Network Science Summer School

Universiteit Utrecht

Day program

10:00–12:00:

Introduction to network science

12:00-13:00

Lunch

13:00–16:30:

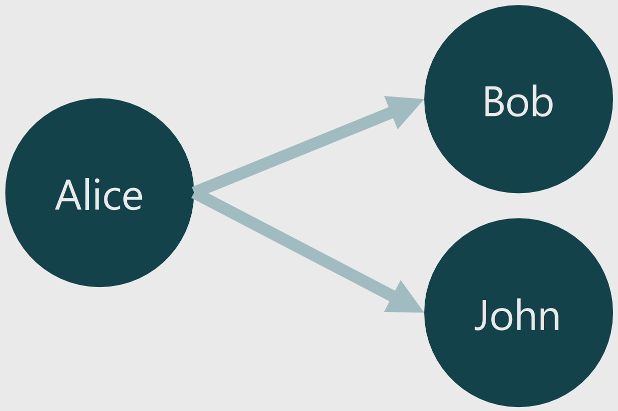
Network representation

Centrality

Intro to linear algebra

Why? Most analysis on networks rely on matrix multiplication

Network representation



Adjacency list: (edgelist)

- Adv: It is dense: Only keeping edges
- Disadvantage: Hard to work with

Source	Target	Weigth
Alice	Bob	1
Alice	John	1

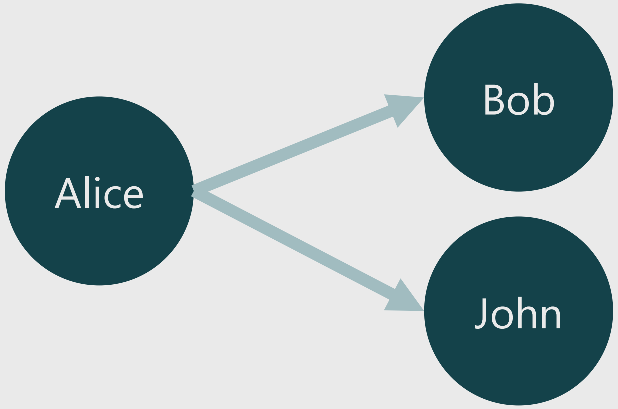
Adjacency matrix:

- Adv: Linear algebra is easy
- Disadvantage: It is sparse (mostly zeros).1E6 nodes → 1 trillion options

Target → ↓ Source	Alice	Bob	John
Alice	0	1	1
Bob	0	0	0
John	0	0	0

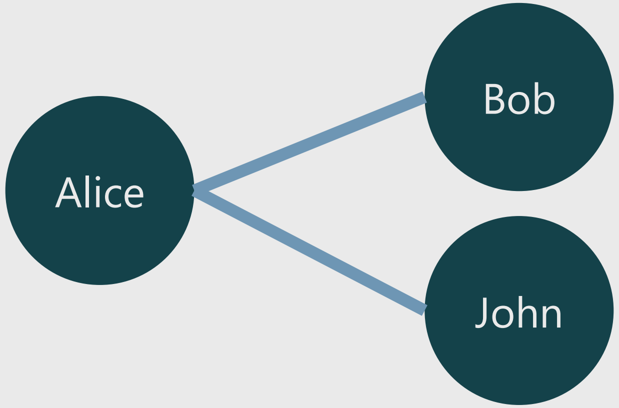
In computers → Sparse matrices: Best of both worlds

Directed networks



Target → ↓ Source	Alice	Bob	John
Alice	0	1	1
Bob	0	0	0
John	0	0	0

Undirected networks



Target → ↓ Source	Alice	Bob	John
Alice	0	1	1
Bob	1	0	0
John	1	0	0

Some terms

A =

Target → ↓ Source	Alice	Bob	John
Alice	0	1	1
Bob	0	0	0
John	0	0	0

Diagonal

Trace = Sum of elements in the diagonal

Identity matrix (I) =

	1	0	0
	0	1	0
	0	0	1

I @ A = A

Transpose (A^T, A') =

(python) A.T

Target → ↓ Source	Alice	Bob	John
Alice	0	0	0
Bob	1	0	0
John	1	0	0

Symmetric matrix: A = A.T (e.g. undirected network)

Python exercise notebook 2, ex.1

Python:

- Convert between formats
- Plot matrix

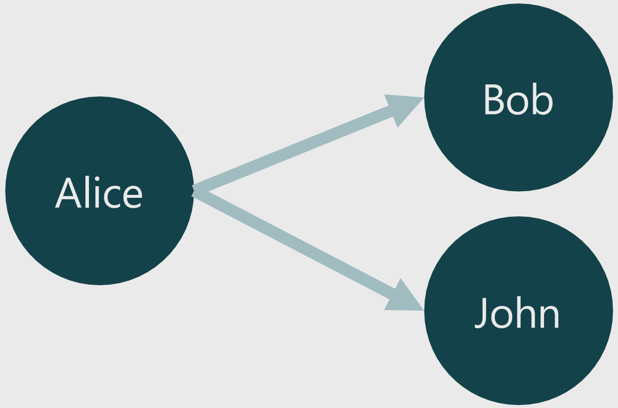
Transposing = reversing the edges

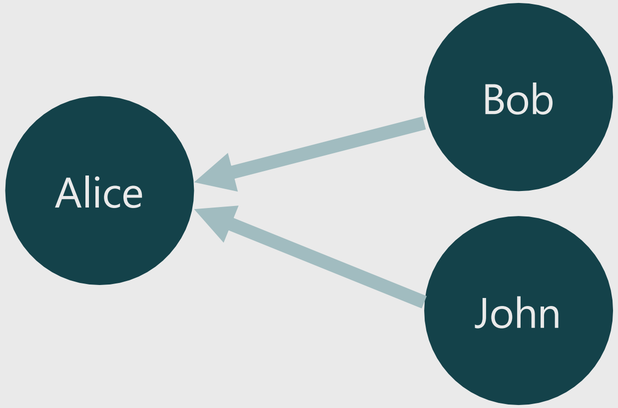
A =

Target → ↓ Source	Alice	Bob	John
Alice	0	1	1
Bob	0	0	0
John	0	0	0

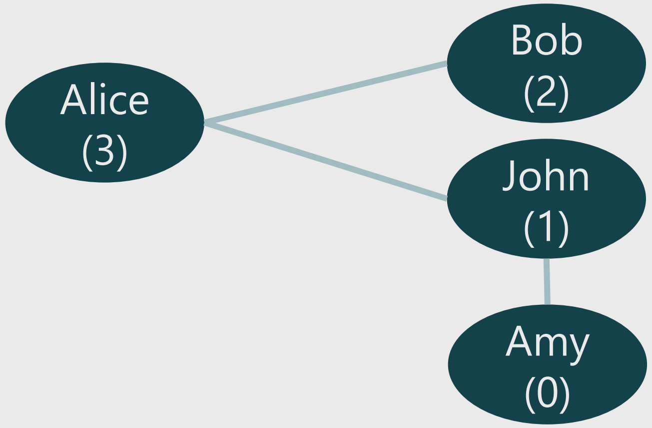
A.T =

Target → ↓ Source	Alice	Bob	John
Alice	0	0	0
Bob	1	0	0
John	1	0	0





Matrix multiplication: sum



Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	1	0	0	0
John	1	0	0	1
Amy	0	0	1	0

@

Node	kids
Alice	3
Bob	2
John	1
Amy	0

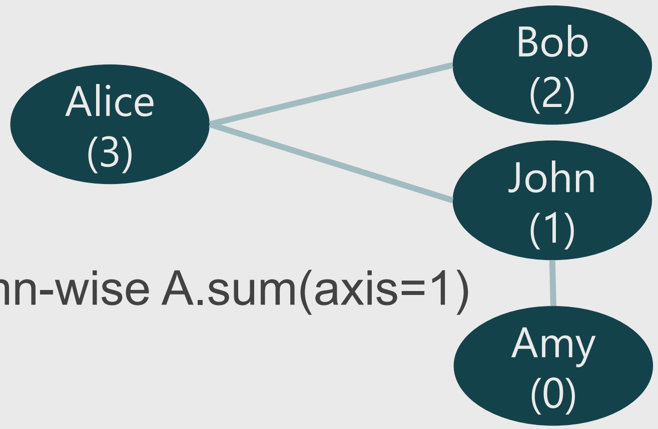
=

Node	kids
Alice	0*3 + 1*2 + 1*1 + 0*0 = 3
Bob	1*3 + 0*2 + 0*1 + 0*0 = 3
John	1*3 + 0*2 + 0*1 + 1*0 = 3
Amy	0*3 + 0*2 + 1*1 + 0*0 = 1

A @ M = SM

(N x N) @ (N x 1) = (N x 1)

Matrix multiplication: average



Divide by the degree. We get it by summing the adjacency elements column-wise A.sum(axis=1)

A @ M / A.sum(1)

(N x N) @ (N x 1) / (N x 1) = (N x 1) / (N x 1) = (N x 1)

Target → ↓ Origin	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	1	0	0	0
John	1	0	0	1
Amy	0	0	1	0

@

Node	Kids
Alice	3
Bob	2
John	1
Amy	0

divided by

Target → ↓ Source	Sum
Alice	2
Bob	1
John	2
Amy	1

=

Node	Kids
Alice	3
Bob	3
John	3
Amy	1

divided by

Target → ↓ Source	Sum
Alice	2
Bob	1
John	2
Amy	1

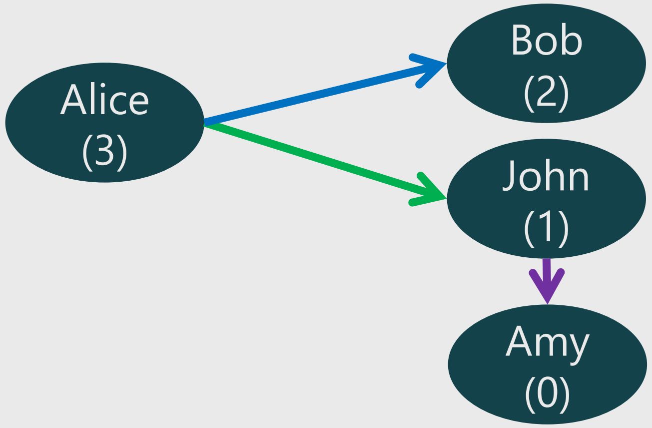
=

Node	Kids
Alice	1.5
Bob	3
John	1.5
Amy	1

Python exercise notebook 2, ex.2

Calculate the average number of children of your friends using matrix multiplication

Matrix multiplication: paths



Interpretation A: Presence of path between node i and j

A^2 =

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	0	0	0	0
John	0	0	0	1
Amy	0	0	0	0

@

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	0	0	0	0
John	0	0	0	1
Amy	0	0	0	0

=

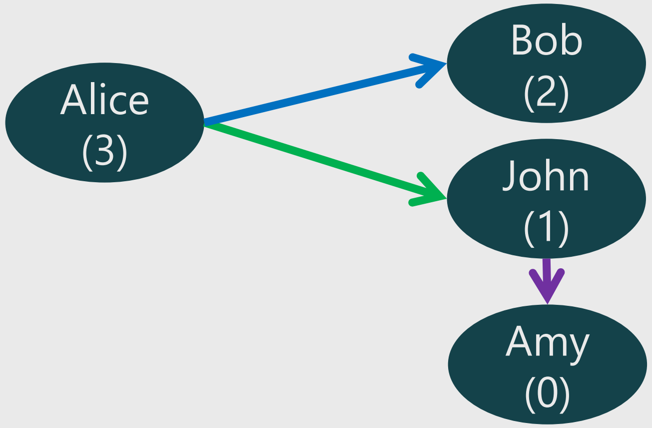
Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	0	0	1
Bob	0	0	0	0
John	0	0	0	0
Amy	0	0	0	0

From you → to your neighbors

From your neighbors → to their neighbors

You → the neig. of your neig.

Matrix multiplication: paths



Interpretation A: Presence of path between node i and j
Interpretation A^2: Number of path between node i and j in two steps
Interpretation A^3: Number of path between node i and j in three steps
…

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	0	0	0	0
John	0	0	0	1
Amy	0	0	0	0

@

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	0	0	0	0
John	0	0	0	1
Amy	0	0	0	0

=

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	0	0	1
Bob	0	0	0	0
John	0	0	0	0
Amy	0	0	0	0

Alice → Alice (0) * Alice → Amy (0)
+ Alice → Bob (1) * Bob → Amy (0)
+ Alice → John (1) * John → Amy (1)
+ Alice → Alice (0) * Alice → Amy (1)

Matrix multiplication: number of people reached in <3 steps

Number of paths in two or three steps from node i to node j: $\boldsymbol{N = A + A^2 + A^3}$
We need to remove duplicate paths: ***N = N>0***
We need to remove paths from us to ourselves ***N.setdiag(0)***

Python exercise notebook 2, ex.3a

Matrix multiplication: number of triangles

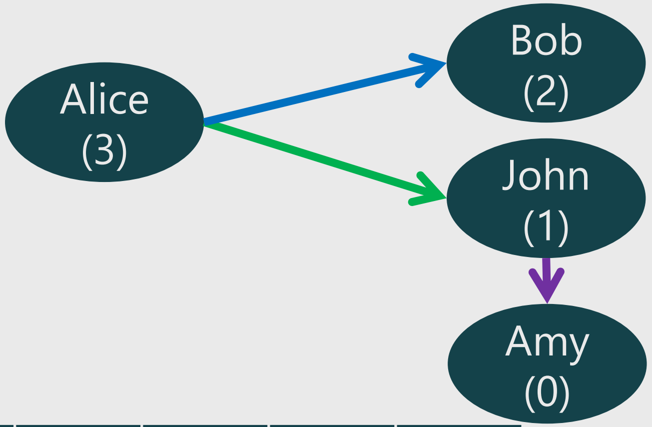
We are interested in the diagonal of ***A^3***

A triangle is a path in from node *i* to node *i.*

The diagonal contains the number of triangles in which node *I* appears.

Undirected network? Divide the triangles by two (two directions)
Counting the total number of triangles? Divide the trace by 3 (each triangle has 3 members)

Matrix multiplication: number of triangles



A^2

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	0	0	1
Bob	0	0	0	0
John	0	0	0	0
Amy	0	0	0	0

@

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	0	0	0	0
John	0	0	0	1
Amy	0	0	0	0

=

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	0	0	0
Bob	0	0	0	0
John	0	0	0	0
Amy	0	0	0	0

Alice → Alice in two steps * Alice → Alice (0)
Alice → Bob in two steps * Bob -> Alice (0)
Alice → John in two steps * John -> Alice (0)
Alice → Amy in two steps * Amy -> Alice (0)

Diagonal of A^3

Alice → X_1 * X_1 → X_1 * X_1 → Alice +
Alice → X_1 * X_1 → X_2 * X_2 → Alice +
...

Python exercise notebook 2, ex.3b

Centrality measures

Nice explanations:

https://aksakalli.github.io/2017/07/17/network-centrality-measures-and-their-visualization.html

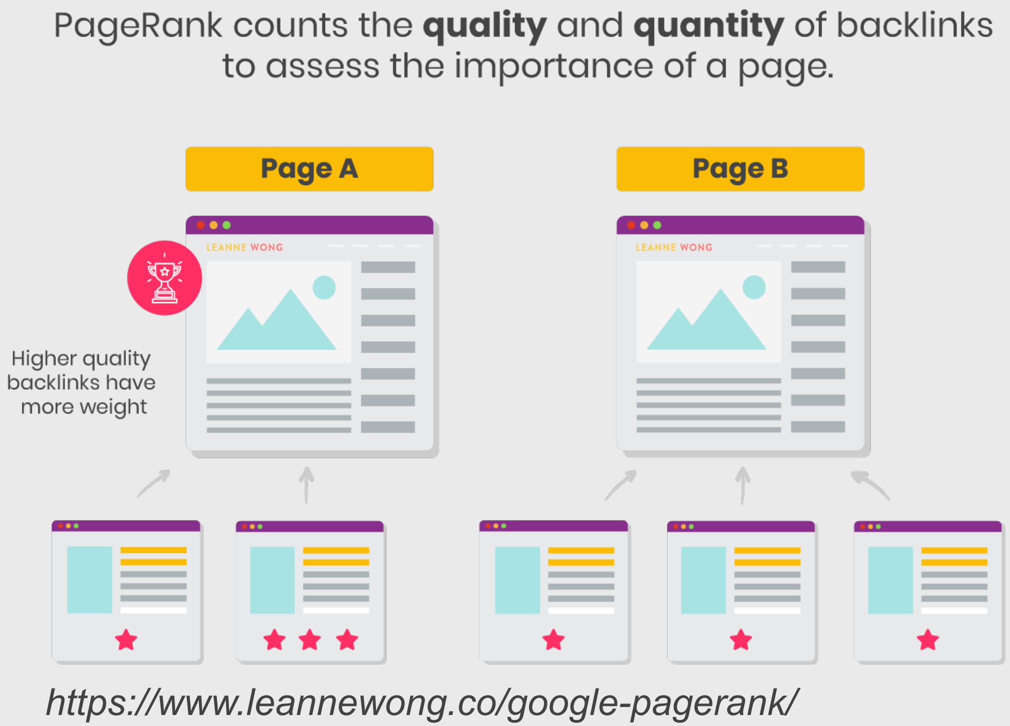
Networks: an introduction (Newman)

Motivating examples

How to stop the spread of diseases?

Important nodes: bridges

How to sort Google results?



Important nodes: those linked by important nodes

Centrality

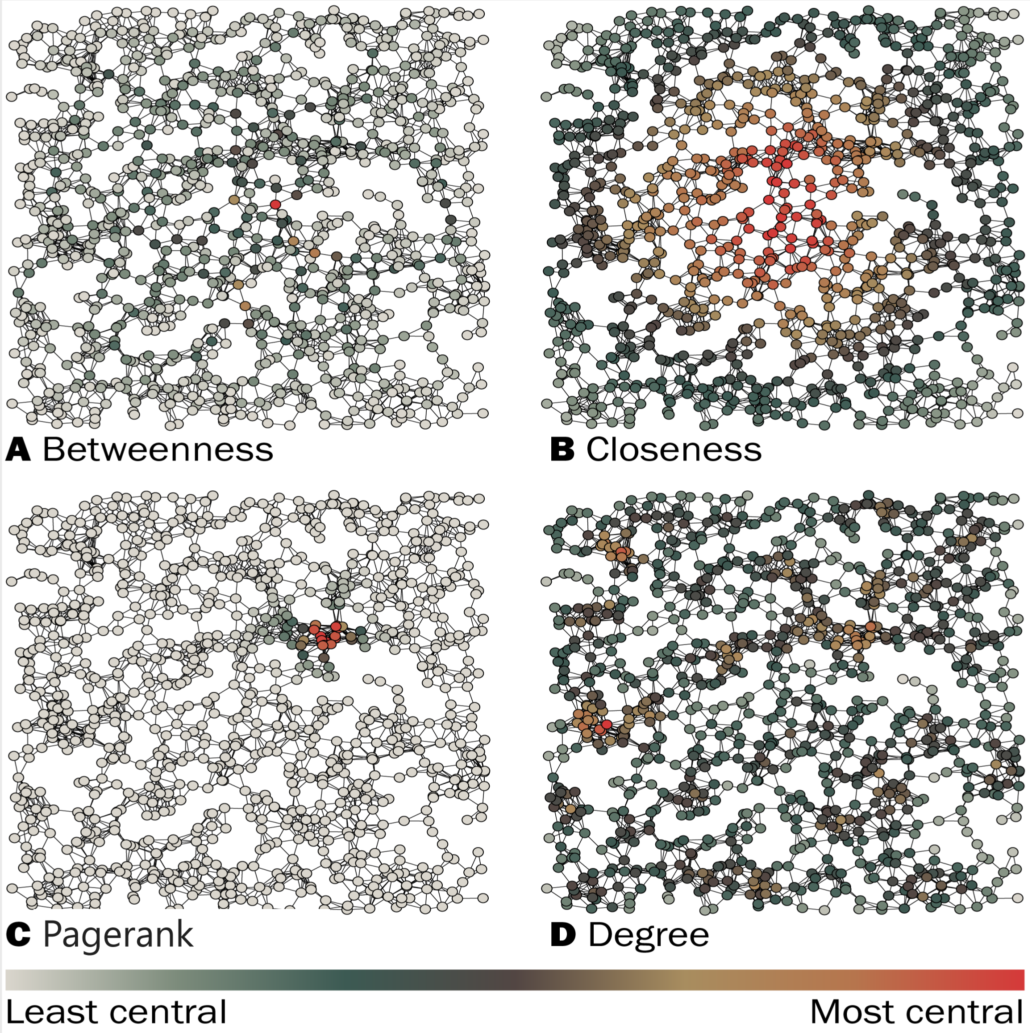
Who are the key actors in the network?

Centrality measures provide answers to this question.

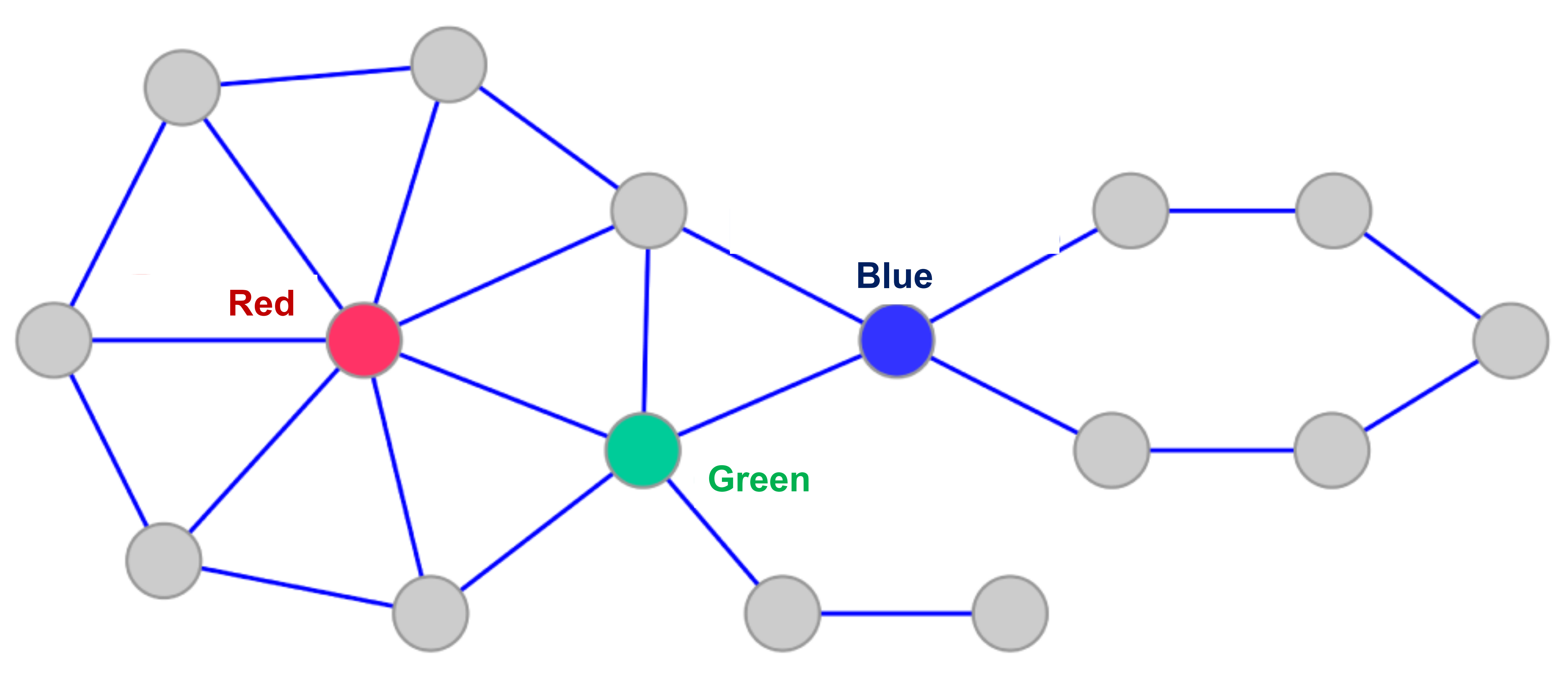
Different centrality measures define importance in different ways :

- *Degree*: Connected to many nodes
- *Closeness*: Close to all other nodes
- *Betweenness*: In the middle of shortest paths
- *Pagerank*: Connected to important nodes

Centrality identify *the most important nodes.* It does not quantify the importance of nodes in general. The relative rankings of non-important nodes may be meaningless.



Which node has higher degree/betweenness/closeness?



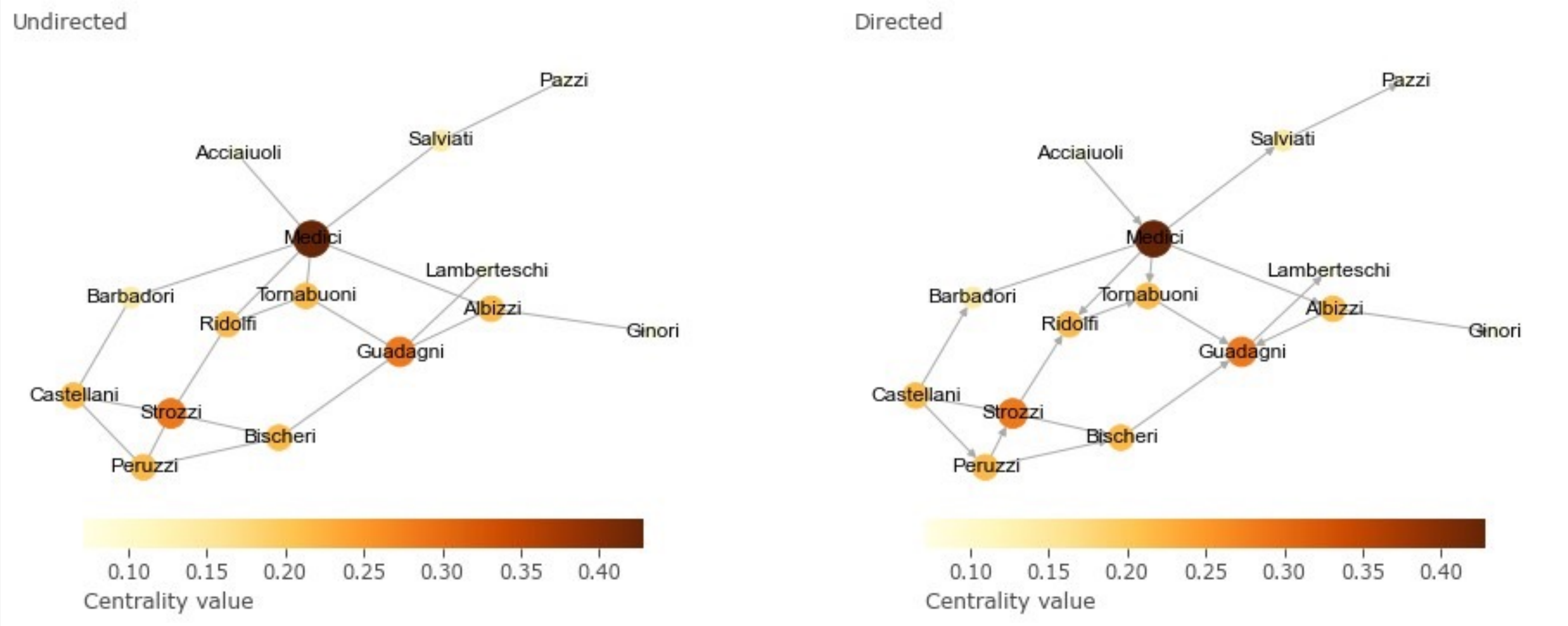
https://cytoscape.org/cytoscape-tutorials/presentations/modules/network-analysis/index.html

Degree centrality = $^{d_i}/_{(N-1)}$

d_i = degree of node i

$N - 1$ = number of nodes - 1 (max. potential number of partners without self-edges)

Measures the **local** influence of the node



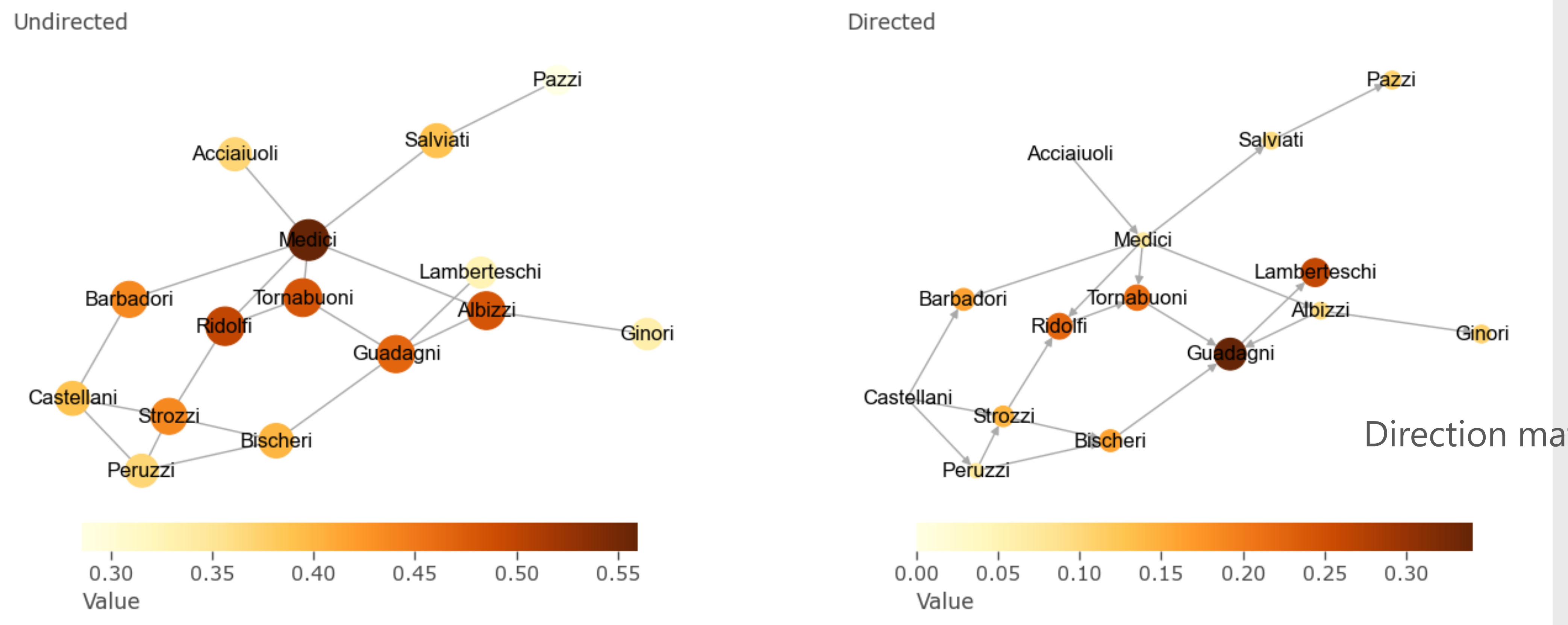
Closeness centrality = $^1/_{l_i}$

l_i = average distance of node i to all other nodes := $\boldsymbol{l_i = \frac{1}{N}\sum_j d_{ij}}$

$\boldsymbol{d_{ij}}$ = shortest distance from node i to node j

Only useful in fully connected networks

Measures the **most central** node in the network (closest to get to all other nodes)



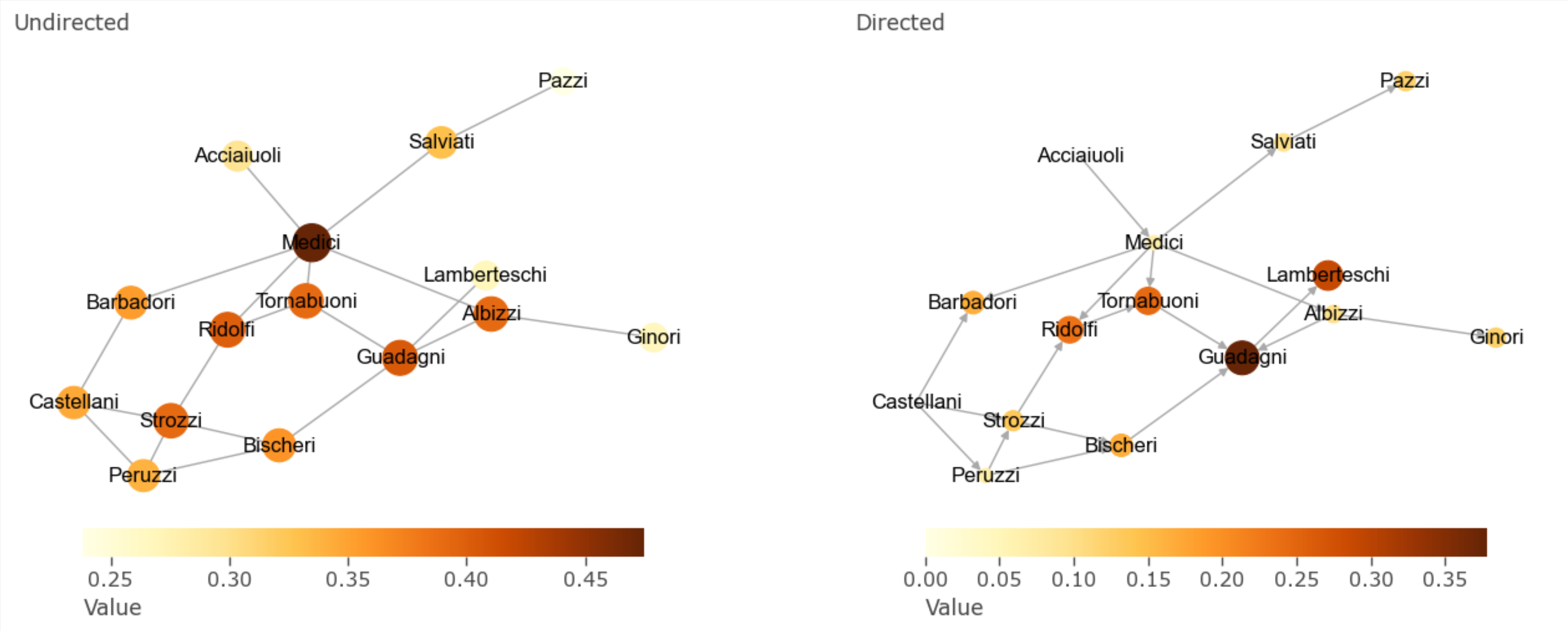
Direction matters!

Harmonic closeness centrality = $\frac{1}{(N-1)}\sum_{ij|i\neq j} \frac{1}{d_{ij}}$

d_{ij} = shortest distance from node i to node j

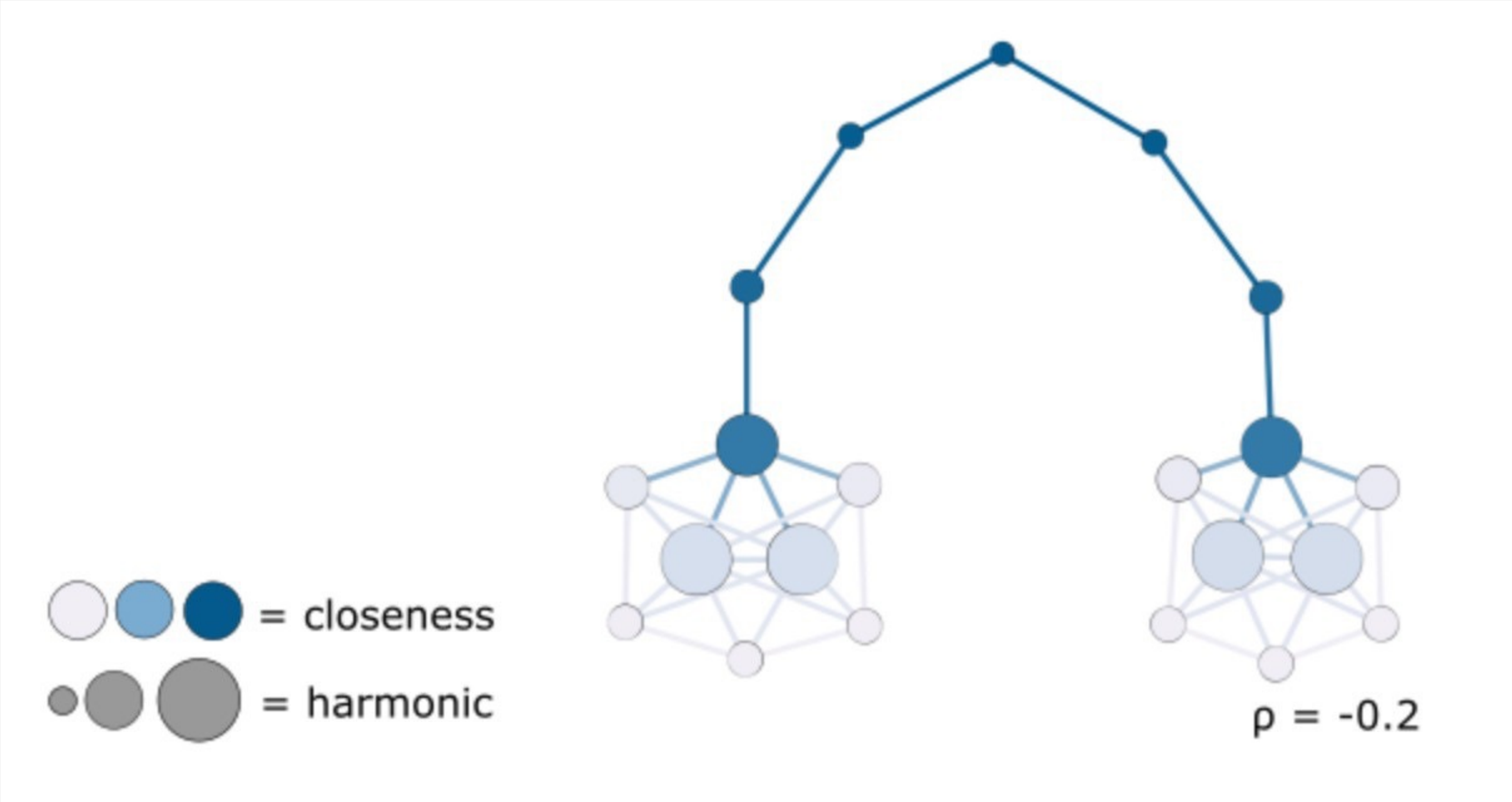
Useful also in disconnected networks. Gives more weight to closer nodes.

Measures the **most central** node in the network (harmonic average)



Bavelas, 1950

Closeness vs harmonic closeness



https://symbio6.nl/en/blog/analysis/harmonic-centrality

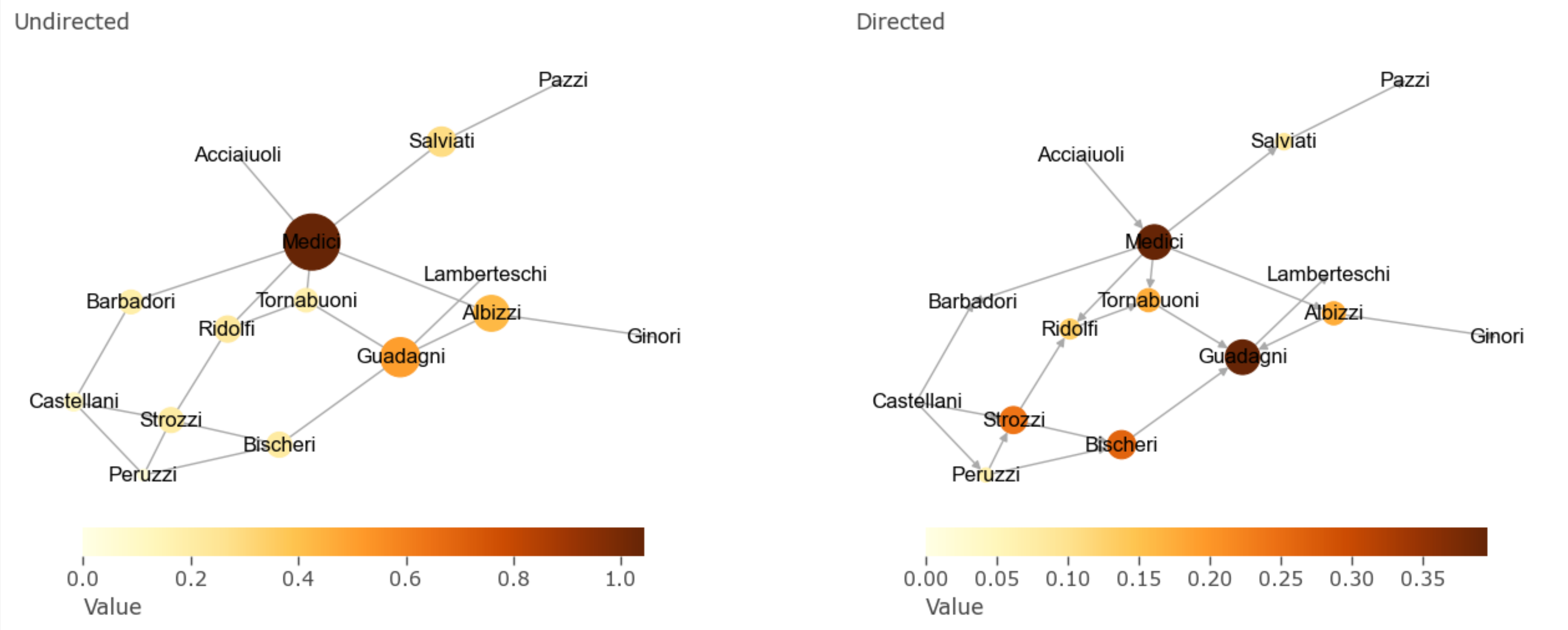
Betweeness centrality = $^1/_{n^2} \sum_{st} n^i_{st}$

n^i_{st} = 1/g if node *i* lies on the *g* shortest paths between nodes *s* and *t*

Assumptions:

- every pair of nodes in the network exchanges messages at the same average rate
- messages always take the shortest available path though the network

Measures **brokerage** in the network ➔ disruption of these nodes = disruption of communication



Freeman (1977), and Anthonisse (1971, unpublished)

Eigenvector centrality = $\lambda^{-1} \sum_j A_{ij} e_j$

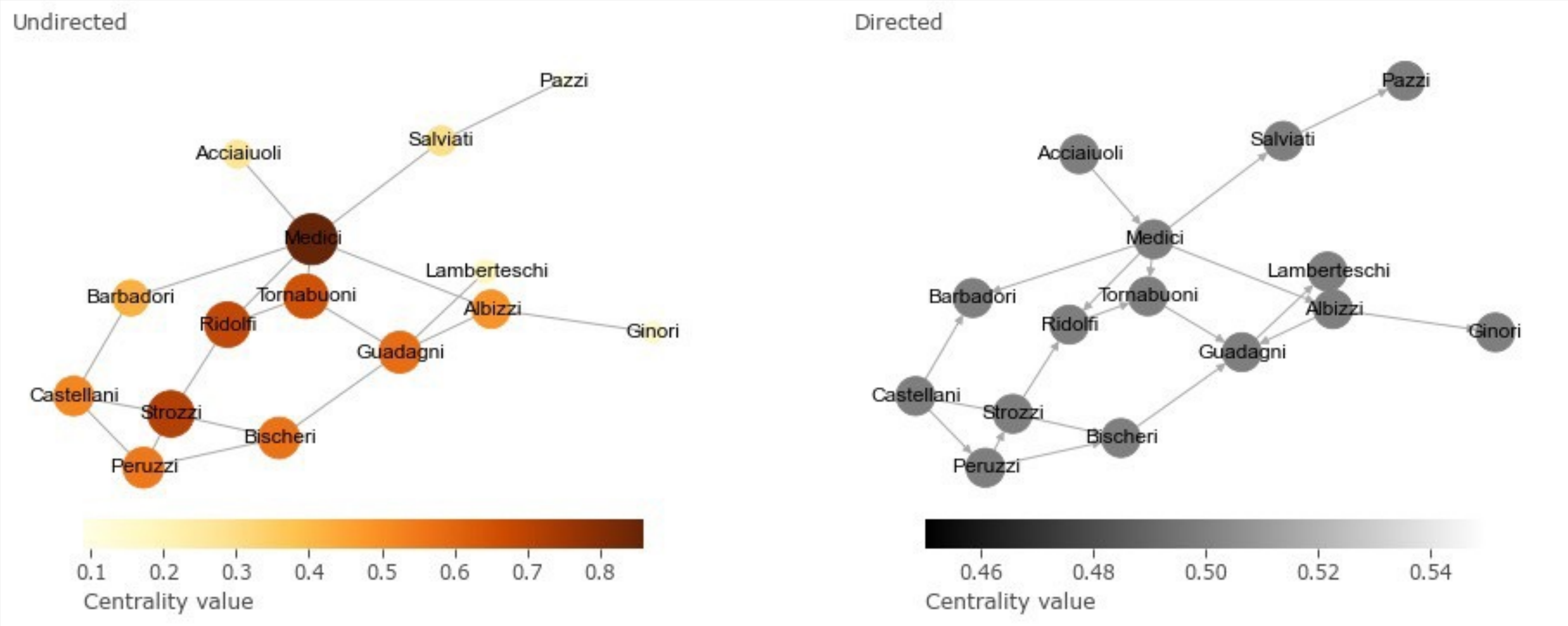
Takes into account how central your neighbors are.

e_j = eigenvector centrality of node j

λ = largest eigenvalue

Measures total **influence** in the network (assuming all nodes are the same)

Only for undirected, fully-connected networks!



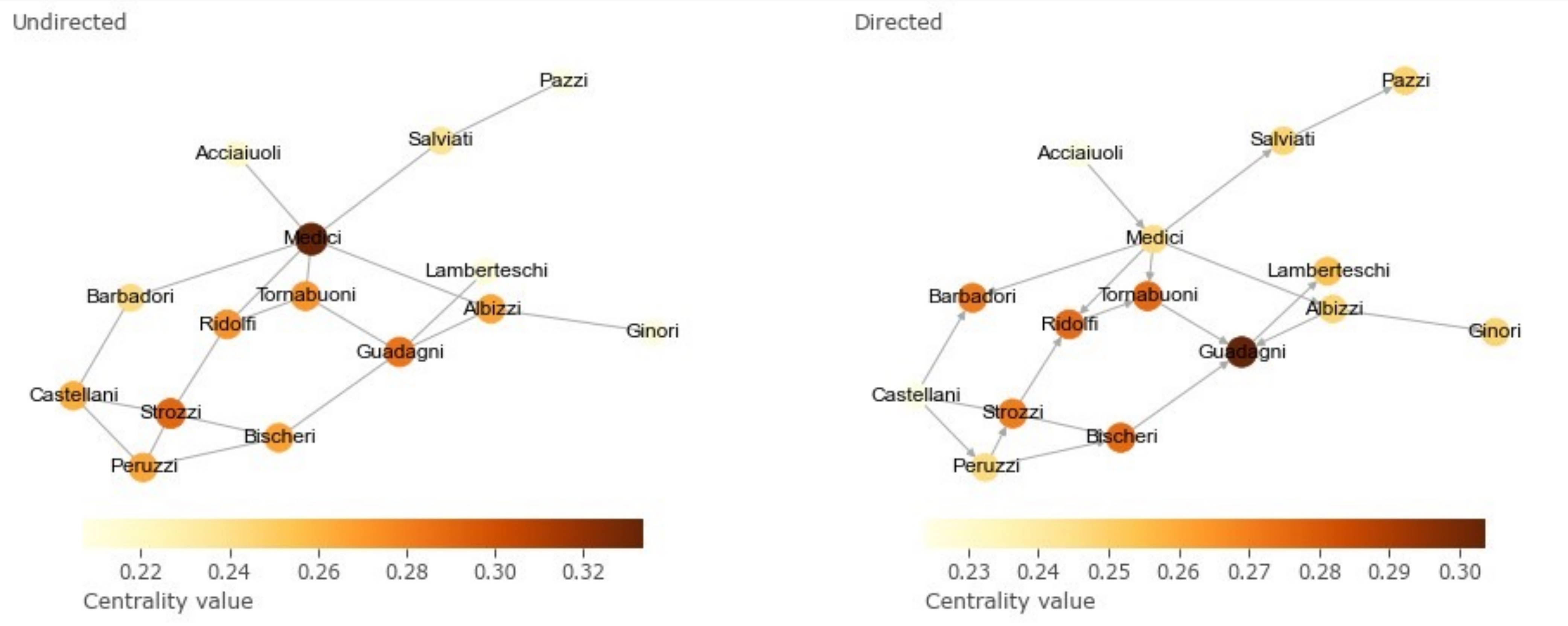
Bonacich, 1987

Katz centrality = $\alpha \sum_j A_{ij} k_j + \beta$

k_j = Katz centrality of node *j*

Takes into account how central your neighbors are, **each node has a minimum value of β,** and the balance between the constant and the eigenvector part is controlled by α

Measures total **influence** in the network (assuming all nodes are the same)



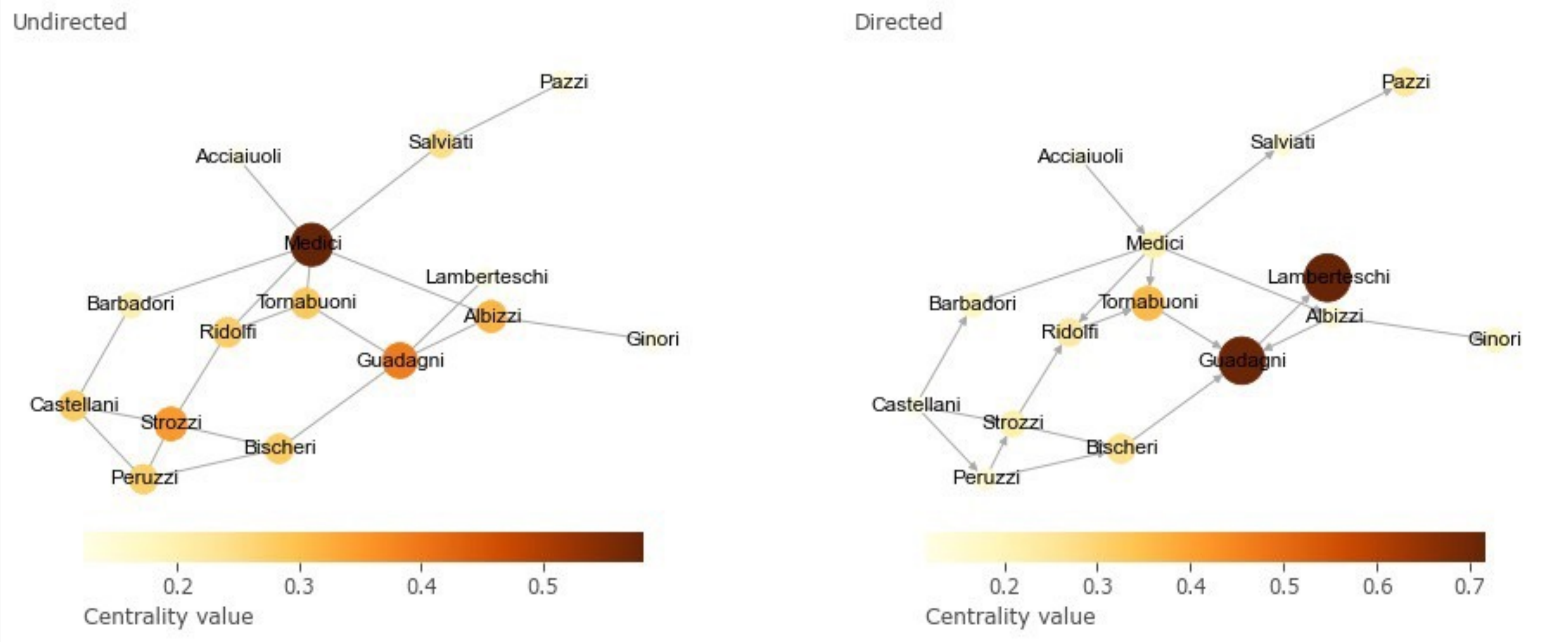
Katz, 1953

PageRank centrality = $(1-\alpha)\sum_j \frac{A_{ij}\cdot p_j}{d_j} + \alpha$

d_j = Degree of node *j*. p_j = Pagerank centrality of node j

Takes into account how central your neighbors are. Each node has a minimum value of α. The pagerank of a node is α plus **the pagerank of your neighbors** (normalized by their out-degree)

Measures total **influence** in the network (assuming all nodes are similar)



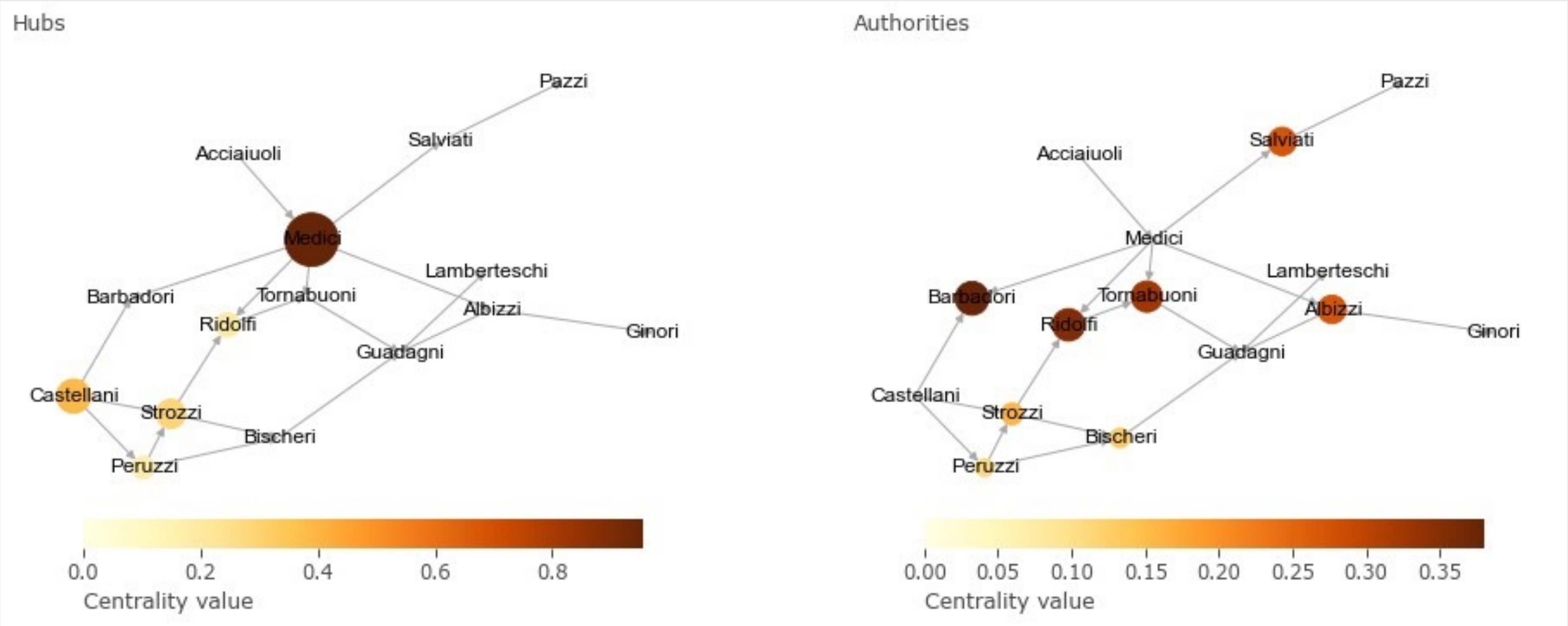
Bonacich, 1987

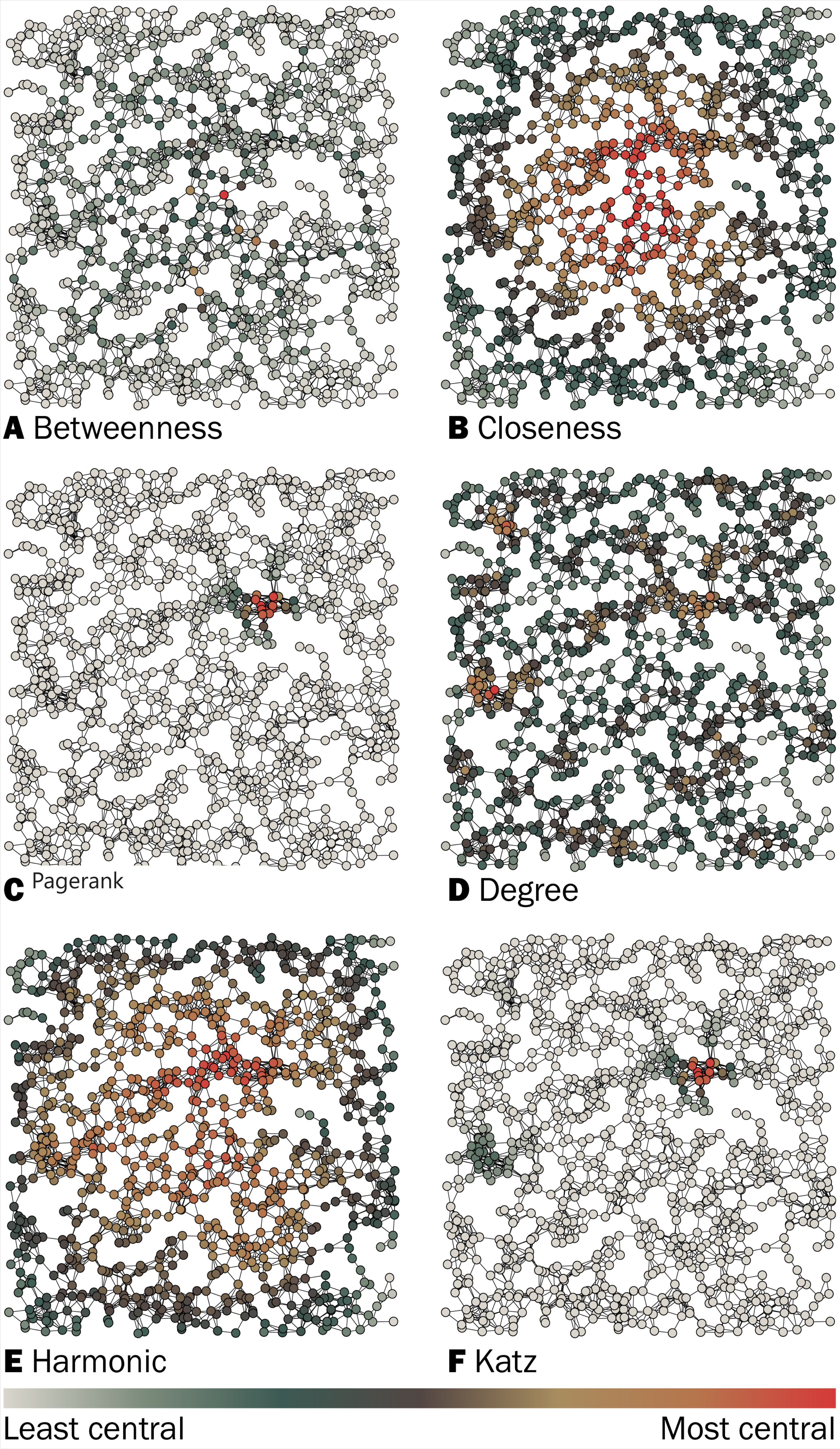
Hubs and authorities (HITS)

A node may be important if it points to others with high centrality, e.g., a review article pointing to prestigious articles

Authorities are nodes that contain useful information on a topic of interest and **hubs** are nodes that tell us where the best authorities are to be found (Newman). Two centralities: authority (a) and hub (h) centrality. Only for directed networks!

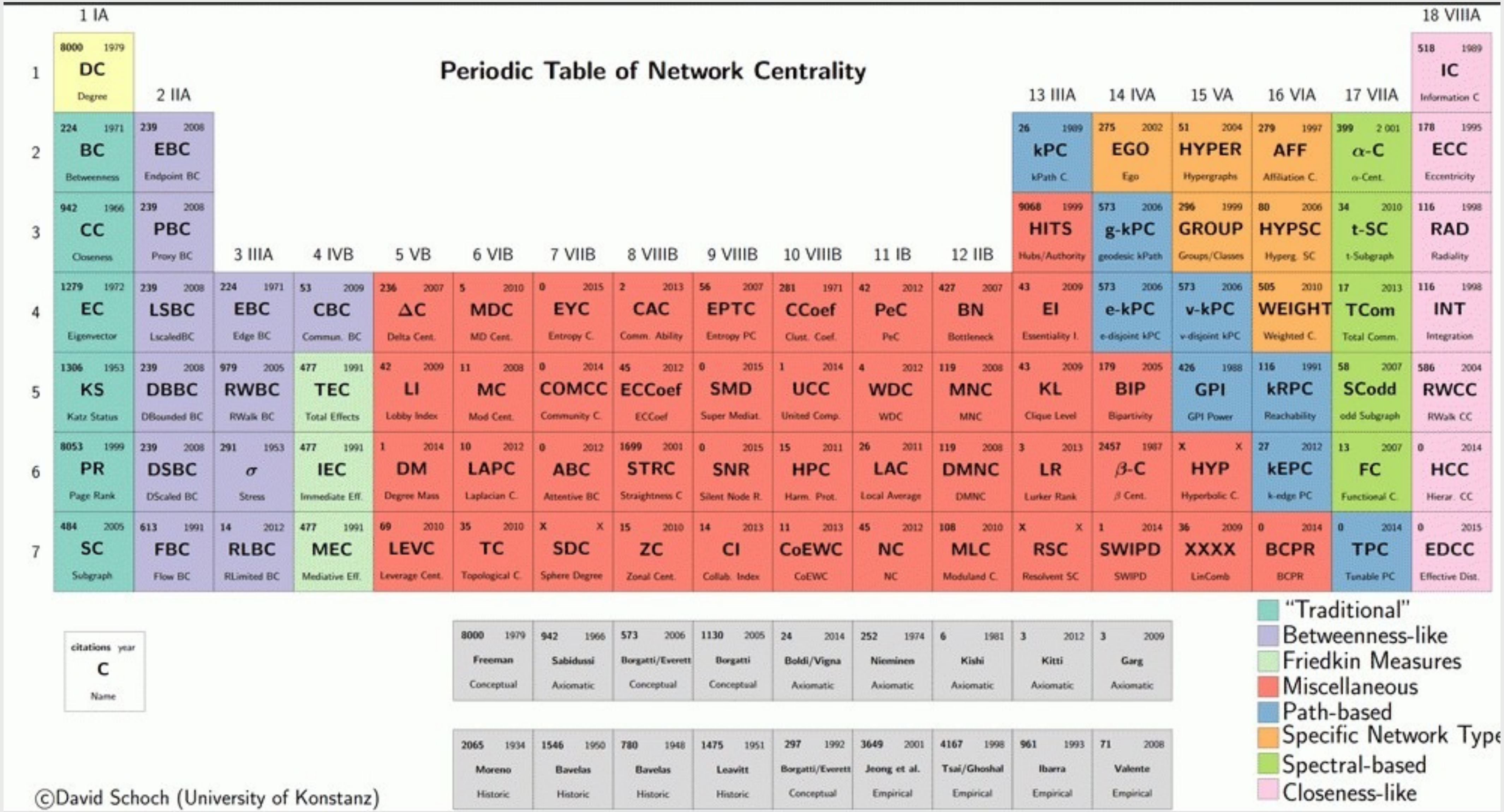
$h_i = \alpha \sum_j A_{ij} a_j$ and $a_i = \alpha \sum_j A_{ij} h_j$




A Betweenness
B Closeness
C Pagerank
D Degree
E Harmonic
F Katz
Least central
Most central

Use a centrality measure that fits your theory, not the one that gives you the best results

Consider what is the real objective (e.g. is it to enable low-income individuals to increase their social capital?) *(https://petterhol.me/2019/01/11/the-importance-of-being-earnest-about-node-importance/)*



Chains

Sometimes data is represented as chains

- Life trajectory: Aranda → León → Vermont → Amsterdam
- Ownership chain: (right figure)

They allow you to do other analysis:

- Importance of the node based on how often it is found in between
- Importance of the node based on how many people jump to you

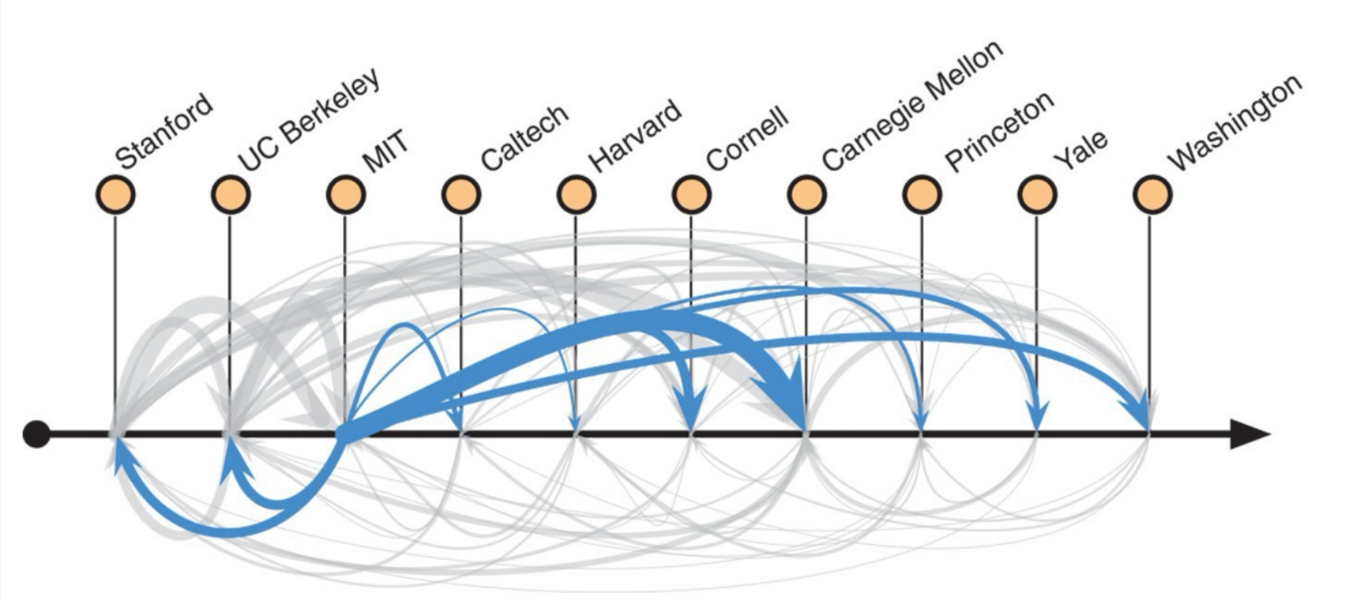
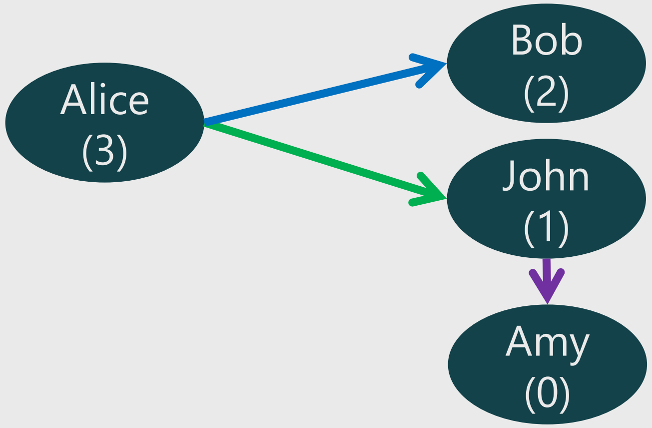


Fig. 1 Prestige hierarchies in faculty hiring networks. *Clauset, Arbesman and Larremore (2015)*

Practical 2:
Exercise 4 and 5

Linear algebra and centrality measures

Matrix multiplication: paths



Interpretation A: Presence of path between node i and j

A^2 =

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	0	0	0	0
John	0	0	0	1
Amy	0	0	0	0

@

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	0	0	0	0
John	0	0	0	1
Amy	0	0	0	0

=

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	0	0	1
Bob	0	0	0	0
John	0	0	0	0
Amy	0	0	0	0

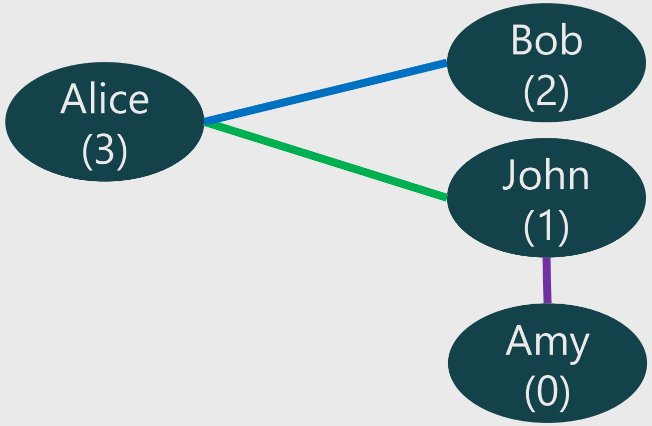
From you → to your neighbors

From your neighbors → to their neighbors

You → the neig. of your neig.

Degree

= paths of length 1



Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	1	0	0	0
John	1	0	0	1
Amy	0	0	1	0

@

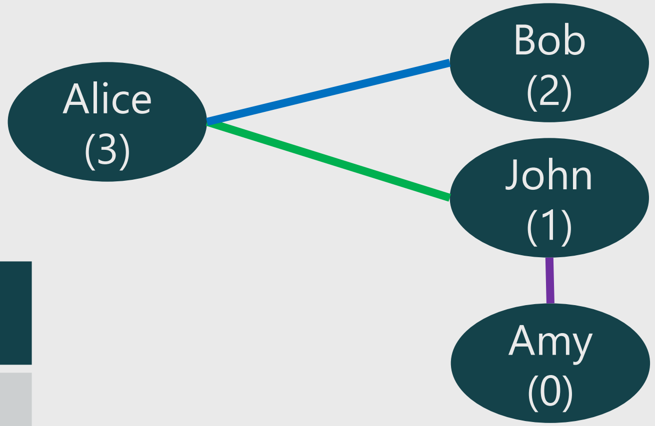
Alice	1
Bob	1
John	1
Amy	1

=

	Degree
Alice	2
Bob	1
John	2
Amy	1

Eigenvector

= paths over all possible lengths



Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	1	0	0	0
John	1	0	0	1
Amy	0	0	1	0

@

Alice	1
Bob	1
John	1
Amy	1

=

	Degree
Alice	2
Bob	1
John	2
Amy	1

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	1	1	0
Bob	1	0	0	0
John	1	0	0	1
Amy	0	0	1	0

@

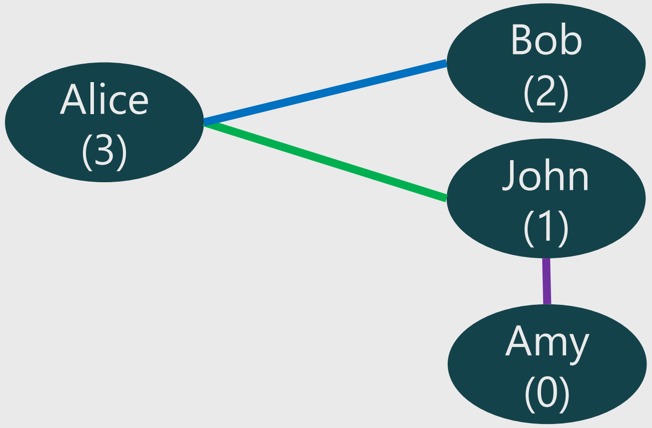
Alice	2
Bob	1
John	2
Amy	1

=

Alice	3
Bob	2
John	3
Amy	2

...

Another view on matrix multiplications: Random walks on undirected networks



Transition matrix (row-normalized A)

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	0.5	0.5	0
Bob	1	0	0	0
John	0.5	0	0	0.5
Amy	0	0	1	0

@

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	0.5	0.5	0
Bob	1	0	0	0
John	0.5	0	0	0.5
Amy	0	0	1	0

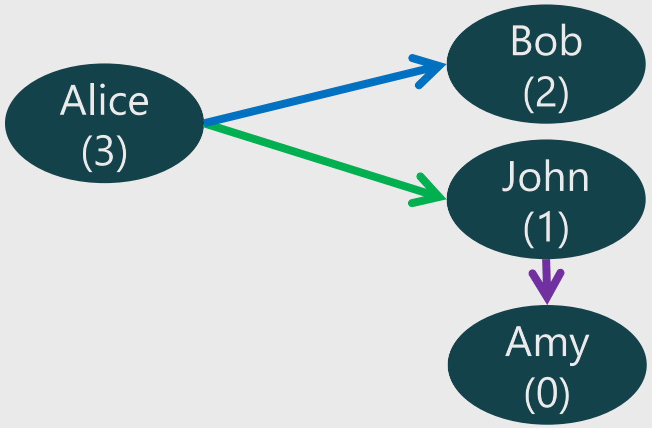
=

A random walker starting in Alice will go 50% of the times to Bob, 50% of the times to John.

- From Bob it goes 100% of the times back to Alice
- From John it goes 50% of the times to John, 50% back to Alice

If we let the random walker walk forever ➔ The fraction of time spend at each node converges to the **degree centrality** of the node

Another view on matrix multiplications: Random walks on directed networks



Transition matrix (row-normalized A)

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	0.5	0.5	0
Bob	0	0	0	0
John	0	0	0	1
Amy	0	0	0	0

@

Target → ↓ Source	Alice	Bob	John	Amy
Alice	0	0.5	0.5	0
Bob	0	0	0	0
John	0	0	0	1
Amy	0	0	0	0

=

A random walker starting in Alice will go 50% of the times to Bob, 50% of the times to John.

- From Bob it gets trapped
- From John it goes 100% of the times to Amy and gets trapped

If we let random walkers walk forever ➔ They gets trapped in the extremes!

Solution: PageRank (the alpha parameter can be understood as a teletransportation probablity)

Practical 3:
Working with networks using Gephi

- Follow this tutorial (slides 1–23 only!): https://gephi.org/users/quick-start/
- In community detection use the "stochastic blockmodel" instead of modularity maximization (or try both)
- You can choose to use our own data (https://tinyurl.com/network-game) or the Twitter or PPI data.

Python exercise
notebook 2, ex.7

Recap of today

There is important information encoded in relationships/interactions
Modeling systems using networks allow us to study that information
We can represent networks using adjacencies matrixes or adjacencies lists
Network science is closely linked to linear algebra (matrix multiplication).

You can now:

- **Describe networks**: number of edges and nodes, components, density, assortativity, clustering, diameter, degree distributions.
- **Test hypothesis** using network models.
- Find the most important nodes using **centrality measures:**
  - Random paths:
    - Degree
    - Eigenvector / PageRank / Katz / Hubs and Authorities
  - Shortest paths
    - Closeness / Harmonic: Distance to all other nodes
    - Betweenness: Presence in shortest paths between other nodes

Tomorrow:

- Network Models: allow us to generate networks and test hypothesis
- Statistical Models: REM, ERGM, SAOM