Data Mining

Link Analysis Algorithms

Page Rank,
Hubs and authorities

Slides of Anand Rajaraman, Jeffrey D. Ullman

Link Analysis Algorithms

- Motivation

- Page Rank
- Topic-Specific Page Rank
- Hubs and Authorities

- Conclusion

Motivation: Link Analysis

- Search engines
  - Serve user's information needs
  - Find relevant results based on keywords
- Spammers
  - Try to attract traffic to their sites
  - Misguide search engines
    - Link farms, fake keywords, …
- Idea: use links to determine importance

Link Analysis Algorithms

- Motivation
- Page Rank
- Topic-Specific Page Rank
- Hubs and Authorities
- Conclusion

Pagerank

(Larry Page and Sergey Brin)

- Assess the importance of a page based on links

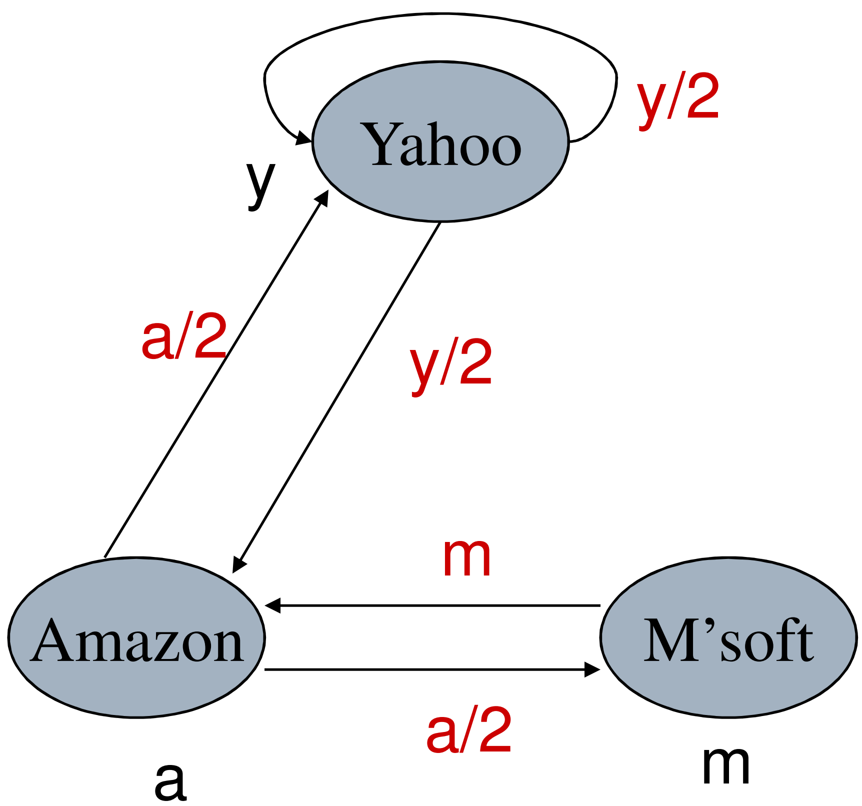
- Measure relative importance of a web page. A page is important if many other important pages link to it
  - Recursive definition

Simple recursive formulation

- Each link's vote is proportional to the importance of its source page
- If page P with importance x has n outlinks, each link gets x/n votes
- Page P's own importance is the sum of the votes on its inlinks

Simple "flow" model

The web in 1839



$$y = y/2 + a/2$$

$$a = y/2 + m$$

$$m = a/2$$

Solving the flow equations

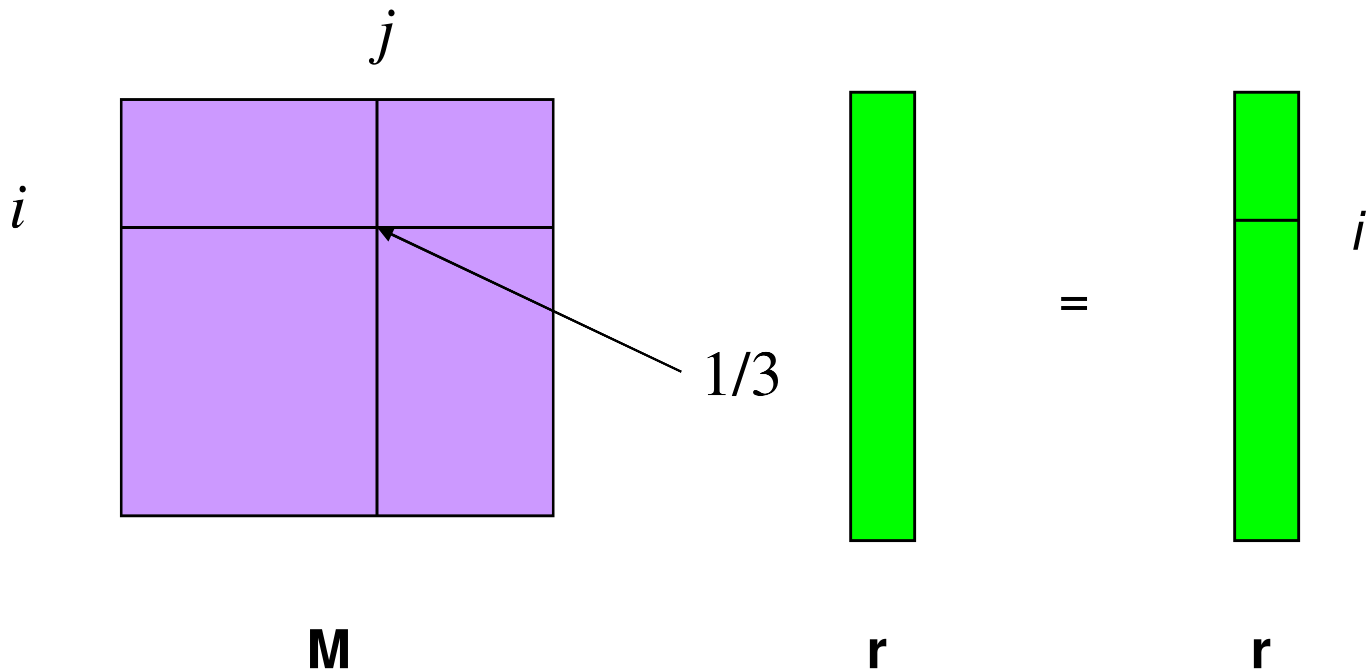
- 3 equations, 3 unknowns, no constants
  - No unique solution
  - All solutions equivalent modulo scale factor
- Additional constraint forces uniqueness
  - $y+a+m = 1$
  - $y = 2/5$, $a = 2/5$, $m = 1/5$
- Gaussian elimination method works for small examples, but we need a better method for large graphs

Matrix formulation

- Matrix **M** has one row and one column for each web page
- Suppose page j has n outlinks
  - If $j \Rightarrow i$, then $M_{ij}=1/n$
  - Else $M_{ij}=0$
- **M** is a column stochastic matrix
  - Columns sum to 1
- Suppose **r** is a vector with one entry per web page
  - r_i is the importance score of page i
  - Call it the rank vector
  - $|\mathbf{r}| = 1$

Example

Suppose page *j* links to 3 pages, including *i*



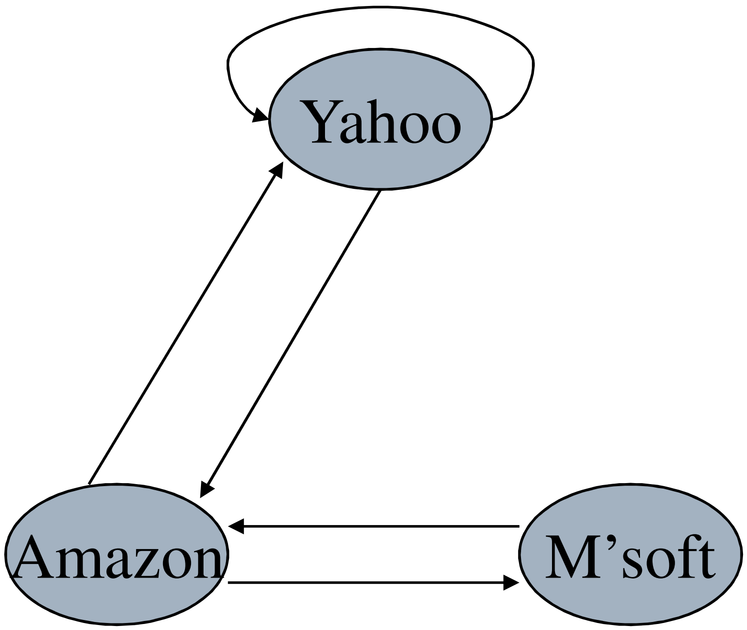
Eigenvector formulation

- The flow equations can be written

$$\mathbf{r} = \mathbf{M}\mathbf{r}$$

- So the rank vector is an eigenvector of the stochastic web matrix
  - In fact, its first or principal eigenvector, with corresponding eigenvalue 1

Example



$$y = y/2 + a/2$$
$$a = y/2 + m$$
$$m = a/2$$

$$\begin{array}{c|ccc|} & y & a & m \\ \hline y & 1/2 & 1/2 & 0 \\ a & 1/2 & 0 & 1 \\ m & 0 & 1/2 & 0 \\ \hline \end{array}$$

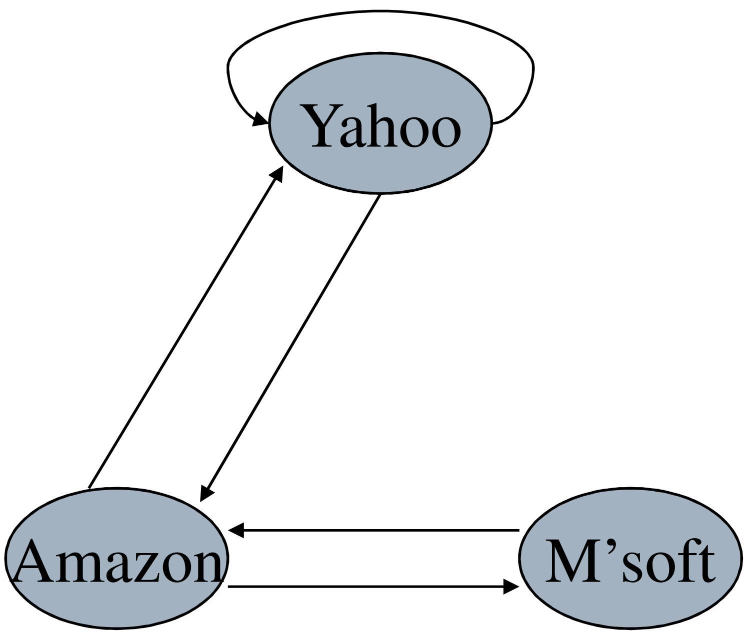
r = **Mr**

$$\begin{bmatrix} y \\ a \\ m \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} y \\ a \\ m \end{bmatrix}$$

Power Iteration method

- Simple iterative scheme (aka relaxation)
- Suppose there are N web pages
- Initialize: $\mathbf{r}^0 = [1/N,\ldots.,1/N]^T$
- Iterate: $\mathbf{r}^{k+1} = \mathbf{M}\mathbf{r}^k$
- Stop when $|\mathbf{r}^{k+1} - \mathbf{r}^k|_1 < \varepsilon$
  - $|\mathbf{x}|_1 = \sum_{1 \le i \le N} |x_i|$ is the L_1 norm
  - Can use any other vector norm e.g., Euclidean

Power Iteration Example



$$\begin{array}{c|ccc|}
 & y & a & m \\
\hline
y & 1/2 & 1/2 & 0 \\
a & 1/2 & 0 & 1 \\
m & 0 & 1/2 & 0 \\
\hline
\end{array}$$

$$\begin{array}{c} y \\ a \\ m \end{array} = \begin{array}{c} 1/3 \\ 1/3 \\ 1/3 \end{array} \quad \begin{array}{c} 1/3 \\ 1/2 \\ 1/6 \end{array} \quad \begin{array}{c} 5/12 \\ 1/3 \\ 1/4 \end{array} \quad \begin{array}{c} 3/8 \\ 11/24 \\ 1/6 \end{array} \quad \ldots \quad \begin{array}{c} 2/5 \\ 2/5 \\ 1/5 \end{array}$$

Random Walk Interpretation

- Imagine a random web surfer
  - At any time t, surfer is on some page P
  - At time $t+1$, the surfer follows an outlink from P uniformly at random
  - Ends up on some page Q linked from P
  - Process repeats indefinitely
- Let $\mathbf{p}(t)$ be a vector whose i^{th} component is the probability that the surfer is at page i at time t
  - $\mathbf{p}(t)$ is a probability distribution on pages

The stationary distribution

- Where is the surfer at time t+1?
  - Follows a link uniformly at random
  - $\mathbf{p}(t+1) = \mathbf{Mp}(t)$
- Suppose the random walk reaches a state such that $\mathbf{p}(t+1) = \mathbf{Mp}(t) = \mathbf{p}(t)$
  - Then $\mathbf{p}(t)$ is called a stationary distribution for the random walk
- Our rank vector $\mathbf{r}$ satisfies $\mathbf{r} = \mathbf{Mr}$
  - So it is a stationary distribution for the random surfer

Existence and Uniqueness

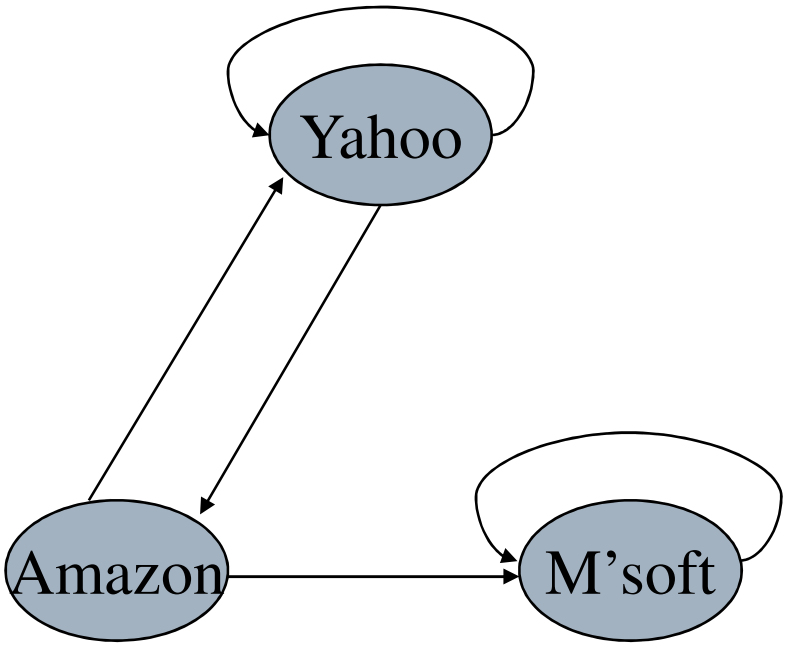
A central result from the theory of random walks (aka Markov processes):

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 $t = 0$.

Spider traps

- A group of pages is a spider trap if there are no links from within the group to outside the group
  - Random surfer gets trapped
- Spider traps violate the conditions needed for the random walk theorem

Microsoft becomes a spider trap



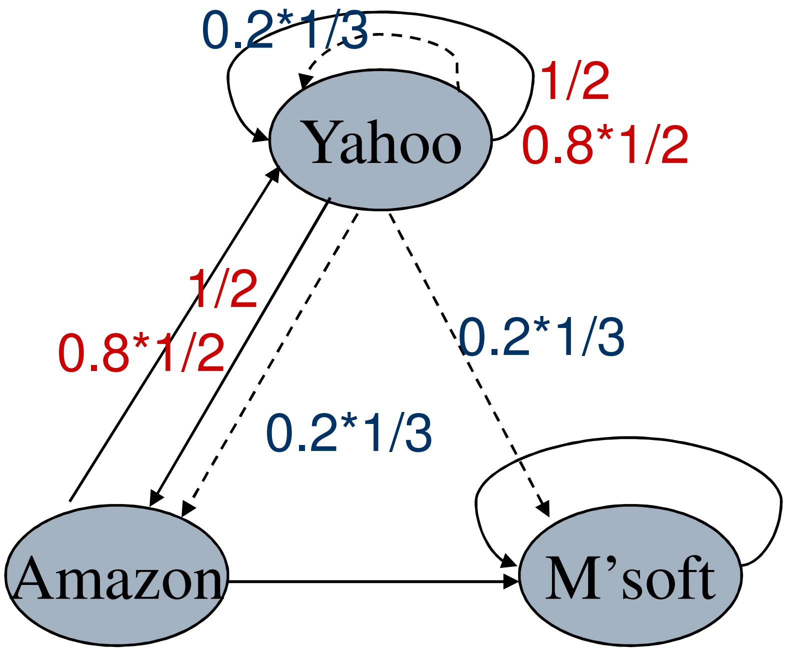
$$\begin{array}{c|ccc|} & y & a & m \\ \hline y & 1/2 & 1/2 & 0 \\ a & 1/2 & 0 & 0 \\ m & 0 & 1/2 & 1 \\ \hline \end{array}$$

y		1	1	3/4	5/8		0
a	=	1	1/2	1/2	3/8	. . .	0
m		1	3/2	7/4	2		3

Random teleports

- The Google solution for spider traps
- At each time step, the random surfer has two options:
  - With probability β, follow a link at random
  - With probability 1-β, jump to some page uniformly at random
  - 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 ($\beta = 0.8$)

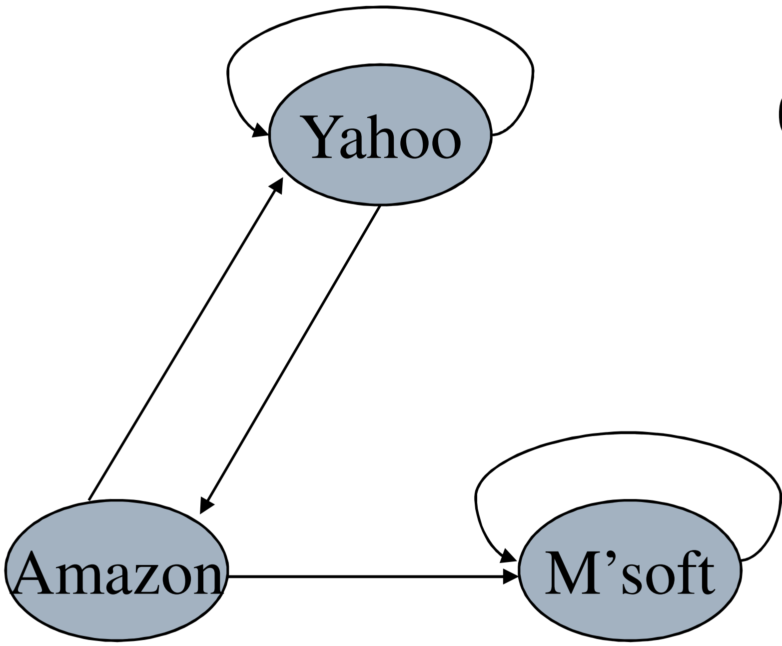


$$\begin{array}{cc} & y \\ \begin{array}{c} y \\ a \\ m \end{array} & \boxed{\begin{array}{c} 1/2 \\ 1/2 \\ 0 \end{array}} \end{array} \qquad 0.8 * \begin{array}{c} y \\ \boxed{\begin{array}{c} 1/2 \\ 1/2 \\ 0 \end{array}} \end{array} + 0.2 * \begin{array}{c} y \\ \boxed{\begin{array}{c} 1/3 \\ 1/3 \\ 1/3 \end{array}} \end{array}$$

$$0.8 \boxed{\begin{array}{ccc} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{array}} + 0.2 \boxed{\begin{array}{ccc} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{array}}$$

$$\begin{array}{cc} \begin{array}{c} y \\ a \\ m \end{array} & \boxed{\begin{array}{ccc} 7/15 & 7/15 & 1/15 \\ 7/15 & 1/15 & 1/15 \\ 1/15 & 7/15 & 13/15 \end{array}} \end{array}$$

Random teleports ($\beta = 0.8$)



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

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

$$\begin{array}{cccccccc} y & & 1 & 1.00 & 0.84 & 0.776 & & 7/11 \\ a & = & 1 & 0.60 & 0.60 & 0.536 & \ldots & 5/11 \\ m & & 1 & 1.40 & 1.56 & 1.688 & & 21/11 \end{array}$$

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 $j \Rightarrow i$ 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 redistribute evenly

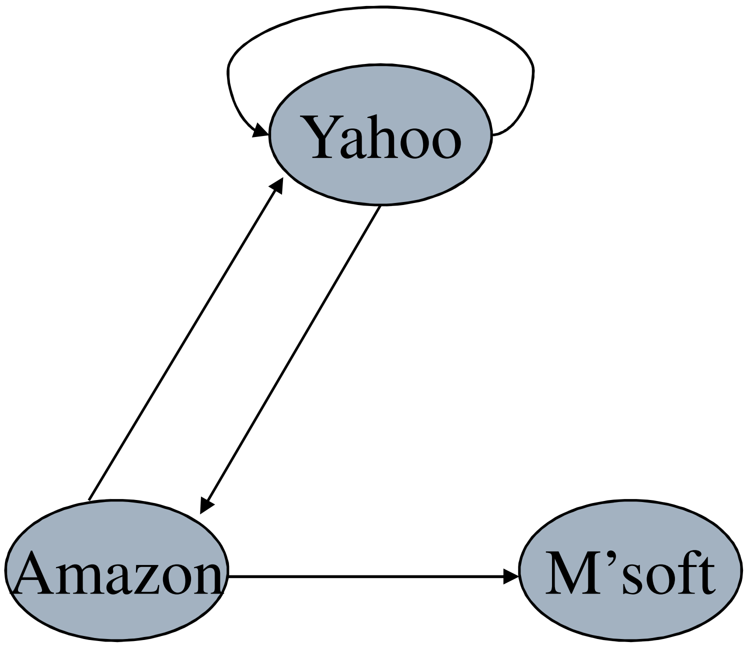
Page Rank

- Construct the NxN matrix **A** as follows
  - $A_{ij} = \beta M_{ij} + (1-\beta)/N$
- Verify that **A** is a stochastic matrix
- The page rank vector **r** is the principal eigenvector of this matrix
  - satisfying **r** = **Ar**
- Equivalently, **r** is the stationary distribution of the random walk with teleports

Dead ends

- Pages with no outlinks are "dead ends" for the random surfer
  - Nowhere to go on next step

Microsoft becomes a dead end



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

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

Non-stochastic!

$$\begin{array}{c} y \\ a \\ m \end{array} = \begin{array}{c} 1 \\ 1 \\ 1 \end{array} \quad \begin{array}{c} 1 \\ 0.6 \\ 0.6 \end{array} \quad \begin{array}{c} 0.787 \\ 0.547 \\ 0.387 \end{array} \quad \begin{array}{c} 0.648 \\ 0.430 \\ 0.333 \end{array} \quad \ldots \quad \begin{array}{c} 0 \\ 0 \\ 0 \end{array}$$

Dealing with dead-ends

- Teleport
  - Follow random teleport links with probability 1.0 from dead-ends
  - Adjust matrix accordingly
- Prune and propagate
  - Preprocess the graph to eliminate dead-ends
  - Might require multiple passes
  - Compute page rank on reduced graph
  - Approximate values for deadends by propagating values from reduced graph

Computing page rank

- Key step is matrix-vector multiplication
  - $\mathbf{r}^{new} = \mathbf{A}\mathbf{r}^{old}$
- Easy if we have enough main memory to hold $\mathbf{A}$, $\mathbf{r}^{old}$, $\mathbf{r}^{new}$
- Say N = 1 billion pages
  - We need 4 bytes for each entry (say)
  - 2 billion entries for vectors, approx 8GB
  - Matrix A has N^2 entries
    - 10^{18} is a large number!

Rearranging the equation

$\mathbf{r} = \mathbf{A}\mathbf{r}$, where

$A_{ij} = \beta M_{ij} + (1-\beta)/N$

$r_i = \sum_{1 \le j \le N} A_{ij}\, r_j$

$r_i = \sum_{1 \le j \le N} [\beta M_{ij} + (1-\beta)/N]\, r_j$

$\quad = \beta \sum_{1 \le j \le N} M_{ij}\, r_j + (1-\beta)/N \sum_{1 \le j \le N} r_j$

$\quad = \beta \sum_{1 \le j \le N} M_{ij}\, r_j + (1-\beta)/N$, since $|\mathbf{r}| = 1$

$\mathbf{r} = \beta \mathbf{M}\mathbf{r} + [(1-\beta)/N]_N$

where $[x]_N$ is an N-vector with all entries x

Sparse matrix formulation

- We can rearrange the page rank equation:
  - $\mathbf{r} = \beta\mathbf{M}\mathbf{r} + [(1-\beta)/N]_N$
  - $[(1-\beta)/N]_N$ is an N-vector with all entries $(1-\beta)/N$
- **M** is a sparse matrix!
  - 10 links per node, approx 10N entries
- So in each iteration, we need to:
  - Compute $\mathbf{r}^{new} = \beta\mathbf{M}\mathbf{r}^{old}$
  - Add a constant value $(1-\beta)/N$ to each entry in $\mathbf{r}^{new}$

Sparse matrix encoding

- Encode sparse matrix using only nonzero entries
  - Space proportional roughly to number of links
  - say 10N, or 4*10*1 billion = 40GB
  - still won't fit in memory, but will fit on disk

source node	degree	destination nodes
0	3	1, 5, 7
1	5	17, 64, 113, 117, 245
2	2	13, 23

Basic Algorithm

- Assume we have enough RAM to fit $\mathbf{r}^{new}$, plus some working memory
  - Store $\mathbf{r}^{old}$ and matrix **M** on disk

Basic Algorithm:

- Initialize: $\mathbf{r}^{old} = [1/N]_N$
- Iterate:
  - Update: Perform a sequential scan of **M** and $\mathbf{r}^{old}$ to update $\mathbf{r}^{new}$
  - Write out $\mathbf{r}^{new}$ to disk as $\mathbf{r}^{old}$ for next iteration
  - Every few iterations, compute $|\mathbf{r}^{new}-\mathbf{r}^{old}|$ and stop if it is below threshold
    - Need to read in both vectors into memory

Update step

Initialize all entries of $\mathbf{r}^{new}$ to $(1-\beta)/N$

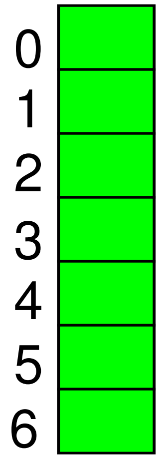
For each page p (out-degree n):

    Read into memory: p, n, $dest_1,\ldots,dest_n$, $r^{old}(p)$

    for j = 1..n:

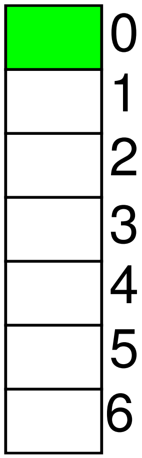
        $r^{new}(dest_j) \mathrel{+}= \beta * r^{old}(p)/n$

r^{new}



src	degree	destination
0	3	1, 5, 6
1	4	17, 64, 113, 117
2	2	13, 23

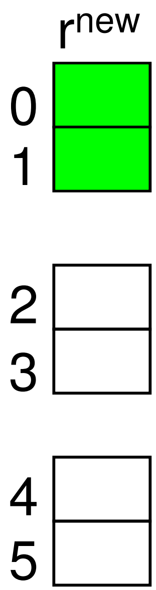
r^{old}



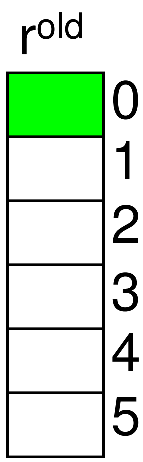
Analysis

- In each iteration, we have to:
  - Read $\mathbf{r}^{old}$ and $\mathbf{M}$
  - Write $\mathbf{r}^{new}$ back to disk
  - IO Cost = $2|\mathbf{r}| + |\mathbf{M}|$
- What if we had enough memory to fit both $\mathbf{r}^{new}$ and $\mathbf{r}^{old}$?
- What if we could not even fit $\mathbf{r}^{new}$ in memory?
  - 10 billion pages

Block-based update algorithm



src	degree	destination
0	4	0, 1, 3, 5
1	2	0, 5
2	2	3, 4



Link Analysis Algorithms

- Motivation

- Page Rank
- Topic-Specific Page Rank
- Hubs and Authorities

- Conclusion

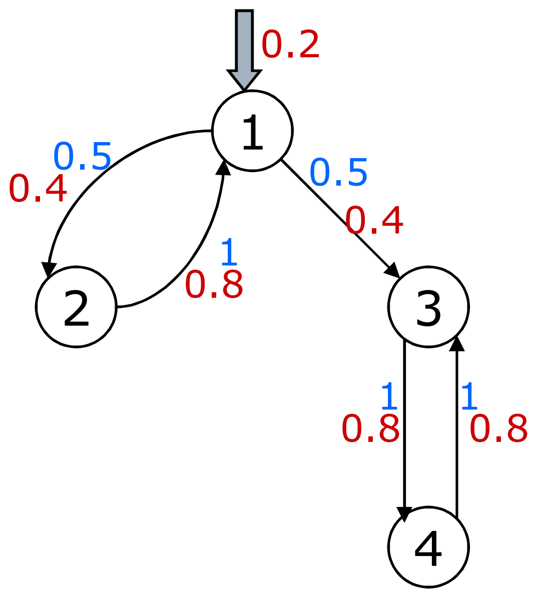
Topic-Specific Page Rank

- Instead of generic popularity, can we measure popularity within a topic?
  - E.g., computer science, health
- Bias the random walk
  - When the random walker teleports, he picks a page from a set S of web pages
  - S contains only pages that are relevant to the topic
  - E.g., Open Directory (DMOZ) pages for a given topic (www.dmoz.org)
- For each teleport set S, we get a different rank vector $\mathbf{r_S}$

Matrix formulation

- $A_{ij} = \beta M_{ij} + (1-\beta)/|S|$ if $i \in S$
- $A_{ij} = \beta M_{ij}$ otherwise
- Show that **A** is stochastic
- We have weighted all pages in the teleport set S equally
  - Could also assign different weights to them

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 page rank vector differently from the unbiased page rank case.

Link Analysis Algorithms

- Motivation
- Page Rank
- Topic-Specific Page Rank
- Hubs and Authorities
- Conclusion

Hubs and Authorities

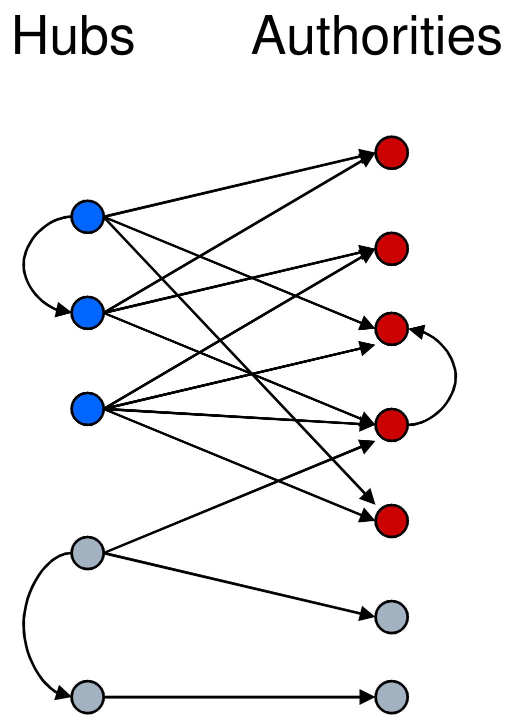
- Suppose we are given a collection of documents on some broad topic
  - e.g., stanford, evolution, iraq
  - perhaps obtained through a text search
- Can we organize these documents in some manner?
  - Page rank offers one solution
  - HITS (Hypertext-Induced Topic Selection) is another
    - proposed at approx the same time (1998)

HITS Model

- Interesting documents fall into two classes

1. Authorities are pages containing useful information
   - course home pages
   - home pages of auto manufacturers
2. Hubs are pages that link to authorities
   - course bulletin
   - list of US auto manufacturers

Idealized view



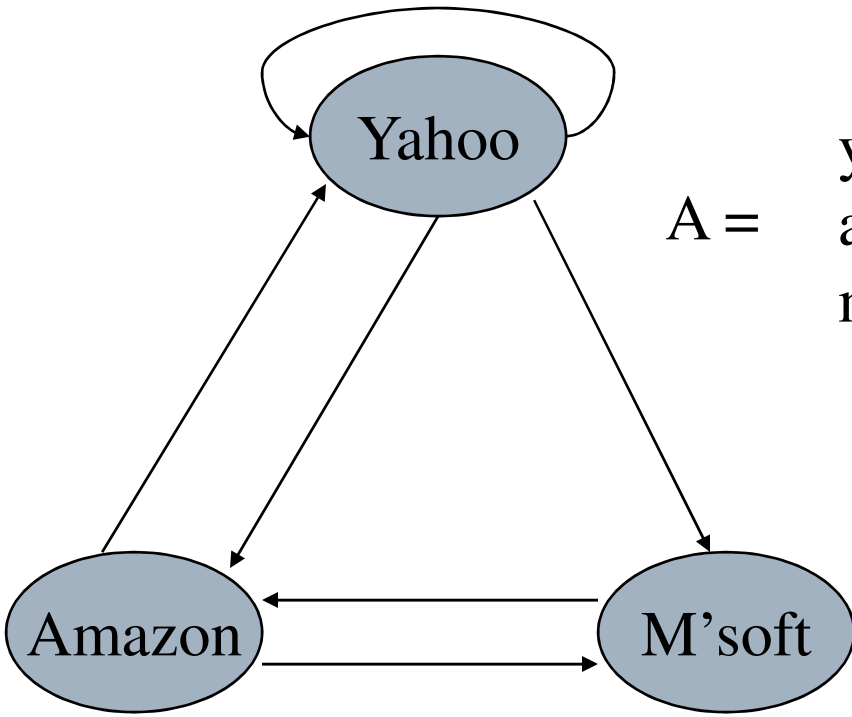
Mutually recursive definition

- 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**

Transition Matrix A

- HITS uses a matrix $A[i, j] = 1$ if page i links to page j, 0 if not
- A^T, the transpose of A, is similar to the PageRank matrix M, but A^T has 1's where M has fractions

Example



$$A = \begin{array}{c|ccc|} & y & a & m \\ \hline y & 1 & 1 & 1 \\ a & 1 & 0 & 1 \\ m & 0 & 1 & 0 \\ \hline \end{array}$$

Hub and Authority Equations

- The hub score of page P is proportional to the sum of the authority scores of the pages it links to
  - $\mathbf{h} = \lambda A \mathbf{a}$
  - Constant λ is a scale factor
- The authority score of page P is proportional to the sum of the hub scores of the pages it is linked from
  - $\mathbf{a} = \mu A^T \mathbf{h}$
  - Constant μ is scale factor

Iterative algorithm

- Initialize **h**, **a** to all 1's
- $\mathbf{h} = \mathbf{A}\mathbf{a}$
- Scale **h** so that its max entry is 1.0
- $\mathbf{a} = \mathbf{A}^T\mathbf{h}$
- Scale **a** so that its max entry is 1.0
- Continue until **h**, **a** converge

Example

$$A = \boxed{\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{matrix}} \qquad A^T = \boxed{\begin{matrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{matrix}}$$

a(yahoo)	=	1	1	1	1	. . .	1
a(amazon)	=	1	1	4/5	0.75	. . .	0.732
a(m'soft)	=	1	1	1	1	. . .	1

h(yahoo)	=	1	1	1	1	. . .	1.000
h(amazon)	=	1	2/3	0.71	0.73	. . .	0.732
h(m'soft)	=	1	1/3	0.29	0.27	. . .	0.268

Existence and Uniqueness

$\mathbf{h} = \lambda A \mathbf{a}$

$\mathbf{a} = \mu A^T \mathbf{h}$

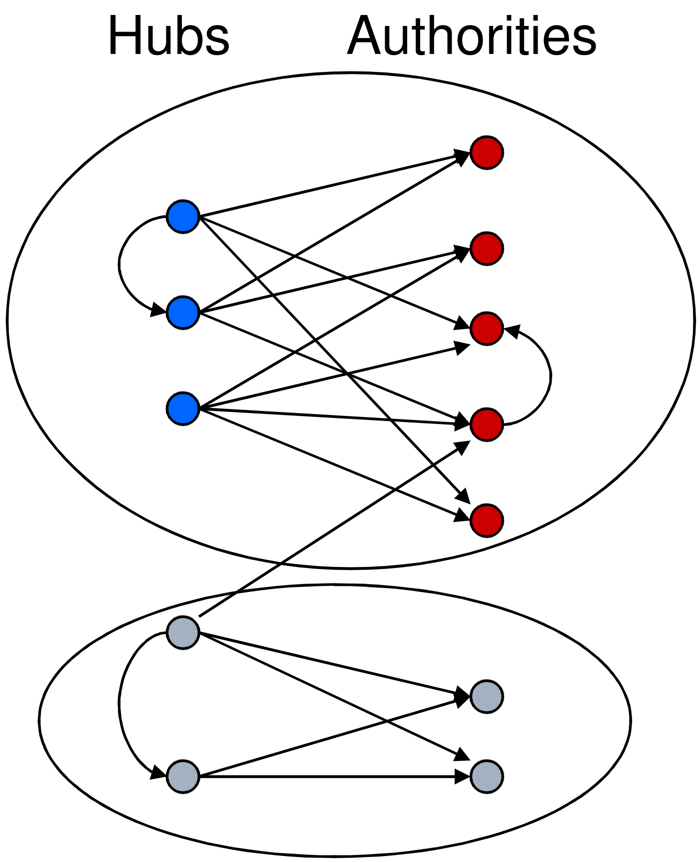
$\mathbf{h} = \lambda\mu A A^T \mathbf{h}$

$\mathbf{a} = \lambda\mu A^T A \ \mathbf{a}$

Under reasonable assumptions about **A**,
the dual iterative algorithm converges to vectors
h* and **a*** such that:

- **h*** is the principal eigenvector of the matrix AA^T
- **a*** is the principal eigenvector of the matrix A^TA

Bipartite cores



Most densely-connected core (primary core)

Less densely-connected core (secondary core)

Secondary cores

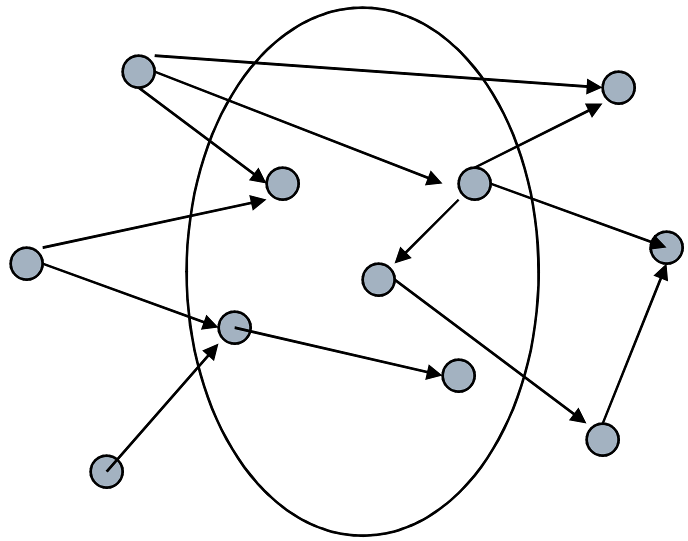
- A single topic can have many bipartite cores
  - corresponding to different meanings, or points of view
  - abortion: pro-choice, pro-life
  - evolution: darwinian, intelligent design
  - jaguar: auto, Mac, NFL team, *panthera onca*
- How to find such secondary cores?

Finding secondary cores

- Once we find the primary core, we can remove its links from the graph
- Repeat HITS algorithm on residual graph to find the next bipartite core
- Roughly, correspond to non-primary eigenvectors of AA^T and A^TA

Creating the graph for HITS

- We need a well-connected graph of pages for HITS to work well



Link Analysis Algorithms

- Motivation

- Page Rank
- Topic-Specific Page Rank
- Hubs and Authorities

- Conclusion

Page Rank and HITS

- Page Rank and HITS are two solutions to the same problem
  - What is the value of an inlink from S to D?
  - In the page rank model, the value of the link depends on the links **into** S
  - In the HITS model, it depends on the value of the other links **out of** S
- The destinies of Page Rank and HITS post-1998 were very different