Link Analysis and spam

Slides adapted from

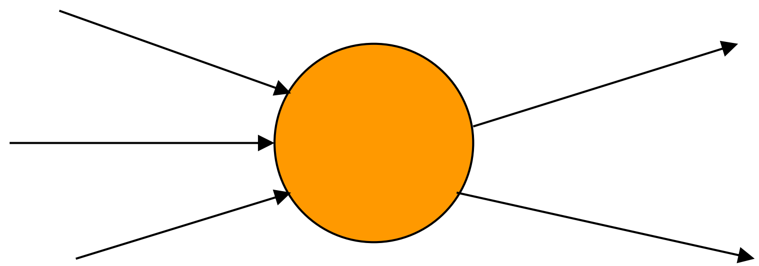
- Information Retrieval and Web Search, Stanford University, Christopher Manning and Prabhakar Raghavan
- CS345A: Data Mining. Stanford University, Anand Rajaraman, Jeffrey D. Ullman

Query processing

- Relevance: retrieve all pages meeting the query, and order them with relevance (based on e.g., vector space model)
- Popularity: order pages by their link popularity
- Combination: merge the results using popularity and relevance

Query-independent ordering

- First generation: using link counts as simple measures of popularity.
- Two basic suggestions:
  - Undirected popularity:
    - page 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).

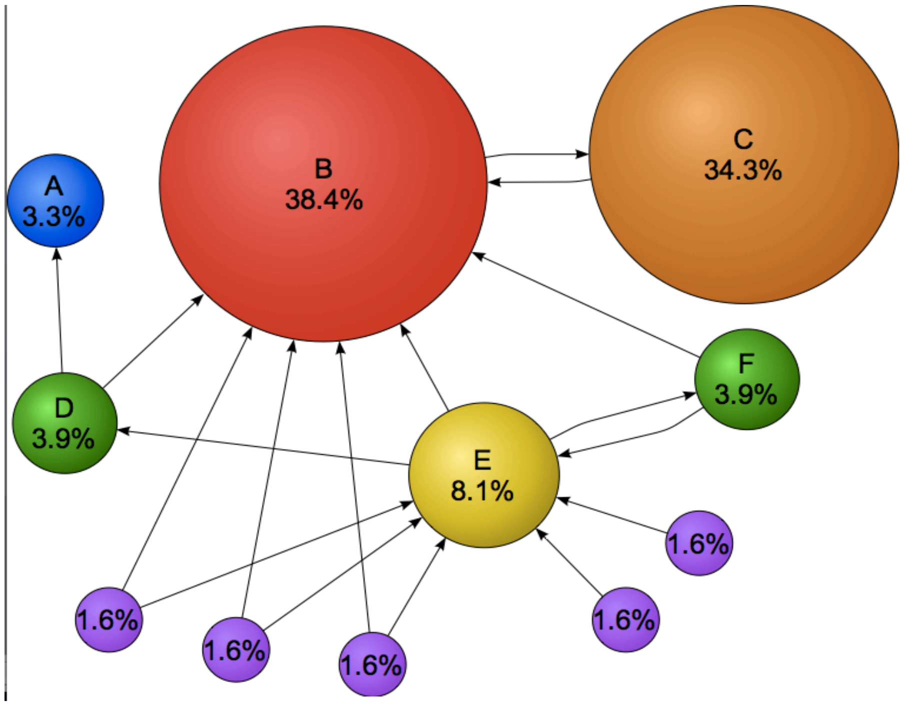


Voting by in-links

- The importance of a page can not be decided only by the internal features of the page
- The ‘quality’ can be judged from links pointing to the page
- Link is an implicit endorsement
- Link may convey different meanings
  - May be criticism, paid ad
  - In aggregate, it is a collective endorsement.

Pagerank

- in-links are not equal, because
  - Web pages are not equally "important". A vote from an important page weighs more than a less important page (if same number of votes are casted)
  - A vote weighs more if the page gives only few votes (links)
- In order the decide the pagerank value of B, you need to have the value of C.
- It is a recursive question!
- Originated from studies in citation-network



Picture from Wikipedia.com

An example web graph

Obtained by a random walk of 10000 steps on the University of Notre Dame web

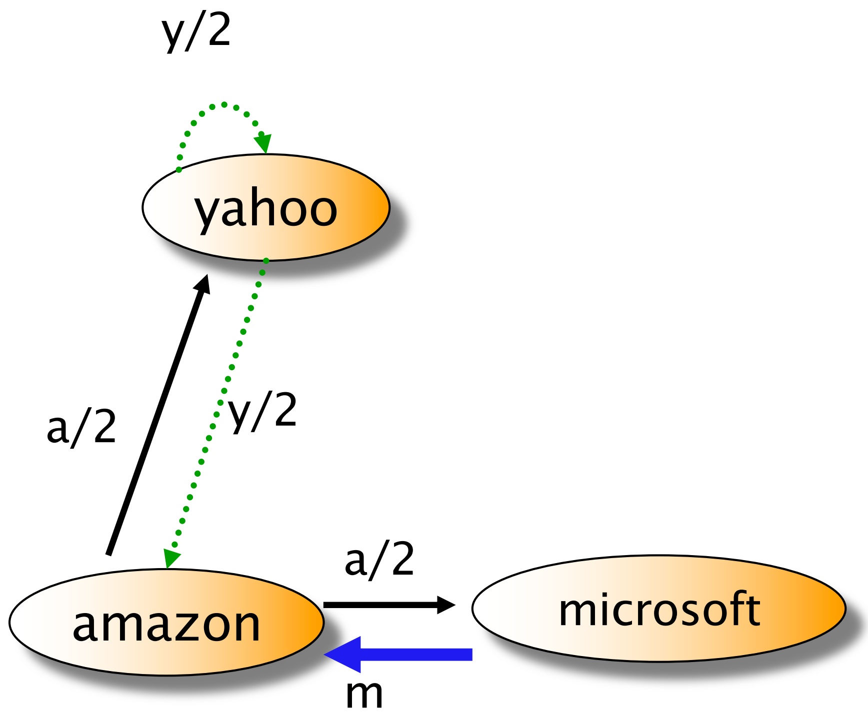
Original graph is from

http://snap.stanford.edu/data/index.html

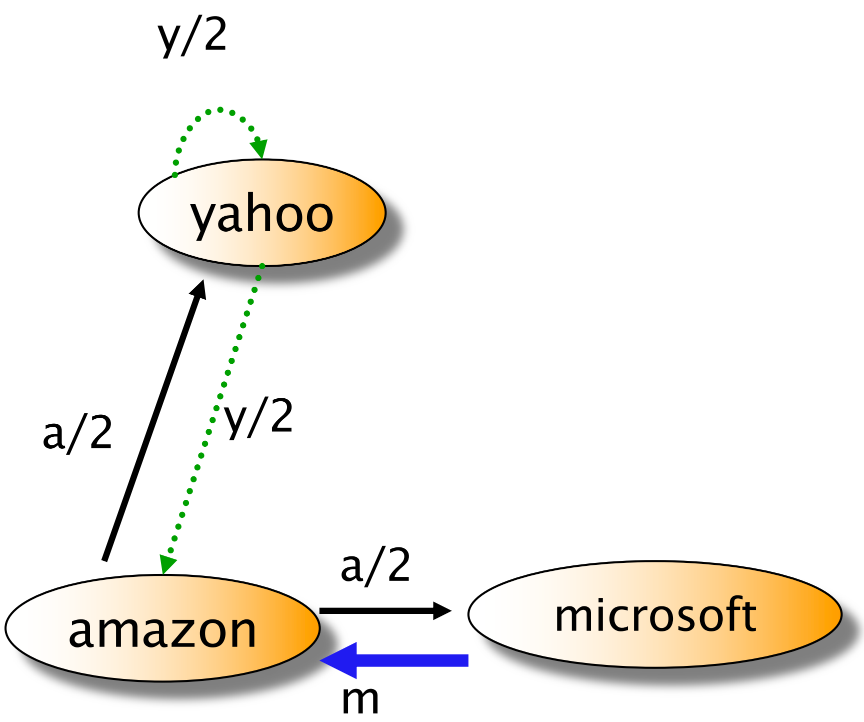
Note the spammed page

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



- There are three web pages
  - Yahoo gives out two votes, each worth y/2
  - Amazon gives out two votes, each worth a/2
  - Microsoft gives out one vote

$y = y/2 + a/2$

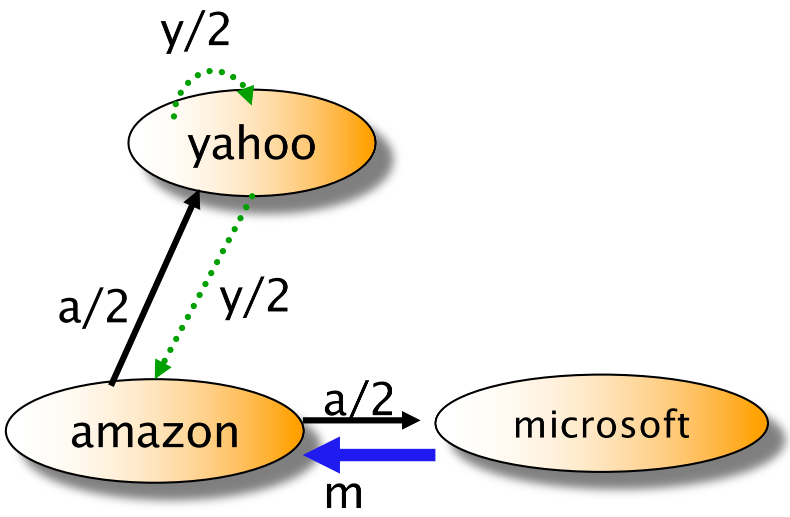
$a = y/2 + m$

$m = a/2$

Solving the flow equation

- 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
- Again, scalability is key in computer science.

Matrix formulation



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

r=Mr

$$r_i = \sum_{j=1}^{N} M_{ij} r_j$$

$$r_1 = 0.5 r_1 + 0.5 r_2 + 0 r_3$$

- Matrix M has one row and one column for each web page
- Suppose page j has n outlinks
  - If j -> i, then Mij=1/n
  - else Mij=0
- M is a *column* stochastic matrix
  - Columns sum to 1
  - Usually rows 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

Matrix formulation

	y	a	m
y	1/2	1/2	0
a	½		1
m		½	

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

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

r=Mr

Power Iteration method

$$Initialize: r_0 = \left(1/N \quad \ldots \quad 1/N\right)^T$$

$$Iterate: r_{k+1} = Mr_k$$

$$stop\ when \quad \left|r_{k+1} - r_k\right|_1 < \varepsilon$$

$$\left|x\right|_1 = \sum_{1 \le i \le N} \left|x_i\right| \text{ is the } L_1 \text{ norm}$$

- There are many methods to find r.
- Power method is the most simple one
- It is known to be slow compared with other methods
- Yet Google uses this because
  - It is simple
  - It saves space
  - Empirically it converges quickly (~ 50 iterations)

Power iteration method

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

1/2* 1/3 +1/2*1/3
=1/3

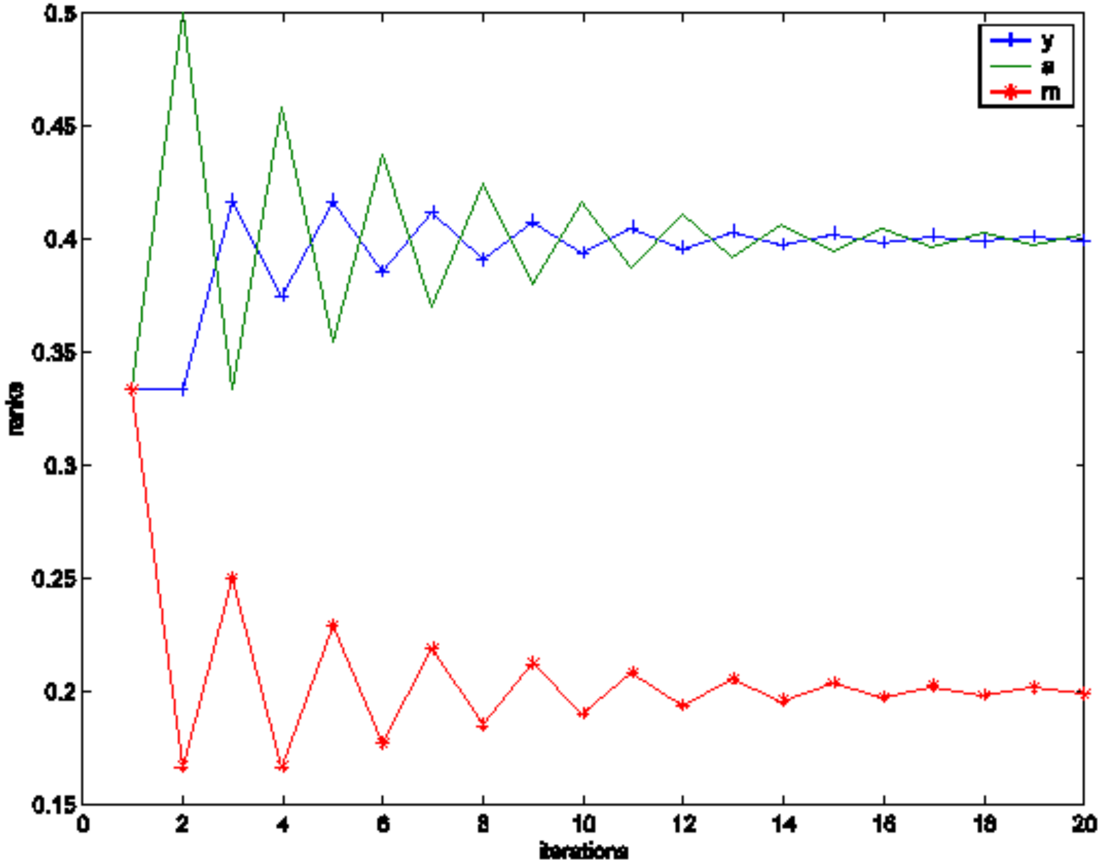
$$\begin{pmatrix} y \\ a \\ m \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix}, \begin{pmatrix} 1/3 \\ 1/2 \\ 1/6 \end{pmatrix}, \begin{pmatrix} 5/12 \\ 1/3 \\ 1/4 \end{pmatrix}, \begin{pmatrix} 3/8 \\ 11/24 \\ 1/6 \end{pmatrix}, \ldots \begin{pmatrix} 2/5 \\ 2/5 \\ 1/5 \end{pmatrix}$$

Initially, (y a m)=(1/3 1/3 1/3)

1/2* 1/3 +1*1/3
=1/2

The matlab program

The rank values fluctuate, then reach a steady state



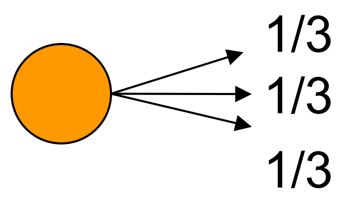

```
M=[1/2,1/2,0;  1/2,0,0; 0,1/2,1];
r=[1/3;1/3;1/3]
interval=1:20;
for i=interval
   x(i)=r(1);
   y(i)=r(2);
   z(i)=r(3);
   r=M*r
end;
Plot (interval, x, '+-', interval, y,'-', interval, z,'*-');
```

Questions to be answered

- Will the iteration converge? Or continue infinitely?
- When will it converge?
- When it converges, is there an intuitive interpretation for the values of r?
- Will it converge to one vector, or multiple vectors?
- If it converges, how many steps does it take?

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 ith component is the probability that the surfer is at page i at time t
  - 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

  p(t+1) = M p(t)

- Suppose the random walk reaches a state such that

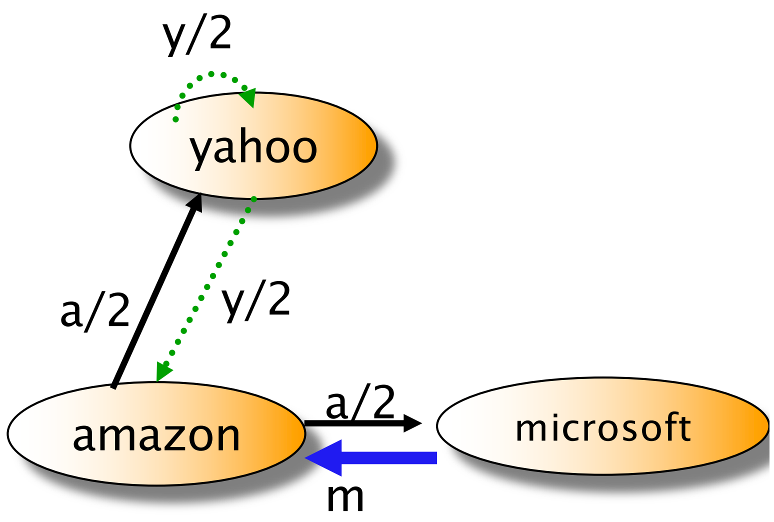
  p(t+1) = M p(t) = p(t)

  - Then p(t) is called a stationary distribution for the random walk

- Our rank vector r satisfies Mr=r
- So it is a stationary distribution for the random surfer
- The limiting r is an eigenvector of M
  - v is an eigenvector of M if $Mv=\lambda v$
- r is also the principal eigenvector
  - The associated eigenvalue is the largest

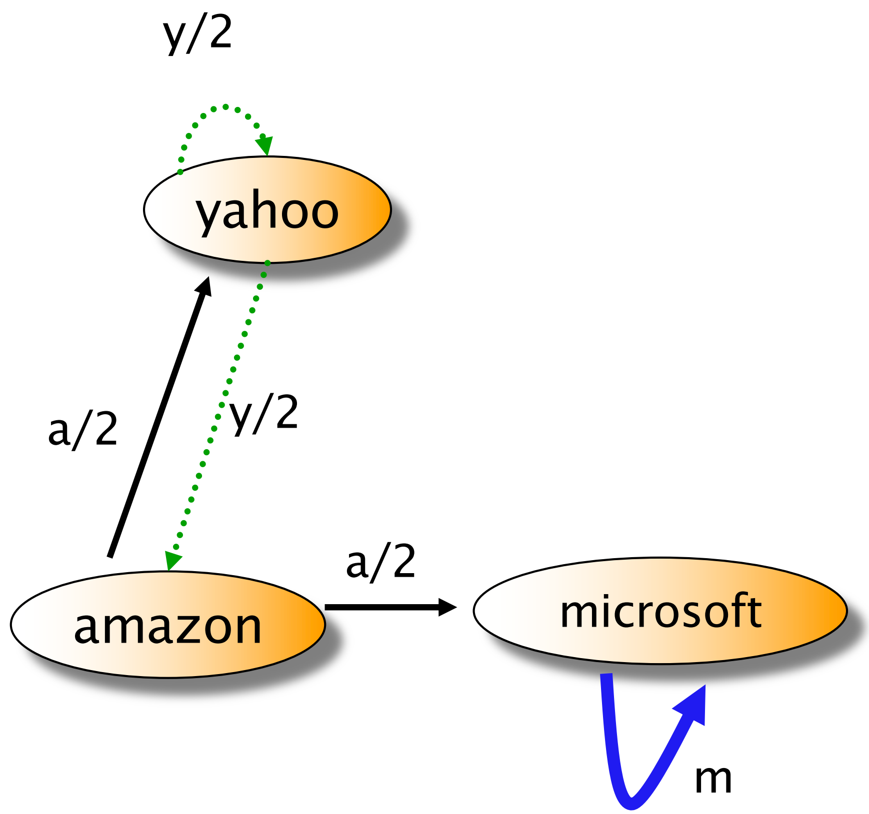
Existence and Uniqueness

- 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.
- Strongly connected
- Aperiodic: return to state i can occur any time
  - Bipartite graph is periodic with period 2.



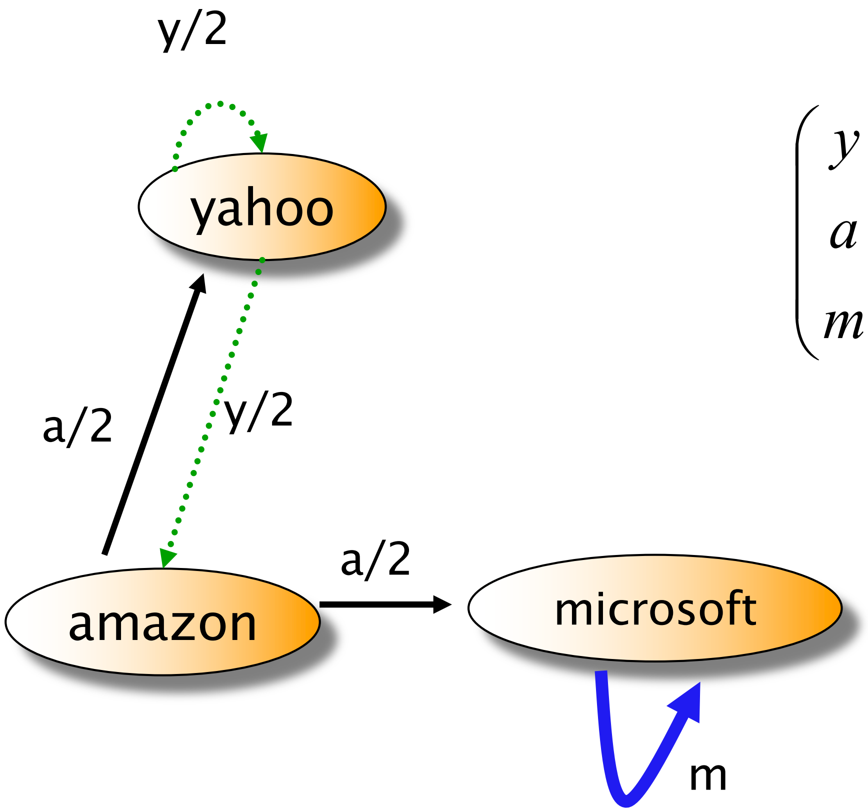
Spider trap

- 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…

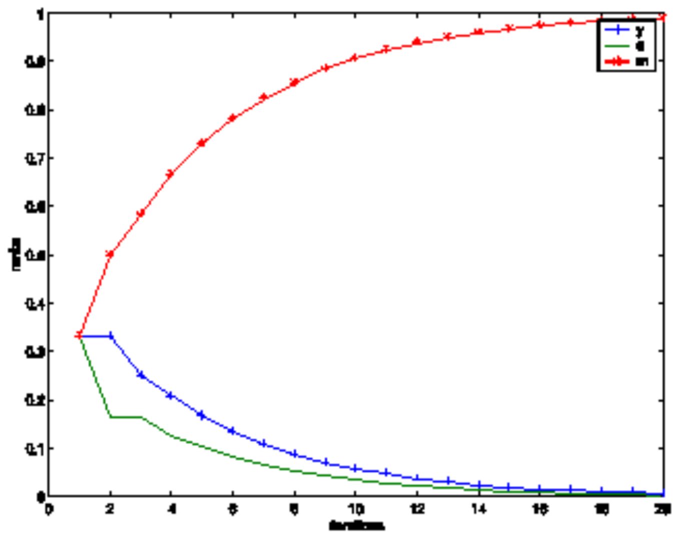
$$\begin{pmatrix} y \\ a \\ m \end{pmatrix} = \begin{pmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{pmatrix} \begin{pmatrix} y \\ a \\ m \end{pmatrix}$$



$$\begin{pmatrix} y \\ a \\ m \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix}, \begin{pmatrix} 2/6 \\ 1/6 \\ 3/6 \end{pmatrix}, \begin{pmatrix} 3/12 \\ 2/12 \\ 7/12 \end{pmatrix}, \begin{pmatrix} 5/24 \\ 3/24 \\ 16/24 \end{pmatrix}, \ldots \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Both yahoo and amazon have zero pageRank

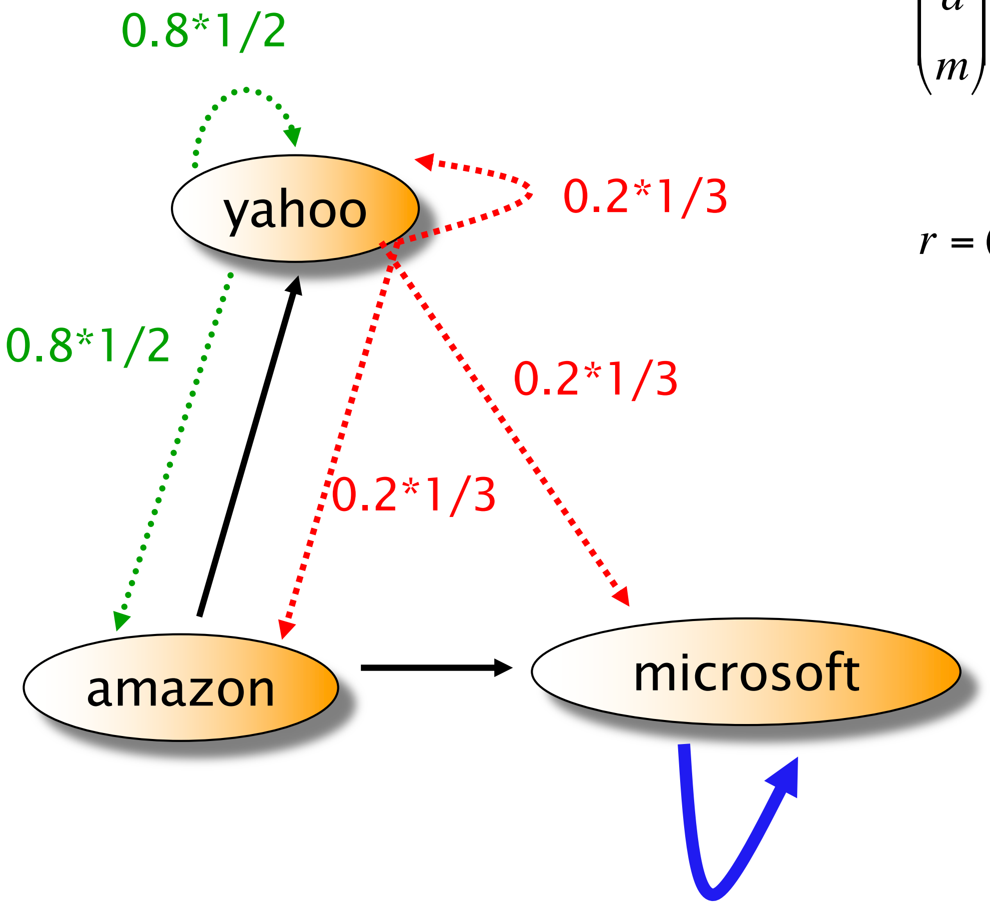
The spider trap can contain a group of pages



Random teleporting

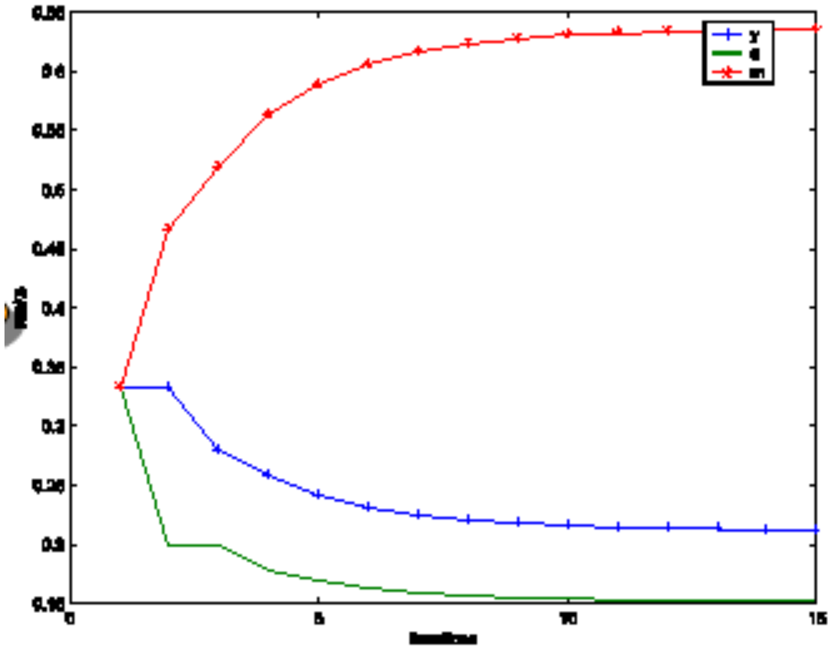
- The Google solution for spider traps
- At each time step, the random surfer has two options:
  - With probability b, follow a link at random
  - With probability 1-b, jump to some page uniformly at random
  - Common values for b are in the range 0.8 to 0.9
- Surfer will teleport out of spider trap within a few time steps

Random teleports (beta=0.8)

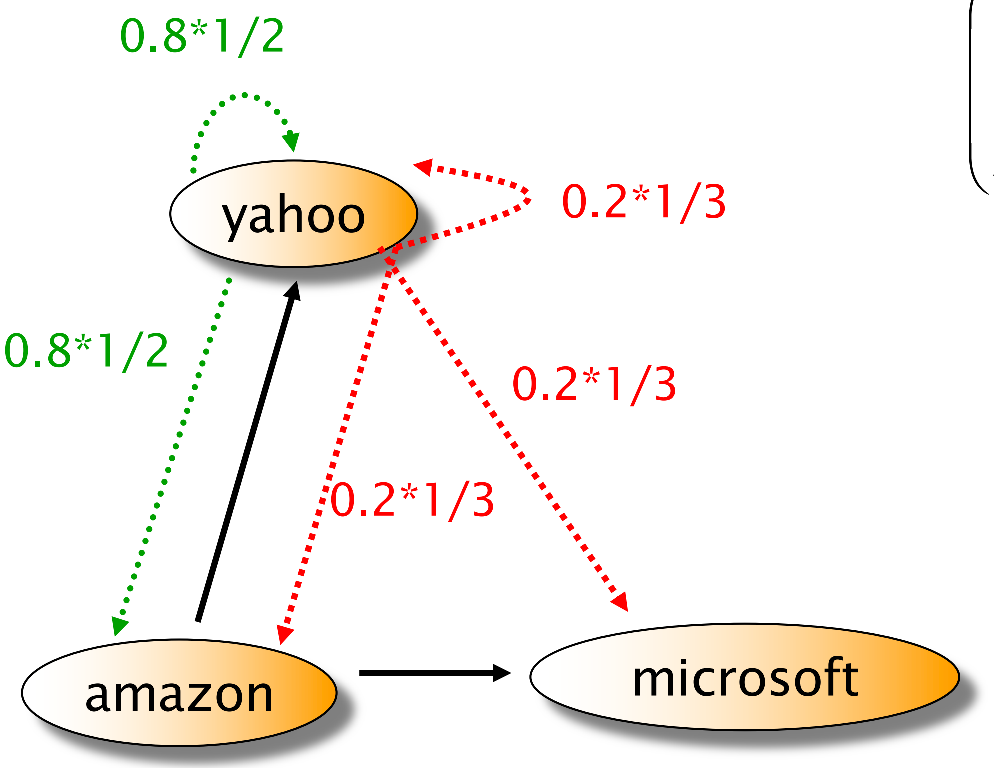


$$\begin{pmatrix} y \\ a \\ m \end{pmatrix} = 0.8 \begin{pmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{pmatrix} \begin{pmatrix} y \\ a \\ m \end{pmatrix} + 0.2 \begin{pmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{pmatrix} \begin{pmatrix} y \\ a \\ m \end{pmatrix}$$

$$r = (\beta M + (1-\beta)\frac{1}{n} e e^T) r$$



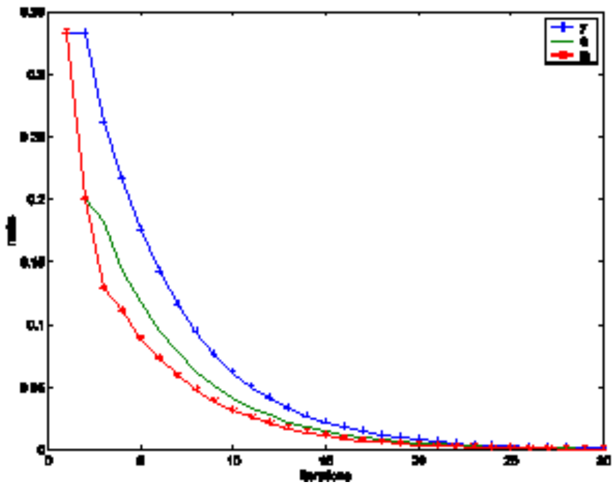
Dead end



$$\begin{pmatrix} y \\ a \\ m \end{pmatrix} = 0.8\begin{pmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \end{pmatrix}\begin{pmatrix} y \\ a \\ m \end{pmatrix} + 0.2\begin{pmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{pmatrix}\begin{pmatrix} y \\ a \\ m \end{pmatrix}$$

$$\begin{pmatrix} y \\ a \\ m \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix}, \begin{pmatrix} 0.33 \\ 0.2 \\ 0.2 \end{pmatrix}, \begin{pmatrix} 0.2622 \\ 0.1822 \\ 0.1289 \end{pmatrix}, \begin{pmatrix} 0.2160 \\ 0.1431 \\ 0.1111 \end{pmatrix}, \ldots \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

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



Dealing with dead-ends

- Teleport
  - Follow random teleport links with probability 1.0 from dead-ends
  - Adjust matrix accordingly

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

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

- Prune and propagate
  - Preprocess the graph to eliminate dead-ends
  - Might require multiple passes
  - Compute page rank on reduced graph
  - Approximate values for dead ends by propagating values from reduced graph

Summarize PageRank

- Derived from Markov chain
- Importance (prestige, authority) of the page is the probability to reach the page in random walk
- Power method is used to calculate the probability
- when it is Ergodic Markov chain
  - It converges
  - It converges to a unique value
  - It converges quickly
  - Need to deal with non-Ergodic Markov chain in practice
    - Random teleporting makes the states (pages) connected
    - Dead end page makes the matrix no longer stochastic

Simple Pagerank in java

- http://introcs.cs.princeton.edu/java/16pagerank/

- Using Matlab or Octave
  - Using power iteration
  - Calculate the eigenvector

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)

For citation network in academic papers

- Citation network is mostly acyclic
- Readers may surf both down stream and upstream (follow cited and citing papers)
- Author and other data may also play a role

Graph embedding and random walk

- Most graph embedding algorithms are based on random walk
  - E.g., DeepWalk, Node2Vec
  - Create random walk paths.
  - Treat path as ‘text’, then run word embedding algorithms on the ‘text’
- The difference is about how to control the random walk
- Random walk path need to be long (e.g., 100 steps)
  - Note that PageRank restarts around 10.

The reality

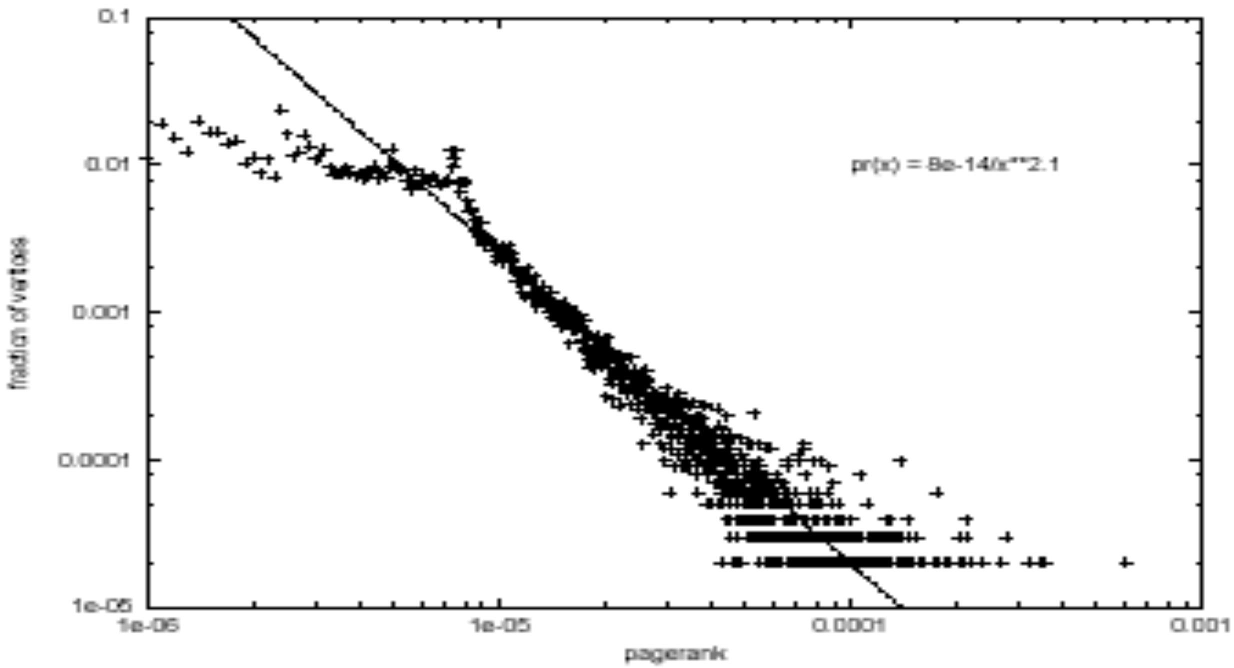
- Pagerank is used in google, but is hardly the full story of ranking
  - Many sophisticated features are used
  - Some address specific query classes
  - Machine learned ranking heavily used
- Pagerank still very useful for things like crawl policy

Topic Specific Pagerank

- Goal – pageRank values that depend on query *topic*
  - Query "random walk" most probably in academia
- Topic-specific ranking: 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 (Open Directory Project) categories based on a query & user -specific distribution over the categories
  - Teleport to a page uniformly at random within the chosen topic
- For topic-specific query, it is hard to implement: can't compute PageRank at query time

PageRank in real world

- Log Plot of PageRank Distribution of Brown Domain (*.brown.edu)



G.Pandurangan, P.Raghavan,E.Upfal,"Using PageRank to characterize Webstructure" ,COCOON 2002

Google's secret list (from searchengineland.com)

- Eric Schmidt, Sept 16, 2010
  - Presence of search term in HTML title tag
  - Use of bold around search term
  - Use of header tags around search term
  - Presence of search term in anchor text leading to page
  - PageRank of a page
  - PageRank / authority of an entire domain
  - Speed of web site
  - …

There are 200 variables in google algorithm

- At PubCon, Matt Cutts mentioned that there were over 200 variables in the Google Algorithm

- Domain
  - Age of Domain
  - History of domain
  - KWs in domain name
  - Sub domain or root domain?
  - TLD of Domain
  - IP address of domain
  - Location of IP address / Server

- Architecture
  - HTML structure
  - Use of Headers tags
  - URL path
  - Use of external CSS / JS files

- Content
  - Keyword density of page
  - Keyword in Title Tag
  - Keyword in Meta Description (Not Meta Keywords)
  - Keyword in KW in header tags (H1, H2 etc)
  - Keyword in body text
  - Freshness of Content

- Per Inbound Link
  - Quality of website/page linking in
  - Age of website /page
  - Relevancy of page's content
  - Location of link (Footer, Navigation, Body text)
  - Anchor text if link
  - Title attribute of link
  - Alt tag of images linking
  - Country specific TLD domain
  - Authority TLD (.edu, .gov)
  - Location of server
  - Authority Link (CNN, BBC, etc)

- Cluster of Links
  - Uniqueness of Class C address.

- Internal Cross Linking
  - No. of internal links to page
  - Location of link on page
  - Anchor text of FIRST text link (Bruce Clay's point at PubCon)

- Penalties
  - Over Optimisation
  - Purchasing Links
  - Selling Links
  - Comment Spamming
  - Cloaking
  - Hidden Text
  - Duplicate Content
  - Keyword stuffing
  - Manual penalties
  - Sandbox effect (Probably the same as age of domain)

- Miscellaneous
  - JavaScript Links
  - No Follow Links

- Pending
  - Performance / Load of a website
  - Speed of JS

- Misconceptions
  - XML Sitemap (Aids the crawler but doesn't help rankings)
  - PageRank (General Indicator of page's performance)

Web search, SEO, Spam

Slides adapted from

– Information Retrieval and Web Search, Stanford University, Christopher Manning and Prabhakar Raghavan

– CS345A, Winter 2009: Data Mining. Stanford University, Anand Rajaraman, Jeffrey D. Ullman

Spam (SEO)

- 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 definition
- SEO (search engine optimization) industry might disagree!
- Approximately 10-15% of web pages are spam
- Spamming also happens in online social networks

Motivation for SEO and/or SPAM

- You have a page that will generate lots of revenue for you if people visit it.
  - Commercial, political, religious, lobbies
- Therefore, you would like to direct visitors to this page.
- One way of doing this: get your page ranked highly in search results.
- How can I get my page ranked highly?
  - Contractors (Search Engine Optimizers) for lobbies, companies
  - Web masters
  - Hosting services

Spamming techs

- Boosting techniques
  - Techniques for achieving high relevance/importance for a web page
  - Term (content) spamming
    - 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
- Hiding techniques
  - Techniques to hide the use of boosting
    - From humans and web crawlers

Term Spamming

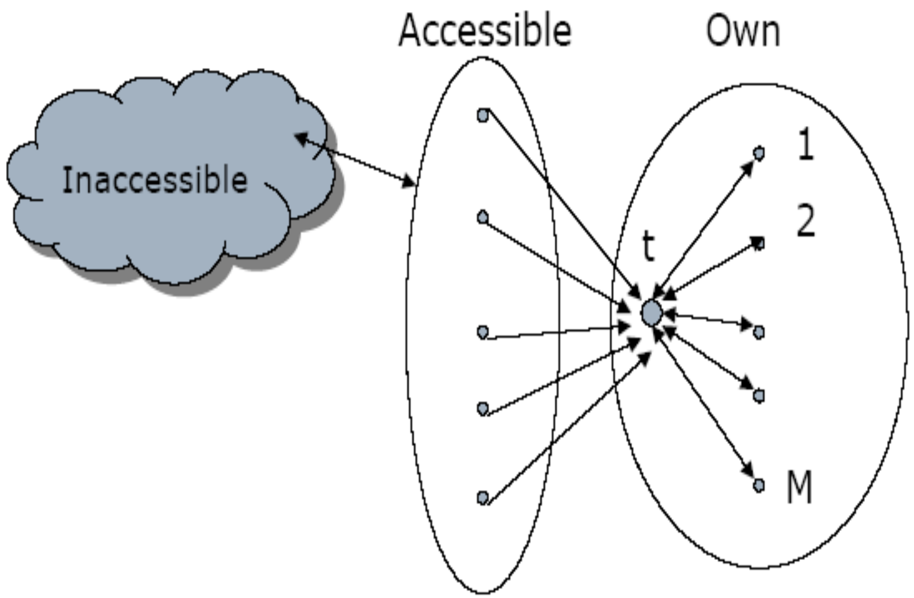
- Repetition
  - of one or a few specific terms
  - Goal is to subvert TF/IDF ranking schemes, so that the ranking is increased
  - First generation engines relied heavily on *tf/idf*
  - e.g. The top-ranked pages for the query **`maui resort`** were the ones containing the most **`maui`**'s and **`resort`**'s
  - 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
- Dumping
  - of a large number of unrelated terms
  - e.g., copy entire dictionaries, so that the page is matched no matter what is the query

Term spam target

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

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

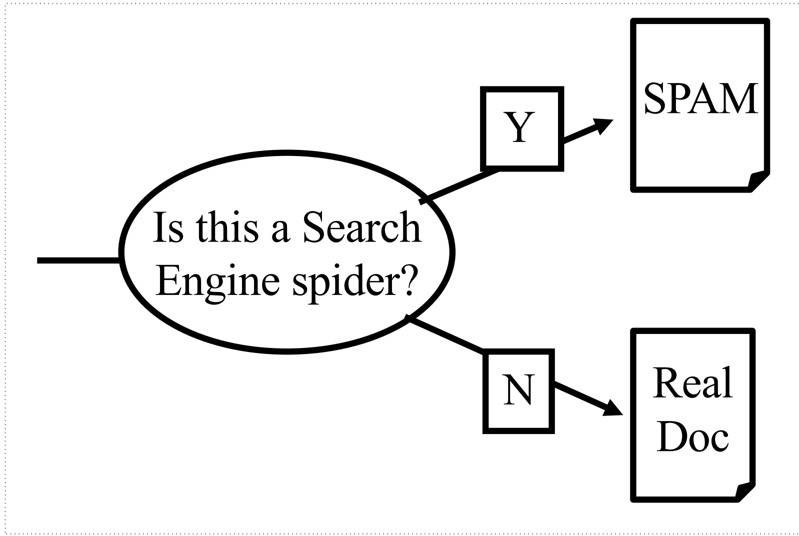


Link farm

- Create lots of links pointing to the page you want to promote
- Put these links on pages with high (or at least non-zero) PageRank
  - Newly registered domains (domain flooding)
  - A set of pages that all point to each other to boost each other's PageRank (mutual admiration society)
  - Pay somebody to put your link on their highly ranked page
  - Leave comments that include the link on blogs

Hiding techniques

- Content hiding
  - Use same color for text and page background
  - Stylesheet tricks
  - …
- Cloaking
  - Return different page to crawlers and browsers
  - Serve fake content to search engine spider
  - DNS cloaking: Switch IP address. Impersonate



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

The war against spam

- Quality signals - Prefer authoritative pages based on:
  - Votes from authors (linkage signals)
  - Votes from users (usage signals)
- Policing of URL submissions
  - Anti robot test
- Limits on meta-keywords
- Robust link analysis
  - Ignore statistically implausible linkage (or text)
  - Use link analysis to detect spammers (guilt by association)
- Spam recognition by machine learning
  - Training set based on known spam
- Family friendly filters
  - Linguistic analysis, general classification techniques, etc.
  - For images: flesh tone detectors, source text analysis, etc.
- Editorial intervention
  - Blacklists
  - Top queries audited
  - Complaints addressed
  - Suspect pattern detection