What is this Page Known for? Computing Web Page Reputations

Alberto Mendelzon

University of Toronto

http://www.cs.toronto.edu/~mendel

Joint work with Davood Rafiei

Outline

- Scenarios and motivation
- Three definitions of rank

  Page Rank

  Reputation Measure

  Hubs and Authorities
- The TOPIC prototype
- Future work

Scenarios

- Search engine Search-U-Matic just returned 60,000 pages on the query “liver disease.” Where should I start looking?
- We’re spending $200K/year maintaining our web pages. What topics are they known for?
- Prof. X, an expert on Icelandic sagas, is up for tenure. I wonder how well known her research is on the Web.
- How does our site rank in popularity among all the Linux sites?

Idea:

- analyze links to find pages that are better/better known/more authoritative than others *on some topics*

Defining Rank

- **Citation analysis**: *Rank(p)* = number of papers that cite paper p.
- On the web, citation = link. Just use *in-degree* of a node in the web graph.

Problems:

- All links are not created equal. Yahoo is much better maintained than my home page
- Topic independent: high rank on “Gilligan’s Island” doesn’t imply high rank on “brain surgery.”

Definition 1: Page Rank

(Brin and Page 1998, Google; Geller 1978 in bibliometrics)

- Problem: given page *p*, compute its rank

A page is good if lots of good pages point to it.

One level random walk model:

At each step:

- with prob d>0 jump to a random page, or

- with prob (1-d) follow a random link from the current page

Page Rank of page p = probability, in the limit, of hitting page p

Page Rank Equation

$$R(p) = (1-d)\sum_{q \to p} \frac{R(q)}{Out(q)} + \frac{d}{N}$$

Computed by iterative method during crawling

- Limitation:

  query and **topic**-independent

Definition 2: Reputation Measurement

- Problem: Given page p and topic t, compute the rank of p on t, $RM(p,t)$

  Let $I(t,p)$ = number of pages on topic t that point to p

  Let N_t = number of pages on topic t

$$RM(p,t) = I(t,p) / N_t$$

- Compute:

  With search engine, queries "`+link:p +t`" and "`+t`"

Definition 3: Hubs and Authorities

(Kleinberg, 1998)

- Problem: Given topic t, find pages p with high rank on t

A page is a good hub for t if it points to good authorities on t

A page is a good authority on t if good hubs for t point to it

Algorithm to find authorities on t:

- Issue the query “t” to a search engine
- Take the first N answers, add pages at distance 1
- Compute hubs and authorities for t within this set

A two-level random walk model

- with probability d>0 jump to random page that contains term t
- with probability (1-d) follow random link **forward/backward** from the current page, alternating directions

Pages accumulate

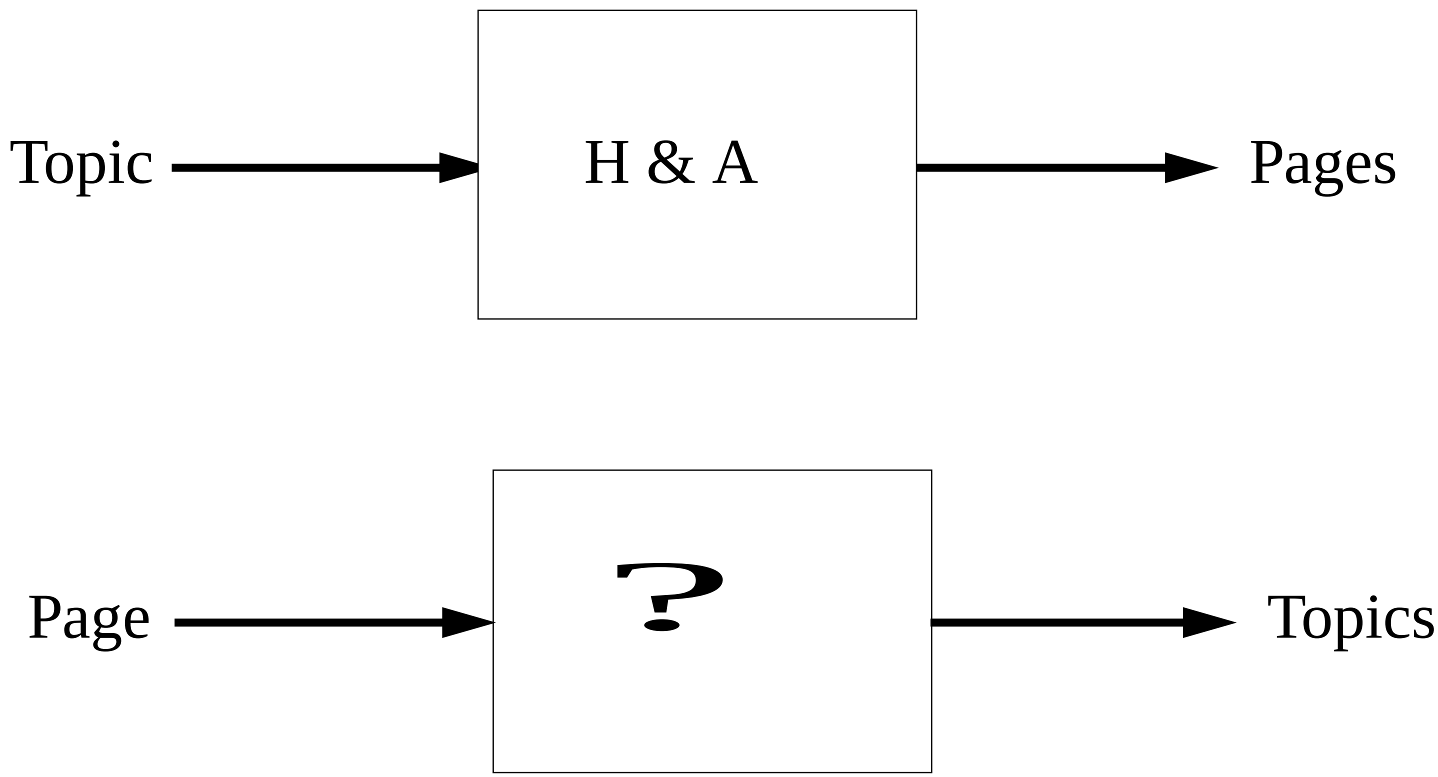
- forward visits
- backward visits

- **A(p,t)** = probability of a forward visit to page p when searching for term t = **Authority rank** of page p on term t

- **H(p,t)** = probability of a backward visit to page p when searching for term t = **Hub rank** of page p on term t

Theorem If $d>0$, the two-level random walk has unique stationary probability distributions A(p,t) and H(p,t).

Inverting H&A computation



Two Solutions

- *Search engine solution*: a large crawl of the web is available. Find authorities on t for each term t
- *Real-time solution*: approximate the search engine solution by starting with some set of pages and the terms that appear in them, and iteratively expanding this set

Search Engine Solution (bottom up)

For every page p and term t

$$A(p,t) = H(p,t) = \frac{1}{2N_t}, \text{ if t appears in p}$$

$$A(p,t) = H(p,t) = 0 \text{ otherwise.}$$

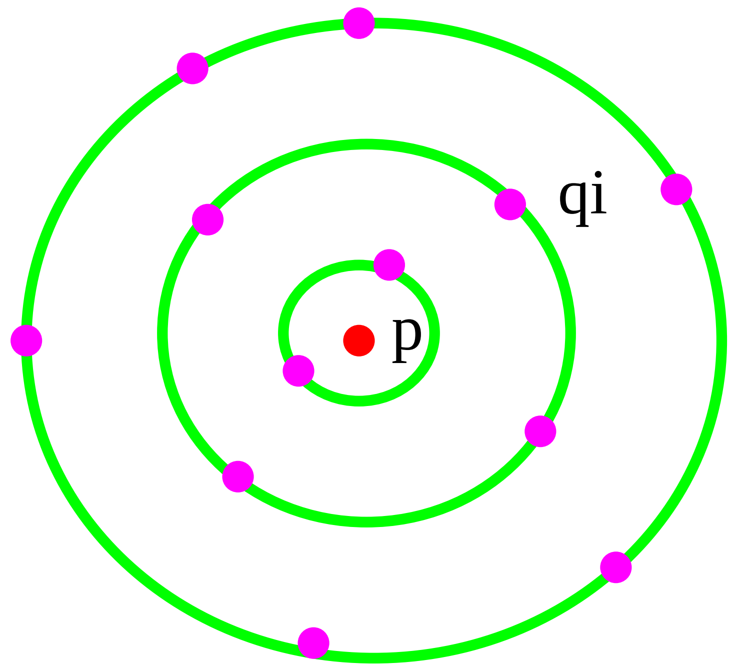
While changes occur

$$A(p,t) = (1-d)\sum_{q \to p} \frac{H(q,t)}{Out(q)} + \begin{cases} \dfrac{d}{2N_t} & \text{if t appears in page p;} \\ 0 & \end{cases}$$

$$H(p,t) = (1-d)\sum_{p \to q} \frac{A(q,t)}{In(q)} + \begin{cases} \dfrac{d}{2N_t} & \text{if t appears in page p} \\ 0 & \end{cases}$$

Real-time Solution: **(top down)**

Set of pages:



Set of terms: all terms t that appear in p or some of the qi’s

Real-time algorithm (Using the one-level model for simplicity)

$$R(p, t) = \frac{d}{N_t}$$

For $i = 1,2,...,k$

For each path $q_1 \rightarrow q_2 \rightarrow \ldots \rightarrow q_i \rightarrow p$,

For each term t in page q_1

$$R(p, t) = R(p, t) + \left(\frac{(1-d)^i}{\prod_{j=1}^{i} Out(q_i)} \right) \frac{d}{N_t}$$

Simplification

k=1, $Out(q)$ = constant

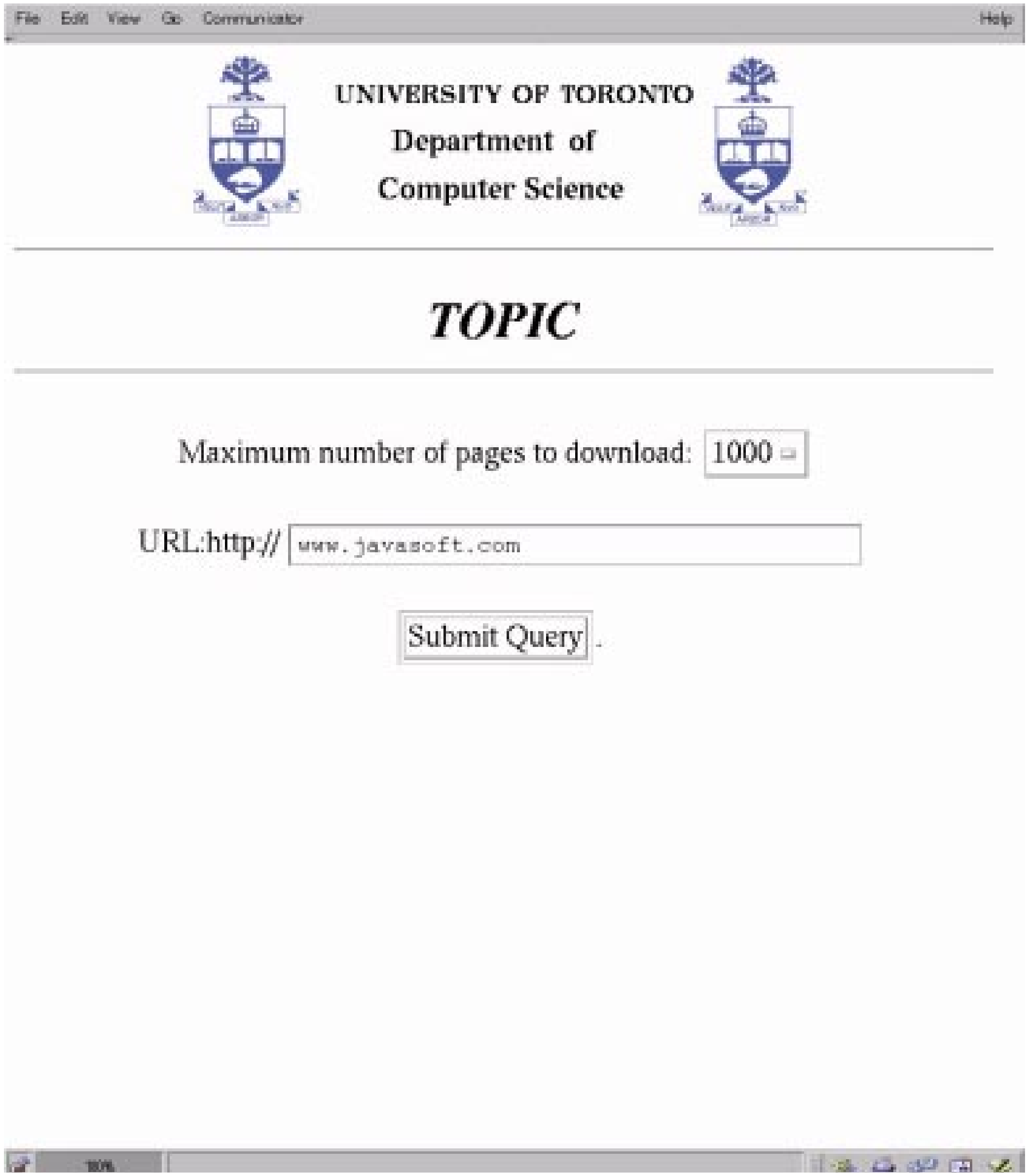
$$R(p,t) = C \times \sum_{q \to p} \frac{1}{N_t}$$

That is, $R(p,t) \sim I(t,p) / N_t$ (Definition 2)

TOPIC (TOronto Page Influence Computation)

- A crude approximation:
- Given page p
  - Find 1,000 pages q that link to p (using Altavista)
  - From each q “snippet,” extract all terms t
  - Remove internal links and duplicate snippets
  - Remove stop words and rare terms
  - Apply the real-time algorithm with d = 0.10, k = 1, Out(q) = 7.2

www.cs.toronto.edu/db/topic



Example

- www.macleans.ca

1.Maclean's Magazine
2.macleans
3.Canadian Universities

Example: authorities on (+censorship +net)

- www.eff.org

  Anti-censorship, Join the Blue Ribbon, Blue Ribbon Campaign, Electronic Frontier Foundation

- www.cdt.org

  Center for Democracy and Technology, Communications Decency Act, Censorship, Free Speech, Blue Ribbon

- www.aclu.org

  ACLU, American Civil Liberties Union, Communications Decency Act

Example: Personal Home Pages

- www.w3.org/People/Berners-Lee

  History of the Internet, Tim Berners-Lee, Internet History, W3C

- www-db.stanford.edu/~ullman

  Jeffrey D. Ullman, Database Systems, Data Mining, Programming Languages

- www.neci.nj.nec.com/homepages/giles.html

  Lee Giles, Neural Networks, Machine learning

- www-cs-faculty.stanford.edu/~knuth

  Don Knuth, TeX Users, LaTex, Linux, CTAN

Example: Institutional Home Page

- www.almaden.ibm.com:

  IBM Almaden Research Center, Data Mining, Visualization, ACM, guide, scanning

- www.research.microsoft.com:

  Knowledge Discovery, Download, Data Mining, Computer Vision, Language, ACM, Computer Science, Artificial

Example: Institutional Home Page

- www.neci.nj.nec.com

  Watermarking

  Search engines

  Computer vision

  Neural networks

  Othello

Example: Canadian CS Departments

www.cs.toronto.edu (8400)

Russian History, Neural, Travel, Hockey

www.cs.utoronto.ca (3644)

Search Engines, Ice Hockey, League, Neural, Neural Networks

www.cs.ualberta.ca (10557)

University of Alberta, Virtual Reality, Language, Chess, Artificial

www.cs.ubc.ca (17598)

Confocal, Periodic Table, Anime, Computer Science, Manga

www.cs.sfu.ca (2055)

Whales, Simon Fraser University, Data Mining, Reasoning

Comparing Reputations

	CNN	BBC	ABC	wired .com
Int'l News	0.0237	0.0097	0.0003	0.0044
Weather	0.0121	0.0052	0.0008	0.0006
Sports	0.0070	0.0004	0	0.0028
Entertainment	0.0040	0.0015	0.0013	0.0012
Travel	0.0030	0.0008	0.0012	0.0005
Technology	0.0017	0.0006	0.0006	0.0079
Business	0.0017	0.0006	0.0004	0.0031

Limitations

- Simplistic notion of “topic”
- Use of snippets
- Some topics are not well represented on the Web
- All links are equal

Current/Future Work

- Systematic evaluation
- Combination of link- and content-based ranking
- Applications

    Reputation server

    Search engine ranking