PageRank & HITS

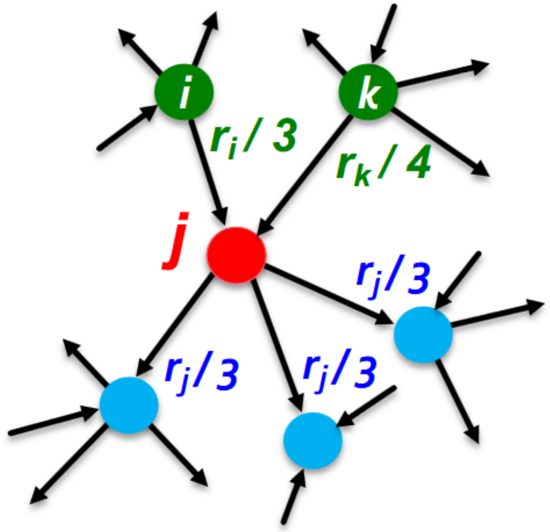
CE642: Social and Economic Networks
Maryam Ramezani
Sharif University of Technology
maryam.ramezani@sharif.edu

01

PageRank

PageRank

- **A "vote" from an important page is worth more:**
  - Each link's vote is proportional to the **importance** of its source page
  - If page ***i*** with importance r_i has d_i out-links, each link gets r_i / d_i votes
  - Page ***j***'s own importance r_j is the sum of the votes on its in-links



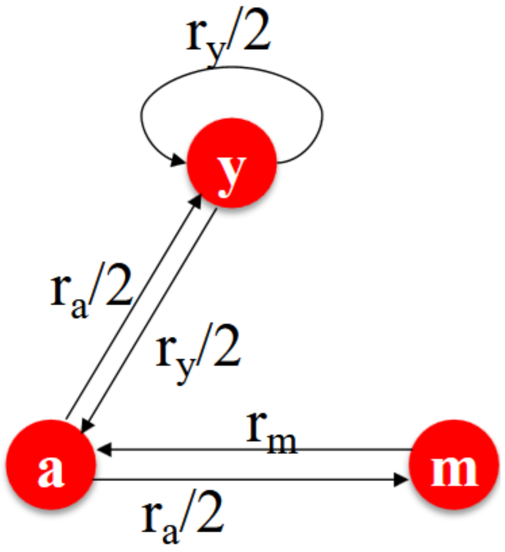
$$r_j = r_i/3 + r_k/4$$

PageRank: Flow View

- **A page is important if it is pointed to by other important pages**
- **Define "rank" r_j for node j**

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

d_i **… out-degree of node i**



"Flow" equations:

$r_y = r_y/2 + r_a/2$

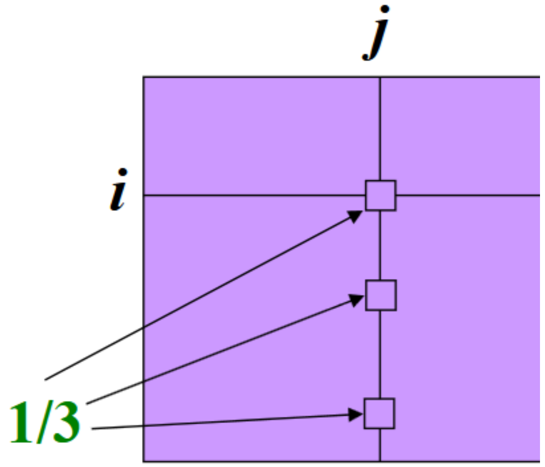
$r_a = r_y/2 + r_m$

$r_m = r_a/2$

You might wonder: Let's just use Gaussian elimination to solve this system of linear equations. Bad idea!

PageRank: Matrix View

- **Stochastic adjacency matrix M**
  - Let page j have d_j out-links
  - If $j \rightarrow i$, then $M_{ij} = \frac{1}{d_j}$
    - M is a **column stochastic matrix**
      - **Columns** sum to **1**

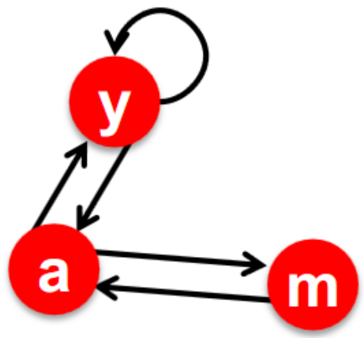


M

- **Rank vector r:** An entry per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$
- **The flow equations can be written**

$$r = M \cdot r \qquad r_j = \sum_{i \rightarrow j} \frac{r_i}{d_i}$$

PageRank Example



	r_y	r_a	r_m
r_y	½	½	0
r_a	½	0	1
r_m	0	½	0

$$
\begin{aligned}
r_y &= r_y/2 + r_a/2 \\
r_a &= r_y/2 + r_m \\
r_m &= r_a/2
\end{aligned}
$$

$$
\underbrace{\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix}}_{\boldsymbol{r}} = \underbrace{\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix}}_{\boldsymbol{M}} \underbrace{\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix}}_{\boldsymbol{r}}
$$

PageRank: Matrix View

$$C_p(v_i) = \alpha \sum_{j=1}^{n} A_{j,i} \frac{C_p(v_j)}{d_j^{\text{out}}} + \beta$$

What if the degree is zero?

$$\begin{cases} d_j^{out} > 0 \\ D = diag(d_1^{\text{out}}, d_2^{\text{out}}, \ldots, d_n^{\text{out}}) \end{cases}$$

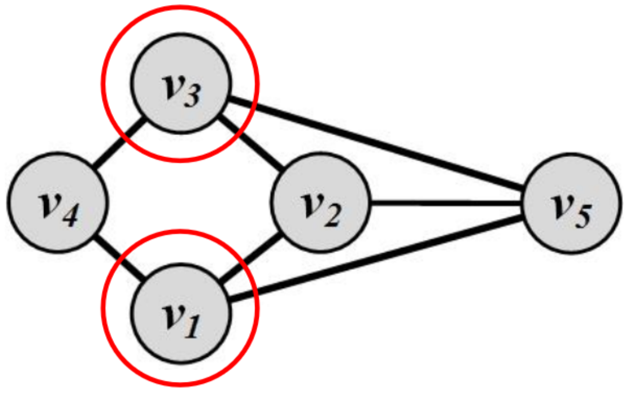
$$\mathbf{C}_p = \alpha A^T D^{-1} \mathbf{C}_p + \beta \mathbf{1}$$

$$\mathbf{C}_p = \beta(\mathbf{I} - \alpha A^T D^{-1})^{-1} \cdot \mathbf{1}$$

Similar to Katz Centrality, in practice, $\boldsymbol{\alpha} < \mathbf{1}/\boldsymbol{\lambda}$, where λ is the largest eigenvalue of $A^T D^{-1}$. In undirected graphs, the largest eigenvalue of $A^T D^{-1}$ is $\boldsymbol{\lambda} = 1$; therefore, $\boldsymbol{\alpha} < \mathbf{1}$.

PageRank Example

- We assume α=0.95 < 1 and and $\beta = 0.1$

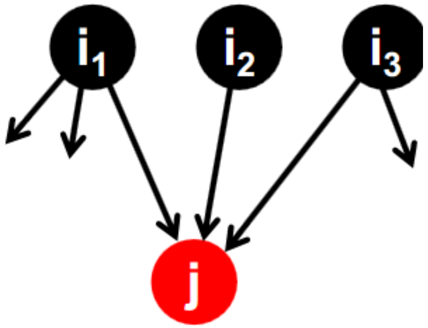


$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

$$\mathbf{C}_p = \beta(\mathbf{I} - \alpha A^T D^{-1})^{-1} \cdot \mathbf{1} = \begin{bmatrix} 2.14 \\ 2.13 \\ 2.14 \\ 1.45 \\ 2.13 \end{bmatrix}$$

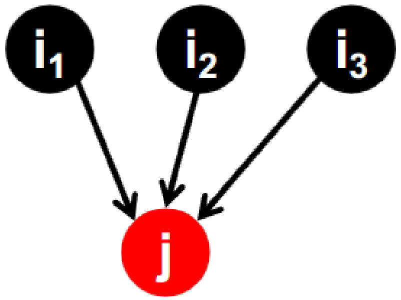
Connection to Random Walk

- **Imagine a random web surfer:**
  - At any time $\boldsymbol{t}$, surfer is on some page i
  - At time $\boldsymbol{t+1}$, the surfer follows an out-link from $\boldsymbol{i}$ uniformly at random
  - Ends up on some page $\boldsymbol{j}$ linked from $\boldsymbol{i}$
  - Process repeats indefinitely
- **Let:**
  - $\boldsymbol{p(t)}$ … vector whose i^{th} coordinate is the prob. that the surfer is at page i at time t
  - So, $\boldsymbol{p(t)}$ is a probability distribution over pages



$$r_j = \sum_{i \to j} \frac{r_i}{\mathrm{d_{out}(i)}}$$

Stationary Distribution



$$p(t+1) = \mathrm{M} \cdot p(t)$$

- **Where is the surfer at time $t+1$?**
  - Follow a link uniformly at random

    $$\boldsymbol{p(t+1) = M \cdot p(t)}$$

- Suppose the random walk reaches a state

  $$\boldsymbol{p(t+1) = M \cdot p(t) = p(t)}$$

  then $\boldsymbol{p}(t)$ is **stationary distribution** of a random walk

- **Our original rank vector** $\boldsymbol{r}$ satisfies $\boldsymbol{r = M \cdot r}$
  - **So, $\boldsymbol{r}$ is a stationary distribution for the random walk**

How to solve PageRank?

- **The flow equation:**

  $$1 \cdot \boldsymbol{r} = \boldsymbol{M} \cdot \boldsymbol{r}$$

$$\underbrace{\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix}}_{\boldsymbol{r}} = \underbrace{\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 1 \\ 0 & \frac{1}{2} & 0 \end{bmatrix}}_{\boldsymbol{M}} \underbrace{\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix}}_{\boldsymbol{r}}$$

- So the **rank vector** $\boldsymbol{r}$ is an **eigenvector** of the stochastic ajd. matrix $\boldsymbol{M}$ (with eigenvalue 1)
  - Starting from any vector $\boldsymbol{u}$, the limit $\boldsymbol{M}(\boldsymbol{M}(\ldots \boldsymbol{M}(\boldsymbol{M}\,\boldsymbol{u})))$ is the **long-term distribution** of the surfers.
    - **PageRank** = Limiting distribution = **principal eigenvector** of M
    - Note: If $\boldsymbol{r}$ is the limit of the product $\boldsymbol{M}\boldsymbol{M} \ldots \boldsymbol{M}\boldsymbol{u}$, then $\boldsymbol{r}$ satisfies the **flow equation** $1 \cdot \boldsymbol{r} = \boldsymbol{M}\boldsymbol{r}$
    - So $\boldsymbol{r}$ is the **principal eigenvector** of $\boldsymbol{M}$ with eigenvalue 1
- **We can now efficiently solve for *r*!**
  - The method is called **Power iteration**

Power Iteration for PageRank

- **Given a web graph with *N* nodes, where the nodes are pages and edges are hyperlinks**
- **Power iteration:** a simple iterative scheme
  - Initialize: $\boldsymbol{r}^0 = [1/N, \ldots., 1/N]^T$
  - Iterate: $\boldsymbol{r}^{(t+1)} = \boldsymbol{M} \cdot \boldsymbol{r}^t$
  - Stop when $|\boldsymbol{r}^{(t+1)} - \boldsymbol{r}^t|_1 < \varepsilon$

$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i}$$

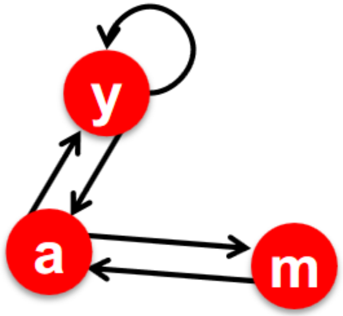
d_i …. out-degree of node i

$|x|_1 = \sum_1^N |x_1|$ is the **L1** norm
Can use any other vector norm, e.g., Euclidean

About 50 iterations is sufficient to estimate the limiting solution.

PageRank Example

- **Power Iteration:**
  - Set $r_j \leftarrow 1/N$
  - **1:** $r'_j \leftarrow \sum_{i \to j} \frac{r_i}{d_i}$
  - **2:** If $|r - r'| > \varepsilon$:
    - $r \leftarrow r'$
  - **3:** go to **1**
- **Example:**



	y	a	m
y	½	½	0
a	½	0	1
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + rm$$
$$r_m = r_a/2$$

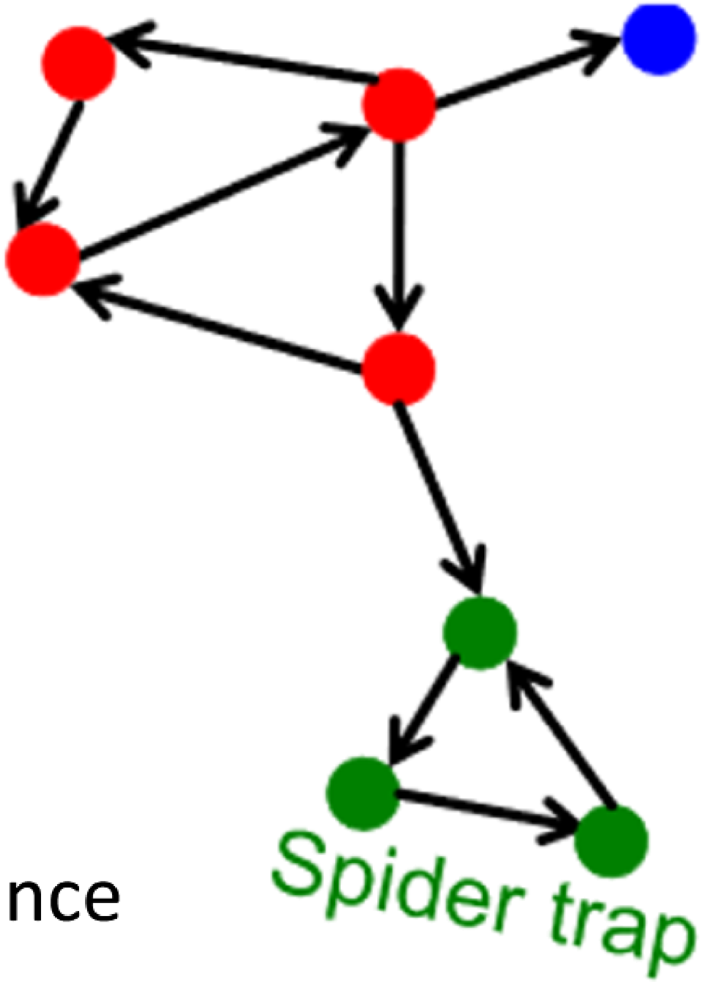
$$\begin{pmatrix} r_y \\ r_a \\ r_m \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} \begin{pmatrix} 1/3 \\ 3/6 \\ 1/6 \end{pmatrix} \begin{pmatrix} 5/12 \\ 1/3 \\ 3/12 \end{pmatrix} \begin{pmatrix} 9/24 \\ 11/24 \\ 1/6 \end{pmatrix} \ldots \begin{pmatrix} 6/15 \\ 6/15 \\ 3/15 \end{pmatrix}$$

Iteration 0, 1, 2, …

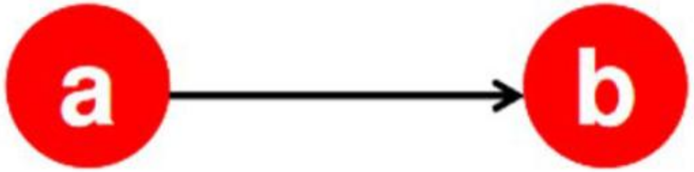
PageRank Problems

Two problems:

- **(1)** Some pages are **dead ends** (have no out-links)
  - Such pages cause importance to "leak out"
- **(2)** **Spider traps** (all out-links are within the group)
  - Eventually spider traps absorb all importance



“Dead End” Problem



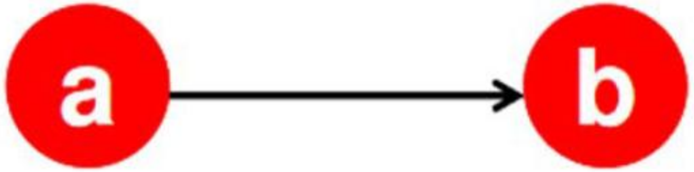
$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_\mathrm{i}}$$

- Example:

$$\begin{matrix} r_a \\ r_b \end{matrix} = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{matrix}$$

Iteration 0, 1, 2, …

“Dead End” Problem



$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{\mathrm{d}_\mathrm{i}}$$

- **Example:**

$$\begin{array}{c} r_a \\ r_b \end{array} = \begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{array}$$

Iteration 0, 1, 2, …

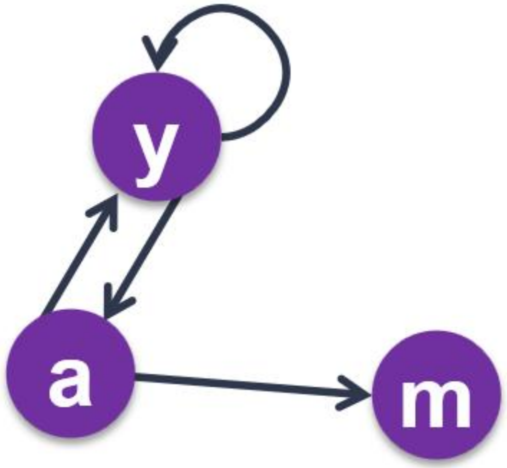
“Dead End” Problem

Power Iteration:

Set $r_j = 1$

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

And iterate



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2$$
$$r_m = r_a/2$$

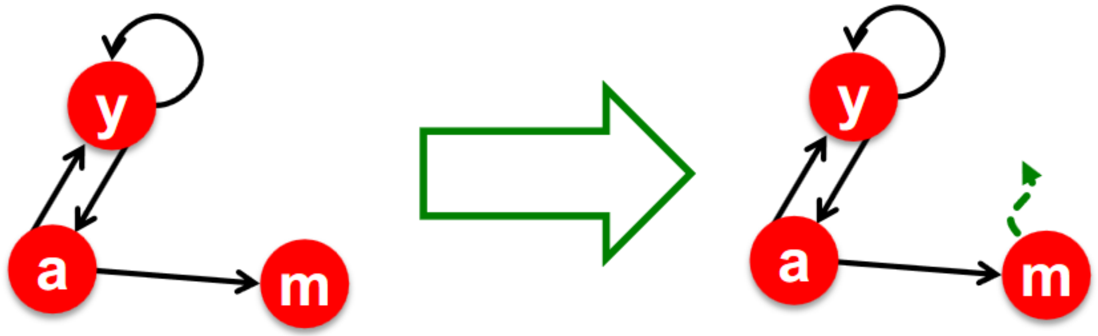
$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \ldots & 0 \\ 1/3 & 1/6 & 1/12 & 2/24 & & 0 \end{matrix}$$

Iteration 0, 1, 2, …

Here the PageRank “leaks” out since the matrix is not stochastic.

Solution to “Dead End” Problem

- **Teleports:** Follow random teleport links with total probability **1.0** from dead-ends
  - Adjust matrix accordingly

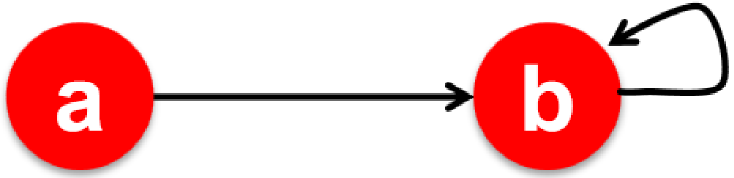


	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

	y	a	m
y	½	½	⅓
a	½	0	⅓
m	0	½	⅓

"Spider Trap" Problem

- **The "Spider trap" problem:**



$$r_j^{(t+1)} = \sum_{i \to j} \frac{r_i^{(t)}}{d_i}$$

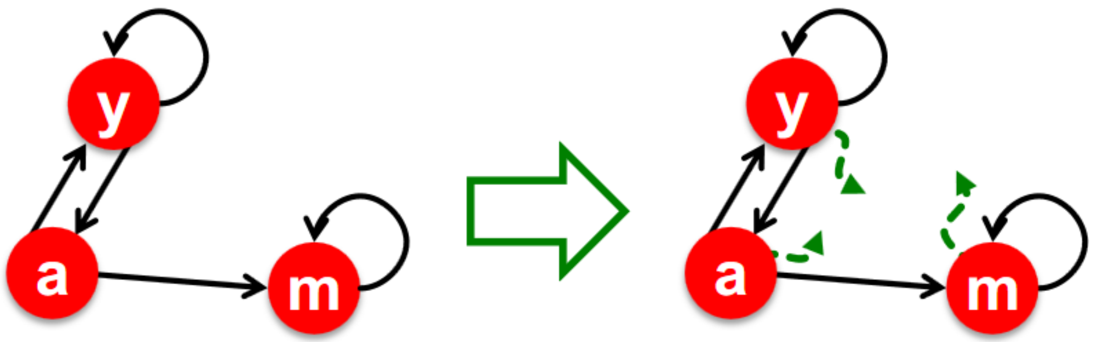
- **Example:**

Iteration: 0, 1, 2, 3…

$$\begin{matrix} r_a \\ r_b \end{matrix} = \begin{matrix} 1 \\ 0 \end{matrix} \left| \begin{matrix} 0 \\ 1 \end{matrix} \right| \begin{matrix} 0 \\ 1 \end{matrix} \left| \begin{matrix} 0 \\ 1 \end{matrix} \right.$$

Solution to "Spider Trap" Problem

- **Solution for spider traps: At each time step, the random surfer has two options**
  - With prob. β, follow a link at random
  - With prob. **1-β,** jump to a random page
  - Common values for β are in the range 0.8 to 0.9
- **Surfer will teleport out of spider trap within a few time steps**



Why Teleports Solve the Problem?

Why are dead-ends and spider traps a problem and why do teleports solve the problem?

- **Spider-traps** are not a problem, but with traps PageRank scores are **not** what we want
  - **Solution:** Never get stuck in a spider trap by teleporting out of it in a finite number of steps
- **Dead-ends** are a problem
  - The matrix is not column stochastic so our initial assumptions are not met
  - **Solution:** Make matrix column stochastic by always teleporting when there is nowhere else to go

The Google Matrix

- <u>**Google's solution that does it all:**</u>
  At each step, random surfer has two options:
  - With probability β, follow a link at random
  - With probability $1\text{-}\beta$, jump to some random page
- **PageRank equation** [Brin-Page, 98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N}$$

d_i … out-degree of node i

This formulation assumes that M has no dead ends. We can either preprocess matrix M to remove all dead ends or explicitly follow random teleport links with probability 1.0 from dead-ends.

The Google Matrix

- **PageRank equation** [Brin-Page, '98]

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N}$$

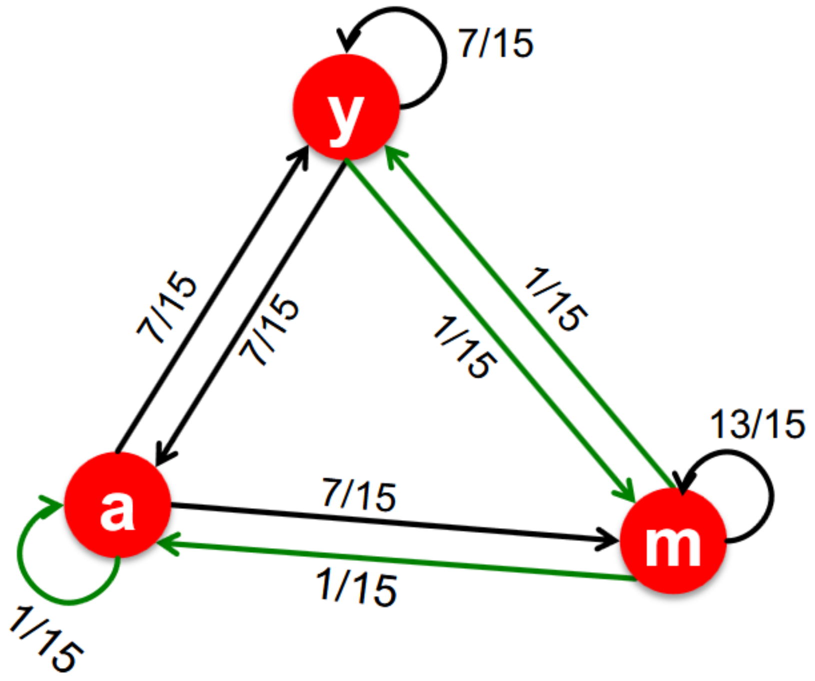
- **The Google Matrix G:**

$[1/N]_{N \times N}$…N by N matrix
where all entries are 1/N

$$G = \beta\, M + (1-\beta)\left[\frac{1}{N}\right]_{N \times N}$$

- **We have a recursive problem: $\boldsymbol{r = G \cdot r}$**
  And the Power method still works!
- **What is β?**
  - In practice $\beta = 0.8, 0.9$ (make 5 steps on avg., jump)

Random Teleports ($\beta = 0.8$)



$$0.8 \underbrace{\begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}}_{M} + 0.2 \underbrace{\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}}_{[1/N]_{NxN}}$$

$$G = \begin{array}{c} y \\ a \\ m \end{array} \begin{bmatrix} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{bmatrix}$$

$$\begin{array}{l} y \\ a \\ m \end{array} = \begin{array}{c} 1/3 \\ 1/3 \\ 1/3 \end{array} \quad \begin{array}{c} 0.33 \\ 0.20 \\ 0.46 \end{array} \quad \begin{array}{c} 0.24 \\ 0.20 \\ 0.52 \end{array} \quad \begin{array}{c} 0.26 \\ 0.18 \\ 0.56 \end{array} \quad \cdots \quad \begin{array}{c} 7/33 \\ 5/33 \\ 21/33 \end{array}$$

Conclusion

- PageRank solves for $\boldsymbol{r} = \boldsymbol{G}\boldsymbol{r}$ and can be efficiently computed by power iteration of the stochastic adjacency matrix ($\boldsymbol{G}$)
- Adding random uniform teleportation solves issues of dead-ends and spider-traps

PageRank Problems

- **Measures generic popularity of a page**
  - Biased against topic-specific authorities
  - **Solution:** Topic-Specific PageRank (next)
- **Uses a single measure of importance**
  - Other models e.g., hubs-and-authorities
  - **Solution:** Hubs-and-Authorities (next)
- **Susceptible to Link spam**
  - Artificial link topographies created in order to boost page rank
  - **Solution:** TrustRank (next)

02

Topic-Specific PageRank

Topic-Specific PageRank

- **Instead of generic popularity, can we measure popularity within a topic?**
- **Goal:** Evaluate Web pages not just according to their popularity, but by how close they are to a particular topic, e.g. “sports” or “history.”
- **Allows search queries to be answered based on interests of the user**
  - **Example:** Query “Trojan” wants different pages depending on whether you are interested in sports or history.

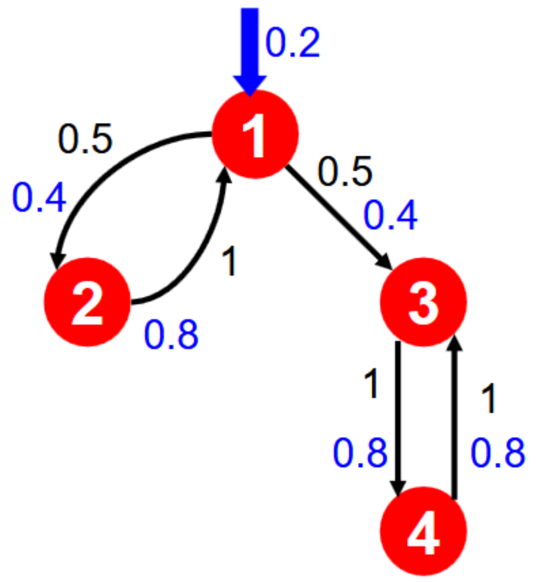
Topic-Specific PageRank

- Assume each walker has a small probability of "teleporting" at any step
- **Teleport can go to:**
  - Any page with equal probability
    - To avoid dead-end and spider-trap problems
  - A topic-specific set of "relevant" pages (teleport set)
    - For topic-sensitive PageRank.
- **Idea: Bias the random walk**
  - When walked teleports, she pick a page from a set S
  - S contains only pages that are relevant to the topic
    - E.g., Open Directory (DMOZ) pages for a given topic
  - For each teleport set S, we get a different vector r_S

Topic-Specific PageRank

- **Let:**
  - $A_{ij} = \beta M_{ij} + (1-\beta) / |S|$ if $i \in S$
    βM_{ij} otherwise
  - **A** is stochastic!
- We have weighted all pages in the teleport set S equally
  - Could also assign different weights to pages!
- **Compute as for regular PageRank:**
  - Multiply by M, then add a vector
  - Maintains sparseness

Topic-Specific PageRank Example



Suppose S = {1}, β = 0.8

Node	Iteration			
	0	**1**	**2…**	**stable**
1	1.0	0.2	0.52	0.294
2	0	0.4	0.08	0.118
3	0	0.4	0.08	0.327
4	0	0	0.32	0.261

Note how we initialize the PageRank vector differently from the unbiased PageRank case.

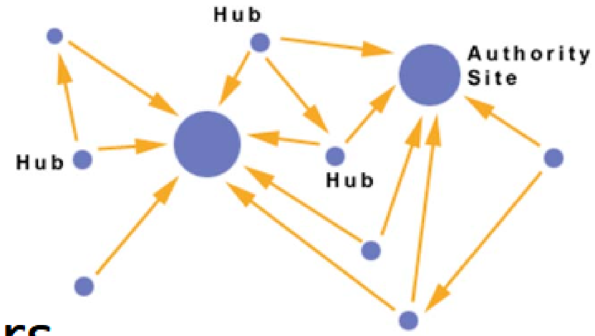
03

HITS (Hypertext-Induced Topic Selection)

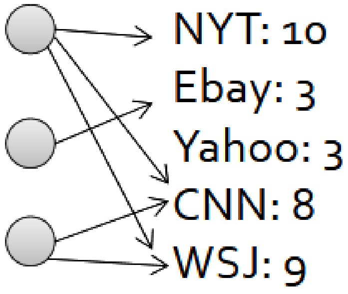
Hubs and Authorities

Interesting pages fall into two classes:

1. **Authorities** are pages containing useful information
   - Newspaper home pages
   - Course home pages
   - Home pages of auto manufacturers



2. **Hubs** are pages that link to authorities
   - List of newspapers
   - Course bulletin
   - List of US auto manufacturers

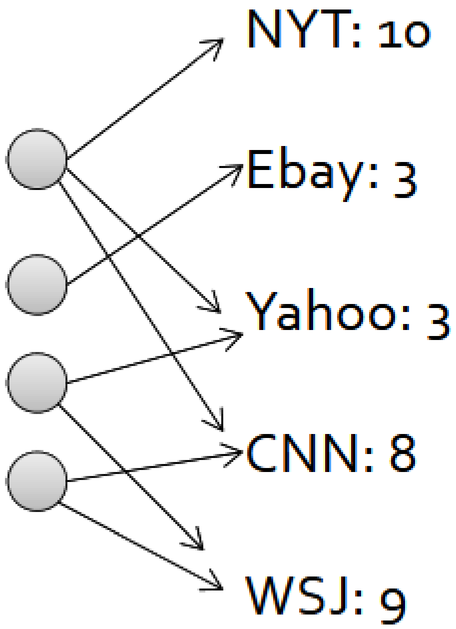


Hubs and Authorities

- **Hubs and Authorities**

  Each page has 2 scores:

  - **Quality as an expert (hub):**
    - Total sum of votes of pages pointed to
  - **Quality as an content (authority):**
    - Total sum of votes of experts
  - Principle of repeated improvement



Hubs and Authorities

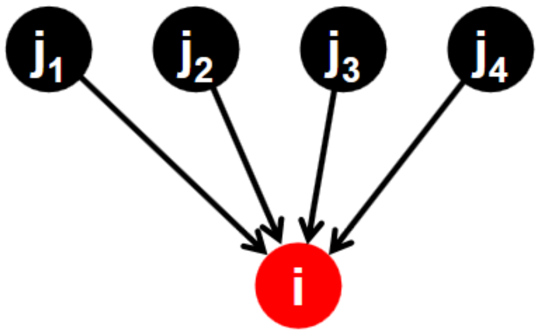
- A good hub links to many good authorities
- A good authority is linked from many good hubs
- Model using two scores for each node:
  - **Hub** score and **Authority** score
  - Represented as vectors h and a

Hubs and Authorities

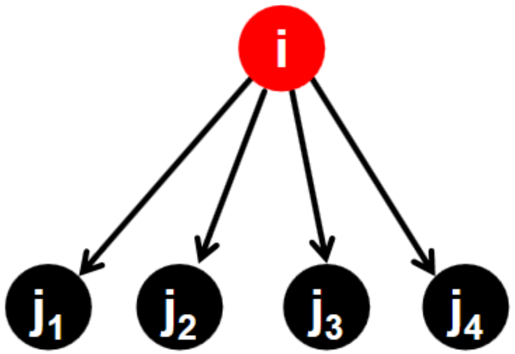
- **Each page *i* has 2 scores:**
  - Authority score: a_i
  - Hub score: h_i

HITS algorithm:

- Initialize: $a_j = 1, \; h_i = 1$
- Then keep iterating:
  - $\forall i$: Authority: $a_i = \sum_{j \to i} h_j$
  - $\forall i$: Hub: $h_i = \sum_{i \to j} a_j$
  - $\forall i$: normalize: $\sum_j a_j = 1, \sum_j h_j = 1$



$$a_i = \sum_{j \to i} h_j$$



$$h_i = \sum_{i \to j} a_j$$

Transition Matrix A

- **HITS converges to a single stable point**
- Slightly change the notation:
  - Vector $a = (a_1 \ldots, a_n)$, $h = (h_1 \ldots, h_n)$
  - Adjacency matrix (n x n): $A_{ij}=1$ if $i \rightarrow j$
- **Then:**

$$h_i = \sum_{i \rightarrow j} a_j \Leftrightarrow h_i = \sum_j A_{ij} a_j$$

- So: $h = A\,a$
- And likewise: $a = A^T\,h$

Hubs and Authorities Equations

- The hub score of page *i* is proportional to the sum of the authority scores of the pages it links to: $h = \lambda A a$
  - Constant λ is a scale factor, $\lambda=1/\sum h_i$
- The authority score of page i is proportional to the sum of the hub scores of the pages it is linked from: $a = \mu A^T h$
  - Constant μ is scale factor, $\mu=1/\sum a_i$

Iterative Algorithm

- The HITS algorithm:
  - Initialize **h**, **a** to all 1's
  - Repeat:
    - $\mathbf{h} = \mathbf{A}\,\mathbf{a}$
    - Scale **h** so that its sums to 1.0
    - $\mathbf{a} = \mathbf{A}^T\,\mathbf{h}$
    - Scale **a** so that its sums to 1.0
  - Until **h**, **a** converge (i.e., change very little)

Hubs and Authorities Equations

- **HITS algorithm in new notation:**
  - Set: $a = h = 1^n$
  - **Repeat:**
    - $h = A\,a, \quad a = A^T h$
    - Normalize
- Then: $a = \underbrace{A^T \underbrace{(A\,a)}_{\text{new h}}}_{\text{new a}}$

a is being updated (in 2 steps):

$A^T(A\,a) = (A^T A)\,a$

h is updated (in 2 steps):

$A\,(A^T h) = (A\,A^T)\,h$

- Thus, in $2k$ steps:

$$a = (A^T A)^k\,a$$

$$h = (A\,A^T)^k\,h$$

Repeated matrix powering

Hubs and Authorities Equations

- $h = \lambda A a$
- $a = \mu A^T h$
- $h = \lambda \mu A A^T h$
- $a = \lambda \mu A^T A a$

$\lambda = 1/\sum h_i$
$\mu = 1/\sum a_i$

- Under reasonable assumptions about A, the HITS iterative algorithm converges to vectors h^* and a^*:
  - h^* is the principal eigenvector of matrix $A A^T$
  - a^* is the principal eigenvector of matrix $A^T A$

Conclusion

- **PageRank and HITS are two solutions to the same problem:**
  - **What is the value of an in-link from *u* to *v*?**
  - In the PageRank model, the value of the link depends on the links **into** *u*
  - In the HITS model, it depends on the value of the other links **out of** *u*

04

TrustRank

Idea

- **Basic principle: Approximate isolation**
  - It is rare for a "good" page to point to a "bad" (spam) page
- Sample a set of "seed pages" from the web
- Have an oracle (human) identify the good pages and the spam pages in the seed set
  - Expensive task, so we must make seed set as small as possible

Idea

- Call the subset of seed pages that are identified as "good" the "trusted pages"
- Perform a topic-sensitive PageRank with teleport set = trusted pages.
  - **Propagate trust through links:**
    - Each page gets a trust value between 0 and 1
- Use a threshold value and mark all pages below the trust threshold as spam

Simple Model

- Set trust of each trusted page to 1
- Suppose trust of page p is t_p
  - Set of out-links o_p
- For each $q \in o_p$, p confers the trust:
  - $\beta\, t_p / |o_p|$ for $0 < \beta < 1$
- Trust is additive
  - Trust of p is the sum of the trust conferred on p by all its in-linked pages
- Note similarity to Topic-Specific PageRank
  - Within a scaling factor, TrustRank = PageRank with trusted pages as teleport set

Any Question?