CS206 --- Electronic Commerce

Dan Boneh
Yoav Shoham
Jeff Ullman
others . . .

1

High-Level Overview

- Discovering buyers and sellers
  - Buyers finding sellers
    - Search engines
  - Sellers finding buyers
    - Data mining
- Making a deal
  - Auctions
- Executing the deal
  - Payments, security

2

About the Course

- Minimal prerequisites:
  - CS106, CS107
  - Mathematical and algorithmic "sophistication"
- Emphasis on *technology*, not "what you need to know to start your very own dot-com."

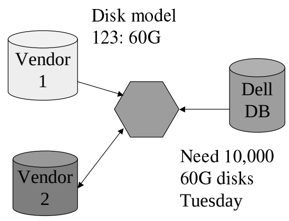
3

Issue: B2B Versus B2C

- Businesses buy/sell on-line.
  - Specialized transactions: RFP, reserve, query inventory, etc.
  - Catalogs support purchases, design.
    - Integration of supplier catalogs.
- High-value auctions.
  - e.g., bandwidth for wireless.

4

Typical Buyer: Dell



5

Technical Problems

- Transport standards, e.g. HTTP, RPC.
- Standards for interpreting messages, e.g., SOAP.
  - What is requested? What is offered? Terms?
- Lexicons or "ontologies."
  - Is 60G the same number of bytes always?

6

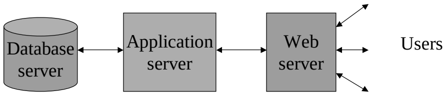
Technical Problems 2

- Integration, wrappers, middleware.
  - Different suppliers have different back-end systems. How do they talk to the hub?
- Security, authorization.
  - Who is allowed to see what?
  - Who is allowed to make decisions?
  - How do you keep out intruders?

B2C

- Many more participants.
- Payment an integral part of the process.
  - Identification, secure transfer.
- Sellers succeed by helping the buyer search.
- Massive auction site(s).

Typical Seller: Amazon



Title	Author	Price
Howl	G'brg	49.95
. . .	. . .	. . .

Queries,
Accounts,
etc.

Technical Problems

- Balancing DB/Web/App servers, distributing load.
- Wise use of (Web-page) real estate.
  - Pick a few good things to pitch to the known customer.
  - Requires complex data-mining.
    - Example: Amazon figured out I like Vivaldi and similar composers. End in "i"? Italian renaissance? Composers bought by others who buy Vivaldi CD's?

Technical Problems 2

- Exchange of sensitive information, e.g., credit-card numbers.
- Keeping stored, personal data secret.
- Managing auctions.
  - Example: 10 matching placemats for sale.
    - A: $4/each for <= 4.
    - B: $3/each for exactly 7.
    - C: $2/each for <= 6.

Finding Sellers

- A major use of search engines is finding pages that offer an item for sale.
- How do search engines find the right pages?
- We'll study:
  - Google's PageRank technique and other "tricks"
  - "Hubs and authorities."

Page Rank

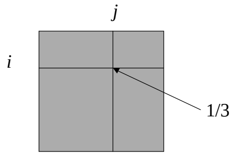
- Intuition: solve the recursive equation: "a page is important if important pages link to it."
- In high-falutin' terms: compute the principal eigenvector of the stochastic matrix of the Web.
  - A few fixups needed.

Stochastic Matrix of the Web

- Enumerate pages.
- Page *i* corresponds to row and column *i*.
- *M*[*i,j*] = 1/*n* if page *j* links to *n* pages, including page *i*; 0 if *j* does not link to *i*.
  - Seems backwards, but allows multiplication by *M* on the left to represent "follow a link."

Example

Suppose page *j* links to 3 pages, including *i*



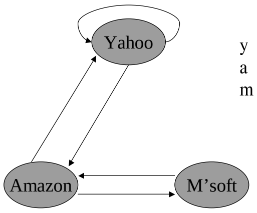
Random Walks on the Web

- Suppose *v* is a vector whose *i*-th component is the probability that we are at page *i* at a certain time.
- If we follow a link from *i* at random, the probability distribution of the page we are then at is given by the vector *Mv*.

Random Walks 2

- Starting from any vector *v*, the limit *M*(*M*(…*M*(*Mv*)…)) is the distribution of page visits during a random walk.
- Intuition: pages are important in proportion to how often a random walker would visit them.
- The math: limiting distribution = principal eigenvector of *M* = PageRank.

Example: The Web in 1839



$$\begin{array}{c|ccc|} & y & a & m \\ \hline y & 1/2 & 1/2 & 0 \\ a & 1/2 & 0 & 1 \\ m & 0 & 1/2 & 0 \\ \hline \end{array}$$

Simulating a Random Walk

- Start with the vector v = [1,1,…,1] representing the idea that each Web page is given one unit of "importance."
- Repeatedly apply the matrix M to v, allowing the importance to flow like a random walk.
- Limit exists, but about 50 iterations is sufficient to estimate final distribution.

Example

- Equations $v = Mv$:
  - $y = y/2 + a/2$
  - $a = y/2 + m$
  - $m = a/2$

$$\begin{array}{l} y \\ a \\ m \end{array} = \begin{array}{ccccccc} 1 & 1 & 5/4 & 9/8 & & 6/5 \\ 1 & 3/2 & 1 & 11/8 & \dots & 6/5 \\ 1 & 1/2 & 3/4 & 1/2 & & 3/5 \end{array}$$

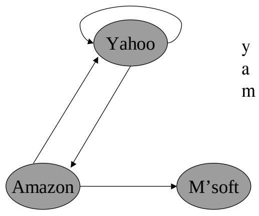
Solving The Equations

- Because there are no constant terms, these 3 equations in 3 unknowns do not have a unique solution.
- Add in the fact that $y+a+m=3$ to solve.
- In Web-sized examples, we cannot solve by Gaussian elimination; we need to use *relaxation* (= iterative solution).

Real-World Problems

- Some pages are "dead ends" (have no links out).
  - Such a page causes importance to leak out.
- Other (groups of) pages are *spider traps* (all out-links are within the group).
  - Eventually spider traps absorb all importance.

Microsoft Becomes Dead End



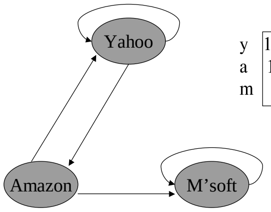
$$\begin{array}{c|ccc|} & y & a & m \\ \hline y & 1/2 & 1/2 & 0 \\ a & 1/2 & 0 & 0 \\ m & 0 & 1/2 & 0 \\ \hline \end{array}$$

Example

- Equations $v = Mv$:
  - $y = y/2 + a/2$
  - $a = y/2$
  - $m = a/2$

$$\begin{array}{l} y \\ a \\ m \end{array} = \begin{array}{ccccccc} 1 & 1 & 3/4 & 5/8 & & 0 \\ 1 & 1/2 & 1/2 & 3/8 & \dots & 0 \\ 1 & 1/2 & 1/4 & 1/4 & & 0 \end{array}$$

M’soft Becomes Spider Trap



$$\begin{array}{c|ccc|} & y & a & m \\ \hline y & 1/2 & 1/2 & 0 \\ a & 1/2 & 0 & 0 \\ m & 0 & 1/2 & 1 \\ \hline \end{array}$$

Example

- Equations $v = Mv$:
  - $y = y/2 + a/2$
  - $a = y/2$
  - $m = a/2 + m$

y		1	1	3/4	5/8		0
a	=	1	1/2	1/2	3/8	. . .	0
m		1	3/2	7/4	2		3

Google Solution to Traps, Etc.

- “Tax” each page a fixed percentage at each interation.
- Add the same constant to all pages.
- Models a random walk in which surfer has a fixed probability of abandoning search and going to a random page next.

Ex: Previous with 20% Tax

- Equations $v = 0.8(Mv) + 0.2$:
  - $y = 0.8(y/2 + a/2) + 0.2$
  - $a = 0.8(y/2) + 0.2$
  - $m = 0.8(a/2 + m) + 0.2$

y		1	1.00	0.84	0.776		7/11
a	=	1	0.60	0.60	0.536	. . .	5/11
m		1	1.40	1.56	1.688		21/11

General Case

- In this example, because there are no dead-ends, the total importance remains at 3.
- In examples with dead-ends, some importance leaks out, but total remains finite.

Solving the Equations

- Because there are constant terms, we can expect to solve small examples by Gaussian elimination.
- Web-sized examples still need to be solved by relaxation.

Search-Engine Architecture

- ◆All search engines, including Google, select pages that have the words of your query.
- ◆Give more weight to the word appearing in the title, header, etc.
- ◆Inverted indexes speed the discovery of pages with given words.

Google Anti-Spam Devices

- ◆Early search engines relied on the words on a page to tell what it is about.
  - Led to "tricks" in which pages attracted attention by placing false words in the background color on their page.
- ◆Google trusts the words in anchor text
  - Relies on others telling the truth about your page, rather than relying on you.

Use of Page Rank

- ◆Pages are ordered by many criteria, including the PageRank and the appearance of query words.
  - "Important" pages more likely to be what you want.
- ◆PageRank is also an antispam device.
  - Creating bogus links to yourself doesn't help if you are not an important page.

Hubs and Authorities

Distinguishing Two Roles for Pages

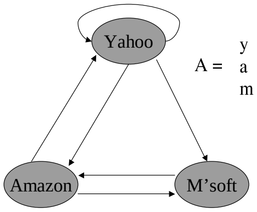
Hubs and Authorities

- ◆Mutually recursive definition:
  - A hub links to many authorities;
  - An authority is linked to by many hubs.
- ◆Authorities turn out to be places where information can be found.
  - Example: CS206 class-notes files.
- ◆Hubs tell who the authorities are.
  - Example: CS206 resources page.

Transition Matrix *A*

- ◆H&A uses a matrix $A[i,j] = 1$ if page i links to page j, 0 if not.
- ◆A^T, the transpose of A, is similar to the PageRank matrix M, but A^T has 1's where M has fractions.

Example



$$A = \begin{array}{c|ccc} & y & a & m \\ \hline y & 1 & 1 & 1 \\ a & 1 & 0 & 1 \\ m & 0 & 1 & 0 \end{array}$$

Using Matrix A for H&A

- Powers of A and A^T diverge in size, so we need scale factors.
- Let **h** and **a** be vectors measuring the "hubbiness" and authority of each page.
- Equations: $\mathbf{h} = \lambda A\mathbf{a}$; $\mathbf{a} = \mu A^T\mathbf{h}$.
  - Hubbiness = scaled sum of authorities of linked pages.
  - Authority = scaled sum of hubbiness of linked predecessors.

Consequences of Basic Equations

- From $\mathbf{h} = \lambda A\mathbf{a}$; $\mathbf{a} = \mu A^T\mathbf{h}$ we can derive:
  - $\mathbf{h} = \lambda\mu AA^T\mathbf{h}$
  - $\mathbf{a} = \lambda\mu A^TA\,\mathbf{a}$
- Compute **h** and **a** by iteration, assuming initially each page has one unit of hubbiness and one unit of authority.
  - Pick an appropriate value of $\lambda\mu$.

Example

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad A^T = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad AA^T = \begin{bmatrix} 3 & 2 & 1 \\ 2 & 2 & 0 \\ 1 & 0 & 1 \end{bmatrix} \quad A^TA = \begin{bmatrix} 2 & 1 & 2 \\ 1 & 2 & 1 \\ 2 & 1 & 2 \end{bmatrix}$$

a(yahoo)	=	1	5	24	114	. . .	1+sqrt(3)
a(amazon)	=	1	4	18	84	. . .	2
a(m'soft)	=	1	5	24	114	. . .	1+sqrt(3)
h(yahoo)	=	1	6	28	132	. . .	1.000
h(amazon)	=	1	4	20	96	. . .	0.735
h(m'soft)	=	1	2	8	36	. . .	0.268

Solving the Equations

- Solution of even small examples is tricky, because the value of $\lambda\mu$ is one of the unknowns.
  - Each equation like $y=\lambda\mu(3y+2a+m)$ lets us solve for $\lambda\mu$ in terms of y, a, m; equate each expression for $\lambda\mu$.
- As for PageRank, we need to solve big examples by relaxation.