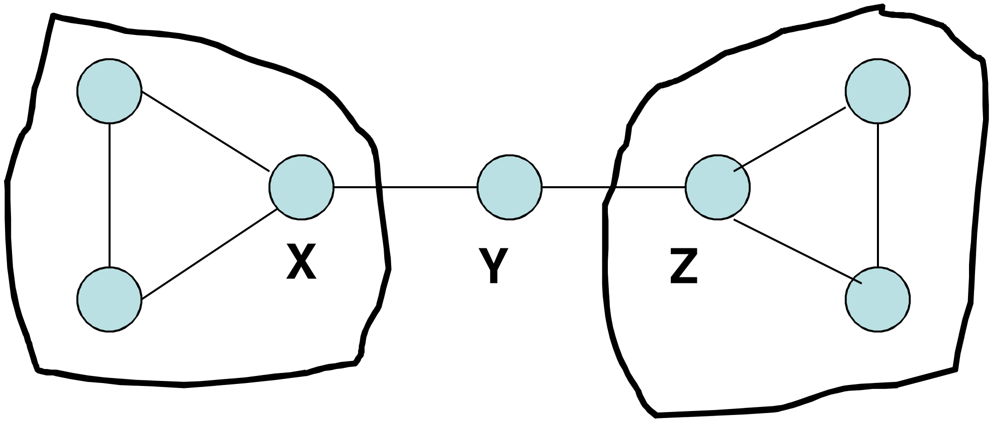
Centrality Measures

Dr. Natarajan Meghanathan
Professor of Computer Science
Jackson State University
E-mail: natarajan.meghanathan@jsums.edu

Centrality

- Tells us which nodes are important in a network based on the topological structure of the network (instead of just looking at the popularity of nodes)
  - How influential a person is within a social network
  - Which genes play a crucial role in regulating systems and processes
  - Infrastructure networks: if the node is removed, it would critically impede the functioning of the network.



Nodes X and Z have higher Degree

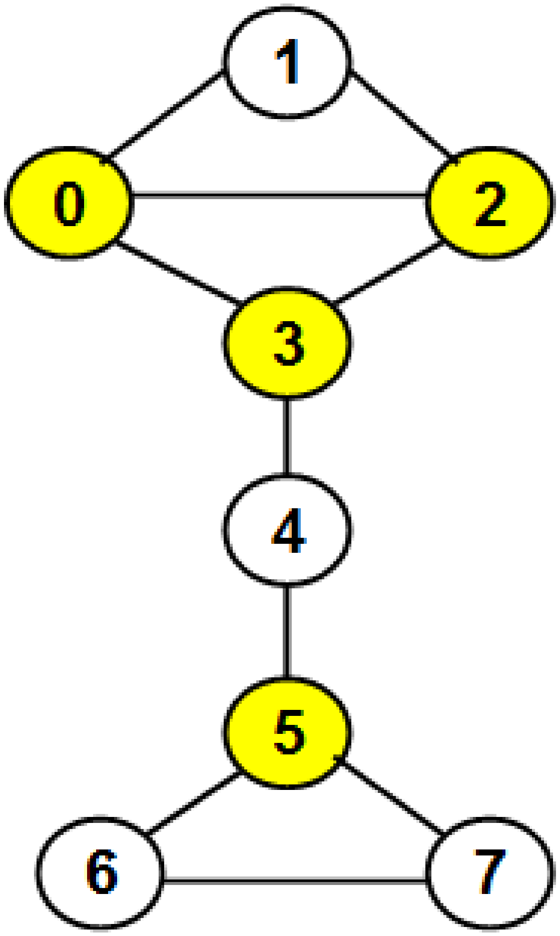
Node Y is more central from the point of view of
Betweenness – to reach from one end to the other
Closeness – can reach every other vertex in the fewest number of hops

Centrality Measures

- Degree-based Centrality Measures
  - Degree Centrality: measure of the number of vertices adjacent to a vertex (degree)
  - Eigenvector Centrality: measure of the degree of the vertex as well as the degree of its neighbors

- Shortest-path based Centrality Measures
  - Betweeness Centrality: measure of the number of shortest paths a node is part of
  - Closeness Centrality: measure of how close is a vertex to the other vertices [sum of the shortest path distances]
  - Farness Centrality: captures the variation of the shortest path distances of a vertex to every other vertex

Degree Centrality



$$
\begin{array}{cc}
 & \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\
\begin{array}{c} 0\\1\\2\\3\\4\\5\\6\\7 \end{array} &
\begin{bmatrix}
0 & 1 & 1 & 1 & 0 & 0 & 0 & 0\\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0\\
1 & 0 & 1 & 0 & 1 & 0 & 0 & 0\\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1\\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1\\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{bmatrix}
\end{array}
\times
\begin{bmatrix} 1\\1\\1\\1\\1\\1\\1\\1 \end{bmatrix}
=
\begin{bmatrix} 3\\2\\3\\3\\2\\3\\2\\2 \end{bmatrix}
\begin{array}{c} \text{Vertex IDs} \\ 0\\1\\2\\3\\4\\5\\6\\7 \end{array}
$$

Adjacency Matrix

Column Vector

Degree Centrality

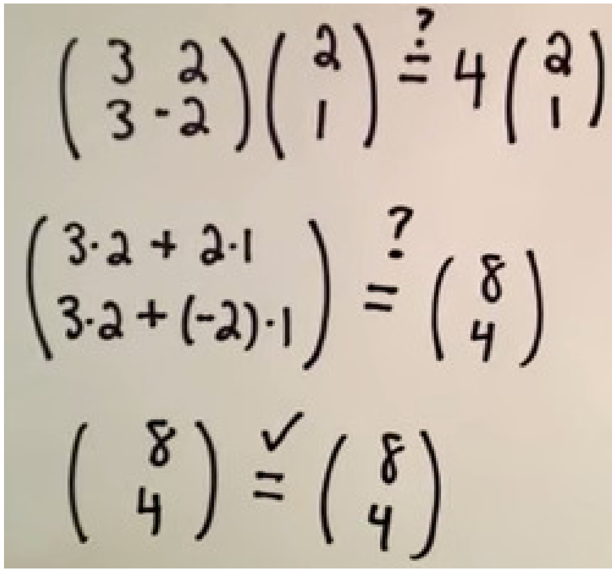
Weakness: Very likely that more than one vertex has the same degree and not possible to uniquely rank the vertices

Eigenvalue and Eigenvector

- Let A be an nxn matrix.
- A scalar λ is called an Eigenvalue of A if there is a non-zero vector X such that AX = λX. Such a vector X is called an Eigenvector of A corresponding to λ.
- Example: $\begin{bmatrix} 2 \\ 1 \end{bmatrix}$ is an Eigenvector of A = $\begin{bmatrix} 3 & 2 \\ 3 & -2 \end{bmatrix}$ for λ = 4

An n x n square matrix has 'n' eigenvalues and the corresponding Eigenvectors

The eigenvector corresponding to the largest eigenvalue is called the Principal Eigenvector



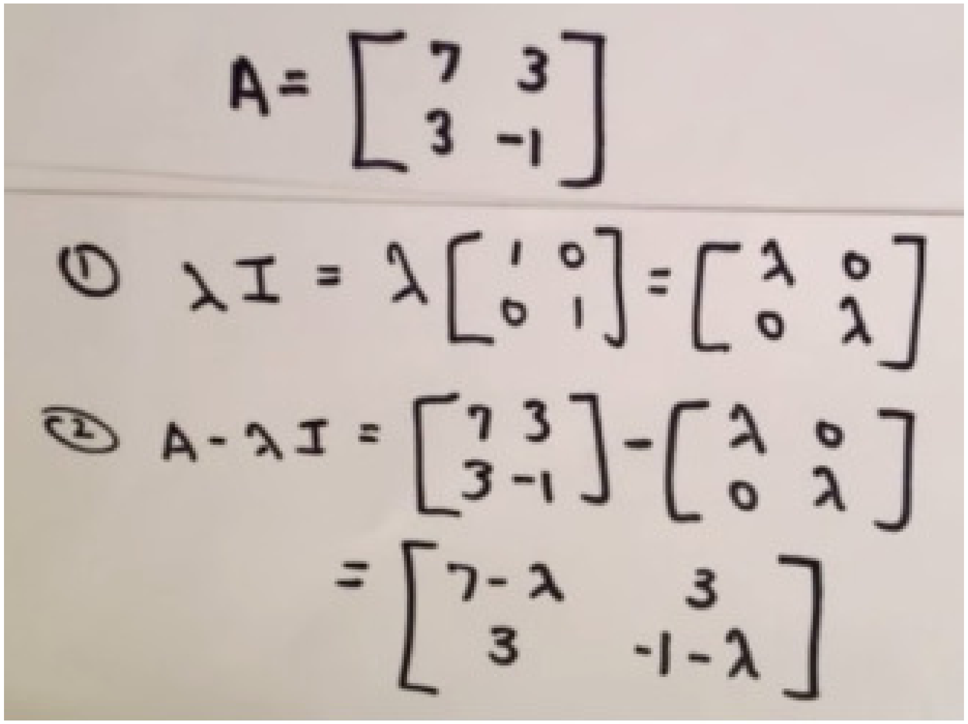
$$\begin{pmatrix} 3 & 2 \\ 3 & -2 \end{pmatrix}\begin{pmatrix} 2 \\ 1 \end{pmatrix} \overset{?}{=} 4\begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} 3\cdot 2 + 2\cdot 1 \\ 3\cdot 2 + (-2)\cdot 1 \end{pmatrix} \overset{?}{=} \begin{pmatrix} 8 \\ 4 \end{pmatrix}$$

$$\begin{pmatrix} 8 \\ 4 \end{pmatrix} \overset{\checkmark}{=} \begin{pmatrix} 8 \\ 4 \end{pmatrix}$$

The largest eigenvalue is also called the Spectral radius

Finding Eigenvalues and Eigenvectors

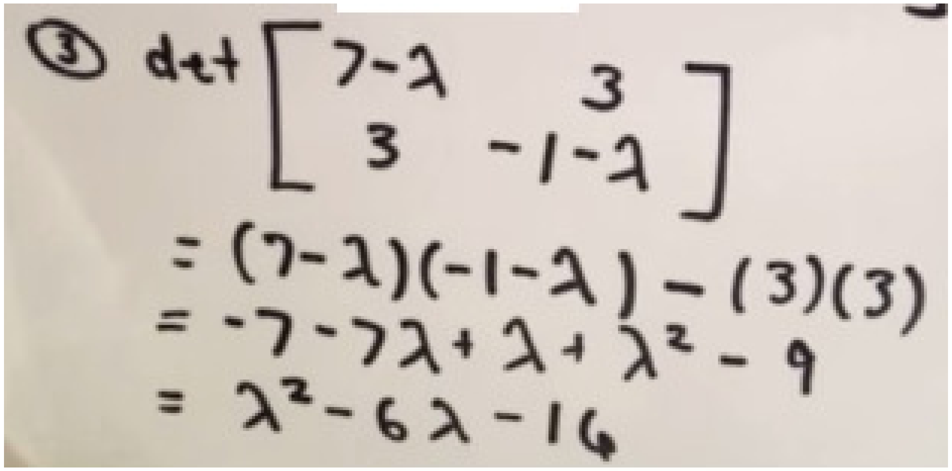


$$A = \begin{bmatrix} 7 & 3 \\ 3 & -1 \end{bmatrix}$$

① $\lambda I = \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$

② $A - \lambda I = \begin{bmatrix} 7 & 3 \\ 3 & -1 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$

$= \begin{bmatrix} 7-\lambda & 3 \\ 3 & -1-\lambda \end{bmatrix}$



③ $\det \begin{bmatrix} 7-\lambda & 3 \\ 3 & -1-\lambda \end{bmatrix}$

$= (7-\lambda)(-1-\lambda) - (3)(3)$

$= -7 - 7\lambda + \lambda + \lambda^2 - 9$

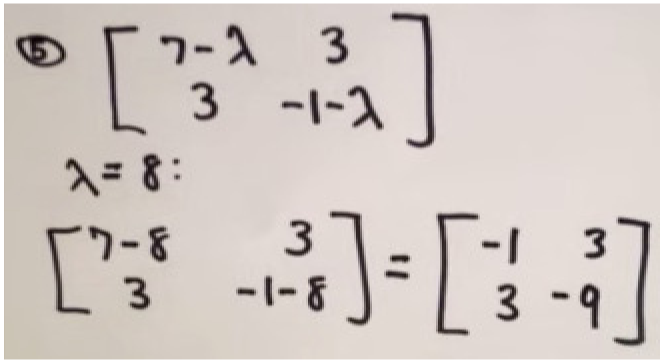
$= \lambda^2 - 6\lambda - 16$

(4) Solving for λ:

$(\lambda - 8)\ (\lambda + 2) = 0$

$\lambda = 8$ and $\lambda = -2$ are the Eigen values

(5) Consider $A - \lambda I$



⑤ $\begin{bmatrix} 7-\lambda & 3 \\ 3 & -1-\lambda \end{bmatrix}$

$\lambda = 8$:

$\begin{bmatrix} 7-8 & 3 \\ 3 & -1-8 \end{bmatrix} = \begin{bmatrix} -1 & 3 \\ 3 & -9 \end{bmatrix}$

$= B$

Solve $B X = 0$

$$\begin{bmatrix} -1 & 3 \\ 3 & -9 \end{bmatrix} \begin{bmatrix} X1 \\ X2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

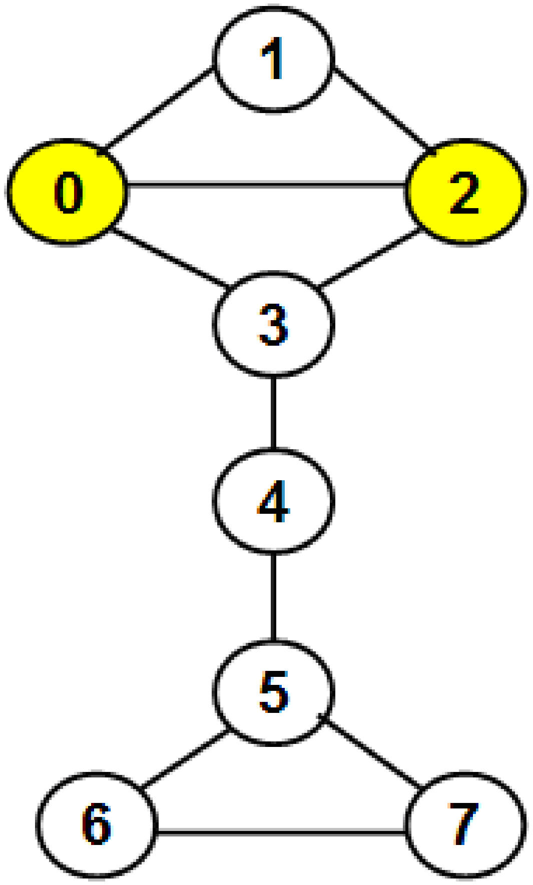
$-X1 + 3X2 = 0 \rightarrow$ $X1 = 3X2$

$3X1 - 9X2 = 0 \rightarrow$ $3X1 = 9X2 \Rightarrow X1 = 3X2$

If $X2 = 1$;
$X1 = 3$

$\begin{bmatrix} 3 \\ 1 \end{bmatrix}$ is an eigenvector for **$\lambda = 8$**

Eigenvector Centrality (1)



Iteration 1

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \\ 3 \\ 3 \\ 2 \\ 3 \\ 2 \\ 2 \end{bmatrix} \equiv \begin{bmatrix} 0.416 \\ 0.277 \\ 0.416 \\ 0.416 \\ 0.277 \\ 0.416 \\ 0.277 \\ 0.277 \end{bmatrix}$$

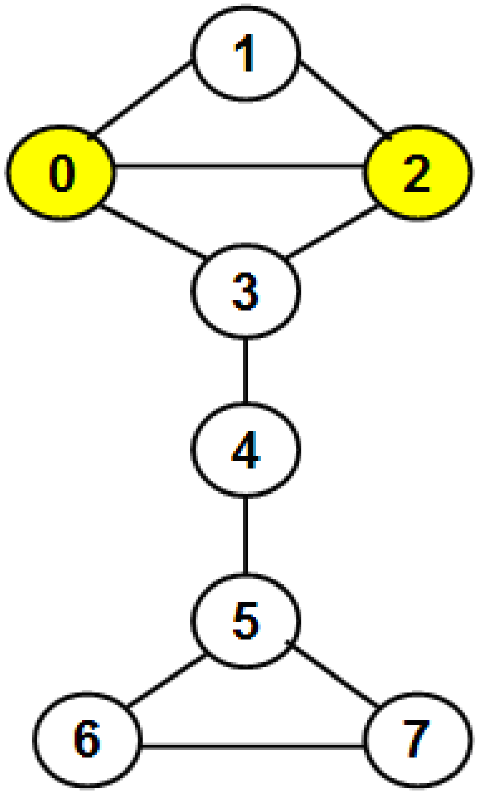
Normalized Value = 7.21

Iteration 2

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0.416 \\ 0.277 \\ 0.416 \\ 0.416 \\ 0.277 \\ 0.416 \\ 0.277 \\ 0.277 \end{bmatrix} = \begin{bmatrix} 1.109 \\ 0.832 \\ 1.109 \\ 1.109 \\ 0.832 \\ 0.831 \\ 0.693 \\ 0.693 \end{bmatrix} \equiv \begin{bmatrix} 0.428 \\ 0.321 \\ 0.428 \\ 0.428 \\ 0.321 \\ 0.321 \\ 0.268 \\ 0.268 \end{bmatrix}$$

Normalized Value = 2.59

Eigenvector Centrality (2)



Iteration 3

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \mathbf{x} \begin{bmatrix} 0.428 \\ 0.321 \\ 0.428 \\ 0.428 \\ 0.321 \\ 0.321 \\ 0.268 \\ 0.268 \end{bmatrix} = \begin{bmatrix} 1.177 \\ 0.856 \\ 1.284 \\ 1.177 \\ 0.749 \\ 0.857 \\ 0.589 \\ 0.589 \end{bmatrix} \equiv \begin{bmatrix} 0.441 \\ 0.321 \\ 0.481 \\ 0.441 \\ 0.281 \\ 0.321 \\ 0.221 \\ 0.221 \end{bmatrix}$$

Normalized Value = 2.67

Iteration 4

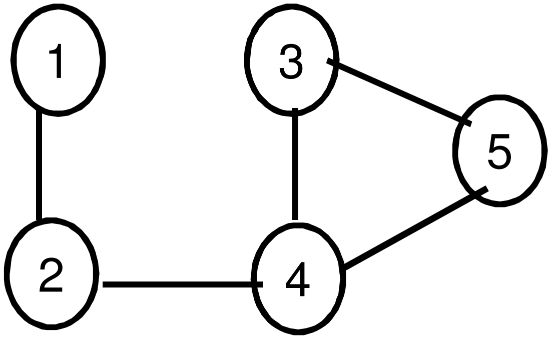
$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \mathbf{x} \begin{bmatrix} 0.441 \\ 0.321 \\ 0.481 \\ 0.441 \\ 0.281 \\ 0.321 \\ 0.221 \\ 0.221 \end{bmatrix} = \begin{bmatrix} 1.243 \\ 0.922 \\ 1.203 \\ 1.203 \\ 0.762 \\ 0.723 \\ 0.542 \\ 0.542 \end{bmatrix} \equiv \begin{bmatrix} 0.471 \\ 0.349 \\ 0.456 \\ 0.456 \\ 0.289 \\ 0.274 \\ 0.205 \\ 0.205 \end{bmatrix}$$

Normalized Value = 2.64

<u>After 7 iterations</u>

Vertex ID	0	1	2	3	4	5	6	7
Principal Eigenvector	0.489	0.364	0.489	0.467	0.264	0.232	0.155	0.155

EigenVector Centrality Example (1)



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

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

Iteration 1

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 3 \\ 2 \end{bmatrix} \equiv \begin{bmatrix} 0.213 \\ 0.426 \\ 0.426 \\ 0.639 \\ 0.426 \end{bmatrix}$$

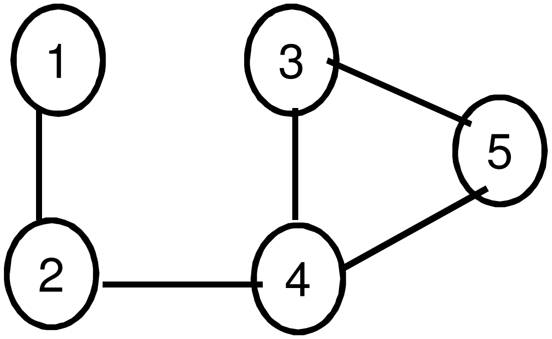
Normalized Value = 4.69

Iteration 2

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0.213 \\ 0.426 \\ 0.426 \\ 0.639 \\ 0.426 \end{bmatrix} = \begin{bmatrix} 0.426 \\ 0.852 \\ 1.065 \\ 1.278 \\ 1.065 \end{bmatrix} \equiv \begin{bmatrix} 0.195 \\ 0.389 \\ 0.486 \\ 0.584 \\ 0.486 \end{bmatrix}$$

Normalized Value = 2.19

EigenVector Centrality Example (1)



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

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

Iteration 3

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0.195 \\ 0.389 \\ 0.486 \\ 0.584 \\ 0.486 \end{bmatrix} = \begin{bmatrix} 0.389 \\ 0.779 \\ 1.07 \\ 1.361 \\ 1.07 \end{bmatrix} \equiv \begin{bmatrix} 0.176 \\ 0.352 \\ 0.484 \\ 0.616 \\ 0.484 \end{bmatrix}$$

Normalized Value = 2.21

Iteration 4

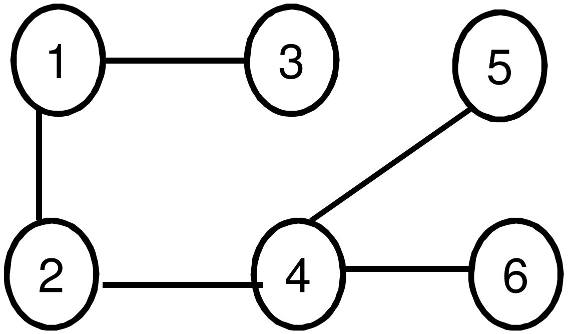
$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0.176 \\ 0.352 \\ 0.484 \\ 0.616 \\ 0.484 \end{bmatrix} = \begin{bmatrix} 0.352 \\ 0.792 \\ 1.100 \\ 1.320 \\ 1.100 \end{bmatrix}$$

Normalized Value = 2.21 converges

Eigen Vector Centrality

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{bmatrix} 0.176 \\ 0.352 \\ 0.484 \\ 0.616 \\ 0.484 \end{bmatrix}$$

EigenVector Centrality Example (2)



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

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

Iteration 1

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 1 \\ 3 \\ 1 \\ 1 \end{bmatrix} \equiv \begin{bmatrix} 0.447 \\ 0.447 \\ 0.224 \\ 0.671 \\ 0.224 \\ 0.224 \end{bmatrix}$$

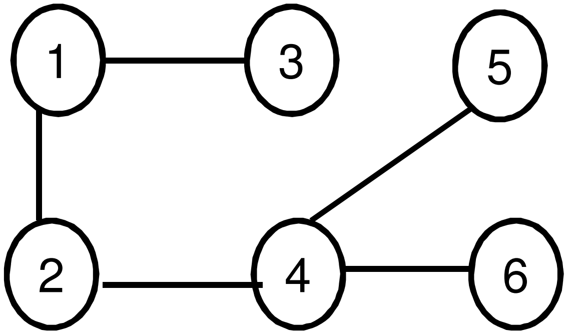
Normalized Value = 4.472

Iteration 2

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.447 \\ 0.447 \\ 0.224 \\ 0.671 \\ 0.224 \\ 0.224 \end{bmatrix} = \begin{bmatrix} 0.671 \\ 0.671 \\ 0.447 \\ 0.895 \\ 0.671 \\ 0.671 \end{bmatrix} \equiv \begin{bmatrix} 0.401 \\ 0.401 \\ 0.267 \\ 0.535 \\ 0.401 \\ 0.401 \end{bmatrix}$$

Normalized Value = 1.674

EigenVector Centrality Example (2)



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

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

<u>Iteration 3</u>

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.401 \\ 0.401 \\ 0.267 \\ 0.535 \\ 0.401 \\ 0.401 \end{bmatrix} = \begin{bmatrix} 0.668 \\ 0.936 \\ 0.401 \\ 1.203 \\ 0.535 \\ 0.535 \end{bmatrix} \equiv \begin{bmatrix} 0.357 \\ 0.500 \\ 0.214 \\ 0.643 \\ 0.286 \\ 0.286 \end{bmatrix}$$

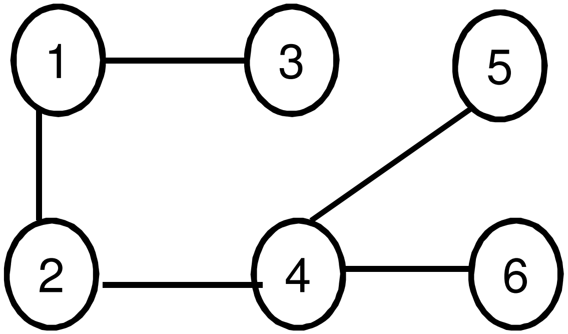
Normalized Value = 1.872

<u>Iteration 4</u>

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.357 \\ 0.500 \\ 0.214 \\ 0.643 \\ 0.286 \\ 0.286 \end{bmatrix} = \begin{bmatrix} 0.714 \\ 1.000 \\ 0.357 \\ 1.072 \\ 0.643 \\ 0.643 \end{bmatrix} \equiv \begin{bmatrix} 0.376 \\ 0.526 \\ 0.188 \\ 0.564 \\ 0.338 \\ 0.338 \end{bmatrix}$$

Normalized Value = 1. 901

EigenVector Centrality Example (2)



Iteration 5

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.376 \\ 0.526 \\ 0.188 \\ 0.564 \\ 0.338 \\ 0.338 \end{bmatrix} = \begin{bmatrix} 0.714 \\ 0.940 \\ 0.376 \\ 1.202 \\ 0.564 \\ 0.564 \end{bmatrix} \equiv \begin{bmatrix} 0.376 \\ 0.494 \\ 0.198 \\ 0.632 \\ 0.297 \\ 0.297 \end{bmatrix}$$

Normalized Value = 1. 901 converges

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

Let X0 = $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

EigenVector Centrality

$$\begin{bmatrix} 0.376 \\ 0.494 \\ 0.198 \\ 0.632 \\ 0.297 \\ 0.297 \end{bmatrix}$$

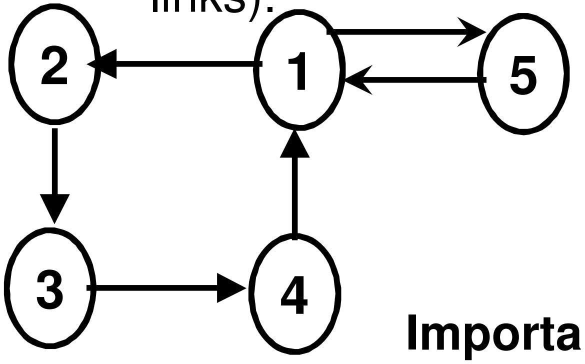
Node Ranking

4
2
1
5
6
3

Note that we typically stop when the EigenVector values converge.
For exam purposes, we will Stop when the Normalized value converges.

Eigen Vector Centrality for Directed Graphs

- For directed graphs, we can use the Eigen Vector centrality to evaluate the “importance” of a node (based on the out-degree Eigen Vector) and the “prestige” of a node (through the in-degree Eigen Vector)
  - A node is considered to be more important if it has out-going links to nodes that in turn have a larger out-degree (i.e., more out-going links).
  - A node is considered to have a higher “prestige”, if it has in-coming links from nodes that themselves have a larger in-degree (i.e., more in-coming links).



0	1	0	0	1
0	0	1	0	0
0	0	0	1	0
1	0	0	0	0
1	0	0	0	0

Out-going links based Adj. Matrix

Importance of Nodes (Out-deg. Centrality)

Node	Score
1	0.5919
4	0.4653
5	0.4653
3	0.3658
2	0.2876

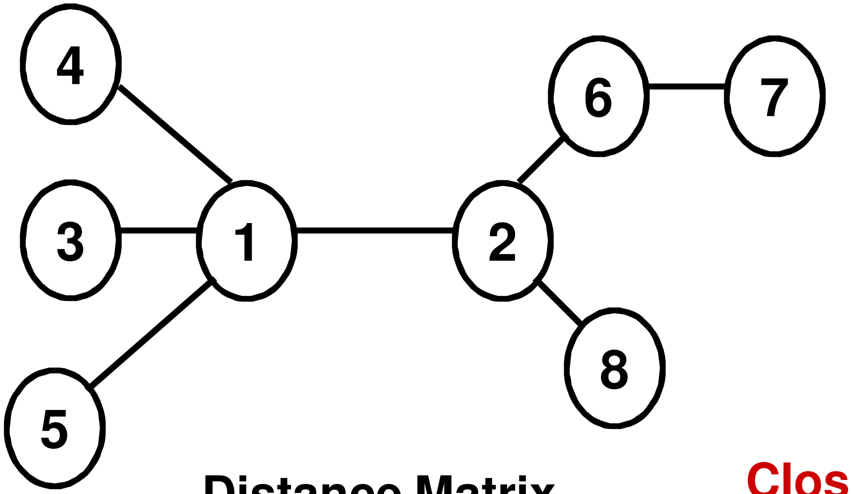
0	0	0	1	1
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
1	0	0	0	0

In-coming links based Adj. Matrix

Prestige of Nodes (In-deg. Centrality)

Node	Score
1	0.5919
2	0.4653
5	0.4653
3	0.3658
4	0.2876

Closeness and Farness Centrality



Distance Matrix

Closeness

	1	2	3	4	5	6	7	8	**Sum of distances**
1	**0**	**1**	**1**	**1**	**1**	**2**	**3**	**2**	11
2	**1**	**0**	**2**	**2**	**2**	**1**	**2**	**1**	11
3	**1**	**2**	**0**	**2**	**2**	**3**	**4**	**3**	17
4	**1**	**2**	**2**	**0**	**2**	**3**	**4**	**3**	17
5	**1**	**2**	**2**	**2**	**0**	**3**	**4**	**3**	17
6	**2**	**1**	**3**	**3**	**3**	**0**	**1**	**2**	15
7	**3**	**2**	**4**	**4**	**4**	**1**	**0**	**3**	21
8	**2**	**1**	**3**	**3**	**3**	**2**	**3**	**0**	17

Principal Eigenvalue
η1 = 16.315

Farness

Principal Eigenvector
δ1 =
[0.2527
0.2518
0.3771
0.3771
0.3771
0.3278
0.4439
0.3763]

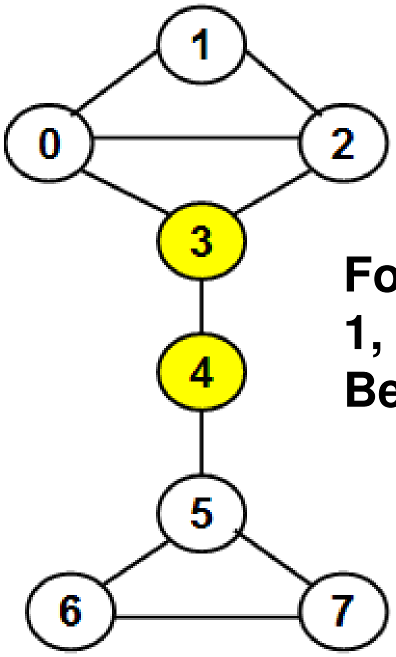
Ranking of Nodes

Score	Node ID
0.2518	2
0.2527	1
0.3278	6
0.3763	8
0.3771	3
0.3771	4
0.3771	5
0.4439	7

Betweeness Centrality

$$BWC(i) = \sum_{j \neq k \neq i} \frac{sp_{jk}(i)}{sp_{jk}}$$

- We will now discuss how to find the total number of shortest paths between any two vertices *j* and *k* as well as to find out how many of these shortest paths go through a vertex *i* (*j* ≠ *k* ≠ *i*).
- Use Breadth First Search (BFS) to find the shortest path tree from vertex j to every other vertex k
  - Root vertex j is at level 0
  - Vertices that are 1-hop away from j are at level 1; 2-hops away from j are at level 2, and so on.
  - The number of shortest paths from *j* to a vertex *k* at level *p* is the sum of the number of shortest paths from *j* to the neighbors of *k* in the original graph that are at level *p*-1
  - The number of shortest paths from *j* to *k* that go through vertex *i* is the maximum of the number of shortest paths from *j* to *i* and the number of shortest paths from *k* to *i*.



For vertices
1, 6 and 7
Betw.C = 0

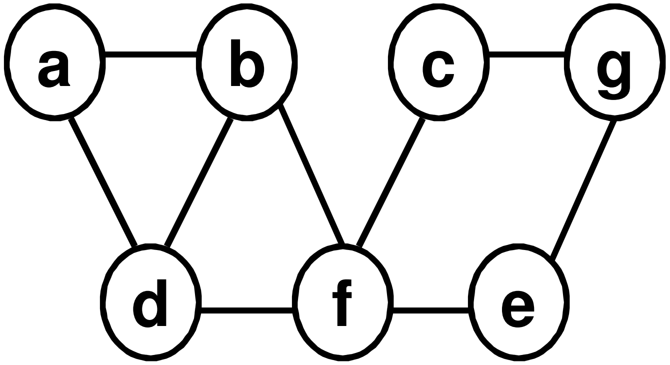
Betweeness for Vertex 5
Pair (6,0) --->1 / 1
Pair (7,0) --->1 / 1
Pair (6,1) --->2 / 2
Pair (7,1) --->2 / 2
Pair (6,2) --->1 / 1
Pair (7,2) --->1 / 1
Pair (6,3) --->1 / 1
Pair (7,3) --->1 / 1
Pair (6,4) --->1 / 1
Pair (7,4) --->1 / 1
Total of all Betweeness
(Vertex 5): 10

Betweeness for Vertex 0
Pair (3,1) --->1 / 2
Pair (4,1) --->1 / 2
Pair (5,1) --->1 / 2
Pair (6,1) --->1 / 2
Pair (7,1) --->1 / 2
Total of all Betweeness
(Vertex 0): 2.5

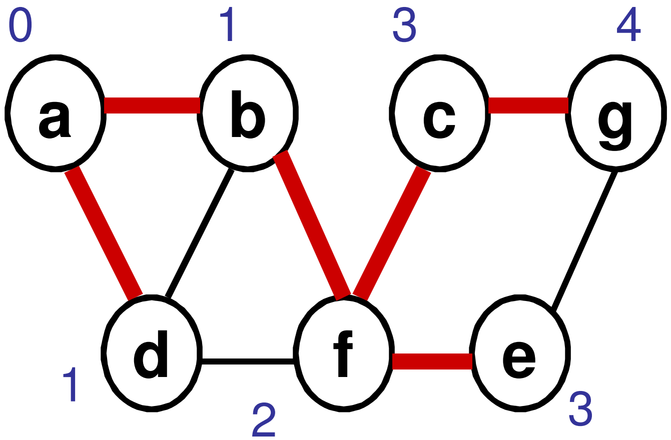
Betweeness for Vertex 3
Pair (4,0) --->1 / 1
Pair (5,0) --->1 / 1
Pair (6,0) --->1 / 1
Pair (7,0) --->1 / 1
Pair (4,1) --->2 / 2
Pair (5,1) --->2 / 2
Pair (6,1) --->2 / 2
Pair (7,1) --->2 / 2
Pair (4,2) --->1 / 1
Pair (5,2) --->1 / 1
Pair (6,2) --->1 / 1
Pair (7,2) --->1 / 1
Total of all Betweeness
(Vertex 3): 12

Betweeness for Vertex 2
Pair (3,1) --->1 / 2
Pair (4,1) --->1 / 2
Pair (5,1) --->1 / 2
Pair (6,1) --->1 / 2
Pair (7,1) --->1 / 2
Total of all Betweeness
(Vertex 2): 2.5

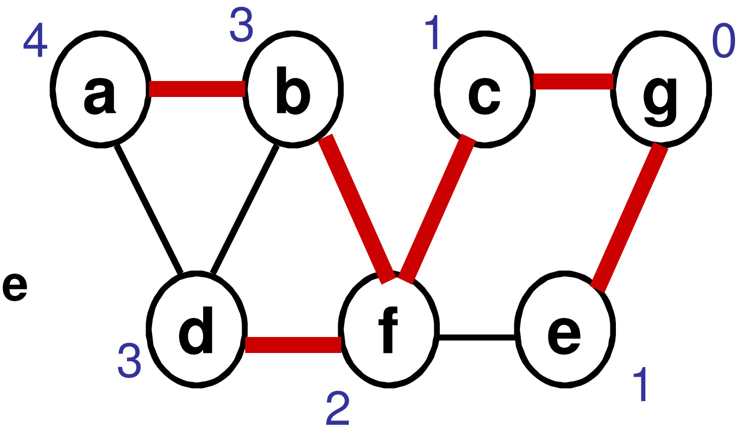
Betweeness for Vertex 4
Pair (5,0) --->1 / 1
Pair (6,0) --->1 / 1
Pair (7,0) --->1 / 1
Pair (5,1) --->2 / 2
Pair (6,1) --->2 / 2
Pair (7,1) --->2 / 2
Pair (5,2) --->1 / 1
Pair (6,2) --->1 / 1
Pair (7,2) --->1 / 1
Pair (5,3) --->1 / 1
Pair (6,3) --->1 / 1
Pair (7,3) --->1 / 1
Total of all Betweeness
(Vertex 4): 12

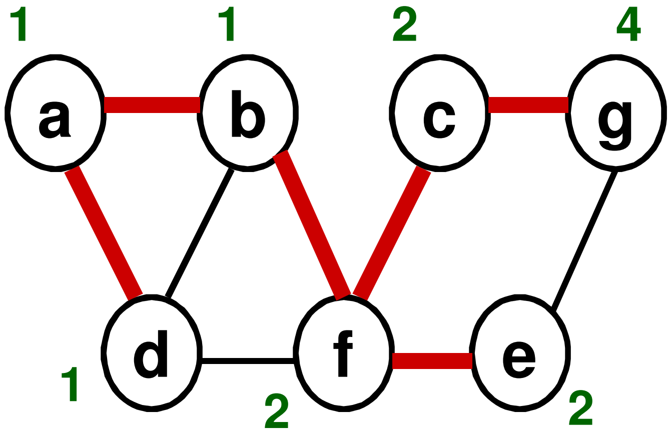


# shortest paths from a to g that go through c
is the maximum (# shortest paths from a to c,
# shortest paths from g to c)
= max (2, 1) = 2

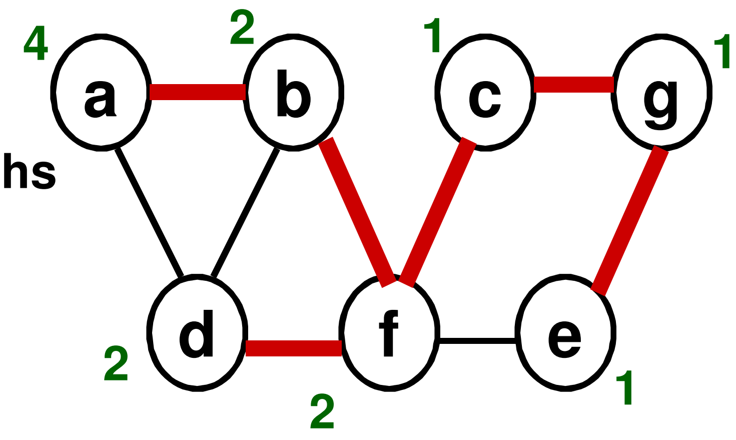


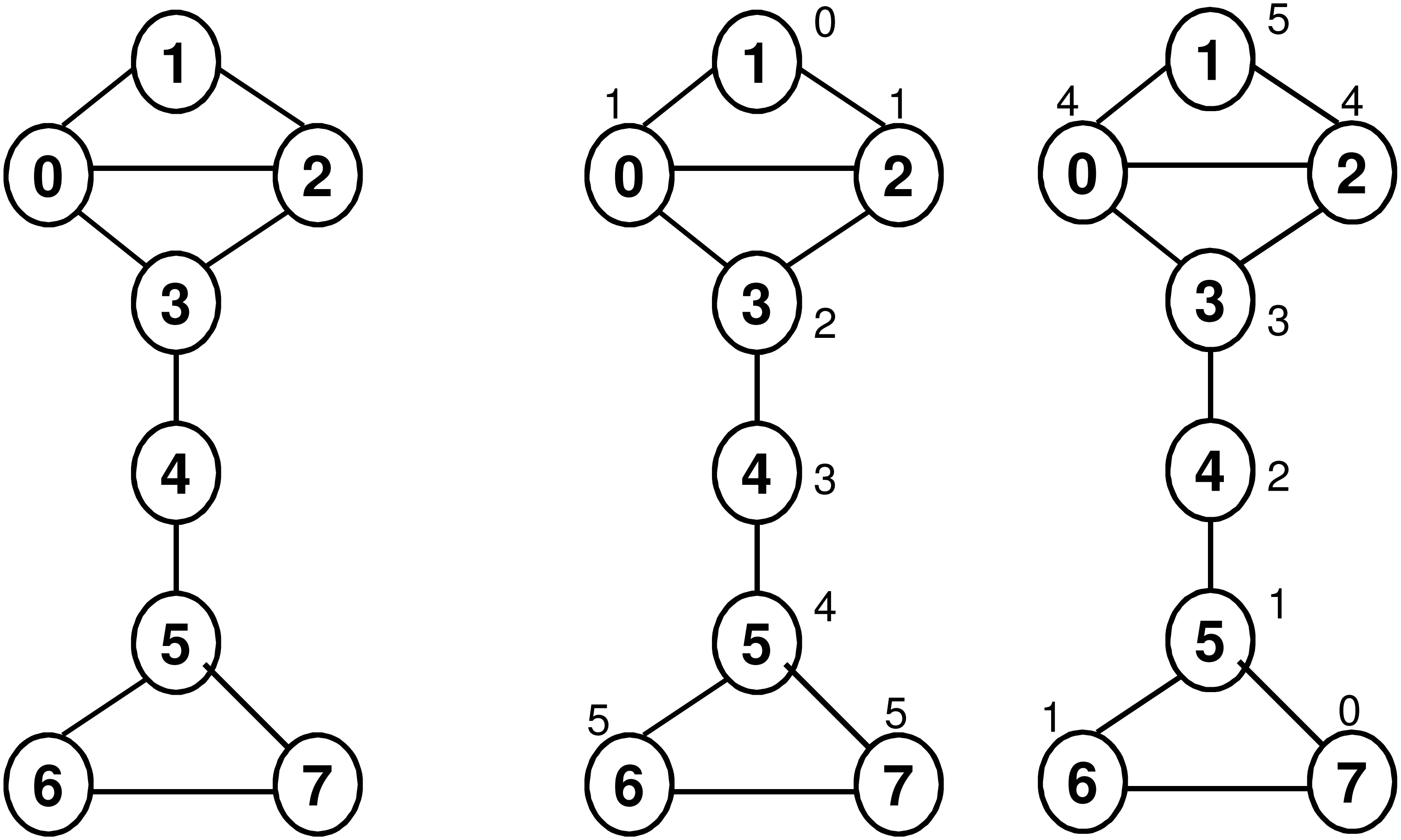
Levels of Vertices on the BFS tree



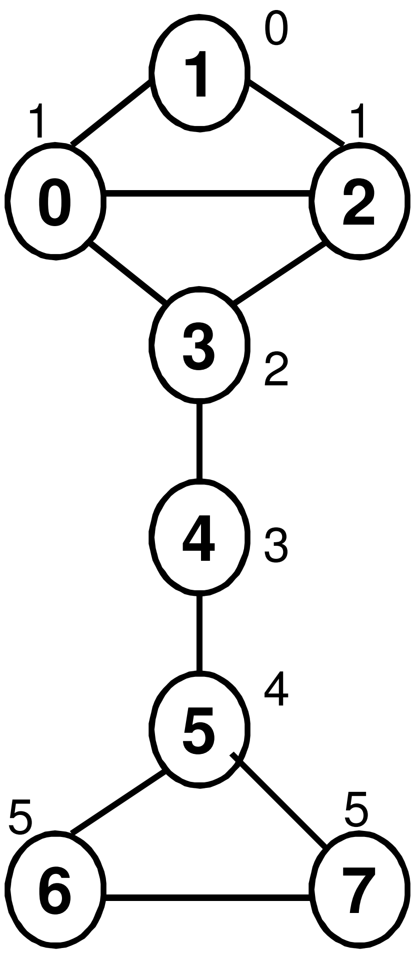


# shortest paths from the root to the other vertices

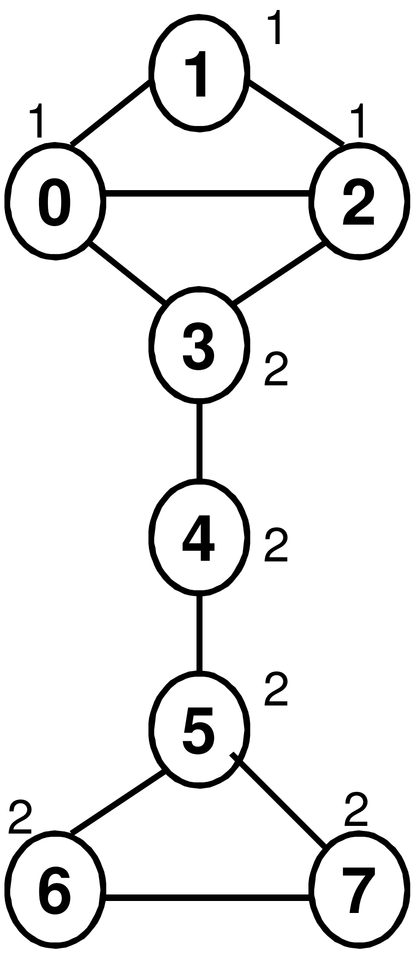




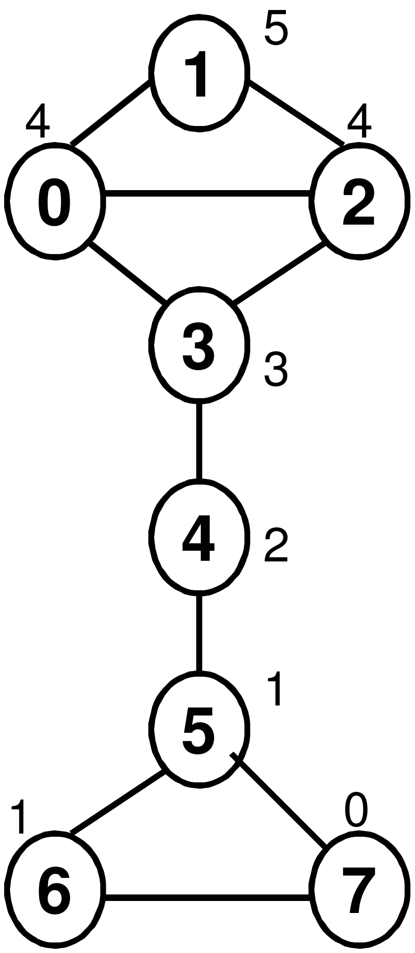
To determine how many
Shortest paths from nodes
1 to 7 that go through
node 4.



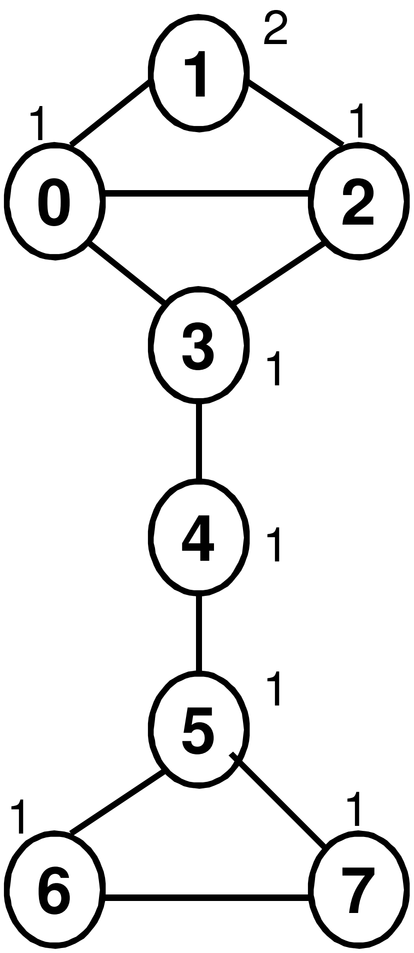
BFS Tree rooted at Vertex 1



# shortest paths from vertex 1 to the other vertices



BFS Tree rooted at Vertex 7



# shortest paths from vertex 7 to the other vertices

To determine how many Shortest paths from nodes 1 to 7 that go through node 4: = Max(2, 1) = 2

Subgraph Centrality

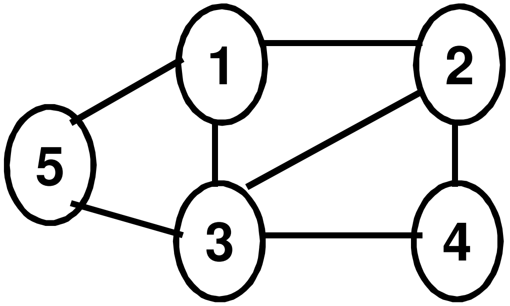
- The subgraph centrality of a node is a measure of the number of sub graphs a node is part of.
  - Gives more importance to the smaller sub graphs
  - Measured as the weighted sum of the number of closed walks of particular length (*l* = 1, 2, 3, ….) that a node is part of. The weights are 1/*l*!
  - For a given adjacency matrix A, A^{l} gives the number of closed walks of length *l* from a vertex to another vertex (incl. itself).

In closed form

$$\boxed{SubGC(i) = \left(e^{A}\right)_{ii} = \sum_{j=1}^{n} \left[\varphi_j(i)\right]^2 e^{\lambda_j}}$$

where φ_j(i) is the ith entry of the jth Eigenvector associated with Eigenvalue λ_j

Subgraph Centrality Example (2)



$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$\lambda 1 = -1.618$
$\lambda 2 = -1.473$
$\lambda 3 = -0.463$
$\lambda 4 = 0.618$
$\lambda 5 = 2.935$

Node IDs

Eigenvector entries		1	2	3	4	5	Eigenvalues	$e^{\lambda j}$
	1	-0.602	0.602	0	-0.372	0.372	λ1 = -1.618	0.2
	2	-0.138	-0.138	0.770	-0.429	-0.429	λ2 = -1.473	0.23
	3	0.510	0.510	-0.307	-0.439	-0.439	λ3 = -0.463	0.63
	4	-0.372	0.372	0	0.602	-0.602	λ4 = 0.618	1.852
	5	0.47	0.47	0.559	0.351	0.351	λ5 = 2.935	18.654

SubGC(Node 1) = { $(-0.602)^2$ * e^(-1.618) + $(-0.138)^2$ e^(-1.473) + $(0.51)^2$ e^(-0.463) + $(-0.372)^2$ * e^(0.618) + $(0.47)^2$ * e^(2.935) } = **4.62**

SubGC(Node 2) = { $(0.602)^2$ * e^(-1.618) + $(-0.138)^2$ e^(-1.473) + $(0.510)^2$ e^(-0.463) + $(0.372)^2$ * e^(0.618) + $(0.47)^2$ * e^(2.935) } = **4.62**

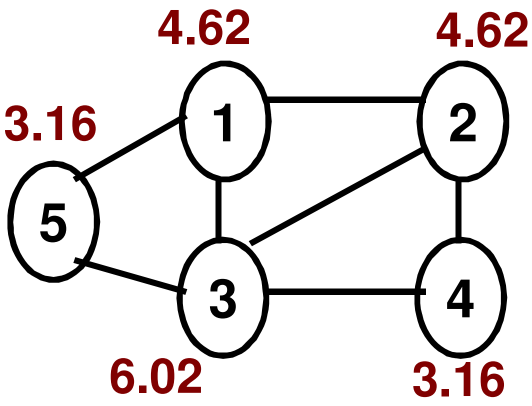
Node IDs

	1	2	3	4	5	Eigenvalues	$e^{\lambda j}$
1	-0.602	0.602	0	-0.372	0.372	λ1 = -1.618	0.2
2	-0.138	-0.138	0.770	-0.429	-0.429	λ2 = -1.473	0.23
3	0.510	0.510	-0.307	-0.439	-0.439	λ3 = -0.463	0.63
4	-0.372	0.372	0	0.602	-0.602	λ4 = 0.618	1.852
5	0.47	0.47	0.559	0.351	0.351	λ5 = 2.935	18.654

SubGC(Node 3) = { $(0)^2$ * e^(-1.618) + $(0.770)^2$ e^(-1.473) + $(-0.307)^2$ e^(-0.463) + $(0)^2$ * e^(0.618) + $(0.559)^2$ * e^(2.935) } = **6.02**

SubGC(Node 4) = { $(-0.372)^2$ * e^(-1.618) + $(-0.429)^2$ e^(-1.473) + $(-0.439)^2$ e^(-0.463) + $(0.602)^2$ * e^(0.618) + $(0.351)^2$ * e^(2.935) } = **3.16**

SubGC(Node 5) = { $(0.372)^2$ * e^(-1.618) + $(-0.429)^2$ e^(-1.473) + $(-0.439)^2$ e^(-0.463) + $(-0.602)^2$ * e^(0.618) + $(0.351)^2$ * e^(2.935) } = **3.16**



Average Subgraph Centrality

$$\langle SC\rangle = \frac{1}{N}\sum_{i=1}^{N} SC(i) = \frac{1}{N}\sum_{i=1}^{N} e^{\lambda_i}$$

For the example graph given here: <SC> = 4.32

Centrality Correlations

Network Graphs Analyzed (1)

- (i) *Zachary's Karate Club*: Social network of friendships (78 edges) between 34 members of a karate club at a US university in the 1970s

- (ii) *Dolphins' Social Network*: An undirected social network of frequent associations (159 edges) between 62 dolphins in a community living off Doubtful Sound, New Zealand

- (iii) *US Politics Books Network*: Nodes represent a total of 105 books about US politics sold by the online bookseller Amazon.com.
  - A total of 441 edges represent frequent co-purchasing of books by the same buyers, as indicated by the "customers who bought this book also bought these other books" feature on Amazon

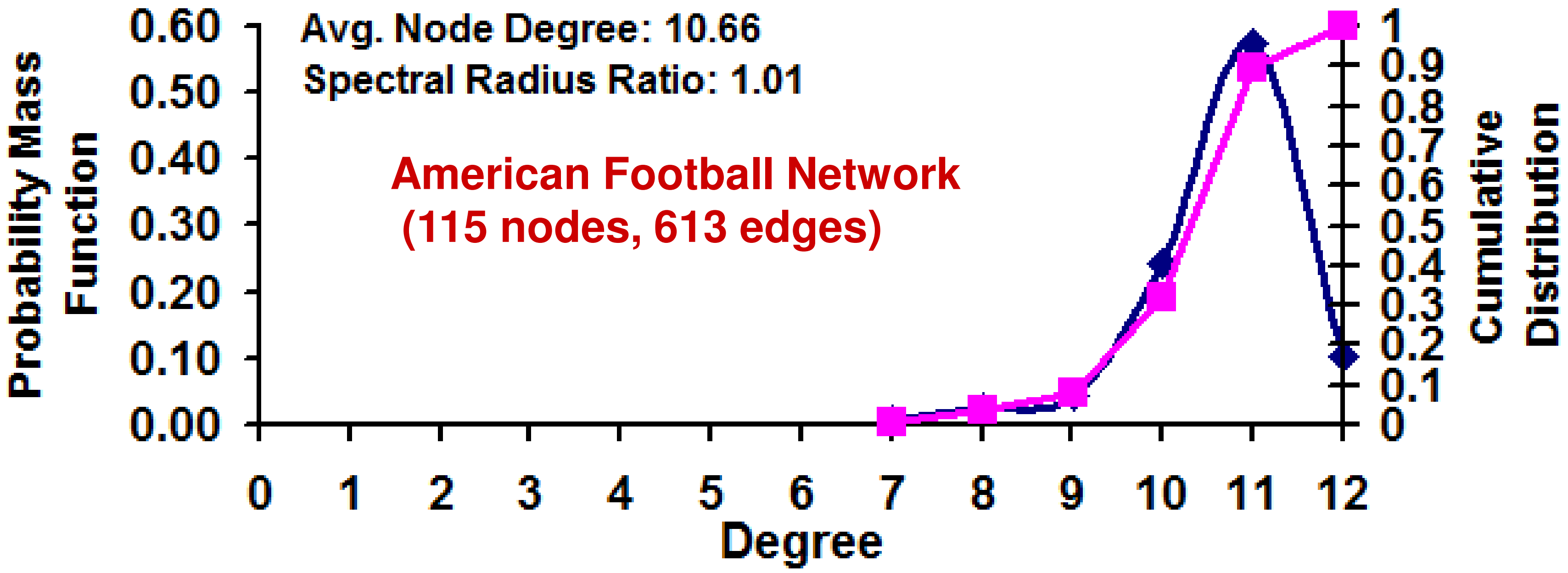
Network Graphs Analyzed (2)

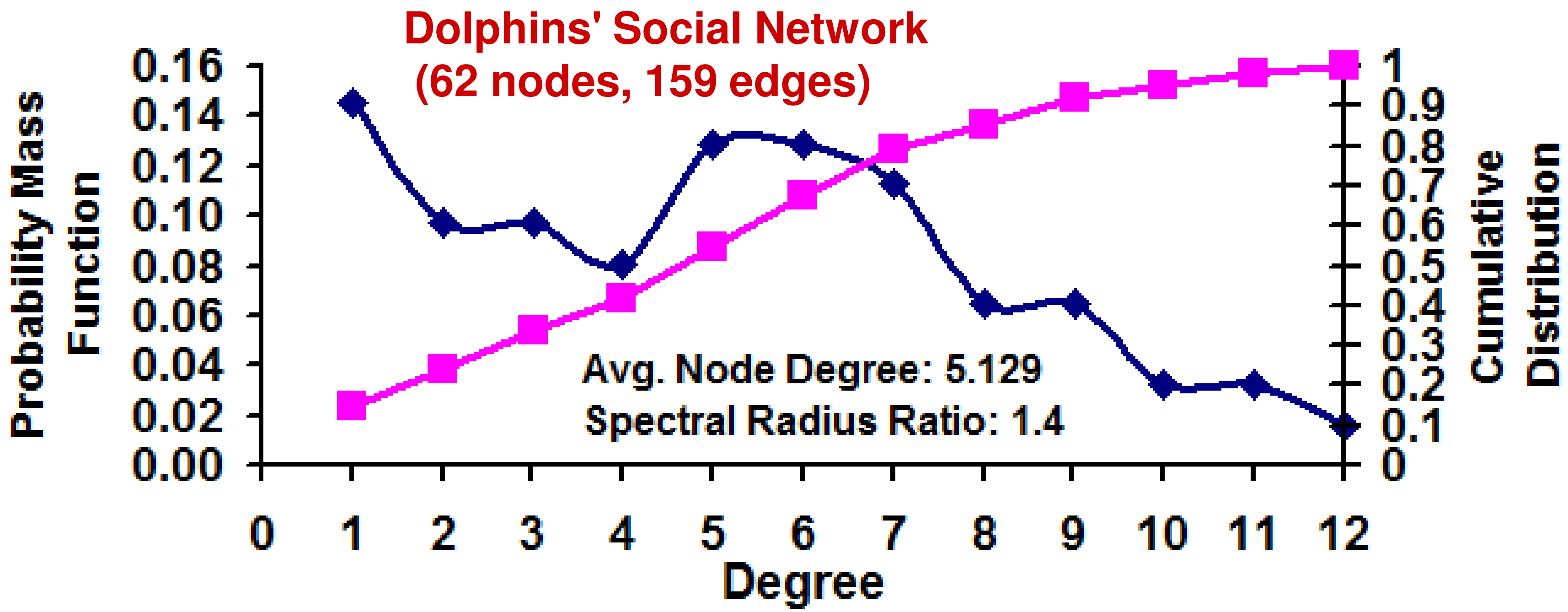
- (iv) ***Word Adjacencies Network***: This is a word co-appearance network representing adjacencies of common adjective and noun in the novel "David Copperfield" by Charles Dickens.
  - A total of 112 nodes represent the most commonly occurring adjectives and nouns in the book. A total of 425 edges connect any pair of words that occur in adjacent position in the text of the book

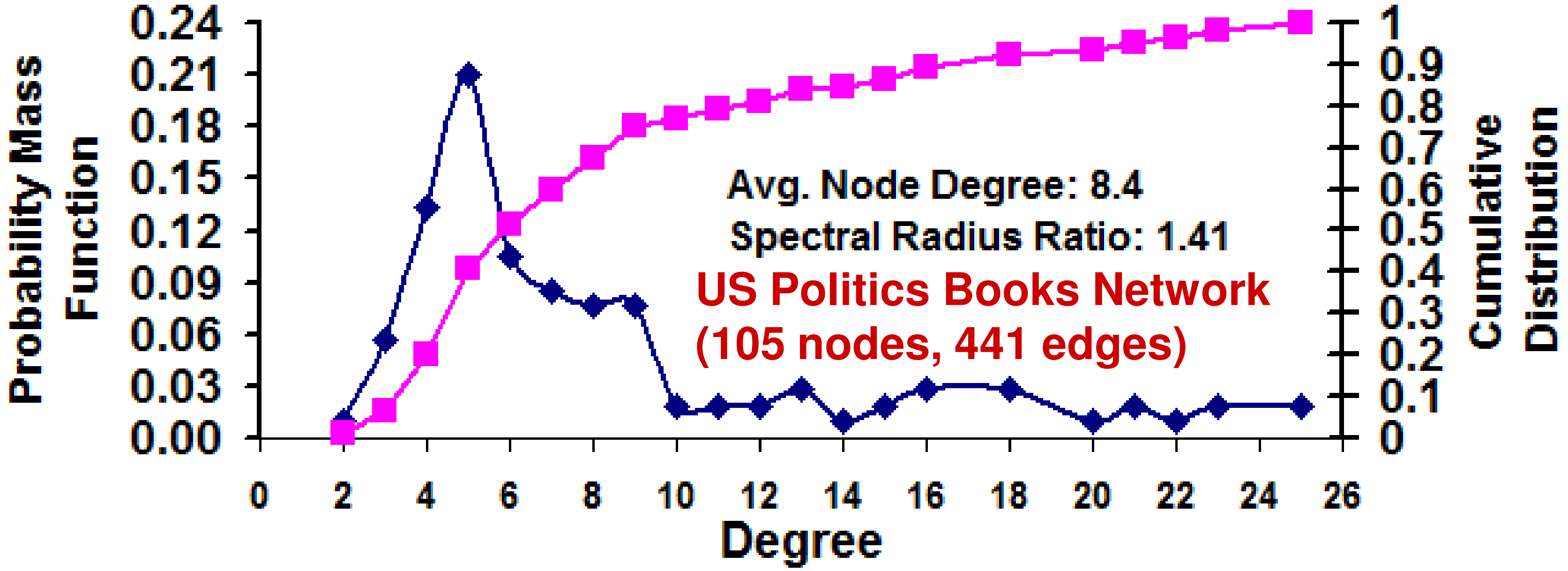
- (v) ***US College Football Network***: Network represents the teams that played in the Fall 2000 season of the US College Football games and their previous rivalry - nodes (115 nodes) are college teams and there is an edge (613 edges) between two nodes if and only if the corresponding teams have competed against each other earlier

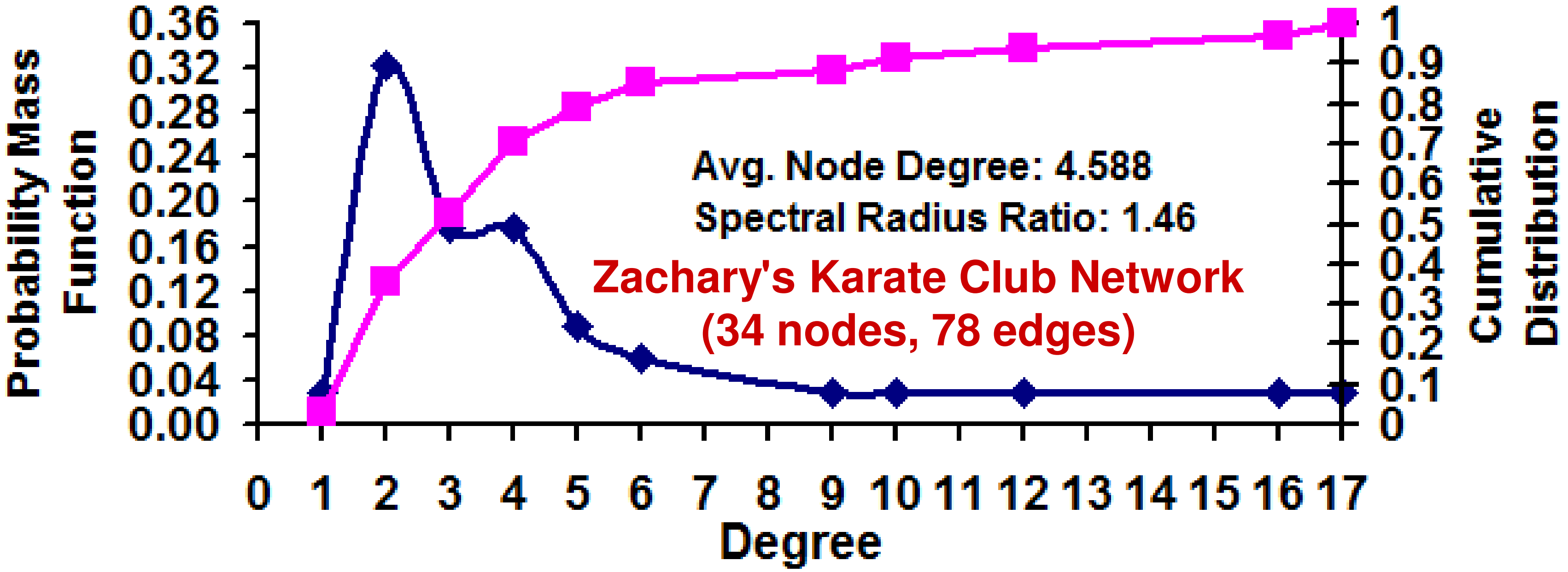
- (vi) ***US Airports* 1997 *Network***: A network of 332 airports in the United States (as of year 1997) wherein the vertices are the airports and two airports are connected with an edge (a total of 2126 edges) if there is at least one direct flight between them in both the directions.

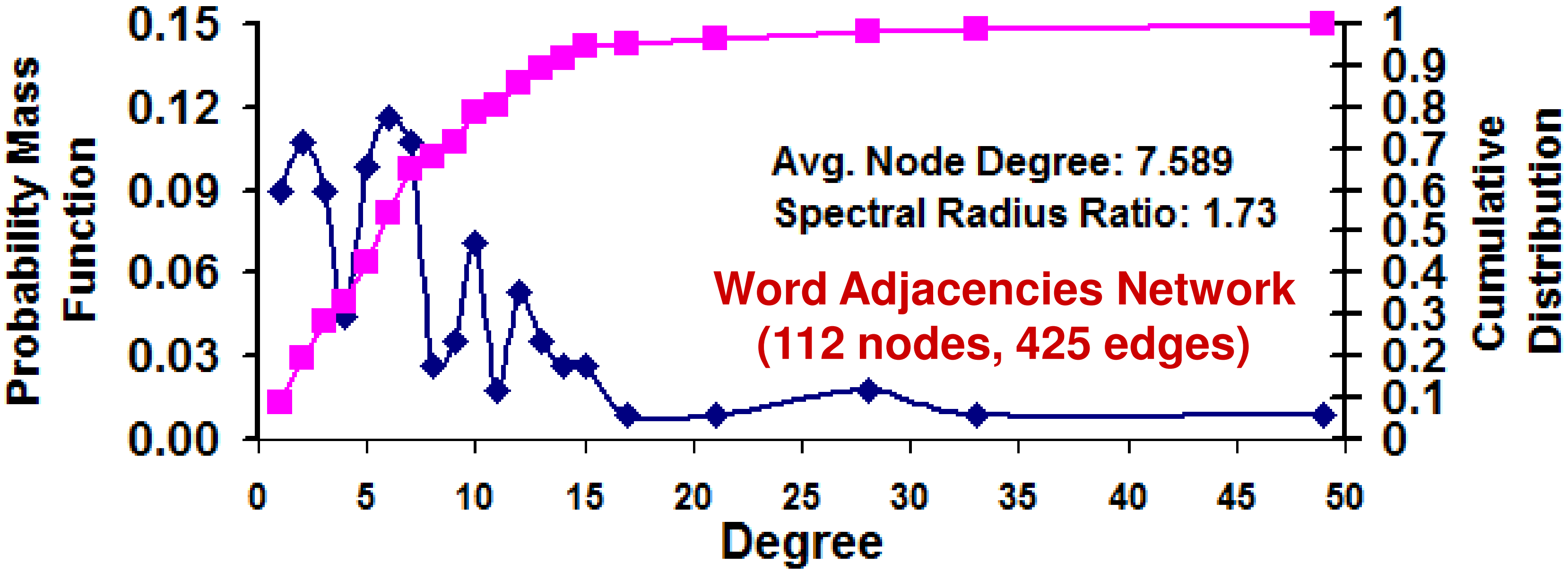
- **Spectral radius ratio**: Ratio of spectral radius (largest Eigenvalue based on the Adjacency matrix) to the average node degree.

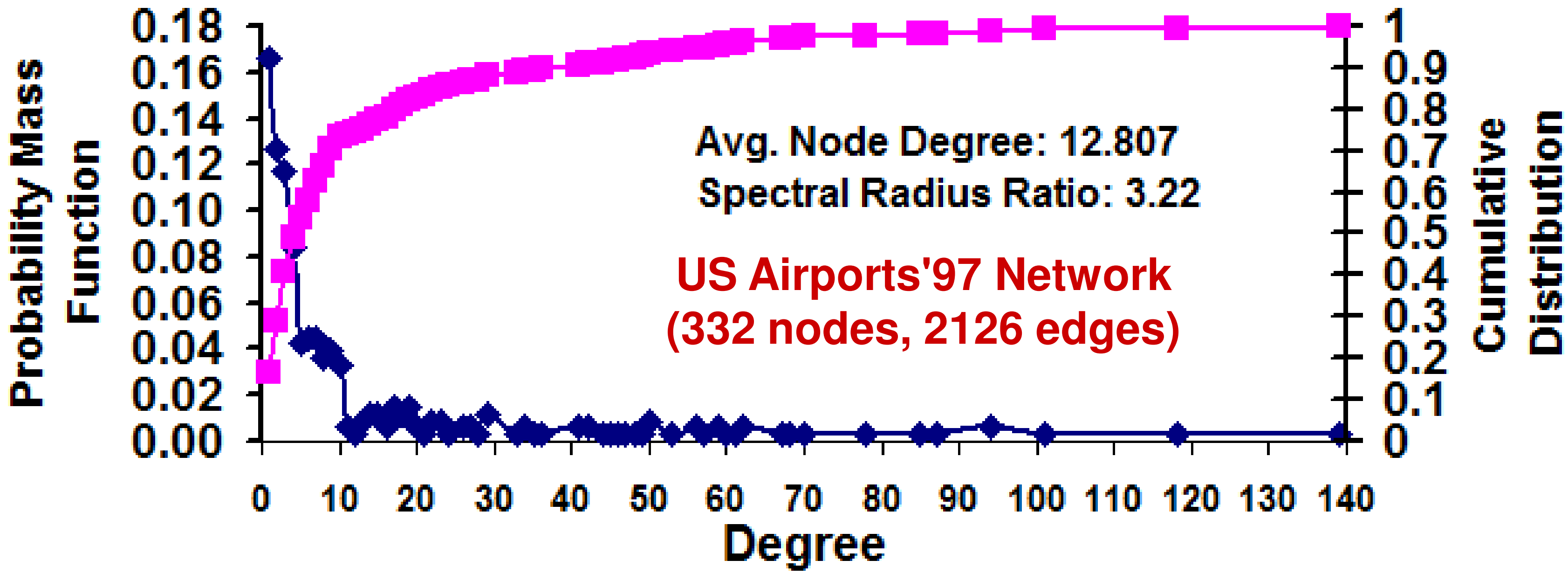

Avg. Node Degree: 10.66
Spectral Radius Ratio: 1.01
American Football Network
(115 nodes, 613 edges)
Probability Mass Function
0.60
0.50
0.40
0.30
0.20
0.10
0.00
Cumulative Distribution
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
1
2
3
4
5
6
7
8
9
10
11
12
Degree


Dolphins' Social Network
(62 nodes, 159 edges)
Avg. Node Degree: 5.129
Spectral Radius Ratio: 1.4
Probability Mass Function
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
Cumulative Distribution
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
1
2
3
4
5
6
7
8
9
10
11
12
Degree


Probability Mass Function
0.24
0.21
0.18
0.15
0.12
0.09
0.06
0.03
0.00
0
2
4
6
8
10
12
14
16
18
20
22
24
26
Degree
Cumulative Distribution
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Avg. Node Degree: 8.4
Spectral Radius Ratio: 1.41
US Politics Books Network
(105 nodes, 441 edges)


Probability Mass Function
0.36
0.32
0.28
0.24
0.20
0.16
0.12
0.08
0.04
0.00
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
Degree
Cumulative Distribution
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Avg. Node Degree: 4.588
Spectral Radius Ratio: 1.46
Zachary's Karate Club Network
(34 nodes, 78 edges)


Probability Mass Function
Cumulative Distribution
Degree
0.15
0.12
0.09
0.06
0.03
0.00
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
5
10
15
20
25
30
35
40
45
50
Avg. Node Degree: 7.589
Spectral Radius Ratio: 1.73
Word Adjacencies Network
(112 nodes, 425 edges)


Probability Mass Function
Cumulative Distribution
Degree
0.18
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
Avg. Node Degree: 12.807
Spectral Radius Ratio: 3.22
US Airports'97 Network
(332 nodes, 2126 edges)

Real-World Networks

Network Index	Real-World Network Graph	# Nodes	# Edges	Spectral Radius Degree Ratio
(i)	Zachary's Karate Club Network	34	78	1.46
(ii)	Dolphins' Social Network	62	159	1.40
(iii)	US Politics Books Network	105	441	1.41
(iv)	Word Adjacencies Network	112	425	1.73
(v)	American College Football Network	115	613	1.01
(vi)	US Airports 1997 Network	332	2126	3.22

Correlation Coefficient

$$CorrCoeff(X,Y)=\frac{\sum_{ID=1}^{n}(X[ID]-\overline{X})*(Y[ID]-\overline{Y})}{\sqrt{\sum_{ID=1}^{N}(X[ID]-\overline{X})^2}\sqrt{\sum_{ID=1}^{N}(Y[ID]-\overline{Y})^2}}$$

High: ≥ 0.75

Moderate: 0.50 – 0.74

Low < 0.50

Correlation Coefficients: Centrality Metrics for Real-World Network Graphs (Networks listed in the increasing order of Number of Nodes)

Net #	Deg EVC	Deg BWC	Deg ClC	Deg FarC	EVC BWC	EVC ClC	EVC FarC	BWC ClC	BWC FarC	ClC FarC
(i)	0.90	0.92	0.77	0.77	0.79	0.91	0.90	0.72	0.72	0.99
(ii)	0.77	0.60	0.71	0.73	0.33	0.71	0.68	0.67	0.71	0.99
(iii)	0.93	0.71	0.58	0.59	0.58	0.53	0.53	0.78	0.79	0.99
(iv)	0.95	0.92	0.84	0.84	0.82	0.93	0.92	0.66	0.66	0.99
(v)	0.87	0.28	0.29	0.29	0.19	0.28	0.28	0.82	0.83	0.99
(vi)	0.95	0.70	0.80	0.80	0.52	0.85	0.84	0.49	0.51	0.99

Correlation Coefficients: Centrality Metrics for Real-World Network Graphs (Networks listed in the increasing order of Spectral Radius Degree Ratio)

Net #	Deg EVC	Deg BWC	Deg ClC	Deg FarC	EVC BWC	EVC ClC	EVC FarC	BWC ClC	BWC FarC	ClC FarC
(v)	0.87	0.28	0.29	0.29	0.19	0.28	0.28	0.82	0.83	0.99
(ii)	0.77	0.60	0.71	0.73	0.33	0.71	0.68	0.67	0.71	0.99
(iii)	0.93	0.71	0.58	0.59	0.58	0.53	0.53	0.78	0.79	0.99
(i)	0.90	0.92	0.77	0.77	0.79	0.91	0.90	0.72	0.72	0.99
(iv)	0.95	0.92	0.84	0.84	0.82	0.93	0.92	0.66	0.66	0.99
(vi)	0.95	0.70	0.80	0.80	0.52	0.85	0.84	0.49	0.51	0.99

Observations: Centrality Correlations

- The degree-based centrality metrics (degree and Eigenvector centralities) are consistently highly correlated for all the six real-world network graphs considered.
- Likewise, though the shortest path-based centrality metrics are only moderately correlated for most of the real-world network graphs, we observe such a correlation to be consistent across the network graphs without much variation in the correlation coefficient values.

- The level of correlation between a degree-based centrality metric and a shortest path-based centrality metric increases with increase in variation of node degree:
  - the two classes of metrics are poorly correlated in regular/random networks and are at the low-end of moderate-level of correlation for real-world networks that are less scale-free.
  - As the real-world networks get more scale-free, the level of correlation between the two classes of centrality metrics is likely to increase.

- The shortest path-based centrality metrics correlate better for regular/random networks and the level of correlation decreases as the networks get increasingly scale-free.

Link Analysis-based Ranking

- We want to rank a node in a graph based on the number of edges pointing to it and/or leaving it as well as based on the rank of the nodes at the other end of these edges.

- Used primarily in web search
  - We model the web as a graph: the pages as nodes and the edges are directed edges – a page citing (having a link to) another page.

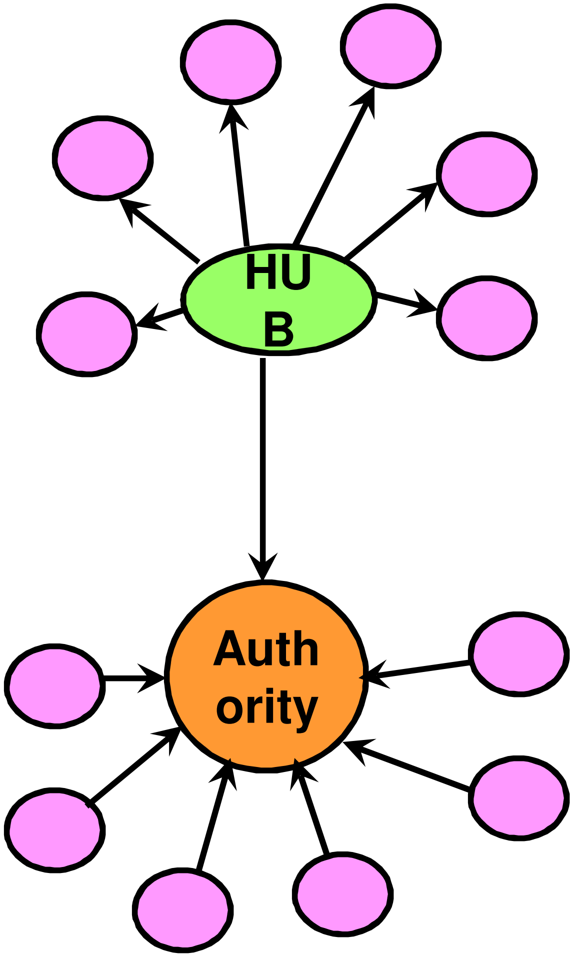
- Hubs and Authorities (HITS) algorithm
- PageRank algorithm

Hypertext Induced Topic Search (HITS) Algorithm

- **Hub:** Node that points to lots of pages
  - Yahoo like directory
- **Authority:** Node to which several other nodes point to
  - The larger the number of nodes pointing to a node, the more authoritative is the view presented by a node on a particular subject
- The HITS algorithm assigns **two scores for each page**:
  - **Authority:** an estimate of the value of the contents of the page
  - **Hub:** an estimate of the value of its links to other pages

- A page is considered to be **more authoritative** if it is referenced by many hub pages that are relevant to a search query
- A page is a **hub page** for a search query if it points to many authoritative pages for that query

- Good **authoritative** and **hub** pages reinforce one another.



A variant of HITS is used by Ask.com

Finding Pages for a Query in HITS

- **Initial Work**
- Step 1: Submit query *q* to a similarity-based engine and record the top *n*, i.e., the root set RS(q) pages.
- Step 2: Expand set RS(q) into the base set BS(q) to include pages pointed by RS(q) pages
- Step 3: Also include into BS(q), the pages pointing to RS(q) pages.
- **Run the HITS algorithm**
  - For each page pj, compute the authority and hub score of pj through a sequence of iterations.
- **After obtaining the final authority and hub scores** for each page, display the search results in the decreasing order of the authority scores. Pages having zero authority scores (nodes with no incoming links – strictly hubs) are listed in the decreasing order of their hub scores.
  - Note: nodes that are strictly hubs still contribute to the authority of the nodes that it points to.

HITS Algorithm

- Let E be the set of links in BS(q) and a link from page pi to pj is denoted by the pair (i, j).
- A: Authority Update Step H: Hub Update Step

$$a(p_j) = \sum_{(i,j)\in E} h(p_i)$$

$$h(p_j) = \sum_{(j,k)\in E} a(p_k)$$

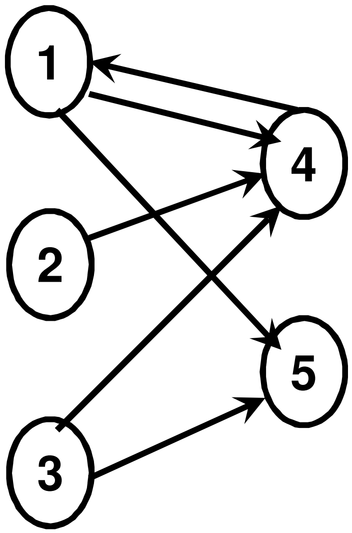
- After each iteration i, we scale the ‘a’ and ‘h’ values:

$$a^{(i)}(p_j) = \frac{a^{(i)}(p_j)}{\sqrt{\sum_k \left(a^{(i)}(p_k)\right)^2}}$$

$$h^{(i)}(p_j) = \frac{h^{(i)}(p_j)}{\sqrt{\sum_k \left(h^{(i)}(p_k)\right)^2}}$$

- As can be noted above, the two steps are interwined: one uses the values computed from the other.
  - In this course, we will follow the asynchronous mode of computation, according to which the authority values are updated first for a given iteration i and then the hub values are updated.
    - The hub values at iteration *i* are using the authority values just computed in iteration *i* (rather than iteration *i* – 1).

HITS Example (1)



Initial
a = [1 1 1 1 1] h = [1 1 1 1 1]

It # 1
a = [1 0 0 3 2] h = [5 3 5 1 0]
After Normalization,
a = [0.26 0 0 0.80 0.53] h = [0.64 0.38 0.64 0.12 0]

It # 2
a = [0.12 0 0 1.66 1.28] h = [2.94 1.66 2.94 0.12 0]
After Normalization,
a = [0.057 0 0 0.79 0.61] h = [0.66 0.37 0.66 0.027 0]

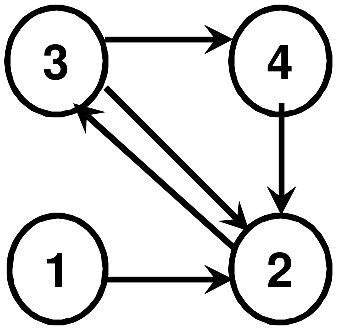
It # 3
a = [0.027 0 0 1.69 1.32] h = [3.01 1.69 3.01 0.027 0]
After Normalization,
a = [0.0126 0 0 0.79 0.61] h = [0.66 0.37 0.66 0.006 0]

It # 4
a = [0.006 0 0 1.69 1.32] h = [3.01 1.69 3.01 0.006 0]
After Normalization,
a = [0.003 0 0 0.79 0.61] h = [0.66 0.37 0.66 0.001 0]

Order Pages Listed after Search

4
5
1
3
2

HITS Example (2)



Initial
a = [1 1 1 1] h = [1 1 1 1]

It # 1
a = [0 3 1 1] h = [3 1 4 3]
After Normalization,
a = [0 0.91 0.30 0.30] h = [0.51 0.17 0.68 0.51]

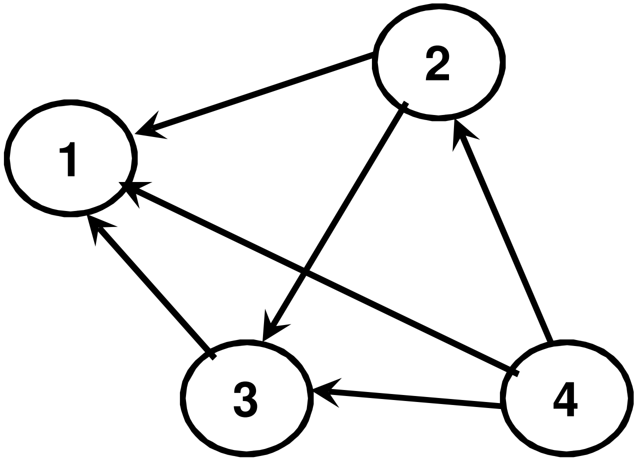
It # 2
a = [0 1.70 0.17 0.68] h = [1.70 0.17 2.38 1.70]
After Normalization,
a = [0 0.92 0.09 0.37] h = [0.50 0.05 0.70 0.50]

It # 3
a = [0 1.70 0.05 0.70] h = [1.70 0.05 2.4 1.70]
After Normalization,
a = [0 0.92 0.027 0.38] h = [0.50 0.014 0.70 0.50]

It # 4
a = [0 1.70 0.014 0.70] h = [1.70 0.014 2.4 1.70]
After Normalization,
a = [0 0.92 0.008 0.38] h = [0.50 0.004 0.71 0.50]

Order Pages Listed after Search

2
4
3
1



HITS Example (3)

Initial
a = [1 1 1 1] h = [1 1 1 1]

It # 1
a = [3 1 2 0] h = [0 5 3 6]
After Normalization,
a = [0.80 0.27 0.53 0] h = [0 0.59 0.36 0.72]

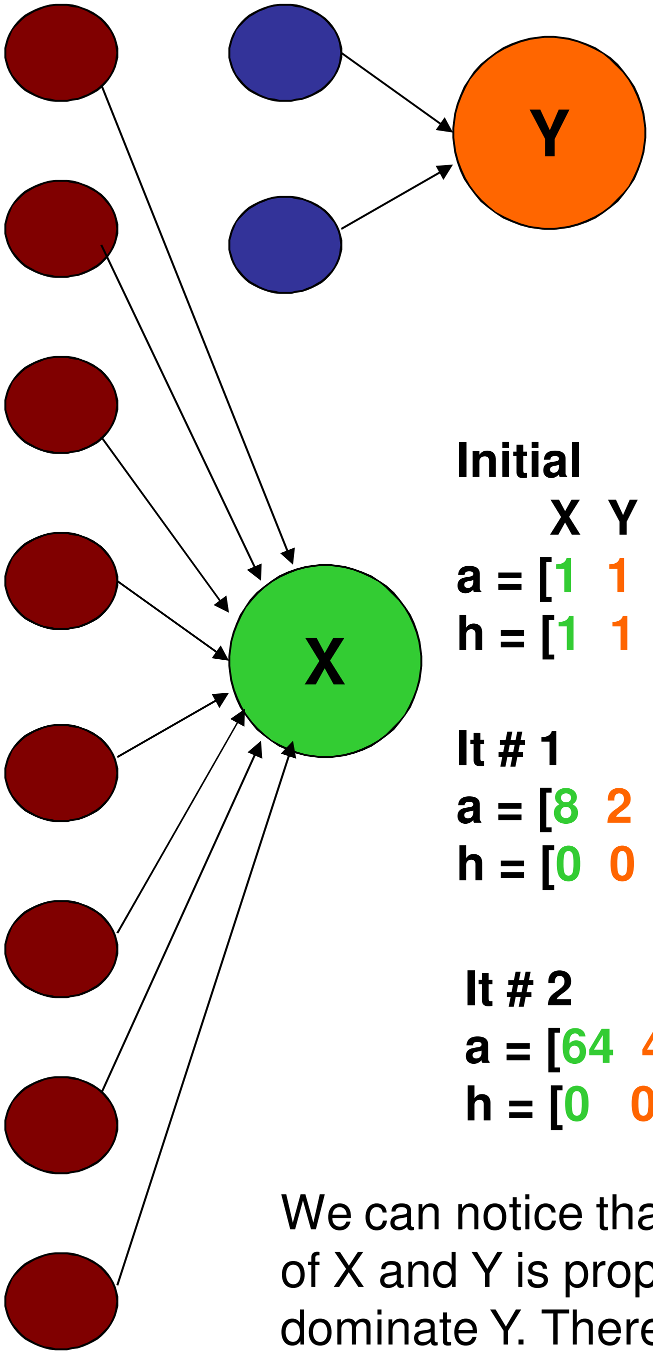
It # 2
a = [1.67 0.72 1.31 0] h = [0 2.98 1.67 3.7]
After Normalization,
a = [0.745 0.32 0.58 0] h = [0 0.59 0.33 0.73]

It #3
a = [1.65 0.73 1.32 0] h = [0 2.97 1.65 3.7]
After Normalization,
a = [0.74 0.32 0.59 0] h = [0 0.59 0.33 0.73]

Order Pages Listed after Search

1
3
2
4

HITS Example (4)



- Assume ‘x’ web-pages point to page X and ‘y’ pages point to page Y, where x >> y. What happens with the hubs and authority values of X and Y respectively?
- Assume no normalization is done at the end of each iteration.

Initial

X Y ←x web-pages →<-y ->

a = [1 1 1 1 1 1 1 1 1 1 1 1]

h = [1 1 1 1 1 1 1 1 1 1 1 1]

It # 1

a = [8 2 0 0 0 0 0 0 0 0 0 0]

h = [0 0 8 8 8 8 8 8 8 8 2 2]

It # 2

a = [64 4 0 0 0 0 0 0 0 0 0 0]

h = [0 0 64 64 64 64… 64 4 4]

We can notice that with each iteration i, the ratio of the authority values of X and Y is proportional to (x/y)^i. After a while, X will completely dominate Y. There is no change in the hub values of X and Y though.

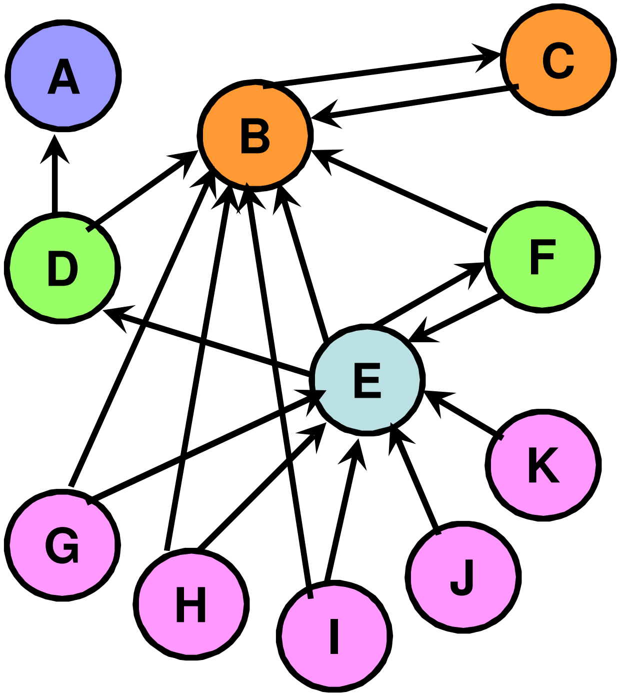
PageRank

- The basic idea is to analyze the link structure of the web to figure out which pages are more authoritative (important) in terms of quality.
- It is a content-independent scheme.
- If Page A has a hyperlink to Page B, it can be considered as a vote of A for B.
  - If multiple pages link to B, then page B is likely to be a good page.
- A page is likely to be good if several other good pages link to it (a bit of recursive definition).
  - Not all pages that link to B are of equal importance.
  - A single link from CNN or Yahoo may be worth several times
- The web pages are first searched based on the content. The retrieved web pages are then listed based on their rank (computed on the original web, unlike HITS that is run on a graph of the retrieved pages).
- The Page Rank of the web pages are indexed (recomputed) for every regular time period.

PageRank (Random Web Surfer)

- Web – graph of pages with the hyperlinks as directed edges.
- Analogy used to explain PageRank algorithm (Random Web Surfer)
- User starts browsing on a random page
- Picks a random out-going link listed in that page and goes there (with a probability 'd', also called damping factor)
  - Repeated forever
- The surfer jumps to a random page with probability 1-d.
  - Without this characteristic, there could be a possibility that someone could just end up oscillating between two pages B and C as in the traversing sequence below for the graph shown aside:

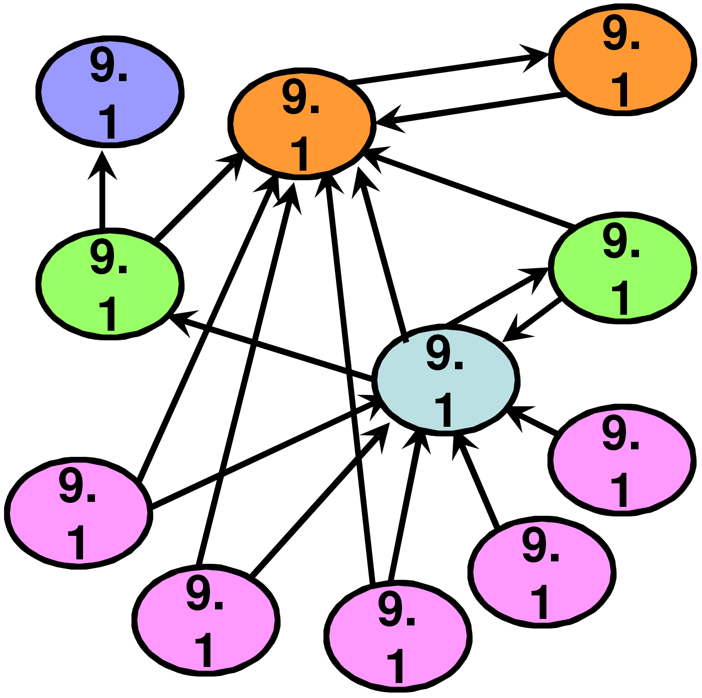
    G → E → F → E → D → B → C



Lets say d = 0.85.
To decide the next page to move, the surfer simply generates a random number, r. If r <= 0.85, then the surfer randomly chooses an out-going link from the existing page. Otherwise, jumps to a randomly chosen page among all the pages, including the current page.

PageRank Algorithm

- PageRank of Page X is the probability that the surfer is at page X at a randomly selected time.
  - Basically the proportion of time, the surfer would spend at page X.
- **PageRank Algorithm**
- **Initial:** Every node in the graph gets the same pagerank. PR(X) = 100% / N, where N is the number of nodes.
- At any time, at the end of each iteration, the page rank of all nodes add up to 100%.
- Actually, the initial pagerank value of a node is the pagerank at any time, if there are no edges in the graph. We have 100% / N chance of jumping to any node in the graph at any time.



Initial PageRank of Nodes

PageRank Algorithm

Page Rank of Node X

$$PR(x) = \frac{(1-d)*100}{N} + d \sum_{y->x} \frac{PR(y)}{Out(y)}$$

Assuming there are NO Sink nodes

- Page Rank of Node X is the probability of being at node X at the current time.
- How can we visit node X from where we are?
  - **(1-d) term: Random Jump:** The probability of ending up at node X because of a random jump from some node, including node X, is 1/N.
  - However, such a random jump itself could occur with a probability of (1-d).
  - This amounts to a probability of (1-d)/N to be at node X due to a random jump.

PageRank Algorithm

Page Rank of Node X

$$PR(x) = \frac{(1-d)*100}{N} + d\sum_{y->x}\frac{PR(y)}{Out(y)}$$

Assuming there are NO Sink nodes

- Page Rank of Node X is the probability of being at node X at the current time.
- How can we visit node X from where we are?
  - **<u>d term: Edge Traversal from a Neighbor:</u>**
  - We could visit node X from one of the nodes that point to node X.
  - Lets say, we are at node Y in the previous iteration. The probability of being at node Y in the previous iteration is PR(Y). We can visit any of Y's neighbors.
  - The probability of visiting node X among the Out(Y) out-going links of node Y is PR(Y) * (1 / Out(Y)) = PR(Y) / Out(Y).
  - Likewise, we could visit X from any of its neighbors.
  - All the probabilities of visiting X from any of its neighbors have to be added, because visiting X from any of its neighbors is independent of the neighbors.
  - The whole event of visiting from a neighbor occurs with a prob. 'd'

PageRank

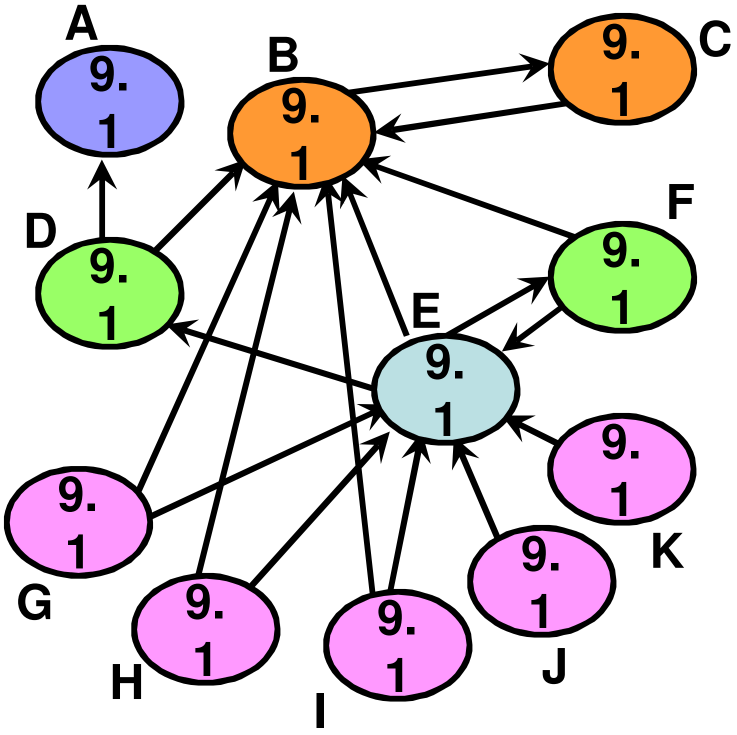
- Since Page Rank PR(X) denotes the probability of being at node X at any time, the sum of the Page Ranks of all the nodes at any time should be equal to 1.

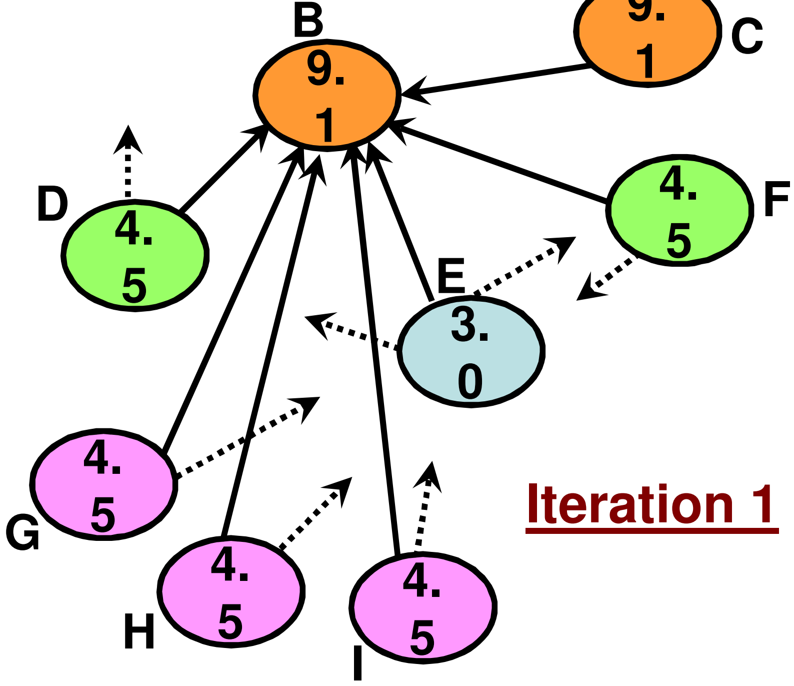
- We can also interpret the traversal from a node Y to node X as node Y contributing a part of its PR to node X (node Y equally shares its PR to the nodes connected to it through its out-going links).

- Implementation:
  - Note that (unlike HITS) we need to use the page rank values of the nodes from the previous iteration to update the page rank values of the nodes in the current iteration.
    - **Need to maintain two arrays at any time t: $PR^{(t-1)}$ and $PR^{(t)}$**

Calculating PageRank of Node B

Initial PageRank of Nodes





Iteration 1

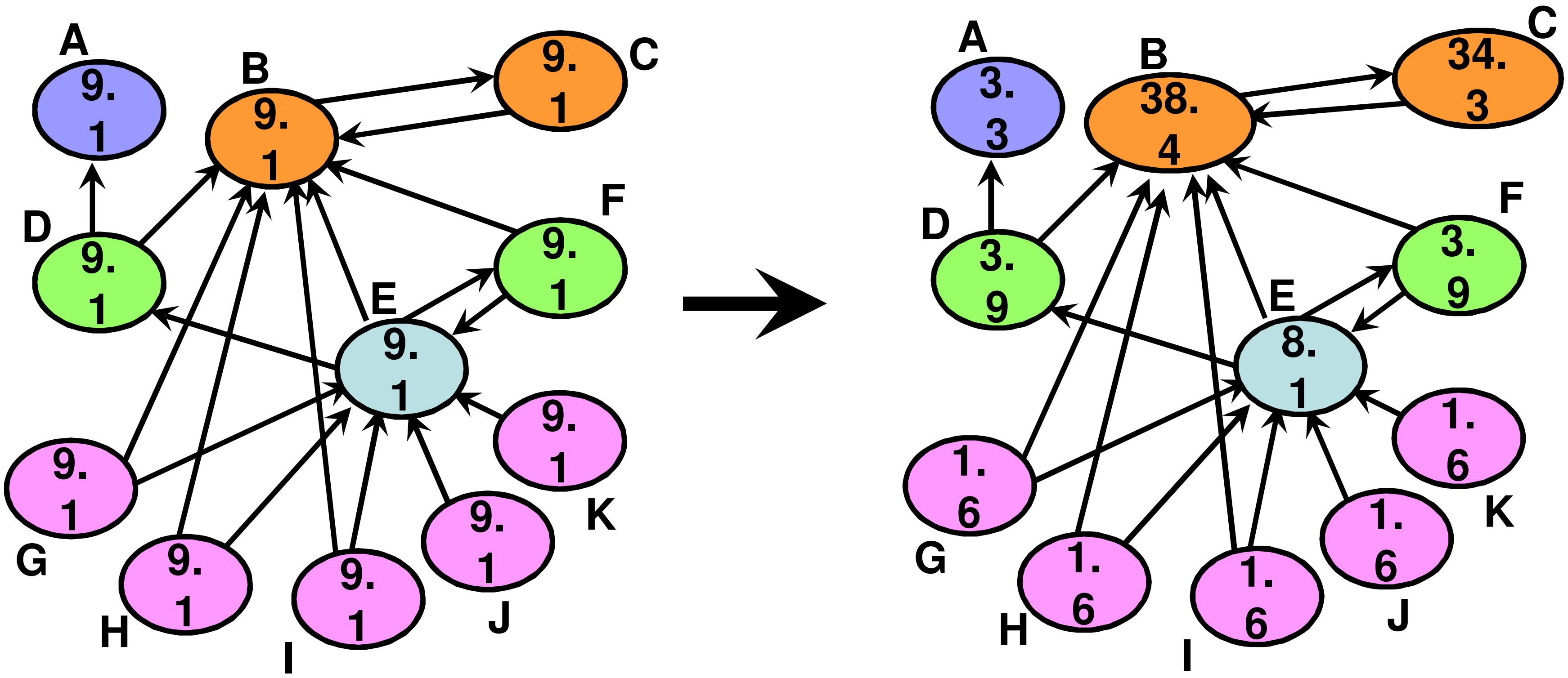
Assume the damping factor d = 0.85

For any iteration,

PR(B) = 0.15 * 9.1 +
0.85 * [PR(C) + ½ PR(D) +
⅓ PR(E) + ½ PR(F) +
½ PR (G) + ½ PR(H) + ½ PR(I)]

For Iteration 1,
Substituting the PR values of the nodes (initial values),
we get PR(B) ≈ 31

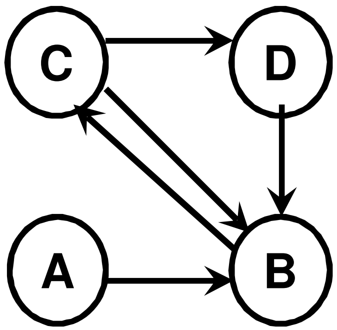
Final PageRank Values for the Sample Graph



PageRank: More Observations

- Algorithm converges (few iterations sufficient)
- For an arbitrary graph, it is pretty difficult to figure out the final page rank values of the nodes.
- Certain inferences could be however made.
- For our sample graph:
  - For nodes that do not have any in-links pointing to them,

    the only way we will end up at these nodes is through a random jump: this happens with a probability (1-d)/N.
    In our case, it is (1-0.85)* 100/11 = 1.6%.
  - Two nodes with links from the same node (symmetric in-links) have the same PR. (nodes D and F) and it will be higher than those nodes without any in-links.
  - One in-link from a node with high PR value contributes significantly to the PR value of a node compared to the in-links from several low PR nodes.
    - In our sample graph, an in-link from node B contributes significantly for node C compared to the several in-links that node E gets from the low-PR nodes. So, the quality of the in-links matters more than the number of in-links.



Note that there are NO sink nodes
(nodes without any out-going links)

Assume damping
Factor d = 0.85

PR(A) = (1-d)*100/4
PR(B) = (1-d)*100/4 + d*[PR(A) + 1/2 * PR(C) + PR(D)]
PR(C) = (1-d)*100/4 + d*[PR(B)]
PR(D) = (1-d)*100/4 + d*[1/2*PR(C)]

Initial
PR(A) = 25
PR(B) = 25
PR(C) = 25
PR(D) = 25

It # 1
PR(A) = 3.75
PR(B) = 56.88
PR(C) = 25
PR(D) = 14.38

It # 2
PR(A) = 3.75
PR(B) = 29.79
PR(C) = 52.10
PR(D) = 14.38

It # 3
PR(A) = 3.75
PR(B) = 41.30
PR(C) = 29.07
PR(D) = 25.89

It # 4
PR(A) = 3.75
PR(B) = 41.29
PR(C) = 38.86
PR(D) = 16.10

It # 5
PR(A) = 3.75
PR(B) = 37.14
PR(C) = 38.85
PR(D) = 20.27

It # 6
PR(A) = 3.75
PR(B) = 40.68
PR(C) = 35.32
PR(D) = 20.26

It # 7
PR(A) = 3.75
PR(B) = 39.17
PR(C) = 38.33
PR(D) = 18.76

It # 8
PR(A) = 3.75
PR(B) = 39.17
PR(C) = 37.04
PR(D) = 20.04

It # 9
PR(A) = 3.75
PR(B) = 39.71
PR(C) = 37.04
PR(D) = 19.49

It # 10
PR(A) = 3.75
PR(B) = 39.25
PR(C) = 37.5
PR(D) = 19.49

Ranking
B
C
D
A

Page Rank Example (1)

Page Rank: Graph with Sink Nodes Motivating Example

- Consider the graph: A → B
- Let d = 0.85
- PR(A) = 0.15*100/2 PR(B) = 0.15*100/2 + 0.85*PR(A)
- Initial: PR(A) = 50, PR(B) = 50
- Iteration 1:
  - PR(A) = 0.15*100/2 = 7.5
  - PR(B) = 0.15*100/2 + 0.85 * 50 = 50.0
  - PR(A) + PR(B) = 57.5
  - Note that the PR values do not add up to 100.
  - This is because, B is not giving back the PR that it receives from A to any other node in the graph. The (0.85*50 = 42.5) value of PR that B receives from A is basically lost.
  - Once we get to B, there is no way to get out of B other than random jump to A and this happens only with probability (1-d).

Page Rank: Sink Nodes (Solution)

- Assume implicitly that the sink node is connected to every node in the graph (including itself).
  - The sink node equally shares its PR with every node in the graph, including itself.
  - If *z* is a sink node, with the above scheme, out(z) = N, the number of nodes in the graph.
- The probability of getting to node X at a given time is still the two terms below:
  - Random jump from any node (probability, 1-d)
  - Visit from a node with in-link to node X (probability, d)

Page Rank of Node X

$$PR(x) = \frac{(1-d)*100}{N} + d\sum_{y->x}\frac{PR(y)}{Out(y)} + \frac{d}{N}\sum_{z->\varphi}PR(z)$$

Explicit out-going links to certain nodes

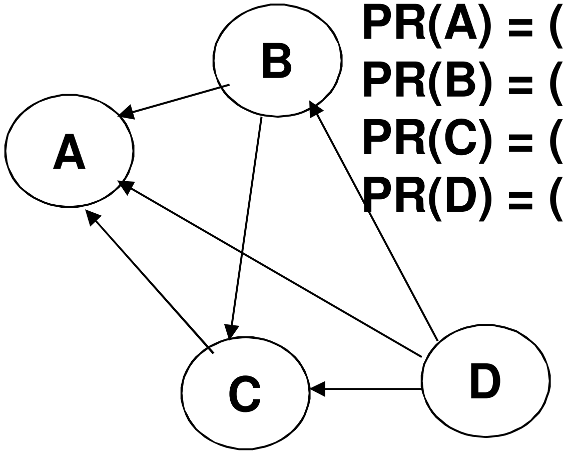
Implicit out-going links to all nodes (sink nodes)

the second term of the original Page Rank formula is now broken between that of nodes with explicit out-going links to one or more selected nodes and the sink nodes with implicit out-going links to all nodes.

Consolidated PageRank Formula

$$PR(x) = \frac{(1-d)*100}{N} + d \sum_{y->x} \frac{PR(y)}{Out(y)} + \frac{d}{N} \sum_{z->\varphi} PR(z)$$

Page Rank Example (2)



PR(A) = (1-d)*100/4 + d [PR(B)/2 + PR(C)/1 + PR(D)/3] + (d/4)*[PR(A)]
PR(B) = (1-d)*100/4 + d [PR(D)/3] + (d/4)*[PR(A)]
PR(C) = (1-d)*100/4 + d [PR(B)/2 + PR(D)/3] + (d/4)*[PR(A)]
PR(D) = (1-d)*100/4 + (d/4)*[PR(A)]

Node Ranking: A, C, B, D

Initial	
PR(A)	25
PR(B)	25
PR(C)	25
PR(D)	25

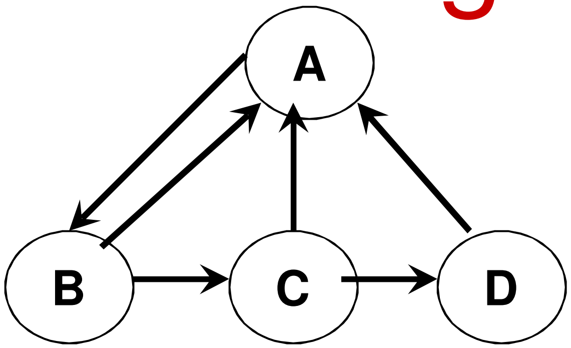
It # 1	
PR(A)	48.02
PR(B)	16.15
PR(C)	26.77
PR(D)	9.063

It # 2	
PR(A)	46.14
PR(B)	16.52
PR(C)	23.386
PR(D)	13.954

It # 3	
PR(A)	44.41
PR(B)	17.51
PR(C)	24.53
PR(D)	13.55

It # 4	
PR(A)	45.32
PR(B)	17.03
PR(C)	24.47
PR(D)	13.18

Page Rank Example (3)



$$PR(A) = (1-d)*100/4 + d*[\tfrac{1}{2}*PR(B) + \tfrac{1}{2}*PR(C) + PR(D)]$$
$$PR(B) = (1-d)*100/4 + d*[PR(A)]$$
$$PR(C) = (1-d)*100/4 + d*[\tfrac{1}{2}*PR(B)]$$
$$PR(D) = (1-d)*100/4 + d*[\tfrac{1}{2}*PR(C)]$$

Initial	
PR(A)	25
PR(B)	25
PR(C)	25
PR(D)	25

It # 1	
PR(A)	46.25
PR(B)	25
PR(C)	14.38
PR(D)	14.38

It # 2	
PR(A)	32.71
PR(B)	43.06
PR(C)	14.38
PR(D)	9.86

It # 3	
PR(A)	36.54
PR(B)	31.55
PR(C)	22.05
PR(D)	9.86

It # 4	
PR(A)	34.91
PR(B)	34.81
PR(C)	17.16
PR(D)	13.12

It # 5	
PR(A)	36.99
PR(B)	33.42
PR(C)	18.54
PR(D)	11.04

It # 6	
PR(A)	35.22
PR(B)	35.12
PR(C)	17.95
PR(D)	11.63

It # 7	
PR(A)	36.19
PR(B)	33.68
PR(C)	18.68
PR(D)	11.38

It # 8	
PR(A)	35.68
PR(B)	34.51
PR(C)	18.06
PR(D)	11.69

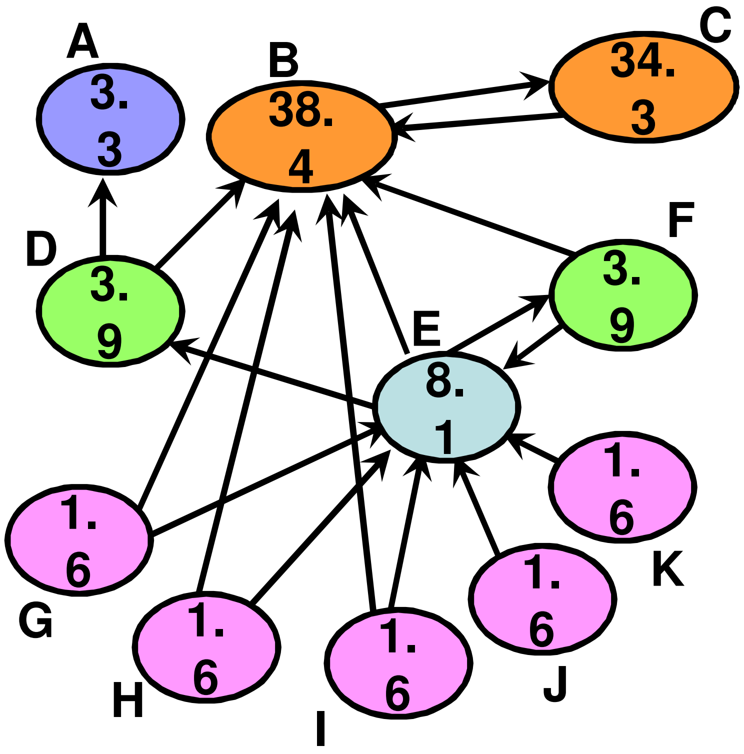
It # 9	
PR(A)	36.03
PR(B)	34.08
PR(C)	18.42
PR(D)	11.43

Node Ranking: A B C D

Computing Huffman Codes for Nodes using their PageRank Values

A	3.3
B	38.4
C	34.3
D	3.9
E	8.1
F	3.9
G	1.6
H	1.6
I	1.6
J	1.6
K	1.6

B	0
C	11
K	10000
I	100010
J	100011
A	10011
G	100100
H	100101
D	10100
F	10101
E	1011



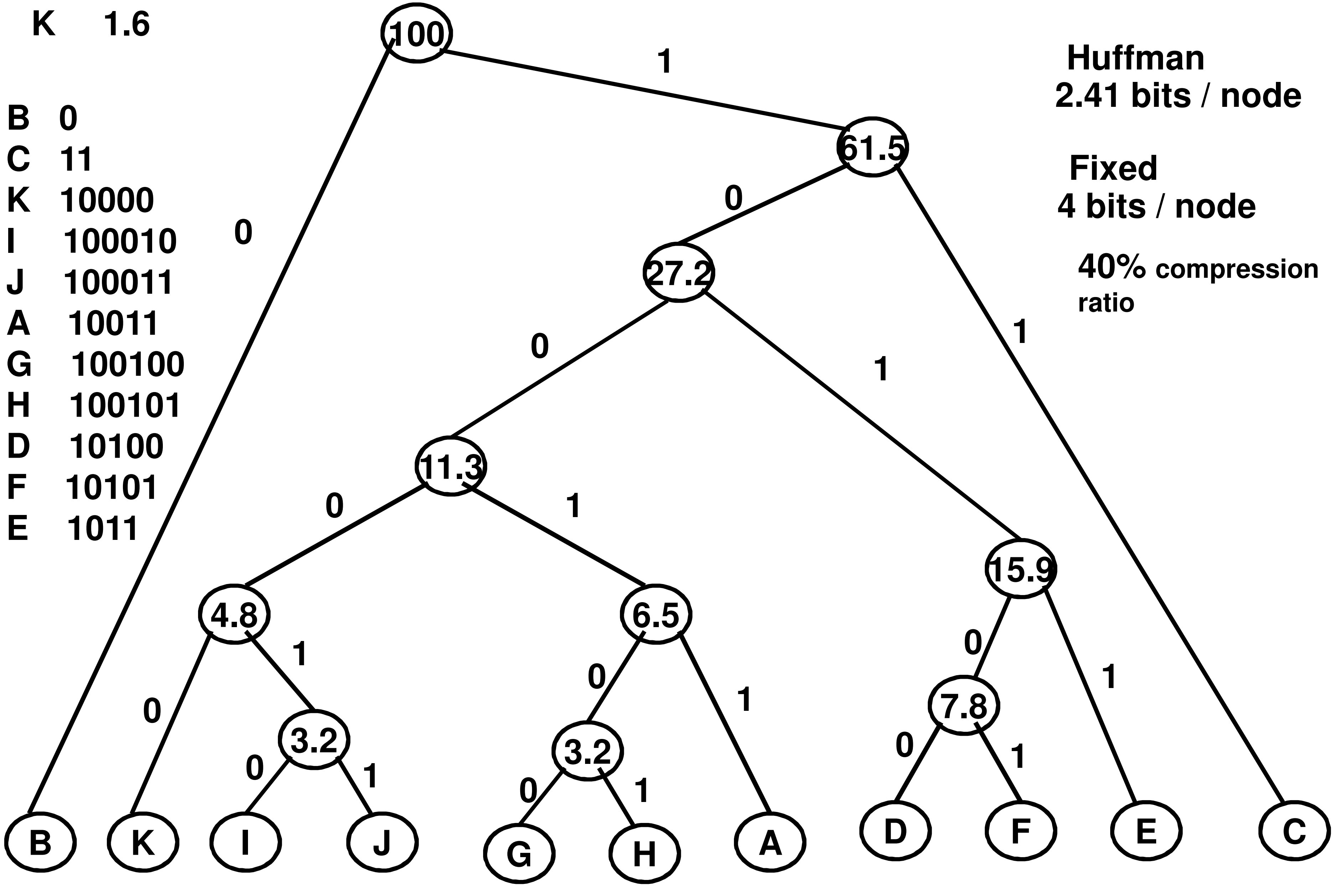
HEBC

100101 1011 0 11

The Huffman codes could be used to efficiently represent paths and frequently used links in the network

A	3.3	B	38.4	C	34.3	D	3.9	E	8.1
F	3.9	G	1.6	H	1.6	I	1.6	J	1.6
K	1.6								

B	0
C	11
K	10000
I	100010
J	100011
A	10011
G	100100
H	100101
D	10100
F	10101
E	1011



Huffman
2.41 bits / node

Fixed
4 bits / node

40% compression
ratio