CS780
Data Mining for Multimedia Data

Web Search and Mining

Dr. Jessica Lin

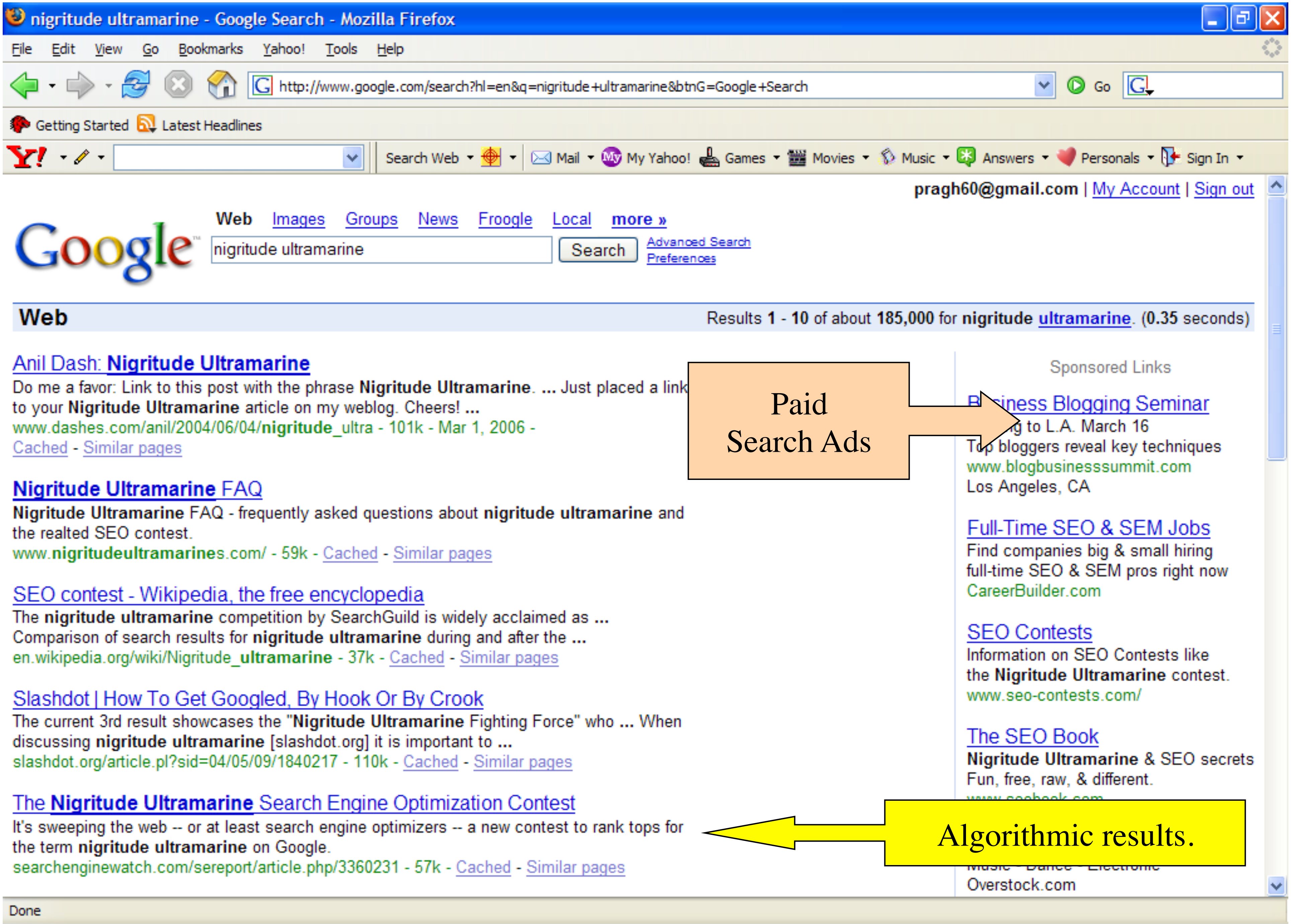
The slides are from Christopher Manning and Prabhakar Raghavan from Stanford University

Brief (non-technical) history

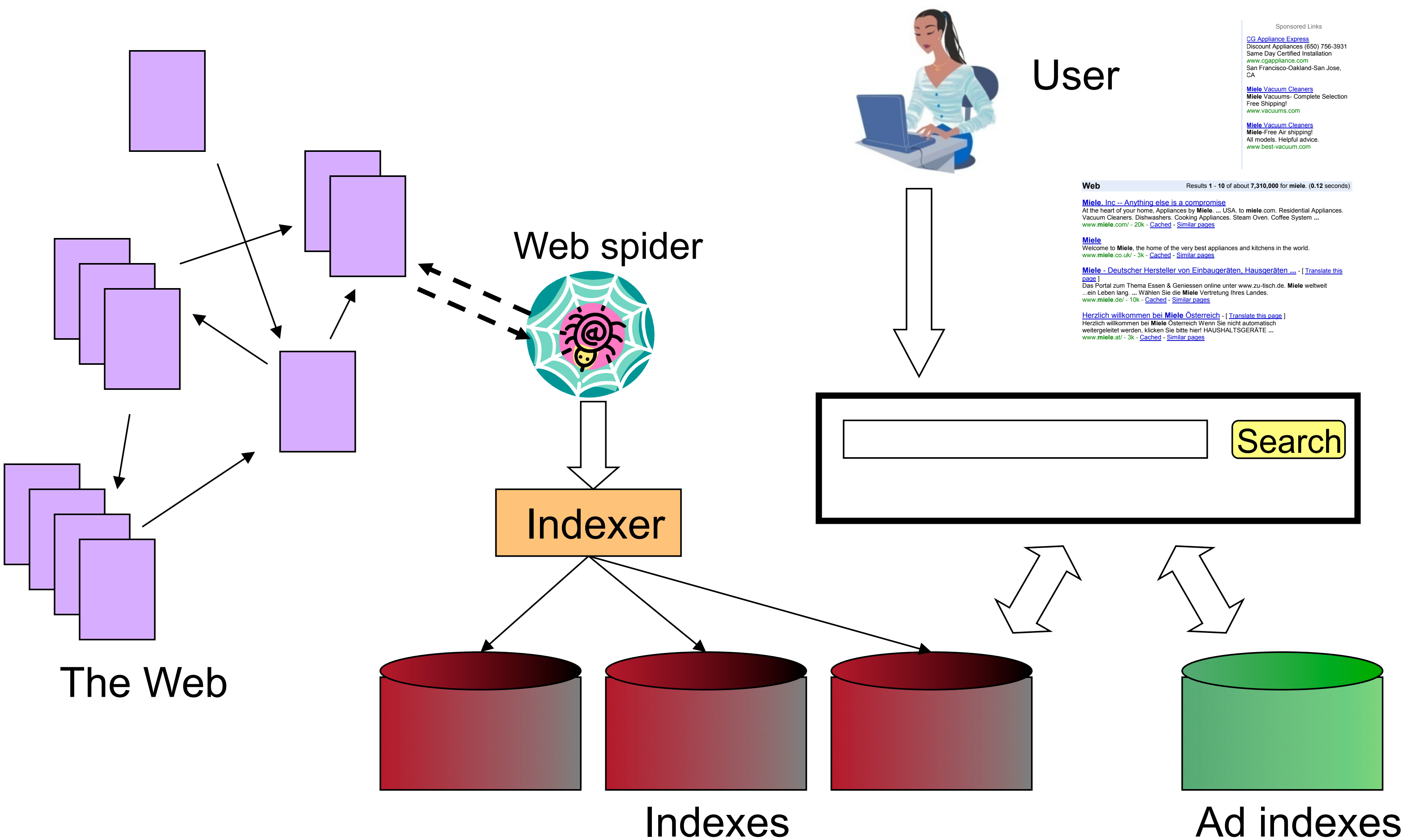
- Early keyword-based engines ca. 1995-1997
  - Altavista, Excite, Infoseek, Inktomi, Lycos
- <u>Paid search</u> ranking: Goto (morphed into Overture.com → Yahoo!)
  - Your search ranking depended on how much you paid
  - Auction for keywords: <u>***casino***</u> was expensive!

Brief (non-technical) history

- 1998+: Link-based ranking pioneered by Google
  - ★ Blew away all early engines
  - ★ Meanwhile Goto/Overture's annual revenues were nearing $1 billion
- Result: Google added paid search "ads" to the side, independent of search results
  - ★ Yahoo followed suit, acquiring Overture (for paid placement) and Inktomi (for search)
- 2005+: Google gains search share, dominating in Europe and very strong in North America
  - ★ 2009: Yahoo! and Microsoft propose combined paid search offering


nigritude ultramarine - Google Search - Mozilla Firefox
File Edit View Go Bookmarks Yahoo! Tools Help
http://www.google.com/search?hl=en&q=nigritude+ultramarine&btnG=Google+Search
Go
Getting Started Latest Headlines
Search Web Mail My Yahoo! Games Movies Music Answers Personals Sign In
pragh60@gmail.com | My Account | Sign out
Google
Web Images Groups News Froogle Local more »
nigritude ultramarine
Search
Advanced Search
Preferences
Web
Results 1 - 10 of about 185,000 for nigritude ultramarine. (0.35 seconds)
Anil Dash: Nigritude Ultramarine
Do me a favor: Link to this post with the phrase Nigritude Ultramarine. ... Just placed a link to your Nigritude Ultramarine article on my weblog. Cheers! ...
www.dashes.com/anil/2004/06/04/nigritude_ultra - 101k - Mar 1, 2006 - Cached - Similar pages
Nigritude Ultramarine FAQ
Nigritude Ultramarine FAQ - frequently asked questions about nigritude ultramarine and the realted SEO contest.
www.nigritudeultramarines.com/ - 59k - Cached - Similar pages
SEO contest - Wikipedia, the free encyclopedia
The nigritude ultramarine competition by SearchGuild is widely acclaimed as ... Comparison of search results for nigritude ultramarine during and after the ...
en.wikipedia.org/wiki/Nigritude_ultramarine - 37k - Cached - Similar pages
Slashdot | How To Get Googled, By Hook Or By Crook
The current 3rd result showcases the "Nigritude Ultramarine Fighting Force" who ... When discussing nigritude ultramarine [slashdot.org] it is important to ...
slashdot.org/article.pl?sid=04/05/09/1840217 - 110k - Cached - Similar pages
The Nigritude Ultramarine Search Engine Optimization Contest
It's sweeping the web -- or at least search engine optimizers -- a new contest to rank tops for the term nigritude ultramarine on Google.
searchenginewatch.com/sereport/article.php/3360231 - 57k - Cached - Similar pages
Sponsored Links
Top bloggers reveal key techniques
www.blogbusinesssummit.com
Los Angeles, CA
Full-Time SEO & SEM Jobs
Find companies big & small hiring full-time SEO & SEM pros right now
CareerBuilder.com
SEO Contests
Information on SEO Contests like the Nigritude Ultramarine contest.
www.seo-contests.com/
The SEO Book
Nigritude Ultramarine & SEO secrets Fun, free, raw, & different.
Overstock.com
Done
Paid Search Ads
Algorithmic results.

Web search basics



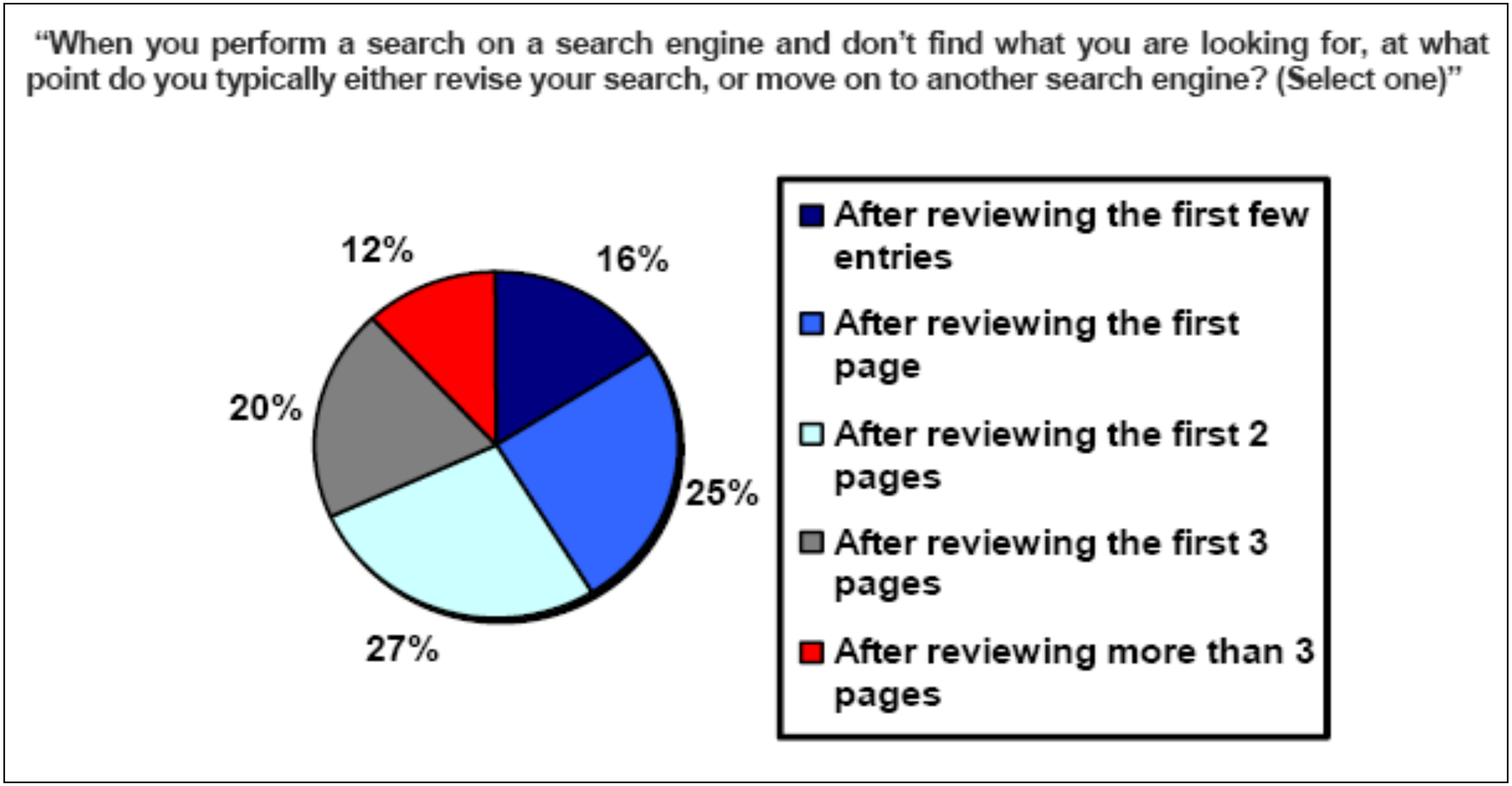
User Needs

- User needs:
  - ★ **Informational** – want to learn about something (~40% / 65%)

    `High blood pressure`
  - ★ **Navigational** – want to go to that page (~25% / 15%)

    `United Airlines`
  - ★ **Transactional** – want to do something (web-mediated) (~35% / 20%)
    - ✓ Access a service — `Fairfax weather`
    - ✓ Downloads — `Mars surface images`
    - ✓ Shop — `iPhone 5s`
  - ★ **Gray areas**
    - ✓ Find a good hub — `Car rental Brasil`
    - ✓ Exploratory search "see what's there"

How far do people look for results?



(Source: iprospect.com WhitePaper_2006_SearchEngineUserBehavior.pdf)

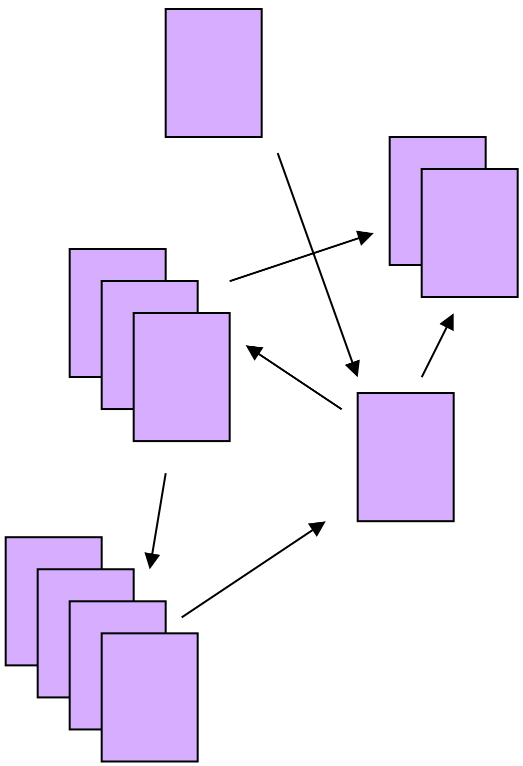
Users' empirical evaluation of results

- Quality of pages varies widely
  - Relevance is not enough
  - Other desirable qualities (non IR!!)
    - Content: Trustworthy, diverse, non-duplicated, well maintained
    - Web readability: display correctly & fast
    - No annoyances: pop-ups, etc
- Precision vs. recall
  - On the web, recall seldom matters
- What matters
  - Precision at 1? Precision above the fold?
  - Comprehensiveness – must be able to deal with obscure queries
    - Recall matters when the number of matches is very small
- User perceptions may be unscientific, but are significant over a large aggregate

Users' empirical evaluation of engines

- Relevance and validity of results
- UI – Simple, no clutter, error tolerant
- Trust – Results are objective
- Coverage of topics for polysemic queries
- Pre/Post process tools provided
  - Mitigate user errors (auto spell check, search assist,…)
  - Explicit: Search within results, more like this, refine ...
  - Anticipative: related searches
- Deal with idiosyncrasies
  - Web specific vocabulary
    - Impact on stemming, spell-check, etc
  - Web addresses typed in the search box

The Web document collection



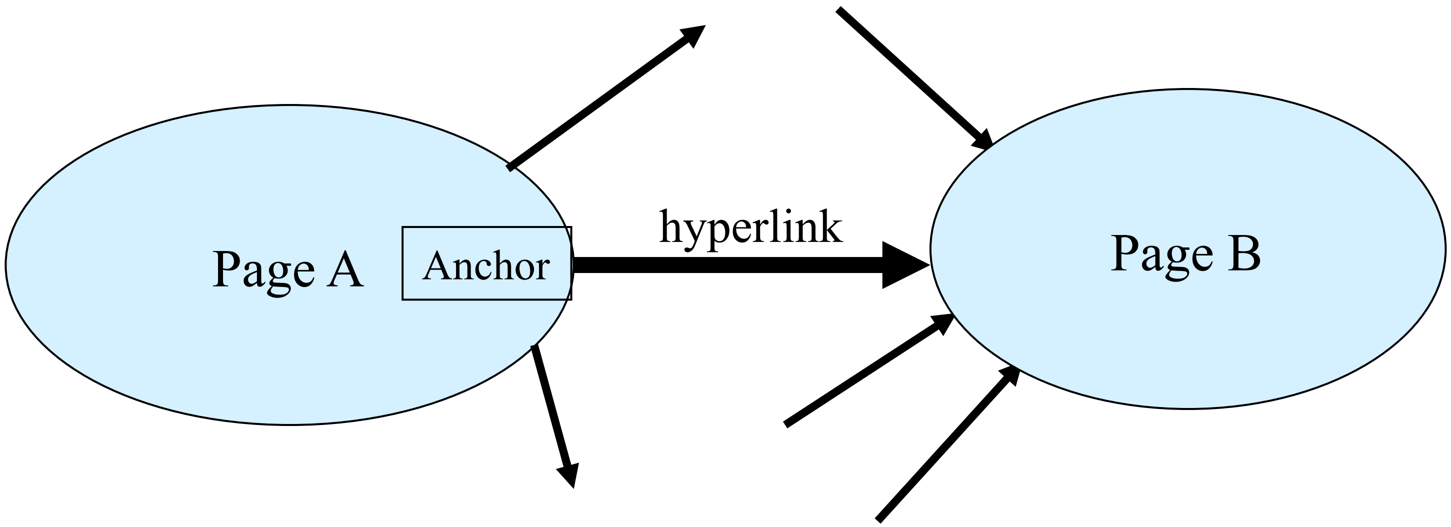
The Web

- No design/coordination
- Distributed content creation, linking, democratization of publishing
- Content includes truth, lies, obsolete information, contradictions …
- Unstructured (text, html, …), semi-structured (XML, annotated photos), structured (Databases)…
- Scale much larger than previous text collections … but corporate records are catching up
- Growth – slowed down from initial "volume doubling every few months" but still expanding
- Content can be *dynamically generated*

Ranking web pages

- Web pages are not equally "important"
  - ★ www.joe-schmoe.com v www.stanford.edu
- Inlinks as votes
  - ★ www.stanford.edu has 23,400 inlinks
  - ★ www.joe-schmoe.com has 1 inlink
- Are all inlinks equal?
  - ★ Recursive question!

The Web as a Directed Graph

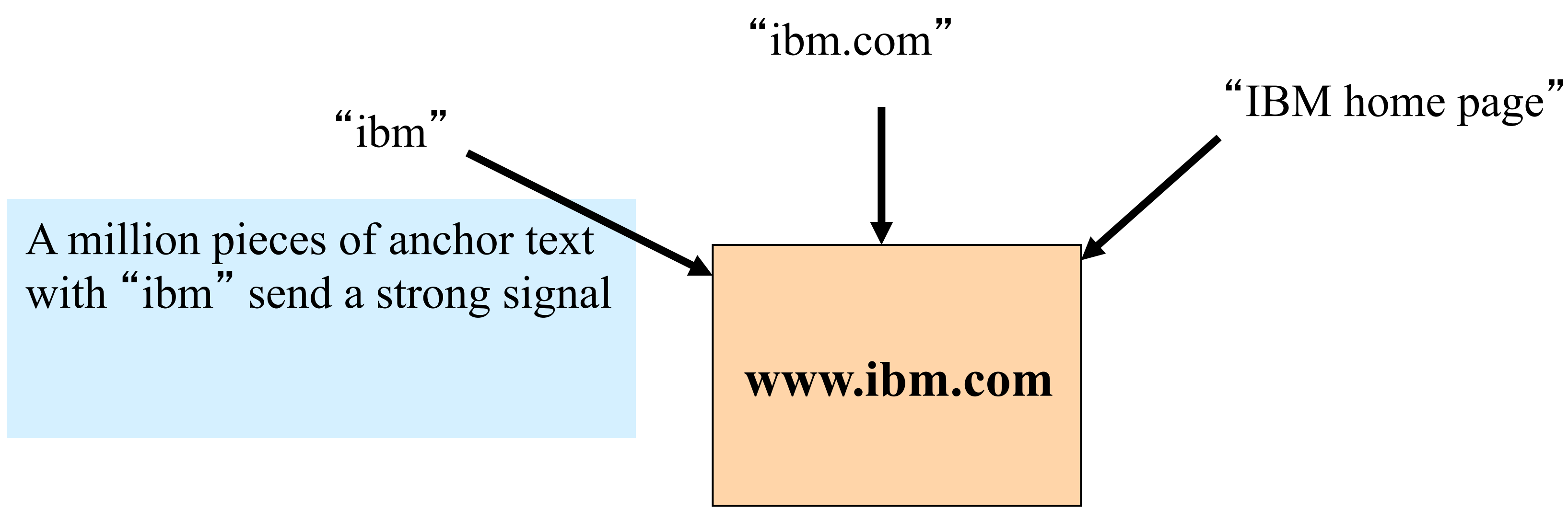


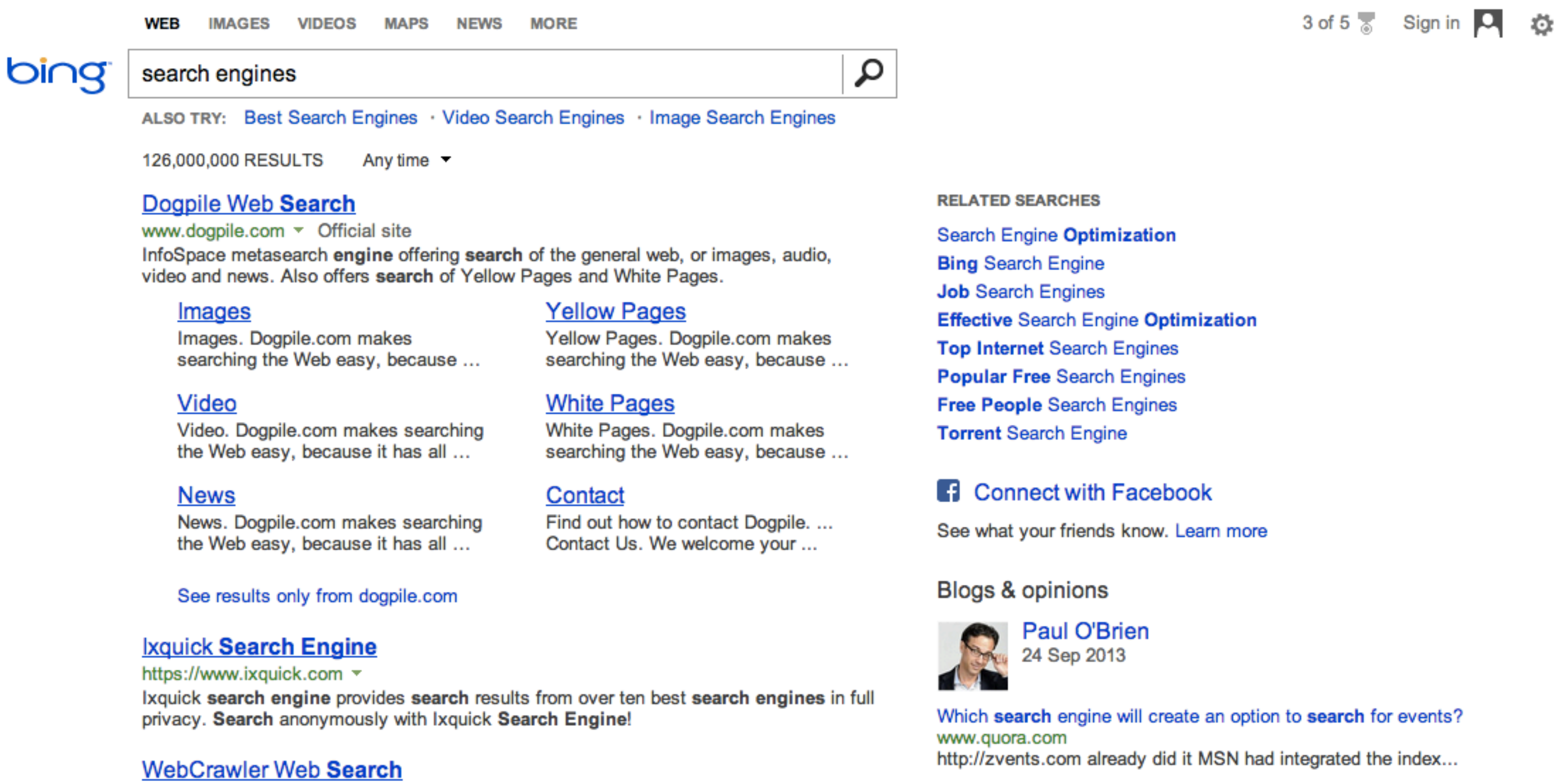
Assumption 1: A hyperlink between pages denotes author perceived relevance (quality signal)

Assumption 2: The text in the anchor of the hyperlink describes the target page (textual context)

Anchor Text
WWW Worm - McBryan [Mcbr94]

- For ***ibm*** how to distinguish between:
  - ★ IBM's home page (mostly graphical)
  - ★ IBM's copyright page (high term freq. for 'ibm')
  - ★ Rival's spam page (arbitrarily high term freq.)

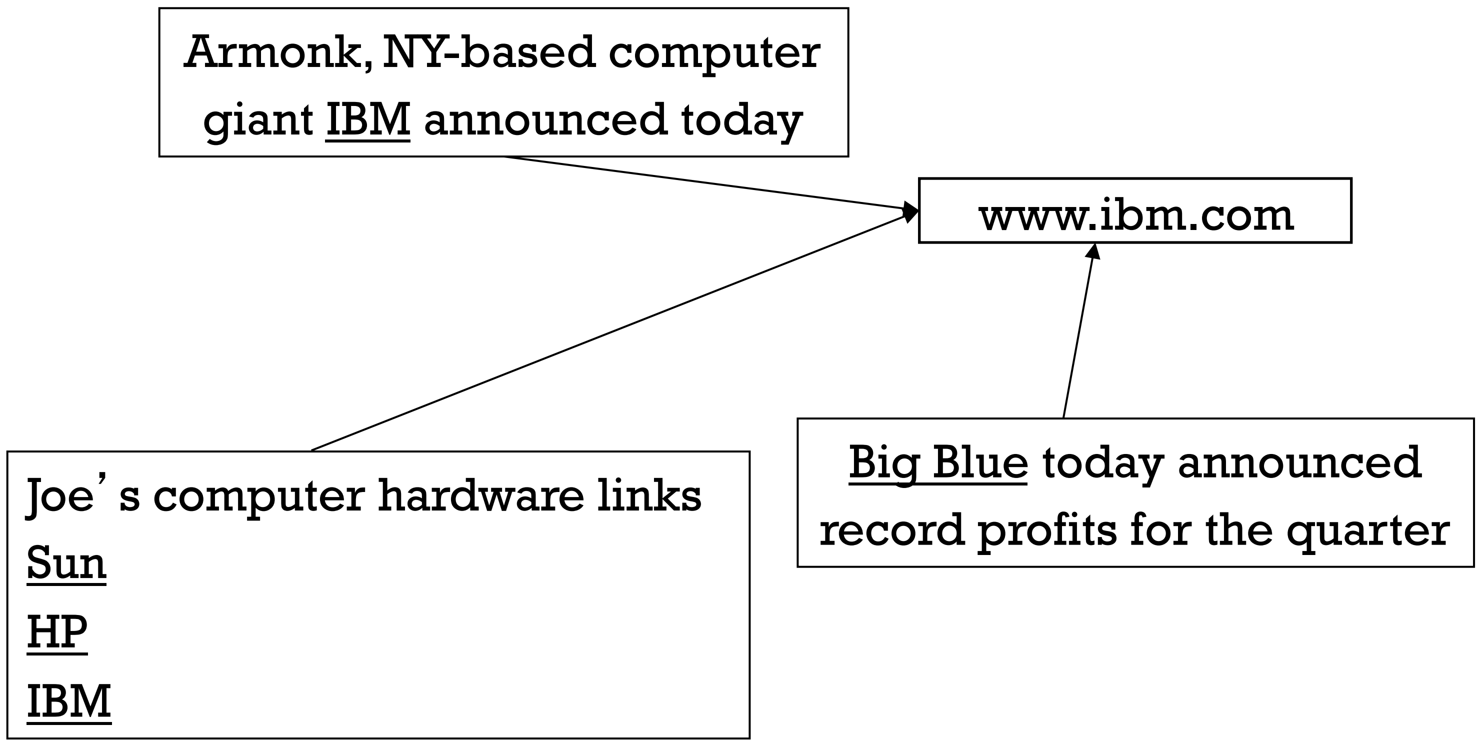




Where is Google? Not on the first page. It appears on the 3rd page!

Indexing anchor text

- When indexing a document *D*, include anchor text from links pointing to *D*.



- Consider a window of text surrounding the anchor text too.

Indexing anchor text

- Can sometimes have unexpected side effects - *e.g.,* ***evil empire***.
- Can score anchor text with weight depending on the authority of the anchor page's website
  - e.g., if we were to assume that content from cnn.com or yahoo.com is authoritative, then trust the anchor text from them

Anchor Text

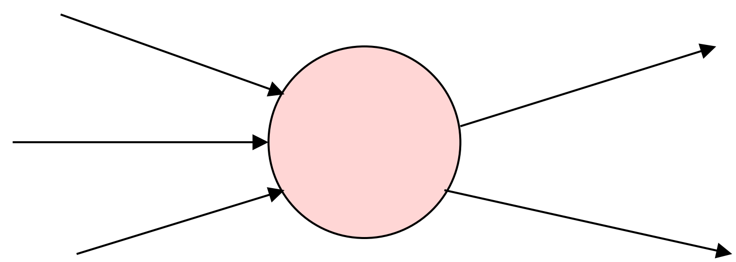
- Other applications
  - Weighting/filtering links in the graph
  - Generating page descriptions from anchor text

Citation Analysis

- Citation frequency
- Co-citation coupling frequency
  - Cocitations with a given author measures “impact”
  - Cocitation analysis
- Bibliographic coupling frequency
  - Articles that co-cite the same articles are related
- Citation indexing
  - Who is this author cited by? (Garfield 1972)

Query-independent ordering

- First generation: using link counts as simple measures of popularity.
- Two basic suggestions:
  - Undirected popularity:
    - Each page gets a score = the number of in-links plus the number of out-links (3+2=5).
  - Directed popularity:
    - Score of a page = number of its in-links (3).



Query processing

- First retrieve all pages meeting the text query (say ***venture capital***).
- Order these by their link popularity (either variant on the previous slide).
- More nuanced – use link counts as a measure of static goodness, combined with text match score

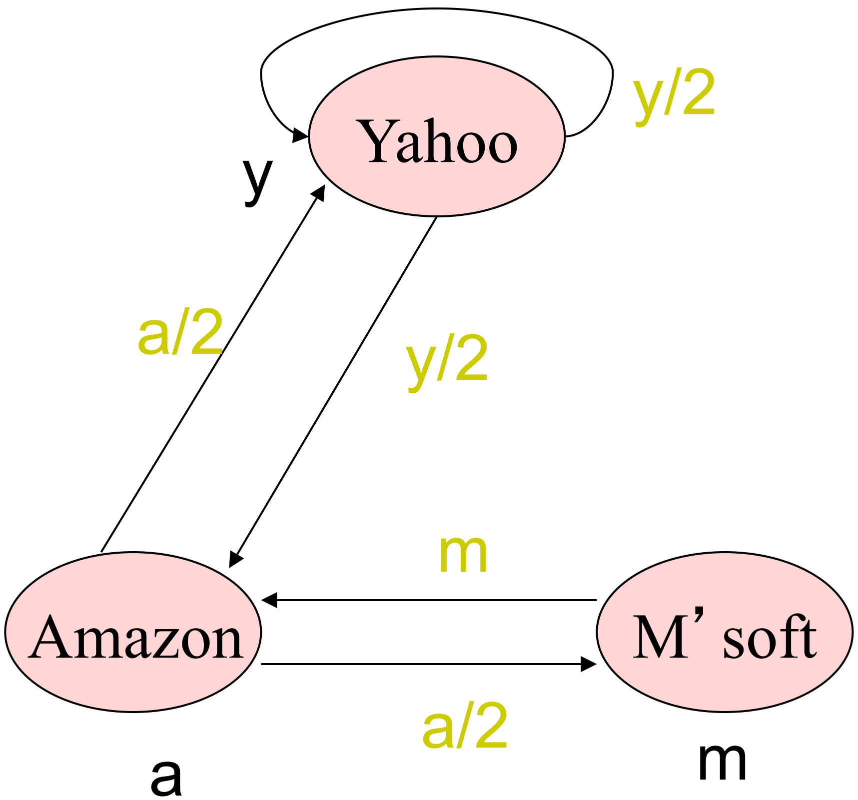
Spamming simple popularity

- *Exercise*: How do you spam each of the following heuristics so your page gets a high score?
  - ★ Each page gets a static score = the number of in-links plus the number of out-links.
  - ★ Static score of a page = number of its in-links.

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

Simple "flow" model



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

Solving the flow equations

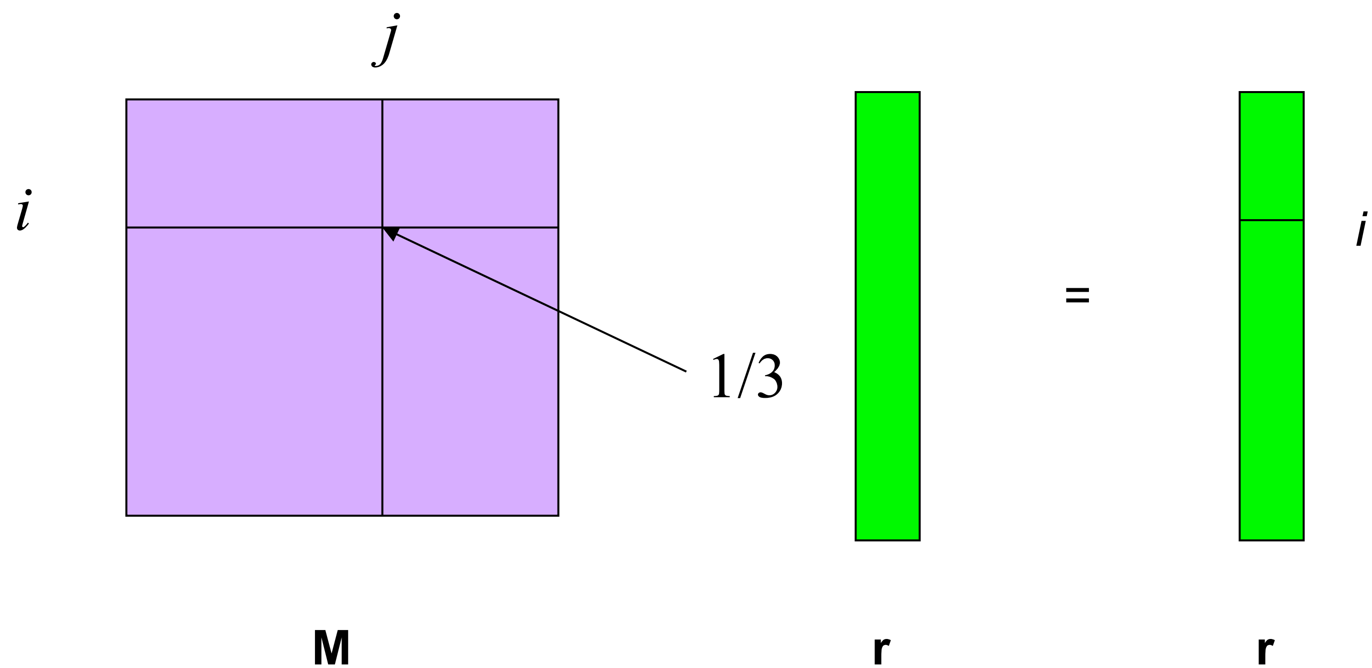
- 3 equations, 3 unknowns, no constants
  - ★ No unique solution
  - ★ All solutions equivalent modulo scale factor
- Additional constraint forces uniqueness
  - ★ $y+a+m = 1$
  - ★ Then $y = 2/5$, $a = 2/5$, $m = 1/5$
- Gaussian elimination method works for small examples, but we need a better method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

Matrix formulation

- Matrix **M** has one row and one column for each web page
- Suppose page j has n outlinks
  - ★ If i is one of j's outlinks, 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

Example

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



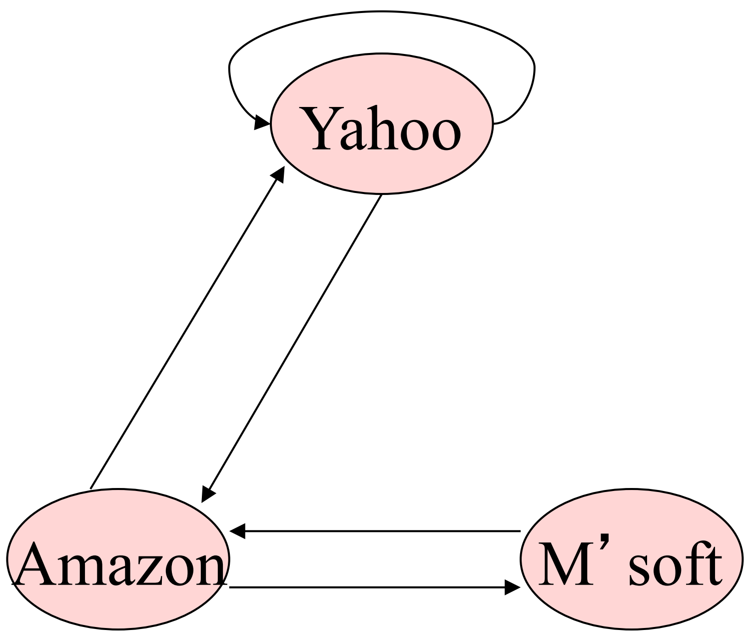
Eigenvector formulation

- 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



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

r = Mr

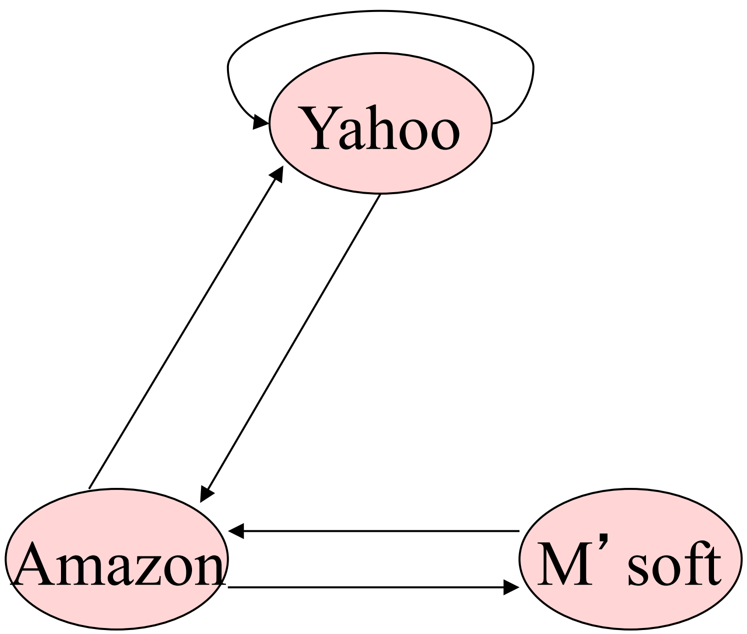
$$y = y/2 + a/2$$
$$a = y/2 + m$$
$$m = a/2$$

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

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 $|\mathbf{r}^{k+1} - \mathbf{r}^k|_1 < \varepsilon$
  - ★ $|\mathbf{x}|_1 = \sum_{1 \cdot i \cdot N}|x_i|$ is the L_1 norm
  - ★ Can use any other vector norm e.g., Euclidean

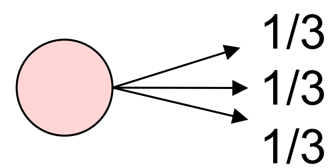
Power Iteration Example



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

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

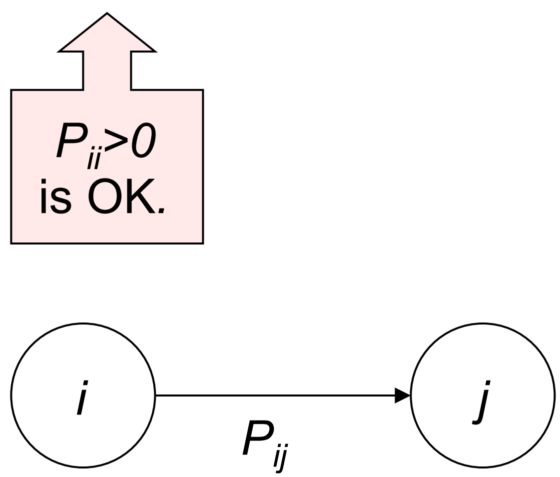
Random Walk Interpretation



- 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 **p**(t) be a vector whose i^{th} component is the probability that the surfer is at page i at time t
  - **p**(t) is a probability distribution on pages
- "In the steady state" each page has a long-term visit rate - use this as the page's score.

Markov chains

- Markov Chains are abstractions of random walk
- A Markov chain consists of n <u>states</u>, plus an $n \times n$ <u>transition probability matrix</u> **P**.
- At each step, we are in exactly one of the states.
- For $1 \leq i,j \leq n$, the matrix entry P_{ij} tells us the probability of j being the next state, given we are currently in state i.



The stationary distribution

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

Existence and Uniqueness

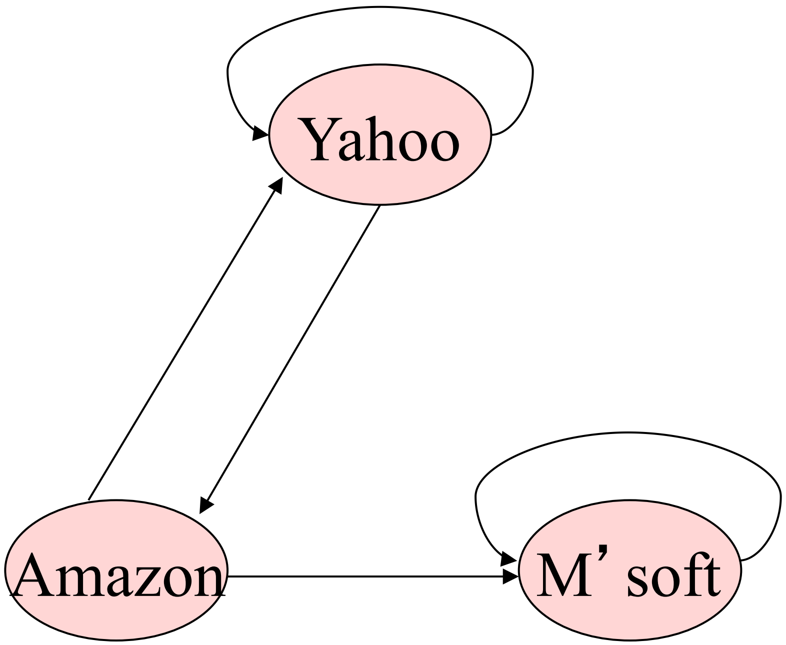
A central result from the theory of random walks (aka Markov processes):

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



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

$$\begin{array}{cc} y \\ a & = \\ m \end{array} \quad \begin{array}{c} 1/3 \\ 1/3 \\ 1/3 \end{array} \quad \begin{array}{c} 1/3 \\ 1/6 \\ 1/2 \end{array} \quad \begin{array}{c} 1/4 \\ 1/6 \\ 7/12 \end{array} \quad \begin{array}{c} 5/24 \\ 1/8 \\ 2/3 \end{array} \quad \begin{array}{c} \\ \ldots \\ \end{array} \quad \begin{array}{c} 0 \\ 0 \\ 3 \end{array}$$

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 $1-\beta$, 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

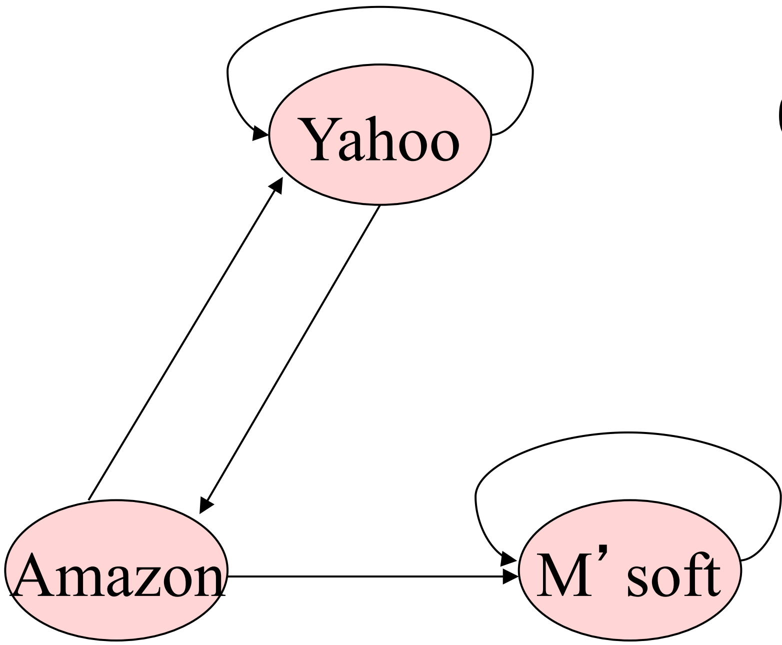
Matrix formulation

- Suppose there are N pages
  - ★ Consider a page j, with set of outlinks Out(j)
  - ★ We have $M_{ij} = 1/|Out(j)|$ when i is in Out(j) 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

Previous example with β=0.8



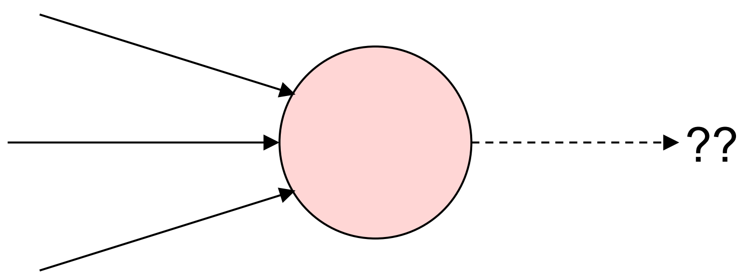
$$0.8 \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

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

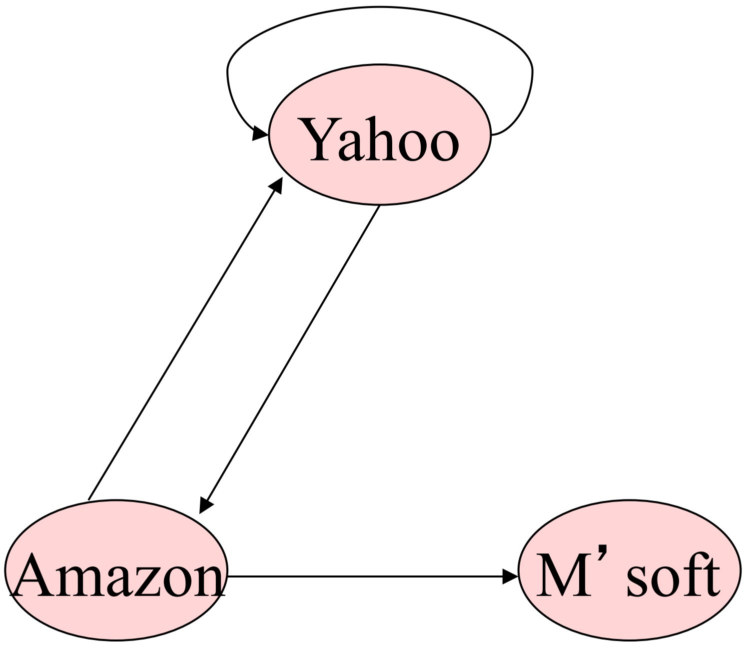
$$\begin{array}{c} y \\ a \\ m \end{array} = \begin{array}{c} 1/3 \\ 1/3 \\ 1/3 \end{array} \quad \begin{array}{c} 1/3 \\ 1/5 \\ 7/15 \end{array} \quad \begin{array}{c} 63/225 \\ 1/5 \\ 129/225 \end{array} \quad \ldots \quad \begin{array}{c} 7/11 \\ 5/11 \\ 21/11 \end{array}$$

Dead ends

- Pages with no outlinks are "dead ends" for the random surfer
  - ★ The web is full of dead-ends.
  - ★ Nowhere to go on next step; random surfer gets stuck



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!

$$\begin{array}{ccccccccc} y & & 1/3 & 0.33 & 0.262 & 0.216 & 0.175 & & 0 \\ a & = & 1/3 & 0.2 & 0.182 & 0.143 & 0.118 & \ldots & 0 \\ m & & 1/3 & 0.2 & 0.129 & 0.111 & 0.089 & & 0 \end{array}$$

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

Pagerank summary

- Preprocessing:
  - Given graph of links, build matrix **P**.
  - From it compute **r**.
  - The entry r_i is a number between 0 and 1: the pagerank of page *i*.
- Query processing:
  - Retrieve pages meeting query.
  - Rank them by their pagerank.
  - Order is query-*independent*.

The reality

- Pagerank is used in google, but is hardly the full story of ranking
  - ★ Many sophisticated features are used
  - ★ Some address specific query classes
  - ★ Machine learned ranking heavily used
- Pagerank still very useful for things like crawl policy

Pagerank: Issues and Variants

- How realistic is the random surfer model?
  - What if we modeled the back button?
  - Surfer behavior sharply skewed towards short paths
  - Search engines, bookmarks & directories make jumps non-random.
- Biased Surfer Models
  - Weight edge traversal probabilities based on match with topic/query (non-uniform edge selection)
  - Bias jumps to pages on topic (e.g., based on personal bookmarks & categories of interest)

Topic Specific Pagerank

- Goal – pagerank values that depend on query *topic*
- Conceptually, we use a random surfer who teleports, with say 10% probability, using the following rule:
  - ✓ Selects a topic (say, one of the 16 top level ODP categories) based on a query & user -specific distribution over the categories
  - ✓ Teleport to a page uniformly at random within the chosen topic
- Sounds hard to implement: can't compute PageRank at query time!

Topic Specific Pagerank

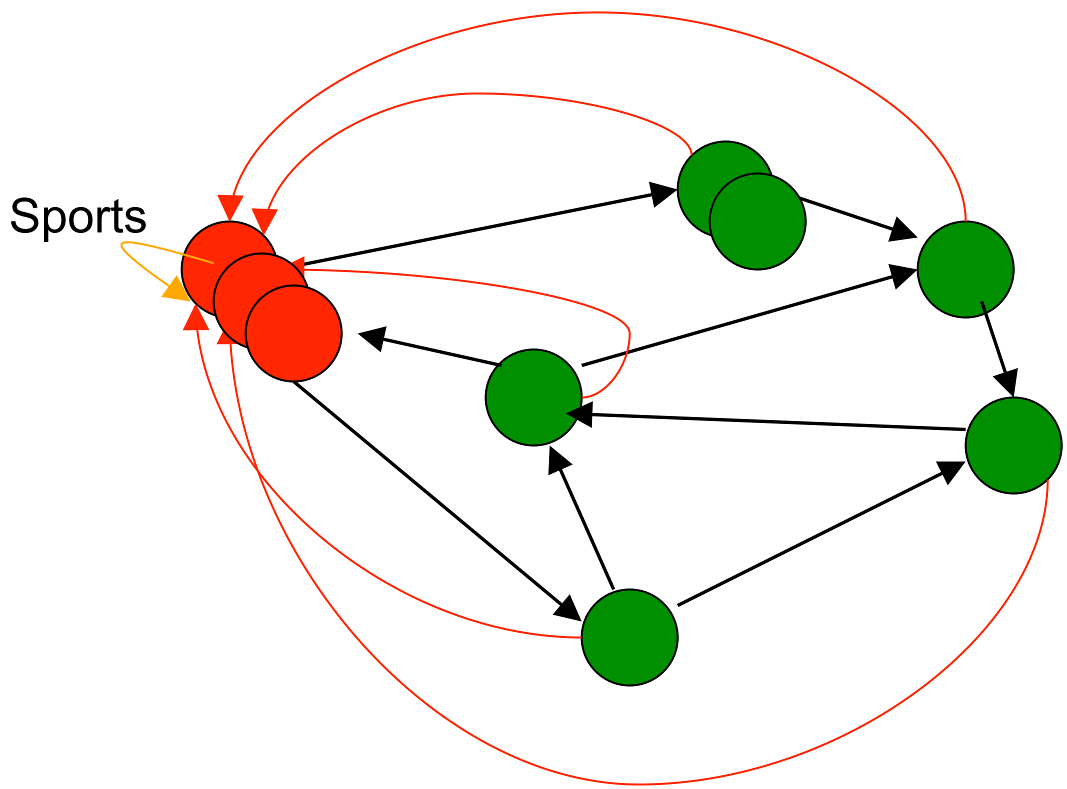
- **Offline**:Compute pagerank for *individual* topics
  - ★ Query independent as before
  - ★ Each page has multiple pagerank scores – one for each ODP category, with teleportation only to that category
- **Online**: Query context classified into (distribution of weights over) topics
  - ★ Generate a dynamic pagerank score for each page – weighted sum of topic-specific pageranks

Influencing PageRank (“Personalization”)

- Input:
  - ★ Web graph W
  - ★ Influence vector **v** over topics
    - **v** : (page → degree of influence)
- Output:
  - ★ Rank vector **r**: (page → page importance wrt **v**)
- **r** = PR(W , **v**)

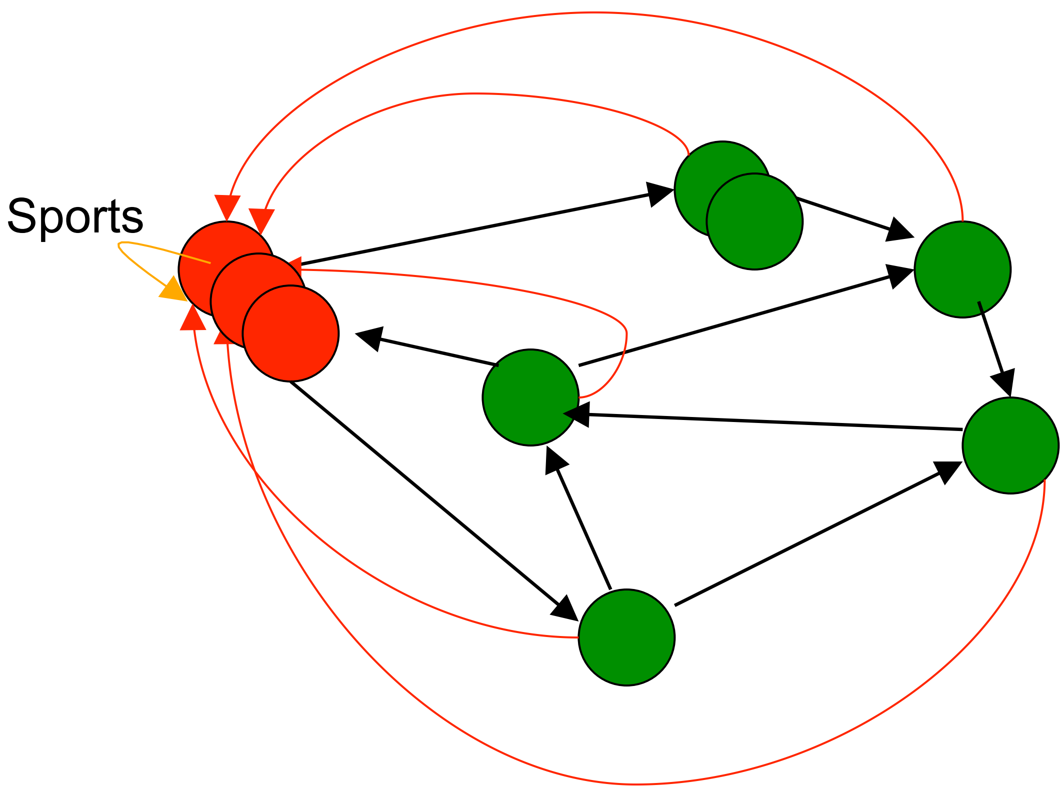
Vector has one component for each topic

Non-uniform Teleportation



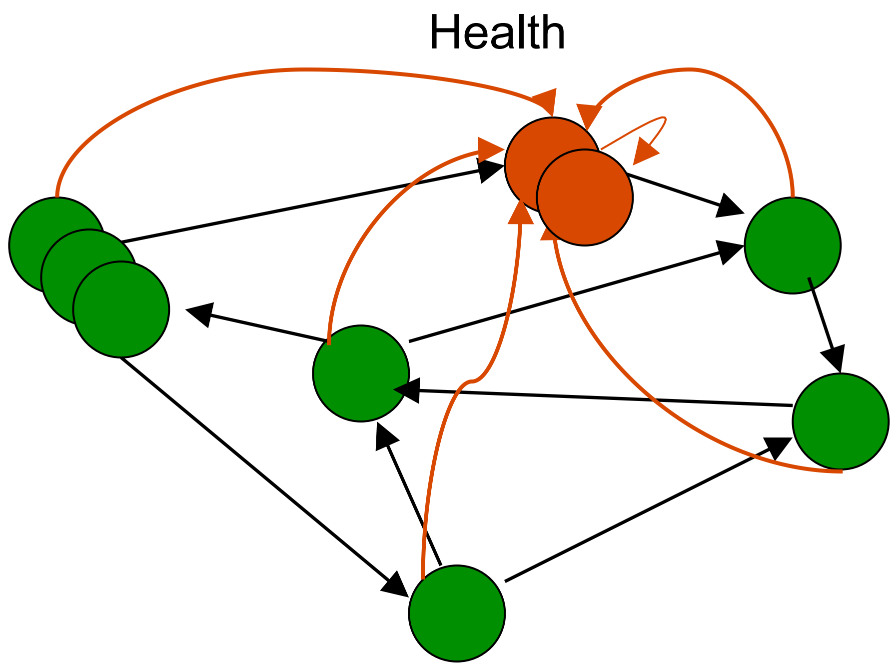
Teleport with 10% probability to a Sports page

Interpretation



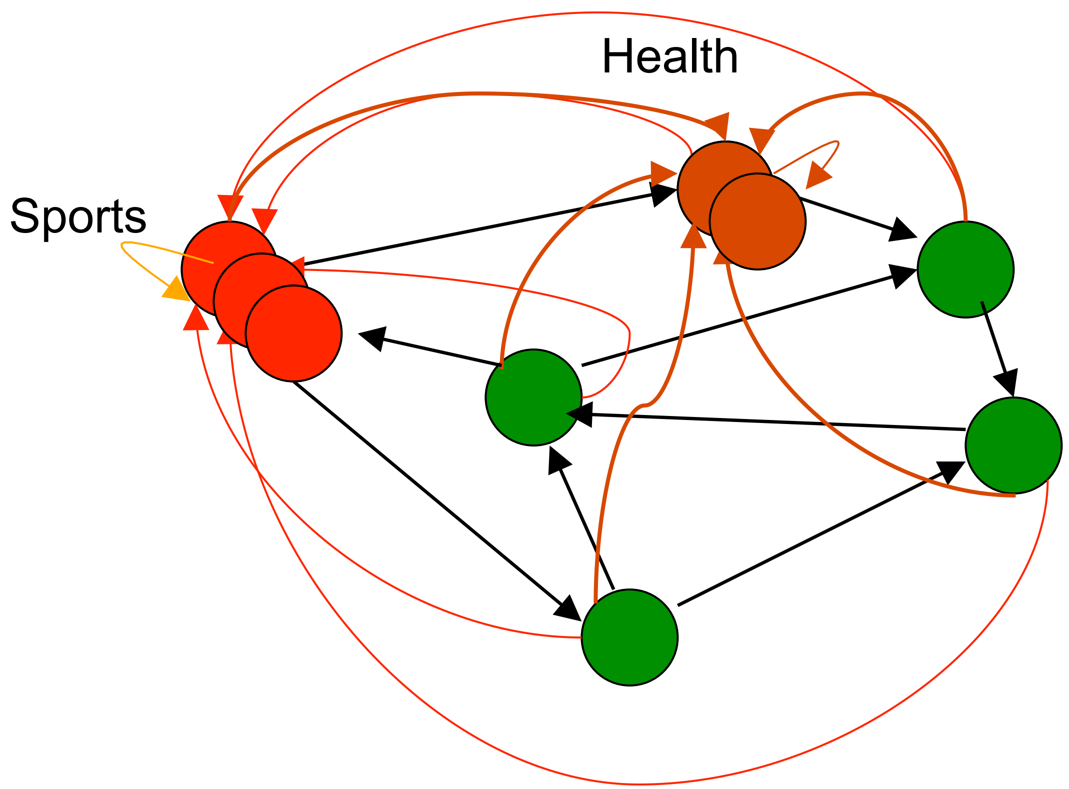
10% Sports teleportation

Interpretation



10% Health teleportation

Interpretation



pr = (0.9 PR_{sports} + 0.1 PR_{health}) gives you:
9% sports teleportation, 1% health teleportation

Kleinberg's Algorithm (HITS)

- 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

Kleinberg's Algorithm (HITS)

- *Main idea:* In many cases, when you search the web using some terms, the most relevant pages may not contain this term (or contain the term only a few times)
  - ★ *Harvard: www.harvard.edu*
  - ★ *Search Engines: yahoo, google, altavista*
  - ★ *Automobile manufacturers: Honda, Toyota…*

- *Authorities* and *hubs*

Hyperlink-Induced Topic Search (HITS)

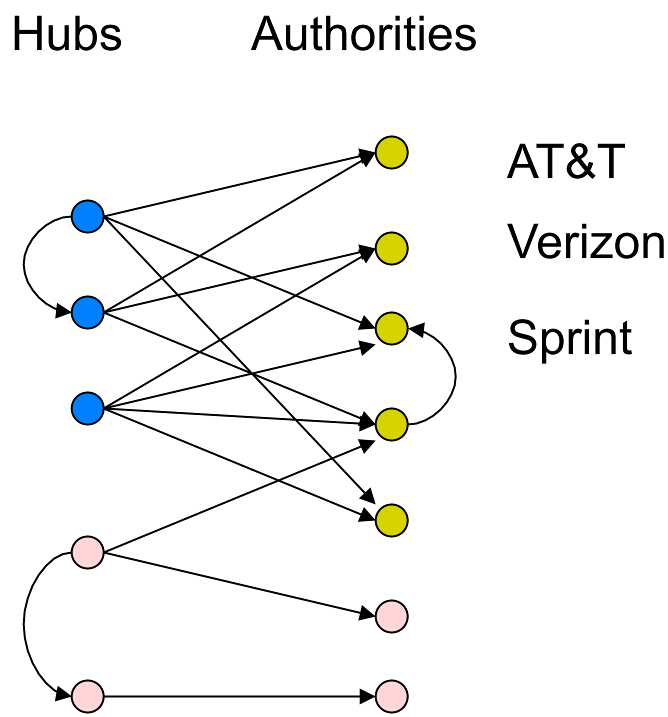
- In response to a query, instead of an ordered list of pages each meeting the query, find two sets of inter-related pages:
  - *Hub pages* are good lists of links on a subject.
    - e.g., “Bob’s list of cancer-related links.”
  - *Authority pages* occur recurrently on good hubs for the subject.
- Best suited for “broad topic” queries rather than for page-finding queries.
- Gets at a broader slice of common *opinion*.

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



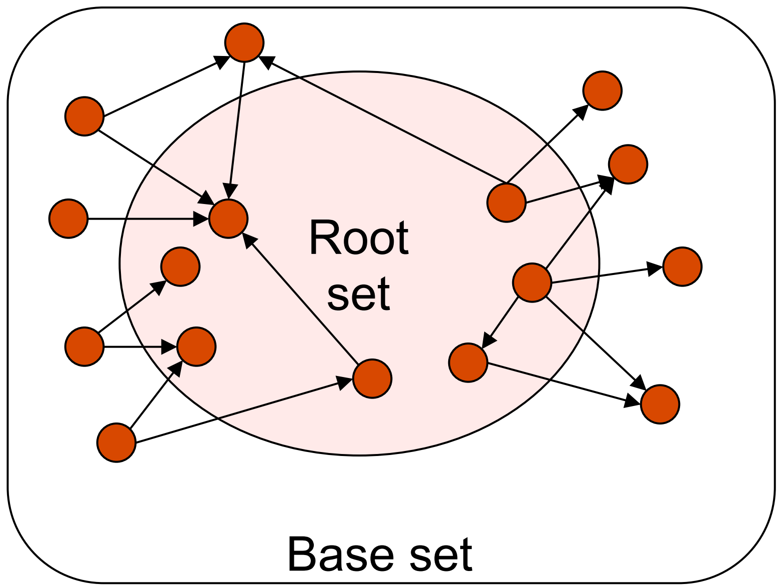
High-level scheme

- Extract from the web a base set of pages that *could* be good hubs or authorities.
- From these, identify a small set of top hub and authority pages;
  - → iterative algorithm

Base set

- Given text query (say ***browser***), use a text index to get all pages containing ***browser.***
  - Call this the <u>root set</u> of pages.
- Add in any page that either
  - points to a page in the root set, or
  - is pointed to by a page in the root set.
- Call this the <u>base set</u>.

Visualization



Assembling the base set

- Root set typically 200-1000 nodes.
- Base set may have thousands of nodes
  - Topic-dependent
- How do you find the base set nodes?
  - Follow out-links by parsing root set pages.
  - Get in-links (and out-links) from a connectivity server

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**

Distilling hubs and authorities

- Compute, for each page *x* in the base set, a hub score *h(x)* and an authority score *a(x).*
- Initialize: for all *x, h(x)←1; a(x) ←1*;
- Iteratively update all *h(x), a(x)*;

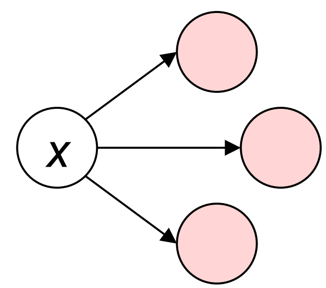
Key

- After iterations
  - ★ output pages with highest *h()* scores as top hubs
  - ★ highest *a()* scores as top authorities.

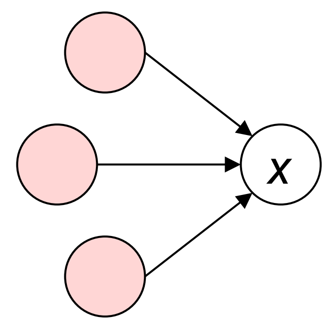
Iterative update

- Repeat the following updates, for all *x*:

$$h(x) \leftarrow \sum_{x \mapsto y} a(y)$$



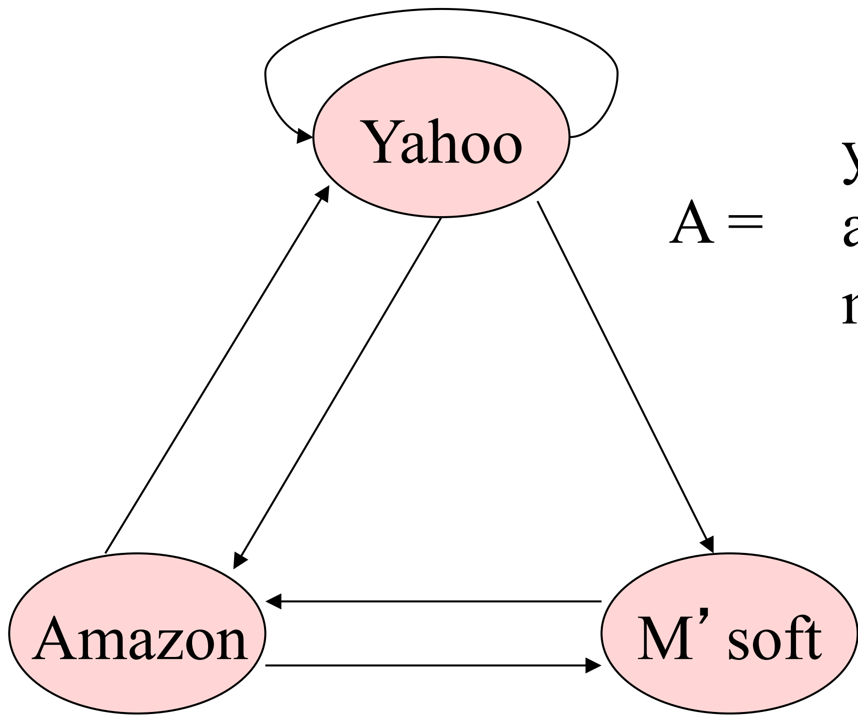
$$a(x) \leftarrow \sum_{y \mapsto x} h(y)$$



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



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

Scaling

- To prevent the *h()* and *a()* values from getting too big, can scale down after each iteration.
- Scaling factor doesn't really matter:
  - ★ we only care about the *relative* values of the scores.

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

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

$$\begin{array}{lcllllll}
a(\text{yahoo}) & = & 1 & 1 & 1 & 1 & \cdots & 1 \\
a(\text{amazon}) & = & 1 & 1 & 4/5 & 0.75 & \cdots & 0.732 \\
a(\text{m'soft}) & = & 1 & 1 & 1 & 1 & \cdots & 1
\end{array}$$

$$\begin{array}{lcllllll}
h(\text{yahoo}) & = & 1 & 1 & 1 & 1 & \cdots & 1.000 \\
h(\text{amazon}) & = & 1 & 2/3 & 0.71 & 0.73 & \cdots & 0.732 \\
h(\text{m'soft}) & = & 1 & 1/3 & 0.29 & 0.27 & \cdots & 0.268
\end{array}$$

Existence and Uniqueness

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

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

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

$\mathbf{a} = \lambda\mu A^TA\,\mathbf{a}$

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

How many iterations?

- Claim: relative values of scores will converge after a few iterations:
  - ★ In fact, suitably scaled, *h()* and *a()* scores settle into a steady state!
- We only require the relative orders of the *h()* and *a()* scores - not their absolute values.
- In practice, ~5 iterations get you close to stability.

Japan Elementary Schools

Hubs

- schools
- LINK Page-13
- “ú–{,ÌŠw Z
- a‰„ ¬Šw Zƒz [ƒ ƒy [ƒW
- 100 Schools Home Pages (English)
- K-12 from Japan 10/...rnet and Education)
- http://www...iglobe.ne.jp/~IKESAN
- ,l,f,j ¬Šw Z,U”N,P‘g•¨Œê
- ÒŠ—’¬—§ ÒŠ—“Œ ¬Šw Z
- Koulutus ja oppilaitokset
- TOYODA HOMEPAGE
- Education
- Cay's Homepage(Japanese)
- –y“ì ¬Šw Z,Ìƒz [ƒ ƒy [ƒW
- UNIVERSITY
- ‰J—³ ¬Šw Z DRAGON97-TOP
- Â‰ª ¬Šw Z,T”N,P‘gƒz [ƒ ƒy [ƒW
- ¶µ°é¼ÂÁ© ¥á¥Ë¥å¡¼ ¥á¥Ë¥å¡¼

Authorities

- The American School in Japan
- The Link Page
- ‰ª è s—§ˆä“c ¬Šw Zƒz [ƒ ƒy [ƒW
- Kids' Space
- ˆÀ é s—§ˆÀ é ¼•” ¬Šw Z
- ‹{ é‹³ˆç‘åŠw• ‘® ¬Šw Z
- KEIMEI GAKUEN Home Page (Japanese)
- Shiranuma Home Page
- fuzoku-es.fukui-u.ac.jp
- welcome to Miasa E&J school
- _“Þ ìŒ§ E‰¡•l s—§’† ì ¼ ¬Šw Z,Ìƒy
- http://www...p/~m_maru/index.html
- fukui haruyama-es HomePage
- Torisu primary school
- goo
- Yakumo Elementary,Hokkaido,Japan
- FUZOKU Home Page
- Kamishibun Elementary School...

Things to note

- Pulled together good pages regardless of language of page content.
- Use *only* link analysis after base set assembled
  - iterative scoring is query-independent.
- Iterative computation after text index retrieval - significant overhead.

Kleinberg's algorithm - results

Eg., for the query 'java':

0.328 www.gamelan.com

0.251 java.sun.com

0.190 www.digitalfocus.com ("the java developer")

Kleinberg's algorithm - discussion

- 'authority' score can be used to find 'similar pages' to page p
- closely related to 'citation analysis', social networks / 'small world' phenomena

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

Web Spam

- Search has become the default gateway to the web
- Very high premium to appear on the first page of search results
  - e.g., e-commerce sites
  - advertising-driven sites

The trouble with paid search ads …

- It costs money. What's the alternative?
- *Search Engine Optimization:*
  - ★ "Tuning" your web page to rank highly in the algorithmic search results for select keywords
  - ★ Alternative to paying for placement
  - ★ Thus, intrinsically a marketing function
- Performed by companies, webmasters and consultants ("Search engine optimizers" or SEO) for their clients
- Some perfectly legitimate, some very shady

Most Expensive Keywords

- http://www.wordstream.com/download/docs/most-expensive-keywords.pdf

What is web spam?

- Spamming = any deliberate action solely in order 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 defintion
  - ★ SEO industry might disagree!
- Some estimated that 60% of all web pages are spam

Search engine optimization (Spam)

- Motives
  - Commercial, political, religious, lobbies
  - Promotion funded by advertising budget
- Operators
  - Contractors (Search Engine Optimizers) for lobbies, companies
  - Web masters
  - Hosting services
- Forums
  - E.g., Web master world (www.webmasterworld.com)
    - Search engine specific tricks
    - Discussions about academic papers ☺

Simplest forms

- First generation engines relied heavily on *tf/idf*
  - ★ The top-ranked pages for the query **`maui resort`** were the ones containing the most **`maui`**'s and **`resort`**'s
- SEOs responded with dense repetitions of chosen terms
  - ★ e.g., **`maui resort maui resort maui resort`**
  - ★ Often, the repetitions would be in the same color as the background of the web page
    - ✓ Repeated terms got indexed by crawlers
    - ✓ But not visible to humans on browsers

Pure word density cannot
be trusted as an IR signal

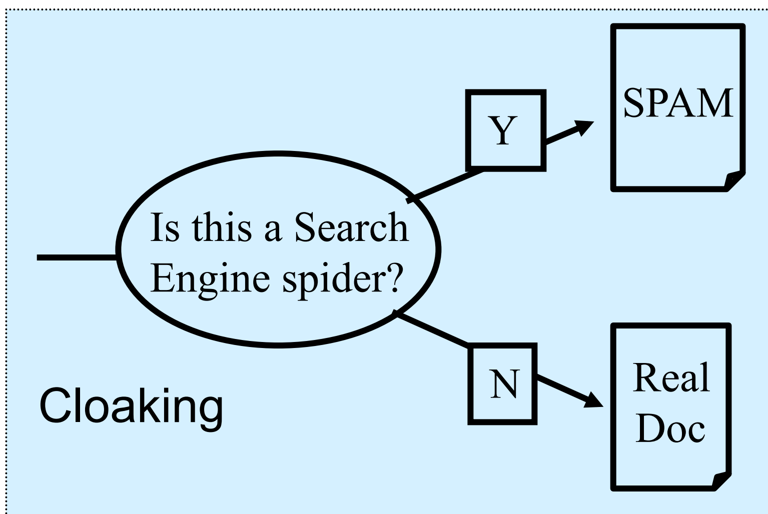
Variants of keyword stuffing

- Misleading meta-tags, excessive repetition
- Hidden text with colors, style sheet tricks, etc.

Meta-Tags =
"… London hotels, hotel, holiday inn, hilton, discount, booking, reservation, sex, mp3, britney spears, viagra, …"

Cloaking

- Serve fake content to search engine spider



More Spamming Techniques

- 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 Spamming

Term spam targets

- Body of web page
- Title
- URL
- HTML meta tags
- Anchor text

More on spam

- Web search engines have policies on SEO practices they tolerate/block
  - ★ http://help.yahoo.com/kb/index?page=answers&startover=y&y=PROD&source=content.landing_search&locale=en_US&question_box=SEO
  - ★ http://www.google.com/intl/en/webmasters/
- Adversarial IR: the unending (technical) battle between SEO's and web search engines
- Research http://airweb.cse.lehigh.edu/

Web Spam Taxonomy

- We follow the treatment by Gyongyi and Garcia-Molina [2004]
- 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
  - ★ We have already seen term spamming earlier
  - ★ Manipulating the text of web pages in order to appear relevant to queries
- Link spamming
  - ★ Creating link structures that boost page rank or hubs and authorities scores

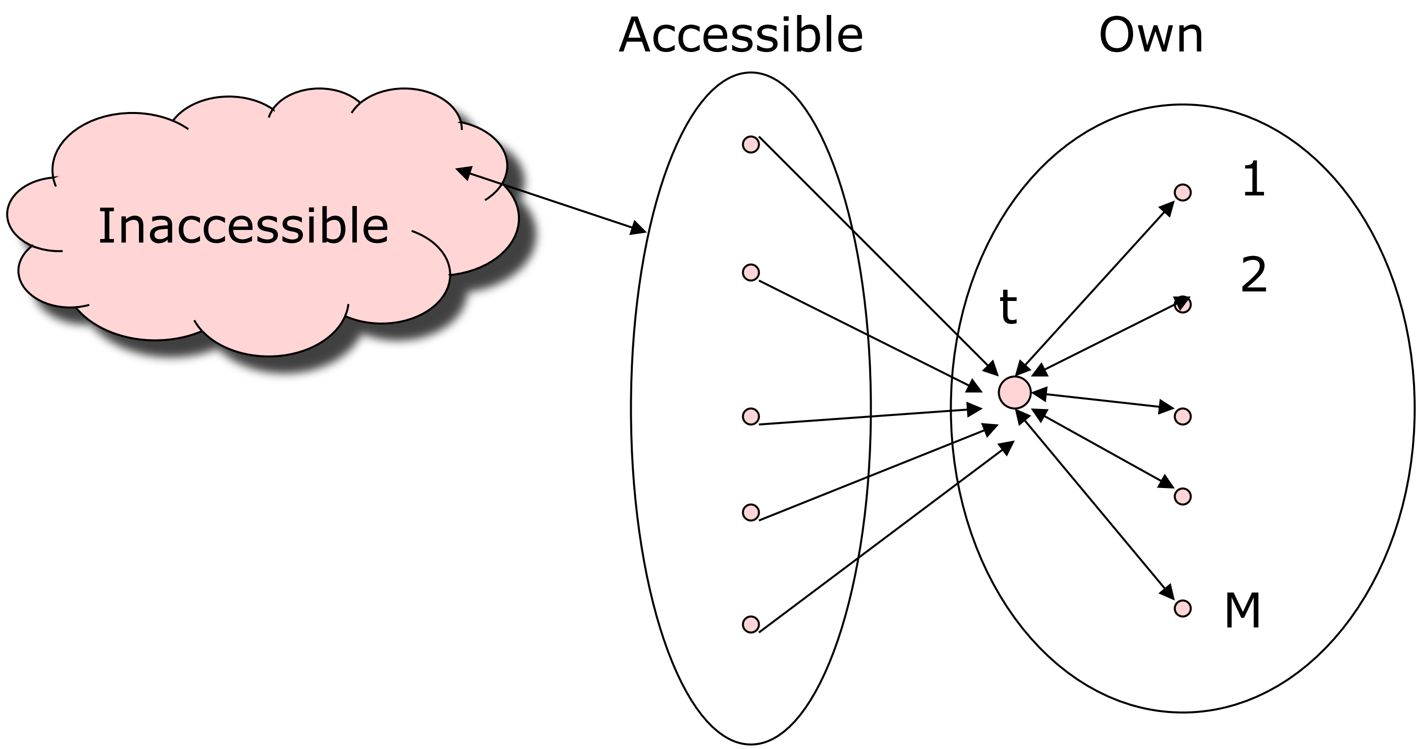
Link spam

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

Link Farms

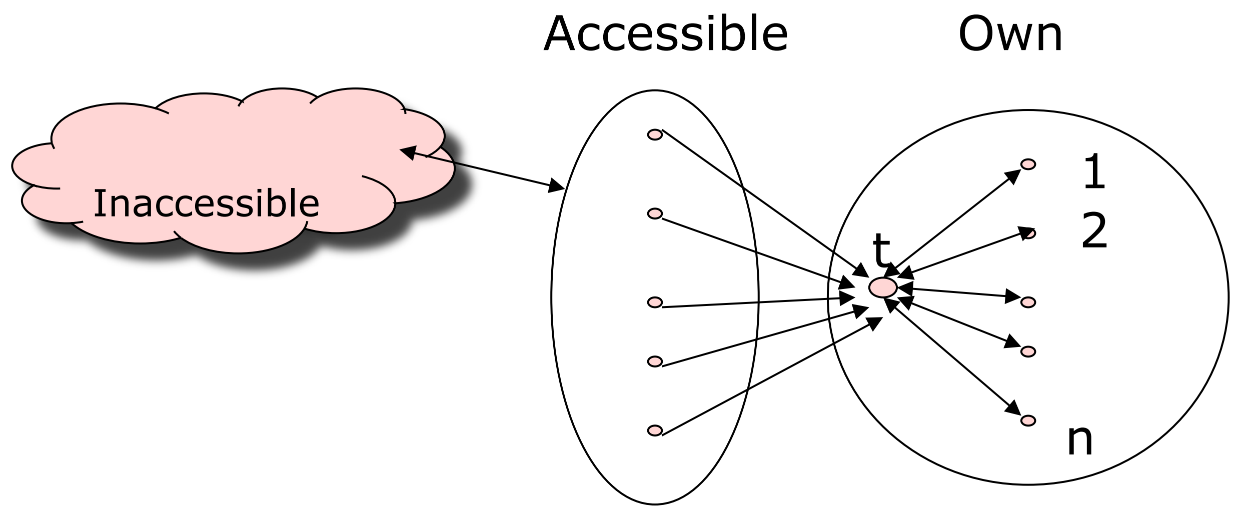
- Spammer's goal
  - ★ Maximize the page rank of target page t
- Technique
  - ★ Get as many links from accessible pages as possible to target page t
  - ★ Construct "link farm" to get page rank multiplier effect

Link Farms



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

Analysis



Suppose rank contributed by accessible pages = x

Let page rank of target page = y

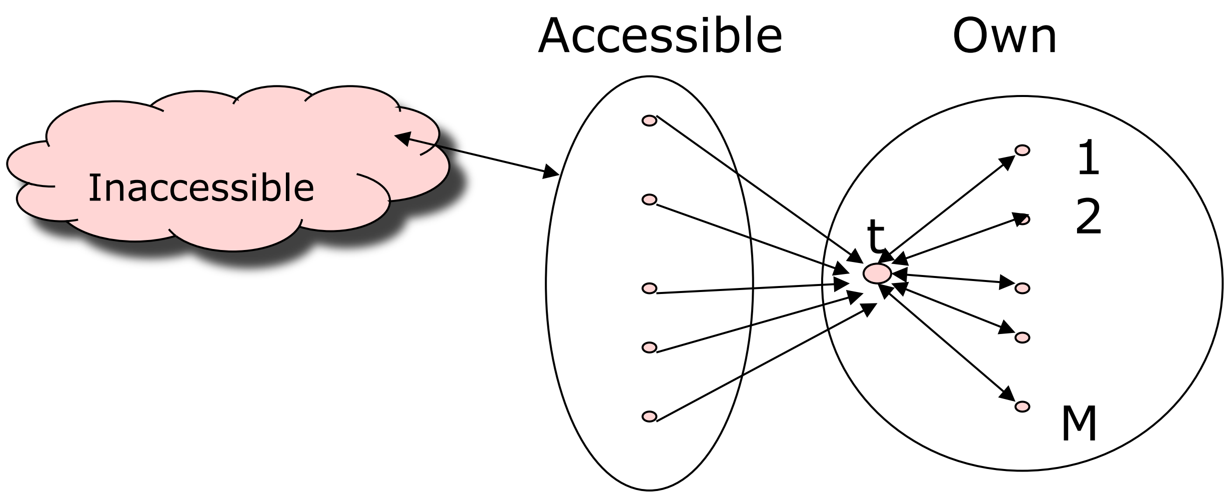
Rank of each “farm” page = by/n + (1-b)/N

$y = x + \beta(n[by/n + (1\text{-}b)/N]) + \boxed{(1\text{-}b)/N}$ Very small; ignore

$\quad = x + b^2y + b(1\text{-}b)n/N + (1\text{-}b)/N$

$y = x/(1\text{-}b^2) + cn/N$ where $c = \beta/(1+\beta)$

Analysis



- $y = x/(1-b^2) + cM/N$ where $c = \beta/(1+\beta)$
- For $b = 0.85$, $1/(1-b^2) = 3.6$
  - ★ Multiplier effect for "acquired" page rank
  - ★ By making M large, we can make y as large as we want

Hiding techniques

- Content hiding
  - ★ Use same color for text and page background
- Cloaking
  - ★ Return different page to crawlers and browsers
- Redirection
  - ★ Alternative to cloaking
  - ★ Redirects are followed by browsers but not crawlers

Detecting Spam

- Term spamming
  - Analyze text using statistical methods e.g., Naïve Bayes classifiers
  - 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, so 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 outlinks from a page, the less scrutiny the page author gives each outlink
  - ★ Trust is “split” across outlinks

Simple model

- Suppose trust of page p is t(p)
  - ★ Set of outlinks 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 inlinked pages
- Note similarity to Topic-Specific Page Rank
  - ★ Within a scaling factor, trust rank = biased page rank 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 page rank
  - ★ Assume high page rank pages are close to other highly ranked pages
  - ★ We care more about high page rank “good” pages

Inverse page rank

- Pick the pages with the maximum number of outlinks
- Can make it recursive
  - ★ Pick pages that link to pages with many outlinks
- Formalize as "inverse page rank"
  - ★ Construct graph G' by reversing each edge in web graph G
  - ★ Page Rank in G' is inverse page rank in G
- Pick top k pages by inverse page rank

Spam Mass

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

Spam mass estimation

$r(p)$ = page rank of page p

$r^{+}(p)$ = page rank of p with teleport into “good” pages only

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

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

Reading

- Ch. 21 Link Analysis from Information Retrieval: http://nlp.stanford.edu/IR-book/pdf/21link.pdf

(Optional) Original papers if you are interested:

- Brin, S. and L. Page (1998). Anatomy of a Large-Scale Hypertextual Web Search Engine. 7th Intl World Wide Web Conf.
- Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. J. ACM 46, 5.
- Gyongyi, Z., Berkhin, P., Garcia-Molina, H., and Pedersen, J. 2006. Link spam detection based on mass estimation. In *Proceedings of the 32nd international Conference on Very Large Data Bases*