CMPE 493
INTRODUCTION TO
INFORMATION RETRIEVAL

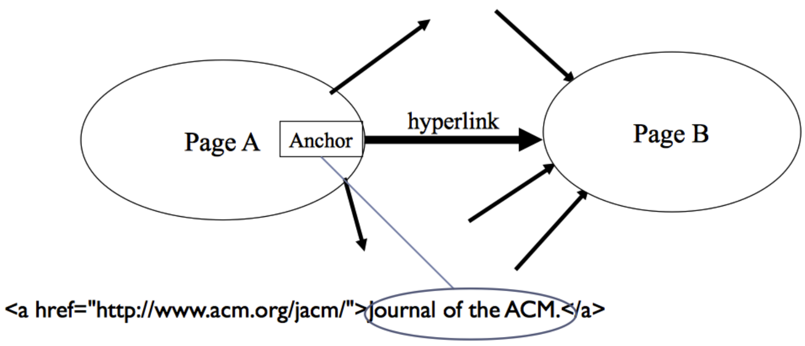
Link Analysis

Department of Computer Engineering, Boğaziçi University
December 14-15, 2020

Today's lecture

- Anchor text
- Link analysis for ranking
  - Pagerank
  - HITS

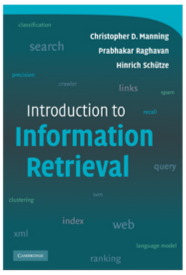
The Web as a Directed Graph



Assumption 1: A hyperlink between pages denotes author perceived relevance (quality signal)

Assumption 2: The text in the anchor of the hyperlink describes the target page (textual context)

Introduction to Information Retrieval



This is the companion website for the following book.

Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, *Introduction to Informat*

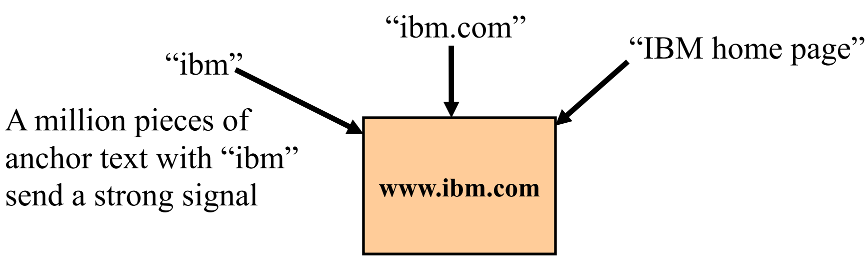
You can order this book at CUP, at your local bookstore or on the internet. The best search

The book aims to provide a modern approach to information retrieval from a computer scie University and at the University of Stuttgart

We'd be pleased to get feedback about how this book works out as a textbook, what is m comments to: informationretrieval (at) yahoogroups (dot) com

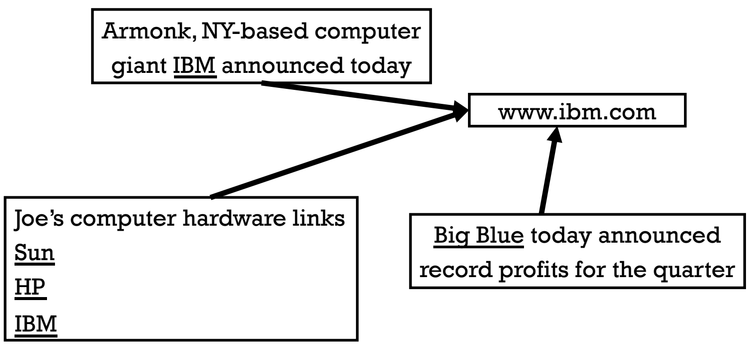
Anchor Text

- For ***ibm*** query how to distinguish between:
  - IBM's home page (mostly graphical)
  - IBM's copyright page (high term freq. for 'ibm')
  - Rival's spam page (arbitrarily high term freq.)



Indexing anchor text

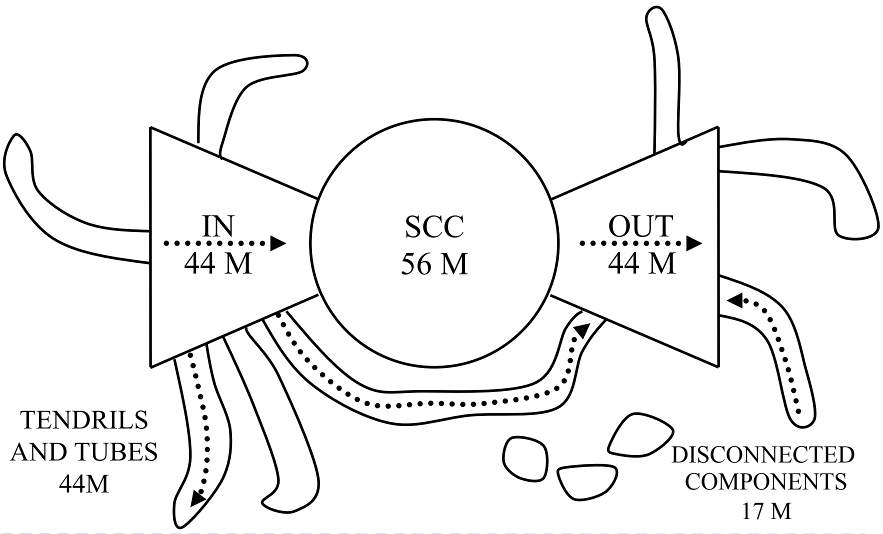
- When indexing a document *D*, include anchor text from links pointing to *D*.



Indexing anchor text

- Can sometimes have unexpected side effects (aka. Google bombs) - *e.g.,* **miserable failure**…

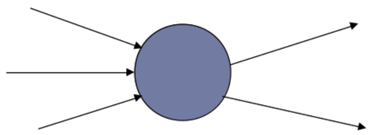
Bow-tie model of the Web



Bröder & al. WWW 2000, Dill & al. VLDB 2001

Query-independent ordering

- First generation: using link counts as simple measures of popularity.
- Two basic suggestions:
  - Undirected popularity:
    - Each page gets a score = the number of in-links plus the number of out-links (3+2=5).
  - Directed popularity:
    - Score of a page = number of its in-links (3).



Query processing

- First retrieve all pages meeting the text query
- Order these by their link popularity (either variant on the previous slide).
- More nuanced – use link counts as a measure of static goodness, combined with text match score

Spamming simple popularity

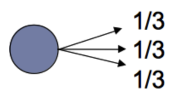
- How do you spam each of the following heuristics so your page gets a high score?
- Each page gets a static score = the number of in-links plus the number of out-links.
- Static score of a page = number of its in-links.

- In general, not all neighbors contribute equally to the importance of a node. Defined as "prestige" in social networks
  - The prestige of a person depends not only on how many friends he/she has, but also on who (how prestigious) his/her friends are.

A measure that models this: PageRank, where each neighbor contributes proportionally to its own score (importance).

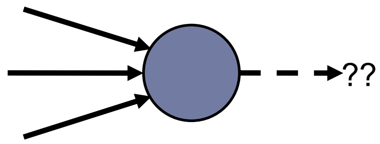
Model behind PageRank: Random walk

- Imagine a browser doing a random walk on web pages:
  - Start at a random page
  - At each step, go out of the current page along one of the links on that page, equiprobably
- "In the steady state" each page has a long-term visit rate.
- This long-term visit rate is the page's PageRank.



Not quite enough

- The web is full of dead-ends.
  - Random walk can get stuck in dead-ends.
  - Makes no sense to talk about long-term visit rates.



Teleporting

- At a dead end, jump to a random web page.
- At any non-dead end, with probability *p*, jump to a random web page.
  - With remaining probability (*1-p*), go out on a random link.
  - *p* ia a parameter (e.g., 0.1, 0.15 etc).

Result of teleporting

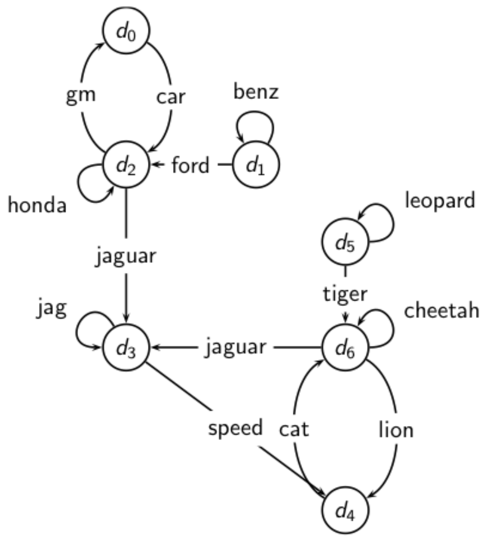
- Now cannot get stuck locally.
- There is a long-term rate at which any page is visited.
- How do we compute this visit rate?

Formalization of random walk: Markov chains

- A Markov chain consists of n states, plus an $n \times n$ transition probability matrix **P**.
  - state = page
  - At each step, we are on exactly one of the pages.
- For $1 \leq i,j \leq n$, the matrix entry P_{ij} tells us the probability that j is the next page, given we are currently on page i.
- Clearly, for all i, $\sum_{j=1}^{N} P_{ij} = 1$

$$i \xrightarrow{P_{ij}} j$$

Example web graph



Link matrix (Adjacency Matrix)

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0							
d_1							
d_2							
d_3							
d_4							
d_5							
d_6							

Link matrix

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0	0	1	0	0	0	0
d_1							
d_2							
d_3							
d_4							
d_5							
d_6							

Link matrix

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0	0	1	0	0	0	0
d_1	0	1	1	0	0	0	0
d_2							
d_3							
d_4							
d_5							
d_6							

Link matrix

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0	0	1	0	0	0	0
d_1	0	1	1	0	0	0	0
d_2	1	0	1	1	0	0	0
d_3	0	0	0	1	1	0	0
d_4	0	0	0	0	0	0	1
d_5	0	0	0	0	0	1	1
d_6	0	0	0	1	1	0	1

Transition probability matrix P

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0							
d_1							
d_2							
d_3							
d_4							
d_5							
d_6							

Transition probability matrix P

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0.00	0.00	1.00	0.00	0.00	0.00	0.00
d_1							
d_2							
d_3							
d_4							
d_5							
d_6							

Transition probability matrix P

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0.00	0.00	1.00	0.00	0.00	0.00	0.00
d_1	0.00	0.50	0.50	0.00	0.00	0.00	0.00
d_2							
d_3							
d_4							
d_5							
d_6							

Transition probability matrix P

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0.00	0.00	1.00	0.00	0.00	0.00	0.00
d_1	0.00	0.50	0.50	0.00	0.00	0.00	0.00
d_2	0.33	0.00	0.33	0.33	0.00	0.00	0.00
d_3							
d_4							
d_5							
d_6							

Transition probability matrix P

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0.00	0.00	1.00	0.00	0.00	0.00	0.00
d_1	0.00	0.50	0.50	0.00	0.00	0.00	0.00
d_2	0.33	0.00	0.33	0.33	0.00	0.00	0.00
d_3	0.00	0.00	0.00	0.50	0.50	0.00	0.00
d_4	0.00	0.00	0.00	0.00	0.00	0.00	1.00
d_5	0.00	0.00	0.00	0.00	0.00	0.50	0.50
d_6	0.00	0.00	0.00	0.33	0.33	0.00	0.33

Long-term visit rate

- Recall: PageRank = long-term visit rate.
- Long-term visit rate of page *d* is the probability that a web surfer is at page *d* at a given point in time.
- Next: what properties must hold of the web graph for the long-term visit rate to be well defined?
- The web graph must correspond to an ergodic Markov chain.

Ergodic Markov chains

- A Markov chain is ergodic if
  - There exists a positive integer T_0, such that for all pairs of states i, j, if it is started at time 0 in state i, then for $T>T_0$, the probability of being in state j at time T is greater than 0.
- A Markov chain is ergodic iff it is irreducible and aperiodic.
  - Irreducibility. Roughly: there is a path from any page to any other page.
  - Aperiodicity. Roughly: The pages cannot be partitioned such that all state transitions occur cyclically from one partition to the other.

Ergodic Markov chains

- Theorem: For any ergodic Markov chain, there is a long-term visit rate for each state.
  - *Steady-state probability distribution.*
- Over a long time-period, we visit each state in proportion to this rate.
- It doesn't matter where we start.

Ergodic Markov chains

- Teleporting makes the web graph ergodic.

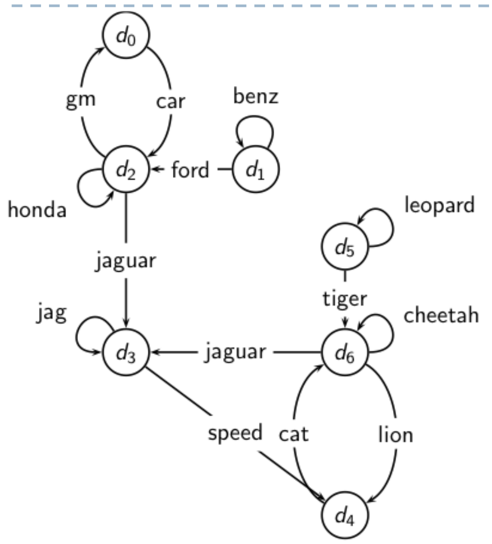
$\Longrightarrow$ Web-graph+teleporting has a steady-state probability distribution.

$\Longrightarrow$ Each page in the web-graph+teleporting has a PageRank.

Teleporting – to get us of dead ends

- At a dead end, jump to a random web page with probability 1/ N .
- Suppose the teleportation rate is given as 10% (note that it is a parameter).
- At a non-dead end:
  - with probability 10%, jump to a random web page (to each with a probability of 0.1/*N*).
  - With remaining probability (90%), go out on a random hyperlink.
    - For example, if the page has 4 outgoing links: randomly choose one with probability (1-0.10)/4=0.225
      - The overall probability is 0.1/N + (1-0.1)/4
- Note: "jumping" from dead end is independent from teleportation rate.

Example web graph



Transition (probability) matrix

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0.00	0.00	1.00	0.00	0.00	0.00	0.00
d_1	0.00	0.50	0.50	0.00	0.00	0.00	0.00
d_2	0.33	0.00	0.33	0.33	0.00	0.00	0.00
d_3	0.00	0.00	0.00	0.50	0.50	0.00	0.00
d_4	0.00	0.00	0.00	0.00	0.00	0.00	1.00
d_5	0.00	0.00	0.00	0.00	0.00	0.50	0.50
d_6	0.00	0.00	0.00	0.33	0.33	0.00	0.33

Transition matrix with teleporting

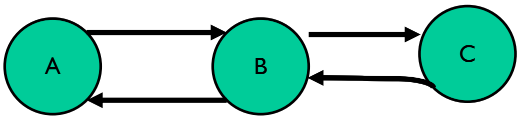
	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0.02	0.02	0.88	0.02	0.02	0.02	0.02
d_1	0.02	0.45	0.45	0.02	0.02	0.02	0.02
d_2	0.31	0.02	0.31	0.31	0.02	0.02	0.02
d_3	0.02	0.02	0.02	0.45	0.45	0.02	0.02
d_4	0.02	0.02	0.02	0.02	0.02	0.02	0.88
d_5	0.02	0.02	0.02	0.02	0.02	0.45	0.45
d_6	0.02	0.02	0.02	0.31	0.31	0.02	0.31

Teleportation rate = 0.14

First row: d0, d1, d3, d4, d5, d6: 0.14*1/7 = 0.02
d2: 0.14*1/7 + 0.86*1 = 0.88

Exercise

Represent the random walk with no teleportation as a Markov chain, for this case:



Adjacency matrix

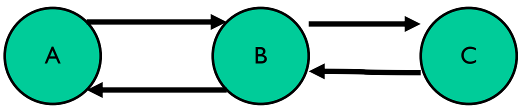
0	1	0
1	0	1
0	1	0

Transition Probability Matrix P

0	1	0
0.5	0	0.5
0	1	0

Exercise

Represent the teleporting random walk discussed before as a Markov chain, for this case :



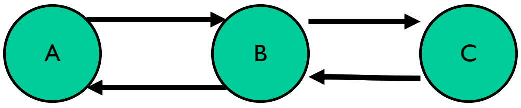
1) Compute the Adjacency Matrix A
2) If a raw in A has no 1, then replace each element by 1/N;

For all other rows proceed as follows

1) Divide each 1 in a row by the numbers of 1s in that row.
2) Multiply the resulting matrix by 1-t ; (t is the teleportation rate)
3) Add t/N to every element of the matrix to obtain P

Exercise

Represent the teleporting random walk discussed before as a Markov chain, for this case (assume t=0.5 for easier calculation) :



Adjacency matrix

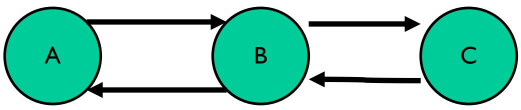
0	1	0
1	0	1
0	1	0

Transition Probability Matrix P

0	1	0
1/2	0	1/2
0	1	0

Exercise

Represent the teleporting random walk discussed before as a Markov chain, for this case (assume t=0.5 for easier calculation) :



Transition Probability Matrix P

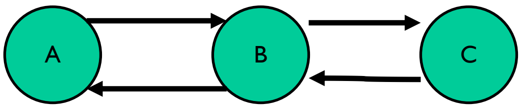
0	1	0
1/2	0	1/2
0	1	0

Transition Probability Matrix P with teleporting
Teleporting: red; random walk: blue

1/2*1/3 =1/6	1/6 + 1*1/2=2/3	1/2*1/3 =1/6
1/6 + 1/2*1/2=5/12	1/2*1/3 =1/6	1/6 + 1/2*1/2=5/12
1/2*1/3 =1/6	1/6 + 1*1/2=2/3	1/2*1/3 =1/6

Exercise

Represent the teleporting random walk discussed before as a Markov chain, for this case (assume t=0.5 for easier calculation) :



Adjacency matrix

0	1	0
1	0	1
0	1	0

Transition Probability Matrix P with teleporting

1/6	2/3	1/6
5/12	1/6	5/12
1/6	2/3	1/6

Probability vectors

- Now: How to compute PageRank
- A probability (row) vector **x** = $[x_1, \ldots x_n]$ tells us where the walk is at any point.
- E.g., [000…1…000] means we're in state *i*.

1 i n

More generally, the vector **x** = $[x_1, \ldots x_n]$ means the walk is in state *i* with probability x_i.

$$\sum_{i=1}^{n} x_i = 1 .$$

Change in probability vector

- If the probability vector is $\mathbf{x} = [x_1, \ldots x_n]$ at this step, what is it at the next step?
- Recall that row *i* of the transition probability Matrix **P** tells us where we go next from state *i*.
- So from **x**, our next state is distributed as **xP**.

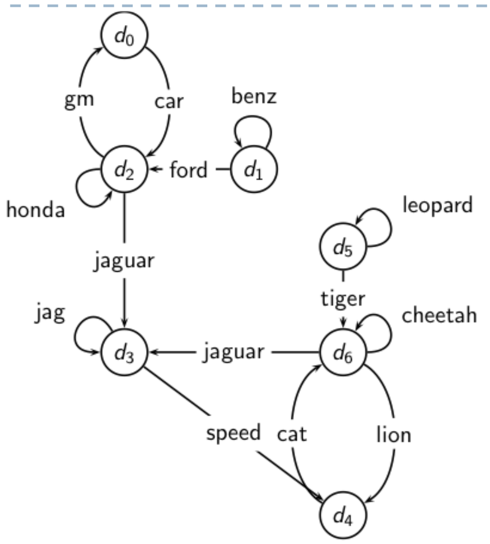
How do we compute this vector?

- Let $\mathbf{a} = [a_1, \ldots a_n]$ denote the row vector of steady-state probabilities.
- If our current position is described by **a**, then the next step is distributed as **aP**.
- But **a** is the steady state, so **a**=**aP**.
- Solving this matrix equation gives us **a**.
  - So **a** is the (left) eigenvector for **P**.
  - (Corresponds to the "principal" eigenvector of **P** with the largest eigenvalue.)
  - Transition probability matrices always have largest eigenvalue 1.

One way of computing **a**: Power Iteration Method

- Recall, regardless of where we start, we eventually reach the steady state **a**.
- Start with any distribution (say **x**=[*10…0*]).
- After one step, we're at **xP**;
- after two steps at $\mathbf{xP}^2$, then $\mathbf{xP}^3$ and so on.
- "Eventually" means for "large" k, $\mathbf{xP}^k = \mathbf{a}$.
- Algorithm: multiply **x** by increasing powers of **P** until the product looks stable.

Example web graph



Transition (probability) matrix

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0.00	0.00	1.00	0.00	0.00	0.00	0.00
d_1	0.00	0.50	0.50	0.00	0.00	0.00	0.00
d_2	0.33	0.00	0.33	0.33	0.00	0.00	0.00
d_3	0.00	0.00	0.00	0.50	0.50	0.00	0.00
d_4	0.00	0.00	0.00	0.00	0.00	0.00	1.00
d_5	0.00	0.00	0.00	0.00	0.00	0.50	0.50
d_6	0.00	0.00	0.00	0.33	0.33	0.00	0.33

Transition matrix with teleporting

	d_0	d_1	d_2	d_3	d_4	d_5	d_6
d_0	0.02	0.02	0.88	0.02	0.02	0.02	0.02
d_1	0.02	0.45	0.45	0.02	0.02	0.02	0.02
d_2	0.31	0.02	0.31	0.31	0.02	0.02	0.02
d_3	0.02	0.02	0.02	0.45	0.45	0.02	0.02
d_4	0.02	0.02	0.02	0.02	0.02	0.02	0.88
d_5	0.02	0.02	0.02	0.02	0.02	0.45	0.45
d_6	0.02	0.02	0.02	0.31	0.31	0.02	0.31

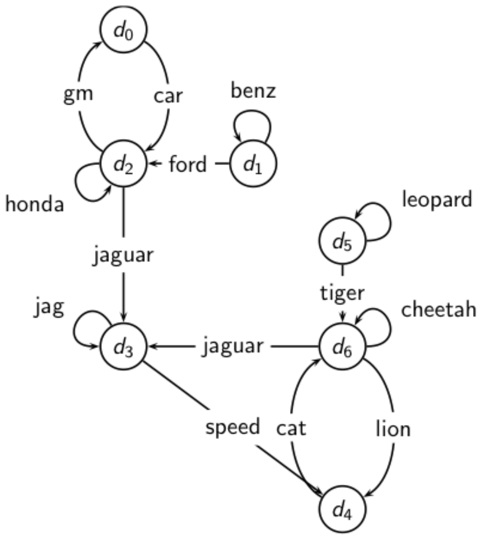
Teleportation rate = 0.14

First row: d0, d3, d4, d5, d6: 0.14*1/7 = 0.02
d2: 0.14*1/7 + 0.86*1 = 0.88

Power method vectors $\vec{x}P^k$

	$\vec{x}$	$\vec{x}P^1$	$\vec{x}P^2$	$\vec{x}P^3$	$\vec{x}P^4$	$\vec{x}P^5$	$\vec{x}P^6$	$\vec{x}P^7$	$\vec{x}P^8$	$\vec{x}P^9$	$\vec{x}P^{10}$
d_0	0.14	0.06	0.09	0.07	0.07	0.06	0.06	0.06	0.06	0.05	0.05
d_1	0.14	0.08	0.06	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
d_2	0.14	0.25	0.18	0.17	0.15	0.14	0.13	0.12	0.12	0.12	0.12
d_3	0.14	0.16	0.23	0.24	0.24	0.24	0.24	0.25	0.25	0.25	0.25
d_4	0.14	0.12	0.16	0.19	0.19	0.20	0.21	0.21	0.21	0.21	0.21
d_5	0.14	0.08	0.06	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
d_6	0.14	0.25	0.23	0.25	0.27	0.28	0.29	0.29	0.30	0.30	0.30

Example web graph



PageRank	
d_0	0.05
d_1	0.04
d_2	0.11
d_3	0.25
d_4	0.21
d_5	0.04
d_6	0.30

PageRank summary

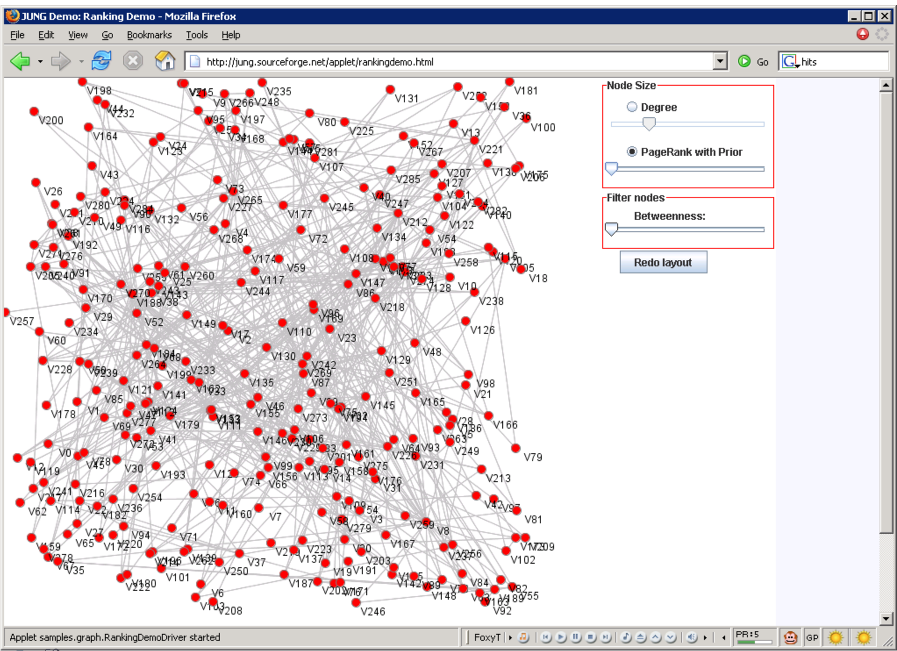
- Preprocessing
  - Given graph of links, build matrix *P*
  - Apply teleportation
  - From modified matrix, compute $\vec{\mathbf{a}}$
  - $\mathbf{a}_i$ is the PageRank of page i.
- Query processing
  - Retrieve pages satisfying the query
  - Rank them by their PageRank (or a combination of PageRank, match score etc.)
  - Return ranked list to the user

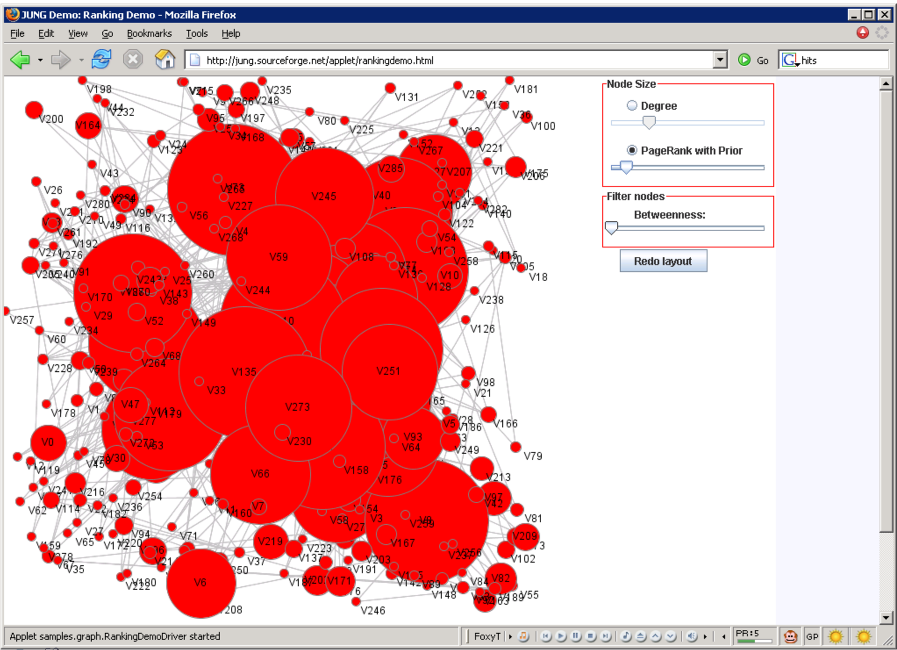
PageRank issues

- Real surfers are not random surfers.
  - Examples of nonrandom surfing: back button, bookmarks, directories – and search!
  - → Markov model is not a good model of surfing.
  - But it's good enough as a model for our purposes.
- Simple PageRank ranking (as described on previous slide) produces bad results for many pages.
  - Consider the query [video service].
  - The Yahoo home page (i) has a very high PageRank and (ii) contains both *video* and *service*.
  - If we rank all Boolean hits according to PageRank, then the Yahoo home page would be top-ranked.
  - Clearly not desirable.

PageRank issues

- In practice: rank according to weighted combination of raw text match, anchor text match, keyword proximity, PageRank & other factors.





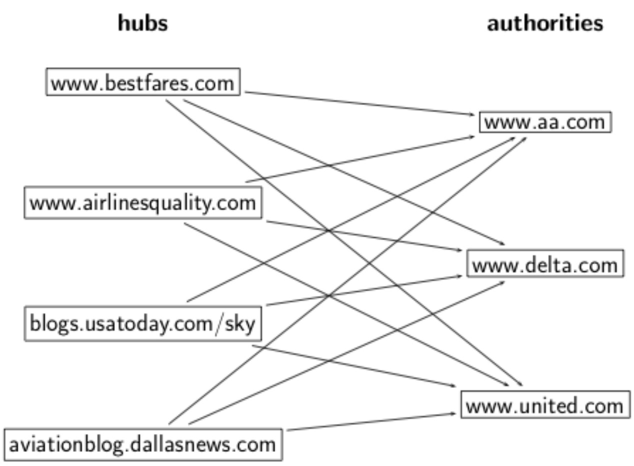
Hyperlink-Induced Topic Search (HITS)

- Developed by Jon Kleinberg and colleagues at IBM Almaden as part of the CLEVER engine.
- HITS is query-specific.
- In response to a query, instead of an ordered list of pages each meeting the query, find <u>two</u> sets of inter-related pages:
  - *Hub pages* are good lists of links on a subject.
    - e.g.,"Bob's list of cancer-related links."
  - *Authority pages* occur recurrently on good hubs for the subject.
- HITS is now used by Ask.com and Teoma.com

Hubs and Authorities

- Thus, a good hub page for a topic *points* to many authoritative pages for that topic.
- A good authority page for a topic is *pointed* to by many good hubs for that topic.
- Circular definition - will turn this into an iterative computation.

Example for hubs and authorities



High-level scheme

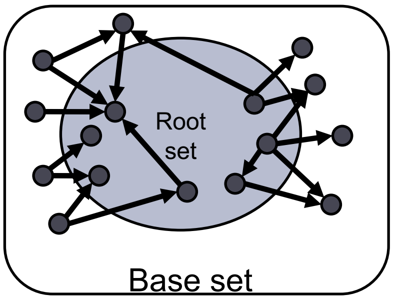
- Extract from the web a base set of pages that *could* be good hubs or authorities.
- From these, identify a small set of top hub and authority pages;

  →iterative algorithm.

Base set

- Given text query (say ***airline***), use a text index to get all pages containing ***airline.***
  - Call this the root set of pages.
- Add in any page that either
  - points to a page in the root set, or
  - is pointed to by a page in the root set.
- Call this the base set.

Visualization



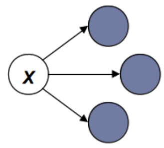
Distilling hubs and authorities

- Compute, for each page *x* in the base set, a <u>hub score</u> *h(x)* and an <u>authority score</u> *a(x)*.
- Initialize: for all *x*, $h(x) \leftarrow 1$; $a(x) \leftarrow 1$;
- Iteratively update all *h(x)*, *a(x)*;
- After iterations
  - output pages with highest *h()* scores as top hubs
  - highest *a()* scores as top authorities.

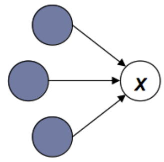
Iterative update

- Repeat the following updates, for all *x*:

$$h(x) \leftarrow \sum_{x \to y} a(y)$$



$$a(x) \leftarrow \sum_{y \to x} h(y)$$



Japan Elementary Schools

Hubs

- schools
- LINK Page-13
- “ú–{,ÌŠw□Z
- □a‰,□¬Šw□Zƒz□[ƒ€ƒy□[ƒW
- 100 Schools Home Pages (English)
- K-12 from Japan 10/...rnet and Education)
- http://www...iglobe.ne.jp/~IKESAN
- ,l,f,j□¬Šw□Z,U”N,P‘g•¨Œê
- □ÒŠ—’¬—§ □ÒŠ—“Œ□¬Šw□Z
- Koulutus ja oppilaitokset
- TOYODA HOMEPAGE
- Education
- Cay's Homepage(Japanese)
- –y“i□¬Šw□Z,Ìƒz□[ƒ€ƒy□[ƒW
- UNIVERSITY
- ‰J—³□¬Šw□Z DRAGON97-TOP
- ŽÂ‰ª□¬Šw□Z,T”N,P‘gƒz□[ƒ€ƒy□[ƒW
- ¶µ° é¼ÂÁ© ¥á¥Ë¥å¡¼ ¥á¥Ë¥å¡¼

Authorities

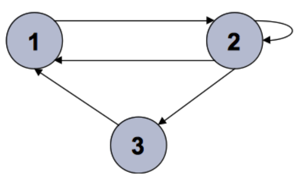
- The American School in Japan
- The Link Page
- ‰ª□èŽs—§ˆä“c□¬Šw□Zƒz□[ƒ€ƒy□[ƒW
- Kids' Space
- ˆÀ□éŽs—§ˆÀ□é□¼•”□¬Šw□Z
- ‹{□é‹³ˆç‘åŠw•□‘®□¬Šw□Z
- KEIMEI GAKUEN Home Page (Japanese)
- Shiranuma Home Page
- fuzoku-es.fukui-u.ac.jp
- welcome to Miasa E&J school
- □_“Þ□ìŒ§□E‰¡•lŽs—§’†□ì□¼□¬Šw□Z,̃y
- http://www...p/~m_maru/index.html
- fukui haruyama-es HomePage
- Torisu primary school
- goo
- Yakumo Elementary,Hokkaido,Japan
- FUZOKU Home Page
- Kamishibun Elementary School...

Things to note

- Pulled together good pages regardless of language of page content.
- Use *only* link analysis after base set assembled
  - iterative scoring is query-independent.

Eigenvector interpretation

- $n \times n$ adjacency matrix **A**:
- each of the n pages in the base set has a row and column in the matrix.
- Entry $A_{ij} = 1$ if page i links to page j, else = 0.



$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 0 & 1 & 0 \\ 2 & 1 & 1 & 1 \\ 3 & 1 & 0 & 0 \end{array}$$

Hub/authority vectors

- View the hub scores *h()* and the authority scores *a()* as vectors with *n* components.
- Recall the iterative updates

$$h(x) \leftarrow \sum_{x \to y} a(y)$$

$$a(x) \leftarrow \sum_{y \to x} h(y)$$

Rewrite in matrix form

- **h=Aa**.
- **a=A^{t}h**.

Recall A^t is the transpose of A.

Substituting, **h=AA^{t}h** and **a=A^{t}Aa**.

Thus, **h** is an eigenvector of **AA^t** and **a** is an eigenvector of **A^{t}A**.

Can use the *power iteration* method (like we did for Pagerank) to compute the eigenvectors.

Resources

- *Introduction to Information Retrieval*, chapter 21.
- Some slides were adapted from
  - Prof. Dragomir Radev's lectures at the University of Michigan:
    - http://clair.si.umich.edu/~radev/teaching.html
  - the book's companion website:
    - http://nlp.stanford.edu/IR-book/information-retrieval-book.html