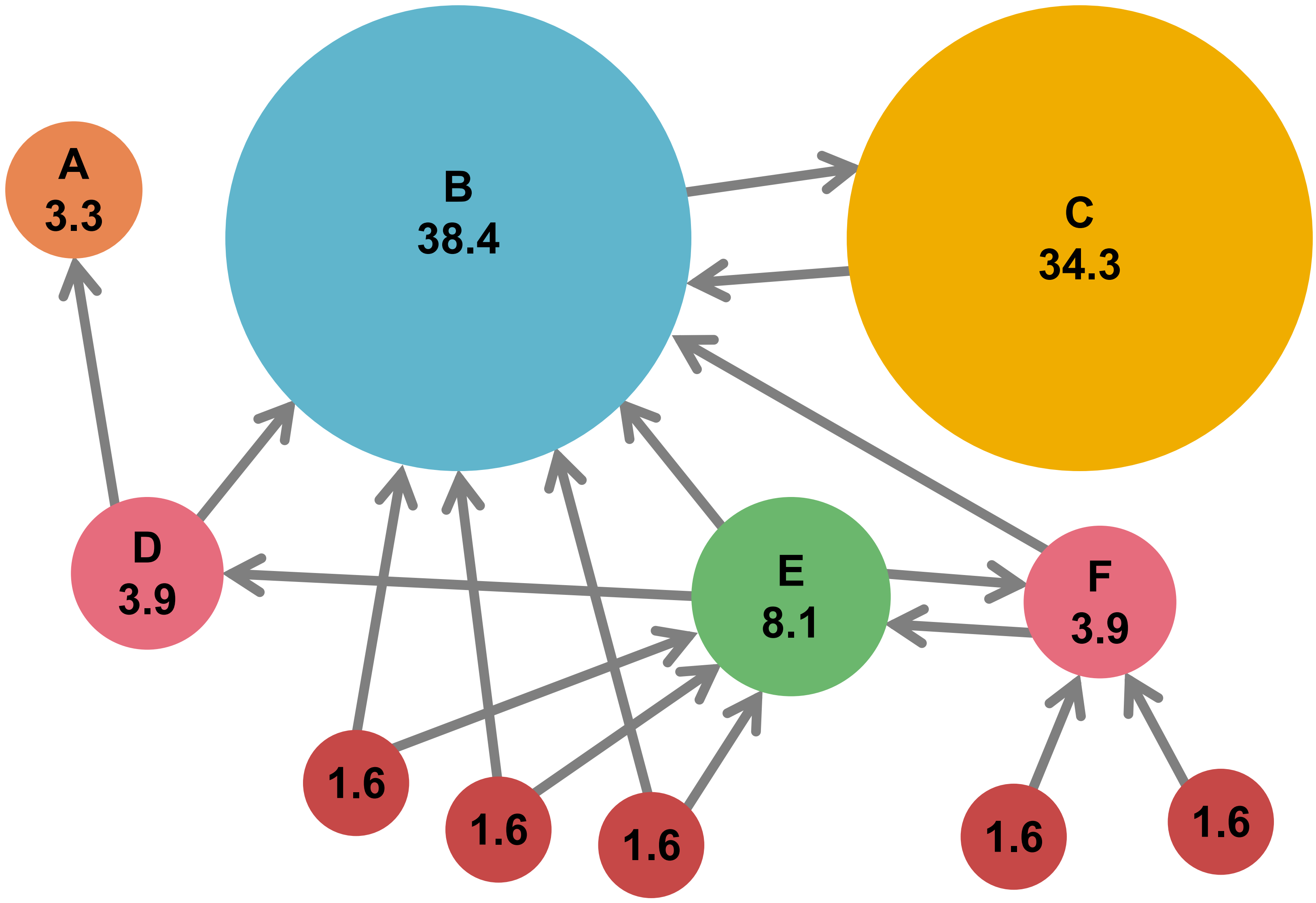
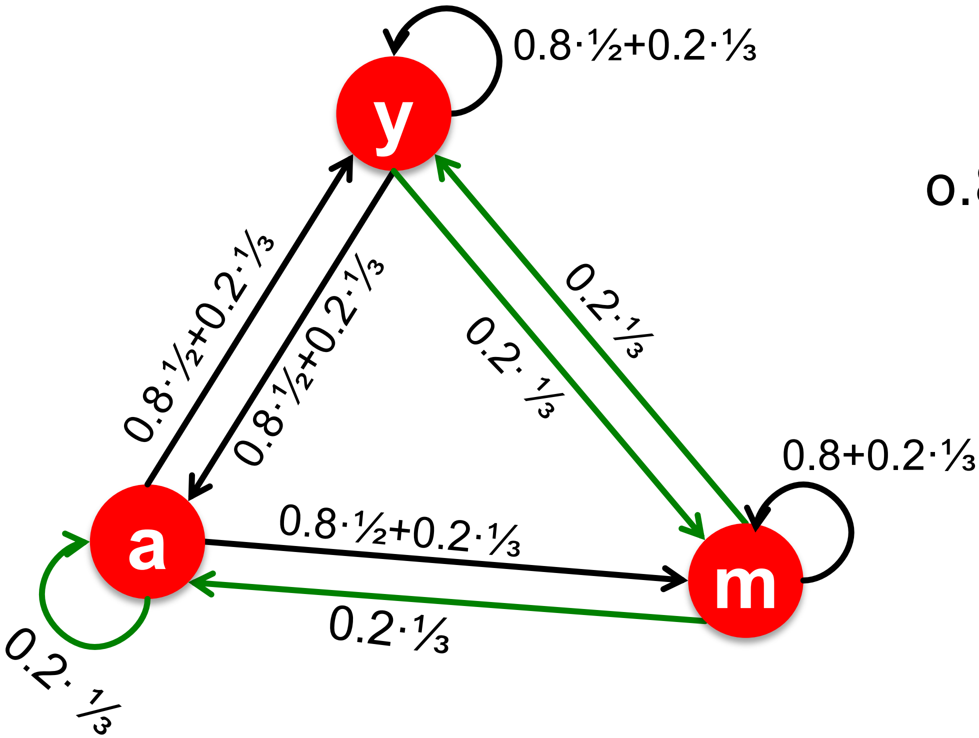
Link Analysis: TrustRank and Web Spam

CSEP590A Machine Learning for Big Data
Tim Althoff
PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

Example: PageRank Scores



Random Teleports ($\beta = 0.8$)



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

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

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

r = A r

PageRank: The Complete Algorithm

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

    $\boldsymbol{r'_j} = \mathbf{0}$ if in-degree of $\boldsymbol{j}$ is $\mathbf{0}$
    - **Now re-insert the leaked PageRank:**

      $\forall \boldsymbol{j}$: $\boldsymbol{r_j^{(t)}} = \boldsymbol{r'_j} + \frac{\mathbf{1}-\boldsymbol{S}}{\boldsymbol{N}}$ **where:** $S = \sum_j r'_j$
    - $\boldsymbol{t} = \boldsymbol{t} + \mathbf{1}$
  - **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**)
- **Uses a single measure of importance**
  - Other models of importance
  - **Solution:** Hubs-and-Authorities
- **Susceptible to Link spam**
  - Artificial link topographies created in order to boost page rank
  - **Solution:** TrustRank **(later today)**

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 also 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, history, or computer security

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 the walker teleports, she picks a page from a set $\boldsymbol{S}$
  - $\boldsymbol{S}$ contains only pages that are relevant to the topic
    - E.g., Open Directory (DMOZ) pages for a given topic/query
  - For each teleport set $\boldsymbol{S}$, we get a different vector $\boldsymbol{r_S}$

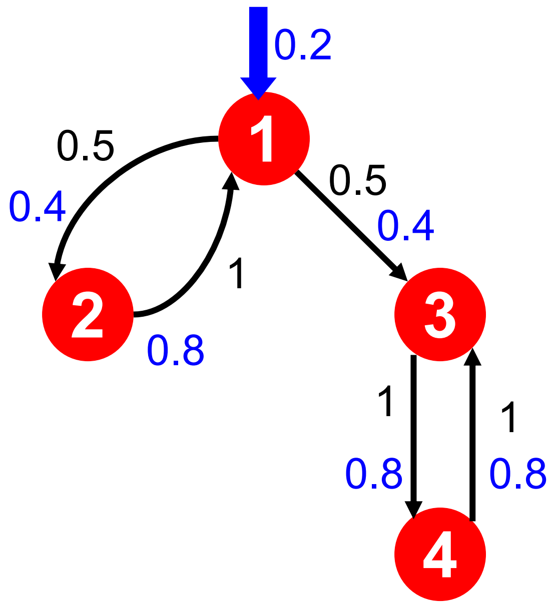
Matrix Formulation

- **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 a stochastic matrix!
- 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	β	r_1	r_2	r_3	r_4
{1}	**0.9**	0.17	0.07	0.40	0.36
{1}	**0.8**	0.29	0.12	0.33	0.26
{1}	**0.7**	0.39	0.14	0.27	0.19

S	β	r_1	r_2	r_3	r_4
{1,2,3,4}	0.8	0.13	0.10	0.39	0.36
{1,2,3}	0.8	0.17	0.13	0.38	0.30
{1,2}	0.8	0.26	0.20	0.29	0.23
{1}	0.8	0.29	0.12	0.33	0.26

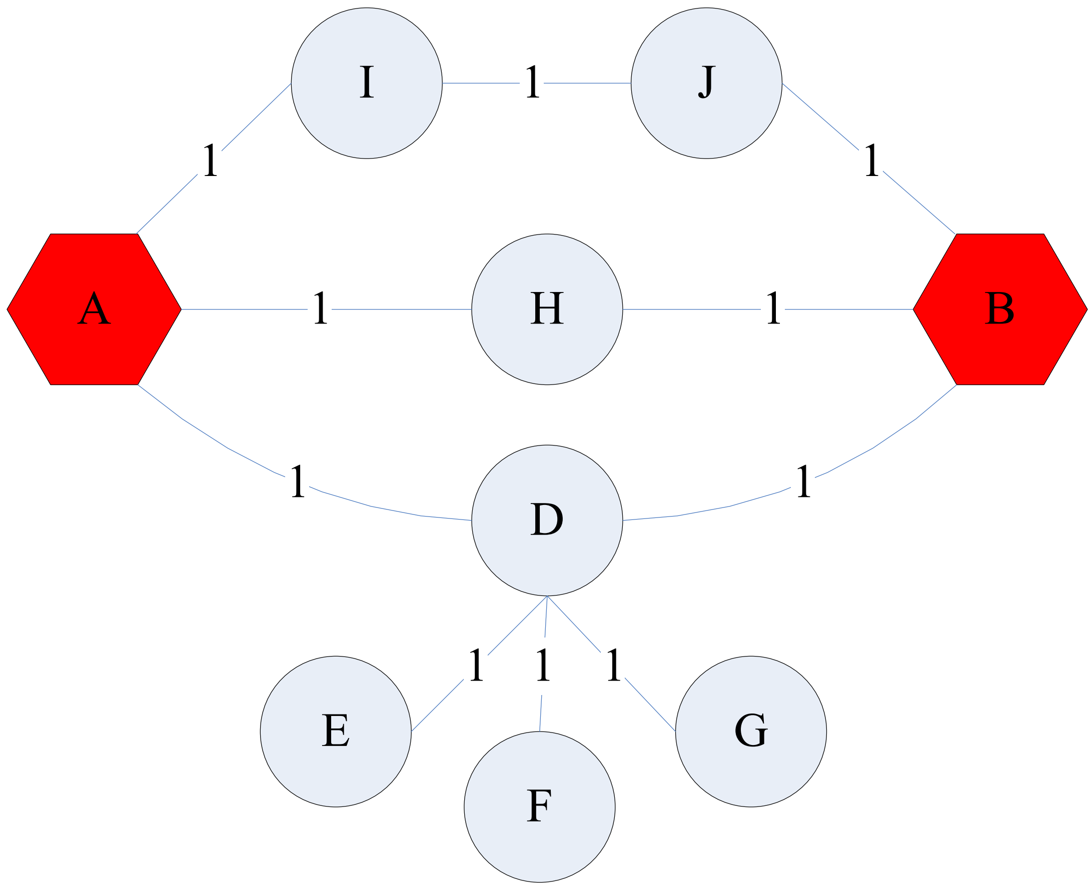
Discovering the Topic Set S

- **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, …

Application to Measuring Proximity in Graphs

Random Walk with Restarts: set *S* is a single node

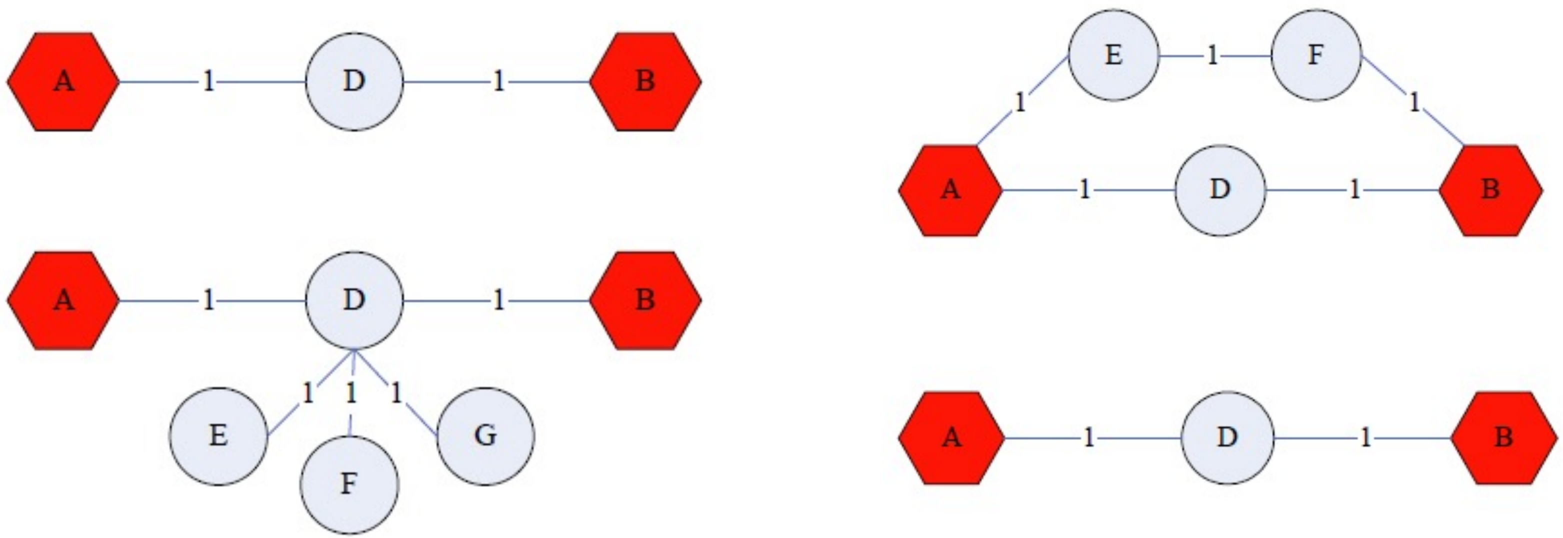
Proximity on Graphs



a.k.a.: Relevance, Closeness, 'Similarity'…

Good proximity measure?

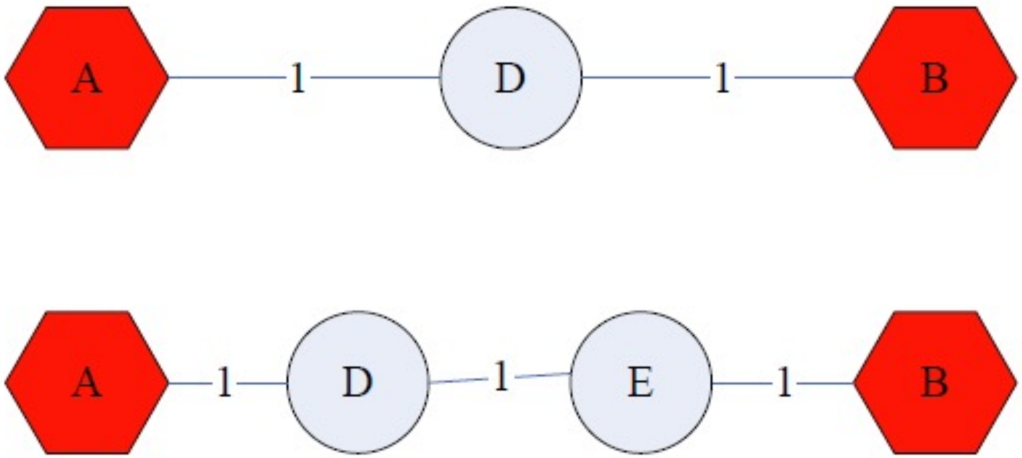
- **Shortest path is not good:**



- **No effect of degree-1 nodes (E, F, G)!**
- Multi-faceted relationships

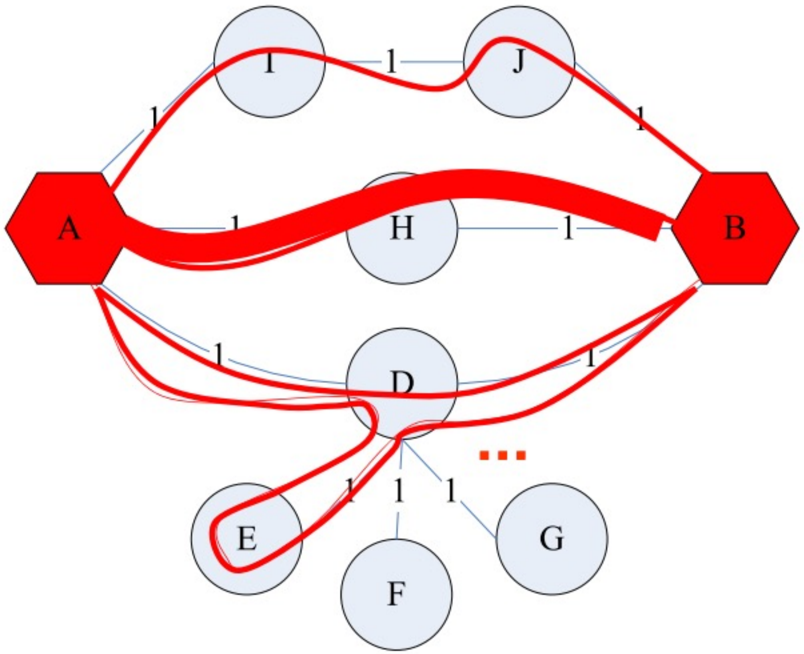
Good proximity measure?

- **Network flow is not good:**



- **Does not punish long paths**

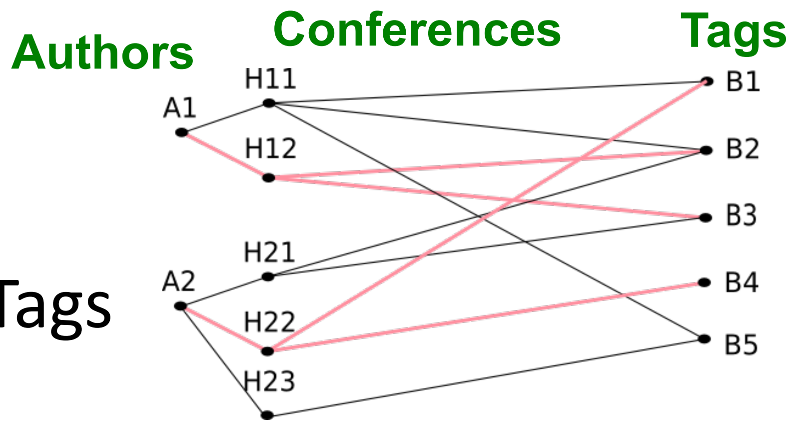
What is a good notion of proximity?



- **Need a method that considers:**
  - Multiple connections
  - Multiple paths
  - Direct and indirect connections
  - Degree of the node

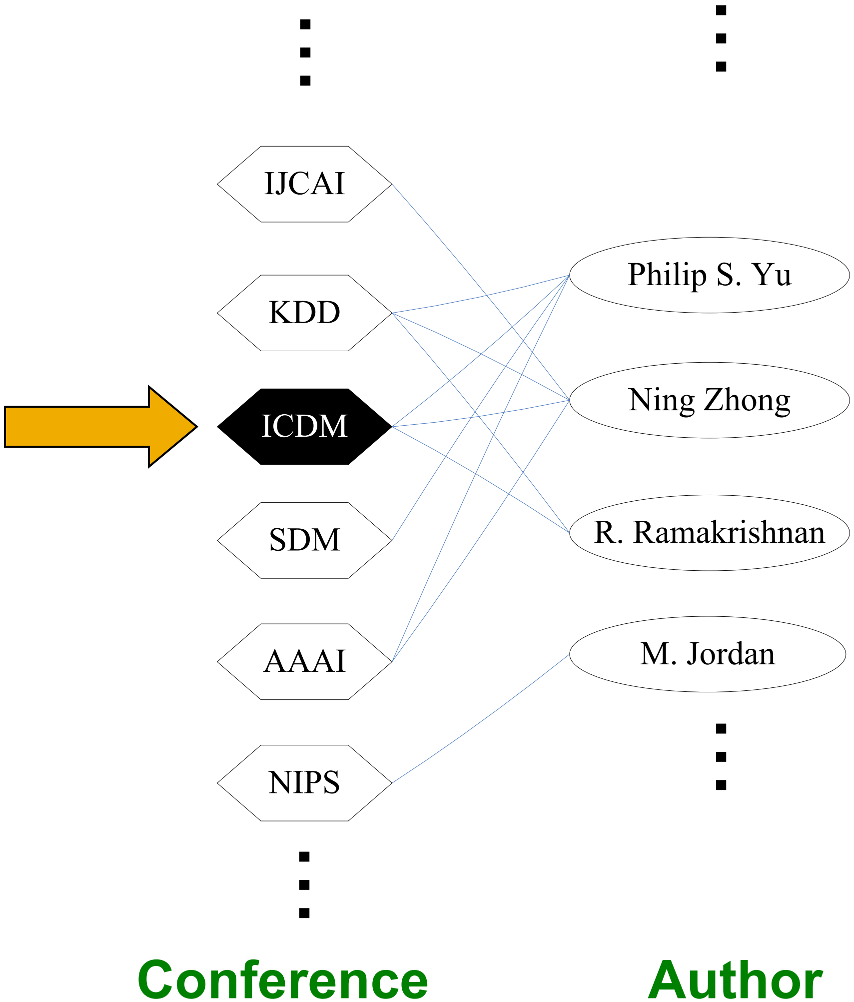
SimRank: Idea

- **SimRank:** Random walks from a **fixed node** on ***k***-partite graphs
- **Setting:** ***k***-partite graph with ***k*** types of nodes
  - E.g.: Authors, Conferences, Tags



- **Topic Specific PageRank from node *u*: teleport set *S* = {*u*}**
- Resulting scores measure similarity/proximity to node ***u***
- **Problem:**
  - Must be done once for each node ***u***
  - Only suitable for sub-Web-scale applications

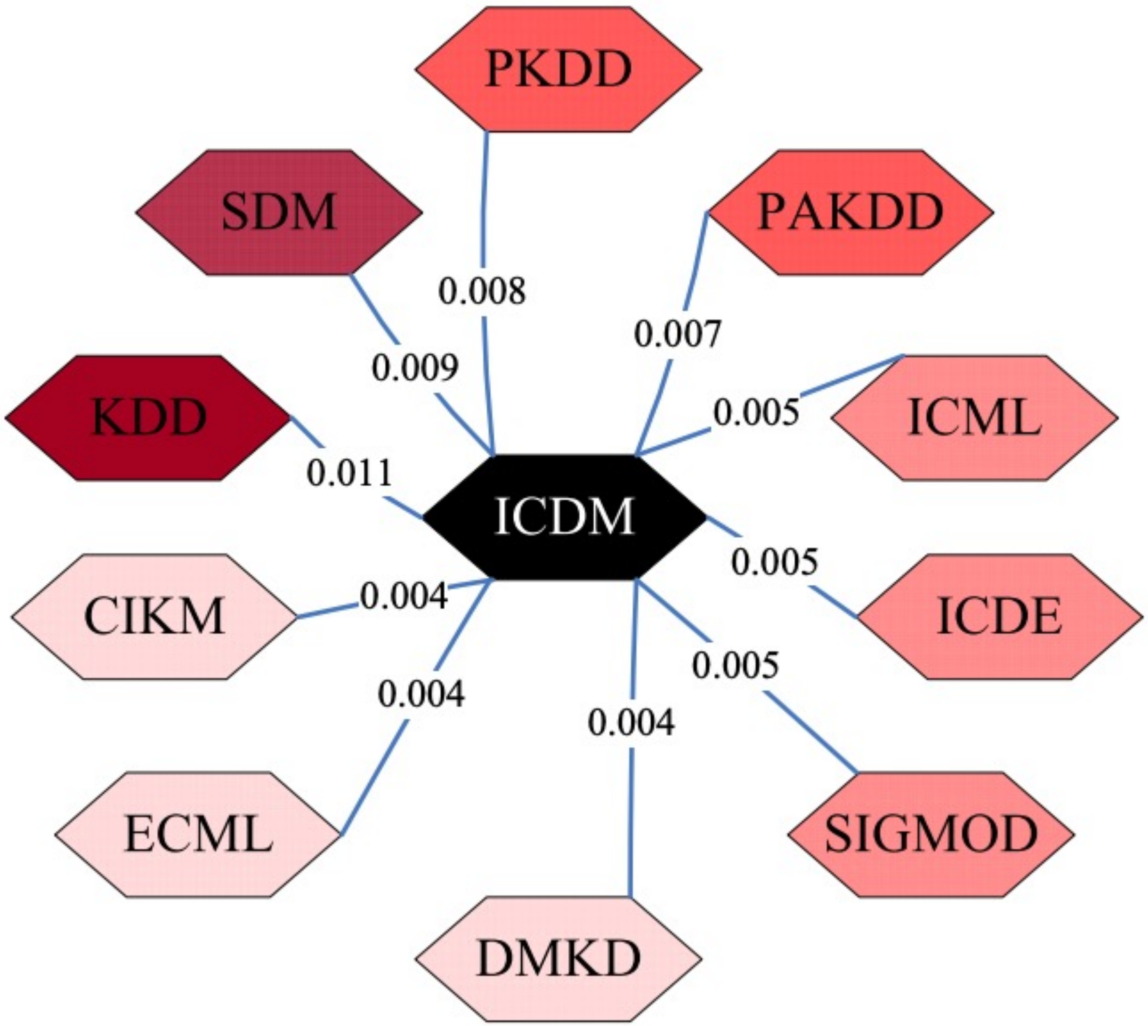
SimRank: Example



Q: What is the most related conference to **ICDM**?

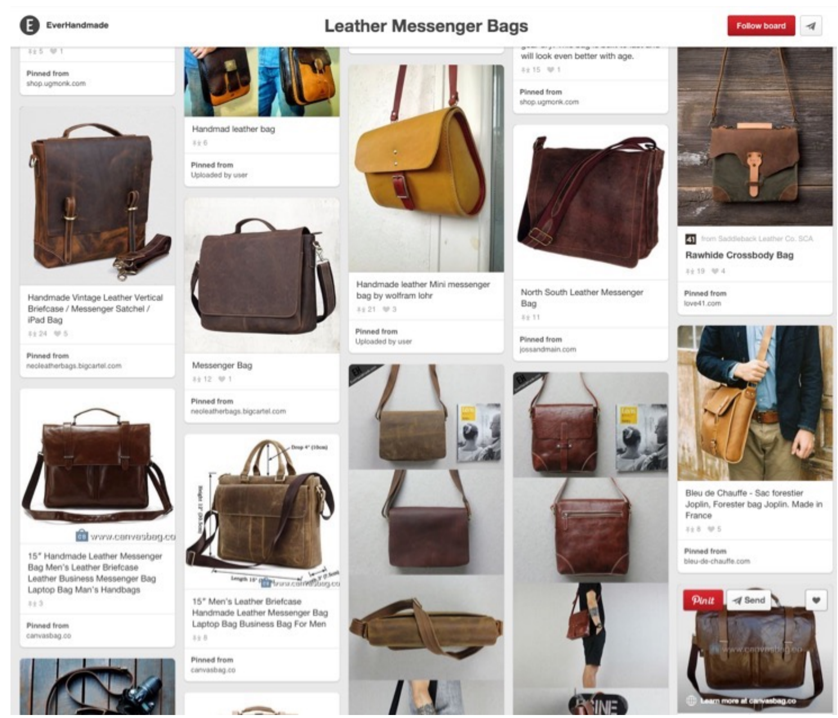
A: Topic-Specific PageRank with teleport set S={ICDM}

SimRank: Example



Pinterest: Pins and Boards

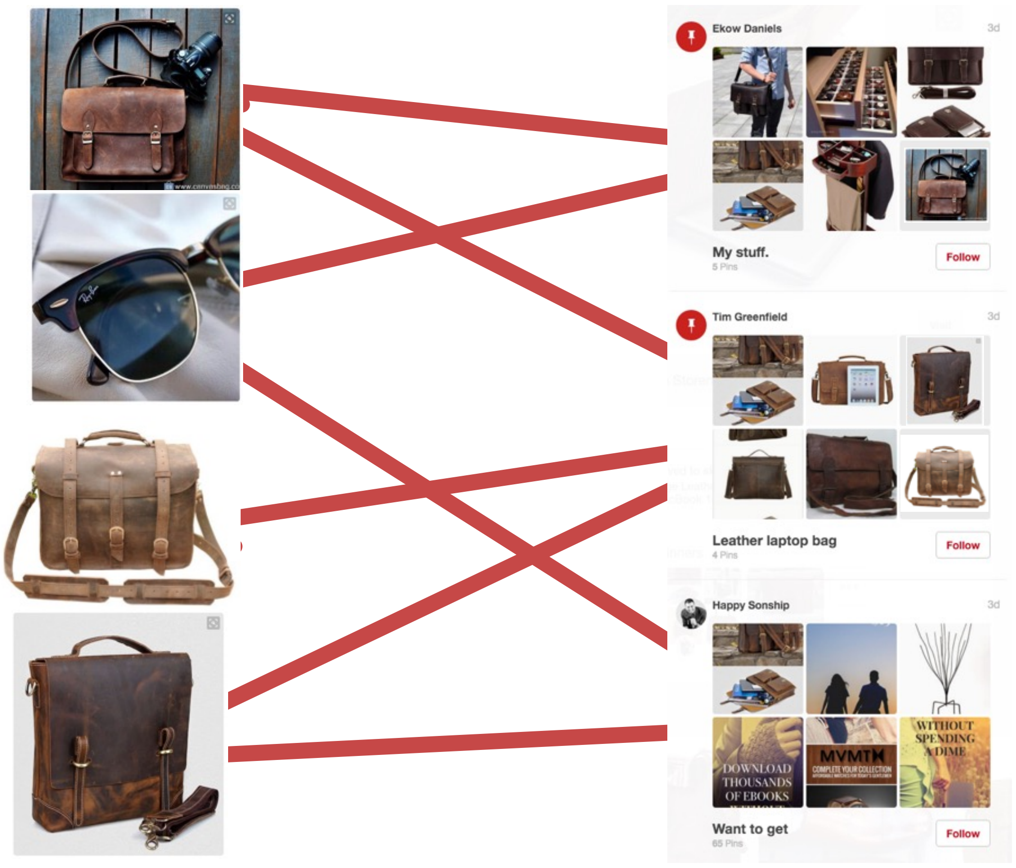
Pin



Board

Pinterest is a Giant Bipartite Graph

- Pins belong to Boards

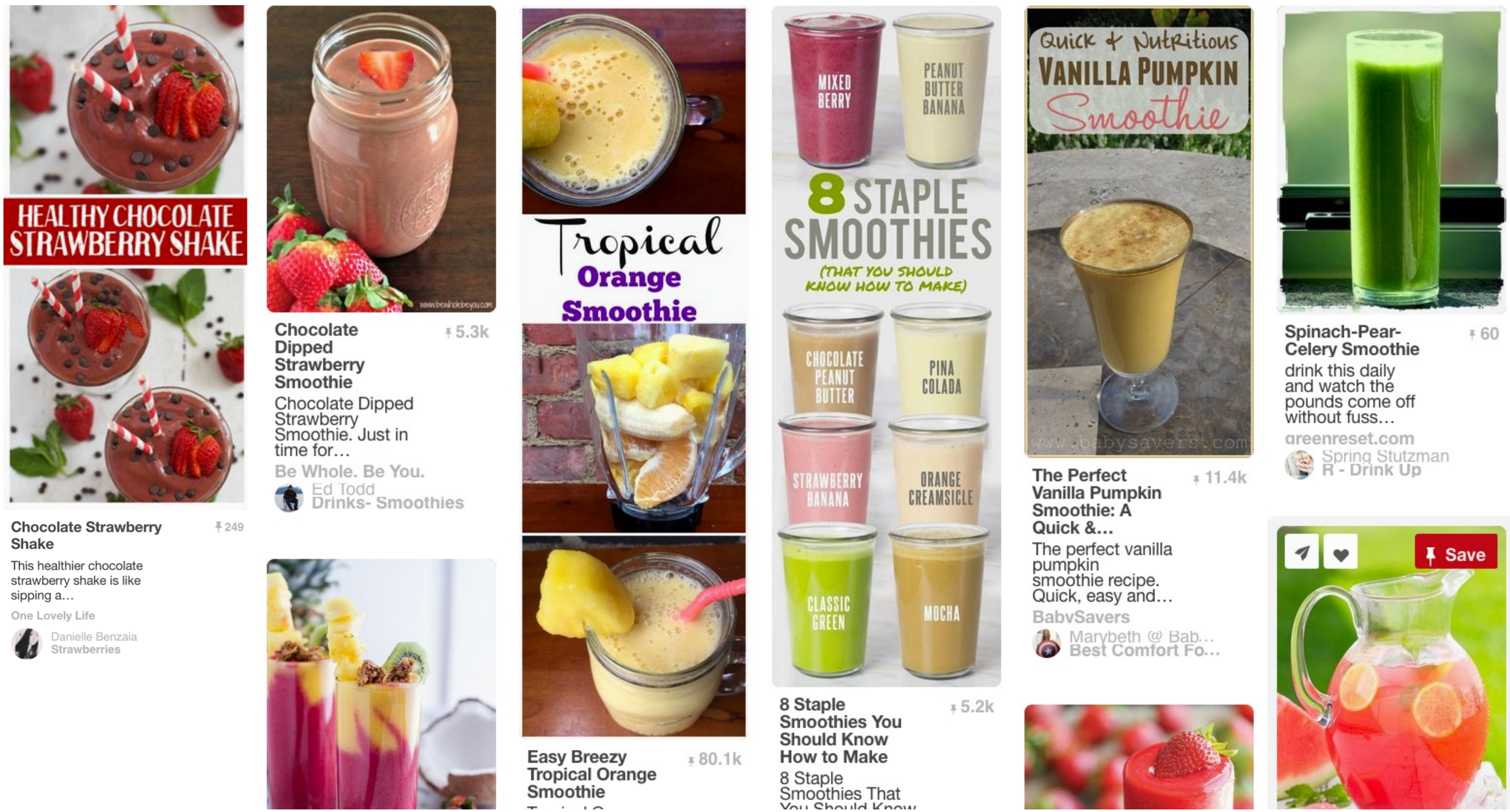


Pins to Pins Recommendations

Input:

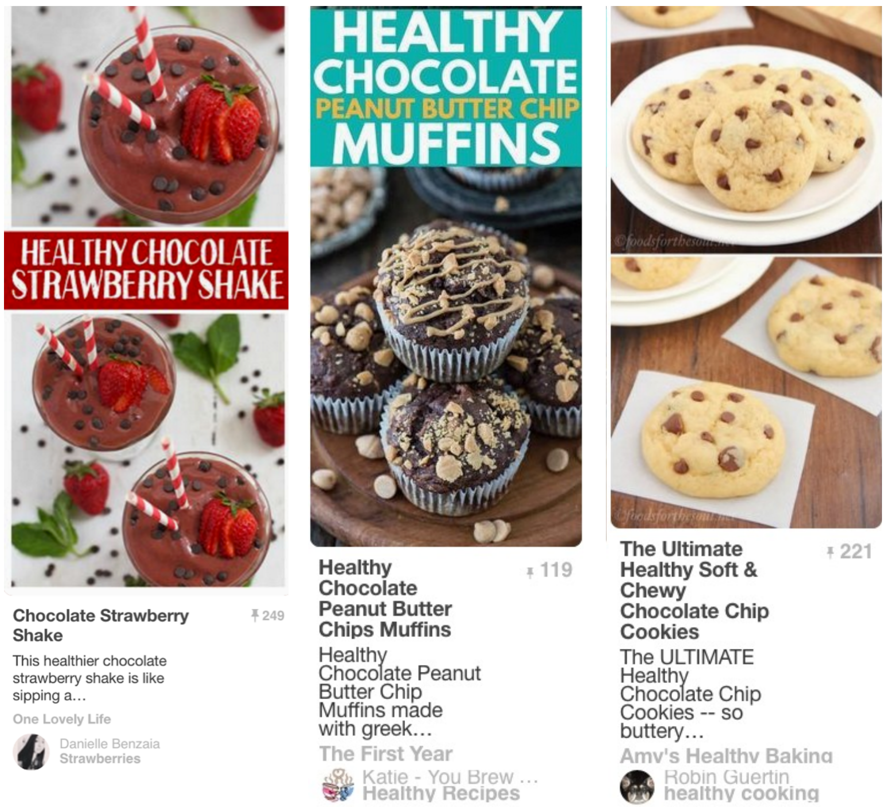
Pins to Pins Recommendations

Input: Recommendations:



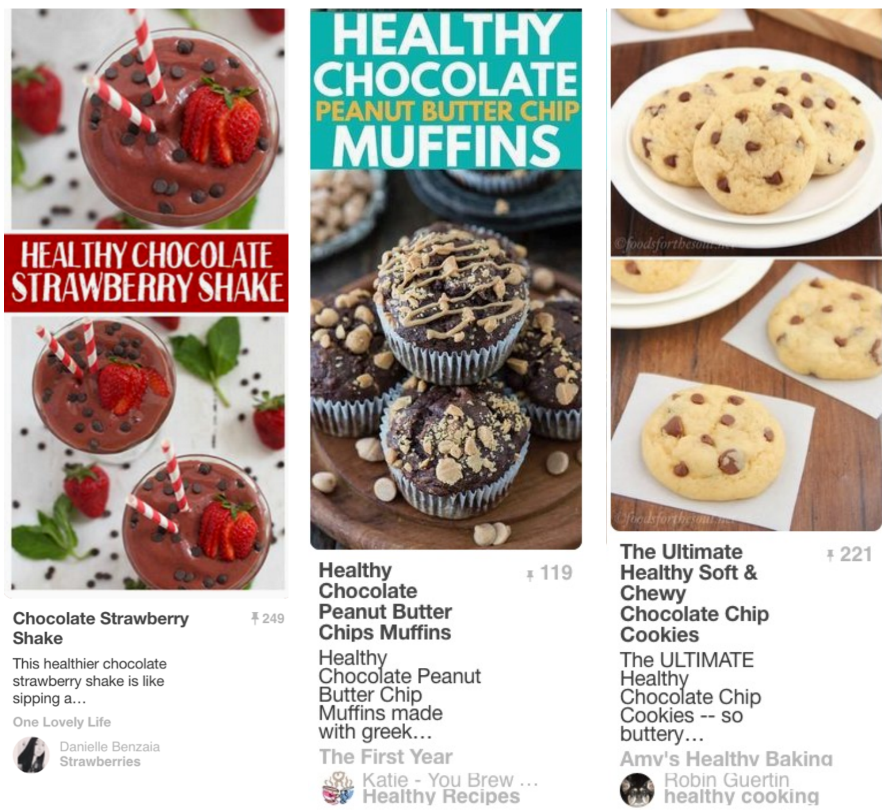
Pins to Pins Recommendations

Input:

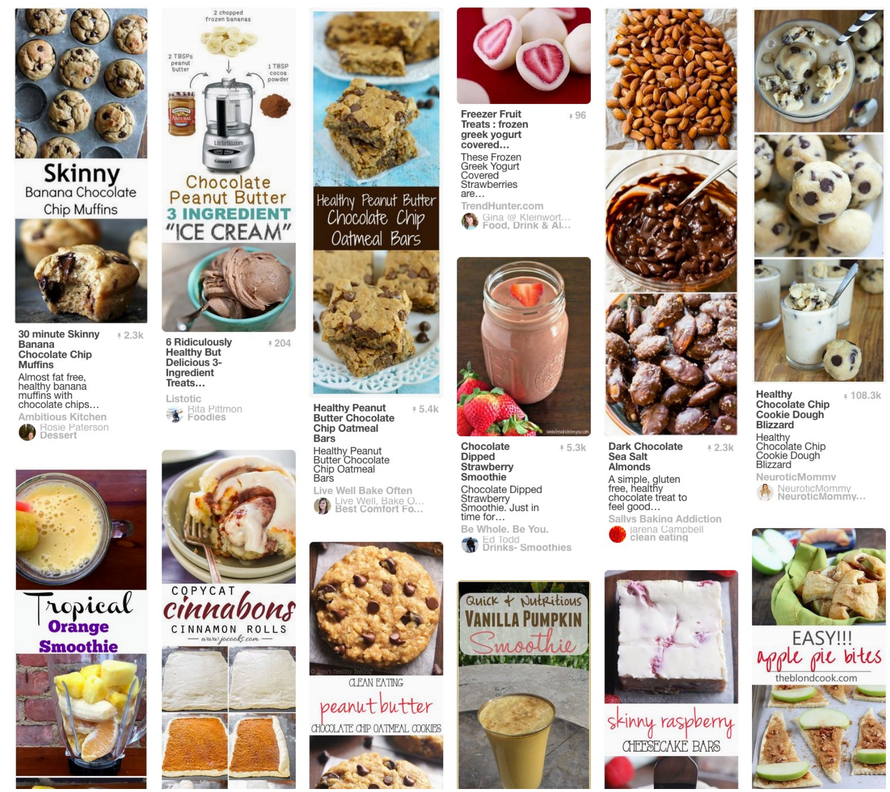


Pins to Pins Recommendations

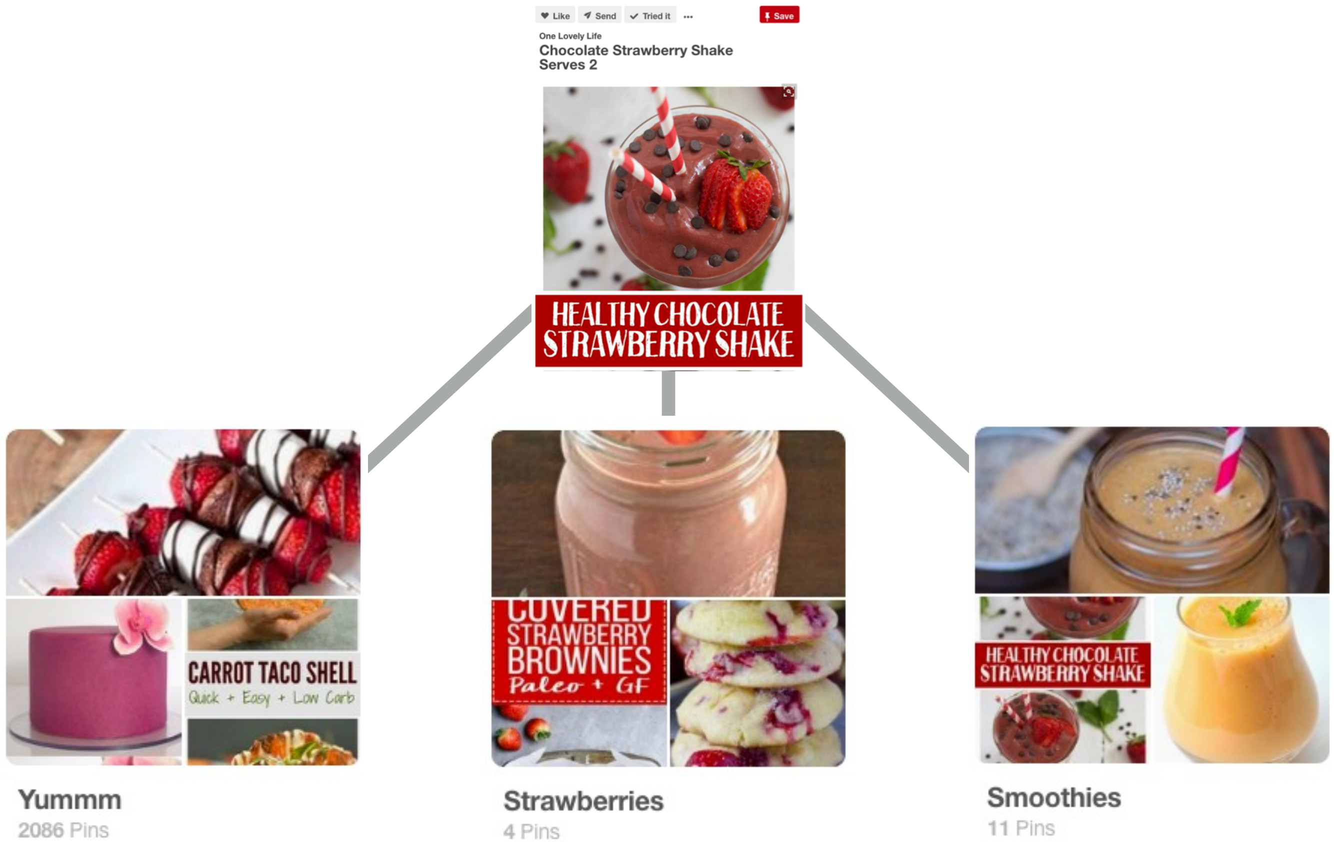
Input:



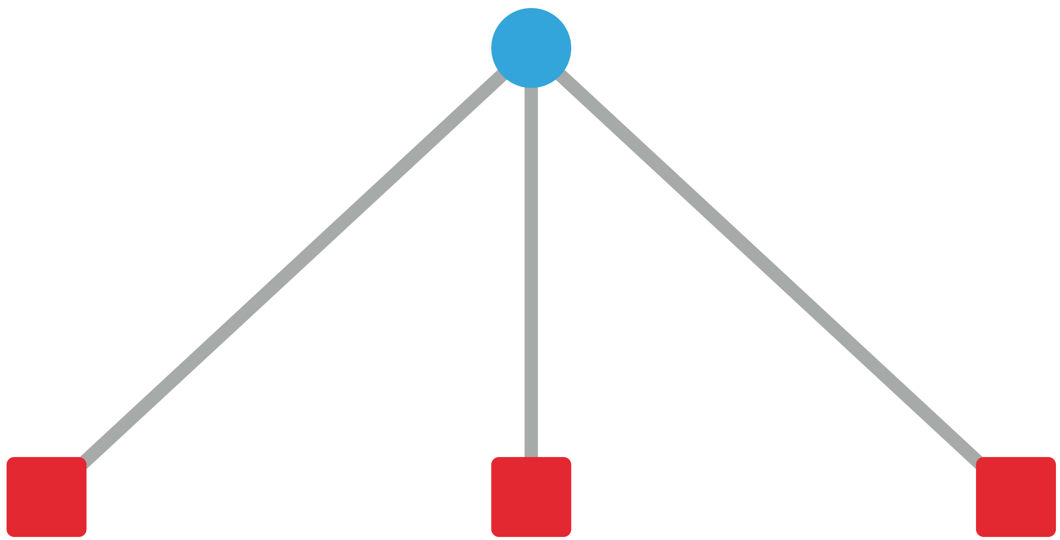
Recommendations:



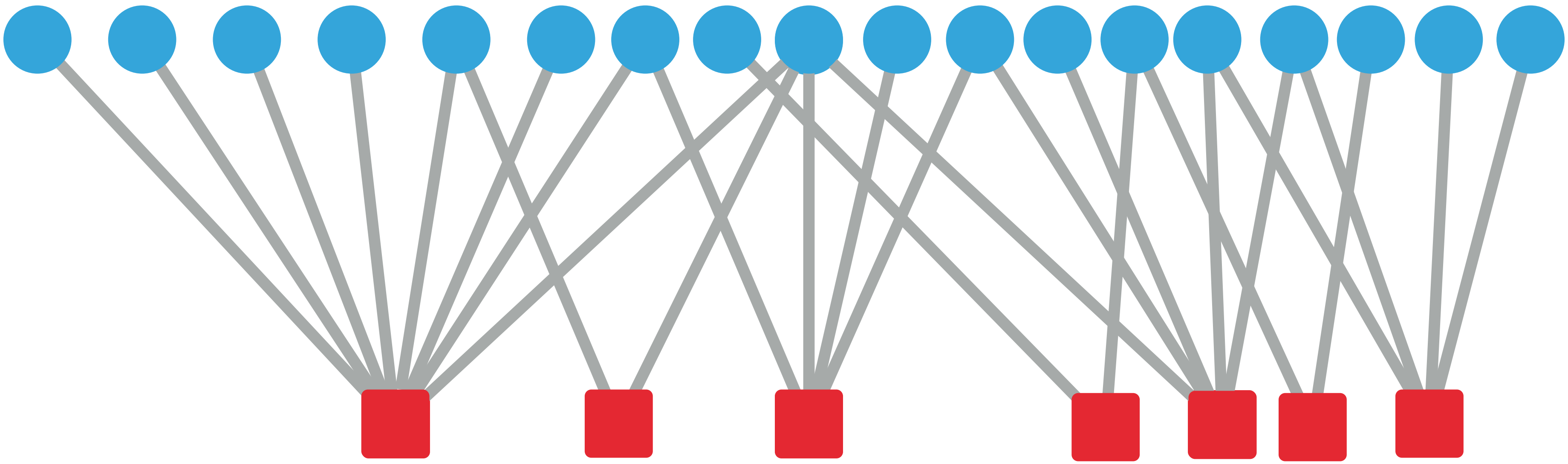
Bipartite Pin And Board Graph



Bipartite Pin And Board Graph

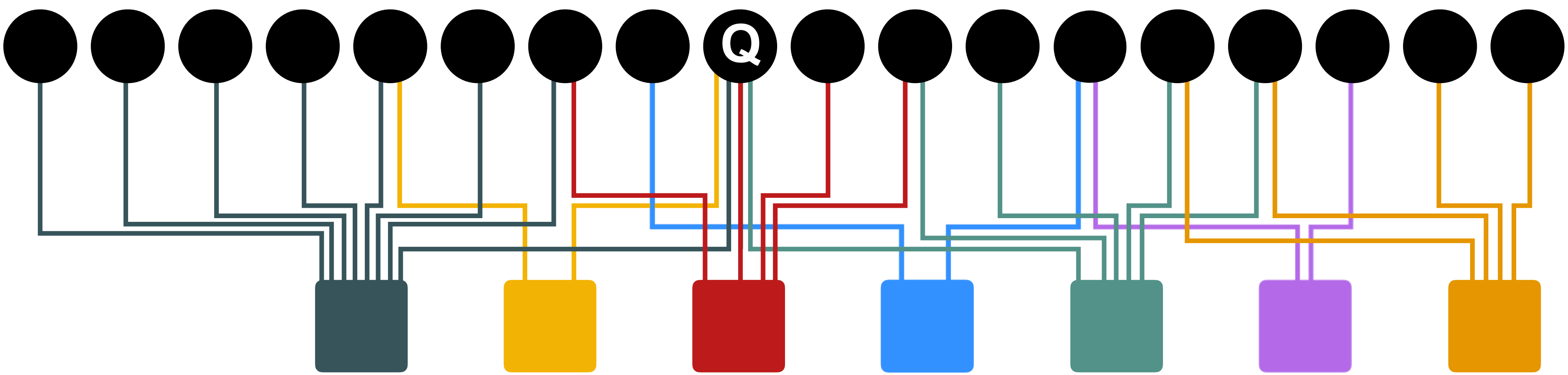


Bipartite Pin And Board Graph



Pixie Random Walks

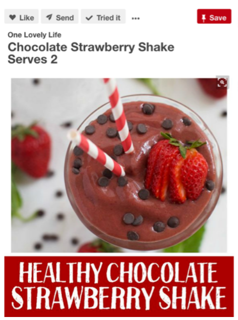
- **Idea:**
  - Every node has some importance
  - Importance gets evenly split among all edges and pushed to the neighbors
- Given a set of QUERY NODES Q, simulate a random walk:



Pixie Random Walk Algorithm

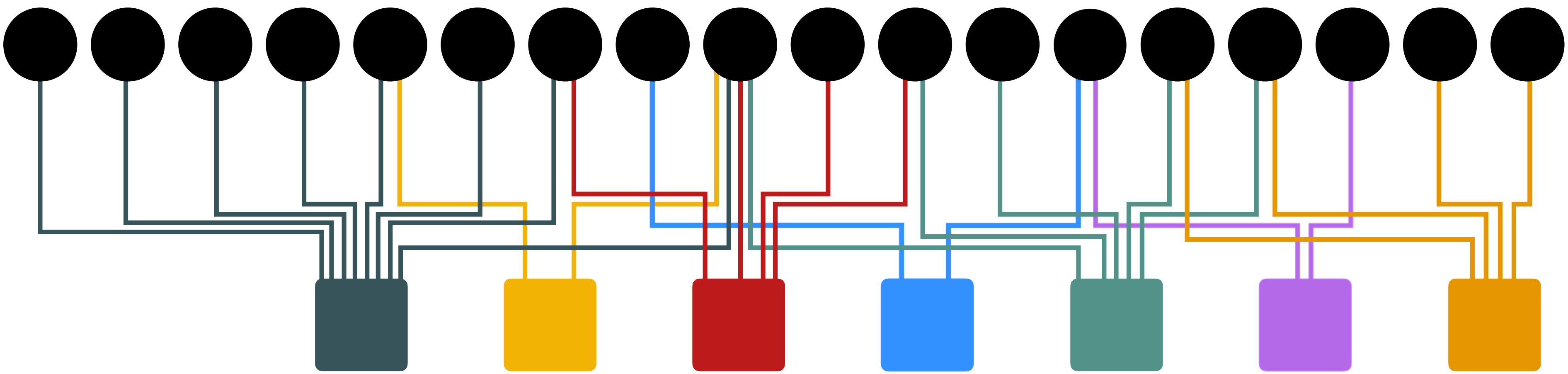
- Proximity to query node(s) ***Q***:

```
ALPHA = 0.5
QUERY_NODES = {
```




```
}
```

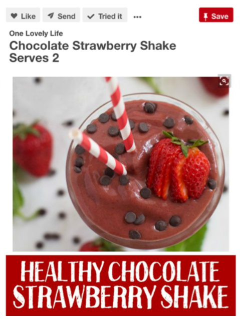
```
pin_node = QUERY_NODES.sample_by_weight()
for i in range(N_STEPS):
    board_node = pin_node.get_random_neighbor()
    pin_node = board_node.get_random_neighbor()
    pin_node.visit_count += 1
    if random() < ALPHA:
        pin_node = QUERY_NODES.sample_by_weight()
```



Pixie Random Walk Algorithm

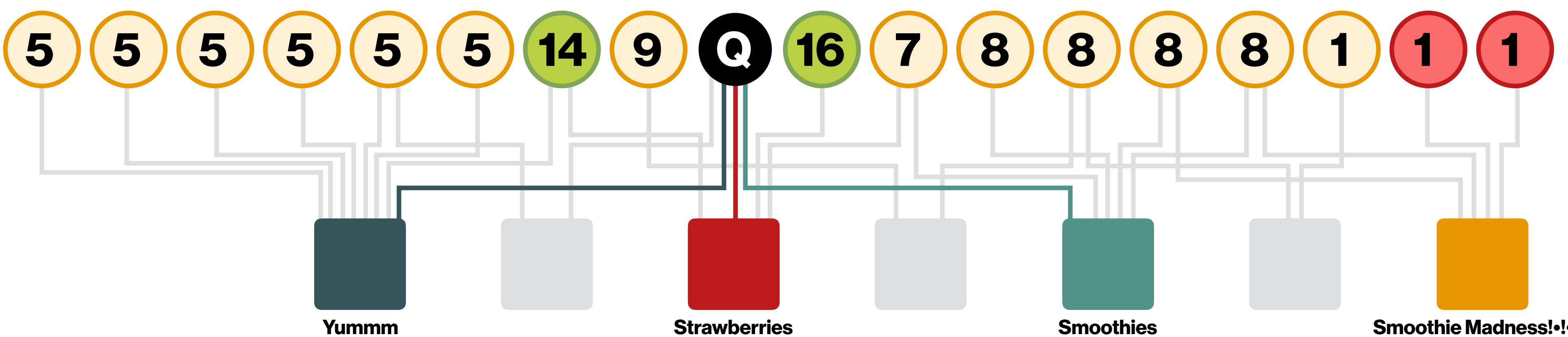
- Proximity to query node(s) ***Q***:

```
ALPHA = 0.5
QUERY_NODES = {
```




```
}
```

```
pin_node = QUERY_NODES.sample_by_weight()
for i in range(N_STEPS):
    board_node = pin_node.get_random_neighbor()
    pin_node = board_node.get_random_neighbor()
    pin_node.visit_count += 1
    if random() < ALPHA:
        pin_node = QUERY_NODES.sample_by_weight()
```



Pixie Recommendations

- **Pixie:**
  - **Outputs top 1k pins with highest visit count**

Extensions:

- **Weighted edges:**
  - The walk prefers to traverse certain edges:
    - Edges to pins in your local language
- **Early stopping:**
  - Don’t need to walk a fixed big number of steps
  - Walk until 1k-th pin has at least 20 visits

Graph Cleaning/Pruning

- **Pinterest graph has 200B edges**
- We don't need all of them!
  - Super popular pins are pinned to millions of boards
    - **Not useful:** When the random walk hits the pin, the signal just disperses. Such pins appear randomly in their recommendations.
- **What we did:** Keep only good boards for pins
  - Compute the similarity between pin's topic vector and each of its boards. Only take boards with high similarity.

Data Type	Number	Size	Memory
Pin Nodes	3 Billion	8 Bytes	24 GiB
Board Nodes	2 Billion	8 Bytes	16 GiB
Undirected Edges	20 Billion	8 Bytes	160 GiB
			208 GiB

Benefits of Pixie

- **Benefits:**
  - Very fast: Given Q, we can output top 1k in 50ms (after doing 100k steps of the random walk)
  - Single machine can run 1500 walks in parallel! (1500 recommendation requests per second)
  - Can fit entire graph in RAM (17B edges, 3B nodes)
  - Can scale it by just adding more machines
- **Today about 70% of all the pins you see at Pinterest are recommended by random walks**

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 (e.g. SimRank):**
  - Topic-Specific PageRank where teleport is always to the same node. S=[0, 0, 0, 0, **1**, 0, 0, 0, 0, 0]

TrustRank:
Combating Spam on the Web

What is Web Spam?

- **Spamming:**
  - Any deliberate action to boost a web page's position in search engine results, incommensurate with the 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 1,000 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 on top of result ranking
    - Copy it into your page, make it “invisible”
- **These and similar techniques are term spam**

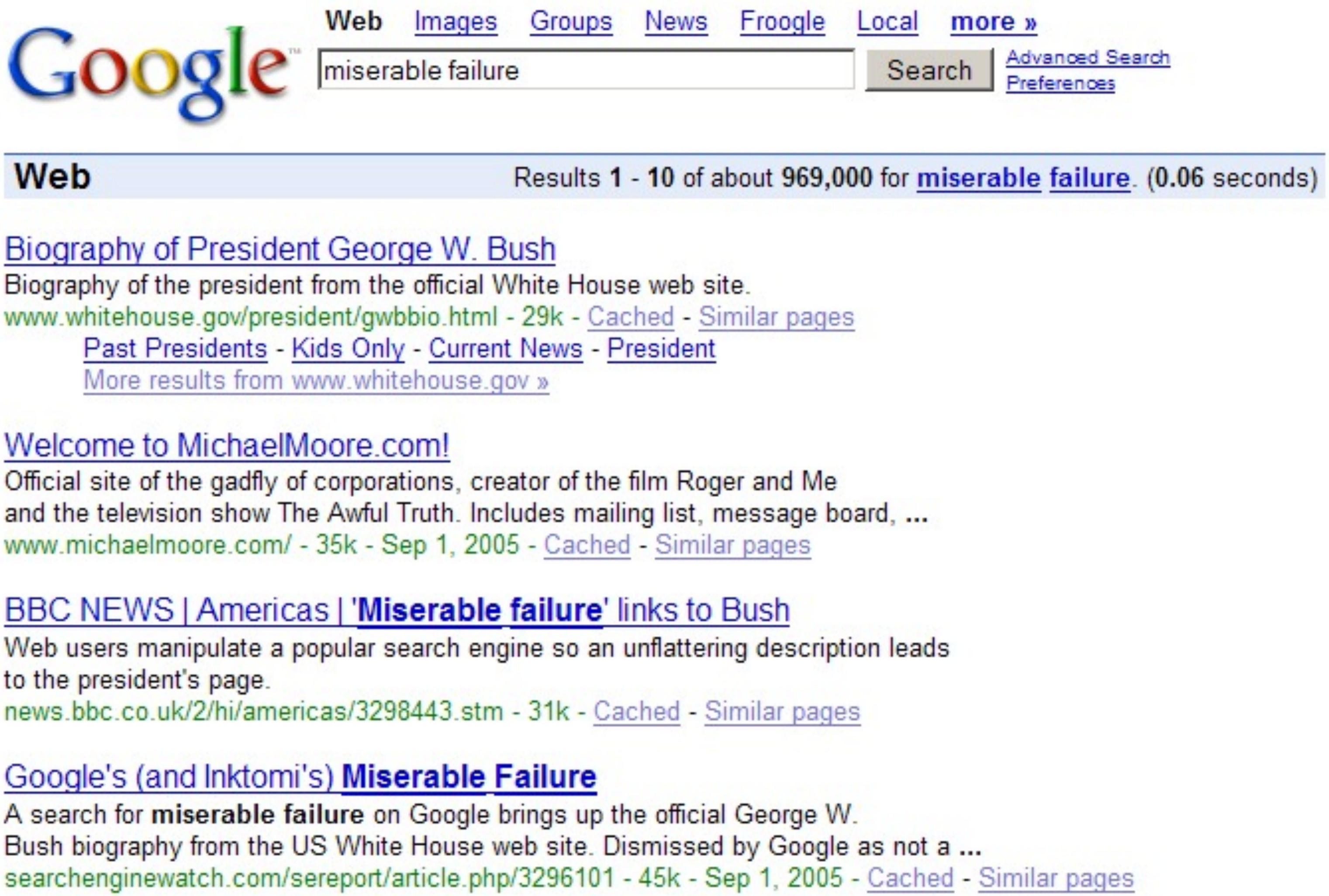
Google's Solution to Term Spam

- **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 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

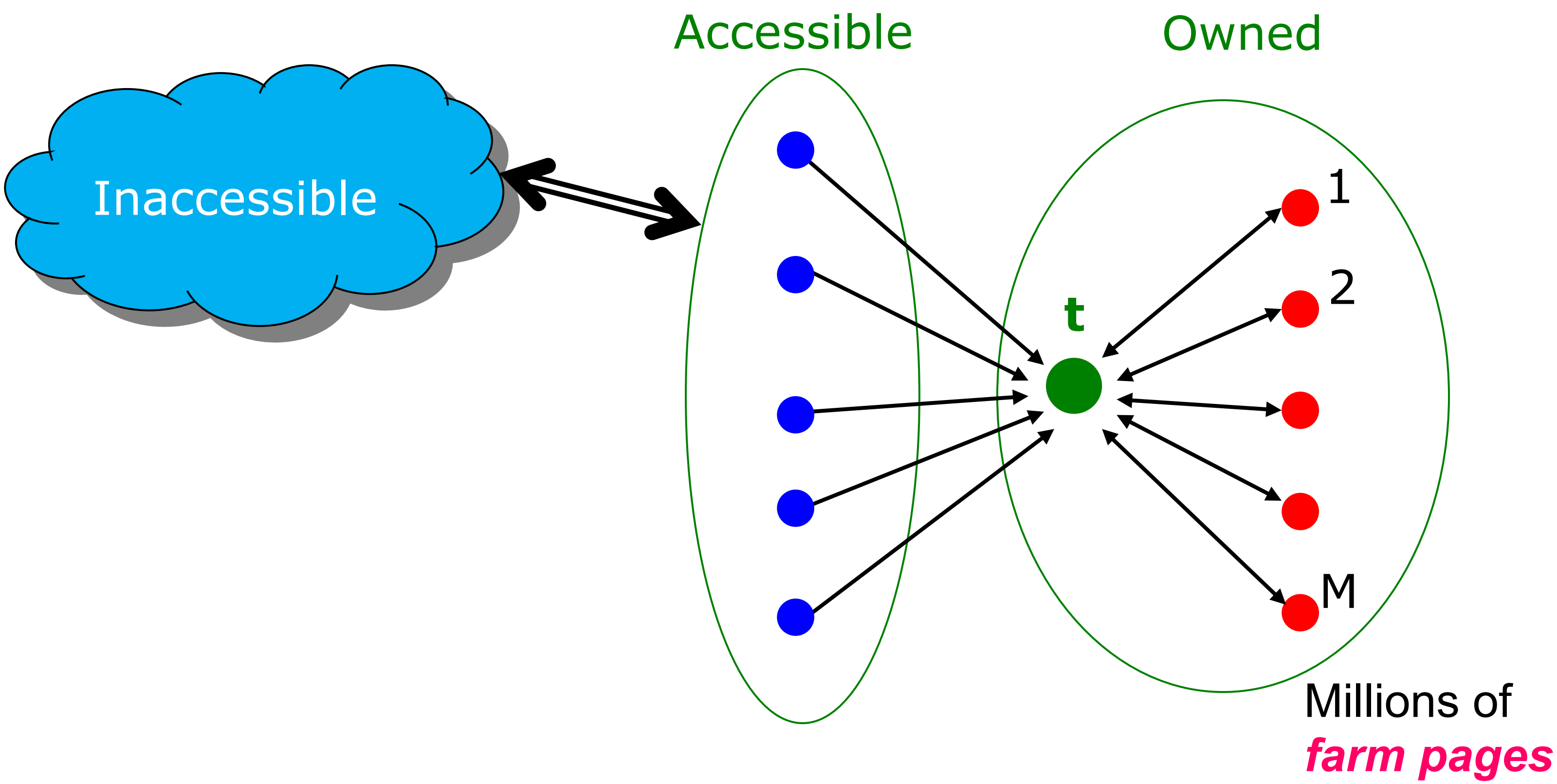
Link Spamming

- **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

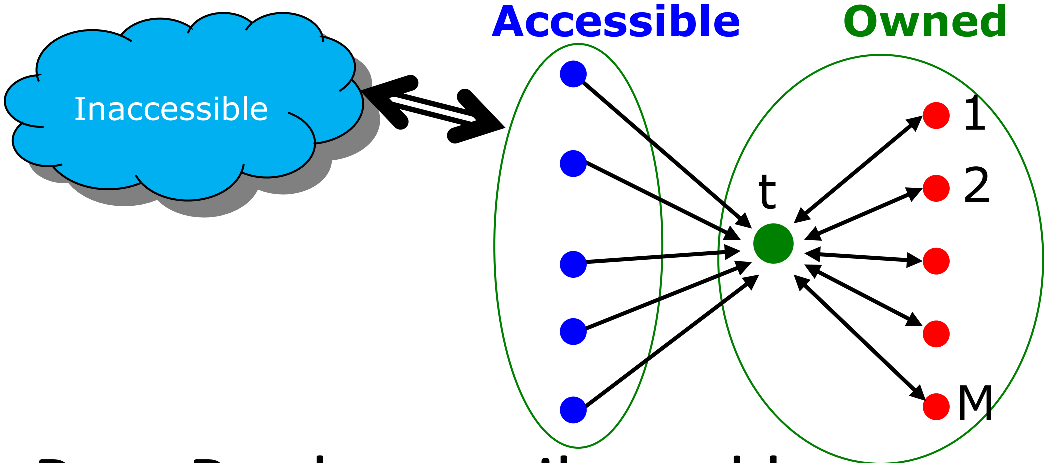
- **Spammer's goal:**
  - Maximize the PageRank of target page ***t***
- **Technique:**
  - Get as many links from accessible pages as possible to target page ***t***
  - Construct "link farm" to get PageRank multiplier effect (next)

Link Farms



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

Analysis



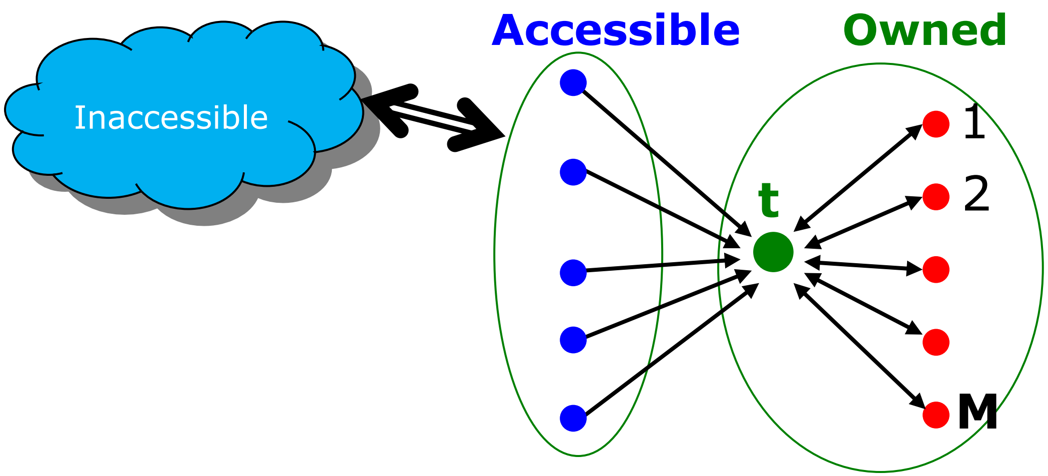
N…# pages on the web
M…# of pages spammer owns

- **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
  Now we solve for ***y***
- $y = \frac{x}{1-\beta^2} + c\frac{M}{N}$ 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**

TrustRank: Combating Spam on the Web

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 $\boldsymbol{q}$
  - $\beta\, \boldsymbol{t_p / |o_p|}$ 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 is additive**
  - Sum up trust from pages linking to target page
- **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
- **Trust attenuation:**
  - The degree of trust conferred by a trusted page decreases with the distance in the graph

Picking the Seed Set

- **Two conflicting considerations:**
  - **Cost:** Human has to inspect each seed page, so seed set must be as small as possible
  - **Coverage:** 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
- **How to do that?**
- **(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

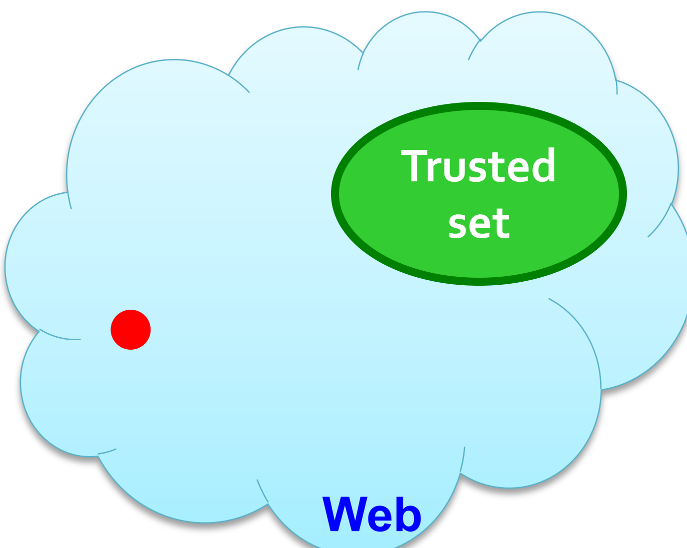
TrustRank

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 ***p***
- $\boldsymbol{r_p^+}$ = PageRank of ***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 *p* =** $\boldsymbol{\frac{r_p^-}{r_p}}$
  - Pages with high spam mass are spam; can filter them out