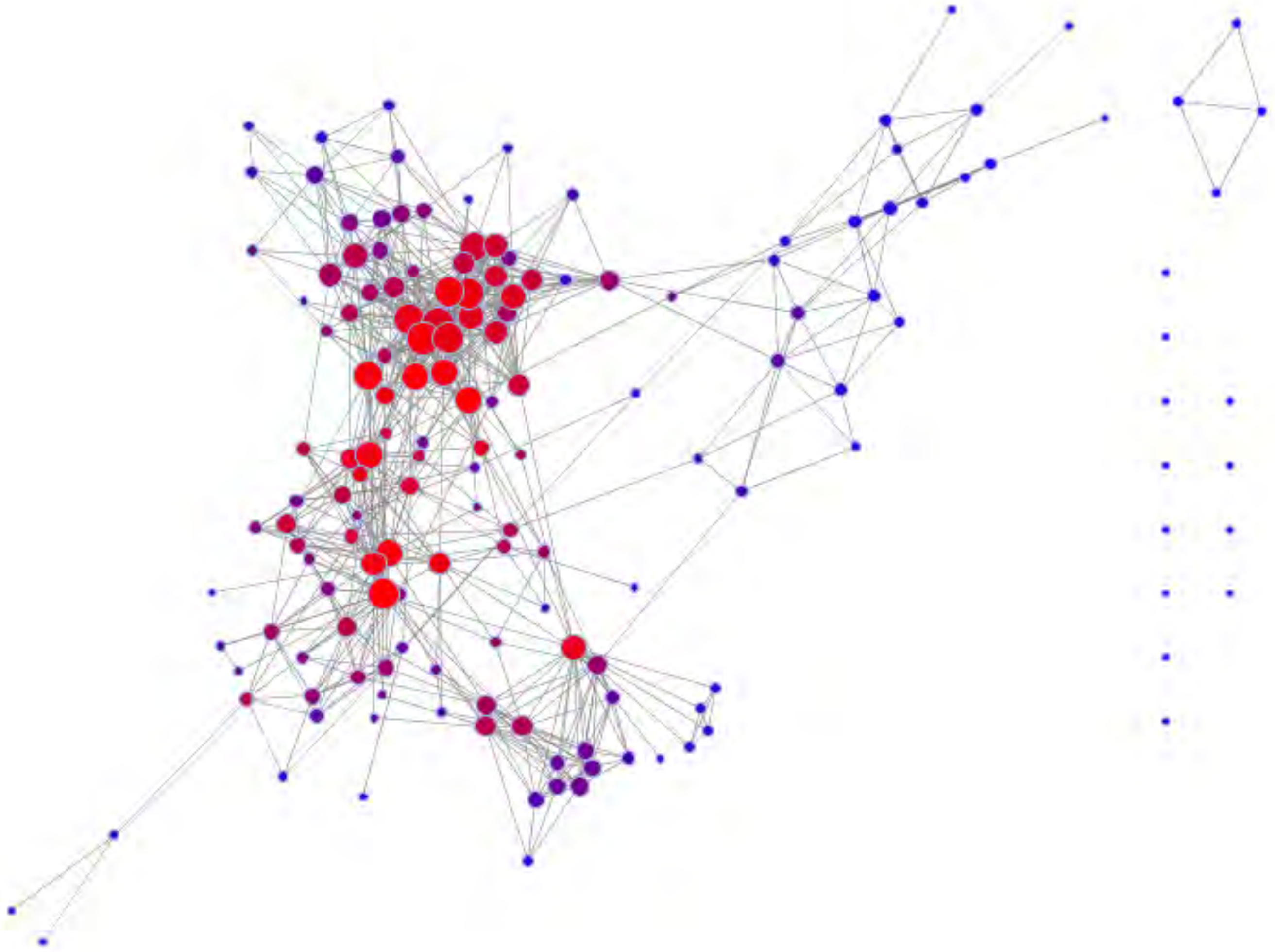
- in: j to i
- out: i to j

Normalized Closeness Centrality

$$C_C^{'}(i) = (C_C(i)) / (N-1) = \frac{N-1}{\sum_{j=1, j\neq i}^{N} d(i,j)}$$

CLOSENESS: EXAMPLES (NORMALIZED)





A TYPICAL FACEBOOK NETWORK

(all of one person's friends)

degree (# direct connections) denoted by size

closeness (length of shortest path to all others) denoted by color

USEFUL ANALYTIC MEASURES FOR GRAPHS

Connected Components – how many disconnected subgraphs?

Density – what % of possible edges exist?

Diameter – within a graph/component, what's the longest (or average) shortest path between nodes?

Degree Distribution – how many nodes have how many neighbors?

Centralization – how close is the average node to all other nodes?

MEASURES FOR FINDING IMPORTANT NODES?

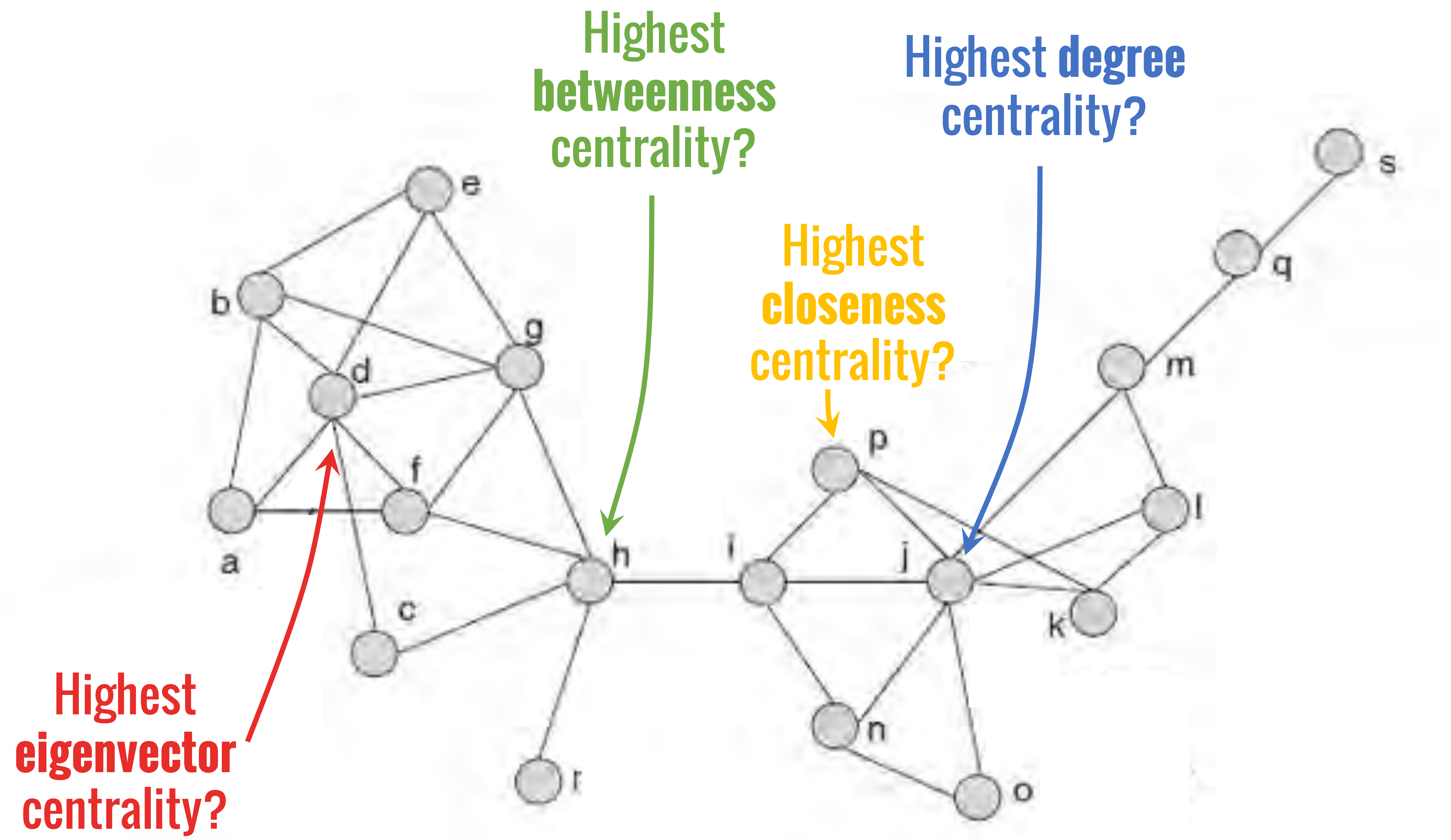
Degree Centrality – find the nodes with the most immediate neighbors. ("most friends")

Betweenness Centrality – find the nodes that sit on the shortest paths between the largest number of node pairs. ("most critical links")

Closeness Centrality – find the nodes that are the closest on average to all of the other nodes. ("closest to everything")

There are other measures as well...

Eigenvector Centrality – find the nodes with the most connections to well-connected nodes. ("best-connected friends")



Alberto Manuel

COMMUNITIES

COMMUNITIES

Various "pressures" (e.g., homophily) lead to similar people linking

Real networks often have small "sub-communities"

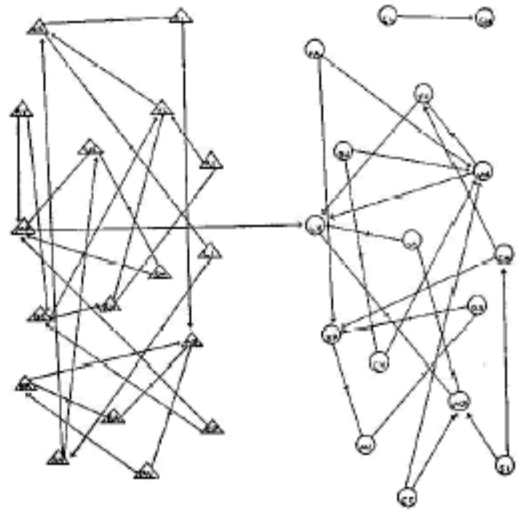
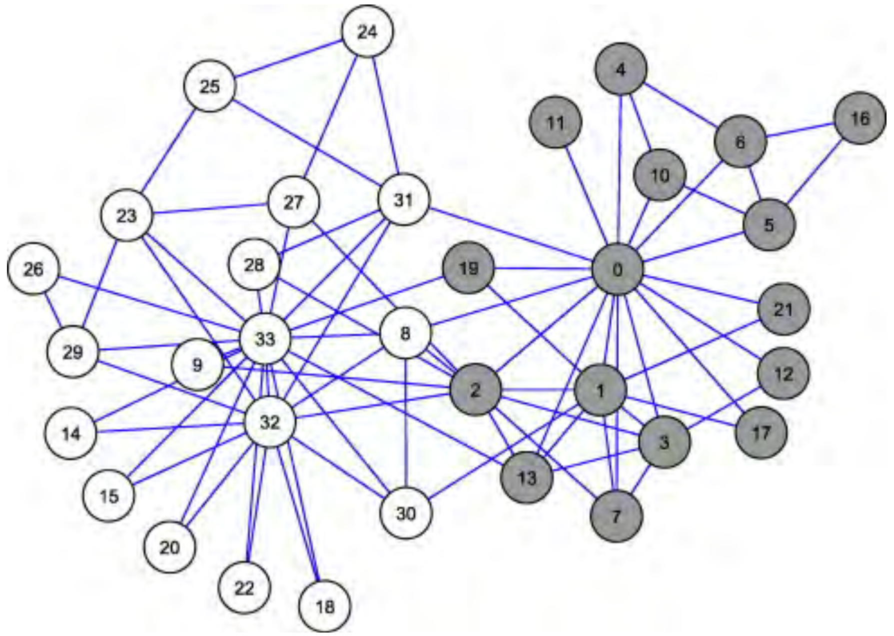


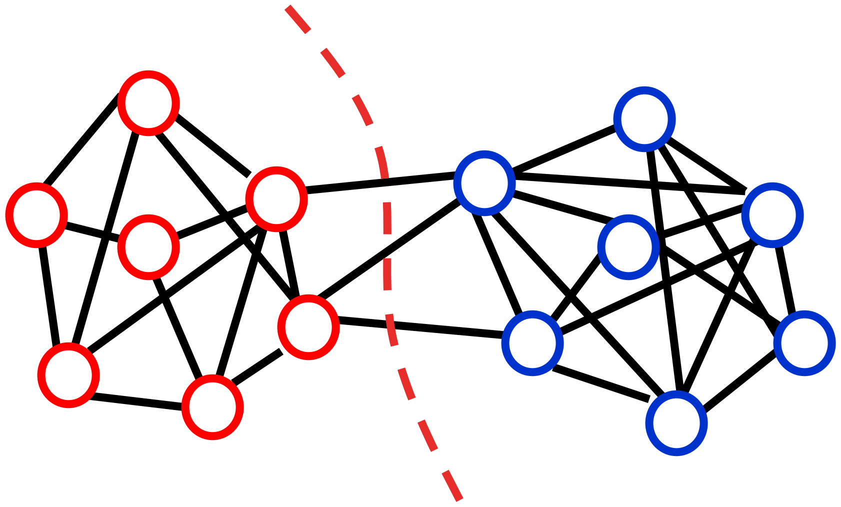
Figure 1. An Attraction Network in a Fourth Grade Class (from Moreno [19], p. 38).



COMMUNITY FINDING

Social and other networks have a **natural community structure**

We want to **discover this structure** rather than impose a certain size of community or fix the number of communities



Can we discover community structure **in an automated way**?

HIERARCHICAL CLUSTERING

Develop a similarity (or dissimilarity) measure between pairs of data points (vertices).

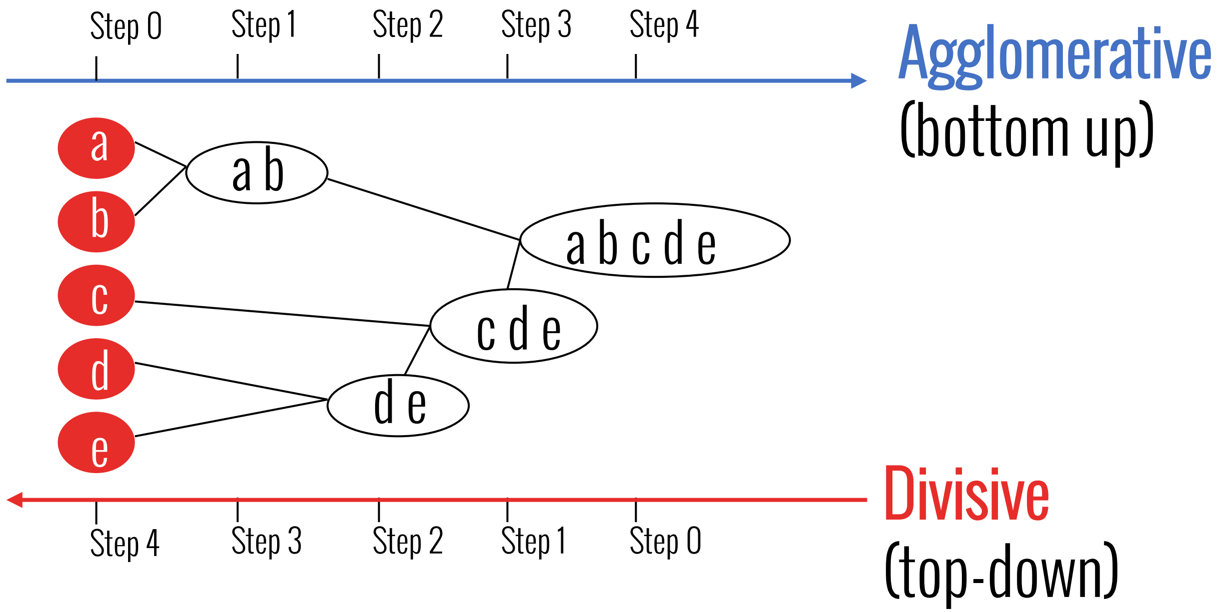
Use the measure as a heuristic to gradually..

Group the most similar data points/clusters (if bottom up)

or

Divide the most dissimilar clusters/data points (if top-down)

PRETTY MUCH LIKE OTHER HIERARCHICAL CLUSTERING APPROACHES … **EXCEPT WE CAN USE PROPERTIES OF THE NETWORK TO DETERMINE (DIS)SIMILARITY**



BOTTOM-UP HIERARCHICAL COMMUNITY FINDING

Develop a **similarity (or dissimilarity)** measure x_{ij} between a pair (i,j) of vertices (or communities).

For example "we are similar if we are connected to the same neighbors"

$$X_{ij} = \sqrt{\sum_{k \neq i,j} \left(A_{ik} - A_{jk}\right)^2}$$

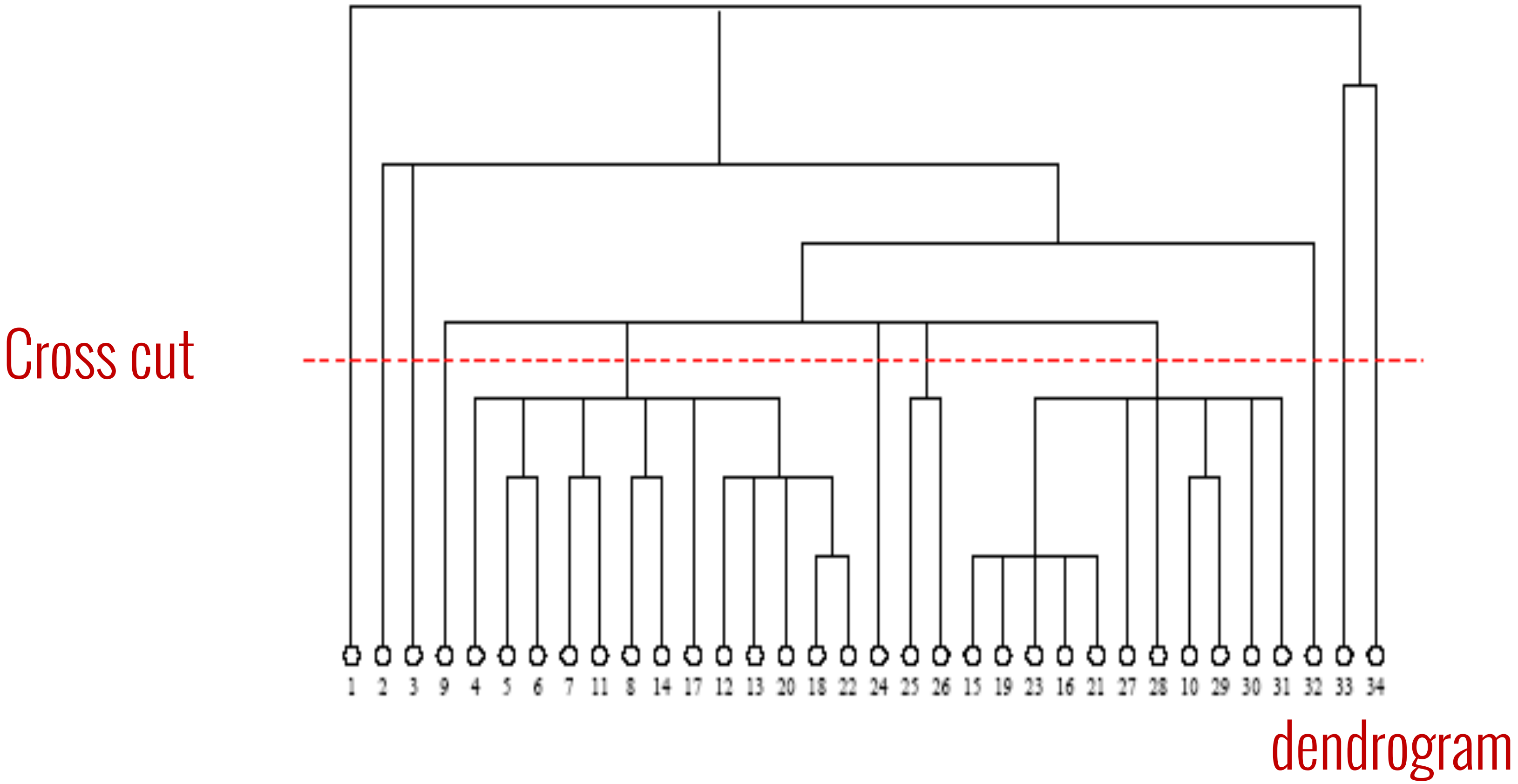
Apply the **hierarchical clustering** and build the dendogram.

- Gradually merge the most similar pair of vertices (communities)
- Update the similarity between communities

Cross cut the dendogram at any level to give the communities at that level.

HIERARCHICAL COMMUNITY FINDING

result: nested components, where one can take a 'slice' at any level of the tree



BETWEENNESS CLUSTERING

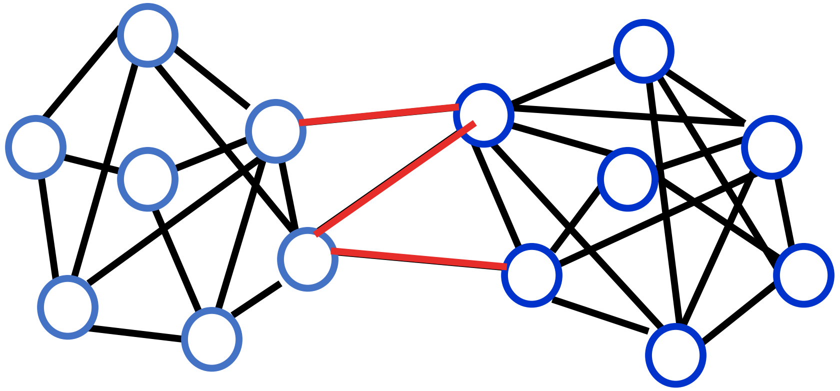
Algorithm

compute the betweenness of all edges

while (betweenness of any edge > threshold):

remove edge with highest betweenness

recalculate betweenness



BETWEENNESS CLUSTERING - DOWNSIDES

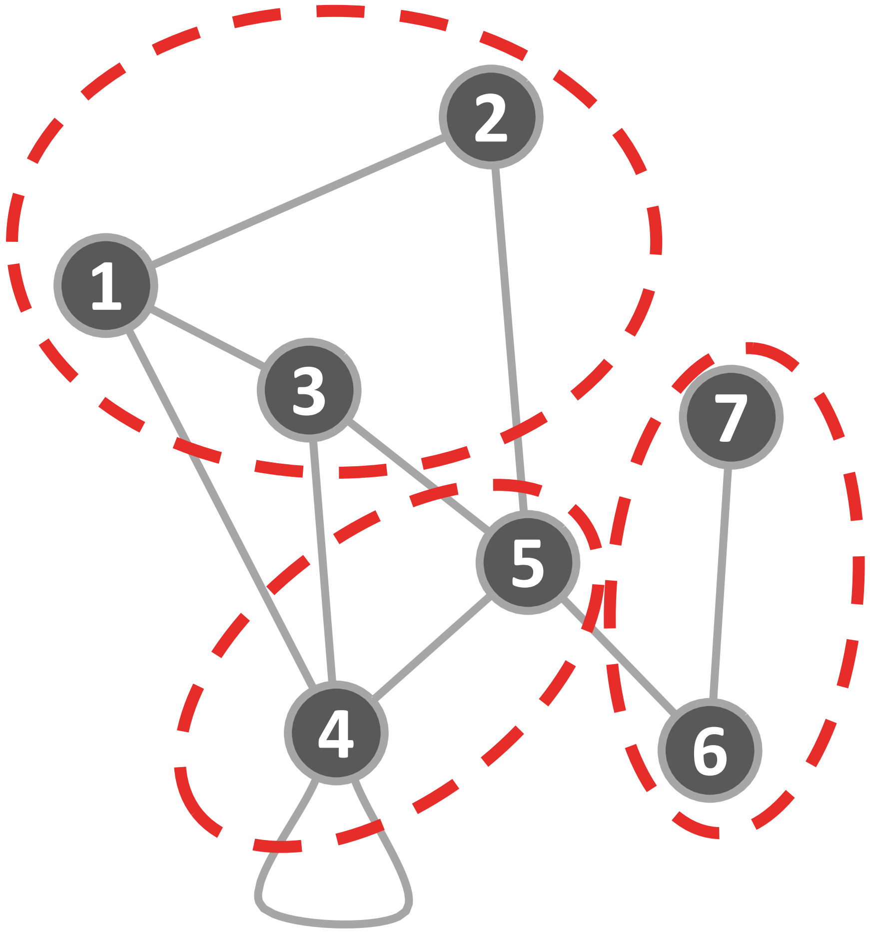
Betweenness **needs to be recalculated at each step**

- Removal of an edge can impact the betweenness of another edge
- **Very expensive**: all pairs shortest path – $O(N^3)$
- May need to repeat up to N times
- **Does not scale well** to more than a few hundred nodes
  (even with the fastest algorithms)
- Still have to choose **how many groups** you want

MODULARITY

How to find **cohesive groups** in a network?

One possibility: Look whether cohesion **within** each community is higher than **outside.**



$$Q = \sum \left(\frac{\textit{edges inside}}{\textit{community}} - \frac{\textit{expected edges}}{\textit{inside community}} \right)$$

MODULARITY

Is there an edge between i and j?
(value from adjacency matrix)

$$Q = \frac{1}{4m} \sum_{\substack{i,j \\ in\ same \\ module}} \left(A_{ij} - \frac{k_i k_j}{2m} \right)$$

Normalization
(m = # of edges)

probability a random edge would go between i and j
(k_n = degree for node *n*)

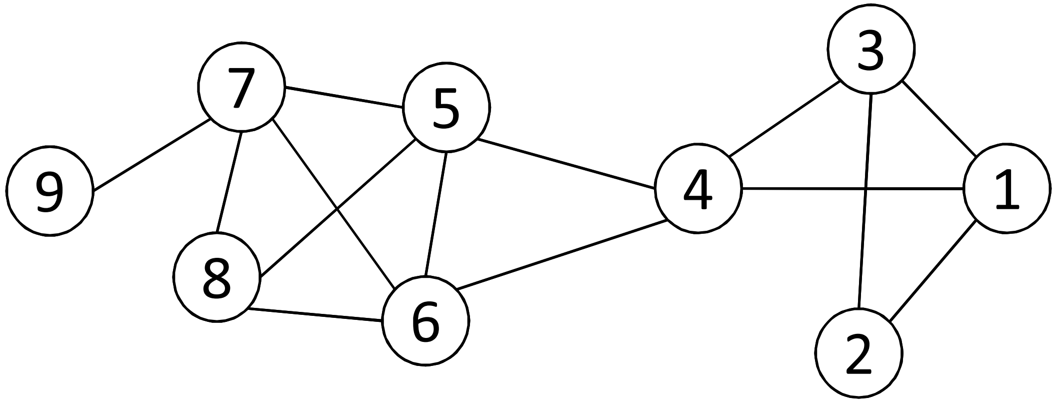
For a random network, Q = 0

the number of edges within a community is no different from what you would expect

MODULARITY

$$Q = \frac{1}{4m} \sum_{\substack{i,j \\ \textit{in same} \\ \textit{module}}} \left(A_{ij} - \frac{k_i k_j}{2m} \right)$$

probability a random edge would go between i and j
(k_n = degree for node *n*)



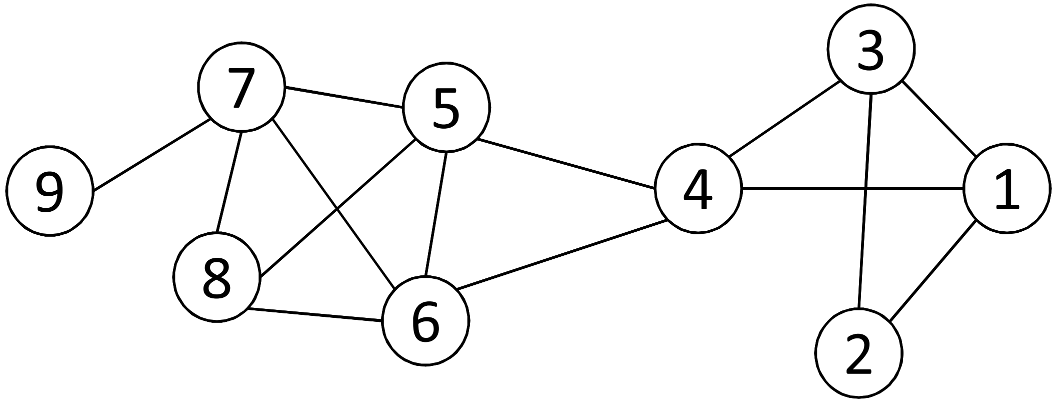
For **i** = **node 1** and **j** = **node 2**

$$\frac{k_i k_j}{2m} = \frac{3*2}{2*14} = 0.214$$

MODULARITY

$$Q = \frac{1}{4m} \sum_{\substack{i,j \\ in\ same \\ module}} \left(A_{ij} - \frac{k_i k_j}{2m} \right)$$

Check to see if they're actually connected



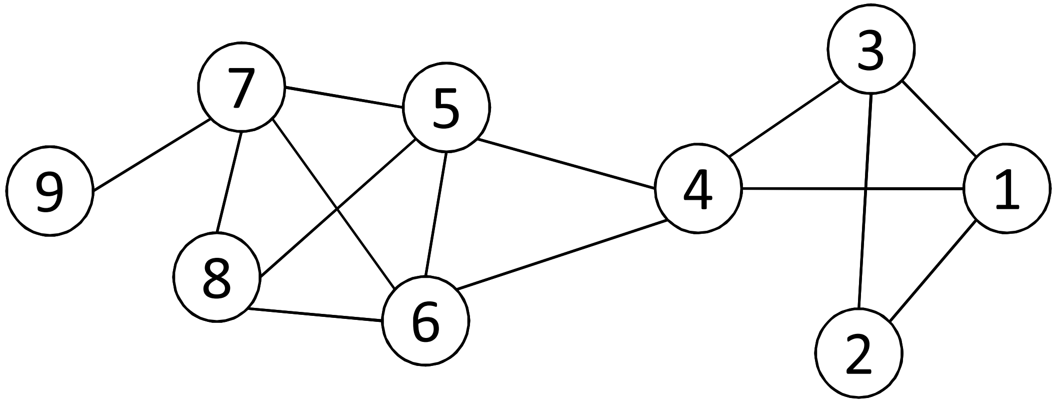
For i = node 1 and j = node 2

$$1 - 0.214 = 0.786$$

MODULARITY

$$Q = \frac{1}{4m} \sum_{\substack{i,j \\ in\ same \\ module}} \left(A_{ij} - \frac{k_i k_j}{2m} \right)$$

Sum for for all pairs in module



Modularity ranges from -1 to 1
More positive if # of edges in group
Is greater than expected.

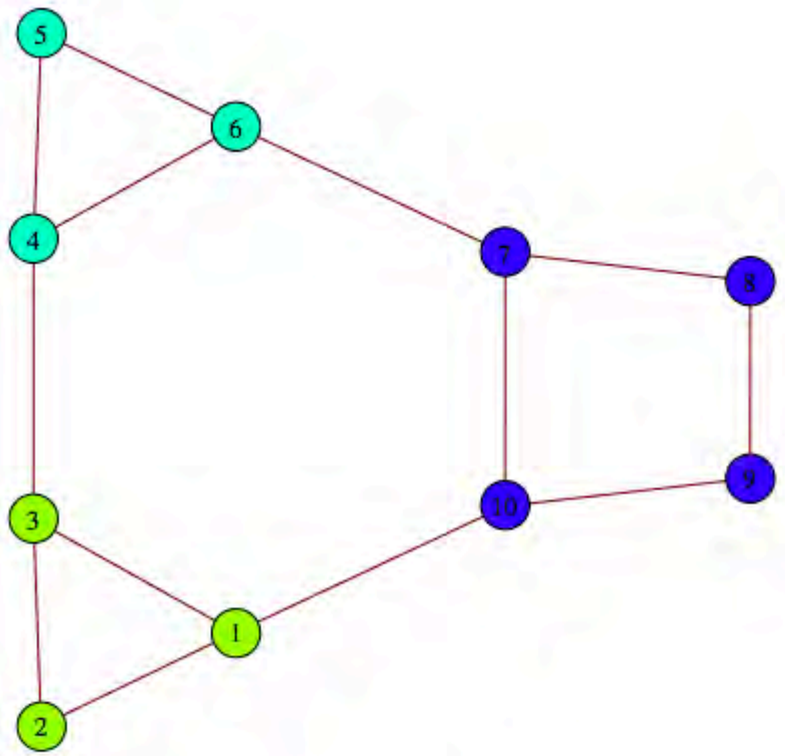
ONE WAY TO APPLY THIS

Finding the configuration with maximum modularity is **NP-complete.**
But good **approximation algorithms** exist.

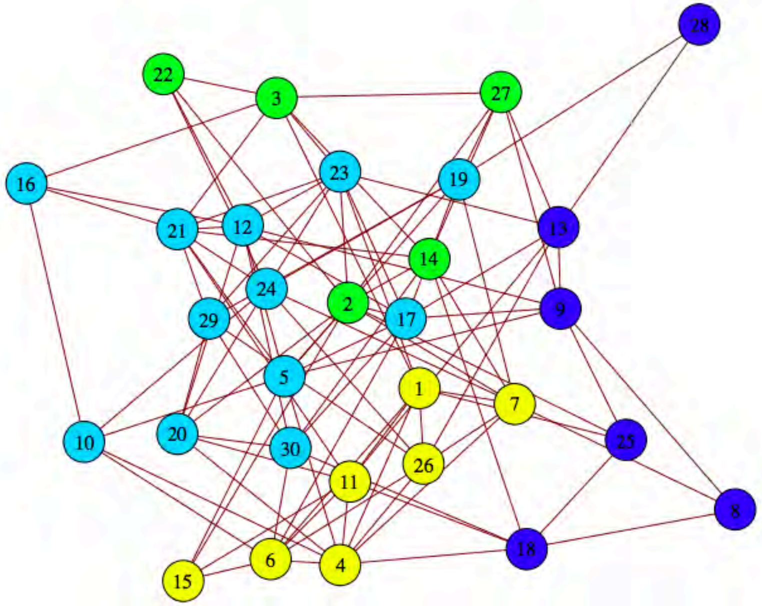
Start with **individual vertices**
Follow a **greedy strategy**:

- successively join clusters with the greatest increase ΔQ in modularity
- stop when the maximum possible ΔQ <= 0 from joining any two

OFTEN PRODUCES NICE BALANCED SETS



Communities Assigned to a small graph



Communities assigned to a random graph

COMMUNITY DETECTION

In NetworkX

`nx.community`

Package with a bunch of different methods, including:

Hierarchical (Girvan-Newman)

```
my_communities = nx.community.girvan_newman(my_graph)
```

K-Clique

```
my_communities = k_clique_communities(my_graph, k=4)
```

+ Label propagation, bi-partitions, fluid-communities, and a bunch of other methods …

In Gephi

1. **Run modularity computations** on your graph to produce community labels for each node
2. Then show partitions by **mapping those labels to color**, size, etc.

SUMMARY

Networks and graphs are very common

Often **hard to visualize** – especially at scale

Some simple metrics/methods can help **identify important nodes** and **separate communities**

Per usual, **many more strategies exist.**

LETS TRY THIS WITH GEPHI & NETWORKX