Link Analysis: PageRank and Similar Ideas

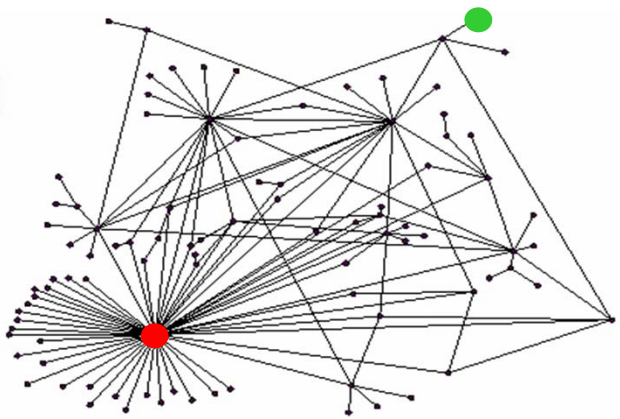
CS246: Mining Massive Datasets

Jure Leskovec, Stanford University

http://cs246.stanford.edu

Recap: PageRank

- **Rank nodes using link structure**



- **PageRank:**
  - Link voting:
    - P with importance x has n out-links, each link gets x/n votes
    - Page R's importance is the sum of the votes on its in-links
  - **Complications:** Spider traps, Dead-ends
  - **At each step, random surfer has two options:**
    - With probability β, follow a link at random
    - With prob. 1-β, jump to some page uniformly at random

Some Problems with Page Rank

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

Topic-Specific PageRank

- **Instead of generic popularity, can we measure popularity within a topic?**
- **Goal:** Evaluate Web pages not just according to their popularity, but by how close they are to a particular topic, e.g. "sports" or "history."
- **Allows search queries to be answered based on interests of the user**
  - **Example:** Query "Trojan" wants different pages depending on whether you are interested in sports or history.

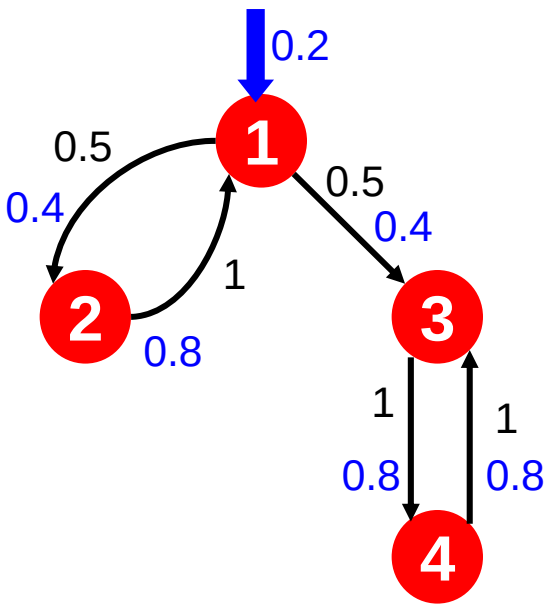
Topic-Specific PageRank

- Assume each walker has a small probability of "teleporting" at any step
- **Teleport can go to:**
  - Any page with equal probability
    - To avoid dead-end and spider-trap problems
  - A topic-specific set of "relevant" pages (teleport set)
    - For topic-sensitive PageRank.
- **Idea: Bias the random walk**
  - When walked teleports, she pick a page from a set *S*
  - *S* contains only pages that are relevant to the topic
    - E.g., Open Directory (DMOZ) pages for a given topic
  - For each teleport set *S*, we get a different vector r_S

Matrix Formulation

- **Let:**
  - $A_{ij} = \beta M_{ij} + (1-\beta) / |S|$ if $i \in S$
    βM_{ij} otherwise
  - **A** is stochastic!
- We have weighted all pages in the teleport set S equally
  - Could also assign different weights to pages!
- **Compute as for regular PageRank:**
  - Multiply by M, then add a vector
  - Maintains sparseness

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.

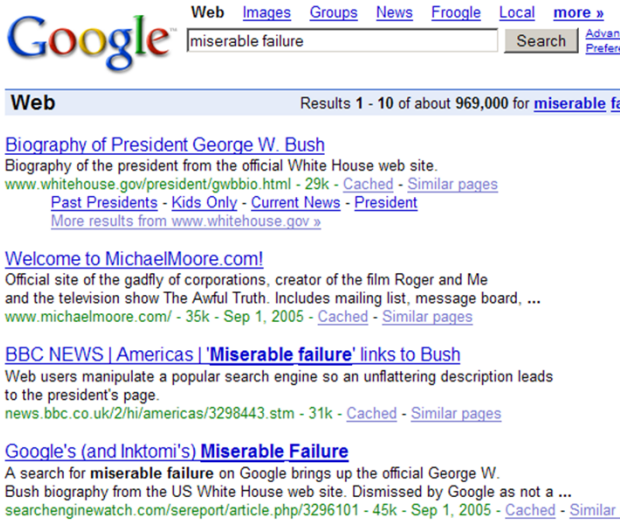
Discovering the Topic

- Create different PageRanks for different topics
  - The 16 DMOZ top-level categories:
    - arts, business, sports,…
- **Which topic ranking to use?**
  - User can pick from a menu
  - 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 bookmarks, …

Web Spam

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 Search

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

First Spammers

- As people began to use search engines to find things on the Web, those with commercial interests tried to exploit search engines to bring people to their own site – whether they wanted to be there or not.
- **Example:**
  - Shirt-seller might pretend to be about "movies."
- **Techniques for achieving high relevance/importance for a web page**

First Spammers: Term Spam

- **How do you make your page appear to be about movies?**
  - **1)** Add the word movie 1000 times to your page
  - Set text color to the background color, so only search engines would see it
  - **2)** Or, run the query "movie" on your target search engine
  - See what page came first in the listings
  - Copy it into your page, make it "invisible"
- **These and similar techniques are term spam**

Google's Solution to Term Spam

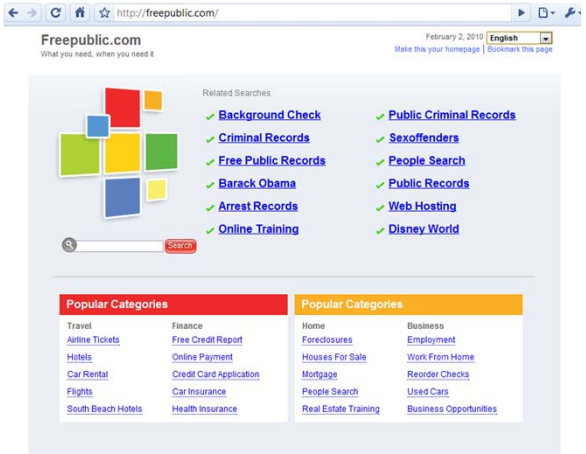
- **Believe what people say about you, rather than what you say about yourself**
  - Use words in the anchor text (words that appear underlined to represent the link) and its surrounding text
- PageRank as a tool to measure the "importance" of Web pages

Why It Works?

- Our hypothetical shirt-seller **loses**
  - Saying he is about movies doesn't help, because others don't say he is about movies
  - His page isn't very important, so it won't be ranked high for shirts or movies
- **Example:**
  - Shirt-seller creates 1000 pages, each links to his with "movie" in the anchor text
  - These pages have no links in, so they get little PageRank
  - So the shirt-seller can't beat truly important movie pages like IMDB

Google vs. Spammers: Round 2

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



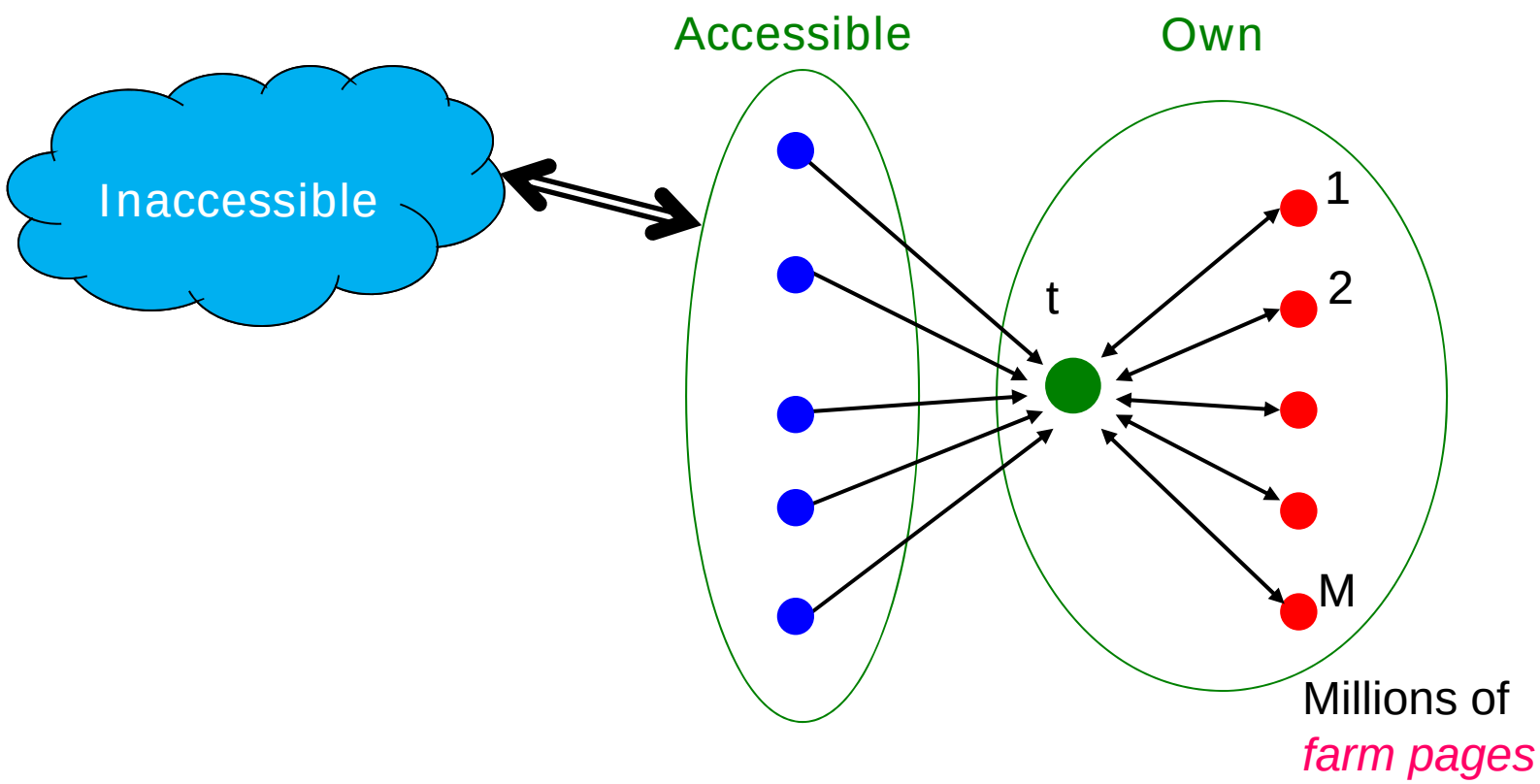
Link Spamming

- **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

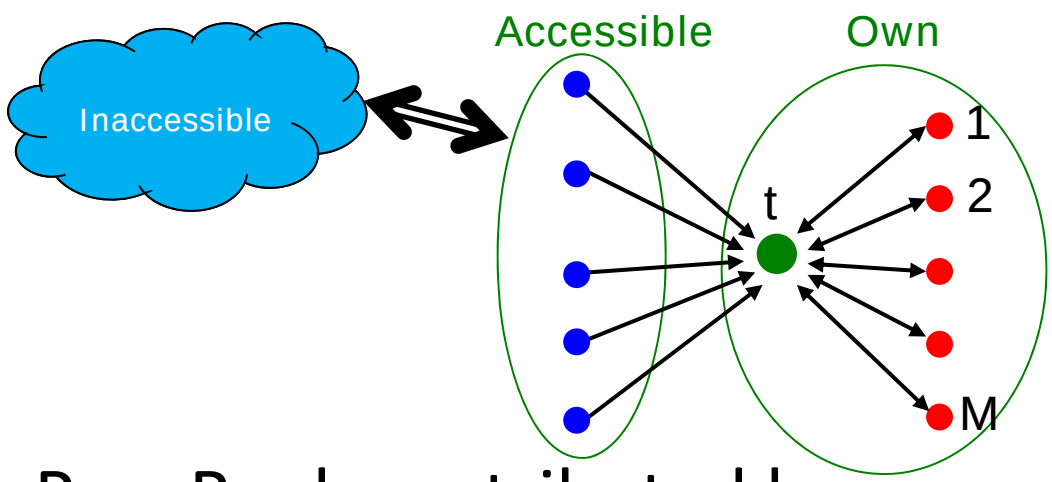
- **Spammer's goal:**
  - Maximize the PageRank of target page *t*
- **Technique:**
  - Get as many links from accessible pages as possible to target page *t*
  - Construct "link farm" to get PageRank multiplier effect

Link Farms



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

Analysis

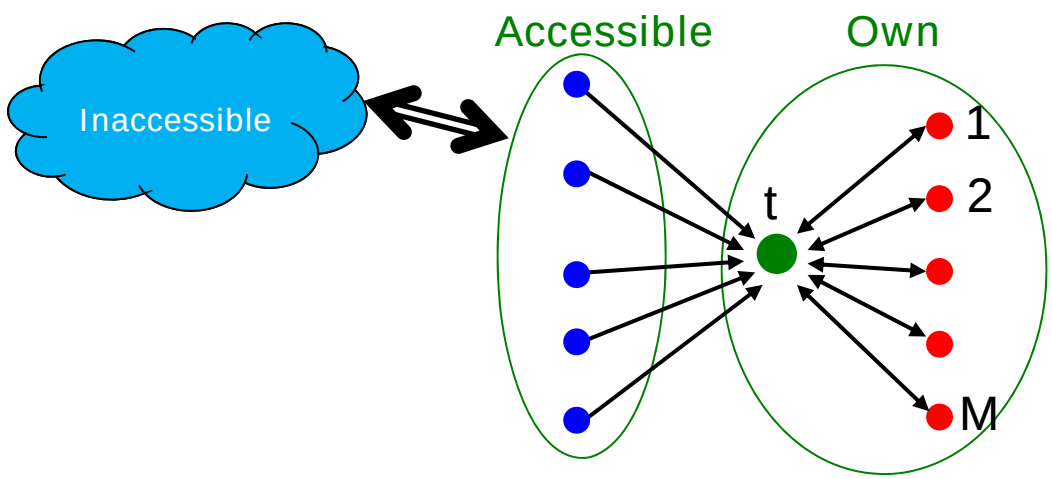


N…# pages on the web
M…# of pages spammer owns

- x: PageRank contributed by accessible pages
- y: PageRank of target page *t*
- Rank of each "farm" page $= \frac{\beta y}{M} + \frac{1-\beta}{N}$
- $y = x + \beta M \left[\frac{\beta y}{M} + \frac{1-\beta}{N}\right] + \frac{1-\beta}{N}$

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

Analysis



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

- $y = \frac{x}{1-\beta^2} + c\frac{M}{N}$ where $c = \frac{\beta}{1-\beta}$
- 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**

Combating Web Spam

Combating Spam

- **Combating term spam**
  - Analyze text using statistical methods
  - Similar to email spam filtering
  - Also useful: Detecting approximate duplicate pages
- **Combating link spam**
  - Detection and blacklisting of structures that look like spam farms
    - Leads to another war – hiding and detecting spam farms
  - **TrustRank** = topic-specific PageRank with a teleport set of "trusted" pages
    - Example: .edu domains, similar domains for non-US schools

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 we must make seed set as small as possible

Trust Propagation

- Call the subset of seed pages that are identified as "good" the "trusted pages"
- Perform a topic-sensitive PageRank with teleport set = trusted pages.
  - **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

Simple Model

- Set trust of each trusted page to 1
- 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

Why is it a good idea?

- **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

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
    - Theory is that you can't get a bad page's rank really high
- Use domains whose membership is controlled, like .edu, .mil, .gov

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:

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

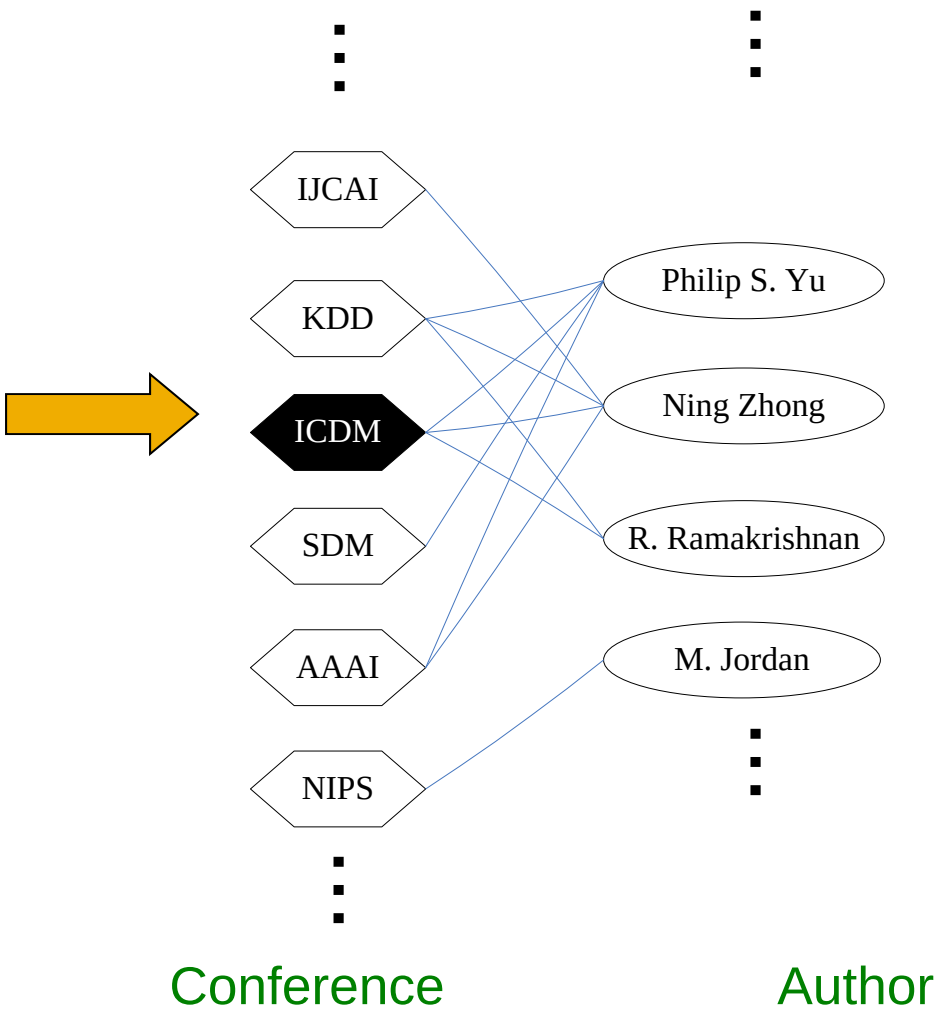
- **Spam mass of $p = r^-(p)/ r(p)$**

SimRank

SimRank: Idea

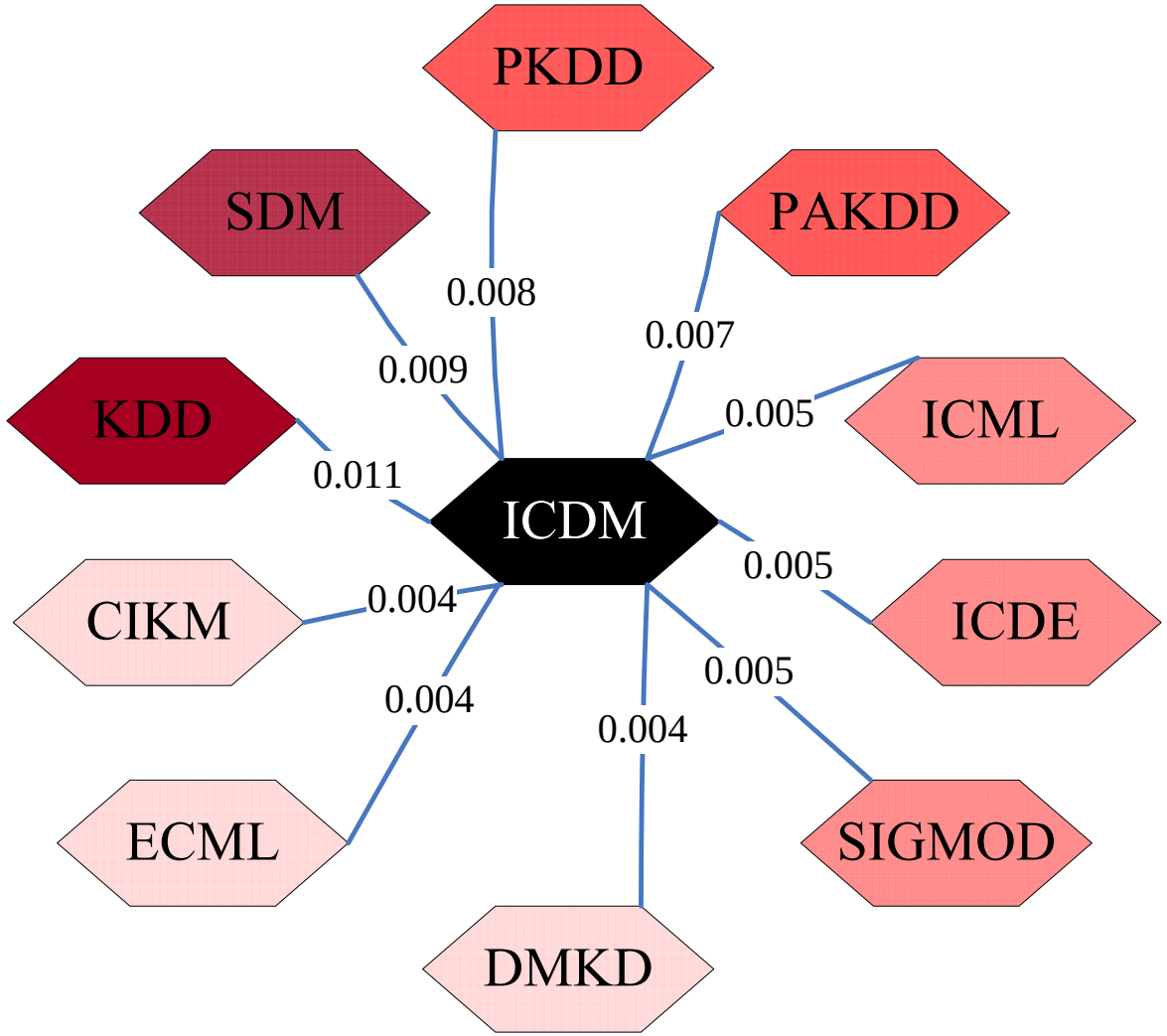
- **SimRank:** Random walks from a fixed node on *k*-partite graphs
- **Setting:** a *k*-partite graph with *k* types of nodes
  - Example: picture nodes and tag nodes.
- Do a Random-Walk with Restarts from a node *u*
  - i.e., teleport set = {*u*}.
- Resulting scores measures similarity to node *u*
- **Problem:**
  - Must be done once for each node *u*
  - Suitable for sub-Web-scale applications

SimRank: Example



Q: What is most related conference to ICDM ?

SimRank: Example



HITS: Hubs and Authorities

Hubs and Authorities

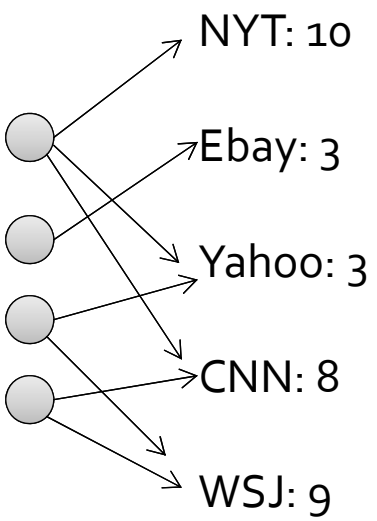
- **HITS** (Hypertext-Induced Topic Selection)
  - is a measure of importance of pages or documents, similar to PageRank
  - Proposed at around same time as PageRank ('98)
- **Goal**: Imagine we want to find good newspapers
  - Don't just find newspapers. Find "experts" – people who link in a coordinated way to good newspapers
- **Idea: Links as votes**
  - **Page is more important if it has more links**
    - In-coming links? Out-going links?

Finding newspapers

- **Hubs and Authorities**

  Each page has 2 scores:

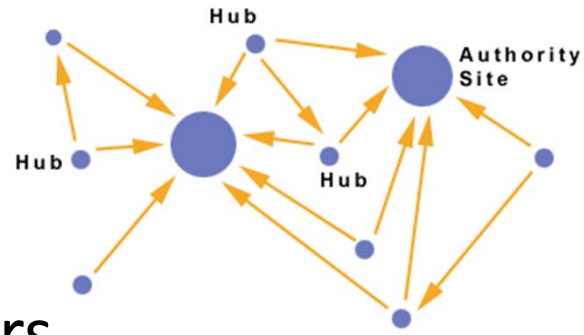
  - **Quality as an expert (hub):**
    - Total sum of votes of pages pointed to
  - **Quality as an content (authority):**
    - Total sum of votes of experts
  - Principle of repeated improvement



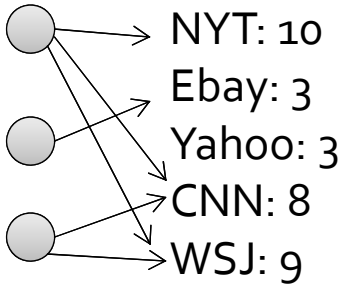
Hubs and Authorities

Interesting pages fall into two classes:

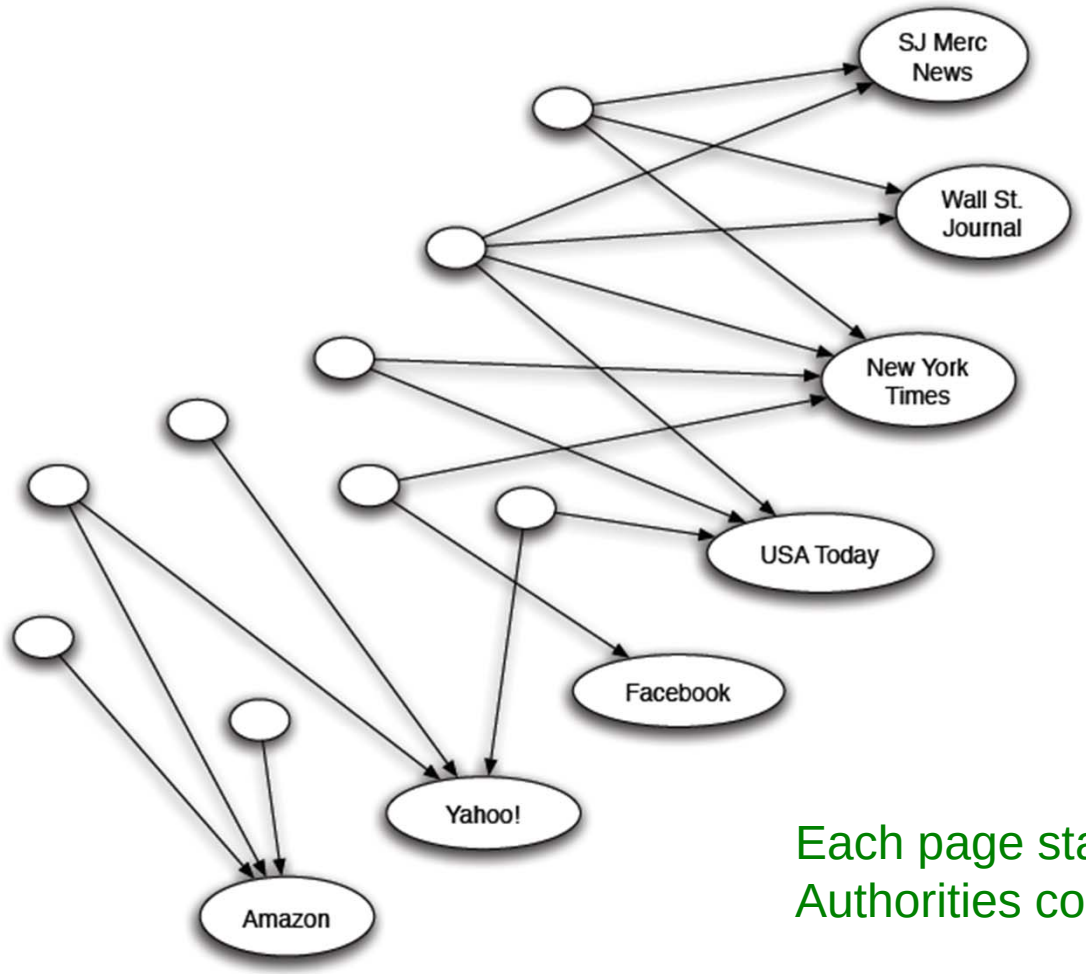
1. **Authorities** are pages containing useful information
   - Newspaper home pages
   - Course home pages
   - Home pages of auto manufacturers



2. **Hubs** are pages that link to authorities
   - List of newspapers
   - Course bulletin
   - List of US auto manufacturers



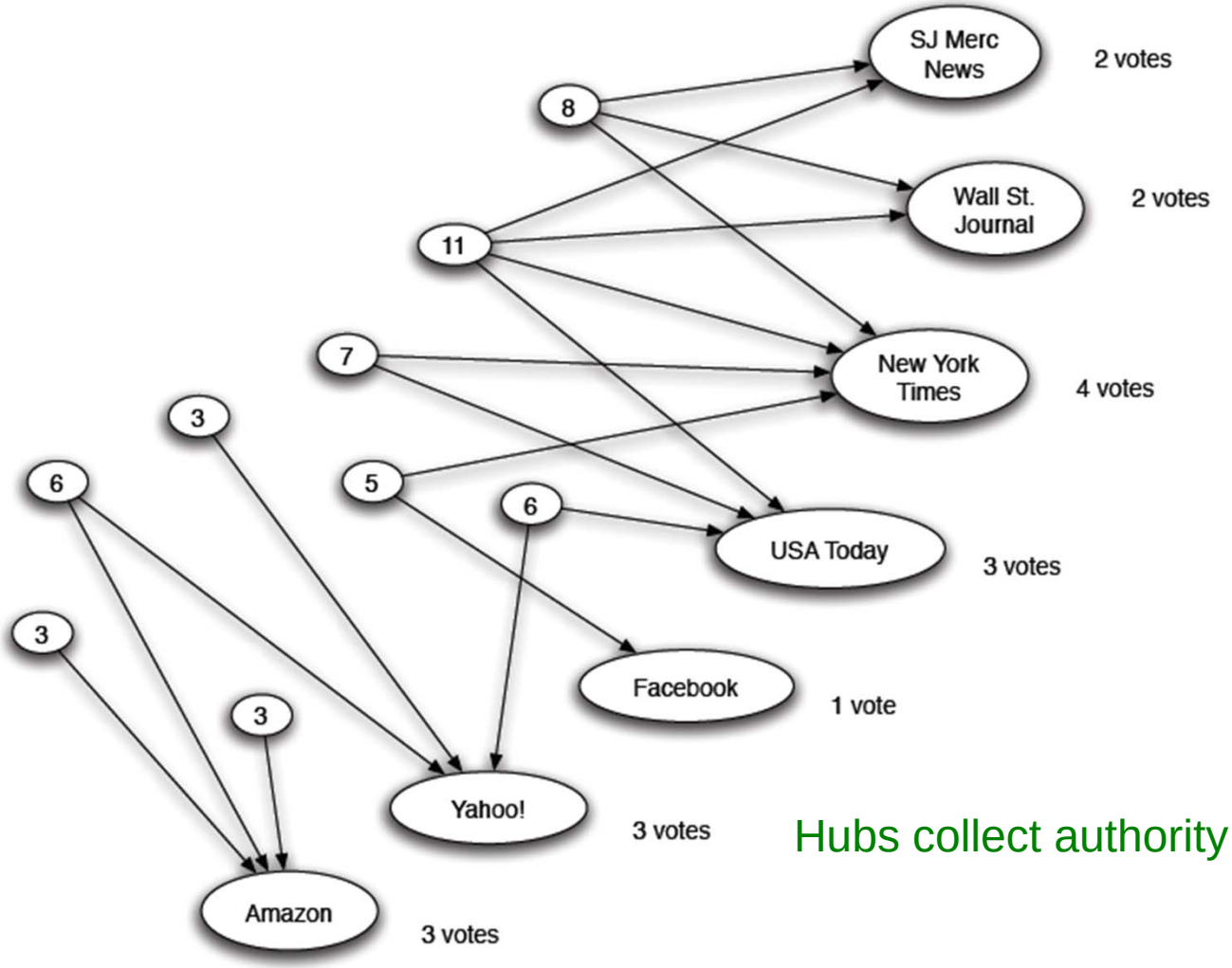
Counting in-links: Authority



Each page starts with hub score 1
Authorities collect their votes

(Note this is idealized example. In reality graph is not bipartite and each page has both the hub and authority score)

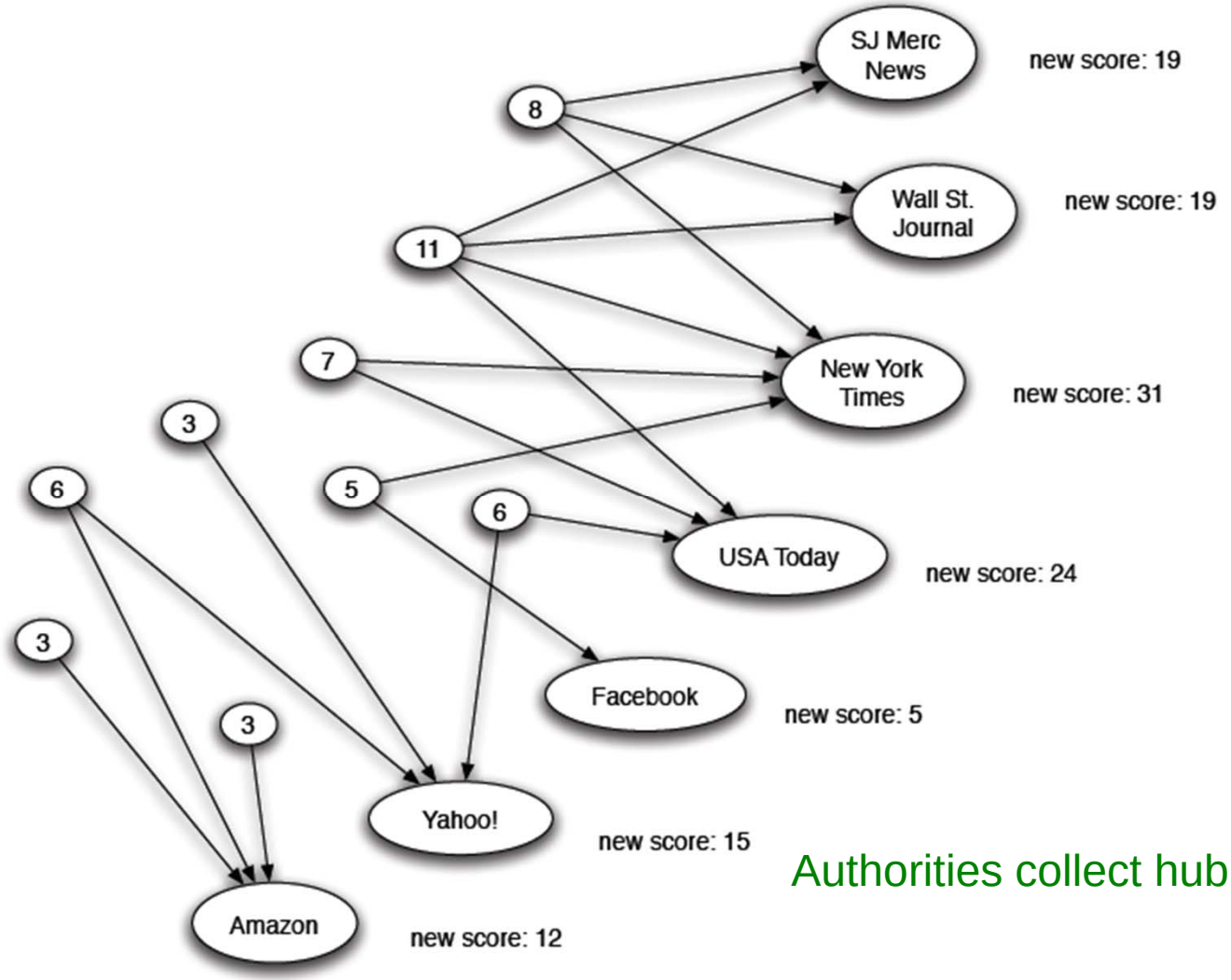
Expert Quality: Hub



Hubs collect authority scores

(Note this is idealized example. In reality graph is not bipartite and each page has both the hub and authority score)

Reweighting



Authorities collect hub scores

(Note this is idealized example. In reality graph is not bipartite and each page has both the hub and authority score)

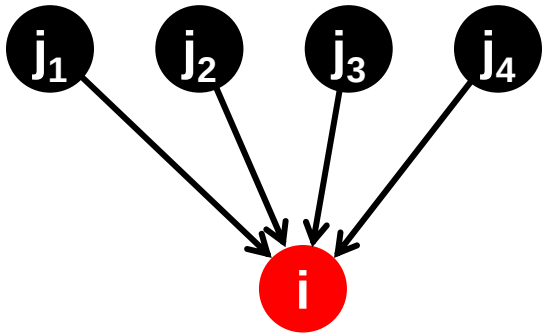
Mutually Recursive Definition

- 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

[Kleinberg '98]

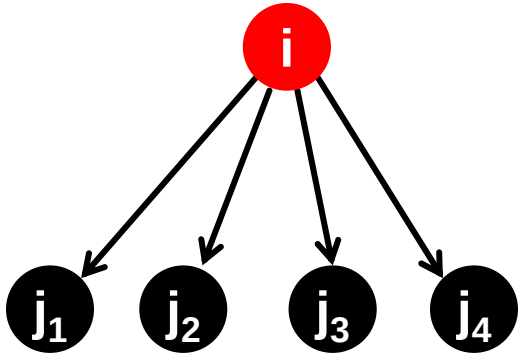
- **Each page *i* has 2 scores:**
  - Authority score: a_i
  - Hub score: h_i



$$a_i = \sum_{j \to i} h_j$$

HITS algorithm:

- Initialize: $a_j = 1,\ h_i = 1$
- Then keep iterating:
  - $\forall i$: Authority: $a_i = \sum_{j \to i} h_j$
  - $\forall i$: Hub: $h_i = \sum_{i \to j} a_j$
  - $\forall i$: normalize: $\sum_j a_j = 1, \sum_j h_j = 1$



$$h_i = \sum_{i \to j} a_j$$

Transition Matrix *A*

[Kleinberg '98]

- **HITS converges to a single stable point**
- Slightly change the notation:
  - Vector $a = (a_1 \ldots, a_n)$, $h = (h_1 \ldots, h_n)$
  - Adjacency matrix (n x n): $A_{ij}=1$ if $i \rightarrow j$
- **Then:**

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

- So: $h = A\,a$
- And likewise: $a = A^T\ h$

Hub and Authority Equations

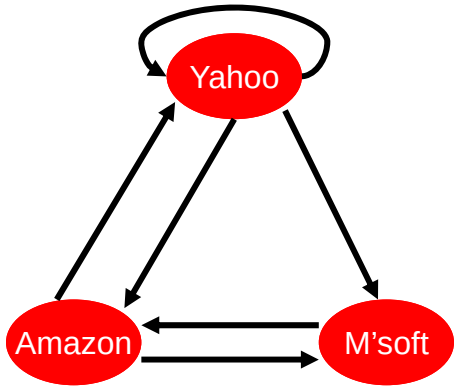
- The hub score of page *i* is proportional to the sum of the authority scores of the pages it links to: $h = \lambda A a$
  - Constant λ is a scale factor, $\lambda=1/\sum h_i$
- The authority score of page i is proportional to the sum of the hub scores of the pages it is linked from: $a = \mu A^T h$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

Iterative algorithm

- The HITS algorithm:
  - Initialize **h**, **a** to all 1's
  - Repeat:
    - $\mathbf{h} = \mathbf{A}\,\mathbf{a}$
    - Scale **h** so that its sums to 1.0
    - $\mathbf{a} = \mathbf{A}^T\,\mathbf{h}$
    - 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

- **HITS algorithm in new notation:**
  - Set: $a = h = 1^n$
  - **Repeat:**
    - $h = A\,a, \quad a = A^T h$
    - Normalize
- Then: $a = \underbrace{A^T \underbrace{(A\,a)}_{\text{new h}}}_{\text{new a}}$
- Thus, in *2k* steps:

$$a = (A^T A)^k\, a$$
$$h = (A\,A^T)^k\, h$$

a is being updated (in 2 steps):

$$A^T(A\,a) = (A^T A)\, a$$

h is updated (in 2 steps):

$$A\,(A^T h) = (A\,A^T)\, h$$

Repeated matrix powering

Existence and Uniqueness

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

$\lambda = 1/\sum h_i$
$\mu = 1/\sum a_i$

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

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

*****Notes from 2012******

Web Spam: Boosting Techniques

- **Techniques for achieving high relevance/importance for a web page**
- 1) Term spamming
  - Manipulating the text of web pages in order to appear relevant to queries
- 2) Link spamming
  - Creating link structures that boost PageRank or Hubs and Authorities scores

1) 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

Detecting Spam

- Analyze text using statistical methods e.g., Naïve Bayes classifiers
  - Similar to email spam filtering
- Also useful: detecting approximate duplicate pages