Personalized PageRank, HITS, Web spam

CS345a: Data Mining

Jure Leskovec and Anand Rajaraman
Stanford University

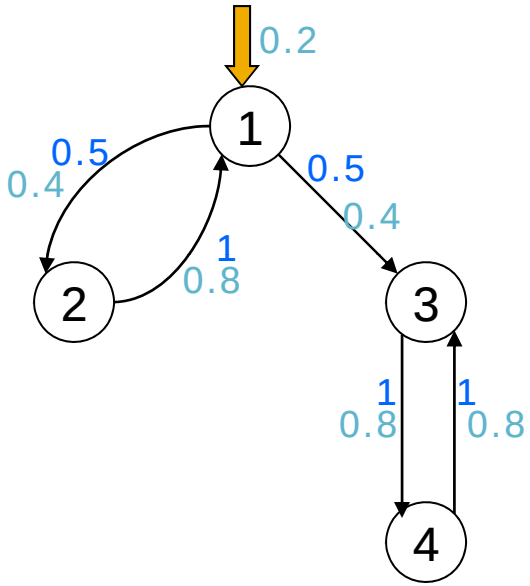
Topic-Specific PageRank

- 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 r_S

Matrix formulation

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

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.

How well does TSPR work?

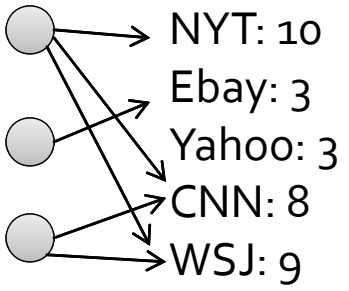
- Experimental results [Haveliwala 2000]
- Picked 16 topics
  - Teleport sets determined using DMOZ
  - E.g., arts, business, sports,…
- "Blind study" using volunteers
  - 35 test queries
  - Results ranked using PageRank and TSPR of most closely related topic
    - E.g., bicycling using Sports ranking
  - In most cases volunteers preferred TSPR ranking

Which topic ranking to use?

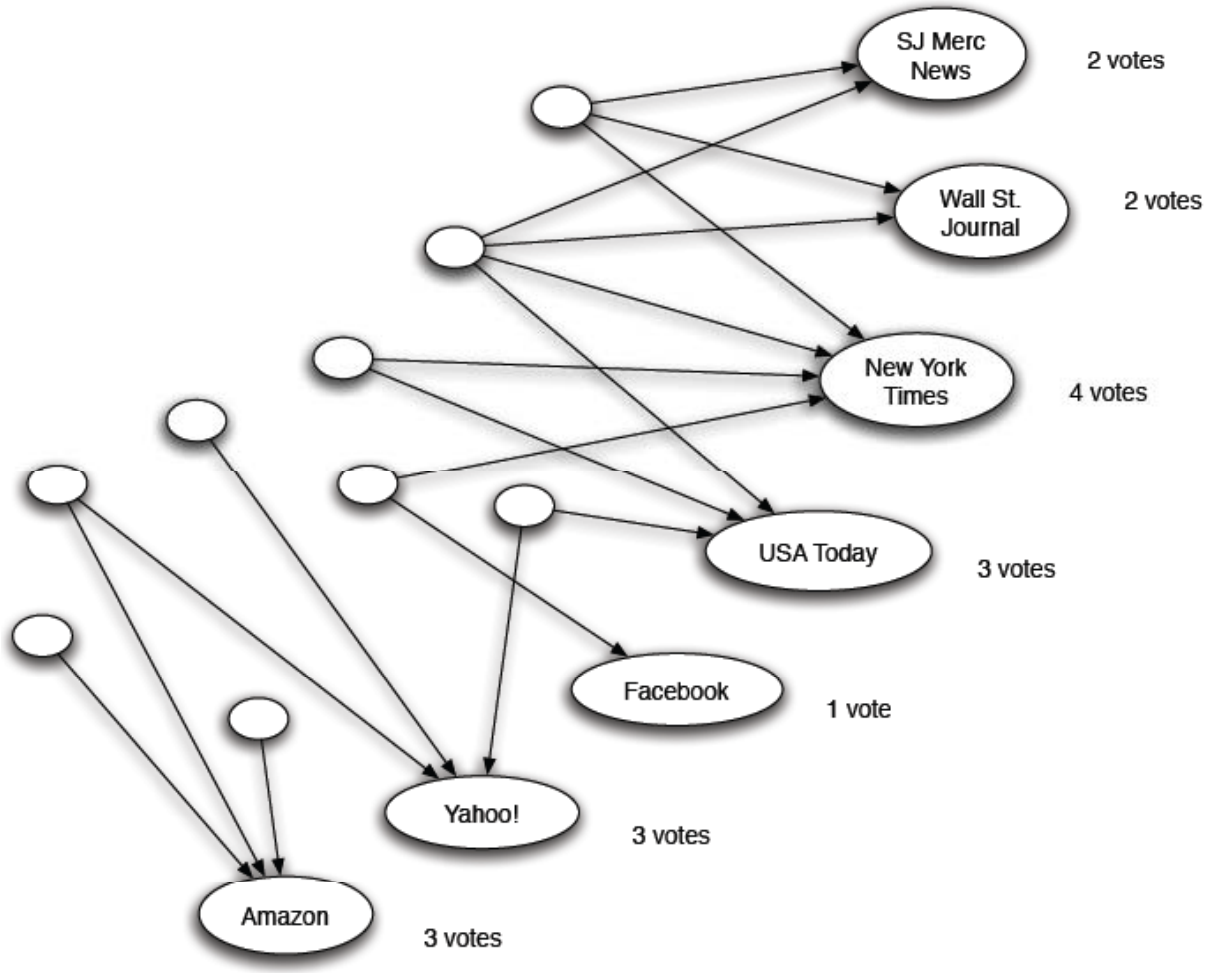
- User can pick from a menu
- Use Naïve Bayes to classify query into a topic
- Can use the context of the query
  - E.g., query is launched from a web page talking about a known topic
  - History of queries e.g., "basketball" followed by "Jordan"
- User context e.g., user's My Yahoo settings, bookmarks, …

Finding newspapers

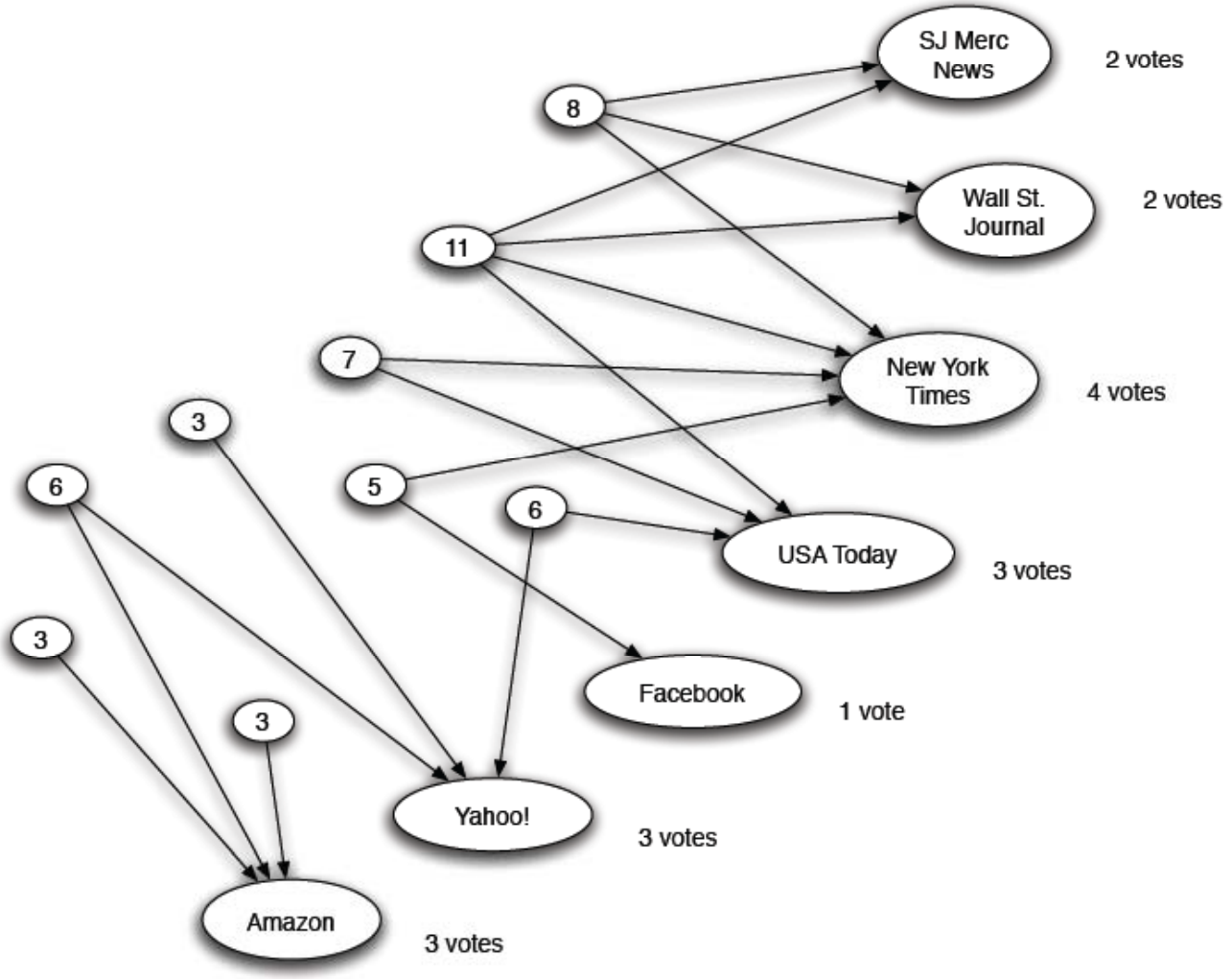
- Goal:
  - Don't just find newspapers but also find "experts" – people who link in a coordinated way to many good newspapers
- Idea: link voting
  - Quality as an expert (hub):
    - Total sum of votes of pages pointed to
  - Quality as an content (authority):
    - Total sum of votes of experts
  - Principle of repeated improvement



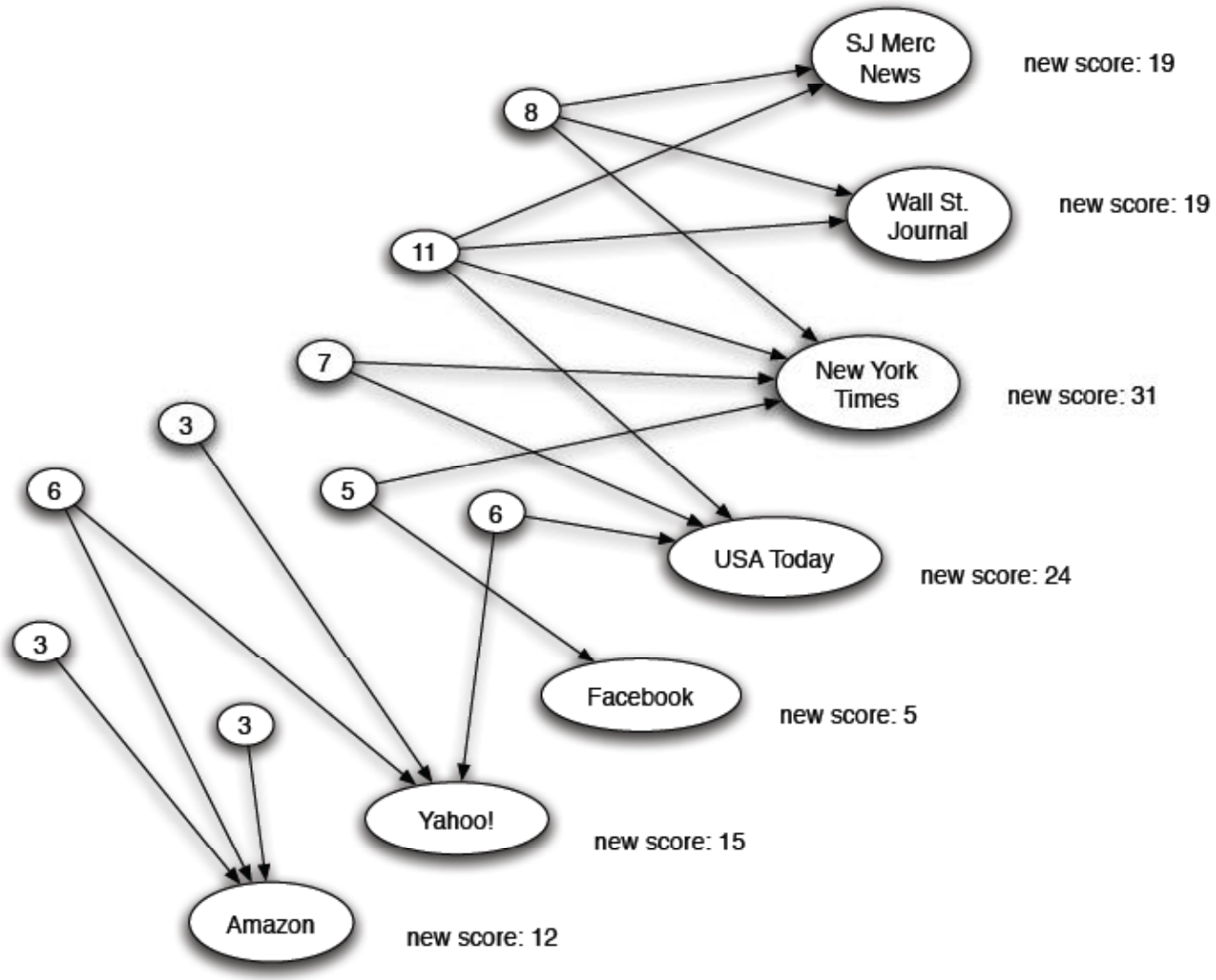
Counting in-links: Authority



Expert quality: Hub



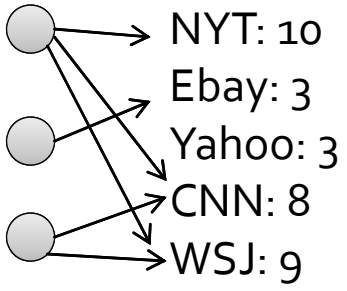
Reweighting



HITS Model

Interesting documents 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



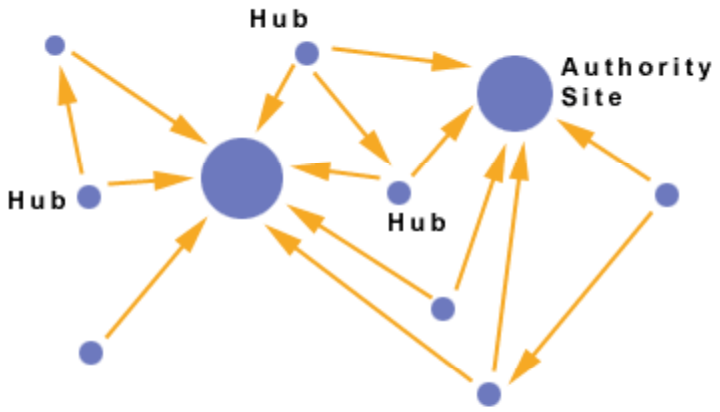
Mutually recursive definition

- 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*

Hubs and Authorities

- Each page *i* has 2 kinds of scores:
  - Hub score: h_i
  - Authority score: a_i
- Algorithm:
  - Initialize: $a_i=h_i=1$
  - Then keep iterating:
    - Authority: $a_j = \sum_{i \to j} h_i$
    - Hub: $h_i = \sum_{i \to j} a_j$
    - Normalize:

$$\sum a_i=1, \sum h_i=1$$



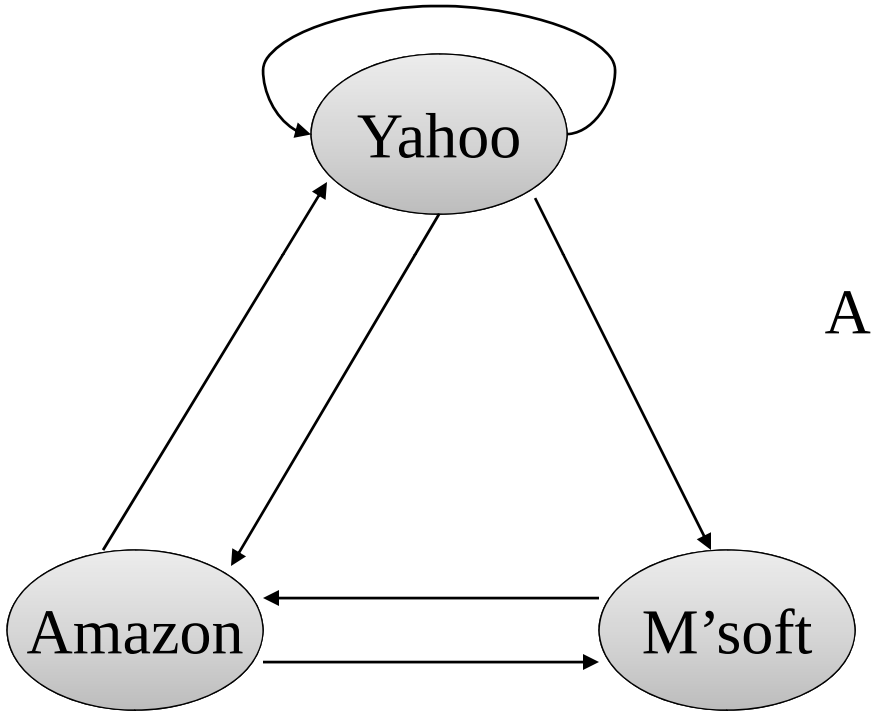
Transition Matrix *A*

- HITS uses adjacency matrix

$$A[i, j] = \begin{cases} 1 & \text{if page } i \text{ links to page } j, \\ 0 & \text{else} \end{cases}$$

- A^T, the transpose of A, is similar to the PageRank matrix M but A^T has 1's where M has fractions

Example



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

Hubs and Authorities

- Notation:
  - Vector a=(a_1…,a_n), h=(h_1…,h_n)
  - Adjacency matrix (n x n): A_{ij}=1 if i→j
- Then:

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

- So: $h = Aa$
- Likewise:

$$a = A^T h$$

Hub and Authority Equations

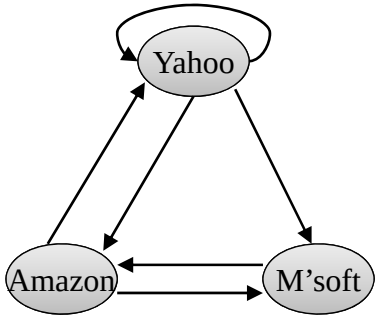
- The hub score of page i is proportional to the sum of the authority scores of the pages it links to: $h = \lambda A a$
  - Constant λ is a scale factor, $\lambda = 1/\sum h_i$
- The authority score of page i is proportional to the sum of the hub scores of the pages it is linked from: $a = \mu A^T h$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

Iterative algorithm

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

Example

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



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

Hubs and Authorities

- Algorithm:
  - Set: $a = h = 1^n$
  - Repeat:
    - $h=Ma$, $a=M^Th$
    - Normalize
- Then: $a=M^T(Ma)$

  new h

  new a
- Thus, in 2k steps:

  $a=(M^TM)^ka$

  $h=(MM^T)^kh$

a is being updated (in 2 steps):

$M^T(Ma)=(M^TM)a$

h is updated (in 2 steps):

$M\,(M^Th)=(MM^T)h$

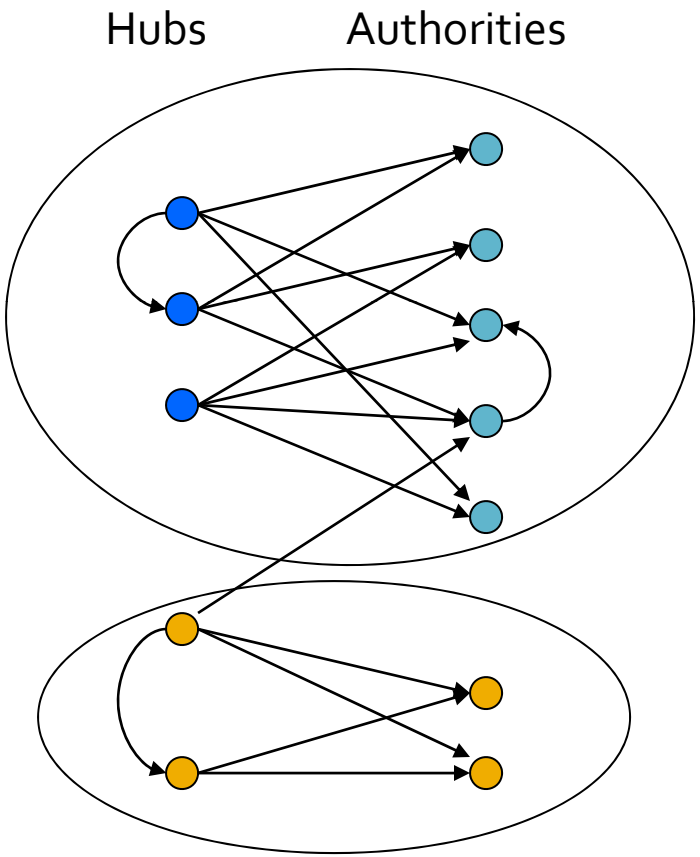
Repeated matrix powering

Existence and Uniqueness

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

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

Bipartite cores



Most densely-connected core (primary core)

Less densely-connected core (secondary core)

Secondary cores

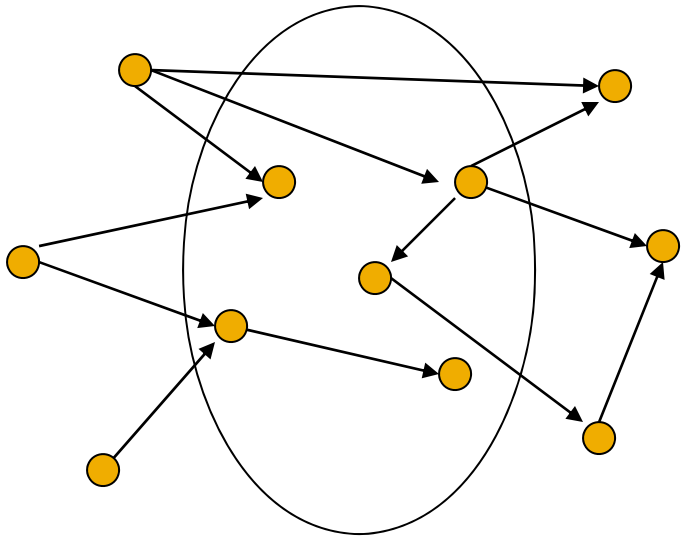
- A single topic can have many bipartite cores
  - Corresponding to different meanings or points of view:
    - abortion: pro-choice, pro-life
    - evolution: darwinian, intelligent design
    - jaguar: auto, Mac, NFL team, *panthera onca*
- How to find such secondary cores?

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:

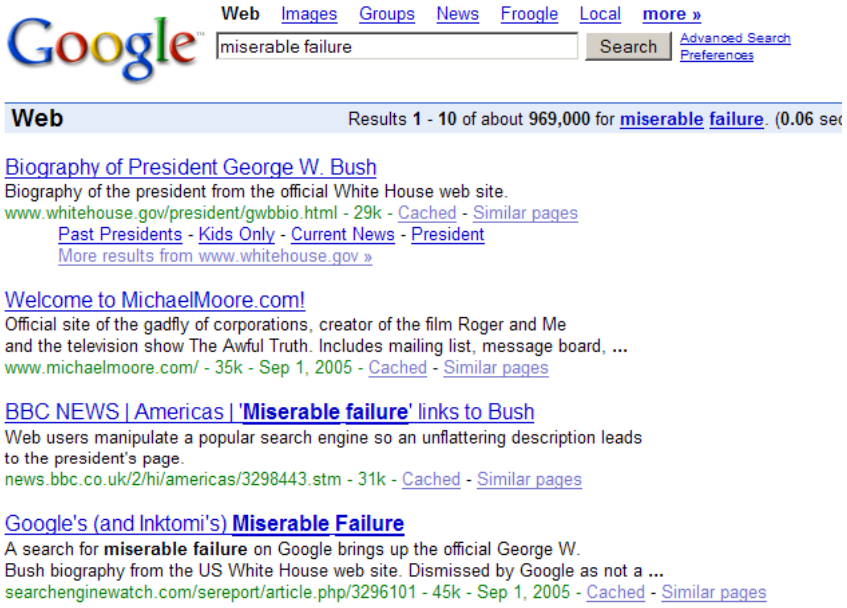


PageRank and HITS

- 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*
- The destinies of PageRank and HITS post-1998 were very different

Web Spam

- Search is the default gateway to the web
- Very high premium to appear on the first page of search results:
  - e-commerce sites
  - advertising-driven sites



What is web spam?

- Spamming:
  - any deliberate action to boost a web page's position in search engine results,
  - incommensurate with page's real value
- Spam:
  - web pages that are the result of spamming
- This is a very broad definition
  - SEO industry might disagree!
  - SEO = search engine optimization
- Approximately 10-15% of web pages are spam

Web Spam Taxonomy

- The treatment by Gyongyi & Garcia-Molina:
- Boosting techniques
  - Techniques for achieving high relevance/importance for a web page
- Hiding techniques
  - Techniques to hide the use of boosting
    - From humans and web crawlers

Boosting techniques

- Term spamming
  - Manipulating the text of web pages in order to appear relevant to queries
- Link spamming
  - Creating link structures that boost PageRank or hubs and authorities scores

Term Spamming

- Repetition:
  - of one or a few specific terms e.g., free, cheap, viagra
  - Goal is to subvert TF-IDF ranking schemes
- Dumping:
  - of a large number of unrelated terms
  - e.g., copy entire dictionaries
- Weaving:
  - Copy legitimate pages and insert spam terms at random positions
- Phrase Stitching:
  - Glue together sentences and phrases from different sources

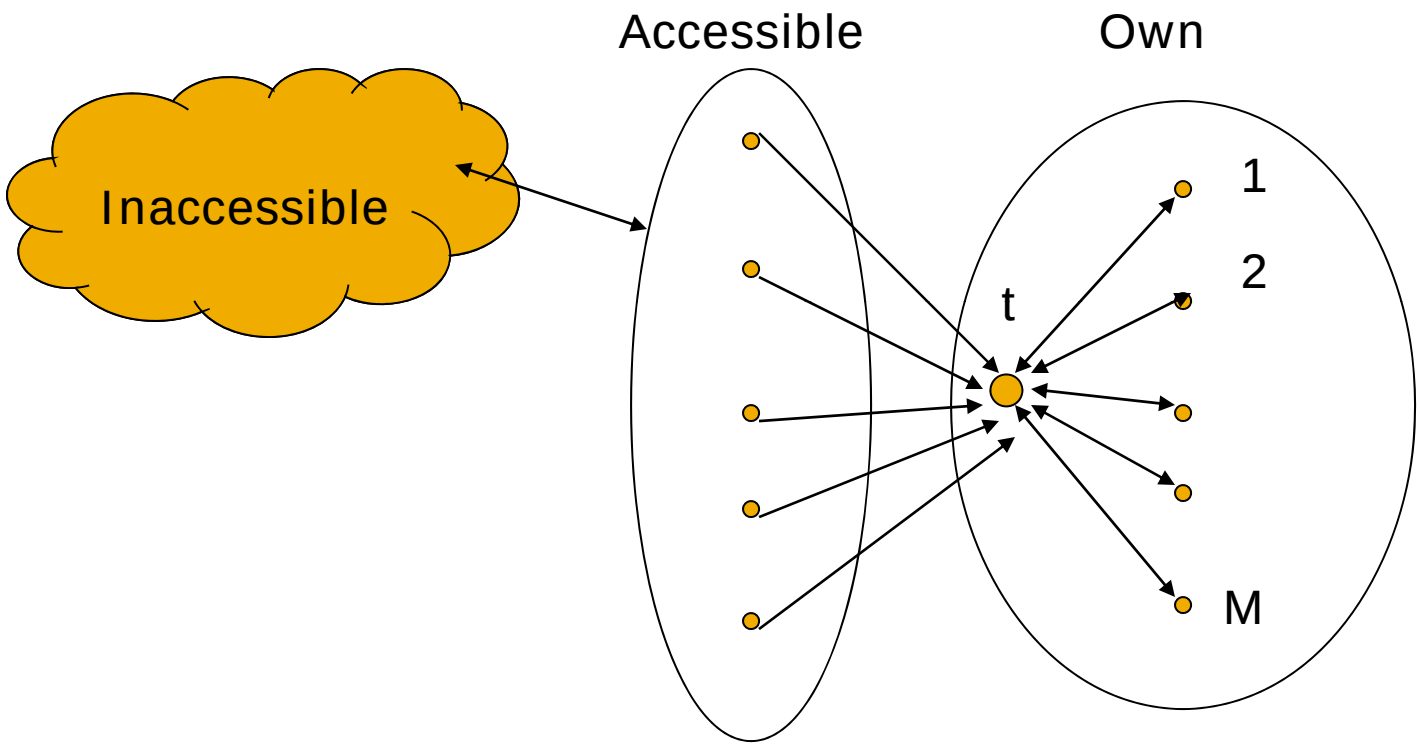
Link spam

- Three kinds of web pages from a spammer's point of view:
  - Inaccessible pages
  - Accessible pages:
    - e.g., blog comments pages
    - spammer can post links to his pages
  - Own pages:
    - Completely controlled by spammer
    - May span multiple domain names

Link Farms

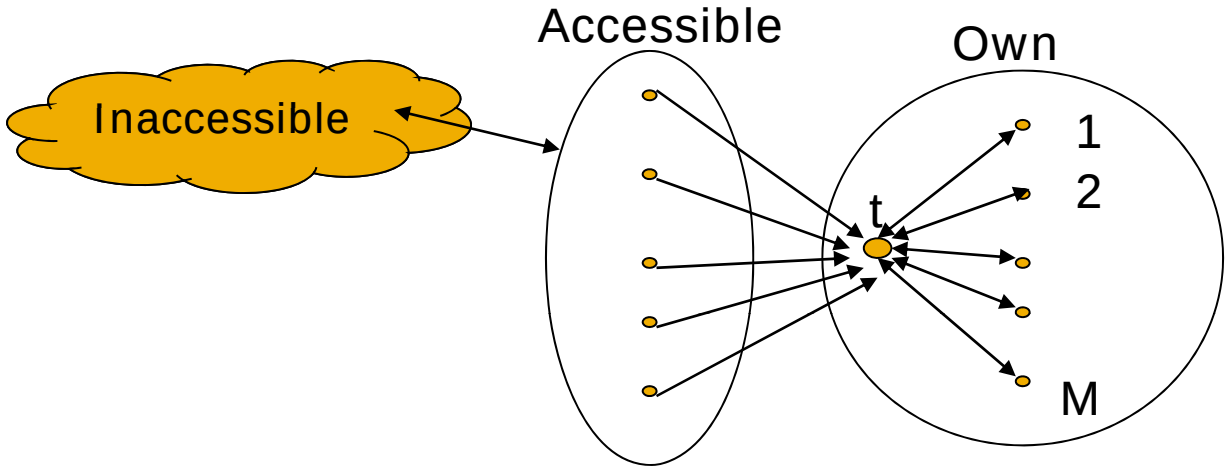
- Spammer's goal:
  - Maximize the PageRank of target page t
- Technique:
  - Get as many links from accessible pages as possible to target page t
  - Construct "link farm" to get PageRank multiplier effect

Link Farms



One of the most common and effective organizations for a link farm

Analysis



N…# pages on the web
M…# of pages
spammer owns

Suppose rank contributed by accessible pages = x
Let PageRank of target page = y
Rank of each "farm" page = $\beta y/M + (1-\beta)/N$

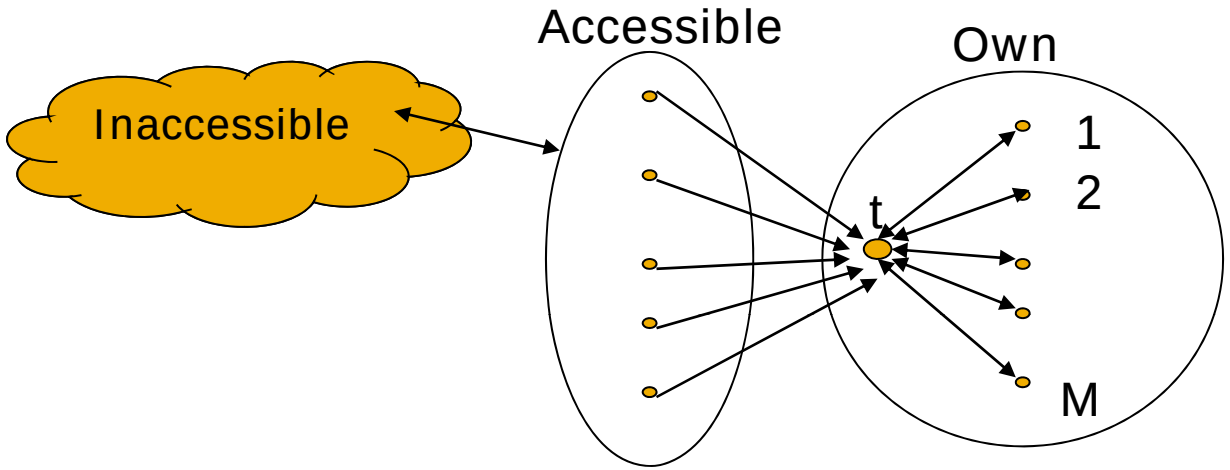
$y = x + \beta M[\beta y/M + (1-\beta)/N] + (1-\beta)/N$

$= x + \beta^2 y + \beta(1-\beta)M/N + \boxed{(1-\beta)/N}$ Very small; ignore

$y = x/(1-\beta^2) + cM/N$

where $c = \beta/(1+\beta)$

Analysis



N…# pages on the web
M…# of pages spammer owns

- $y = x/(1-\beta^2) + cM/N$
  - where $c = \beta/(1+\beta)$
- For $\beta = 0.85$, $1/(1-\beta^2) = 3.6$

- Multiplier effect for "acquired" PageRank
- By making M large, we can make y as large as we want

Detecting Spam

- Term spamming:
  - Analyze text using statistical methods:
    - E.g., Naïve Bayes, Logistic regression
  - Similar to email spam filtering
  - Also useful: detecting approximate duplicate pages
- Link spamming:
  - Open research area
  - One approach: TrustRank

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
  - Must make seed set as small as possible

Trust Propagation

- Call the subset of seed pages that are identified as "good" the "trusted pages"
- Set trust of each trusted page to 1
- 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

Rules for trust propagation

- Trust attenuation:
  - The degree of trust conferred by a trusted page decreases with distance
- Trust splitting:
  - The larger the number of out-links from a page, the less scrutiny the page author gives each out-link
  - Trust is "split" across out-links

Simple model

- 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

Picking the seed set

- Two conflicting considerations:
  - Human has to inspect each seed page, so seed set must be as small as possible
  - Must ensure every "good page" gets adequate trust rank, so need make all good pages reachable from seed set by short paths

Approaches to picking seed set

- Suppose we want to pick a seed set of *k* pages
- PageRank:
  - Pick the top *k* pages by PageRank
  - Assume high PageRank pages are close to other highly ranked pages
  - We care more about high PageRank "good" pages

Inverse PageRank

- Pick the pages with the maximum number of outlinks
- Can make it recursive:
  - Pick pages that link to pages with many out-links
- Formalize as "inverse PageRank"
  - Construct graph G' by reversing edges in G
  - PageRank in G' is inverse page rank in G
- Pick top *k* pages by inverse PageRank

Spam Mass

- In the TrustRank model, we start with good pages and propagate trust
- Complementary view:
  What fraction of a page's PageRank comes from "spam" pages?
- In practice, we don't know all the spam pages, so we need to estimate

Spam mass estimation

- r(p) = PageRank of page p
- $r^+(p)$ = page rank of p with teleport into "good" pages only
- Then:

  $r^-(p) = r(p) - r^+(p)$

- Spam mass of p = $r^-(p)/r(p)$

Good pages

- For spam mass, we need a large set of "good" pages:
  - Need not be as careful about quality of individual pages as with TrustRank
- One reasonable approach
  - .edu sites
  - .gov sites
  - .mil sites

Another approach

- Backflow from known spam pages:
  - Course project from last year's edition of this course
- Still an open area of research…

Reminders:

- Project write-up is due Mon, Feb 1 midnight
  - What is the problem you are solving?
  - What data will you use (where will you get it)?
  - How will you do it?
  - What algorithms/techniques will you use?
  - Who will you evaluate, measure success?
  - What do you expect to submit at the end of the quarter?
- Homework is due on Tue, Feb 2 midnight