Analyzing Massive Data Sets

Summer Semester 2019

Prof. Dr. Peter Fischer

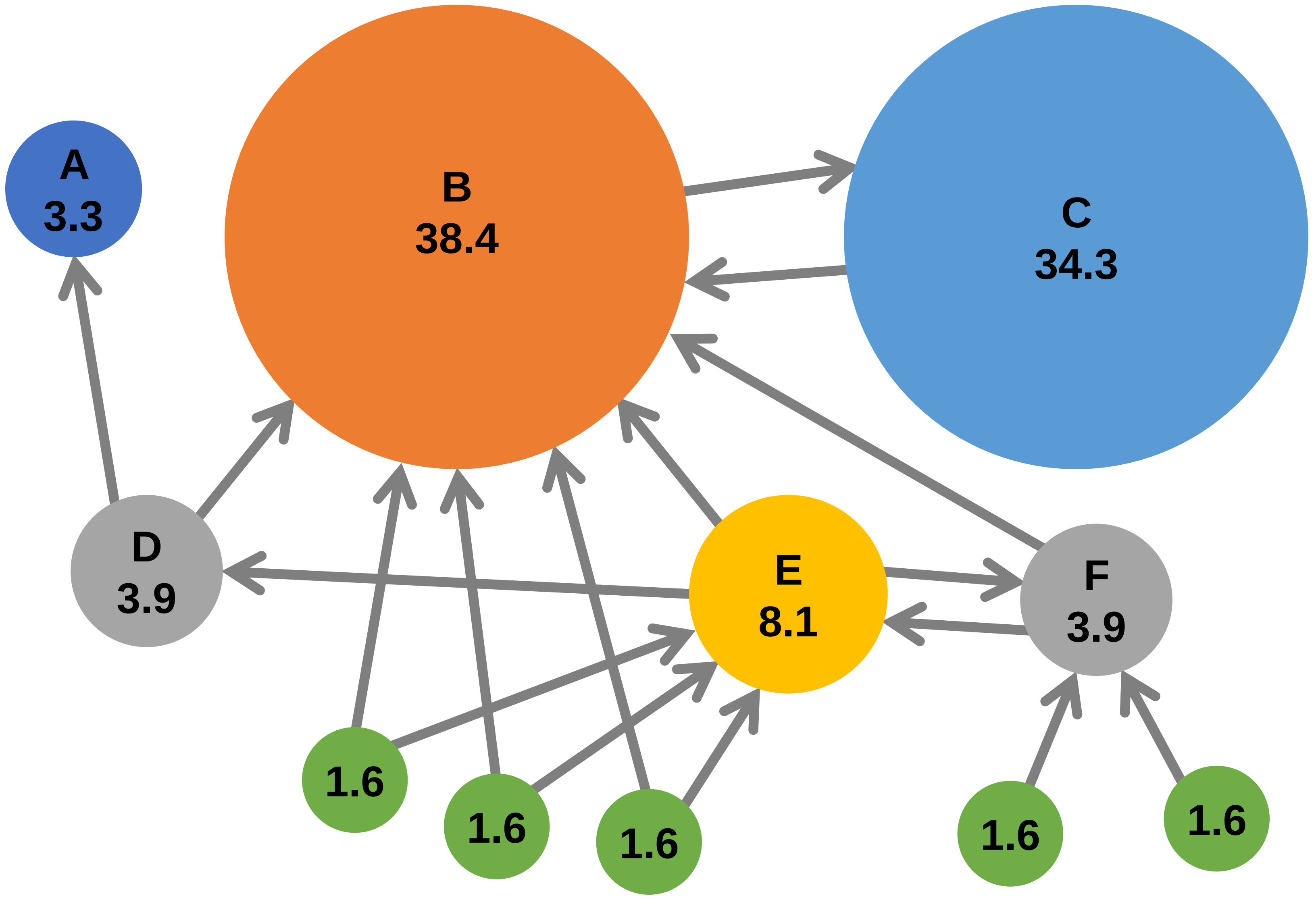
Institut für Informatik

Lehrstuhl für Datenbanken und Informationssysteme

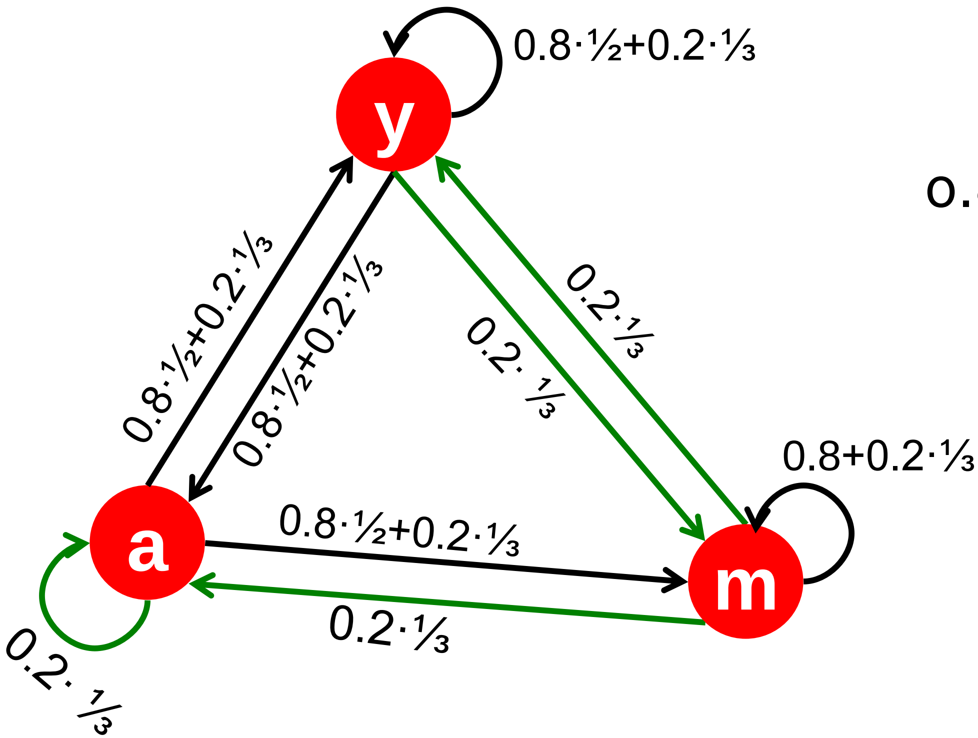
Chapter 7: Link Analysis

Part 2: Trustrank, HITS

Example: PageRank Scores



Random Teleports ($\beta = 0.8$)



$$
0.8 \overset{\textbf{M}}{\begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix}} + 0.2 \overset{[\textbf{1/N}]_{\textbf{NxN}}}{\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}}
$$

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

A

$$
\begin{array}{lcccccc}
y & & 1/3 & 0.33 & 0.24 & 0.26 & & 7/33 \\
a & = & 1/3 & 0.20 & 0.20 & 0.18 & \ldots & 5/33 \\
m & & 1/3 & 0.46 & 0.52 & 0.56 & & 21/33
\end{array}
$$

r = A r

Equivalently: $r = \beta\, M \cdot r + \left[\frac{1-\beta}{N}\right]_N$

PageRank: The Complete Algorithm

- **Input: Graph G and parameter β**
  - Directed graph G with **spider traps** and **dead ends**
  - Parameter β
- **Output: PageRank vector r**
  - **Set:** $r_j^{(0)} = \frac{1}{N}$, $t = 1$
  - **do:**
    - $\forall j$: $\boldsymbol{r'}_j^{(t)} = \sum_{i \to j} \boldsymbol{\beta} \frac{r_i^{(t-1)}}{d_i}$

      $\boldsymbol{r'}_j^{(t)} = \mathbf{0}$ if in-degree of $\boldsymbol{j}$ is **0**
    - **Now re-insert the leaked PageRank:**

      $\forall \boldsymbol{j}$: $\boldsymbol{r}_j^{(t)} = \boldsymbol{r'}_j^{(t)} + \frac{\mathbf{1-S}}{\boldsymbol{N}}$ **where:** $S = \sum_j r'^{(t)}_j$
    - $\boldsymbol{t = t + 1}$
  - **while** $\sum_j \left| r_j^{(t)} - r_j^{(t-1)} \right| > \varepsilon$

If the graph has no dead-ends then the amount of leaked PageRank is **1-β**. But since we have dead-ends the amount of leaked PageRank may be larger. We have to explicitly account for it by computing **S**.

Some Problems with PageRank

- **Measures generic popularity of a page**
  - Will ignore/miss topic-specific authorities
  - **Solution:** Topic-Specific PageRank (**next**)
- **Susceptible to Link spam**
  - Artificial link topographies created in order to boost page rank
  - **Solution:** TrustRank
- **Uses a single measure of importance**
  - Other models of importance
  - **Solution:** Hubs-and-Authorities

Topic-Specific PageRank

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 “Jaguar” wants different pages depending on whether you are interested in animals, cars, Mac OS, or game console history. Wikipedia lists > 50 articles in 9 topic areas
- Naïve solution: PageRank for every user
  - Does not scale!

Topic-Specific PageRank

- Random walker has a small probability of teleporting at any step
- **Teleport can go to:**
  - **Standard PageRank: Any page with equal probability**
    - To avoid dead-end and spider-trap problems
  - **Topic Specific PageRank: A topic-specific set of “relevant” pages (teleport set)**
- **Idea: Bias the random walk**
  - When walker 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/query
  - For each teleport set ***S***, we get a different vector $\boldsymbol{r_S}$
- **Why no changes for link traversal?**

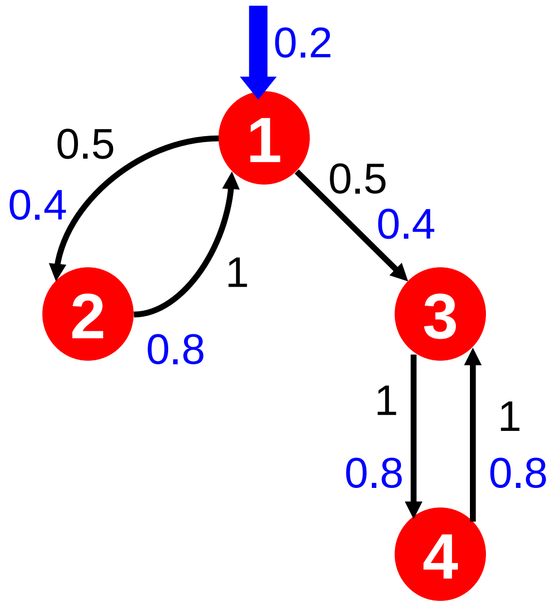
Matrix Formulation

- **To make this work all we need is to update the teleportation part of the PageRank formulation:**

$$\boldsymbol{A_{ij}} = \begin{cases} \boldsymbol{\beta\, M_{ij} + (1-\beta)/|S|} & \textbf{if } \boldsymbol{i \in S} \\ \boldsymbol{\beta\, M_{ij} + 0} & \textbf{otherwise} \end{cases}$$

  - ***A*** is stochastic!
- We 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: Topic-Specific PageRank



Suppose $S = \{1\}$, $\beta = 0.8$

Node	Iteration 0	1	2	…	stable
1	0.25	0.4	0.28		0.294
2	0.25	0.1	0.16		0.118
3	0.25	0.3	0.32		0.327
4	0.25	0.2	0.24		0.261

S={1}, β=0.90:
r=[0.17, 0.07, 0.40, 0.36]
S={1} , β=0.8:
r=[0.29, 0.11, 0.32, 0.26]
S={1}, β=0.70:
r=[0.39, 0.14, 0.27, 0.19]

S={1,2,3,4}, β=0.8:
r=[0.13, 0.10, 0.39, 0.36]
S={1,2,3} , β=0.8:
r=[0.17, 0.13, 0.38, 0.30]
S={1,2} , β=0.8:
r=[0.26, 0.20, 0.29, 0.23]
S={1} , β=0.8:
r=[0.29, 0.12, 0.33, 0.26]

Discovering the Topic Vector S

- **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, …

PageRank: Summary

- **“Normal” PageRank:**
  - Teleports uniformly at random to any node
  - All nodes have the same probability of surfer landing there: **S =** [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
- **Topic-Specific PageRank** also known as **Personalized PageRank:**
  - Teleports to a topic specific set of pages
  - Nodes can have different probabilities of surfer landing there: **S =** [0.1, 0, 0, 0.2, 0, 0, 0.5, 0, 0, 0.2]
- **Random Walk with Restarts:**
  - Topic-Specific PageRank where teleport is always to the same node. S=[0, 0, 0, 0, **1**, 0, 0, 0, 0, 0]

TrustRank:
Combating the 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

Term Spam and Google's Solution

- **Term Spam (e.g., about movies)**
  - **(1)** Add the word movie 1,000 times to your page
  - Set text color to the background color, so only search engines 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"

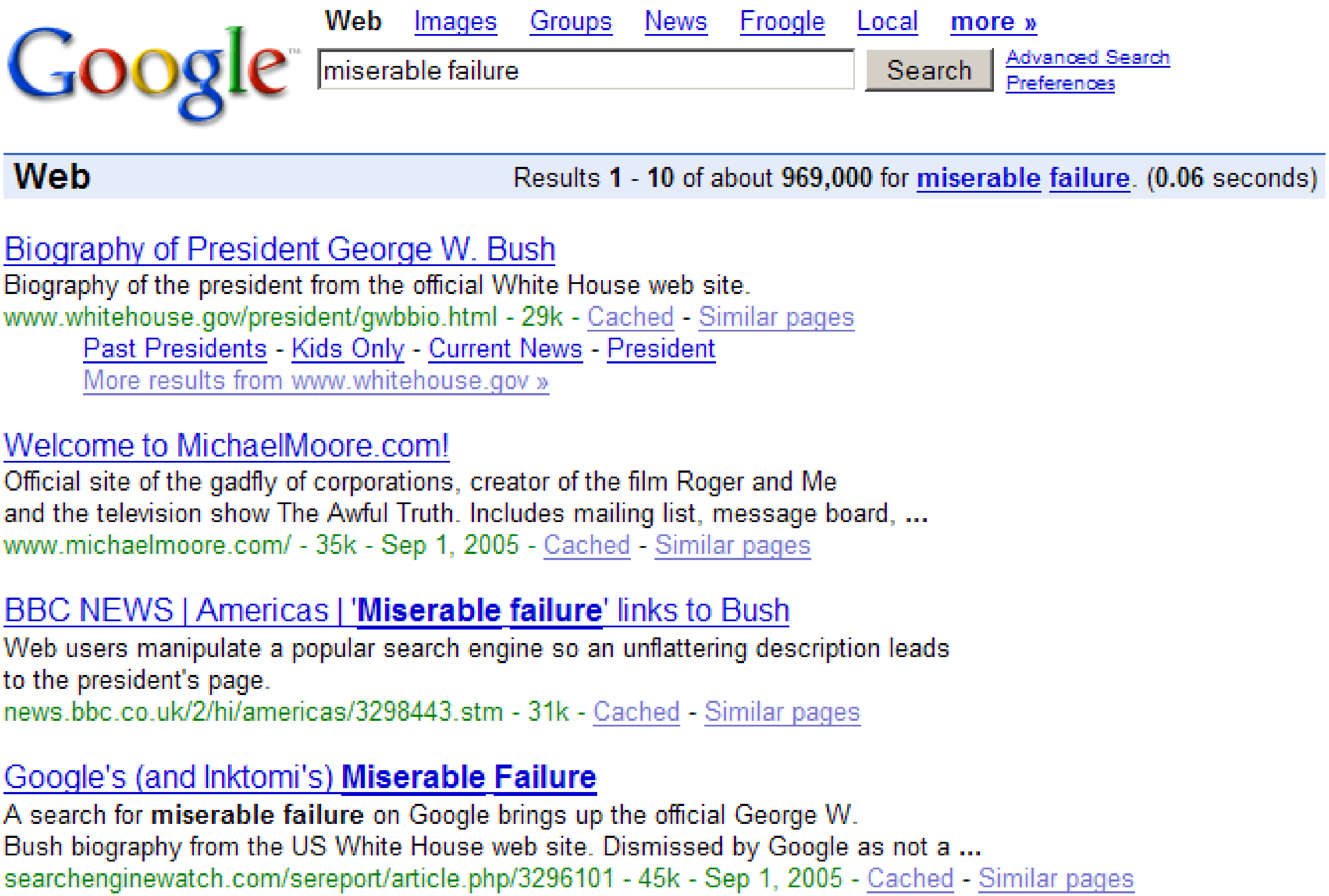
- **Idea:Believe what people say about you, rather than what you say about yourself**
  - Use words in the anchor text (words that appear underlined to represent the link) and its surrounding text

- PageRank as a tool to measure the "importance" of Web pages

Why Does It Work?

- **A shirt-seller stuffing the pages with "movies" 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 1,000 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

Why it does not work?



SPAM FARMING

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

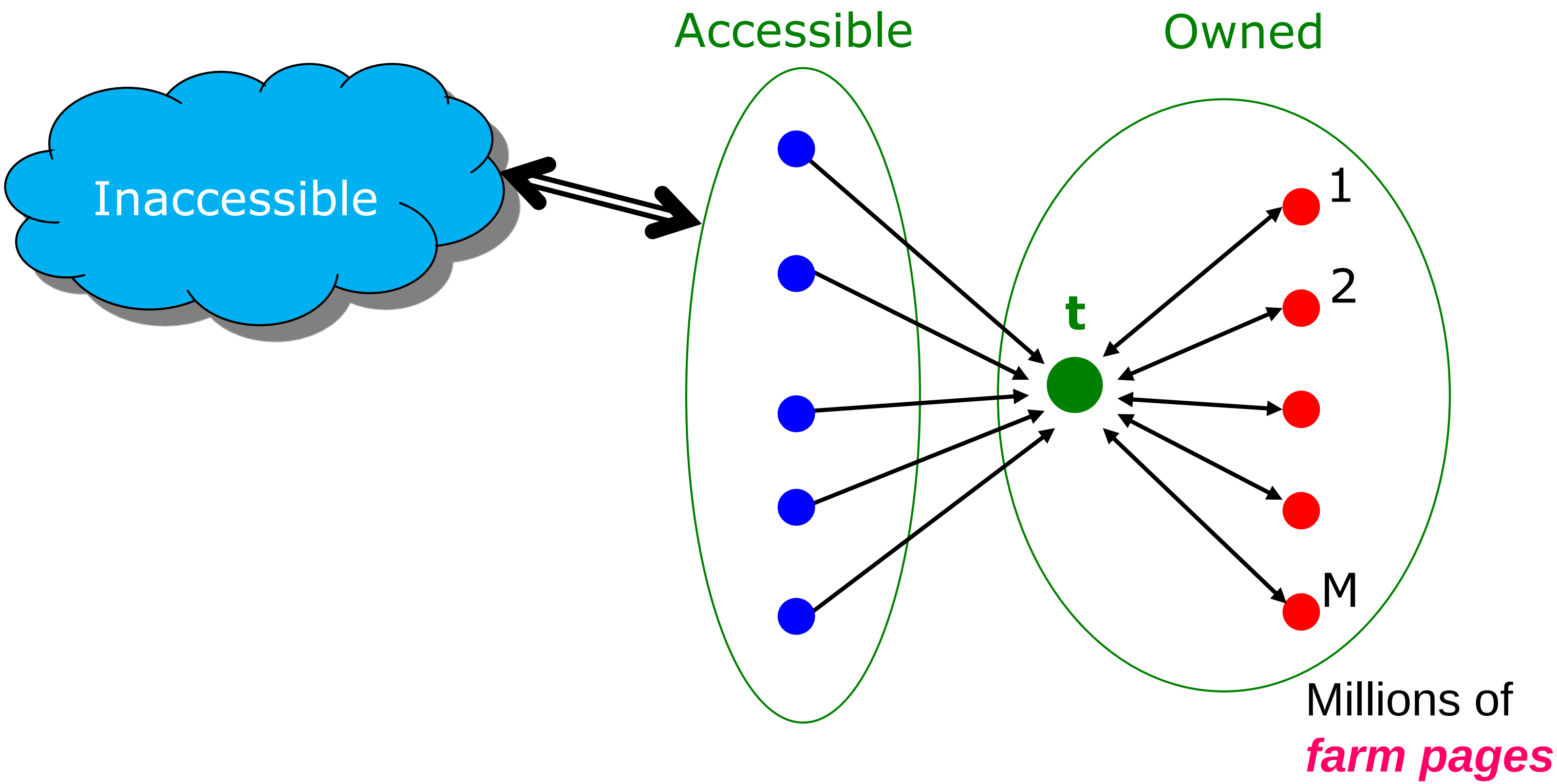
- **Three kinds of web pages from a spammer's point of view**
  - **Inaccessible pages**
  - **Accessible pages**
    - e.g., blog comments pages
    - spammer can post links to his pages
  - **Owned pages**
    - Completely controlled by spammer
    - May span multiple domain names

Link Farms

- **Spammer's goal:**
  - Maximize the PageRank of target page ***t***

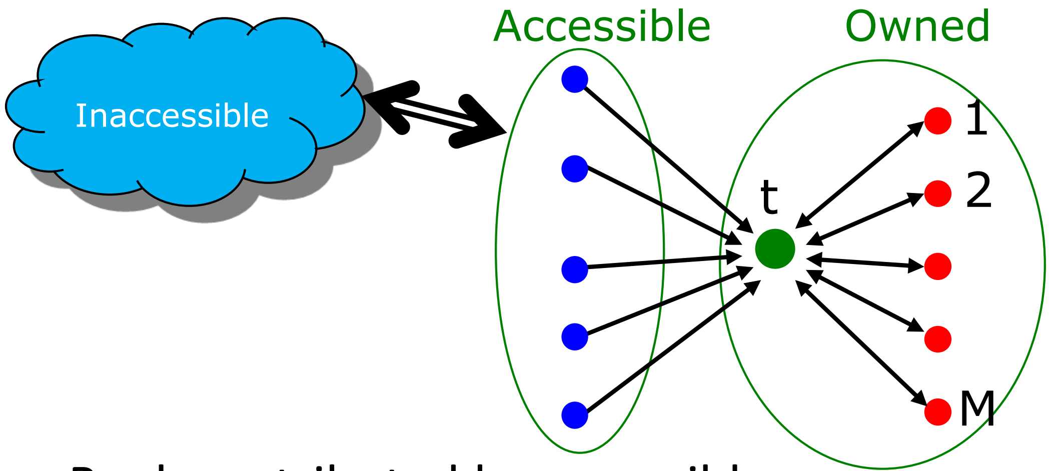
- **Technique:**
  - Get as many links from accessible pages as possible to target page ***t***
  - Construct "link farm" to get PageRank multiplier effect

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

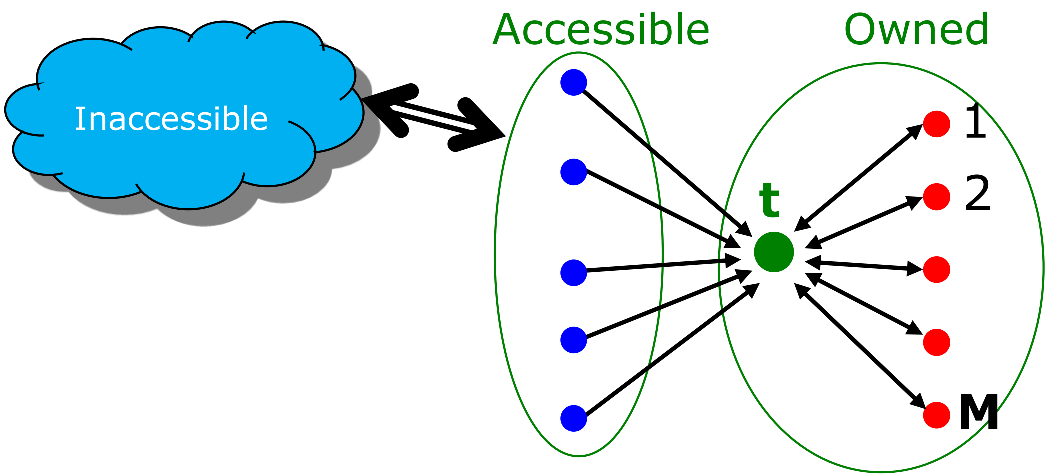
- **x**: PageRank contributed by accessible pages
- **y**: PageRank of target page ***t***
- Rank of each "farm" page $= \frac{\beta \boldsymbol{y}}{M} + \frac{1-\beta}{N}$
- $\boldsymbol{y} = x + \beta M \left[\frac{\beta y}{M} + \frac{1-\beta}{N}\right] + \frac{1-\beta}{N}$

$$= x + \beta^2 y + \frac{\beta(1-\beta)M}{N} + \boxed{\frac{1-\beta}{N}}$$

Very small; ignore
Now we solve for ***y***

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

TrustRank:
Combating the 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**) to 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**

- **Solution 1:** **Use a threshold value and mark all pages below the trust threshold as spam**

Simple Model: Trust Propagation

- **Set trust of each trusted page to 1**
- Suppose trust of page $\boldsymbol{p}$ is $\boldsymbol{t_p}$
  - Page $\boldsymbol{p}$ has a set of out-links $\boldsymbol{o_p}$
- For each $\boldsymbol{q} \in \boldsymbol{o_p}$, $\boldsymbol{p}$ **confers the trust** to $\boldsymbol{q}$
  - $\boldsymbol{\beta\, t_p / |o_p|}$ for $0 < \beta < 1$
- **Trust is additive**
  - Trust of $\boldsymbol{p}$ is the sum of the trust conferred on $\boldsymbol{p}$ 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 the distance in the graph

- **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 (high precision)

  - Must ensure every **good page** gets adequate trust rank, so need make all good pages reachable from seed set by short paths (higher recall)
- **Approaches to pick a seed of k pages**
  - **(1) PageRank:**
    - Pick the top ***k*** pages by PageRank
    - Theory is that you can't get a bad page's rank really high
  - **(2) Use trusted domains** whose membership is controlled, like .edu, .mil, .gov

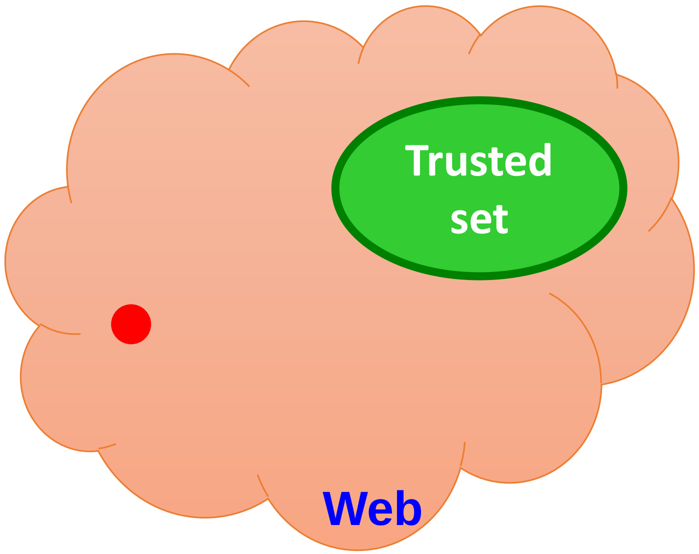
Spam Mass

- 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

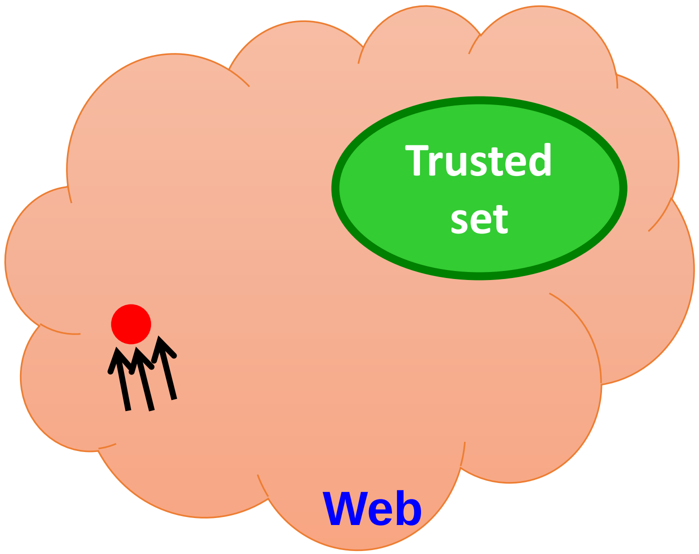
Solution 2:

- $\boldsymbol{r_p}$ = PageRank of page $\boldsymbol{p}$
- $\boldsymbol{r_p^+}$ = PageRank of $\boldsymbol{p}$ with teleport into **trusted** pages only

- **Then:** What fraction of a page's PageRank comes from **spam** pages?

$$\boldsymbol{r_p^- = r_p - r_p^+}$$

- **Spam mass of** $\boldsymbol{p} = \dfrac{\boldsymbol{r_p^-}}{\boldsymbol{r_p}}$
  - Pages with high spam mass are spam.



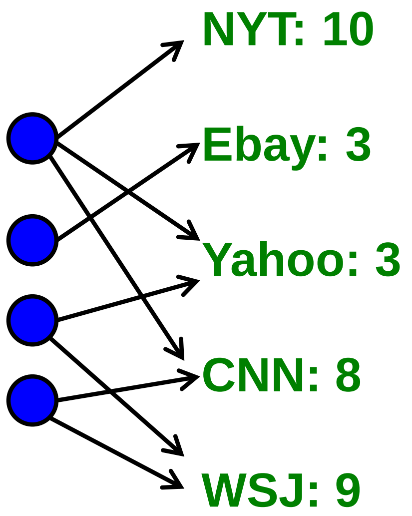
HITS: Hubs and Authorities

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**: Say 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 authorities pointed to
  - **Quality as a content (authority):**
    - Total sum of votes coming from experts



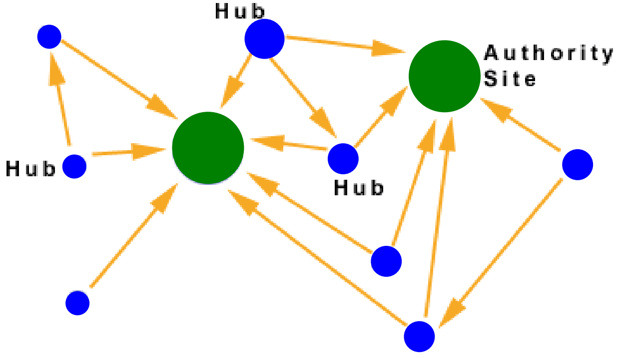
- **Principle of repeated improvement**

Hubs and Authorities

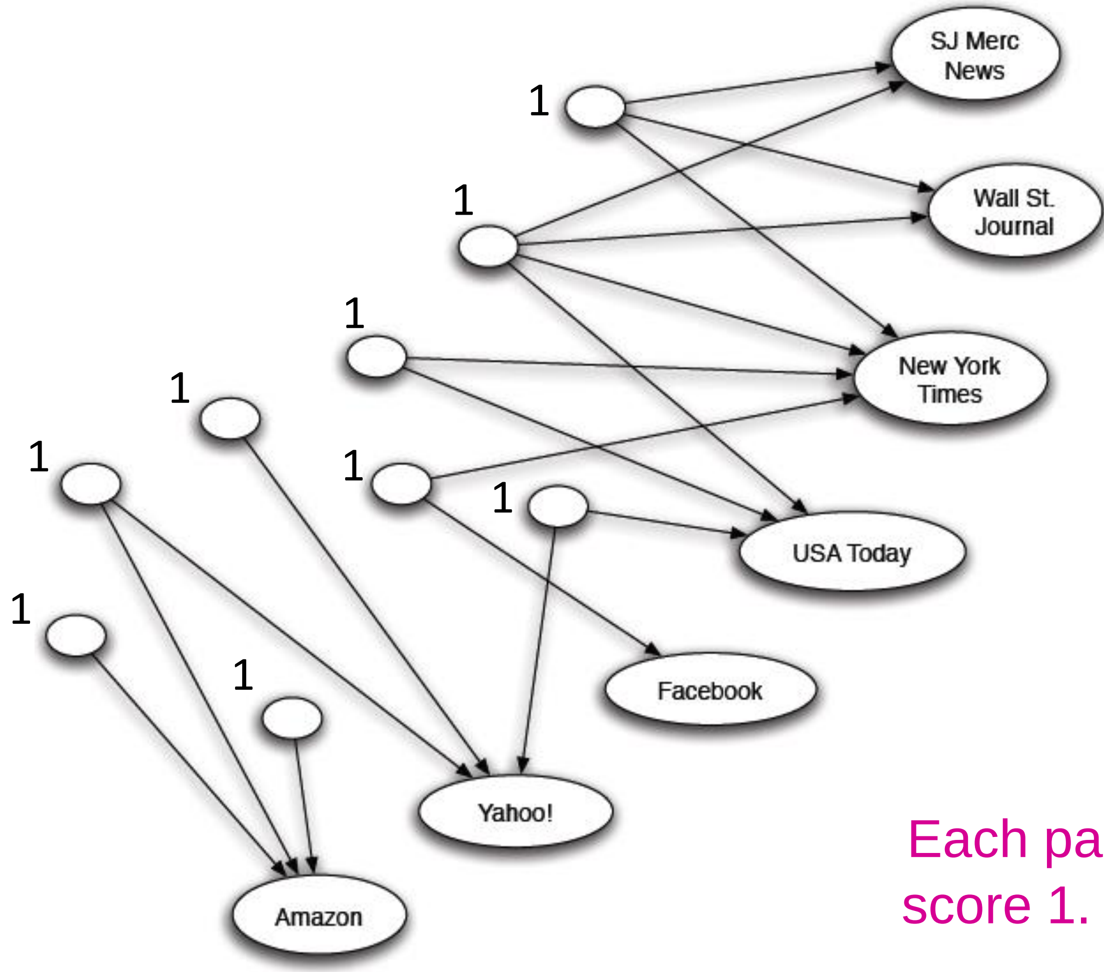
Interesting pages fall into two classes:

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



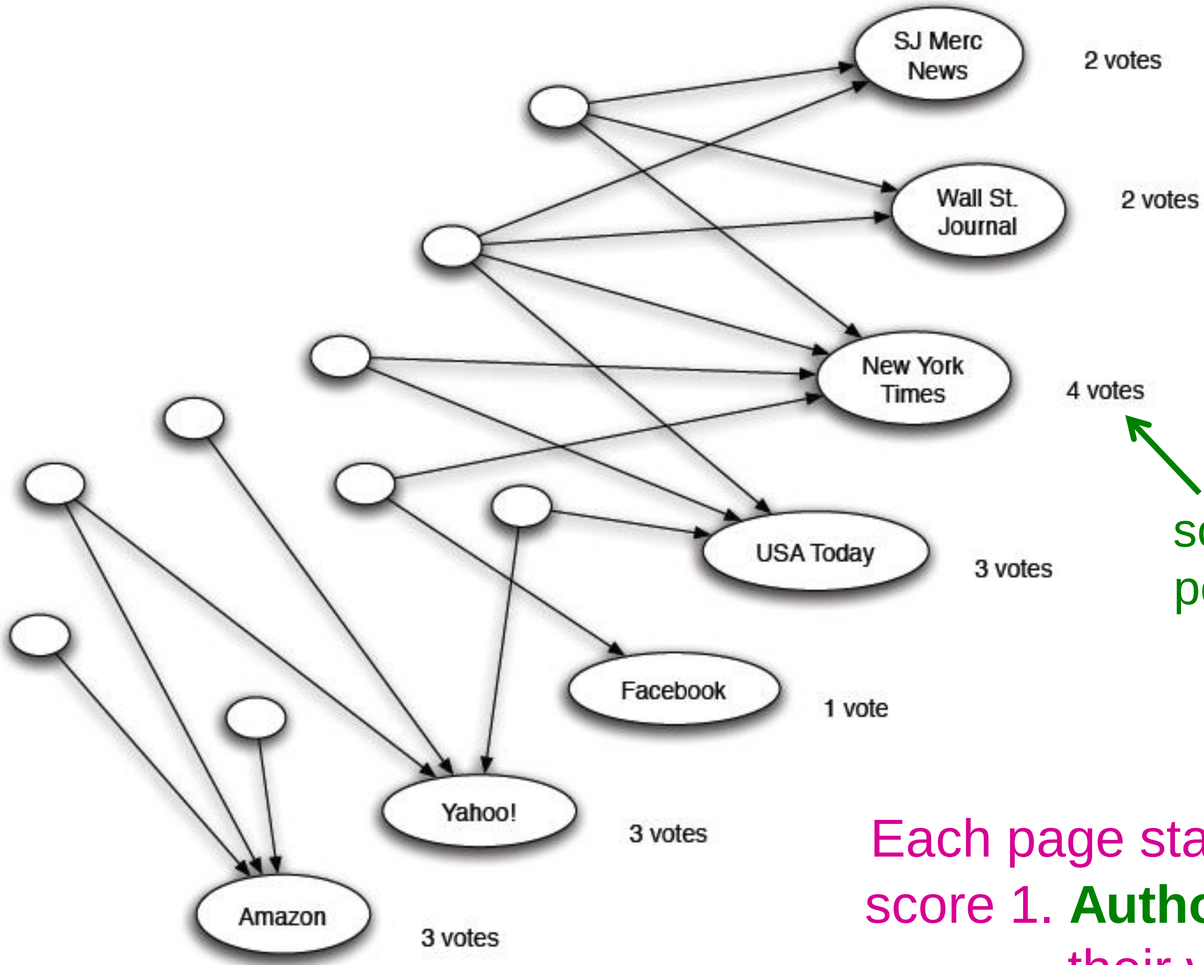
Counting in-links: Authority



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)

Counting in-links: Authority



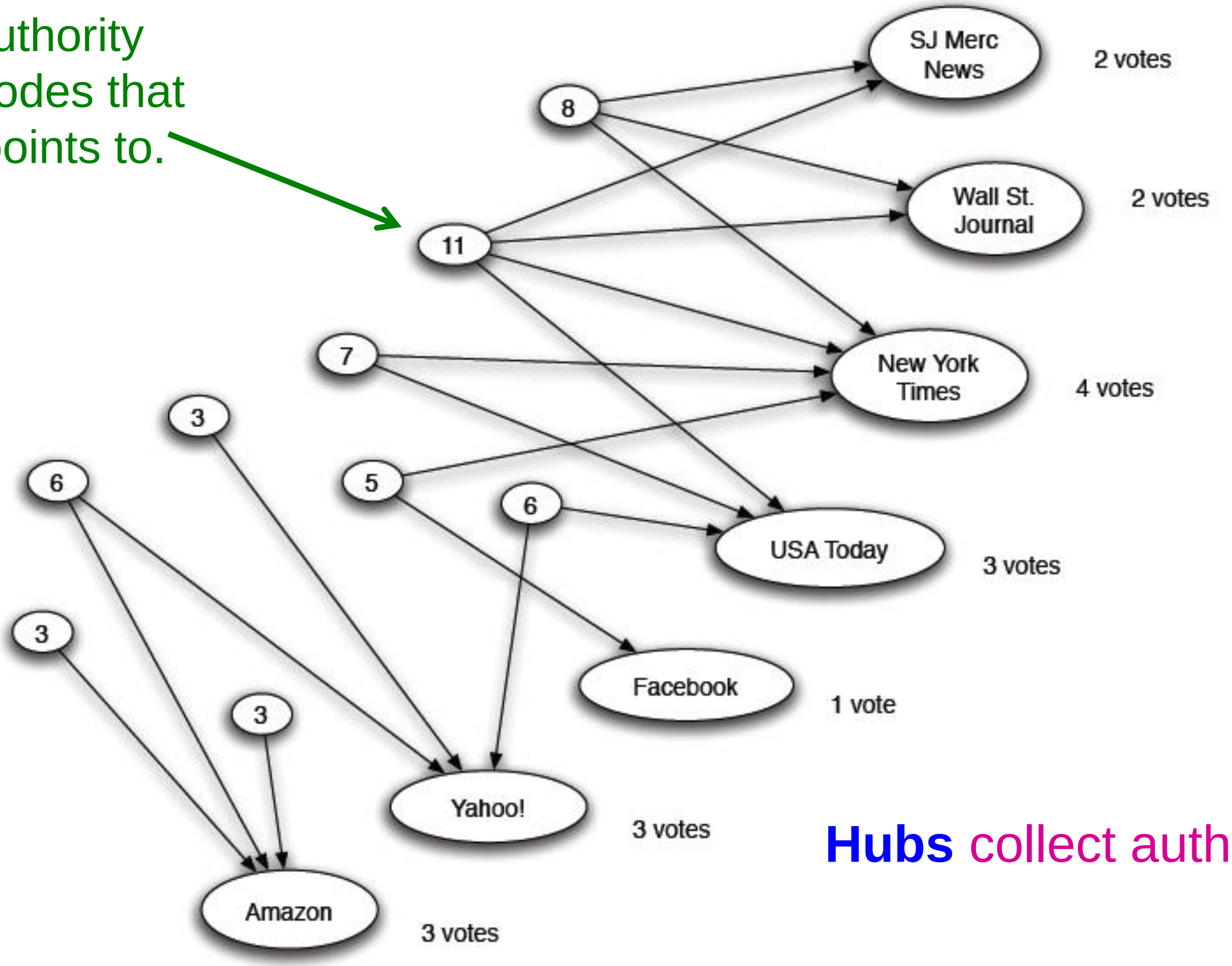
Sum of **hub** scores of nodes pointing to NYT.

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

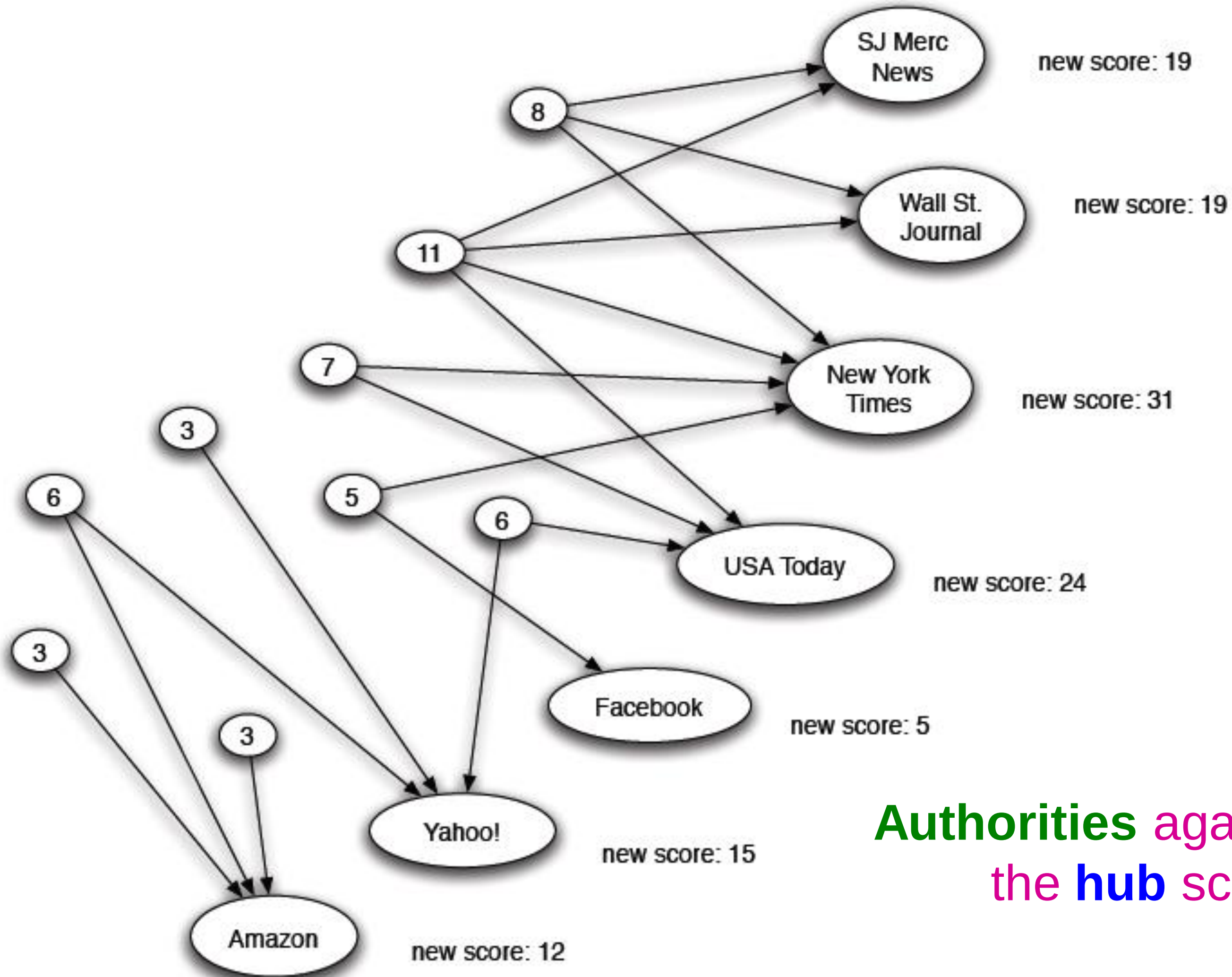
Sum of authority scores of nodes that the node points to.



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 again collect the **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 $\boldsymbol{h}$ and $\boldsymbol{a}$

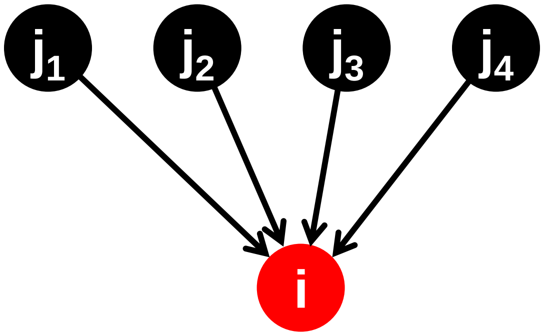
Hubs and Authorities

- **Each page i has 2 scores:**
  - Authority score: $\boldsymbol{a_i}$
  - Hub score: $\boldsymbol{h_i}$

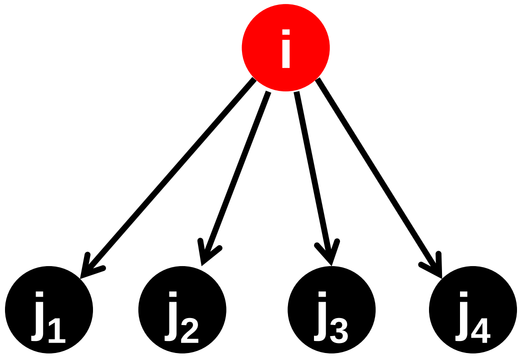
HITS algorithm:

- Initialize: $a_j^{(0)} = 1/\sqrt{\mathrm{N}}, \ \ \mathrm{h}_\mathrm{j}^{(0)} = 1/\sqrt{\mathrm{N}}$
- Then keep iterating until **convergence**:
  - $\forall \boldsymbol{i}$: Authority: $a_i^{(t+1)} = \sum_{\boldsymbol{j \to i}} h_j^{(t)}$
  - $\forall \boldsymbol{i}$: Hub: $h_i^{(t+1)} = \sum_{\boldsymbol{i \to j}} a_j^{(t)}$
  - $\forall \boldsymbol{i}$: Normalize:

$$\sum_i \left(a_i^{(t+1)}\right)^2 = 1, \sum_j \left(h_j^{(t+1)}\right)^2 = 1$$



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



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

Hubs and Authorities

- **HITS converges to a single stable point**
- **Notation:**
  - Vector $\boldsymbol{a} = (a_1 \ldots, a_n)$, $\boldsymbol{h} = (h_1 \ldots, h_n)$
  - Adjacency matrix $\boldsymbol{A}$ (NxN): $\boldsymbol{A_{ij}} = 1$ if $i \rightarrow j$, 0 otherwise
- **Then $\boldsymbol{h_i = \sum_{i \rightarrow j} a_j}$**
  can be rewritten as $\boldsymbol{h_i = \sum_j A_{ij} \cdot a_j}$
  So: $\boldsymbol{h = A \cdot a}$
- **Similarly, $\boldsymbol{a_i = \sum_{j \rightarrow i} h_j}$**
  can be rewritten as $\boldsymbol{a_i = \sum_j A_{ji} \cdot h_j = A^T \cdot h}$

Hubs and Authorities

- **HITS algorithm in vector notation:**
  - Set: $\boldsymbol{a_i} = \boldsymbol{h_i} = \frac{\mathbf{1}}{\sqrt{\boldsymbol{n}}}$

  Repeat until convergence:
  - $\boldsymbol{h} = \boldsymbol{A} \cdot \boldsymbol{a}$
  - $\boldsymbol{a} = \boldsymbol{A^T} \cdot \boldsymbol{h}$
  - Normalize $\boldsymbol{a}$ and $\boldsymbol{h}$
- **Then:** $\boldsymbol{a} = \boldsymbol{A^T} \cdot \underbrace{(\underbrace{\boldsymbol{A} \cdot \boldsymbol{a}}_{\text{new } h})}_{\text{new } a}$

Convergence criterion:

$$\sum_i \left(h_i^{(t)} - h_i^{(t-1)}\right)^2 < \varepsilon$$

$$\sum_i \left(a_i^{(t)} - a_i^{(t-1)}\right)^2 < \varepsilon$$

a **is updated (in 2 steps):**

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

h **is updated (in 2 steps):**

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

Repeated matrix powering

Existence and Uniqueness

- $\boldsymbol{h = \lambda A a}$
- $\boldsymbol{a = \mu A^T h}$
- $\boldsymbol{h = \lambda \mu A A^T h}$
- $\boldsymbol{a = \lambda \mu A^T A a}$

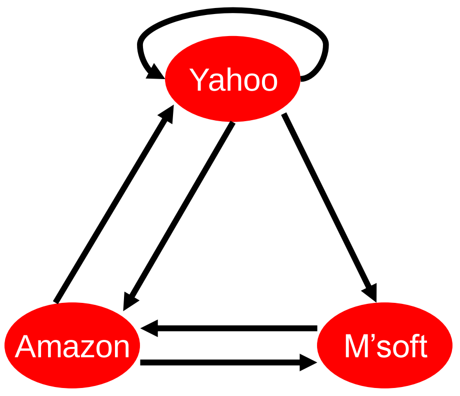
$$\lambda = \frac{1}{\sqrt{\sum h_i^2}}$$

$$\mu = \frac{1}{\sqrt{\sum a_i^2}}$$

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

Example of HITS

$$A = \begin{array}{c|ccc|c} & Y & A & M & \\ \hline & 1 & 1 & 1 & Y \\ & 1 & 0 & 1 & A \\ & 0 & 1 & 0 & M \end{array} \qquad A^T = \begin{array}{c|ccc|c} & Y & A & M & \\ \hline & 1 & 1 & 0 & Y \\ & 1 & 0 & 1 & A \\ & 1 & 1 & 0 & M \end{array}$$



h(yahoo)	=	.58	.80	.80	.79	...	.788
h(amazon)	=	.58	.53	.53	.57	...	.577
h(m'soft)	=	.58	.27	.27	.23	...	.211
a(yahoo)	=	.58	.58	.62	.62	...	.628
a(amazon)	=	.58	.58	.49	.49	...	.459
a(m'soft)	=	.58	.58	.62	.62	...	.628

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**