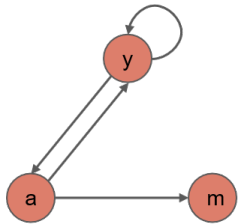
CSCI 4022 Fall 2021
PageRank Wrapup

Announcements:

1. Exam grading not quite done.
2. Work on both HW5 and consider project data sets

PageRank and Teleports

1. With probability β, follow a real link at random.
2. With probability $1-\beta$, *teleport* or jump to any random page, equally likely
3. Common values for β: 0.8-0.9
4. Brin-Page, 1998 (with tens of thousands of citations!) gives the updates PageRank formula:



sparse

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{N}$$

at the end

The resulting transition matrix is given by

of matrix

$$A = \beta M + (1-\beta) \overbrace{\left[\frac{1}{N}\right]_{N\times N}}^{every\ entry\ is\ 1/N}$$

Full PageRank:

The idea: We want the eigenvector with eigenvalue of 1 of the matrix M. Then $Mr = r$ is solved by eigenvalue-eigenvector pair $(1, r)$. Column-stochastic matrices always have *largest* eigenvalue of 1. The resulting algorithm that finds largest eigenvalue, eigenvector pairs is known as **power iteration**.

Power Iteration

1. Suppose that there are N total web pages to rank.
2. Initialize: $r^{(0)} = [1/N, 1/N, \ldots 1/N]$
3. Iterate: $r^{(t+1)} = Mr^{(t)}$
4. Stop if: $d(r^{(t+1)}, r^{(t)})$ is small. A typical stop would be $d(r^{(t+1)}, r^{(t)}) < \varepsilon$ under appropriate norm (like L_1), so $d(r^{(t+1)}, r^{(t)}) = \sum_{i=1}^{N} |r_i^{(t+1)} - r_i^{(t)}|$ for some small ε

PageRank Iteration with Teleports:

1. Compute $\boldsymbol{r}^{new} = M\boldsymbol{r}^{old}$ ← right not col-stoch.
2. Add a constant $(1-\beta)/N$ to every entry in $\boldsymbol{r}^{new}$
3. Renormalize $\boldsymbol{r}^{new}$ so that is sums to 1. **Must** be done if M has dead ends, but can also help with floating point precision in general.
4. Convergence check, as before.

Computing Page Rank

Algorithm: with a couple of short-cuts...

- **Input:** a convergence tolerance ε, graph G, and parameter β.
- **Output:** PageRank vector $\boldsymbol{r}$

Initialize: $\boldsymbol{r}_{old} = 1/N$ for # of nodes N. **Iterate:**

1. Let the link economy spread out importance. **For all** j: $\boldsymbol{r}_j^{new,*} = \sum_{i \to j} \beta \boldsymbol{r}_i^{old}/d_i$ (for all in-links, sum importance/degree of in-link).
2. Then Re-insert the "leaked" or "taxed" PageRank from teleports. **For all** j: $\boldsymbol{r}_j^{new,*} = \boldsymbol{r}_j^{new,*} + \frac{1-S}{N}$. Where we keep a running tally of $S = \sum_j \boldsymbol{r}_j^{new,*}$ in the prior step. This handles **both** normalization and including teleports: take whatever $\boldsymbol{r}_j^{new,*}$ currently sums to, then add what's needed to make it sum to 1... divided equally over all N entries.
3. Check convergence with some norm, e.g. **break** if $d(\boldsymbol{r}_{old}, \boldsymbol{r}_{new}) < \varepsilon$.
4. Update $\boldsymbol{r}_{old} = \boldsymbol{r}_{new}$

Sparse Mat-Vec

Assuming $\boldsymbol{r^{new}}$ fits into main memory, we can run this by accessing $\boldsymbol{r^{old}}$ and M from disk.

One step of power iteration becomes:

1. Initialize all entries of $\boldsymbol{r^{new}} = \frac{1-\beta}{N}$
2. For each page i with out-degree d_i...
   - 2a. Read into memory **column** i from M, so we get $(i, d_i, \{j_1, j_2, \ldots j_{d_i}\})$
   - 2b. For j in $\{j_1, j_2, \ldots j_{d_i}\}$, the destinations of i...:

$$\boldsymbol{r}_j^{new} += \beta \boldsymbol{r}_i^{old}/d_i$$

3.,4. Normalize r, then check convergence.

Alternatively, initialize $\boldsymbol{r^{new}} = 0$ and add $\frac{1-S}{N}$ instead of normalizing r, as the prior slide does.

Sparse Mat-Vec

You may be asking... where did M go? We're trying to not save it into memory by reverting back to the original thought process: importance only flows out along links $i \to j$ **and** flows out relative to the number outlinks of a node d_i.

So those things are all that we want to save and use. We encode only the non-zero entries of M by keeping our links as a list of lists or a dictionary:

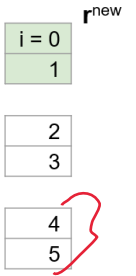
Source node	Degree	Destination nodes
0	6	2, 9, 19, 88, 2019, 3031
1	4	4, 19, 28, 1991
2	2	0, 42
...	...	...

Our loop is then a **for each link** instead of **for each location of M**.

Sparse Mat-Vec

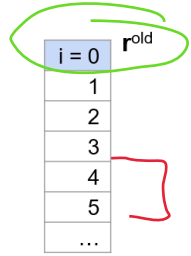
We can even tweak the algorithm if we don't want to have to load all of M or r into memory at once. This is known as **block** updating. To not load r into memory:

1. Break $\boldsymbol{r}^{new}$ into k **blocks** or chunks, and then
2. update one chunk at a time by *scanning* through M and $\boldsymbol{r}_{old}$ once *per block*



M

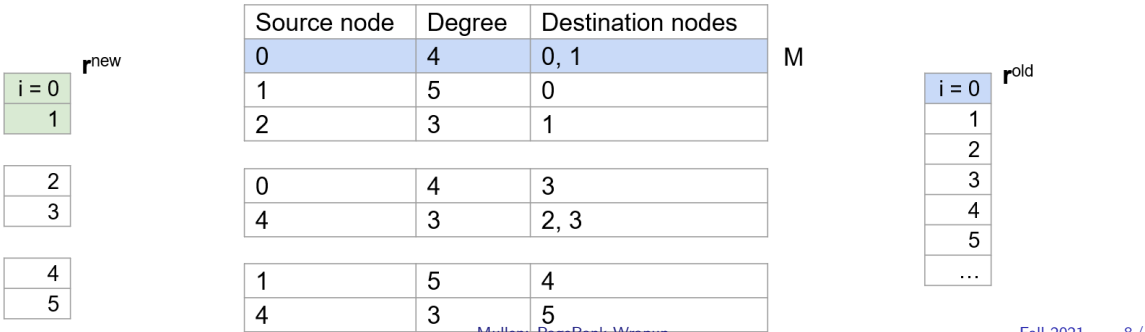
Source node	Degree	Destination nodes
0	4	0, 1, 3, 8
1	5	0, 4, 6, 7, 9
2	3	1, 8, 15
3	1	7
4	3	2, 3, 5
…	…	…



Block-Stripe

The absolute best-case for a huge M is that certain regions of M only link to themselves or one another. *For those regions*, we could save additional time by breaking M into k *stripes*, that correspond to the blocks of r where destination nodes are needed.

This is called *block-stripe* updating, and allows us to not read M repeatedly. Instead it ends up somewhat divided into topic areas!



Shortcomings

1. Measures the generic popularity of a page...
   `www.ipcc.ch` may not be popular overall, but what if you're interested in climate info?

2. Uses a single measure of importance
   Surely there are other ways we could rank web pages for indexing search results?

3. There exist jerks:
   Spammers can create lots of artificial links to their own pages to boost PageRank

"Turns out you're a total jerk and it's just gone undiagnosed all these years."

Shortcomings

1. Measures the generic popularity of a page...
   `www.ipcc.ch` may not be popular overall, but what if you're interested in climate info?
   Solution: topic-specific PageRank (today!!)

2. Uses a single measure of importance
   Surely there are other ways we could rank web pages for indexing search results?
   Solution: hubs-and-authorities (next time!)

3. There exist jerks:
   Spammers can create lots of artificial links to their own pages to boost PageRank
   Solution: Combating link spam: TrustRank (also today!!)

"Turns out you're a total jerk and it's just gone undiagnosed all these years."

Topic-Specific PageRank

What if instead we try to measure a page's popularity within a particular topic?

1. Evaluate web pages both by overall popularity, as well as how "close" they are to a given topic (e.g., "sports", "weather", "movies")
2. Search queries can be answered based on user's interests
3. but... Should the query "sonic" return info about...

3.1 the fast food chain?

3.2 the cartoon hedgehog?

3.3 sound?

Topic-Specific PageRank

Normal PageRank: Each step, some probability of "teleporting" on our random walk.

1. Probability β : follow real link at random.
2. Probability $1 - \beta$: teleport to random page.
3. **All teleport** pages have equal probability

Topic-Specific PageRank

Normal PageRank: Each step, some probability of "teleporting" on our random walk.

1. Probability β : follow real link at random.
2. Probability $1 - \beta$: teleport to random page.
3. **All teleport** pages have equal probability

Topic-specific PageRank: Each step, some probability of "teleporting" on our random walk.

1. Probability β : follow real link at random.
2. Probability $1 - \beta$: teleport to random page.
3. Can only teleport to a **topic-specific** set of relevant pages: some teleport set $\boldsymbol{S}$ that's a subset of all pages.

Topic-Specific PageRank

Topic-specific PageRank: Each step, some probability of "teleporting" on our random walk.

1. Probability β : follow real link at random.
2. Probability $1 - \beta$: teleport to random page.
3. Can only teleport to a **topic-specific** set of relevant pages: some teleport set $\boldsymbol{S}$ that's a subset of all pages.

This **biases** the random walk, or restarts it (from the teleport set S)

1. When the walker teleports, they picks a page from S
2. $\boldsymbol{S}$ contains only pages relevant to the topic (e.g., curated through `https://curlie.org/`)
3. For different teleport sets S, we'll get a different rank vectors r_S

Topic-Specific Computations

Our new algorithm is a change of the transition matrix.

$$\text{Old: } A_{ij} = \beta M_{ij} + \frac{1-\beta}{N}$$

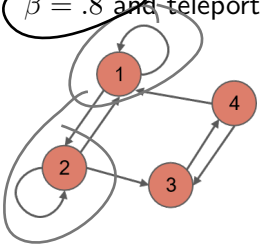
$$\text{New: } A_{ij} = \begin{cases} \beta M_{ij} + \frac{1-\beta}{|S|} & \text{if } i \in S \\ \beta M_{ij} + 0 & \text{else} \end{cases}$$

1. We gave all pages in the teleport set S equal weight $(1/|S|)$
2. and gave pages not in the teleport set 0 weight
3. We could have also assigned *different* weights to different pages

Computation is the same as usual: $r_{new} = M \cdot_{old} + constant$, only the addition is now only nonzero for pages in the teleport set.

Topic-Specific Example

Example: Setup and take a step of topic-specific power iteration for the graph below, with $\beta = .8$ and teleport set $S = \{1, 2\}$.



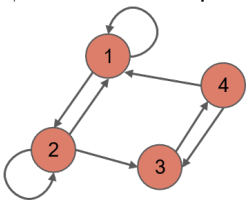
Solution:

$$A = \beta M + \frac{(1-\beta)}{|S|} \cdot S$$

$$= .8 \begin{bmatrix} 1/2 & 1/3 & 0 & 1/2 \\ 1/2 & 1/3 & 0 & 0 \\ 0 & 1/3 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \end{bmatrix} + \frac{.2}{2}$$

Topic-Specific Example

Example: Setup and take a step of topic-specific power iteration for the graph below, with $\beta = .8$ and teleport set $S = \{1, 2\}$.



Solution:

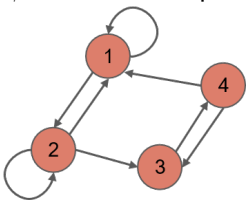
$$S = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

which we combine to:

$$M = \begin{bmatrix} 1/2 & 1/3 & 0 & 1/2 \\ 1/2 & 1/3 & 0 & 0 \\ 0 & 1/3 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Topic-Specific Example

Example: Setup and take a step of topic-specific power iteration for the graph below, with $\beta = .8$ and teleport set $S = \{1, 2\}$.



Solution:

$$S = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

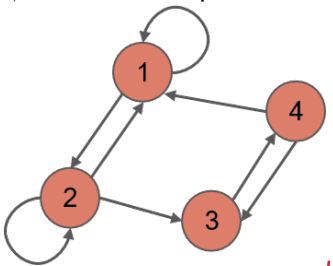
which we combine to:

$$M = \begin{bmatrix} 1/2 & 1/3 & 0 & 1/2 \\ 1/2 & 1/3 & 0 & 0 \\ 0 & 1/3 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$A = 0.8 \begin{bmatrix} 1/2 & 1/3 & 0 & 1/2 \\ 1/2 & 1/3 & 0 & 0 \\ 0 & 1/3 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \end{bmatrix} + \frac{1 - 0.8}{2} \begin{bmatrix} 1/2 & 1/3 & 0 & 1/2 \\ 1/2 & 1/3 & 0 & 0 \\ 0 & 1/3 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Topic-Specific Example

Example: Setup and take a step of topic-specific power iteration for the graph below, with $\beta = .8$ and teleport set $S = \{1, 2\}$.

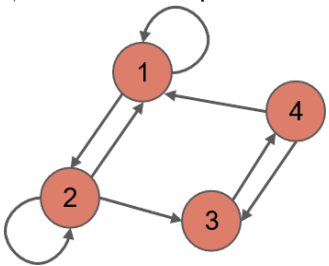


Note that the resulting matrix is still column-stochastic!

$$A = \begin{bmatrix} 5/10 & 11/30 & 1/10 & 5/10 \\ 5/10 & 11/30 & 1/10 & 1/10 \\ 0 & 8/30 & 0 & 4/10 \\ 0 & 0 & 8/10 & 0 \end{bmatrix}$$

Topic-Specific Example

Example: Setup and take a step of topic-specific power iteration for the graph below, with $\beta = .8$ and teleport set $S = \{1, 2\}$.



$$A = \begin{bmatrix} 5/10 & 11/30 & 1/10 & 5/10 \\ 5/10 & 11/30 & 1/10 & 1/10 \\ 0 & 8/30 & 0 & 4/10 \\ 0 & 0 & 8/10 & 0 \end{bmatrix}$$

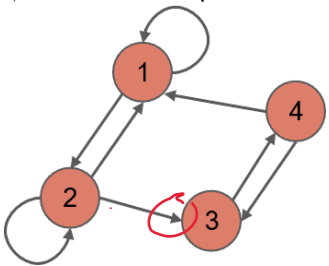
Note that the resulting matrix is still column-stochastic!

Power Iterations yield:

	1	2	3	4
0	0.250	0.250	0.250	0.250
1	0.367	0.267	0.167	0.200
2	0.398	0.318	0.151	0.133
3	0.397	0.344	0.138	0.121
4	0.399	0.351	0.140	0.110
5	0.397	0.353	0.138	0.112
6	0.398	0.353	0.139	0.110
7	0.397	0.353	0.138	0.111
8	0.398	0.353	0.139	0.111
9	0.397	0.353	0.138	0.111
10	0.398	0.353	0.139	0.111

Topic-Specific Example

Example: Setup and take a step of topic-specific power iteration for the graph below, with $\beta = .8$ and teleport set $S = \{1, 2\}$.



$$A = \begin{bmatrix} 5/10 & 11/30 & 1/10 & 5/10 \\ 5/10 & 11/30 & 1/10 & 1/10 \\ 0 & 8/30 & 0 & 4/10 \\ 0 & 0 & 8/10 & 0 \end{bmatrix}$$

Note that the resulting matrix is still column-stochastic!

Power Iterations yield:

	1	2	3	4
0	0.250	0.250	0.250	0.250
1	0.367	0.267	0.167	0.200
2	0.398	0.318	0.151	0.133
3	0.397	0.344	0.138	0.121
4	0.399	0.351	0.140	0.110
5	0.397	0.353	0.138	0.112
6	0.398	0.353	0.139	0.110
7	0.397	0.353	0.138	0.111
8	0.398	0.353	0.139	0.111
9	0.397	0.353	0.138	0.111
10	0.398	0.353	0.139	0.111

Sanity Checks:

Why should 3's PageRank $>$ 4's? Why should 1's PageRank $>$ 2's?

Combating Web Spam

A nice application of topic-specific PageRank is to **combat web spam**.

Here, when we say "spam" we mean pages whose position in search engine rankings has been boosted artificially through deliberate action, incommensurate with the page's real value.

(we aren't talking about junk email or anything like that, for now)

Early spammers...

Want to pretend their site is relevant to common search terms.

Example: suppose you sell T-shirts, but want to appear in a search about "movies".

Combating Web Spam

A nice application of topic-specific PageRank is to **combat web spam**.

Here, when we say "spam" we mean pages whose position in search engine rankings has been boosted artificially through deliberate action, incommensurate with the page's real value.

(we aren't talking about junk email or anything like that, for now)

Early spammers...

Want to pretend their site is relevant to common search terms.

Example: suppose you sell T-shirts, but want to appear in a search about "movies". Use **Term spam** techniques

1. Add the work "movie" many times to your page, but set its color to match the background so only search engines/web crawlers/topic classifying algorithms see it
2. Copy-paste the top-rated "movie" result into your page, then hide it into the background/make it only 1 pixel/etc.

Combating Web Spam

Does this even work? Google's PageRank is based on the *other* pages that link to yours

1. "Believe what people say about you, rather than what you say about yourself"
2. Can look at the words in the anchor text (words underlined in the hyperlink to your page) to see if the page linking to yours thinks you are about movies or not
3. And PageRank is based on the importance of the pages linking into yours

Combating Web Spam

Does this even work? Google's PageRank is based on the *other* pages that link to yours

1. "Believe what people say about you, rather than what you say about yourself"
2. Can look at the words in the anchor text (words underlined in the hyperlink to your page) to see if the page linking to yours thinks you are about movies or not
3. And PageRank is based on the importance of the pages linking into yours

Good: the spammer can't get other legitimate pages to say his page is about movies, so he loses on legitimate PageRank

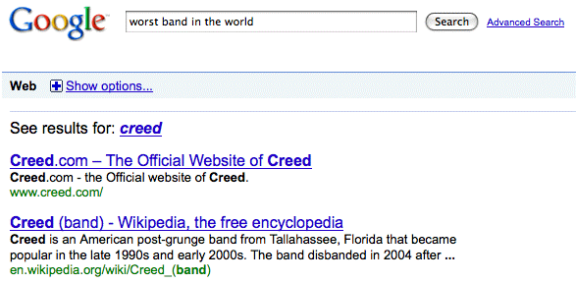
Bad: the spammer could create tons of his own "farm" pages that link to his and include "movie" in the anchor text... but...

Good: farm pages have low PageRank because they would have few/no links into them, so they boost spammer's rank very little.

The good and the bad... and some more bad

(... but kind of funny)

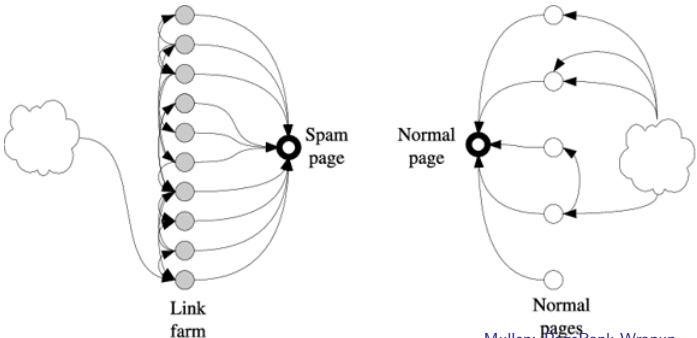
Definition: *Google Bomb*: an attempt to make a search term return a website for an unexpected person or organization when entered in a search engine (typically for satirical or humorous purposes) by the creation of numerous links to that website from pages including the search term in the anchor text.



The good and the bad... and some more bad

So how would a spam to trick PageRank work? A spammer would create lot of pages that *link to* his goal or target page.

Spam farms were developed to concentrate PageRank on a single page. They are meant to create **link spam**, or artificial link structures designed to boost the PageRank of a particular page.



Link Spamming

The addition of spam pages can really wreak havoc on the *teleportation* component of PageRank.

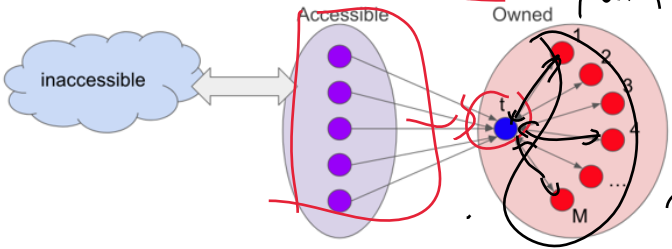
From a spammer's point of view, the web has 3 components:

1. Inaccessible pages (can't control at all)
2. Accessible pages (can post comments/links to his pages, but no other control)
3. Owned pages (completely controlled by the spammer)

Spammer's **goal**: maximize the PageRank of target page t **Technique**:

1. get as many links as possible from accessible pages to t
2. construct a link farm to boost PageRank within the farm, get multiplier effect, and thereby boost PageRank of t

Link Spamming: The Math



form: M pages link to t
rank of t:
each page of the farm:
rank = $\frac{x}{M}$ + teleport

1. Let $x :=$ PageRank contributed to t from *accessible* pages.
2. Let $y :=$ PageRank of t
3. Suppose we build a farm of M pages on a graph of N total nodes. Each farm page will only receive an in-link from t, so each farm page has PageRank of $\frac{\beta y}{M} + \frac{1-\beta}{N}$

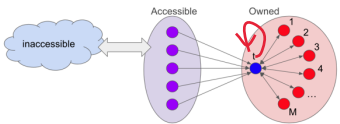
The pagerank y of t is:

$$y = \underbrace{x}_{\text{"legit"}} + \underbrace{\beta M \left[\frac{\beta y}{M} + \frac{1-\beta}{N}\right]}_{farmed} + \frac{1-\beta}{N} \text{ which}$$

we can distribute and rewrite as:

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

Link Spamming: The Math



Solve for y:

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

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

$$\implies y = \frac{x}{1-\beta^2} + \frac{\beta M}{(1+\beta)N} + \underbrace{\frac{1}{(1+\beta)N}}_{small?}$$

$$y \approx \frac{x}{1-\beta^2} + C\frac{M}{N}$$

So how big is this? It depends on two major terms: the ration M/N and the teleport probability.

Link Spamming: The Math

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

1. Since x was the non-farmed rank of t, we call $1/(1-\beta^2)$ the *multiplier* effect of the farm. For example, $\beta = 0.85$ leads to a multiplier of 3.6.
2. By making M large, we can also make y nearly as large as we want!
   - N is enormous though, so M won't ever truly dominate the internet as a whole...
   - but it doesn't have to: it only has to dominate similar results!
3. *Both* $\frac{1}{1-\beta^2}$ and $C = \frac{\beta}{(1-\beta)}$ get larger as $\beta \to 1$. This is because our random walker can get *completely stuck* in the spam farm: it's a giant spider trap!... but it also owns more of the teleport set than it should, since it's M/N proportion of the internet as a whole.

So how do we fix that?

Fighting Link Spam

Option 1: we could try to detect link structures that look like a spam farm.

...but this is a game of cat-and-mouse between search engines and spammers... blech.

Option 2: instead of regular ol' PageRank, use **TrustRank**

Trustrank: topic-specific PageRank where the teleport set = set of *trusted* pages

- E.g., set of all .edu, .gov, .mil pages, or human-curated
- Basic idea: it should be rare for a "good" page to link to a "bad" page

TrustRank: Who to Trust?

TrustRank: topic-specific PageRank where the teleport set = set of *trusted* pages. The challenge is picking the *right set* of **seed pages** or **trusted** pages to be the teleport set.

1. It could be human-curated. But that's labor-intensive since a human has to identify them! We'd end up with a small seed set.
2. We *want* a larger seed set, since we want to ensure that *every* good page has a high enough TrustRank to be at least reachable from the seed set's random walk.
3. One option: run standard PageRank and only take the set of the top k pages, since it's unlikely a spammer can inflate a spam page's range into that top echelon.
4. Another: specify trusted *domains* like .edu, .mil, .gov, etc.

TrustRank Properties

Designating a smaller set to be the authorities on a topic is very similar to local search variants of PageRank.

Topic-Specific PageRank: teleport set has the *expertise* on that topic

- we propagate that expertise throughout the network when we iterate the power method
- pages close to the teleport set have more expertise, pages farther away have less

TrustRank: teleport set has the *trust*

- we propagate that trustworthiness throughout the network:
  run a topic-specific PageRank where the teleport set $=$ trusted pages set
- Implementation note: we can actually start with all trust $= 1$ (benefit of the doubt)...
  The resulting PageRanks are the TrustRanks (slightly different b/c won't sum to 1 if initially all pages have trust $= 1$)
- pages closer to trusted set have more trust, farther away has less

Qualifying Spam

If our goal is to describe whether or not a page is spam, we need a statistic or measure for it!

1. **Option 1**: Pick a threshold TrustRank value; all pages below the threshold marked as spam.
   This risks marking new pages as spam though, among other false positive type errors
2. **Option 2**: With TrustRank, we start with trusted pages and propagate that trust
   Turn it around: Can we estimate the fraction of a page's PageRank comes from spam pages?

Qualifying Spam

If our goal is to describe whether or not a page is spam, we need a statistic or measure for it!

1. **Option 1**: Pick a threshold TrustRank value; all pages below the threshold marked as spam.
   This risks marking new pages as spam though, among other false positive type errors
2. **Option 2**: With TrustRank, we start with trusted pages and propagate that trust
   Turn it around: Can we estimate the fraction of a page's PageRank comes from spam pages?

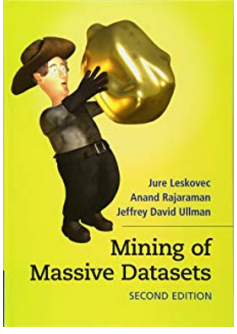
Define:
r_p = PageRank of page p, and r_p^+ = PageRank with teleport into **trusted pages only** r_p^- = PageRank of page p from "spam" pages $r_p^- = r_p - r_p^+$

Definition: The *spam mass* of page p is $\frac{r_p^-}{r_p}$.Pages with sufficiently high spam mass are marked as spam

Acknowledgments

Next time: More on graphs: making out-links count, too!

Some material is adapted/adopted from Mining of Massive Data Sets, by Jure Leskovec, Anand Rajaraman, Jeff Ullman (Stanford University) http://www.mmds.org



Special thanks to Tony Wong for sharing his original adaptation and adoption of slide material.