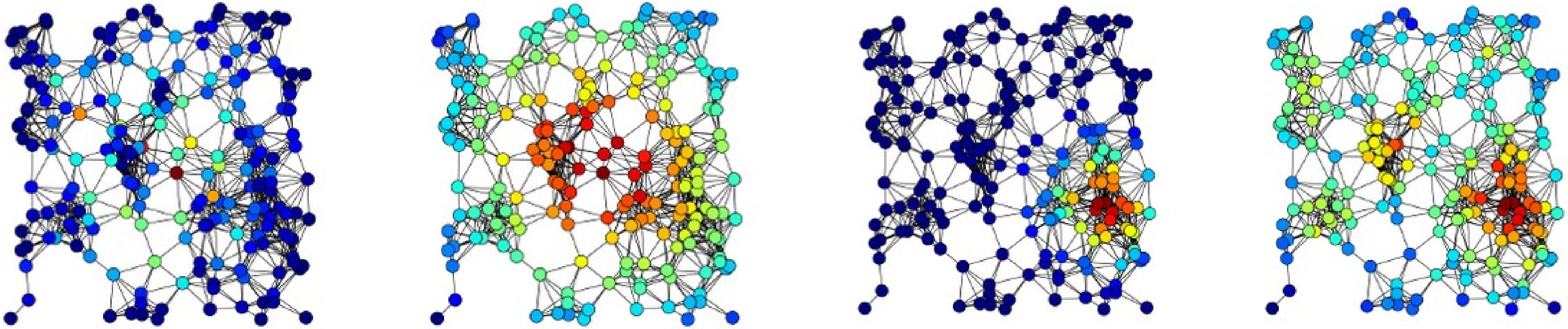
Node Centrality

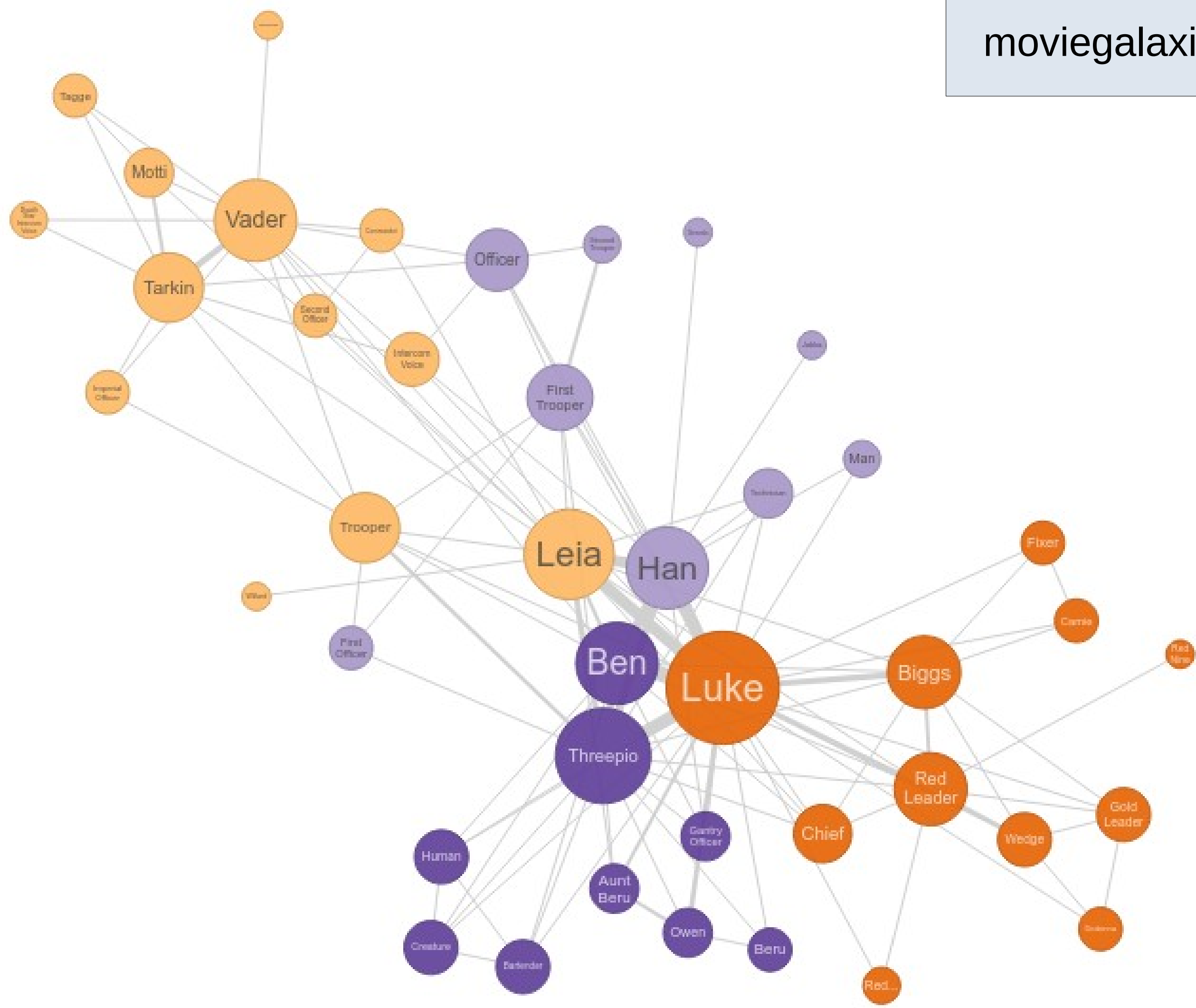
Pedro Ribeiro
(DCC/FCUP & CRACS/INESC-TEC)



(Heavily based on slides from Jure Leskovec and Lada Adamic @ Stanford University)

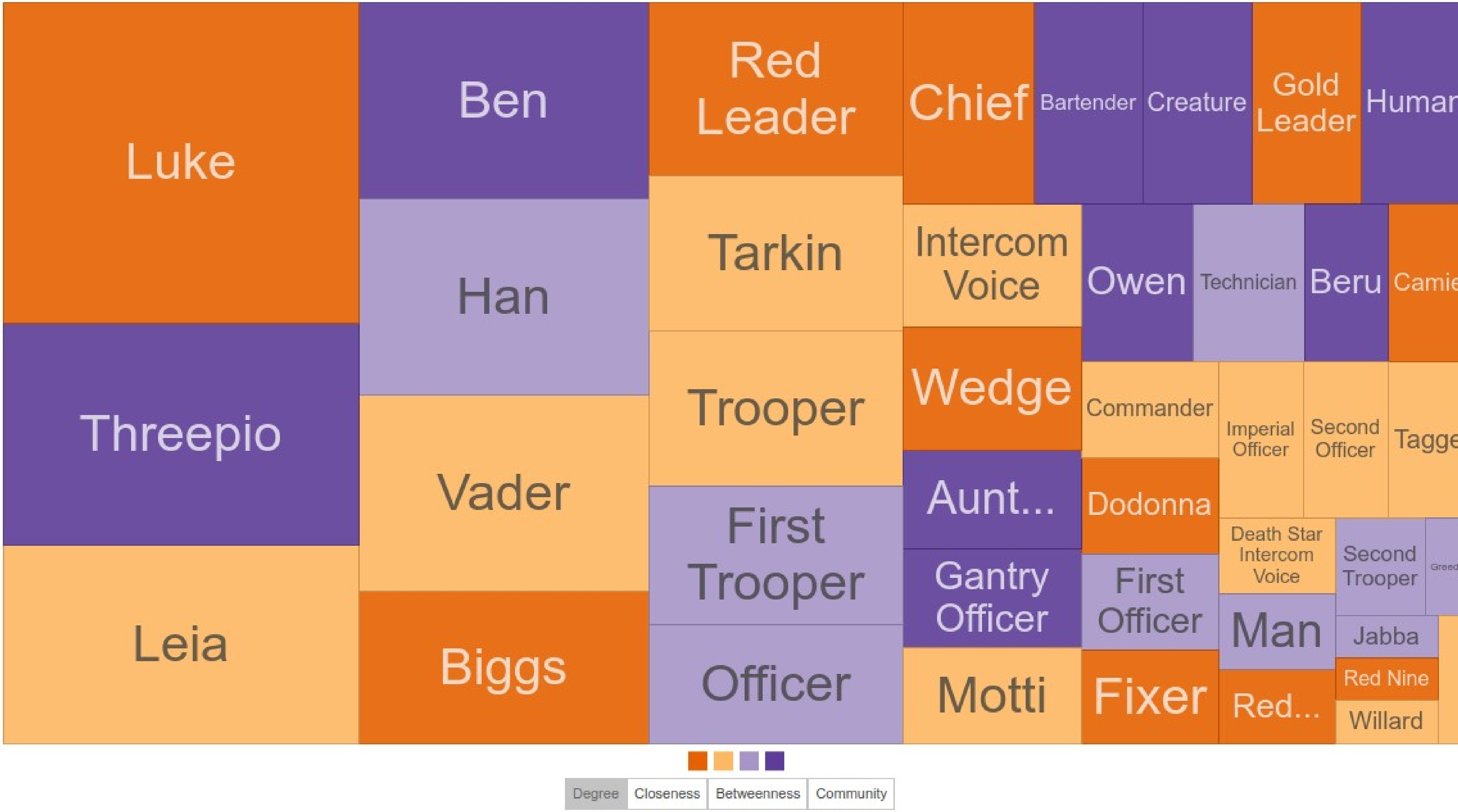
Star Wars IV Network

moviegalaxies.com



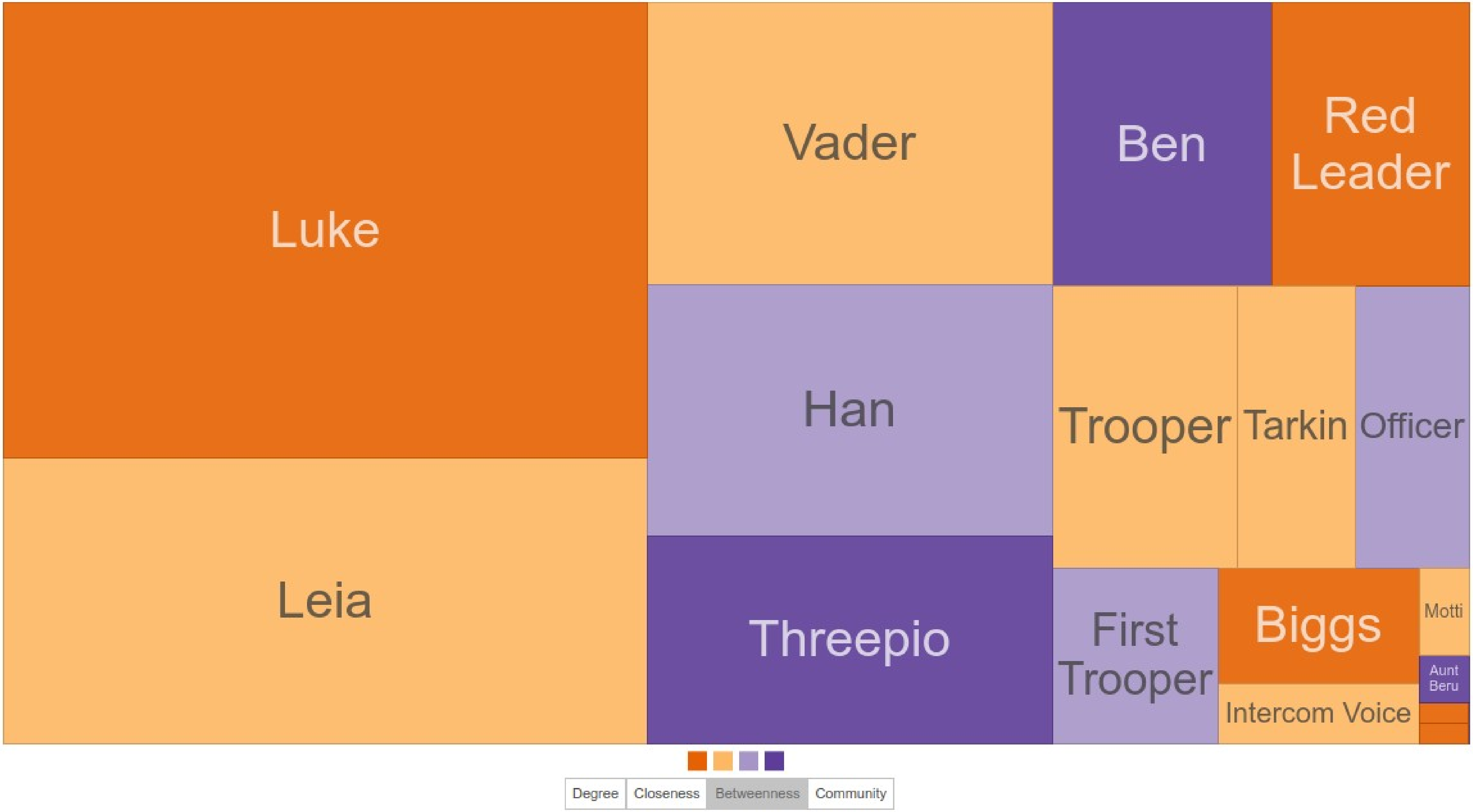
Are all nodes “equal”? How to measure their importance?

Star Wars IV Network



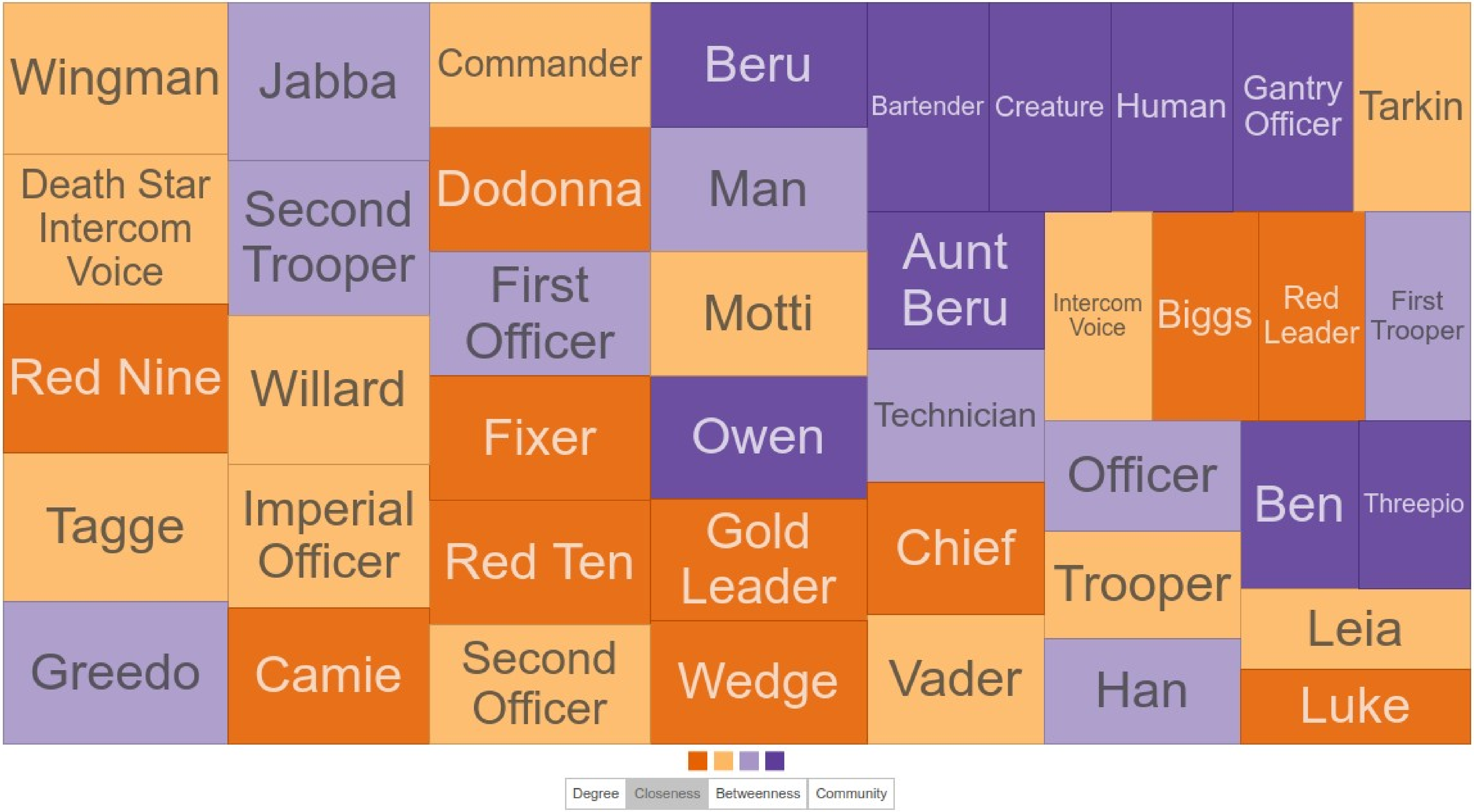
Size proportional to degree: is this the only way?

Star Wars IV Network



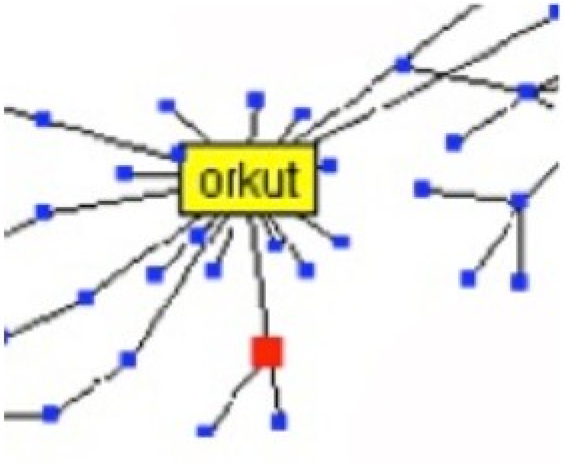
Size proportional to betweenness

Star Wars IV Network



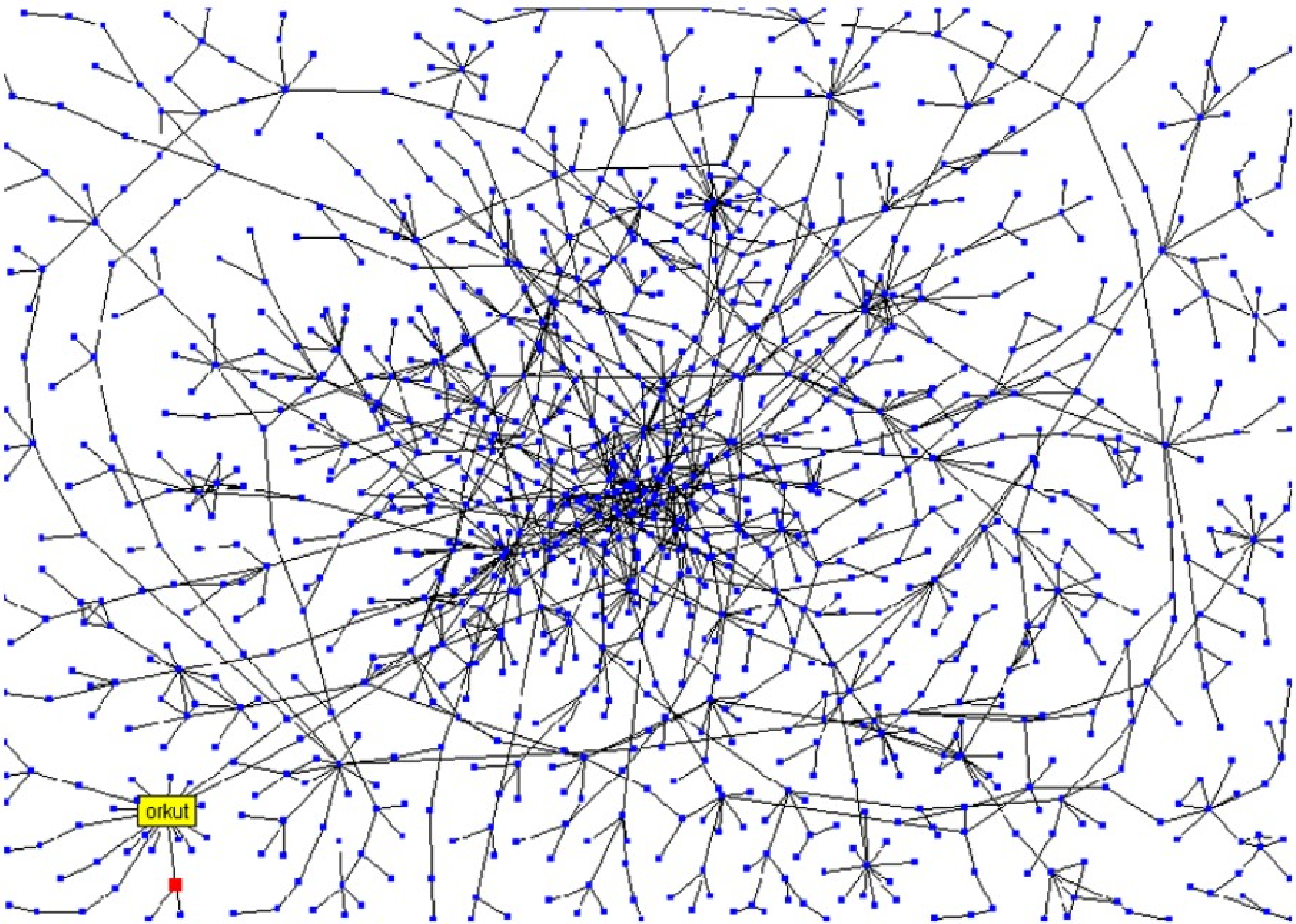
Size proportional to closeness

Why degree is not enough



Why degree is not enough

Stanford Social Web (ca. 1999)

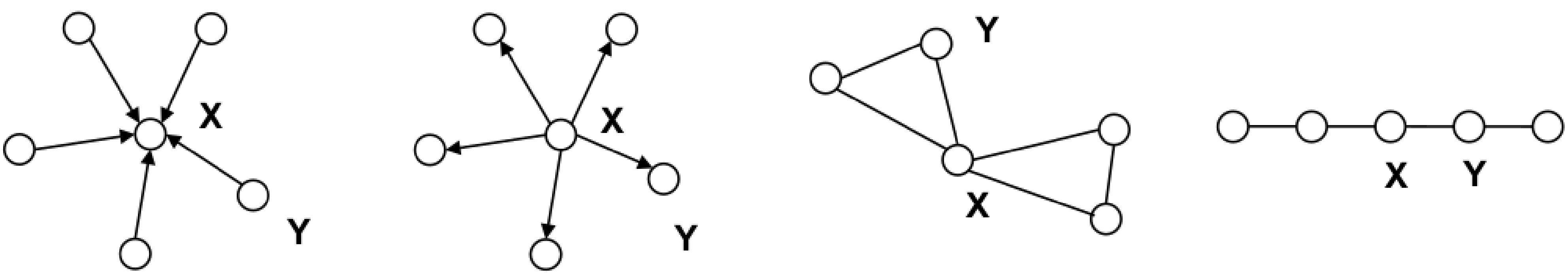


network of personal homepages at Stanford

Different notions of centrality

- **Node Centrality** measures "importance"

In each of the following networks, X has higher centrality than Y according to a particular measure

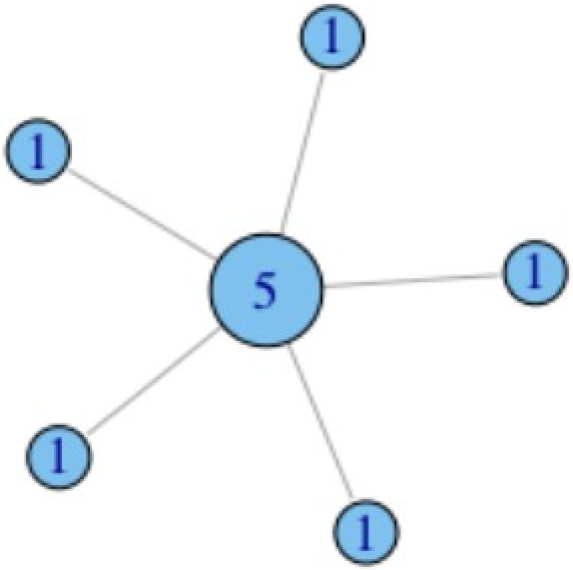


indegree outdegree betweenness closeness

Node Degree

- Let's put some **numbers** to it

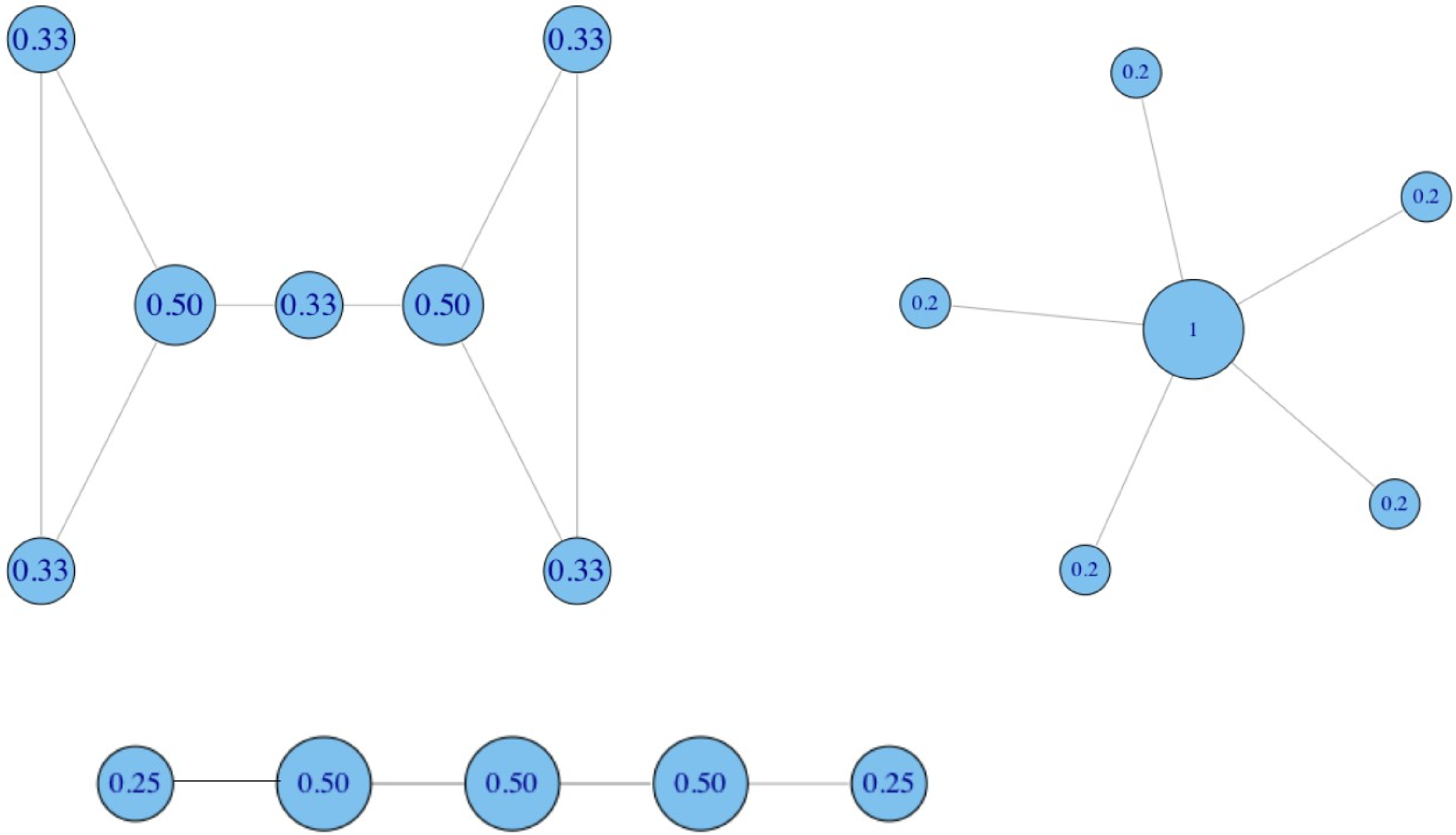
Undirected degree:
e.g. nodes with more friends are more central.



Assumption: the connections that your friend has don't matter, it is what they can do directly that does (e.g. go have a beer with you, help you build a deck...)

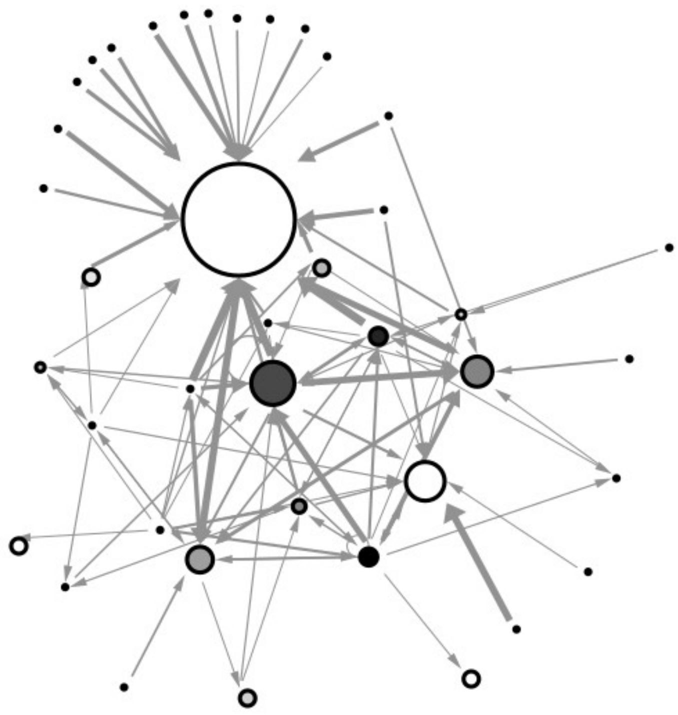
Node Degree

- **Normalization:**
  divide degree by the max. possible, i.e. (N-1)

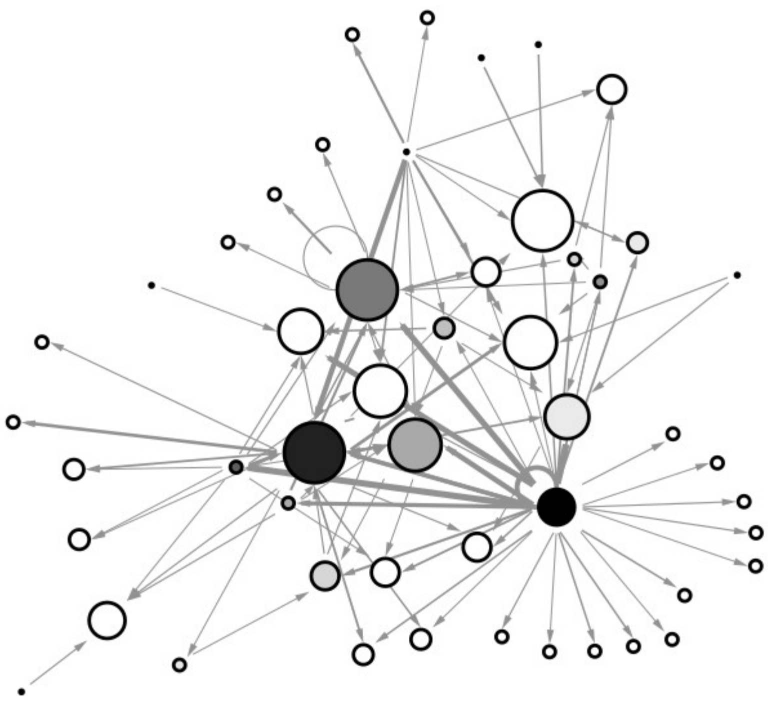


Node Degree

example financial trading networks



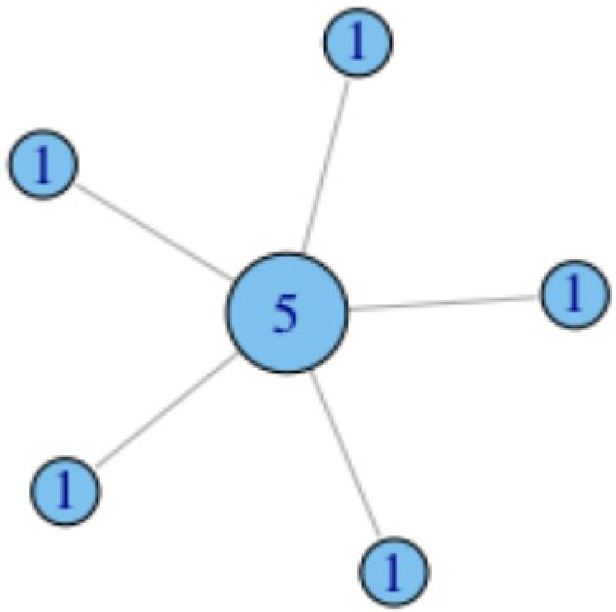
high in-centralization:
one node buying from
many others

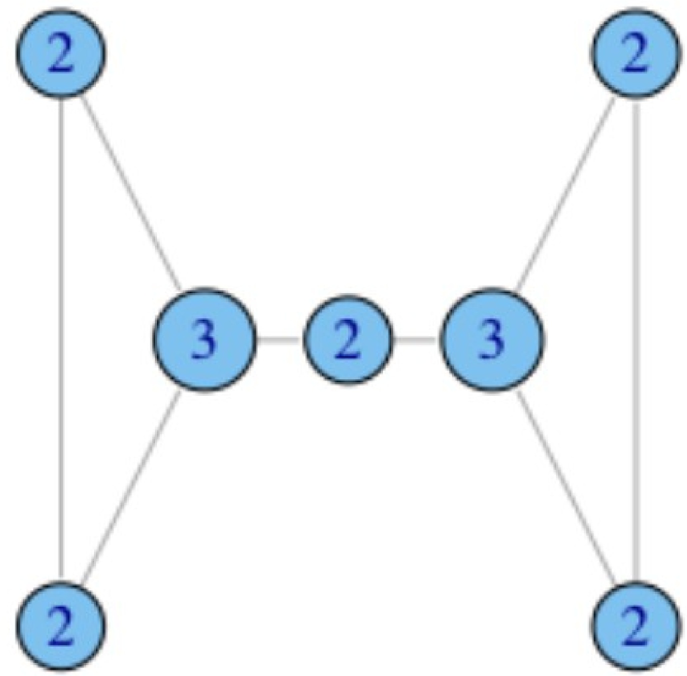


low in-centralization:
buying is more evenly
distributed

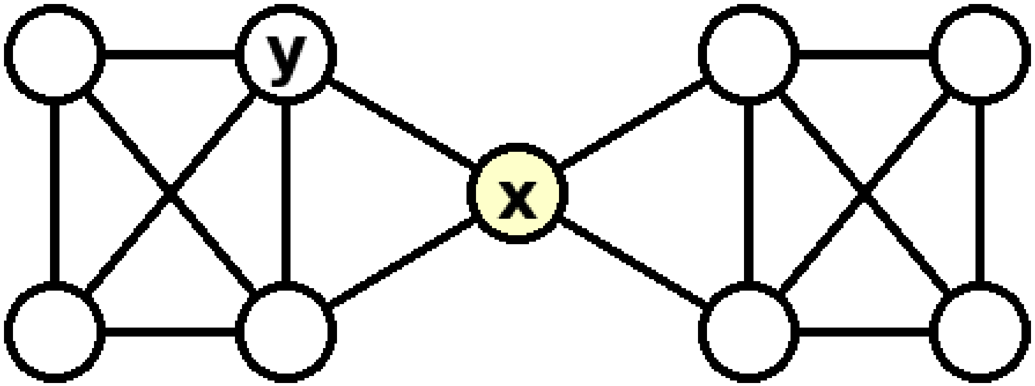
What does degree not capture?

- In what ways does degree fail to capture centrality in the following graphs?

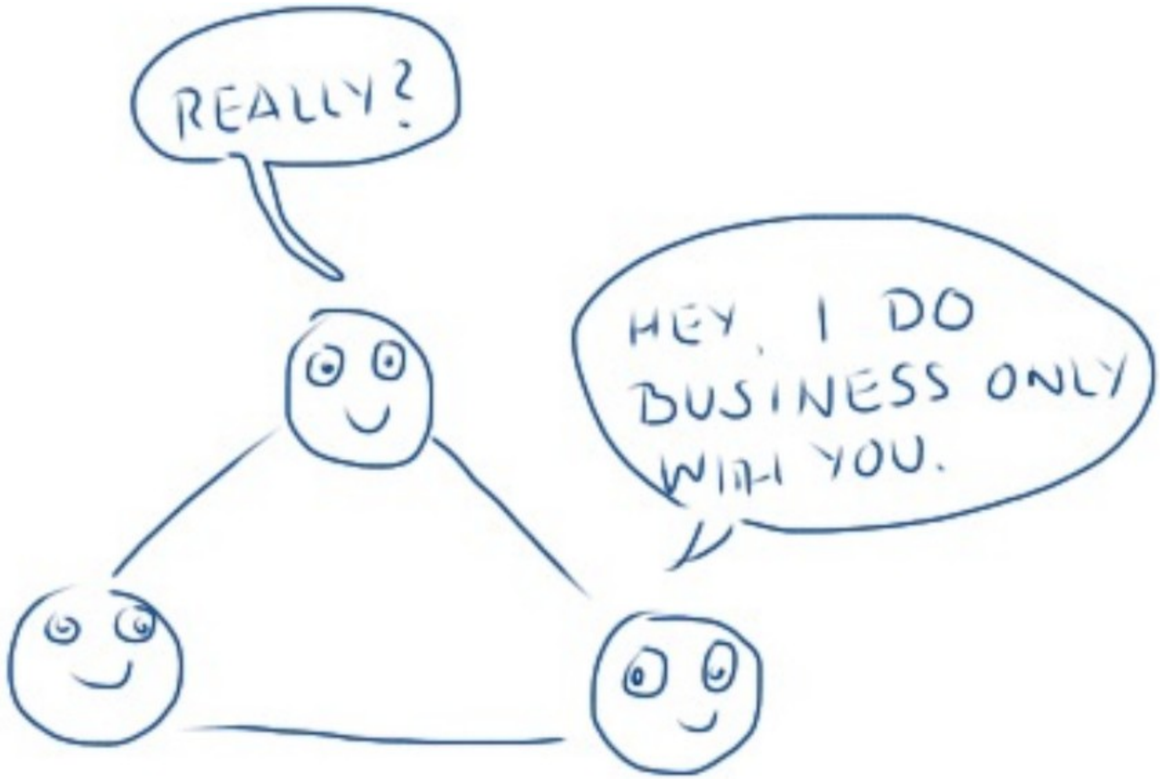




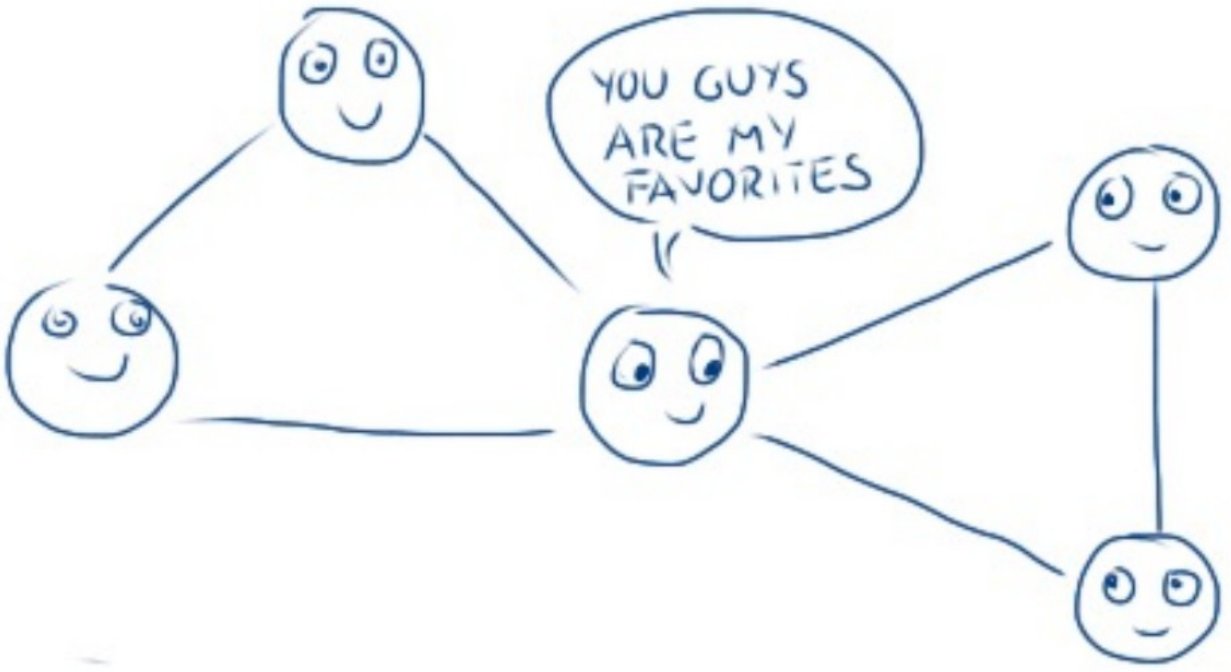
Brokerage not captured by degree



Brokerage: Concept



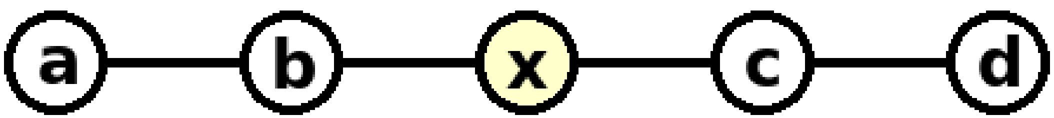
Brokerage: Concept



Capturing Brokerage

- **Betweenness Centrality:**

intuition: how many **pairs of individuals** would have to go through you in order to reach one another in the **minimum number of hops**?



Betweenness: Definition

$$C_B(i) = \sum_{j<k} \frac{g_{jk}(i)}{g_{jk}}$$

Where:

g_{jk} = the number of **shortest paths** connecting nodes *j* and *k*

$g_{jk}(i)$ = the number that node *i* is on.

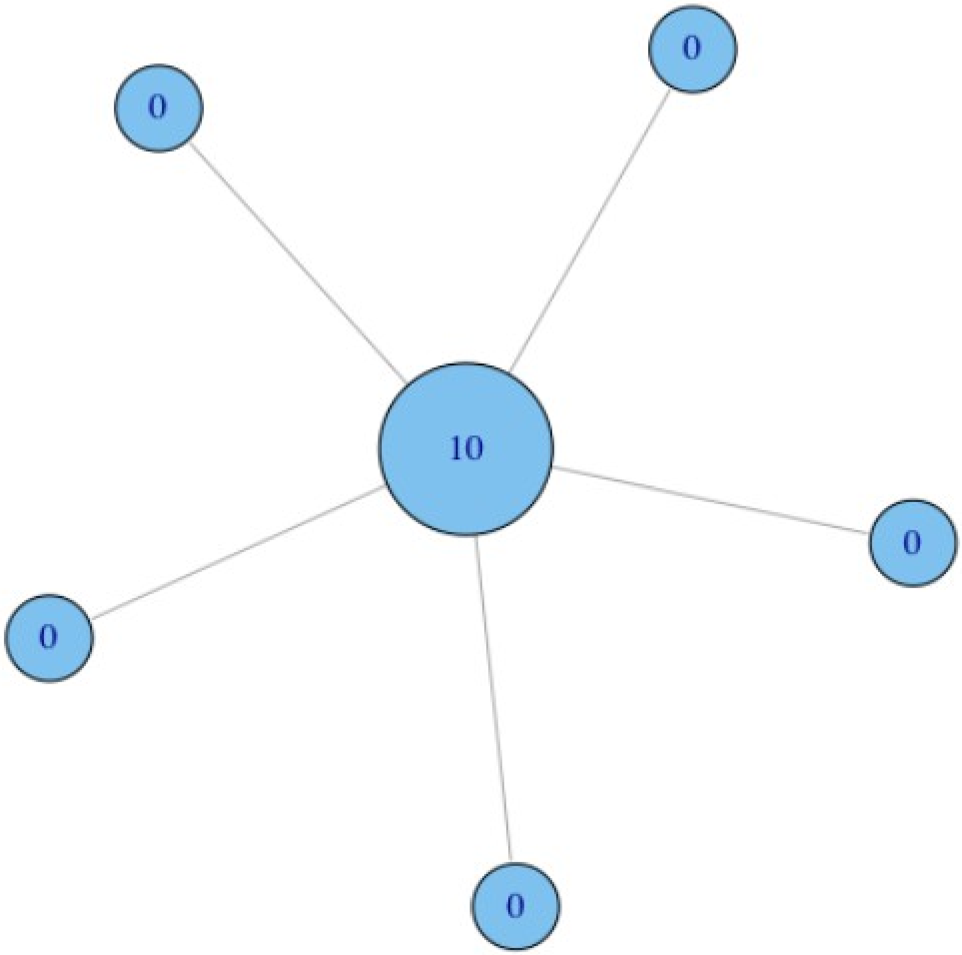
Usually normalized by:

$$C'_B(i) = \frac{C_B(i)}{(n-1)(n-2)/2}$$

number of pairs of vertices excluding the vertex itself

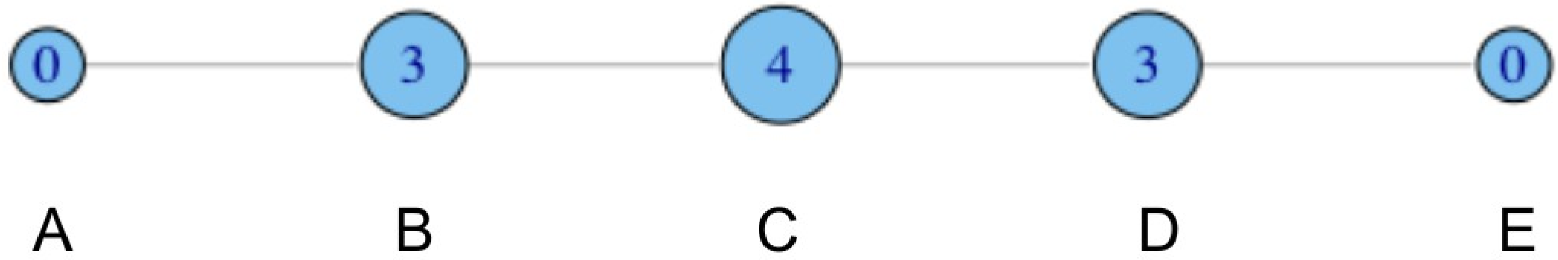
Betweenness: Toy Networks

- Non-normalized version:



Betweenness: Toy Networks

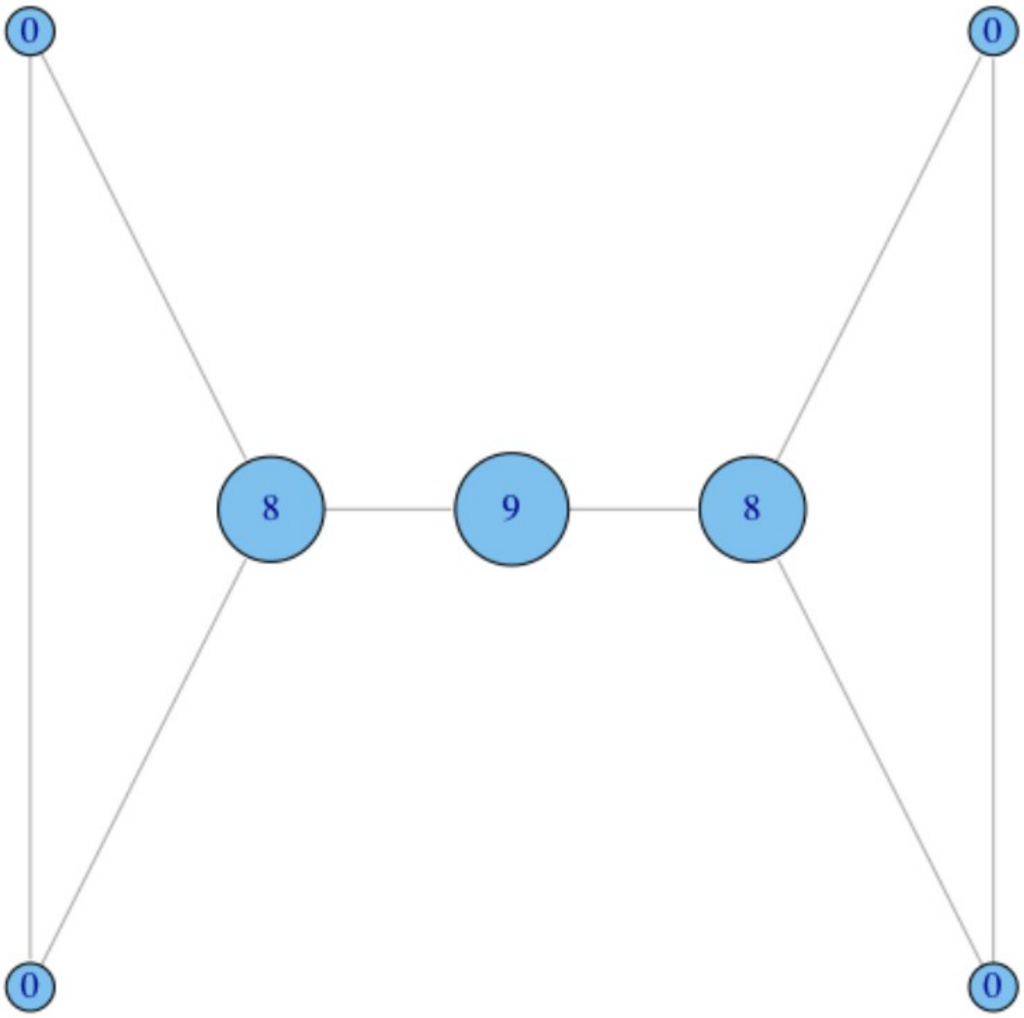
- Non-normalized version:



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

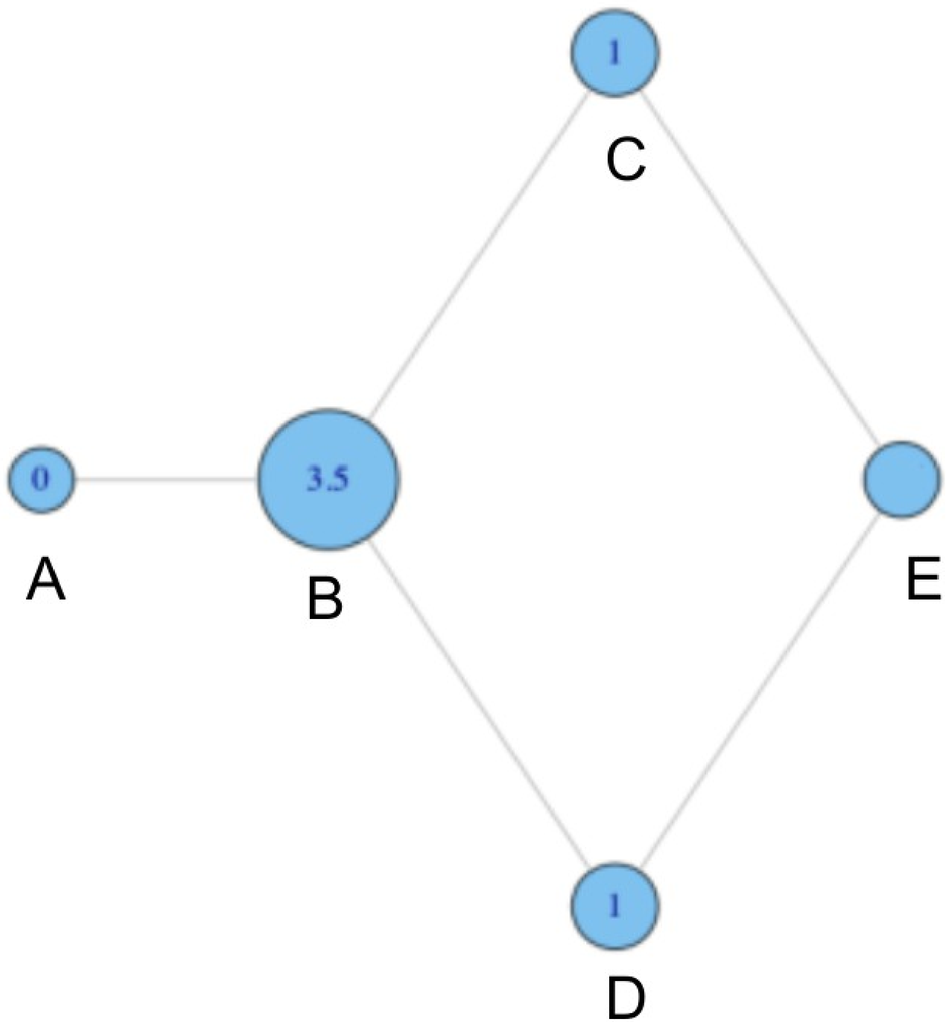
Betweenness: Toy Networks

- Non-normalized version:



Betweenness: Toy Networks

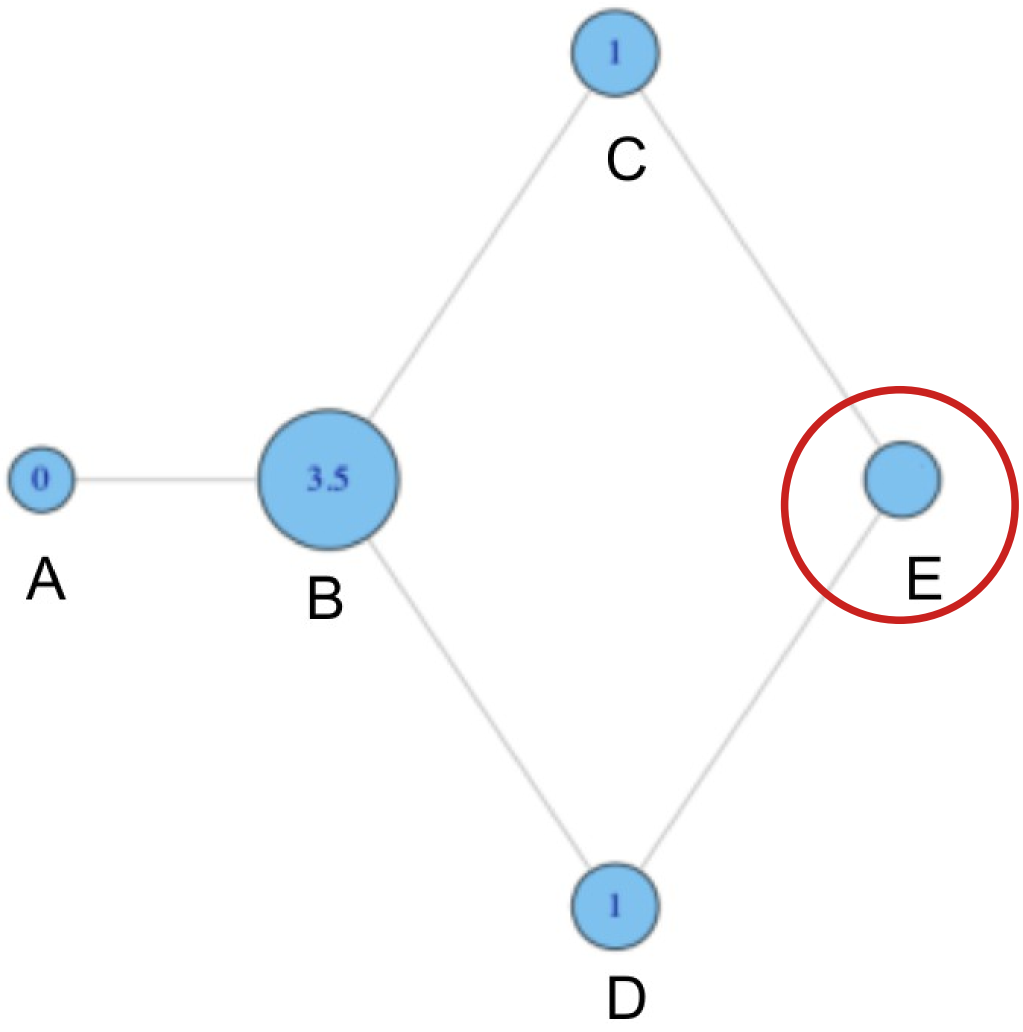
- Non-normalized version:



- why do C and D each have betweenness 1?
- They are both on shortest paths for pairs (A,E), and (B,E), and so must share credit:
  - 1⁄2+1/2 = 1

Betweenness: Toy Networks

- Non-normalized version:

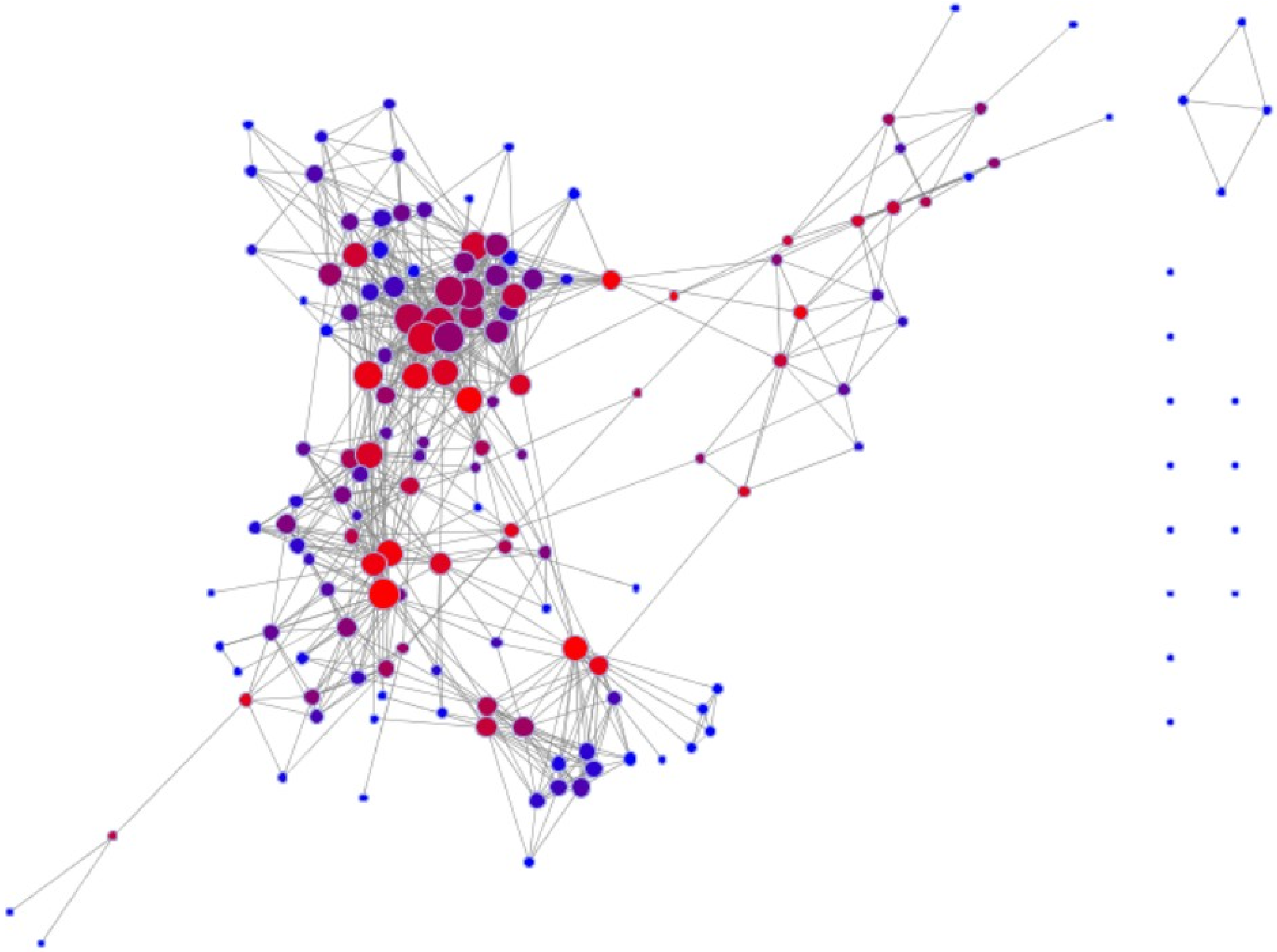


What is the betweenness of node E?

Betweenness: Real Example

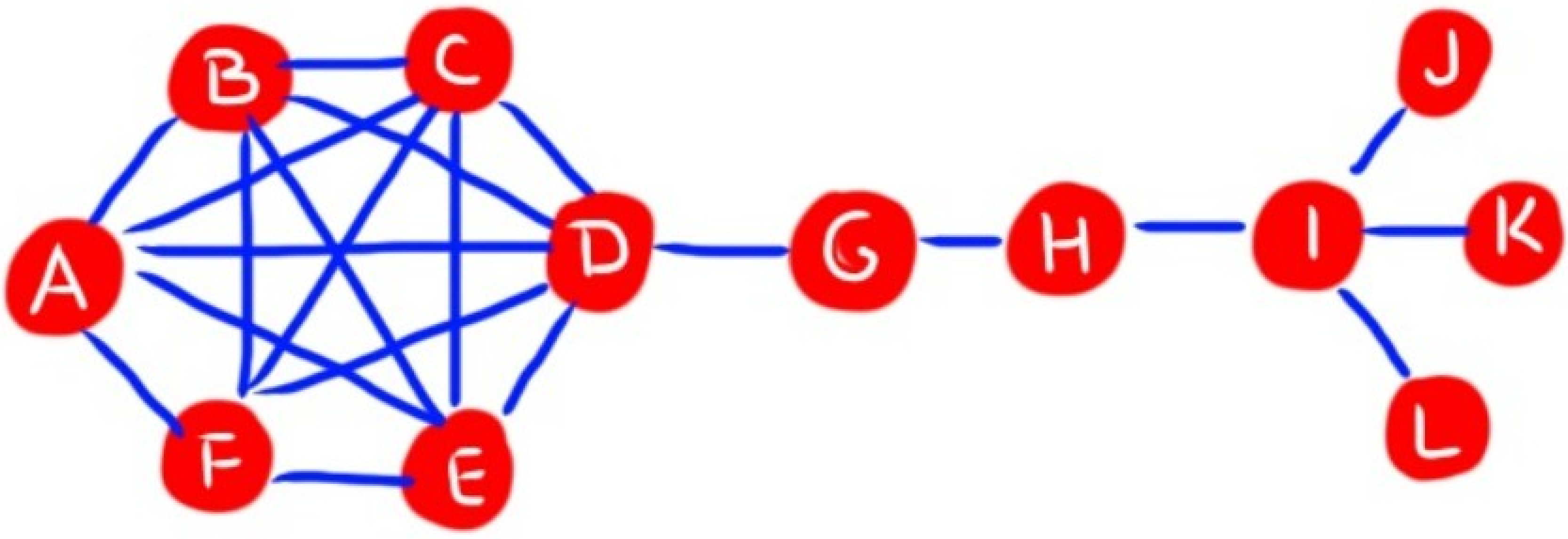
- Social Network (facebook)

  nodes are sized by degree, and colored by betweenness



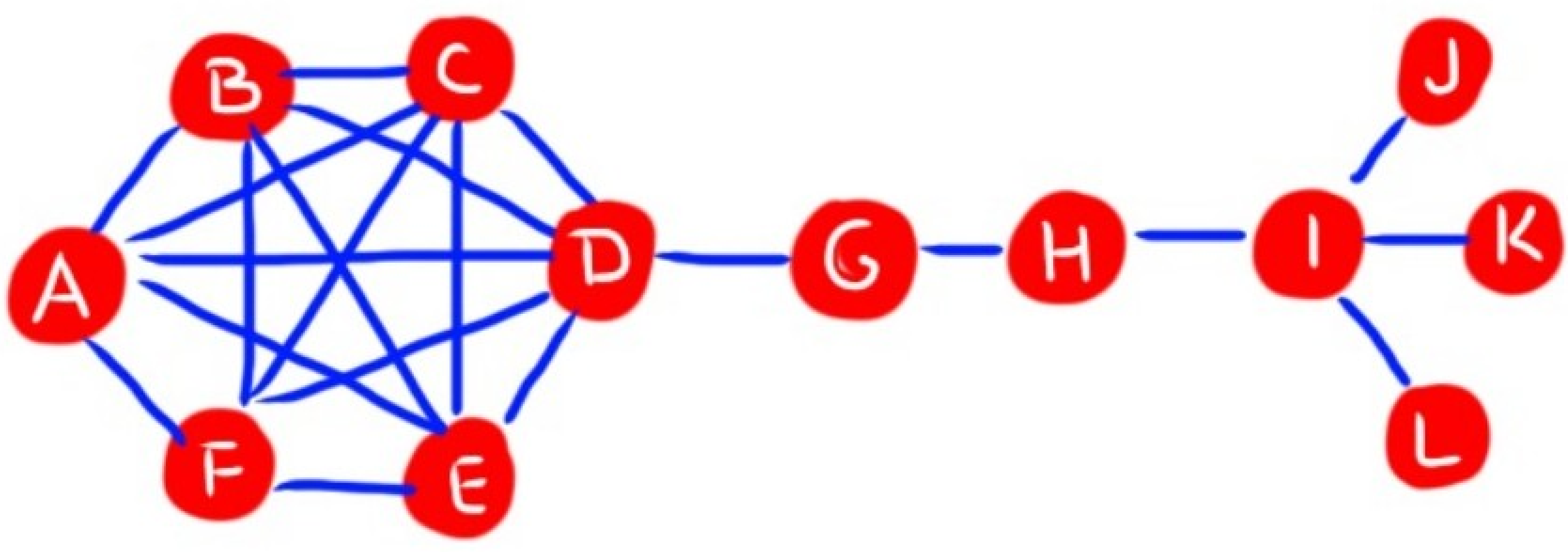
Betweenness: Question

- Find a node that has **high betweenness** but **low degree**



Betweenness: Question

- Find a node that has **low betweenness** but **high degree**

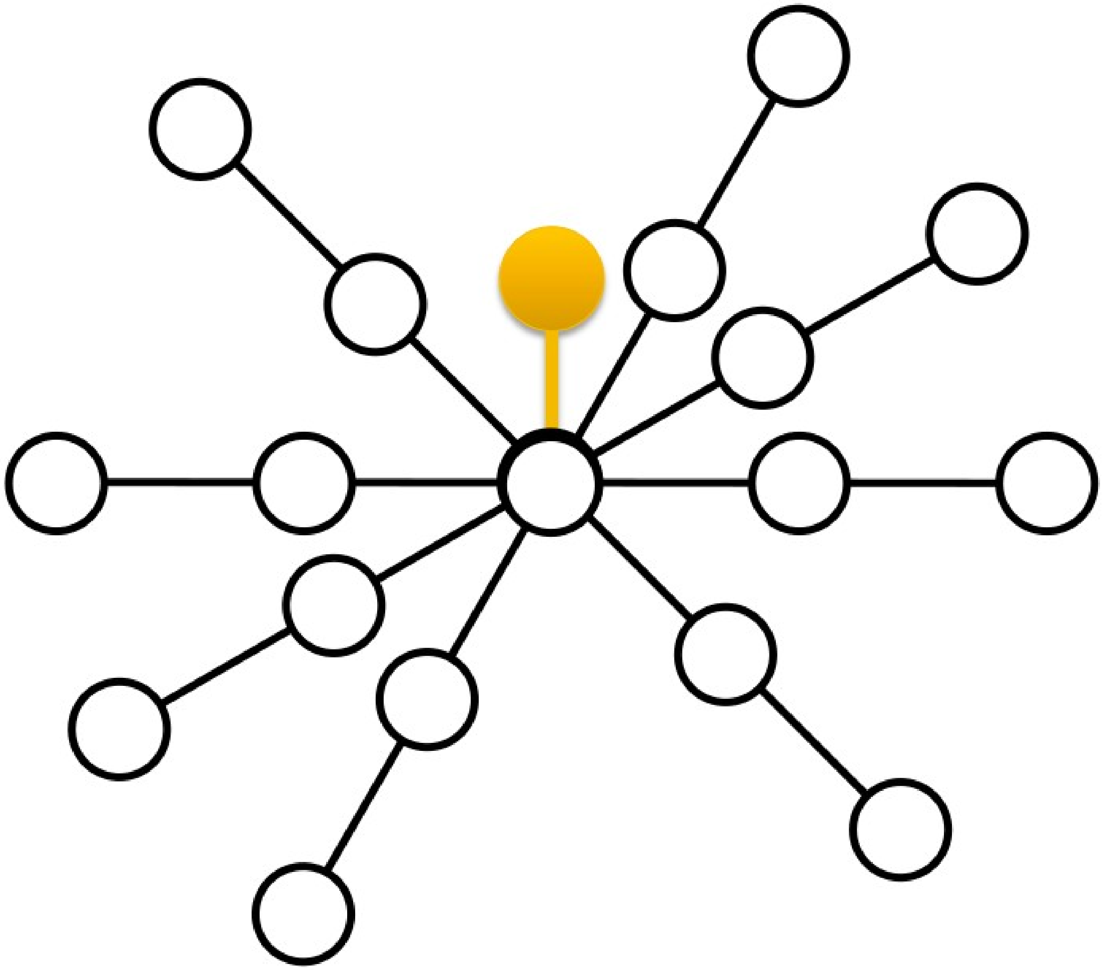


Closeness Centrality

- What if it's not so important to have many direct friends?
- Or be "between" others
- But one still wants to be in the **"middle"** of things, **not too far from the center**

Closeness Centrality

- Need not be in brokerage position



Closeness: Definition

- **Closeness** is based on the **length of the average shortest path** between a node and all other nodes in the network

Closeness Centrality:

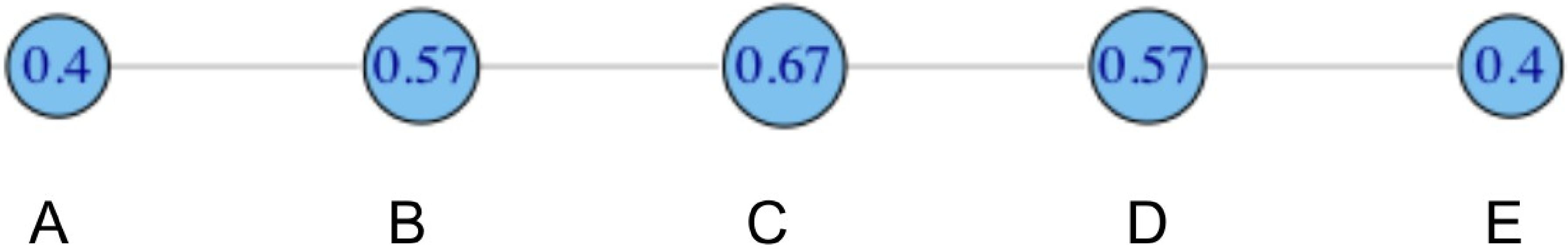
$$C_C(i) = \frac{1}{\sum_{j=1}^{N} d(i,j)}$$

Normalized Closeness Centrality:

$$C'_C(i) = C_C(i) \times (n-1)$$

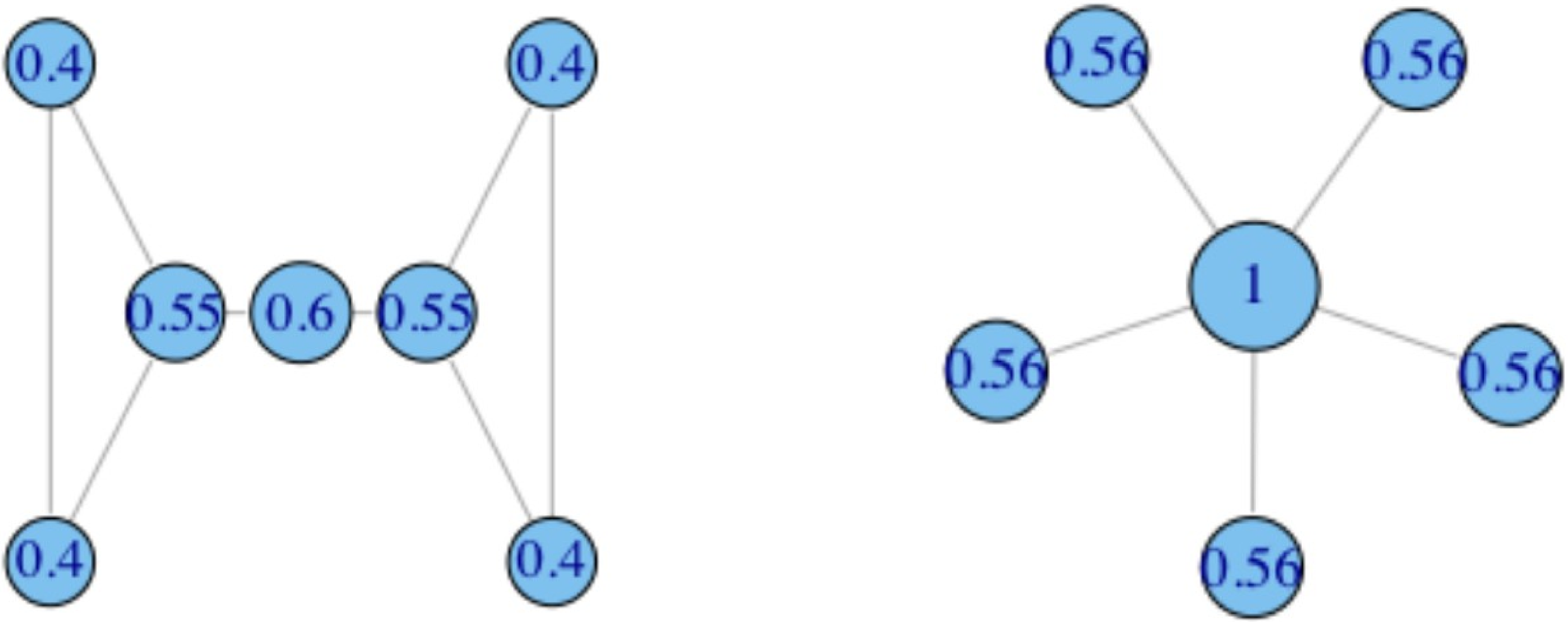
When graphs are big, the -1 can be discarded and we multiply by *n*

Closeness: Toy Networks



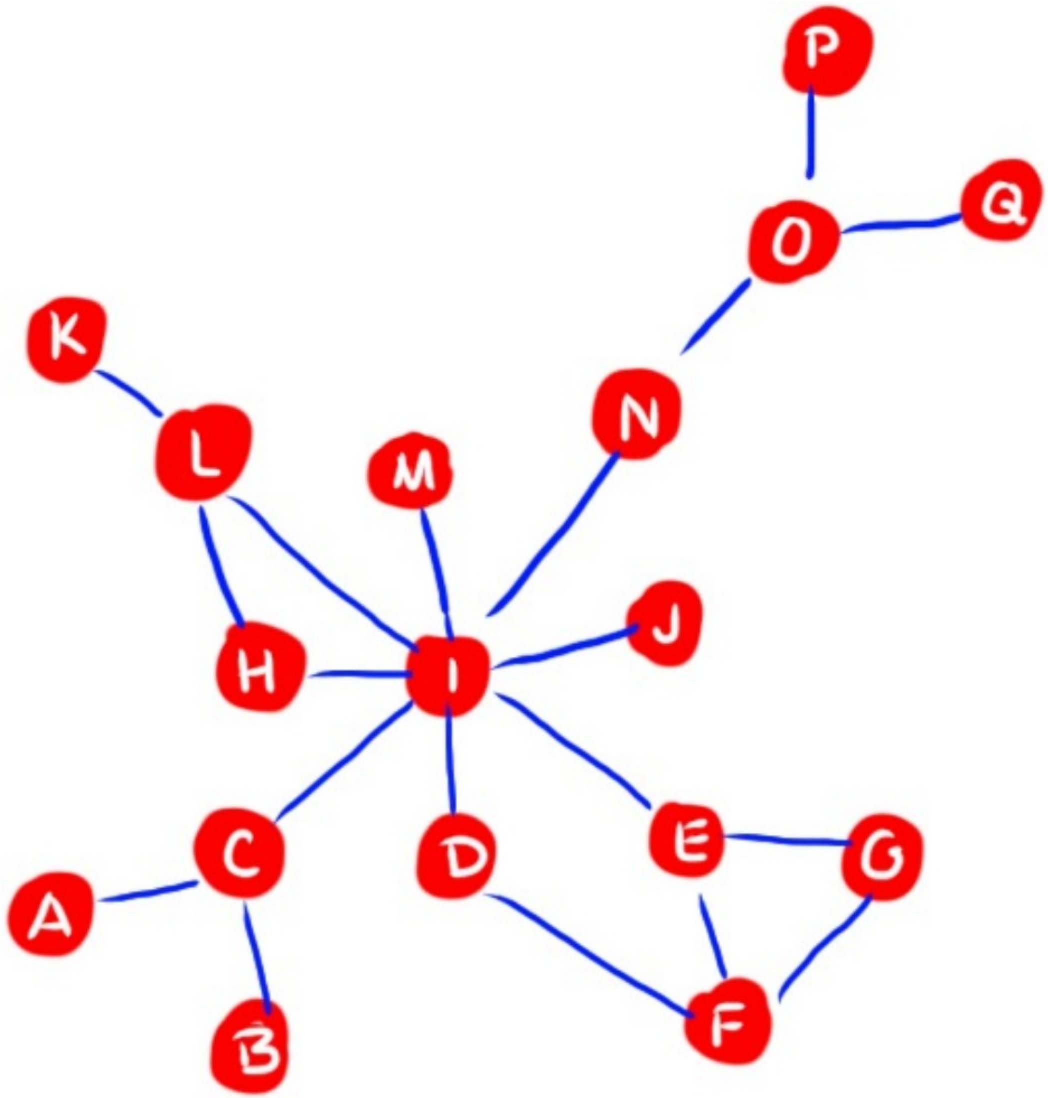
$$C_c^{'}(A) = \left[\frac{\sum_{j=1}^{N} d(A,j)}{N-1} \right]^{-1} = \left[\frac{1+2+3+4}{4} \right]^{-1} = \left[\frac{10}{4} \right]^{-1} = 0.4$$

Closeness: Toy Networks



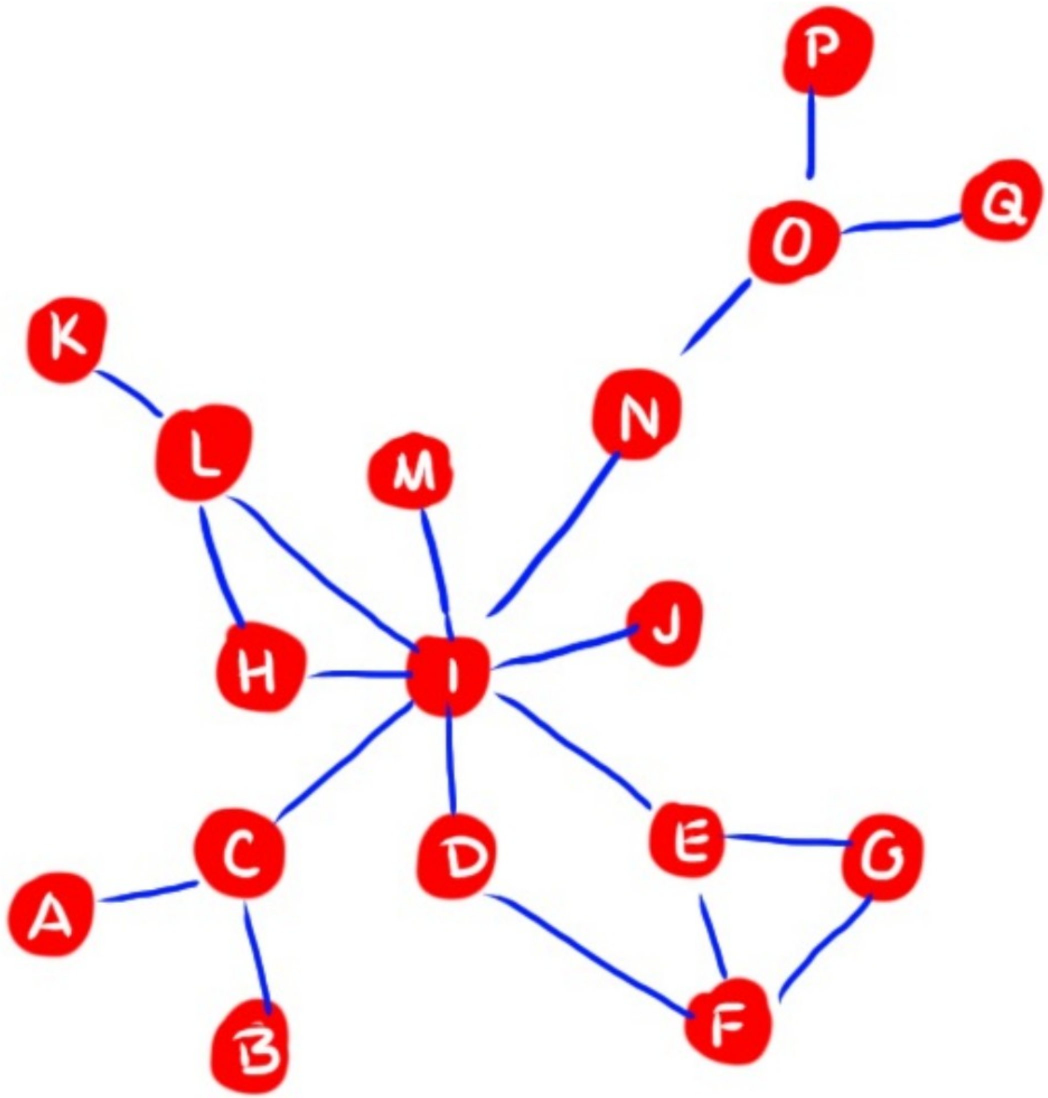
Closeness: Question

- Find a node which has relatively **high degree** but low **closeness**



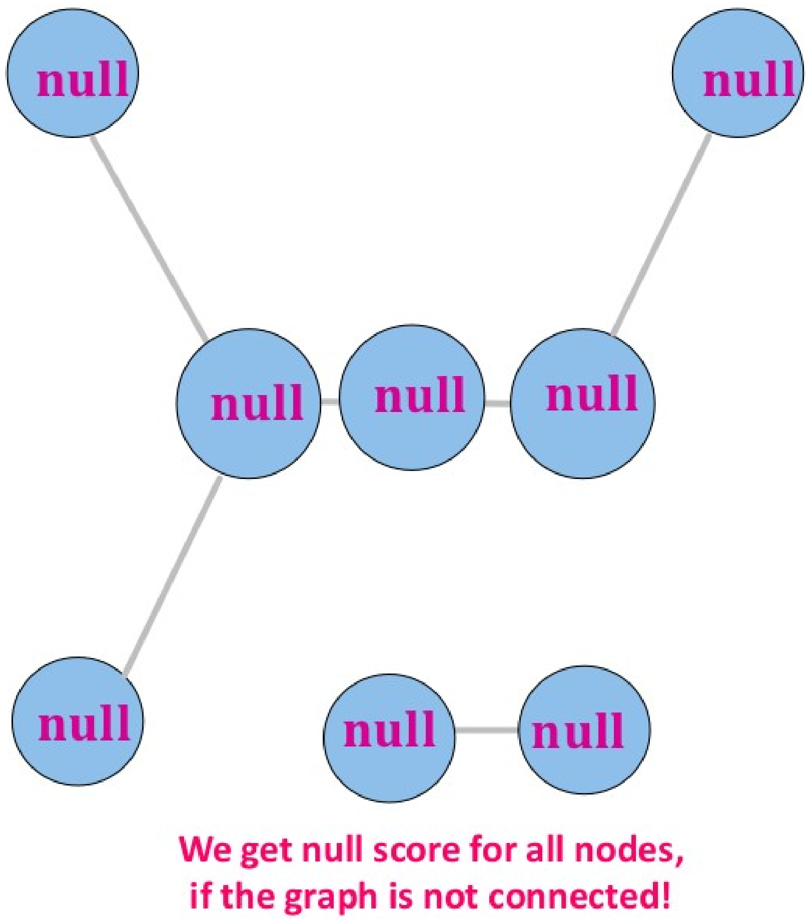
Closeness: Question

- Find a node which has **low degree** but **high closeness**



Closeness: unconnected graph

- What if the graph is **not connected**?



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

instead of *null*, we could also interpret it as 0 if *infinity* is the distance between unconnected nodes

Harmonic: Definition

- Replace the average distance with the **harmonic mean** of all distances

Harmonic Centrality:

$$C_H(i) = \sum_{j \neq i} \frac{1}{d(i,j)} = \sum_{d(i,j) < \infty,\, j \neq i} \frac{1}{d(i,j)}$$

- Strongly correlated to closeness centrality
- Naturally also accounts for nodes *j* that cannot reach *i*
- Can be applied to graphs that are not connected

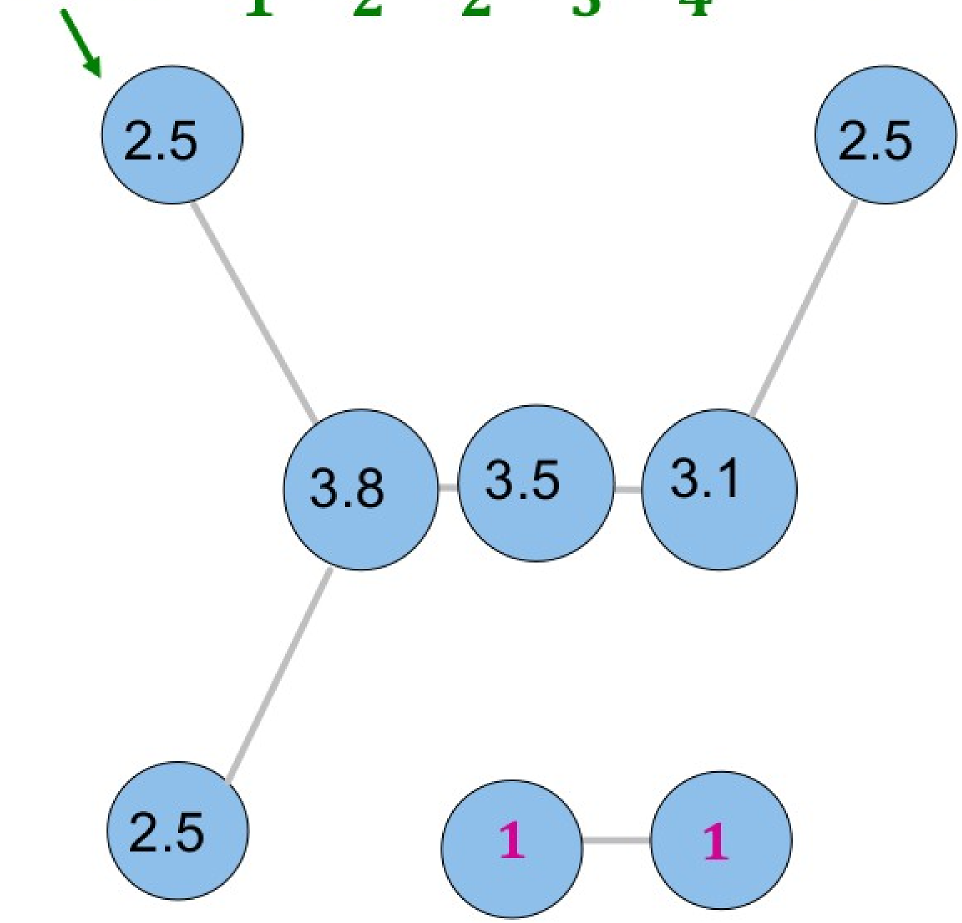
Normalized Harmonic Centrality:

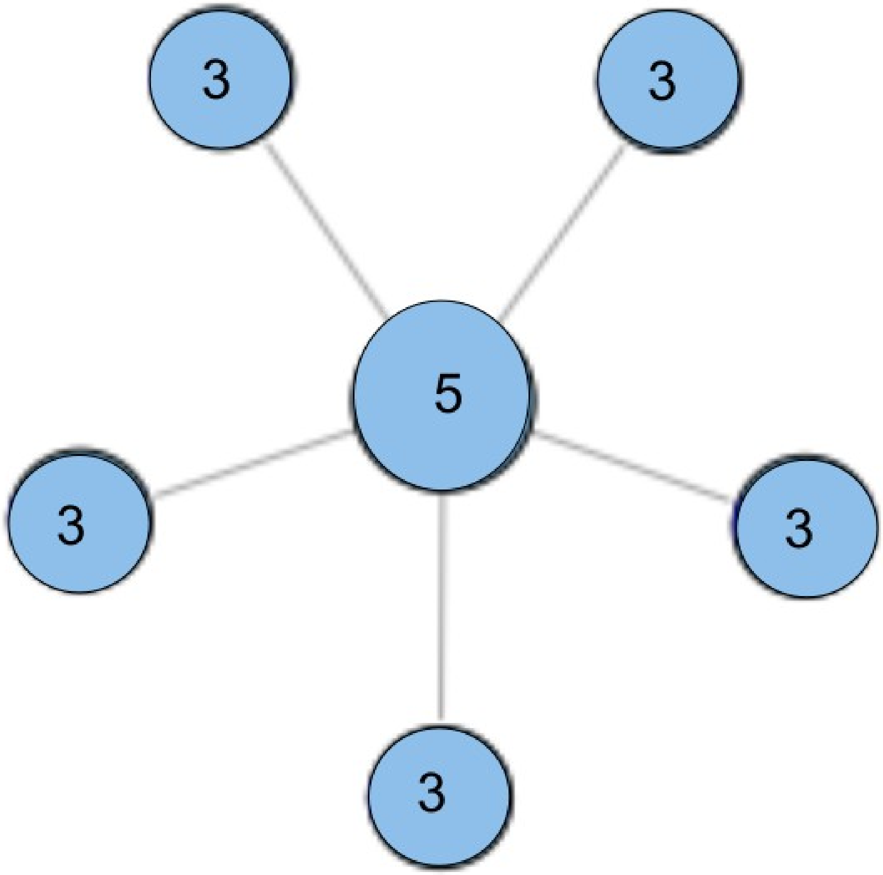
$$C'_H(i) = C_H(i) / (n-1)$$

Harmonic: Toy Networks

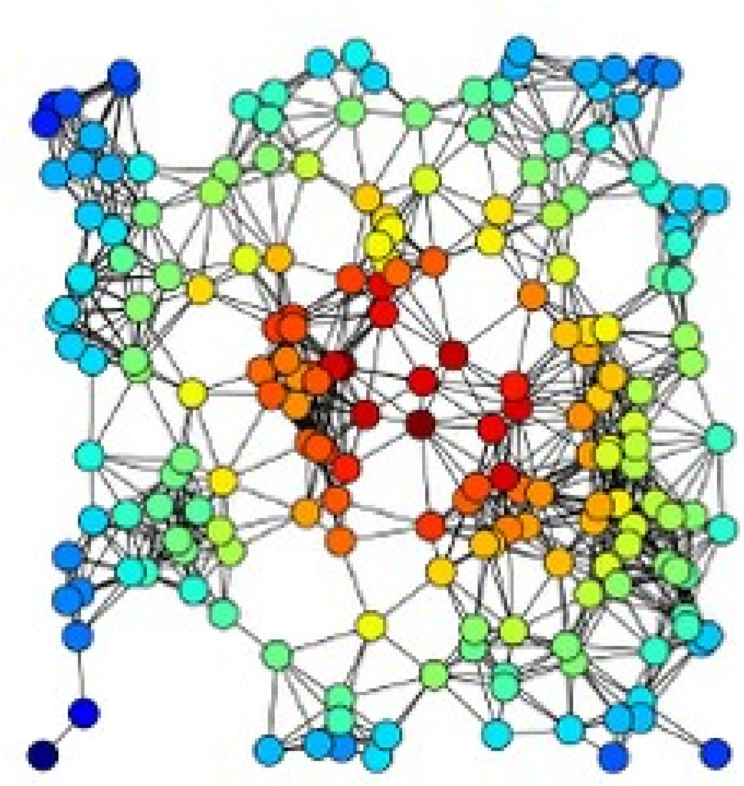
- Non-normalized version:

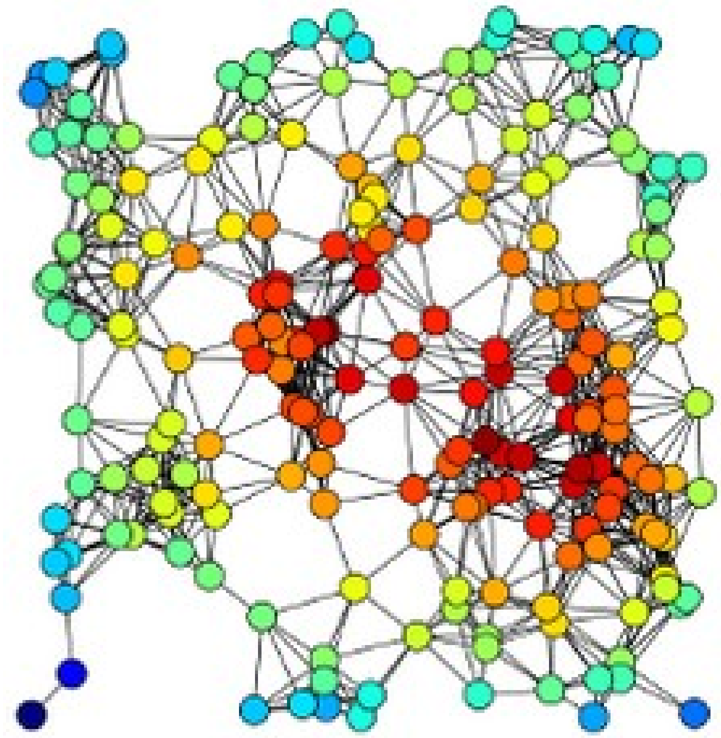
$$c_{harm} = \frac{1}{1} + \frac{1}{2} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} = 2.5$$





Closeness vs Harmonic





Closeness Centrality

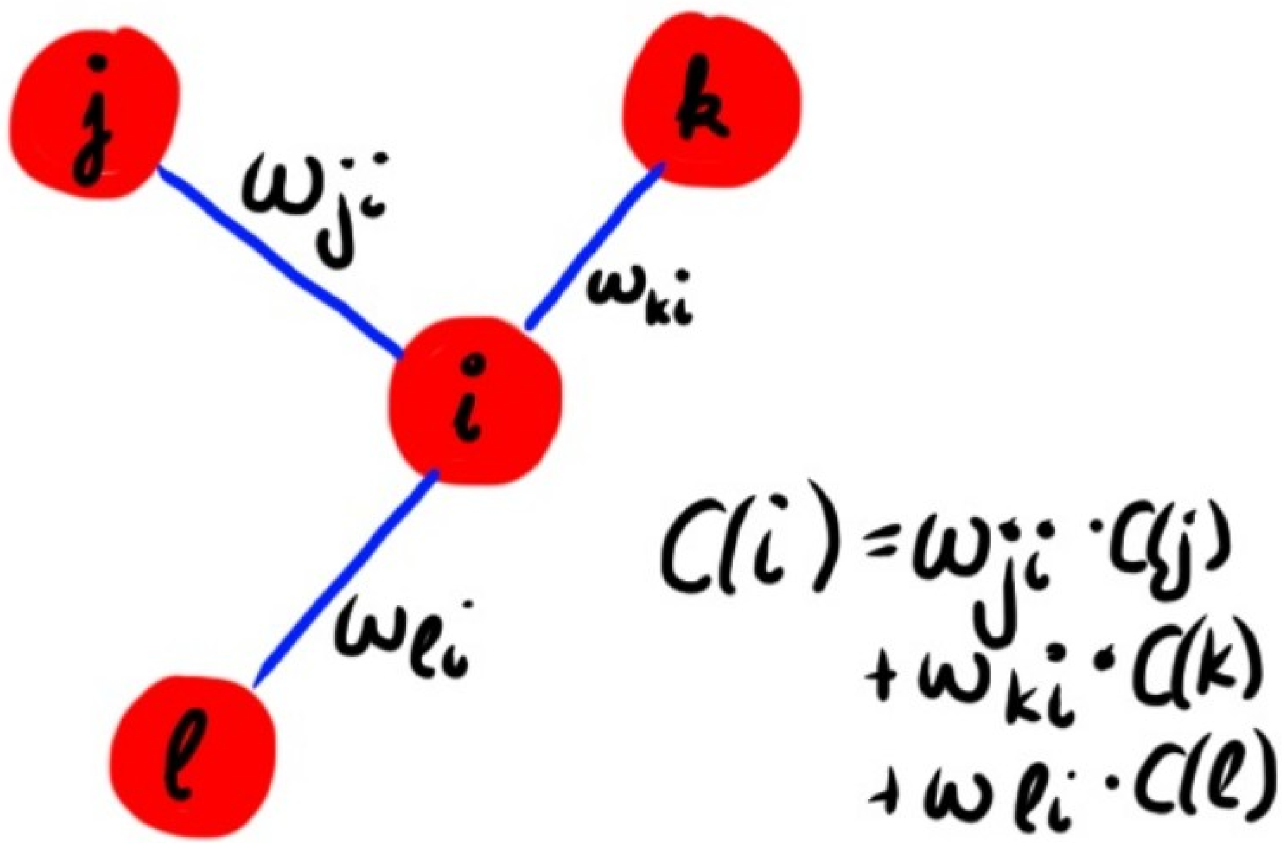
$$C_C(i) = \frac{1}{\sum_{j=1}^{N} d(i,j)}$$

Harmonic Centrality

$$C_H(i) = \sum_{j \neq i} \frac{1}{d(i,j)}$$

Eigenvector Centrality

- How “central” you are depends on how “central” your neighbors are



Eigenvector Centrality

Eigenvector Centrality:

$$C_E(i) = \frac{1}{\lambda} \sum_{j=1}^{n} A_{ji} \times C_E(j)$$

where λ is a constant and
A_{ij} the adjacency matrix (1 if *(i,j)* are connected, 0 otherwise)

(with a small rearrangement) this can we rewritten
in vector notation as in the eigenvector equation

$$Ax = \lambda x$$

where *x* is the eigenvector, and its *i*-th component is the centrality of node *i*

In general, there will be many different eigenvalues λ for which a non-zero eigenvector solution exists. However, the additional requirement that all the entries in the eigenvector be non-negative implies (by the Perron–Frobenius theorem) that only the greatest eigenvalue results in the desired centrality measure

Bonacich eigenvector centrality

also known as Bonacich Power Centrality

$$c_i(\beta) = \sum (\alpha + \beta c_j) A_{ji}$$

- α is a normalization constant
- β determines how important the centrality of your neighbors is
- $\mathbf{A}$ is the adjacency matrix (can be weighted)

Bonacich eigenvector centrality

also known as Bonacich Power Centrality

small β ➔ high attenuation

only your immediate friends matter, and their importance is factored in only a bit

high β ➔ low attenuation

global network structure matters (your friends, your friends' of friends etc.)

β = 0 yields simple degree centrality

$$c_i(\beta) = \sum_j (\alpha \quad \quad) A_{ji}$$

Eigenvector Variants

- There are other **variants** of eigenvector centrality, such as:
  - **PageRank**
    - (normalized eigen vector + random jumps) [we will talk in detail about that later]
  - **Katz Centrality**
    - (connections with distant neighbors are penalized)

$$C_{\mathrm{Katz}}(i) = \sum_{k=1}^{\infty} \sum_{j=1}^{n} \alpha^k (A^k)_{ji}$$

Centrality in Directed Networks

- **Degree:**
  - in and out centrality
- **Betweenness:**
  - Consider only directed paths: $C_B(i) = \sum_{j \neq k} \frac{g_{jk}(i)}{g_{jk}}$
  - When normalizing take care of ordered pairs

$$C'_B(i) = \frac{C_B(i)}{(n-1)(n-2)}$$

number of ordered pairs is 2x the number of unordered

- **Closeness**
  - Consider only directed paths
- **Eigenvector** (already prepared)

Centrality in Weighted Networks

- **Degree:**
  - Sum weights *(non-weighted equals weight=1 for all edges)*
- **Betweenness and Closeness:**
  - Consider weighted distance
- **Eigenvector**
  - Consider weighted adjacency matrix

Node Centralities: Conclusion

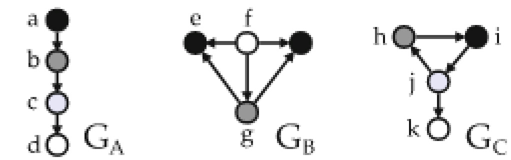
- There are other node centrality metrics, but these are the **"quintessential"**

Finding Dominant Nodes Using Graphlets

David Aparício[✉], Pedro Ribeiro, Fernando Silva, and Jorge Silva

CRACS & INESC-TEC and the Department of Computer Science, Faculty of Sciences, University of Porto, 4169-007 Porto, Portugal
{daparicio,pribeiro,fds}@dcc.fc.up.pt, jorge.m.silva@inesctec.pt

$$D(o) = \left(\lambda \times \sum_{o_i \in \mathcal{I}(o)} \beta^{k-d(o,o_i)}\right) - \left((1-\lambda) \times \sum_{o_j \in \mathcal{S}(o)} \beta^{k-d(o_j,o)}\right)$$



	a	b	c	d
a	-	1	2	3
b	-	-	1	2
c	-	-	-	1
d	-	-	-	-

	e	e	f	g
e	-	-	-	-
e	-	-	-	-
f	1	1	-	1
g	1	1	-	-

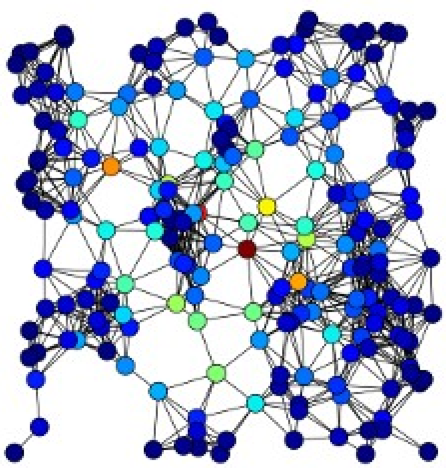
	h	i	j	k
h	-	1	2	3
i	2	-	1	2
j	1	2	-	1
k	-	-	-	-

A subgraph-based ranking system for professional tennis players

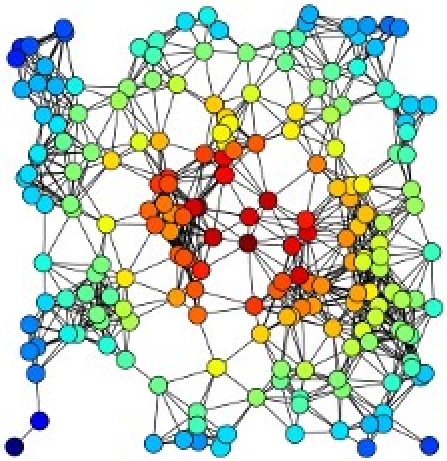
David Aparício, Pedro Ribeiro and Fernando Silva

- Which one to use depends on **what you want to achieve or measure**
  - Worry about understanding the concepts
  - They enlarge your graph vocabulary

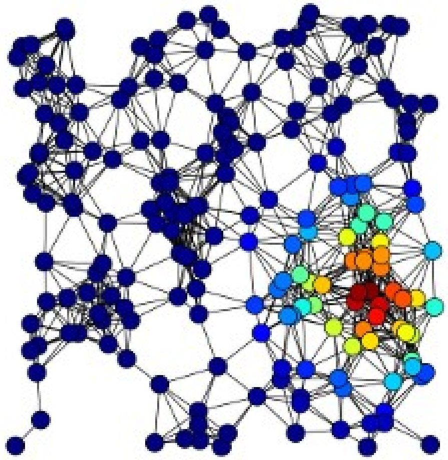
Node Centralities: Conclusion



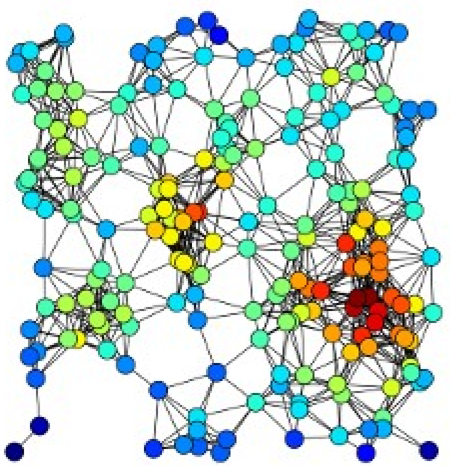
Betweenness



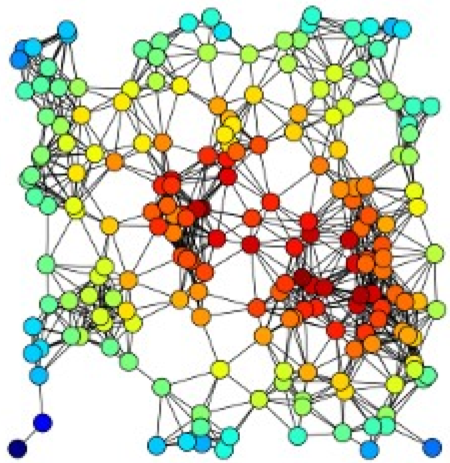
Closeness



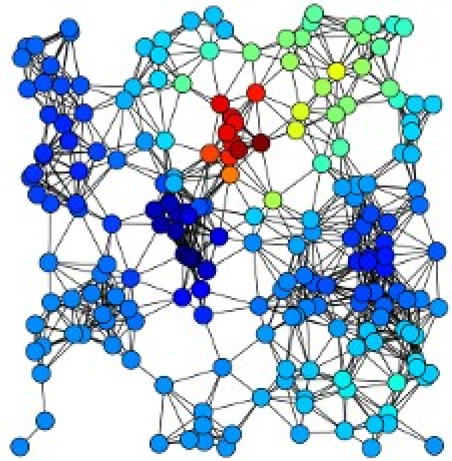
Eigenvector



Degree



Harmonic



Katz

Node Centralities: Conclusion

- All (major) network analysis packages provide them:

neo4j

The #1 Database for Connected Data

Centrality algorithms are used to determi
includes the following centrality algorithn

- Production-quality
  - Page Rank
  - Betweenness Centrality
- Alpha
  - ArticleRank
  - Closeness Centrality
  - Harmonic Centrality
  - Degree Centrality
  - Eigenvector Centrality
  - HITS

NetworkX
Network Analysis in Python

Centrality

Degree

`degree_centrality` (G)	Compute the degree centrality for nodes.
`in_degree_centrality` (G)	Compute the in-degree centrality for nodes.
`out_degree_centrality` (G)	Compute the out-degree centrality for nodes.

Eigenvector

`eigenvector_centrality` (G[, max_iter, tol, ...])	Compute the eigenvector centrality for the graph `G`.
`eigenvector_centrality_numpy` (G[, weight, ...])	Compute the eigenvector centrality for the graph G.
`katz_centrality` (G[, alpha, beta, max_iter, ...])	Compute the Katz centrality for the nodes of the graph G
`katz_centrality_numpy` (G[, alpha, beta, ...])	Compute the Katz centrality for the graph G.

Closeness

`closeness_centrality` (G[, u, distance, ...])	Compute closeness centrality for nodes.
`incremental_closeness_centrality` (G, edge[, ...])	Incremental closeness centrality for nodes.

Current Flow Closeness

`current_flow_closeness_centrality` (G[, ...])	Compute current-flow closeness centrality for nodes.
`information_centrality` (G[, weight, dtype, ...])	Compute current-flow closeness centrality for nodes.

(Shortest Path) Betweenness

`betweenness_centrality` (G[, k, normalized, ...])	Compute the shortest-path betweenness centrality for r

igraph

8. Centrality Measures

8.1. `igraph_closeness` — Closeness centrality calculations for some vertices.
8.2. `igraph_harmonic_centrality` — Harmonic centrality for some vertices.
8.3. `igraph_betweenness` — Betweenness centrality of some vertices.
8.4. `igraph_edge_betweenness` — Betweenness centrality of the edges.
8.5. `igraph_pagerank_algo_t` — PageRank algorithm implementation
8.6. `igraph_pagerank` — Calculates the Google PageRank for the specified vertices.
8.7. `igraph_personalized_pagerank` — Calculates the personalized Google PageRank for the specified vertices.
8.8. `igraph_personalized_pagerank_vs` — Calculates the personalized Google PageRank for the specified vertices.
8.9. `igraph_constraint` — Burt's constraint scores.
8.10. `igraph_maxdegree` — The maximum degree in a graph (or set of vertices).
8.11. `igraph_strength` — Strength of the vertices, weighted vertex degree in other words.
8.12. `igraph_eigenvector_centrality` — Eigenvector centrality of the vertices

- Also all (major) network analysis and visualization platforms:

Gephi makes graphs handy

Cytoscape