CMSC5741 Big Data Tech. & Apps.

Lecture k Analysis

Prof. Michael R. Lyu
Computer Science & Engineering Dept.
The Chinese University of Hong Kong

What's the Mechanism Distinguished Google?

How Google return such kind of rankings (e.g. Charles Kao Wikipedia first then his NobelPrize. Even his passing away)?

One important factor is PageRank score.

How to make PageRank computation scalable

- There are billions of web pages and hyperlinks between them, how to compute their ranking score (e.g., PageRank) efficiently?

Outline

- Web as a Graph
- PageRank
- Topic-Specifi
- Appendix: Trust-Rank

Outline

- Web as a Graph
- PageRank
- Topic-Specifi
- Appendix: Trust-Rank

Web as a Graph

- Web as a directed graph:
  - Nodes: Web pages
  - Edges: Hype

Web as a Directed Graph

Broad Question

- How to organize the Web?
- First try: Human curated Web directories
  - Yahoo, DM
- Second try: Web Search
  - Information Retrieval inv : Find relevant docs in a small and trusted set
    - Newspaper articles, Patents, etc.
  - But: Web is huge, full of untrusted documents, random things, web spam, etc.

Web Search: 2 Challenges

- Web contains many sources of information: Who to "trust"?
  - Trick: Trust oint to each other
- What is the " uery "newspaper"?
  - No single right answer
  - Trick: Pages that actually know about newspapers might all be pointing to many newspapers

Ranking Nodes on the Graph

- All web pages are not equally "important"
  - www.cuhk.
    www.joe-sc
- There is large diversity i web-graph node connectivity.
  - Rank the pages by the link structure!

Link Analysis Algorithms

- We will cover the following Link Analysis approaches for computing importance of nodes in a gr
  - PageRank
  - Topic-Specific (Personali – Rank
  - Web Spam Detection Algorithms, e.g. TrustRank

Outline

- Web as a Graph
- PageRank
- Topic-Specifi
- Appendix: Trust-Rank

Links as Votes

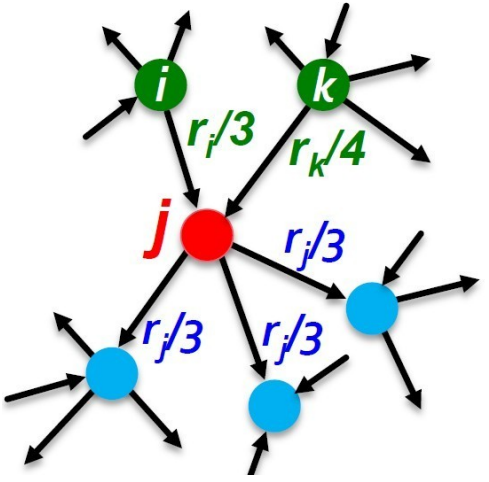
- Idea: Links as votes
  - Page is more important if it has more links
    - In-coming
- Think of in-li
  - www.cuhk.edu.hk has 1 in-link
  - www.joe-schmoe.com has 1 in-link
- Are all in-links equal?
  - Link from important pages count more
  - Recursive question!

Example: PageRank Scores

Simple Recursive Formulation

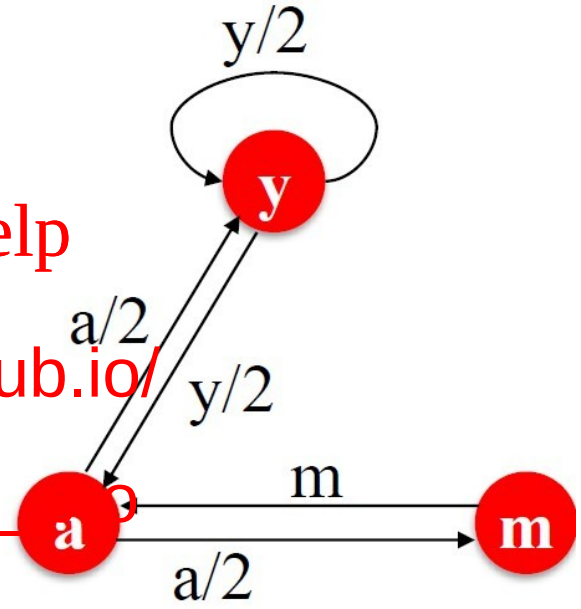
- Each link's vote is proportional to the importance of its source page
- If page *j* with *n* out-links, each link get
- Page *j*'s own importanc um of the votes on its in-links

$$r_j = r_i/3 + r_k/4$$



PageRank: The "Flow" Model

- A "vote" from an important page is worth more
- A page is im
  pointed to b
  important pages r_j
- Define a "rank" for page *j*



"Flow" equations:

$$r_y = r_y/2 + r_a/2$$
$$r_a = r_y/2 + r_m$$
$$r_m = r_a/2$$

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

$d_i \ldots$ out-degress of node i

Solving the Flow Equations

- 3 equations, 3 unknowns,

  no constants
  - No unique solution
  - All solutions – scale factor
- Additional constraint forces uniqueness:
  - $r_y + r_a + r_m = 1$
  - Solution: $r_y = \frac{2}{5}, r_a = \frac{2}{5}, r_m = \frac{1}{5}$
- Gaussian Elimination method works for small examples, but we need a better method for large web-size graphs

PageRank: Matrix Formulation

- Stochastic adjacency matrix M
  - Let page i have d_i out-links
  - If $i \to j$, th
    - M is a colu mns sum to 1
- Rank vector r: vector wi try per page
  - r_i is the importance score of page i
  - $\sum_i r_i = 1$
- The flow equations can be written $r_j = \sum_{i \to j} \frac{r_i}{d_i}$

$$r = M \cdot r$$

Example

- Remember the flow equation: $r_j = \sum_{i \to j} \frac{r_i}{d_i}$
- Flow equation in the matrix form:
  - Suppose page i links to 3 cluding j

Eigenvector Formulation

- The flow equations can be written $r = M \cdot r$
- So the rank vector r is an eigenvector of the stochastic w
  - In fact, its fi vector, with corresponding eigenvalu
    - Largest eigenvalue of $\boldsymbol{M}$ is **1** since $\boldsymbol{M}$ is column stochastic
- We can now efficiently solve for r!
  - The method is called Power iteration.

NOTE: $\boldsymbol{x}$ is an eigenvector with the corresponding eigenvalue λ if:

$$\boldsymbol{Ax} = \boldsymbol{\lambda x}$$

Example: Flow Equation & M

Power Iteration Method

- Given a web graph with *n* nodes, where the nodes are pages and edges are hyperlinks
- Power Iterati tive scheme
  - Suppose th
  - Initialize:
  - Iterate: $\mathbf{r}^{(t+1)} = \mathbf{M} \cdot \mathbf{r}^{(t)}$
  - Stop when $|\mathbf{r}^{(t+1)} - \mathbf{r}^{(t)}|_1 < \epsilon$
    - $|\mathbf{x}|_1 = \sum_{1 \le i \le N} |x_i|$ is the L1 norm

$$r_j = \sum_{i \to j} \frac{r_i}{d_i}$$

$d_i \ldots$ out-degress of node i

Why Power Iteration Works?

- Power iteration:
  - A method for finding dominant eigenvector (the vector corre est eigenvalue)
    -
    -
    - $\mathbf{r^{(3)}} = \mathbf{M} \cdot \mathbf{r^{(2)}} = \mathbf{M}(\mathbf{M^2 r^{(0)}}) = \mathbf{M^3} \cdot \mathbf{r^{(0)}}$
- Claim:
  - Sequence $\mathbf{M} \cdot \mathbf{r^{(0)}}, \mathbf{M^2} \cdot \mathbf{r^{(0)}}, \ldots, \mathbf{M^k} \cdot \mathbf{r^{(0)}}, \ldots$ approaches the dominant eigenvector of ***M***

Why Power Iteration Works?

- Proof:
  - Assume ***M*** has ***n*** linearly independent eigenvectors x^1, x^2, …, x^n with corresponding eigenvalues
  - Vectors x_1, x_2, … x_n form a d thus we can write:

$$\begin{aligned} Mr^{(0)} &= M(c_1x_1 + c_2x_2 + \ldots + c_nx_n) \\ &= c_1(Mx_1) + c_2(Mx_2) + \ldots + c_n(Mx_n) \\ &= c_1(\lambda_1x_1) + c_2(\lambda_2x_2) + \ldots + c_n(\lambda_nx_n) \end{aligned}$$

  - Repeated multiplication on both sides:

$$M^k r^{(0)} = c_1(\lambda_1^k x_1) + c_2(\lambda_2^k x_2) + \ldots + c_n(\lambda_n^k x_n)$$

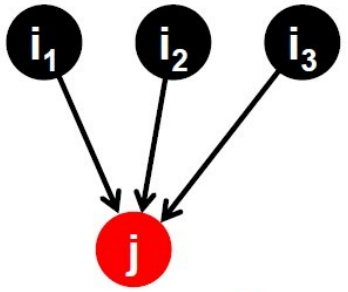
Why Power Iteration Works?

- Proof: (cont.)
  - Repeated multiplication on both sides produces

  - Since $\lambda_1 > \lambda_2$ then
  - Thus $M^k r^{(0)} \approx c_1(\lambda_1^k x_1)$
    - Note if $c_1 = 0$, then the method won't converge

Random Walk Interpretation

- Imagine a random web surfer:
  - At any time t, surfer is on some page *i*
  - At time t+1, the surfer follows an out-link from *i* uniformly at
  - Ends up on some page *j* l *i*
  - Process repeats indefinitely
- Let:
  - p(t) … vector whose i^{th} coordinate is the prob. that the surfer is at page *i* at time *t*
  - So p(t) is a probability distribution over pages



$$r_j = \sum_{i \to j} \frac{r_i}{\mathrm{d}_{\mathrm{out}}(\mathrm{i})}$$

The Stationary Distribution

- Where is the surfer at time t+1
  - Follows a link uniformly at random

- Suppose the random w es a state

$$p(t+1) = M \cdot p(t) = p(t)$$

Then p(t) is stationary distribution of a random walk

- Our original rank vector r satisfies $r = M \cdot r$
  - So r is a stationary distribution for the random walk

PageRank

- Three questions:
  - Does this converge?
  - Does it con ant?
  - Are results reasonable

Does This Converge?

Does it Converge to What We Want?

PageRank: Problems

- Two problems:
  - Spider traps: all out-links are within the group
    - Eventuall ll importance
  - Some page t-links)
    - Such pages cause impo leak out"

Problem: Spider Traps

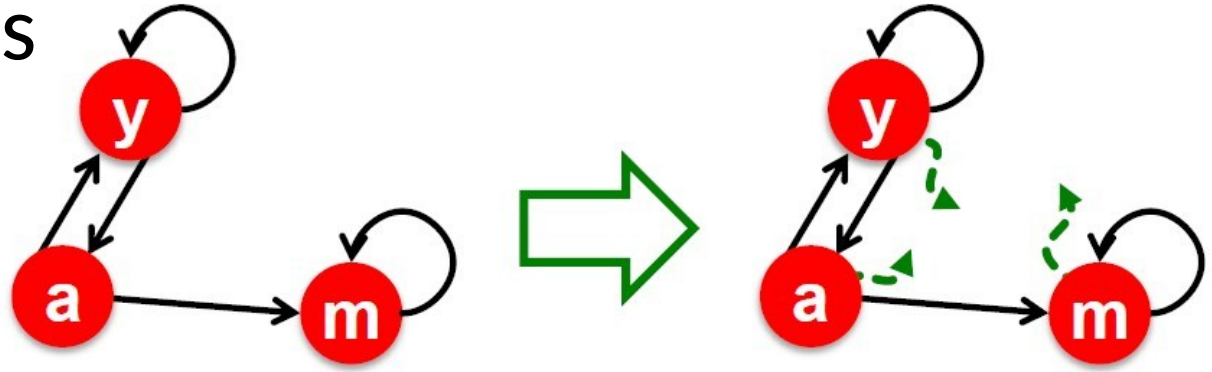
- Power Iteration:
  - Set $r_j = 1$
  -
    - And iterat
- Example

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \ldots & 0 \\ 1/3 & 3/6 & 7/12 & 16/24 & & 1 \end{matrix}$$

Iteration 0, 1, 2, …

Solution: Random Teleports

- The Google solution for spider traps: At each time step, the random surfer has two options
  - With prob. β, follow a link at random
  - With prob. $1-\beta$, jump to a random page
  - Common values for β are 0.8 to 0.9
- Surfer will teleport out of spider trap within a few time steps



Problem: Dead Ends

- Power Iteration:
  - Set $r_j = 1$
  -
    - And iterat
- Example

$$\begin{bmatrix} r_y \\ r_a \\ r_m \end{bmatrix} = \begin{matrix} 1/3 & 2/6 & 3/12 & 5/24 & & 0 \\ 1/3 & 1/6 & 2/12 & 3/24 & \ldots & 0 \\ 1/3 & 1/6 & 1/12 & 2/24 & & 0 \end{matrix}$$

Iteration 0, 1, 2, …

Solution: Always Teleport

- Teleports: Follow random teleport links with probability 1.0 from dead-ends
  - Adjust matr

Why Teleports Solve the Problem?

$$\mathbf{r}^{(\mathbf{t+1})} = Mr^{(t)}$$

- Markov chai

  - Set of states **X**

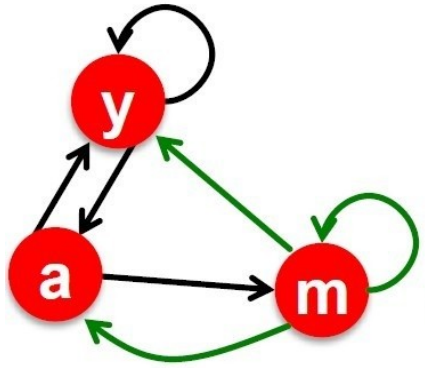
  - Transition matrix **P** wher $P_{ij} = P(X_t = i | X_{t-1} = j)$
  - π specifying the stationary probability of being at each state $x \in X$
  - Goal is to find π such that $\pi = P\pi$

Why Is This Analogy Useful?

- Theory of Markov chains
- Fact: For any start vector, the power method applied to a matrix **P** will converge to a unique stationary vector as long as **P** is stochastic and irreducible and aperiodic.

Make M Stochastic

- Stochastic: Every column sums to 1
- A possible solution: add green links



	y	a	m
y	½	½	1/3
a	½	0	1/3
m	0	½	1/3

$$r_y = r_y/2 + r_a/2 + r_m/3$$
$$r_a = r_y/2 + r_m/3$$
$$r_m = r_a/2 + r_m/3$$

Make M Aperiodic

- A chain is periodic if there exists **k > 1** such that the interval between two visits to some state **s** is alw .
- A possible so ks

Make M Irreducible

- From any state, there is a non-zero probability of going from any one state to any another
- A possible so links

Solution: Random Jumps

- Google's solution that does it all:
  - Makes M stochastic, aperiodic, irreducible
- At each step, s two options:
  - With probability β, follo t random
  - With probability $1-\beta$, ju e random page
- PageRank equation [Brin-Page,98]

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

This formulation assumes that **M** has no dead ends. We can either preprocess matrix **M** to remove all dead ends or explicitly follow random teleport links with probability 1.0 from dead-ends.

The Google Matrix

- PageRank equation [Brin-Page,98]

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

- The Google

- A is stochastic, aperiodic and irreducible, so

$$\mathbf{r}^{(\mathbf{t+1})} = \mathbf{A} \cdot \mathbf{r}^{(\mathbf{t})}$$

- In practice $\beta = 0.8, 0.9$ (make 5 steps and jump)

In-class Practice

- Go to Practice

Computing PageRank

- Key step is matrix-vector multiplication
  - $r^{new} = A \cdot r^{old}$
- Easy if we ha emory to hold A, r^{old}, r^{new}
- Say *N*=1 billion pages
  - We need 4 bytes for each entry (say)
  - 2 billion entries for vectors, approx. 8GB
  - Matrix *A* has N^2 entries: 10^{18} is a large number!

Matrix Formulation

- Suppose there are ***N*** pages
  - Consider page j, with d_j out-links
  - We have M_{ij} and $M_{ij} = 0$ otherwise
- The random teleport is nt to:
  - Adding a teleport link from j to every other page and setting transition prob. to $(1-\beta)/N$
  - Reducing the prob. of following each out-link from $1/|d_j|$ to $\beta/|d_j|$

Rearranging the Equation

Note: Here we assumed **M** has no **dead-ends**

$[x]_N$ … a vector of length *N* with all entries x

Spare Matrix Formulation

- We just rearranged the PageRank equation

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

- M is a sparse ead-ends)
  - 10 links per node, appro ries

- So in each iteration, we need to
  - Compute $r^{new} = A \cdot r^{old}$
  - Add a constant $(1-\beta)/N$ to each entry in r^{new}
    - Note: if *M* contains dead-ends then $\sum_i r_i^{new} < 1$ and we also have to renormalize r^{new} so that it sums to 1

PageRank: The Complete Algorithm

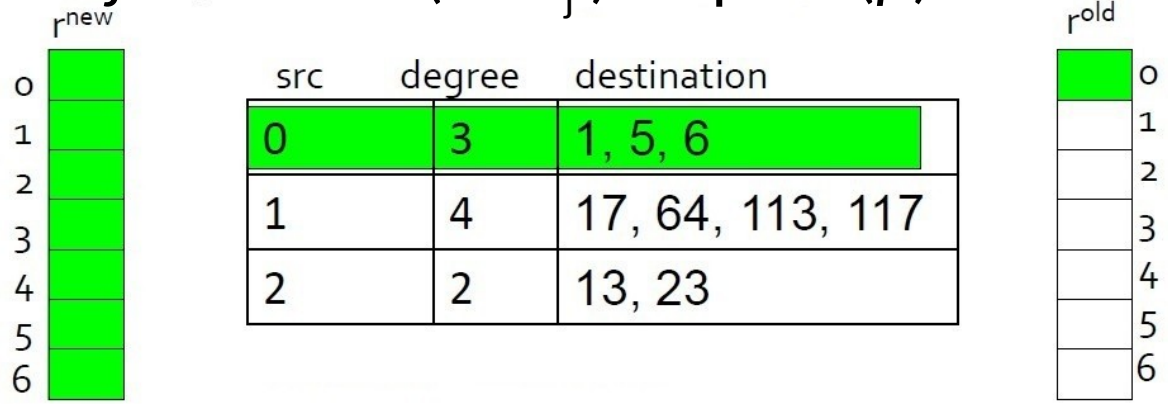
Sparse Matrix Encoding

- Encode sparse matrix using only nonzero entries
  - Space prop umber of links
  - Say 10N, or B
  - Still won't fit in memory t on disk

source node	degree	destination nodes
0	3	1, 5, 7
1	5	17, 64, 113, 117, 245
2	2	13, 23

Basic Algorithm: Update Step

- Assume enough RAM to fit r^{new} into memory
  - Store r^{old} and matrix *M* on disk
- Then 1 step of power iteration is:
  - Initialize all
  - For each page p (of out-
    - Read into memory: *p*, *n*, st$_1$ st$_n$, r^{old} (*p*)
    - For *j*=1…*n*: r^{new} (dest$_j$) += β r^{old} (*p*)/*n*



Analysis

- Assume enough RAM to fit r^{new} into memory
  - Store r^{old} and matrix M on disk
- In each itera
  - Read r^{old} and M
  - Write r^{new} back to disk
  - IO cost = $2|r| + |M|$
- Question:
  - What if we could not even fit r^{new} in memory

Block-based Update Algorithm

Analysis of Block Update

- Similar to nested-loop join in databases
  - Break r^{new} into *k* blocks that fit in memory
  - Scan *M* and lock
- *k* scans of M
  - $k(|M|+|r|) + |r| = k|M| + |r|$
- Can we do better?
  - Hint: *M* is much bigger than *r* (approx. 10-20x), so we must avoid reading it *k* times per iteration

Block-Strip Update Algorithm

Block-Strip Analysis

- Break *M* into stripes
  - Each strip contains only destination nodes in the correspondi
- Some additio stripe
  - But it is usually worth it
- Cost per iteration

$$|M|(1+\epsilon) + (k+1)|r|$$

Some Problems with PageRank

- Measures generic popularity of a page
  - Biased against topic-specific authorities
  - Solution: To k (next)
- Susceptible t
  - Artificial link topographi in order to boost page rank
  - Solution: TrustRank (next)
- Uses a single measure of importance
  - Solution: Hubs-and-Authorities (in further reading)

Outline

- Web as a Graph
- PageRank
- Topic-Specifi
- Appendix: Trust-Rank

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, history and computer security

Topic-Specific PageRank

- Random walker has a small probability of teleporting at any step
- Teleport can
  - Standard Pa ith equal probability
    - To avoid dead-end and spider-trap problems
  - Topic Specific PageRank: A topic-specific set of "relevant" pages (teleport set)

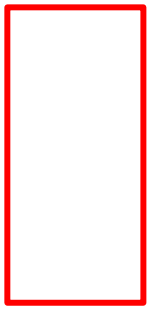
Topic-Specific PageRank

- Idea: Bias the random walk
  - When walker teleports, she picks a page from a set S
  - S contains only pages that are relevant to the topic
    - e.g., Open D on a given topic/query
  - For each teleport set S, w erent vector r_s

Matrix Formulation

- To make this work all we need is to update the teleportation part of the PageRank formulation:
  - A is stochastic!
- We have weighted all pages in the teleport set S equally
  - Could also assign different weights to pages!
- Compute as for standard PageRank

Example



Discovering the Topic

- Create different PageRanks for different topics
  - The 16 DMOZ top-level categories: arts, business, sports, …
- Which topic
  - User can pick from a me
  - 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
  - User context, e.g., user's bookmarks,…

SimiRank: An Application of Personalized PageRank

- SimRank: Random walks from a fixed node on k-partite graphs
- Setting: k-pa types of nodes
  - Example: pi nodes
- How to find nodes simil e *u*?
- Do a Random-Walk with Restarts from node *u*
  - i.e. teleport set $S = \{u\}$

SimiRank(cont.)

- Resulting scores measures similarity/proximity to node *u*
- Generally ap etworks (typically un
- Problems:
  - Must be done once for each node *u*
  - Suitable for sub-Web-scale applications

SimRank: Example

SimRank: Example

Outline

- Web as a Graph
- PageRank
- Topic-Specifi
- Appendix: Trust-Rank
- Skip to conclusion

What is Web Spam?

- Spamming:
  - Any deliberate action to boost a web page's position in s , incommensurate with page's
- Spam:
  - Web pages that are the result of spamming
- Approximately 10-15% of web pages are spam

Web Search

- Early Search Engines
  - Crawl the Web
  - Index page contained
  - Respond to search que pages containing those words

Web Search

- Early page ranking
  - Attempt to order pages matching a search query by "i
  - First search :
    - Number of times query eared
    - Prominence of word position, e.g. title, header

First Spammers

- Those with commercial interests tried to exploit search engines to bring people to their own site – w d 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?
  - Add the wo to your page, set text color t or, so only search engines would see it
  - 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 ords that appear und t the link) and its surrounding text
  - PageRank as a tool to measure the "importance" of Web pages

Why It Works?

- Our hypothetical shirt-seller looses
  - Saying he is about movies doesn't help, because others don't say he is about movies, his page isn't very import nked high for shirts or mo
- 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

Spam Farms

- Once Google became the dominant search engine, spammers began to work out ways to fool Google
- Spam farms
  PageRank on a single pa
- 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
  - Accessible p
    - E.g. blog comments page
    - 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 pages as possible to target page *t*
  - Construct "link farm" to get PageRank multiplier effect

Link Farms

Analysis

- *x*: PageRank cont ges
- *y*: PageRank of target page t
- Rank of each "farm" page = $\frac{\beta y}{M} + \frac{1-\beta}{N}$
-

$$
\begin{aligned}
y &= x + \beta M\left[\frac{\beta y}{M} + \frac{1-\beta}{N}\right] + \frac{1-\beta}{N} \\
&= x + \beta^2 y + \frac{\beta(1-\beta)M}{N} + \frac{1-\beta}{N} \\
&= \frac{x}{1-\beta^2} + c\frac{M}{N}, where\, c = \frac{\beta}{1+\beta}
\end{aligned}
$$

Very small: ignore now we solve for y

Analysis

- Multiplier effect for “acquired” PageRank
- By making *M* large, we can make y as large as we want

TrustRank: Combating Spam

- Combating term spam
  - Analyze text using statistical methods
  - Similar to email spam filtering
  - Also useful: D plicate 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) pag
- Sample a set eb
- Have an oracle (human) fy 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 to Rank with teleport set
  - Propagate trust through
    - Each page gets a trust value between 0 and 1
- Use a threshold value and mark all pages below the trust threshold as spam

Why is It a Good Idea?

- Trust attenuation:
  - The degree of trust conferred by a trusted page decreases w e graph
- Trust splittin
  - The larger the number o s 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
  - Must ensur ts adeq rank, so need make all g s reachable from seed set by short paths

Approaches to Picking Seed Set

- Suppose we want to pick a good set of *k* pages
- How to do that?
- PageRank
  - Pick the top k pages by P
  - Theory is that you can't page's rank really high
- 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.
- Complement
  - What fracti ank 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^+ = PageRank of p with teleport into trusted pages only
- Then: What fraction of a page's PageRank comes from spam pages?

$$r_p^- = r_p - r_p^+$$

- Spam mass of

$$p = \frac{r_p^-}{r_p}$$

One-slide Takeaway

- Web as a Graph
  - Denote the web structure as a graph
- PageRank
  - PageRank s importance of web pages
- Topic-Specific PageRank
  - Evaluate web pages by their popularity as well as particular topic
- Trust-Rank
  - Deal with link spams

Further Reading

- Original PageRank paper: http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf
- An Analytical proaches to Personalizin http://www-cs-student d.edu/~taherh/papers/comparison.pdf

Further Reading

- HITS algorithm: http://www.cs.cornell.edu/home/kleinber/auth.pdf
- Parallel Page http://link.sp nt/pdf/10.1007%2F11735106_22.pdf

Reference

- http://www.stanford.edu/class/cs246/slides/09-pagerank.pdf
- http://www.stanford.edu/class/cs246/slides/10-spam.pdf
- http://i.stanford.edu/~ullman/mmds/ch5.pdf
- http://en.wikipe
- http://en.wikipedia.org/wiki/HI m

In-class Practice

- Compute the final PageRank Score of the given graph, with Google matrix A and assume $\beta = 0.8$
- Show the matrix A, and solve by both Gaussian Elimination and Power Iteration methods