Link Analysis (Part II)

Prof. Wei-Min Shen
University of Southern California

With slide contributions from
J. Leskovec, Anand Rajaraman, Jeffrey D. Ullman

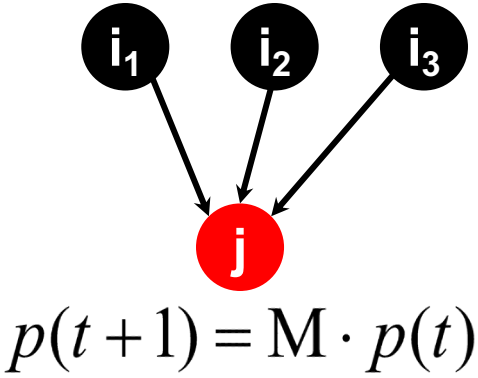
OUTLINE

- Link Analysis (Part II)
  - Solving the flow model: $\boldsymbol{Mv} = \boldsymbol{v}$
    - Power Iteration and Gaussian Elimination
    - Dead-ends and spider traps (teleporting and taxation)
    - PageRank for Search Engines (MapReduce)
  - Topic-specific: (augment for general-popularity)
  - Trust-Rank: (fighting against link spams)
  - Hubs-and-Authorities: (multiple importances)

SPIDER TRAPS AND DEAD-ENDS

The Stationary Distribution

- $v^0 =$



Existence and Uniqueness

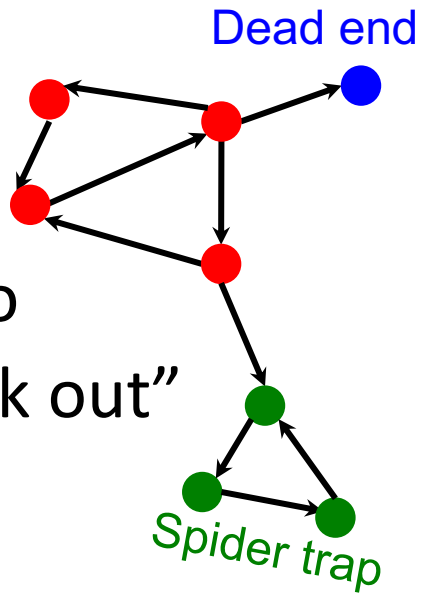
- **A central result from the theory of random walks (a.k.a. Markov processes):**

For graphs that satisfy **certain conditions**, the **stationary distribution is unique** and eventually will be reached no matter what the initial probability distribution at time **t = 0**

PageRank: Problems

Two problems:

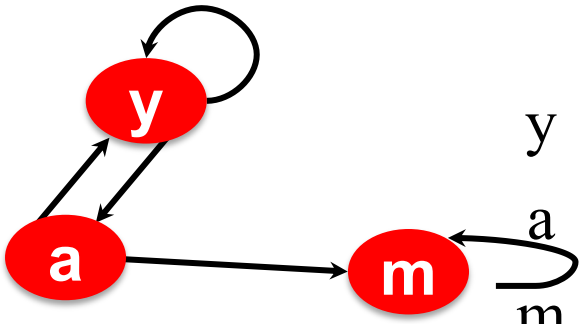
- **(1)** Some pages are **dead ends** (have no out-links)
  - Random walk has "nowhere" to go to
  - Such pages cause importance to "leak out"



- **(2) Spider traps:** (all out-links are within the group)
  - Random walked gets "stuck" in a trap
  - And eventually spider traps absorb all importance

Problem: Spider Traps

All outlinks are within a group



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	1

m is a spider trap

$\mathbf{r}^{k+1} = \mathbf{M}\mathbf{r}^{k}$

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

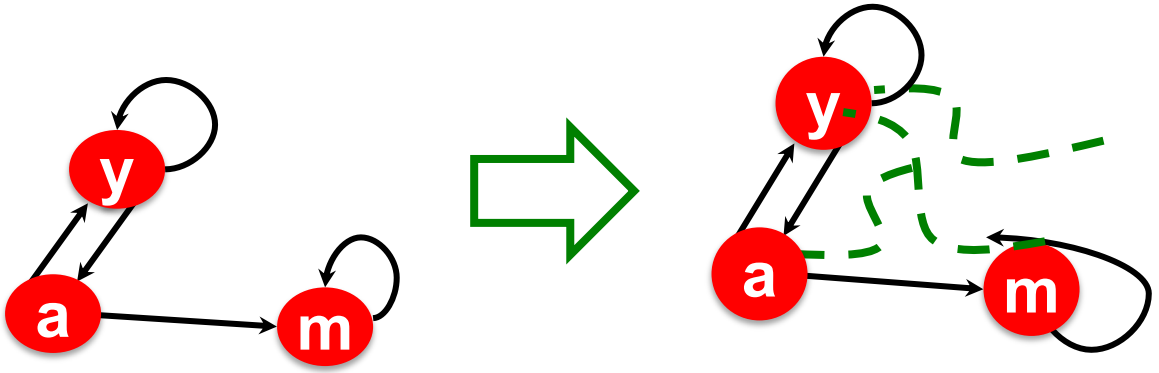
$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} \quad \begin{pmatrix} \\ \\ \\ \end{pmatrix}$$

Iteration 0, 1, 2, …

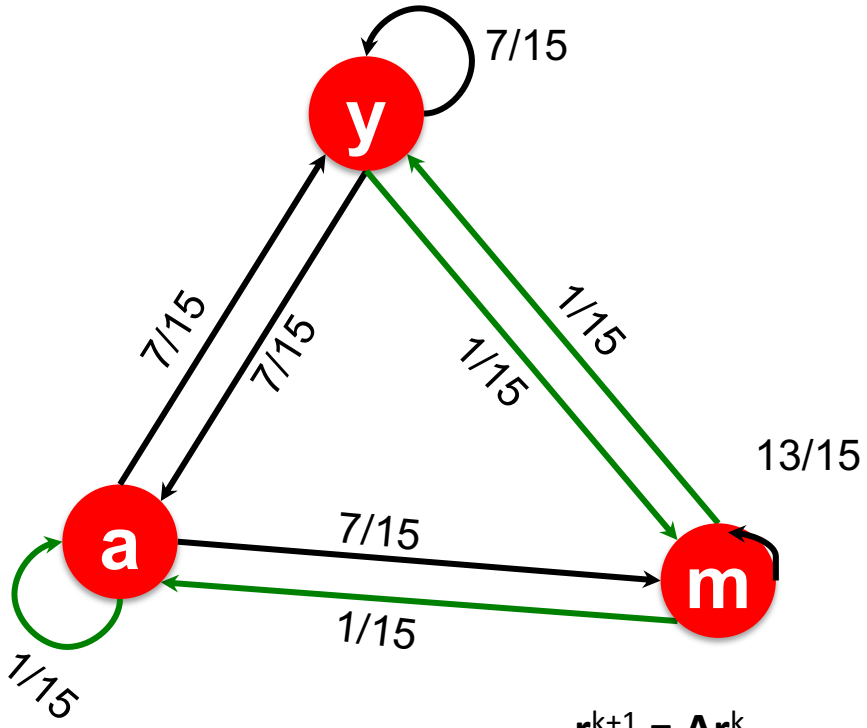
All the PageRank score gets "trapped" in node m

Solution: Teleports!

- **The Google 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 some 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**



Random Teleports (β = 0.8)



$r^{k+1} = Ar^k$

M

$$0.8 \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

$[1/N]_{NxN}$

A

	y	a	m
y	7/15	7/15	1/15
a	7/15	1/15	1/15
m	1/15	7/15	13/15

$$\begin{matrix} r_y \\ r_a \\ r_m \end{matrix} = \begin{matrix} 1/3 & 0.33 & 0.28 & 0.26 & & 7/33 \\ 1/3 & 0.20 & 0.20 & 0.18 & \ldots & 5/33 \\ 1/3 & 0.46 & 0.52 & 0.56 & & 21/33 \end{matrix}$$

Matrix Formulation

- Suppose there are N pages
  - Consider a page j, with set of outlinks O(j)
  - We have $M_{ij} = 1/|O(j)|$ when j->i and $M_{ij} = 0$ otherwise
  - The random teleport is equivalent to
    - adding a teleport link from j to every other page with probability $(1-\beta)/N$
    - reducing the probability of following each outlink from $1/|O(j)|$ to $\beta/|O(j)|$
    - Equivalent: tax each page a fraction $(1-\beta)$ of its score and re-distribute evenly
  - In some sense, "taxation" enables "teleporting"
    - But not the other way around

Taxation

- Deal with spider traps and dead ends
  - By allowing surfer to jump to a random page
  - This jumping is also called “teleporting”

Tax:

$$v' = \beta M v + (1 - \beta)\boldsymbol{e}/n$$

Distribute it evenly

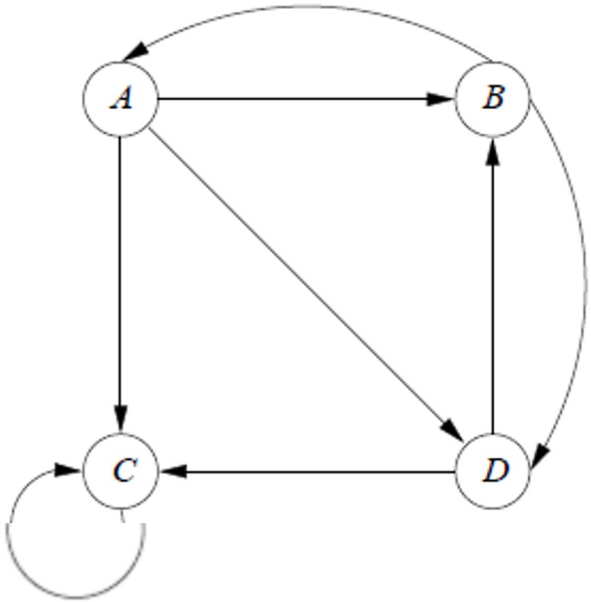
  - β, damping factor, e.g., set to 80%
  - $\boldsymbol{e}$, an n-dimensional vector with all 1’s
  - n, # of nodes in the graph

- This in effect makes graph “strongly-connected”

Taxation on Spider Trap

- Suppose $\beta = .8$

$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 1 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \qquad v^0 = \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}$$



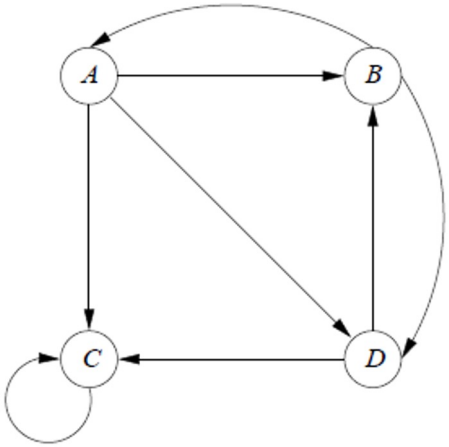
$$v^1 = \begin{bmatrix} 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{2} \\ \frac{1}{3} & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix} \left(.8 \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} \right) + .2 \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 1/10 \\ 1/6 \\ 11/30 \\ 1/6 \end{bmatrix} + \begin{bmatrix} 1/20 \\ 1/20 \\ 1/20 \\ 1/20 \end{bmatrix} = \begin{bmatrix} 3/20 \\ 13/60 \\ 25/60 \\ 13/60 \end{bmatrix}$$

C=0.8*1/4=1/5; A, D = 1/6; total=11/30

(A and D:.8*1/4*(1/3+1/2)=1/6; C:.8*1/4=1/5)

- Suppose $\beta = .8$

$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 1 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \quad v' = .8Mv + .2\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}$$



$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}, \begin{bmatrix} 9/60 \\ 13/60 \\ 25/60 \\ 13/60 \end{bmatrix}, \begin{bmatrix} 41/300 \\ 53/300 \\ 153/300 \\ 53/300 \end{bmatrix}, \begin{bmatrix} 543/4500 \\ 707/4500 \\ 2543/4500 \\ 707/4500 \end{bmatrix}, \ldots, \begin{bmatrix} 15/148 \\ 19/148 \\ 95/148 \\ 19/148 \end{bmatrix}$$

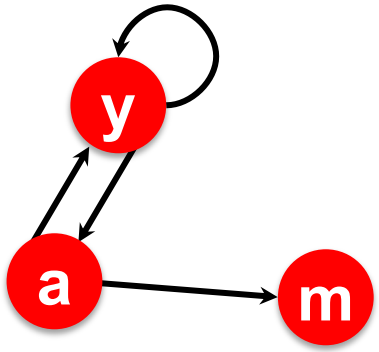
A	C	D	.8(1/3 A)	.8(1/2 D)	.8C	.2 * 1/4	Next C
1/4	1/4	1/4	1/15	1/10	1/5	1/20	25/60 (.4)
9/60	25/60	13/60	3/75	13/150	1/3	1/20	153/300 (.5)
…	…	…	…	…	…	…	…
15/148	95/148	19/148	4/148	19/370	76/148	1/20	95/148 (.64)

decreasing decreasing increasing Fixed (teleport) increasing

Problem: Dead Ends

Pages with no outlinks are "dead ends" for the random surfer:

Nowhere to go on next step



	y	a	m
y	½	½	0
a	½	0	0
m	0	½	**0**

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

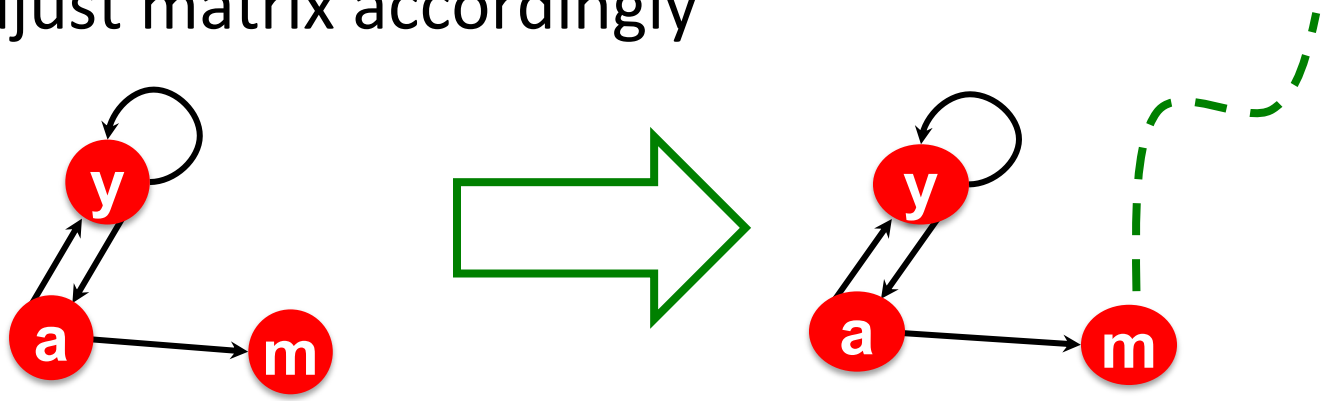
$$\mathbf{r}^{k+1} = \mathbf{A}\mathbf{r}^{k}$$

$$\begin{pmatrix} \\ \\ \\ \\ \end{pmatrix} \begin{vmatrix} -\lambda & 1/2 & 1 & 0 \\ 1/3 & -\lambda & 0 & 1/2 \\ 1/3 & 0 & -\lambda & 1/2 \\ 1/3 & 1/2 & 0 & -\lambda \end{vmatrix} = 0$$

Here the PageRank "leaks" out since the matrix is not column stochastic (not all columns total to 1)

Solution 1 for Dead Ends: Always Teleport!

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



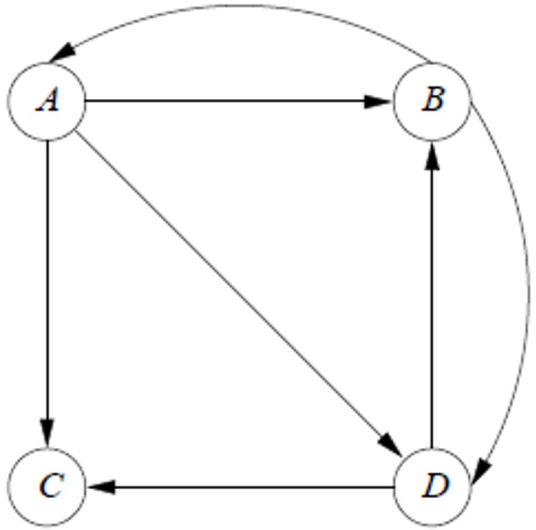
	y	a	m
y	½	½	0
a	½	0	0
m	0	½	0

	y	a	m
y	½	½	1/3
a	½	0	1/3
m	0	½	1/3

Solution 2 for Dead Ends: Taxation

- Suppose β = .8

$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$



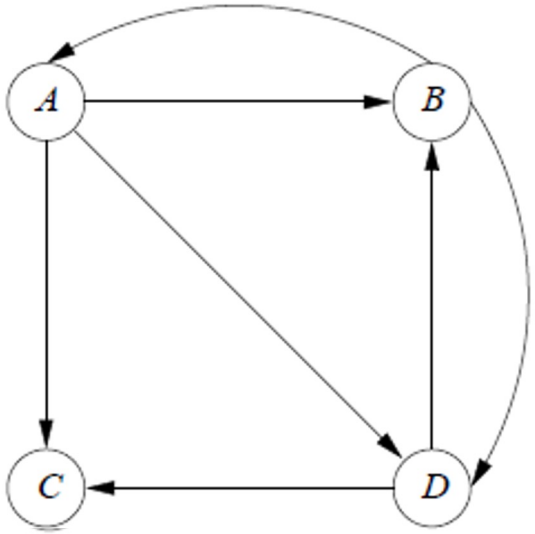
$$v' = .8\begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} v + .2\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix} (.8v) + .2\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}$$

Taxation on Dead End (and others)

- Suppose $\beta = .8$

Total = 1

$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \quad v^0 = \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}$$



Total: 48/60

$$v^1 = \begin{bmatrix} 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix} \left(.8 \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} \right) + .2 \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 1/10 \\ 1/6 \\ 1/6 \\ 1/6 \end{bmatrix} + \begin{bmatrix} 1/20 \\ 1/20 \\ 1/20 \\ 1/20 \end{bmatrix} = \begin{bmatrix} 3/20 \\ 13/60 \\ 13/60 \\ 13/60 \end{bmatrix}$$

Loss: 8/10 * 1/4 + 2/10 *1 – 2/10 *1 = 1/5 = 12/60

sink on c | tax on all | teleport (gain)

Taxation on Dead End (and others)

- Suppose β = .8 ga

Total = 1

Total = 48/60=4/5

$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \quad v^0 = \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}, v^1 = \begin{bmatrix} 3/20 \\ 13/60 \\ 13/60 \\ 13/60 \end{bmatrix}$$

Total: 2/3

$$v^2 = \begin{bmatrix} 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix} \left(.8 \begin{bmatrix} 3/20 \\ 13/60 \\ 13/60 \\ 13/60 \end{bmatrix} \right) + .2 \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 13/150 \\ 19/150 \\ 19/150 \\ 19/150 \end{bmatrix} + \begin{bmatrix} 1/20 \\ 1/20 \\ 1/20 \\ 1/20 \end{bmatrix} = \begin{bmatrix} 41/300 \\ 53/300 \\ 53/300 \\ 53/300 \end{bmatrix}$$

sink on C tax on all teleport gain

Net loss in this step: 0.8*13/60 + 0.2*48/60 – 0.2*1 = 13/75 + 4/25 – 1/5 = 2/15

Total loss (step 2 and 1): 2/15 + 1/5 = 1/3

After 100 Iterations

$$\begin{bmatrix}1/4\\1/4\\1/4\\1/4\end{bmatrix}, \begin{bmatrix}3/20\\13/60\\13/60\\13/60\end{bmatrix}, \begin{bmatrix}41/300\\53/300\\53/300\\53/300\end{bmatrix}, \ldots, \begin{bmatrix}.1\\.13\\.13\\.13\end{bmatrix}$$

Remaining: 1 4/5 2/3 .49

- Net loss = column 2 + column 3 – column 4

At the end of iteration #	Loss due to dead end	Loss due to taxation (20% tax)	Gain due to teleporting	Net loss in step	Total loss in all steps	Remaining ranks
1	8/10 * 1/4	2/10 * 1	2/10 * 1	1/5 (.2)	1/5 (.2)	4/5
2	8/10 * 13/60	2/10 * 4/5	2/10 * 1	2/15 (.13)	1/5+2/15 (.33)	2/3
…	…	…	…	…	…	…

Net loss will be zero when loss and gain balance out

Without taxation (continuously sink)

At the end of iteration #	Lost Prob.	Total loss	Remaining
1	1/4 (.25)	1/4 (.25)	3/4
2	5/24 (.21)	1/4 + 5/24 = 11/24 (.45)	1-11/24 =13/24
3	7/48 (.15)	11/24+7/48 =29/48 (.6)	1-29/48 =19/48
…	…	…	…

With taxation

At the end of iteration #	Loss due to dead end	Loss due to taxation (20% tax)	Gain due to teleporting	Net loss in a step	Total loss in all steps	Remaining ranks
1	8/10 * 1/4	2/10 * 1	2/10 * 1	1/5 (.2)	1/5 (.2)	4/5
2	8/10 * 13/60	2/10 * 4/5	2/10 * 1	2/15 (.13)	1/5+2/15 (.3)	2/3
…	…	…	…	…	…	…

Net loss will be zero when loss and gain balance out

Why Do Teleports Solve the Problem?

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

- **Spider-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**
  - 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

Alternative for Dealing with dead-ends

- **Prune and propagate**
  - Preprocess the graph to eliminate dead-ends
  - This may create new dead-ends
  - Might require multiple passes
  - Compute pagerank on reduced graph
  - Approximate values for dead-ends by propagating values from reduced graph

How is Pagerank Information Used in a Search Engine?

- Each search engine has a **secret formula** that **decides the order in which to show pages** to a user in response to a search query
  - **Google: thought to use over 250 different properties of pages**
- **Search engine decides on a linear order of search results**
- To be considered for ranking, **page has to have at least one of the query search terms**
  - **Unless all search terms are present, little chance of being in top ten results**
- **Among qualified pages, score computed for each**
  - **PageRank is an important component of the score**
  - **Other components:** presence/absence of search terms in headers, links to the page

PageRank Iteration using MapReduce

- If *n* is small enough that each Map task can store the full vector *v* and *v′* in main memory
  - With additional steps to multiply each component of *Mv* by constant β and to add $(1-\beta)/n$
- Given the size of the Web, v is much too large to fit in main memory
- Break M into vertical stripes (chp. 2.3.1) and break v into corresponding horizontal stripes to execute the MapReduce

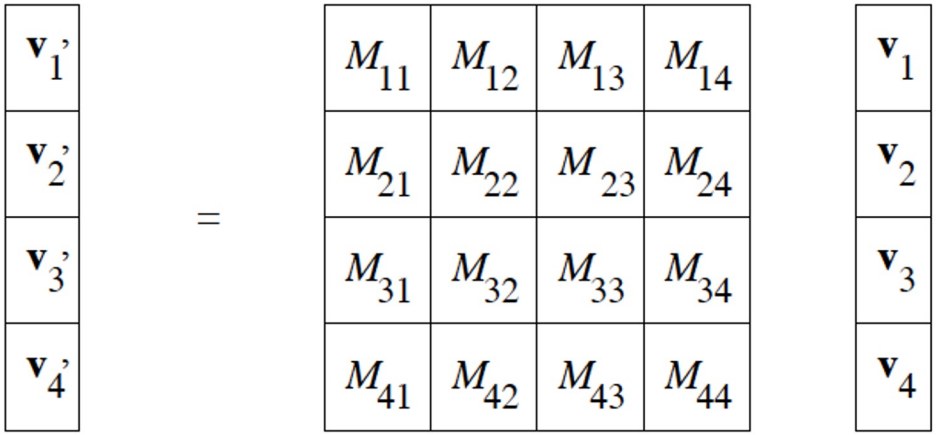
Matrix-Vector Multiplication by MapReduce

- Matrix M of size n*n, vector v of length n
- M v = x
- the vector x of length n, whose i_{th} element x_i is given by

$$x_i = \sum_{j=1}^{n} m_{ij} v_j$$

Use of Combiners to Consolidate the Result Vector

- Partition the vector v into k stripes and matrix into k^2 blocks
- E.g., k=4



$$x_i = \sum_{j=1}^{n} m_{ij} v_j$$

Use of Combiners to Consolidate the Result Vector (Cont'd)

- Use k^2 Map tasks (e.g., 16)
- Each task gets one square of the matrix M (M_{ij}) and one stripe of the vector v (v_j)
  - each stripe of the vector is sent to k (e.g., 4) different Map tasks
  - v_j is sent to the task handling M_{ij} for each of the k possible values of i
- Advantage: keep both the j_{th} stripe of v and the i_{th} stripe of v' in main memory as we process M_{ij}

$$\begin{bmatrix} v_1' \\ v_2' \\ v_3' \\ v_4' \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ M_{41} & M_{42} & M_{43} & M_{44} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}$$

$$x_i = \sum_{j=1}^{n} m_{ij} v_j$$

Note that all terms generated from M_{ij} and v_j contribute to v'_i and no other stripe of v'

PageRank Iteration Using MapReduce

- Assume rank vector **v** does not fit in main memory
- **Block stripe update method**: break **M** into vertical stripes and break **r** into corresponding horizontal stripes
- **Allows us to execute MapReduce efficiently**
- Process no more of **v** at any one Map task than can fit in main memory
- **Trying to use a combiner gets more complicated**: Section 5.2.3
  - **Partition matrix M into k^2 blocks, vector r in k stripes**
  - **Use k^2 Map tasks**
  - Each Map task gets: **one square of matrix: M_{ij}, one stripe of vector, r_j**
  - **Each stripe r_j sent to k Map tasks**, transmitted over network k times
  - **But each M_{ij} transmitted only once**

Some Problems with Page Rank

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

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, history, and computer security

Topic-Specific PageRank

- Random walker has a small probability of teleporting at any step
- **Where a Teleport can go:**
  - **Standard PageRank: Any page with equal probability**
  - **Topic Specific PageRank: A topic-specific set of "relevant" pages (teleport set)**
- **Idea: Bias the random walk**
  - When walker teleports, she picks 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/query
  - **For each teleport set *S*, we get a different vector r_S**

Discover Topic PageRank Vectors r_s for Topic Set S

- **If the search engine can deduce the user's interests, then it can do a better job of returning relevant pages**
  - A private PageRank vector that gives importance of each page to that user (not feasible!)
- **Topic-Sensitive PageRank: creates one vector for some small number of topics**
  - **Bias PageRank to favor pages of that topic**
- **Create different PageRank vectors for different topics**
  - The 16 top-level categories of the Open Directory (DMOZ): arts, business, sports,…
- If we can **determine that the user is interested in one of these topics** (e.g., by content of pages they recently viewed)**, use the PageRank vector for that topic when deciding on ranking of pages** for query results

Biased Random Walk

- Bias toward a set S of pages
  - S called teleport set
  - e_S: a vector with 1 for pages in S, 0 otherwise

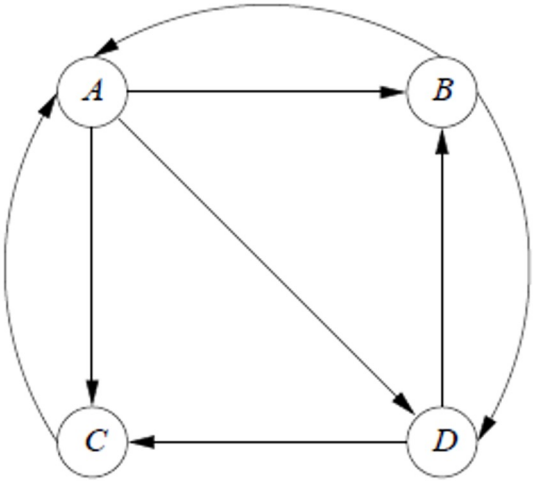
$$\mathbf{v}' = \beta M \mathbf{v} + (1 - \beta)\mathbf{e}_S/|S|$$

Example

- Teleport set S = {B, D}, β = .8

M =

	A	B	C	D
A	0	1/2	1	0
B	1/3	0	0	1/2
C	1/3	0	0	1/2
D	1/3	1/2	0	0

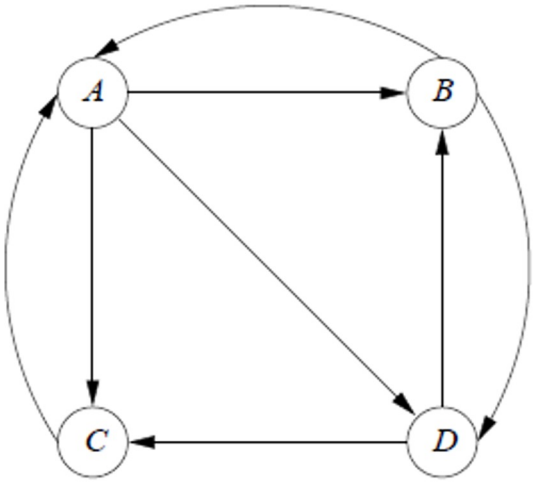


$$\mathbf{v}' = \beta M \mathbf{v} + (1-\beta)\mathbf{e}_S/|S|$$

$$\mathbf{v}' = \begin{bmatrix} 0 & 2/5 & 4/5 & 0 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 0 & 0 & 2/5 \\ 4/15 & 2/5 & 0 & 0 \end{bmatrix} \mathbf{v} + \begin{bmatrix} 0 \\ 1/10 \\ 0 \\ 1/10 \end{bmatrix}$$

0.2 * 1/2

Example



$$\begin{bmatrix}1/4\\1/4\\1/4\\1/4\end{bmatrix}\ \begin{bmatrix}9/24\\5/24\\5/24\\5/24\end{bmatrix}\ \begin{bmatrix}15/48\\11/48\\11/48\\11/48\end{bmatrix}\ \begin{bmatrix}11/32\\7/32\\7/32\\7/32\end{bmatrix},\ldots,\ \begin{bmatrix}3/9\\2/9\\2/9\\2/9\end{bmatrix}$$

Unbiased: A has the largest

$$\begin{bmatrix}0/2\\1/2\\0/2\\1/2\end{bmatrix},\begin{bmatrix}2/10\\3/10\\2/10\\3/10\end{bmatrix},\begin{bmatrix}42/150\\41/150\\26/150\\41/150\end{bmatrix},\begin{bmatrix}62/250\\71/250\\46/250\\71/250\end{bmatrix},\ldots,\begin{bmatrix}54/210\\59/210\\38/210\\59/210\end{bmatrix}$$

Biased: B and D has largest PageRanks

Integrating Topic-Sensitive PageRank Into a Search Engine

1. **Decide on topics** for which to create specialized PageRank vectors

1. **Pick a teleport set for each of these topics**
   - Use that set to **compute topic-sensitive PageRank for that topic**

1. **Determine the topic or set of topics most relevant to a particular search query**
   - **More on this in next slide**

1. **Use the PageRank vector for those topic(s) to order the responses to the search query**

How to Select the Topic Set that is Most Relevant to the Search Query?

- **Tricky**
- **Several methods proposed:**

1. **User can pick from a topic menu**
2. **Infer the topic(s) from the context of the query**
   - **Words that appear in Web pages recently searched by user or recent user queries**
   - E.g., query is launched from a web page talking about a known topic
     - History of queries: e.g., "basketball" followed by "Kobe"
   - **Classification of documents by topic:** studied for decades
3. **Infer the topic(s) from information about the user**
   - E.g., the **user's bookmarks, their interest on Facebook**, etc.

TrustRank:
Combating Web Spam

What is Web Spam?

- **Spamming:**
  - Deliberate action to boost a webpage's position in search results
  - Not commensurate with page's real value
- This is a very broad definition
  - **SEO** industry might disagree!
  - SEO = search engine optimization
- Approximately **10-15%** of web pages are spam

Web Search

- **Early search engines:**
  - Crawl the Web
  - Index pages by words they contained
  - Respond to search queries (lists of words) with the pages containing those words
- **Early page ranking:**
  - Attempt to order pages matching a search query by "importance"
  - **First search engines considered:**
    - **(1) Number of times** query words appeared
    - **(2)** Prominence of **word position**, e.g. title, header

First Spammers: Term Spam

- **How do you make your page appear to be about movies?**
- **(1) Add the word "movie" 1,000 times to your page**
  - Set text color to the background color, so only search engines would see it
- **(2)** Or**, run the query "movie" on your target search engine**
  - See what page came first in the listings
  - **Copy it into your page, make it "invisible"**
- **These and similar techniques are called term spam**

Google's Solution to Term Spam

- **Believe what people say about you, rather than what you say about yourself**
  - Use words in the **anchor text (words that appear underlined to represent the link)** and its surrounding text
- **PageRank** as a tool to measure the "importance" of Web pages
  - **Doesn't depend on content of pages**

Why It Works?

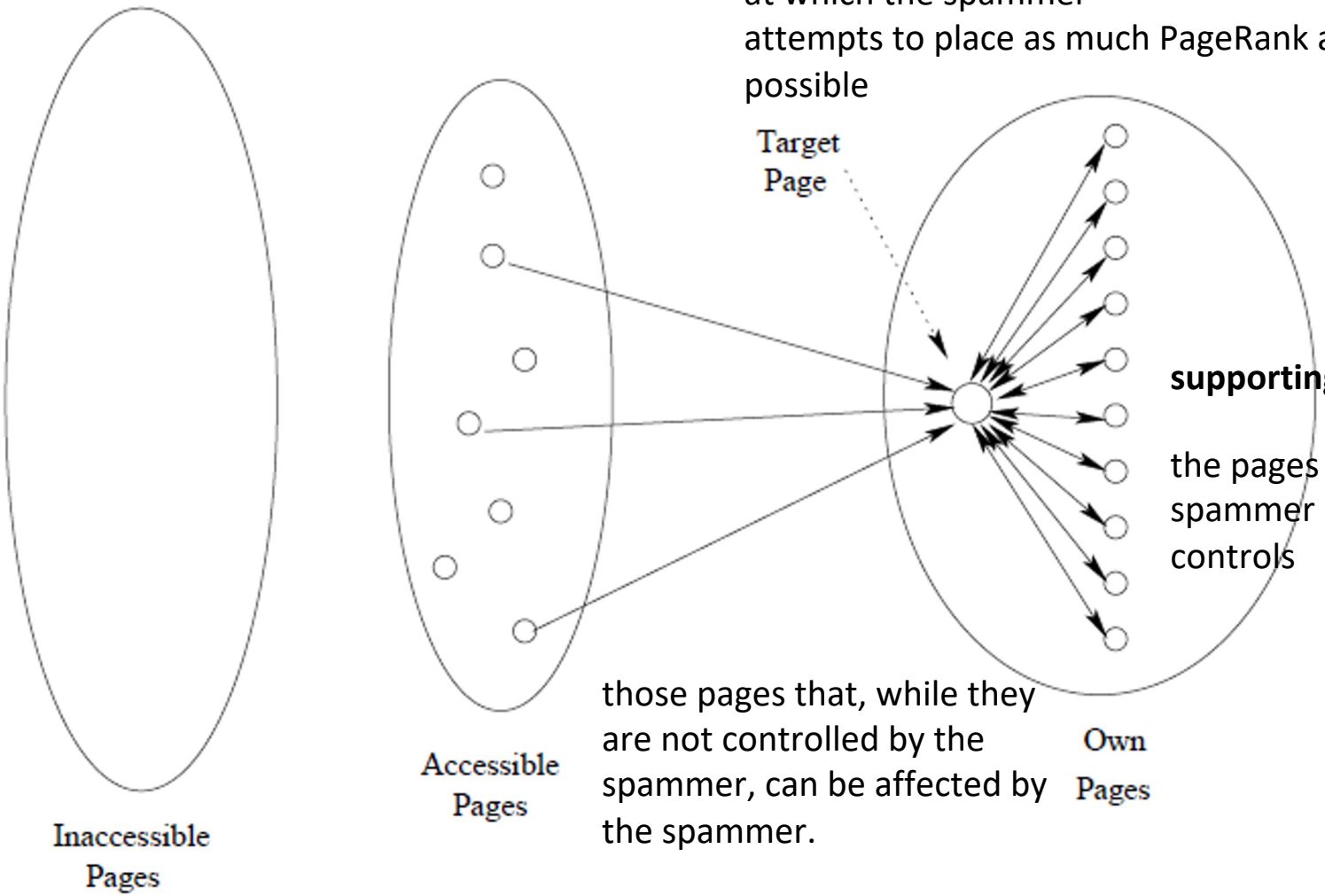
- **Our hypothetical shirt-seller loses**
  - Saying he is about "movies" doesn't help, because **others don't say he is about movies**
  - His page isn't very important, so it **won't be ranked high for shirts or movies**
- **Example:**
  - Shirt-seller creates 1,000 pages, each links to his with "movie" in the anchor text
  - **These pages have no links in, so they get little PageRank**
  - So the shirt-seller **can't beat truly important movie pages, like IMDB**

Google vs. Spammers: Round 2!

- Once Google became the dominant search engine, spammers began to work out ways to fool Google
- **Spam farms** were developed to concentrate PageRank on a single page
- **Link spam:**
  - Creating link structures that boost PageRank of a particular page

Spam Farm

at which the spammer attempts to place as much PageRank as possible



supporting pages

the pages that the spammer owns and controls

those pages that, while they are not controlled by the spammer, can be affected by the spammer.

the pages that the spammer cannot affect. Most of the Web is in this part

Spam Farm Analysis

- **x:** contributed ranks to target page t from accessible pages
- **y:** PageRank of target page
- **m:** # of supporting pages
- $\boldsymbol{\beta}$: damping factor (so 1- β is tax percentage)
- **n:** # of pages in the entire Web
- PageRank of each supporting page:

$$\beta\, y/m + (1 - \beta)/\text{n}$$

Compute PageRank of Target Page

- Contributed by supporting pages
  - $\beta m(\beta\, y/m + (1 - \beta)/n)$
- Contributed by accessible pages
  - x
- Teleporting
  - $(1-\beta)/n$, negligible; $y = x + \beta m(\beta \frac{y}{m} + (1-\beta)/n) = x + \beta^2 y + \frac{\beta(1-\beta)m}{n}$

$$y = \frac{x}{1-\beta^2} + \frac{\beta m}{(1+\beta)n} = \frac{x}{1-\beta^2} + c\frac{m}{n},$$

where $c = \frac{\beta}{1+\beta}$

If β = .85, then $1/(1-\beta^2)$ = 3.6, c = .46

Amplify external contribution by 360%

Plus additional PageRank: 46% of fraction of farm pages in the entire Web

HITS: Hubs and Authorities

HITS Ideas

- A variant of topic-sensitive PageRank
  - Topic = a set of trustworthy pages

- Identify trustworthy pages
  1. Manually examine pages with high Page Ranks
  2. Pick pages within controlled domains: edu, gov, mil
  3. Avoid borderline pages, e.g., blog sites, forums

HITS Algorithm

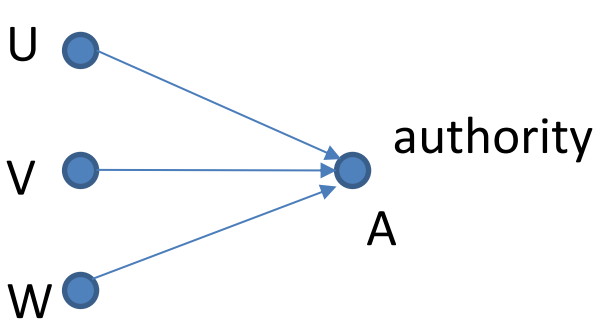
- Hyperlink-induced topic search

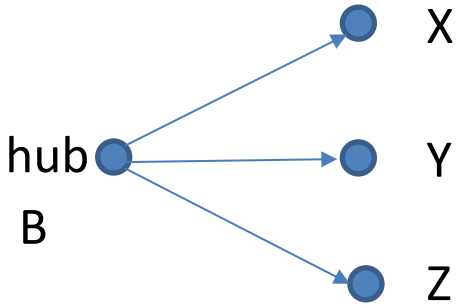
- Each page has two scores
  - Hub: does it provide many pointers to the topic?
  - Authority: how important the page is about topic?

- Examples
  - Hub: department pages listing all courses
  - Authority: individual course pages

Authority and Hub

- A good authority page is one
  - pointed by many good hub pages

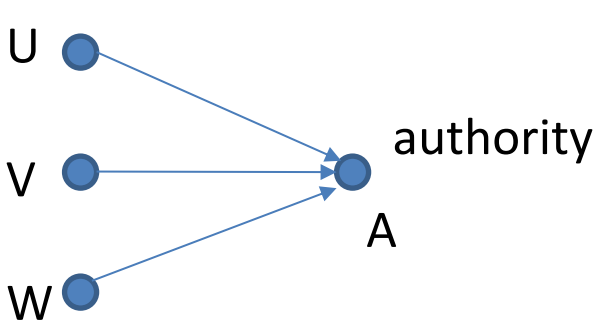


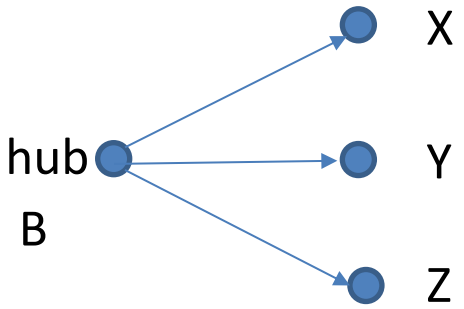


- A good hub page is one
  - pointing to many good authority pages

Computing Authority and Hubbiness

- Authority of a page =
  - sum of hubbiness of pages pointing to it
  - a(A) = h(U) + h(V) + h(W)
- Hubbiness of a page =
  - sum of authority of pages it points to
  - h(B) = a(X) + a(Y) + a(Z)



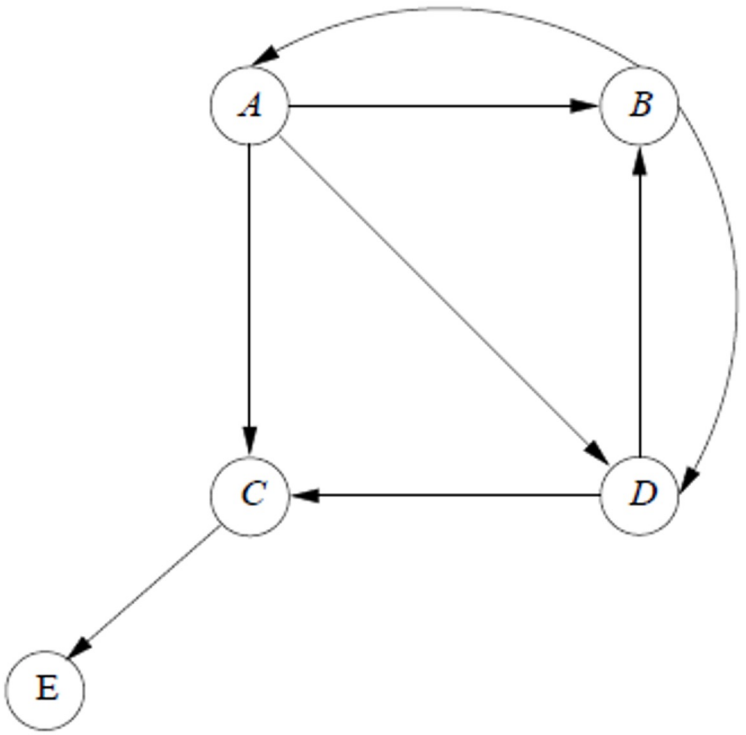


Link Matrix

- L[i,j] = 1 if there is a link from node i to node j
- L^T is the un-normalized M in PageRank

$$L = \begin{array}{c} \text{A} \\ \\ \\ \\ \\ \end{array} \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$L^{\mathrm{T}} = \begin{array}{c} \text{A} \\ \\ \\ \\ \\ \end{array} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$



Computing Authority and Hubbiness

- $\boldsymbol{h} = \lambda L \boldsymbol{a}, \quad \boldsymbol{a} = \mu L^T \boldsymbol{h}$
  - λ, μ: scaling factors to normalize $\boldsymbol{h}$ and $\boldsymbol{a}$
- $\boldsymbol{h} = \lambda\mu L L^T \boldsymbol{h}, \ \boldsymbol{a} = \lambda\mu L^T L \boldsymbol{a}$

- LL^T and $L^T L$ likely dense
  - Expensive to use direct recursion on $\boldsymbol{h}$ and $\boldsymbol{a}$
  - e.g., iteratively computing $\boldsymbol{h} = \lambda\mu L L^T \boldsymbol{h}$

- Use mutual recursion instead

Compute via Mutual Recursion

1. Start with $\boldsymbol{h}$ = a vector of all 1's

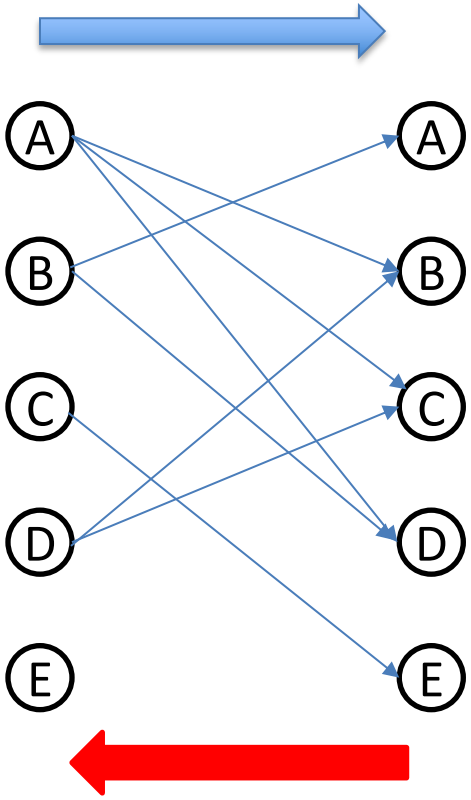
1. Compute $\boldsymbol{a} = L^T\boldsymbol{h}$, then scale it so that largest component is 1

1. Compute $\boldsymbol{h} = L\boldsymbol{a}$ and similarly scaled

1. Repeat steps 2 and 3 until $\boldsymbol{h}$ and $\boldsymbol{a}$ do not change much

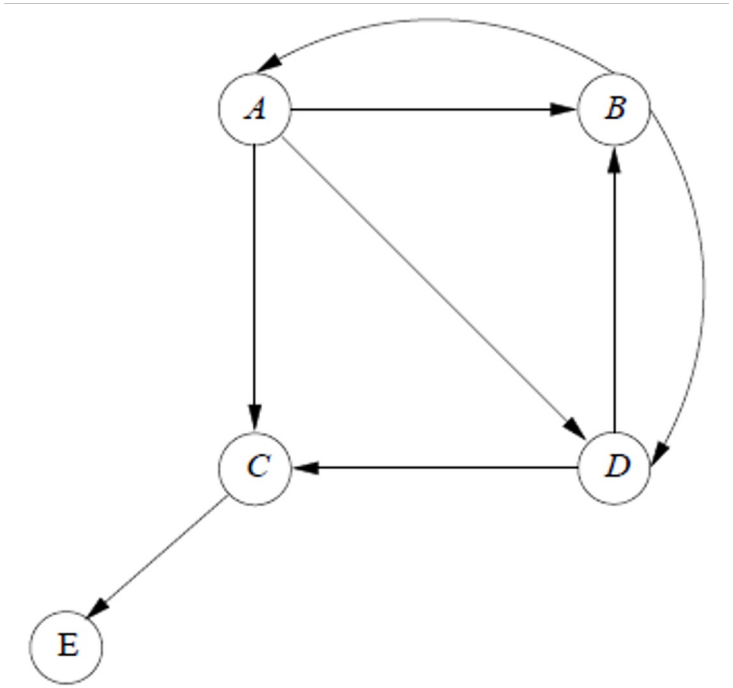
Bipartite View of Iterations

Compute authority from hub



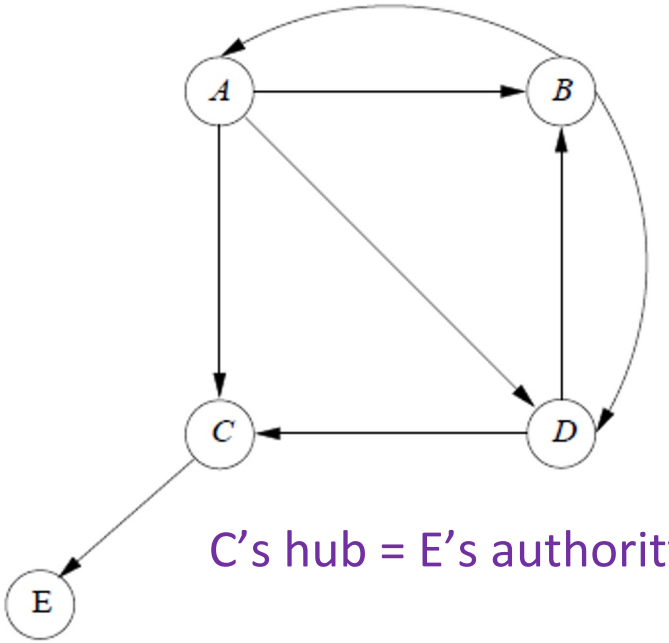
Compute hub from authority

Hub Authority



Example

$$\boldsymbol{h}^0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$



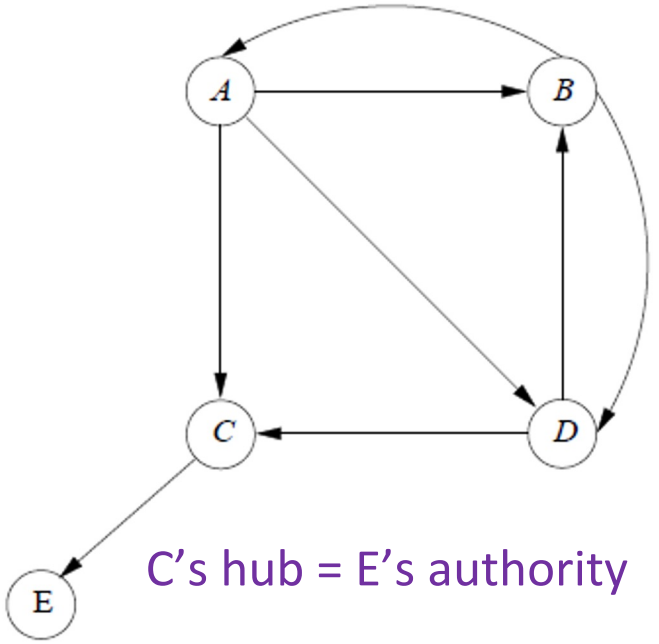
C's hub = E's authority

$$a^0 = L^T \boldsymbol{h}^0 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1 \\ 1 \\ 1 \\ 1/2 \end{bmatrix}$$

Can you predict the changes to hub scores next?

Example

$$\boldsymbol{h}^0 = \begin{bmatrix}1\\1\\1\\1\\1\end{bmatrix} \quad a^0 = \begin{bmatrix}1/2\\1\\1\\1\\1/2\end{bmatrix}$$



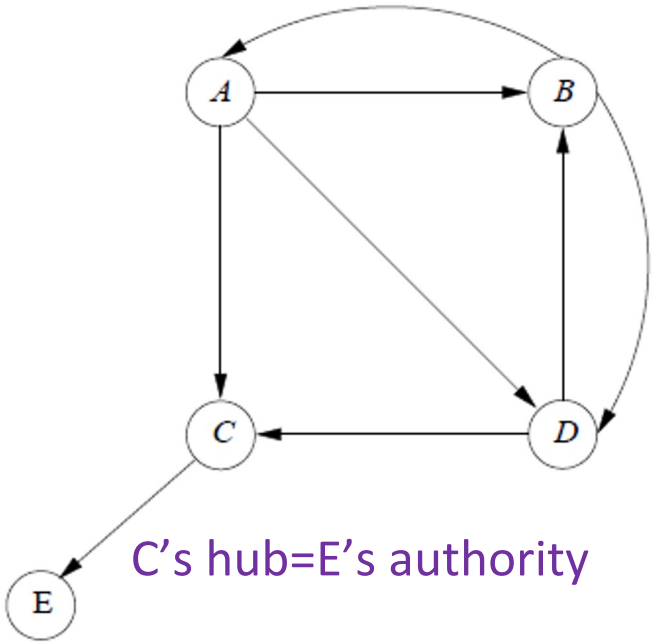
C's hub = E's authority

$$h^1 = L\boldsymbol{a}^0 = \begin{bmatrix}0&1&1&1&0\\1&0&0&1&0\\0&0&0&0&1\\0&1&1&0&0\\0&0&0&0&0\end{bmatrix}\begin{bmatrix}1/2\\1\\1\\1\\1/2\end{bmatrix} = \begin{bmatrix}3\\3/2\\1/2\\2\\0\end{bmatrix} = \begin{bmatrix}1\\1/2\\1/6\\2/3\\0\end{bmatrix}$$

Will E's authority decrease? Will E have > 0 hub score?

Example

$$\boldsymbol{h}^0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad a^0 = \begin{bmatrix} 1/2 \\ 1 \\ 1 \\ 1 \\ 1/2 \end{bmatrix}$$



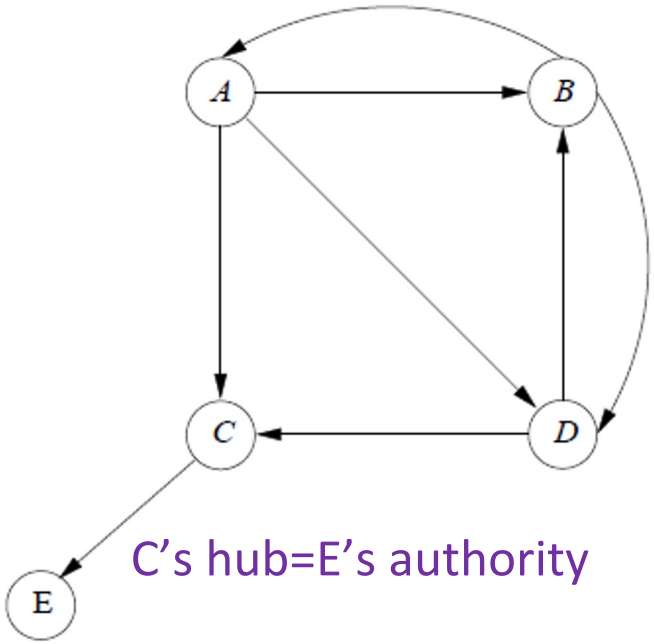
C's hub=E's authority

$$h^1 = \begin{bmatrix} 1 \\ 1/2 \\ 1/6 \\ 2/3 \\ 0 \end{bmatrix} \quad a^1 = \begin{bmatrix} 1/2 \\ 5/3 \\ 5/3 \\ 3/2 \\ 1/6 \end{bmatrix} = \begin{bmatrix} 3/10 \\ 1 \\ 1 \\ 9/10 \\ 1/10 \end{bmatrix} \quad h^2 = \begin{bmatrix} 29/10 \\ 6/5 \\ 1/10 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 12/29 \\ 1/29 \\ 20/29 \\ 0 \end{bmatrix}$$

Decrease due to normalization

Example

$$\boldsymbol{h}^0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad a^0 = \begin{bmatrix} 1/2 \\ 1 \\ 1 \\ 1 \\ 1/2 \end{bmatrix}$$



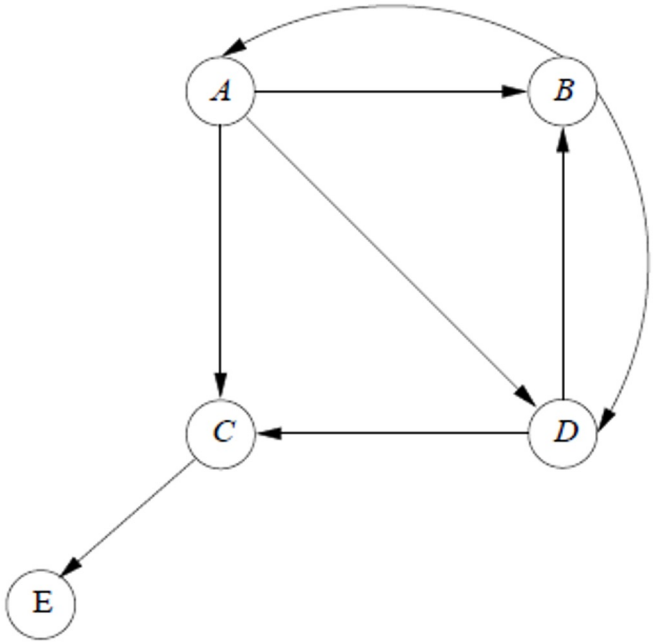
C's hub=E's authority

$$h^1 = \begin{bmatrix} 1 \\ 1/2 \\ 1/6 \\ 2/3 \\ 0 \end{bmatrix} \quad a^1 = \begin{bmatrix} 3/10 \\ 1 \\ 1 \\ 9/10 \\ 1/10 \end{bmatrix} \quad h^2 = \begin{bmatrix} 1 \\ 12/29 \\ 1/29 \\ 20/29 \\ 0 \end{bmatrix} \quad a^2 = \begin{bmatrix} 12/29 \\ 49/29 \\ 49/29 \\ 41/29 \\ 1/29 \end{bmatrix} = \begin{bmatrix} 12/49 \\ 1 \\ 1 \\ 41/49 \\ 1/49 \end{bmatrix}$$

Decrease due to normalization

At the Limit

- A: greatest hub
- B, C: greatest authority



$$\boldsymbol{h}^0 = \begin{bmatrix}1\\1\\1\\1\\1\end{bmatrix} \boldsymbol{h}^1 = \begin{bmatrix}1\\1/2\\1/6\\2/3\\0\end{bmatrix} \boldsymbol{h}^2 = \begin{bmatrix}1\\12/29\\1/29\\20/29\\0\end{bmatrix}, \ldots, \boldsymbol{h}^n = \begin{bmatrix}1\\.36\\0\\.72\\0\end{bmatrix}$$

$$\boldsymbol{a}^0 = \begin{bmatrix}1/2\\1\\1\\1\\1/2\end{bmatrix} \boldsymbol{a}^1 = \begin{bmatrix}3/10\\1\\1\\9/10\\1/10\end{bmatrix} \boldsymbol{a}^2 = \begin{bmatrix}12/49\\1\\1\\41/49\\1/49\end{bmatrix}, \ldots, \boldsymbol{a}^n = \begin{bmatrix}.21\\1\\1\\.79\\0\end{bmatrix}$$

PageRank and HITS

- PageRank and HITS are two solutions to the same problem
- Rank the relative value of pages
- Use these values in search engine algorithms to determine the order of search results