BLG 540E
TEXT RETRIEVAL SYSTEMS

Link Analysis

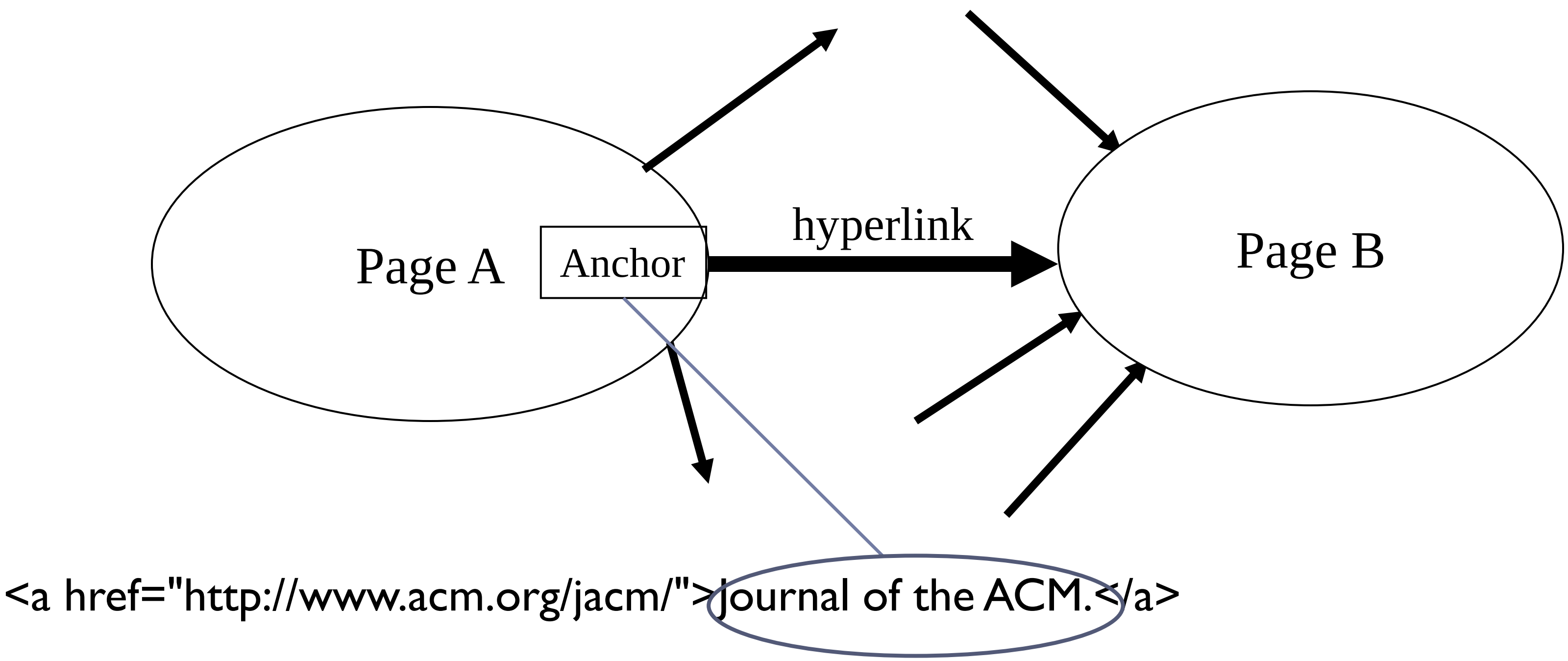
Arzucan Özgür

Faculty of Computer and Informatics, İstanbul Techical University
April 29, 2011

Today's lecture

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

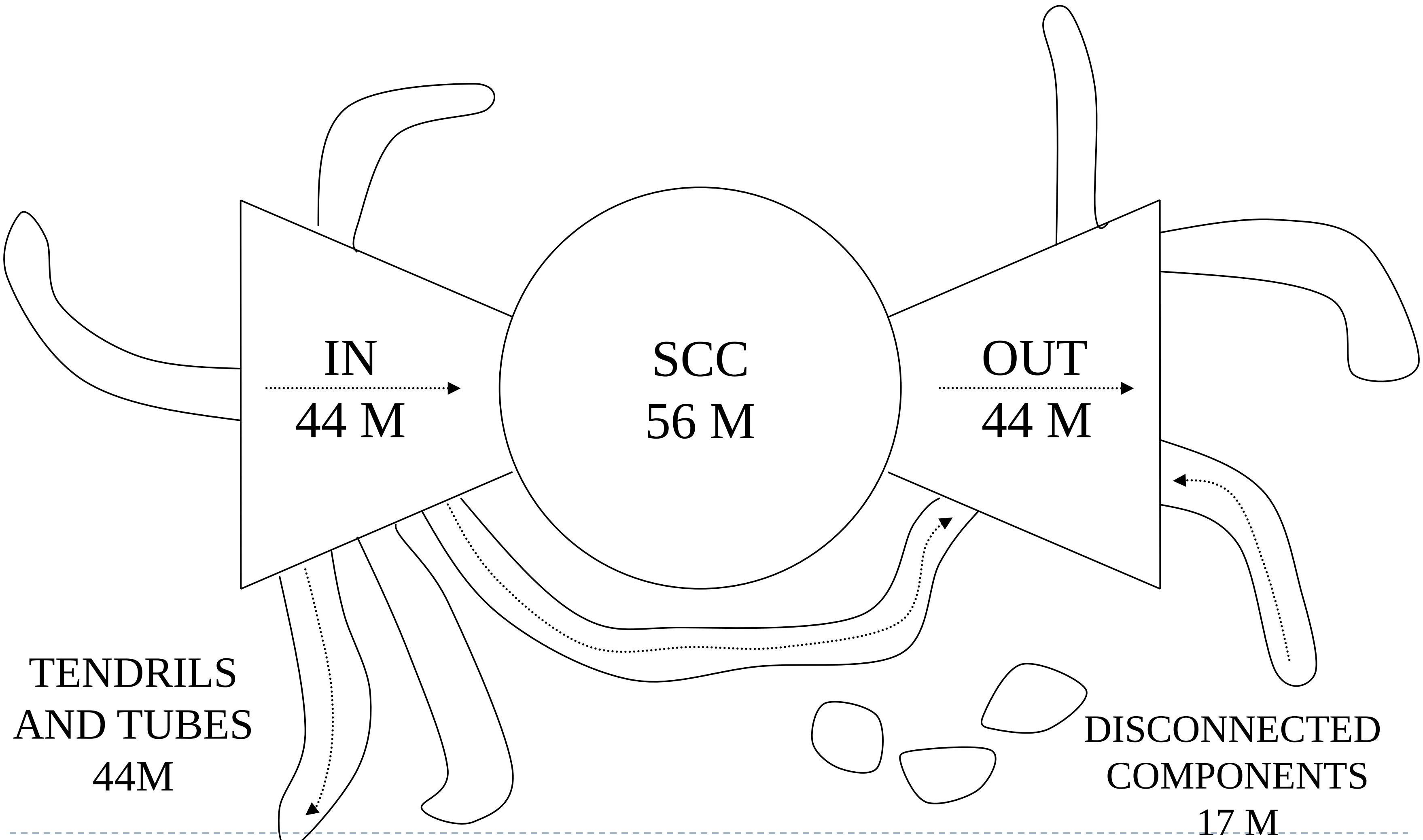
The Web as a Directed Graph



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)

Bow-tie model of the Web



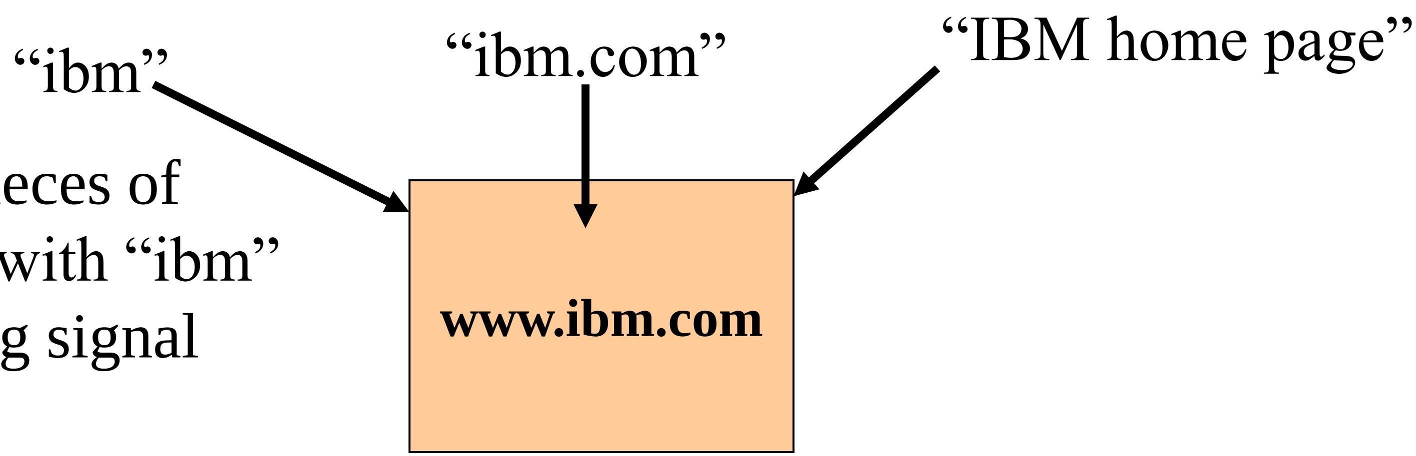
Bröder & al. WWW 2000, Dill & al. VLDB 2001

Anchor Text

WWW Worm - McBryan [Mcbr94]

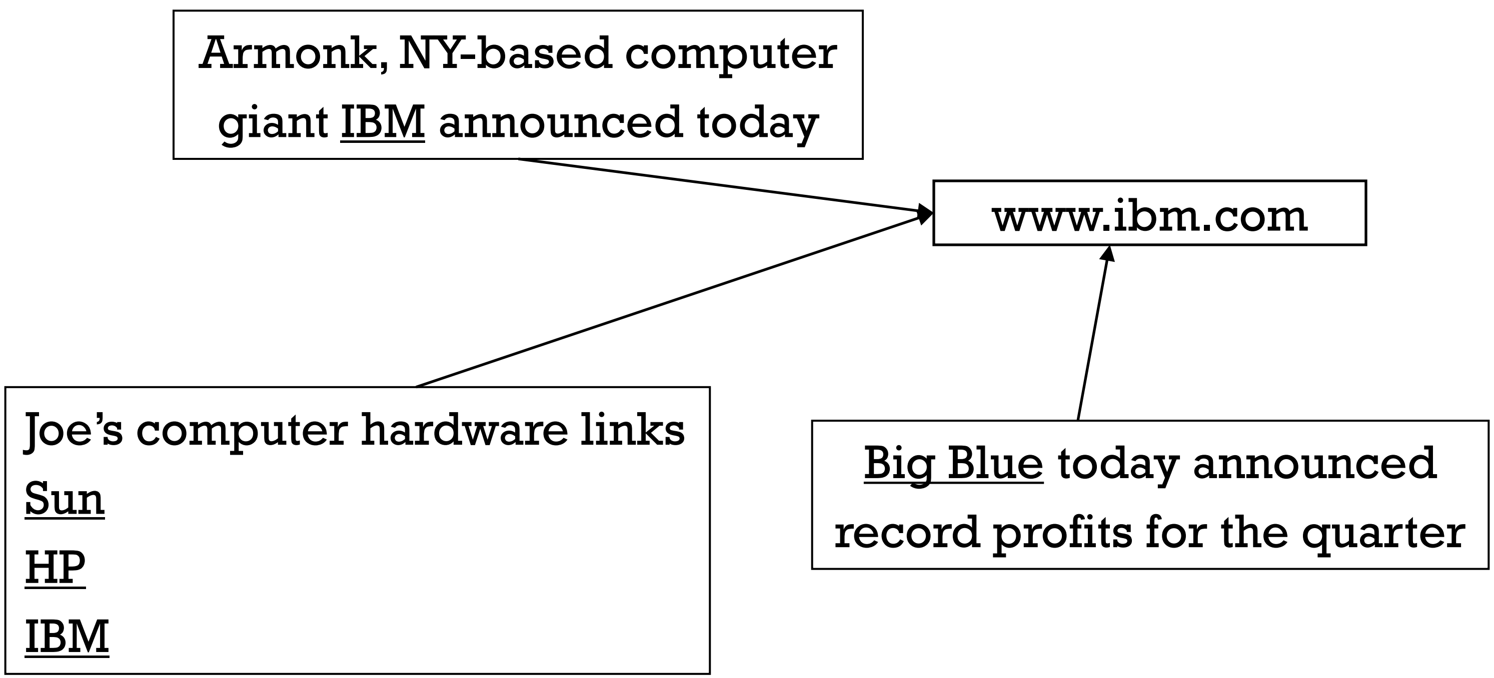
- For ***ibm*** how to distinguish between:
  - IBM's home page (mostly graphical)
  - IBM's copyright page (high term freq. for 'ibm')
  - Rival's spam page (arbitrarily high term freq.)

A million pieces of anchor text with "ibm" send a strong signal



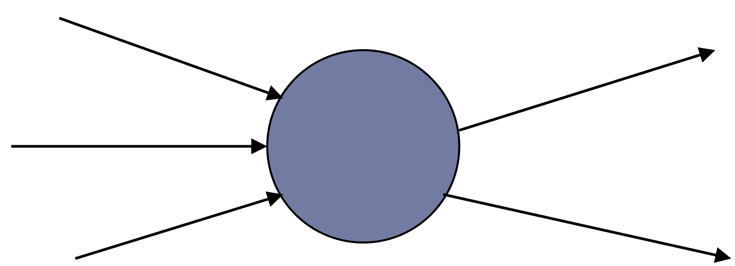
Indexing anchor text

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



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

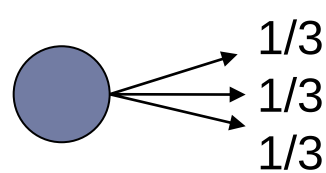
- First retrieve all pages meeting the text query (say ***venture capital***).
- Order these by their link popularity (either variant on the previous slide).
- More nuanced – use link counts as a measure of static goodness (Lecture 5), combined with text match score

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.

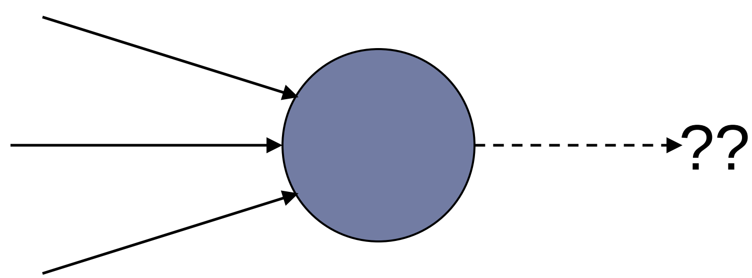
Pagerank scoring

- 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 - use this as the page's score.



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

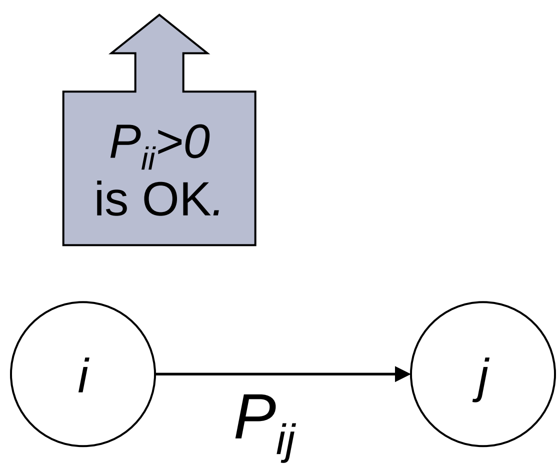
- At a dead end, jump to a random web page.
- At any non-dead end, with probability 10%, jump to a random web page.
  - With remaining probability (90%), go out on a random link.
  - 10% - a parameter.

Result of teleporting

- Now cannot get stuck locally.
- There is a long-term rate at which any page is visited (not obvious, will show this).
- How do we compute this visit rate?

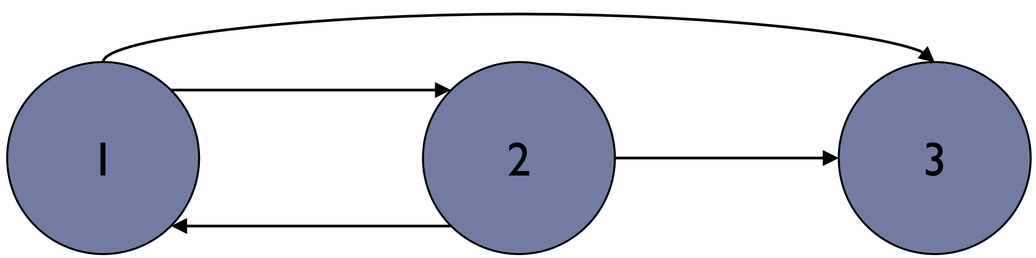
Markov chains

- A Markov chain consists of n <u>states</u>, plus an $n \times n$ <u>transition probability matrix</u> **P**.
- At each step, we are in exactly one of the states.
- For $1 \leq i,j \leq n$, the matrix entry P_{ij} tells us the probability of j being the next state, given we are currently in state i.



Markov chains

- Clearly, for all i, $\sum_{j=1}^{n} P_{ij} = 1.$
- Markov chains are abstractions of random walks.
- *Exercise*: represent the teleporting random walk from 3 slides ago as a Markov chain, for this case:



Ergodic Markov chains

- A Markov chain is ergodic if
  - you have a path from any state to any other
  - For any start state, after a finite transient time T_0, the probability of being in any state at a fixed time $T>T_0$ is nonzero.

Ergodic Markov chains

- For any ergodic Markov chain, there is a unique 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.

Probability vectors

- 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.

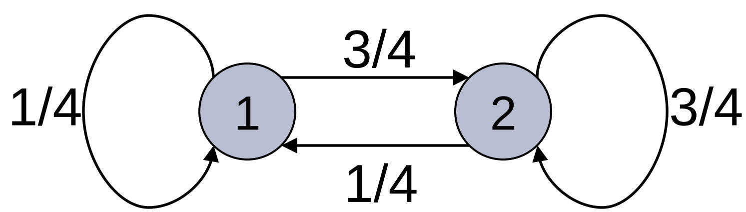
$$\sum_{i=1}^{n} x_i = 1.$$

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**.

Steady state example

- The steady state looks like a vector of probabilities **a** = $(a_1, \ldots a_n)$:
  - a_i is the probability that we are in state i.



For this example, $a_1=1/4$ and $a_2=3/4$.

How do we compute this vector?

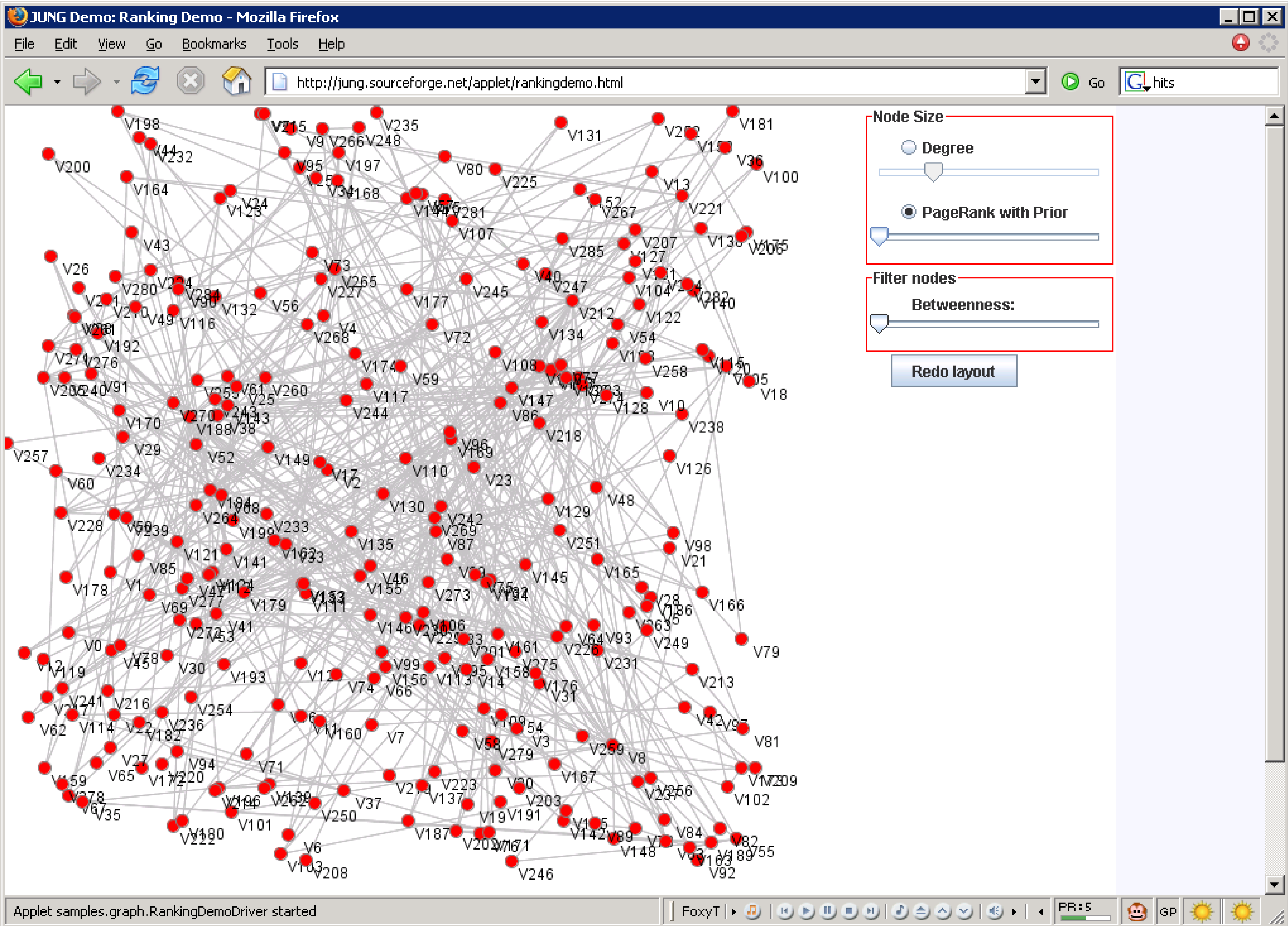
- Let **a** = $(a_1, \ldots\ a_n)$ denote the row vector of steady-state probabilities.
- If our current position is described by **a**, then the next step is distributed as **aP**.
- But **a** is the steady state, so **a**=**aP**.
- Solving this matrix equation gives us **a**.
  - So **a** is the (left)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

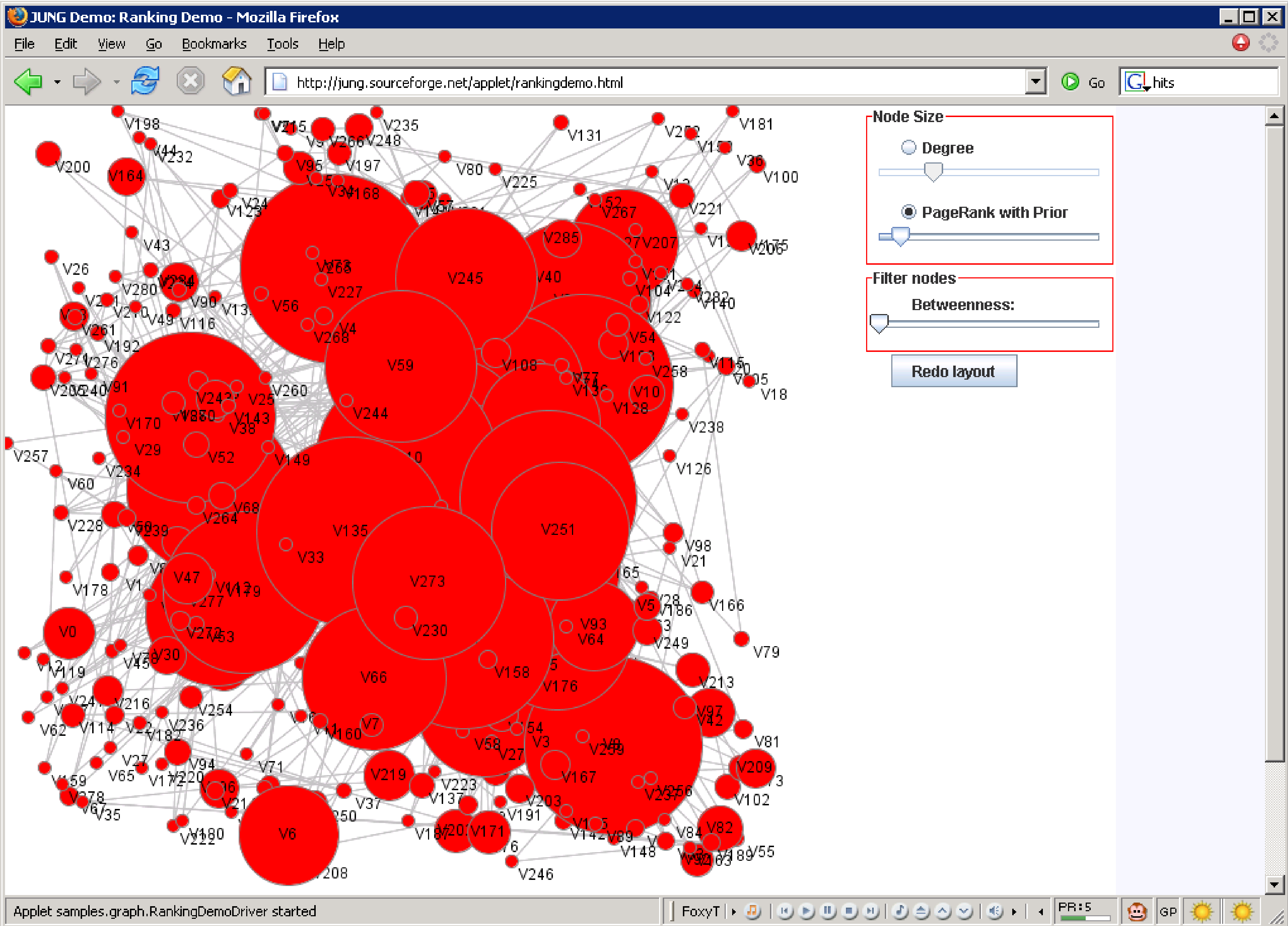
One way of computing **a**

- 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.

PageRank

- Developed at Stanford and allegedly still being used at Google.
- Not query-specific, although query-specific varieties exist.
- In general, each page is indexed along with the anchor texts pointing to it.
- Among the pages that match the user’s query, Google shows the ones with the largest PageRank.
- Google also uses vector-space matching, keyword proximity, anchor text, etc.


JUNG Demo: Ranking Demo - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://jung.sourceforge.net/applet/rankingdemo.html
Go
hits
Node Size
Degree
PageRank with Prior
Filter nodes
Betweenness:
Redo layout
Applet samples.graph.RankingDemoDriver started
FoxyT
PR: 5
GP


JUNG Demo: Ranking Demo - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://jung.sourceforge.net/applet/rankingdemo.html
Go
hits
Node Size
Degree
PageRank with Prior
Filter nodes
Betweenness:
Redo layout
Applet samples.graph.RankingDemoDriver started
FoxyT
PR:5
GP

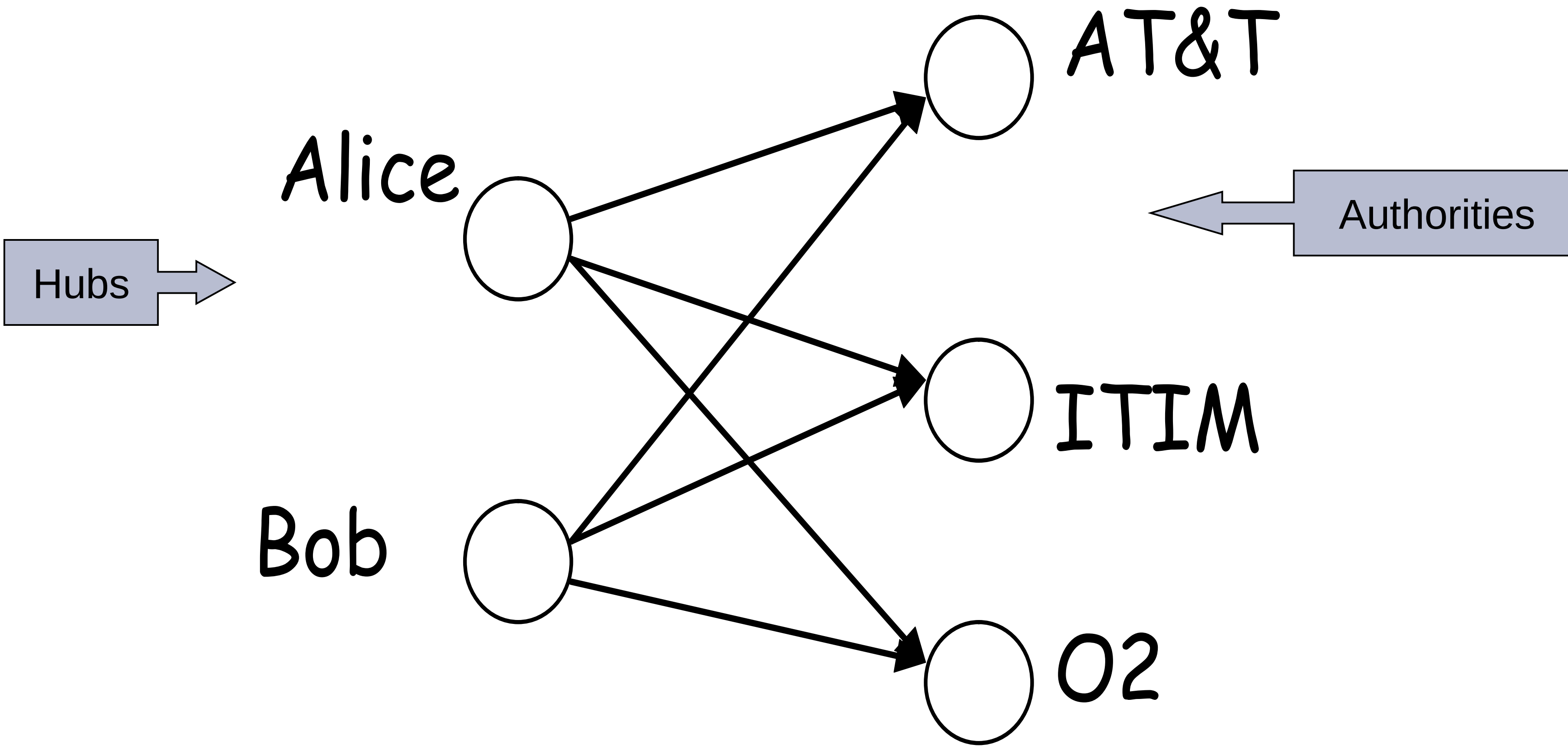
Hyperlink-Induced Topic Search (HITS)

- 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 two 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.

The hope



Mobile telecom companies

High-level scheme

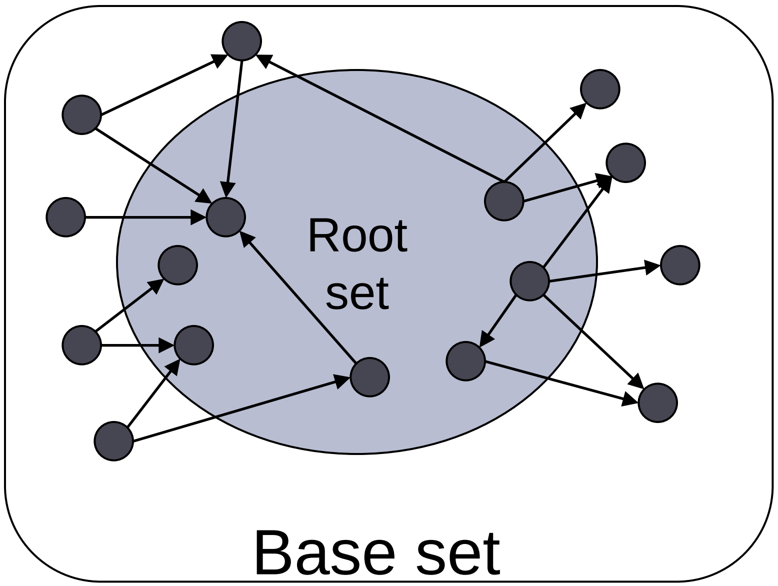
- Extract from the web a base set of pages that *could* be good hubs or authorities.
- From these, identify a small set of top hub and authority pages;

  →iterative algorithm.

Base set

- Given text query (say ***browser***), use a text index to get all pages containing ***browser.***
  - 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



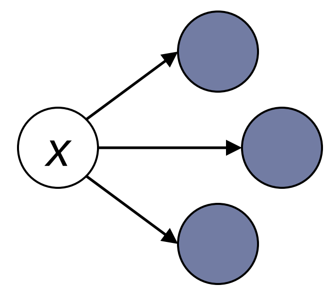
Distilling hubs and authorities

- Compute, for each page x in the base set, a hub score $h(x)$ and an authority score $a(x)$.
- 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.

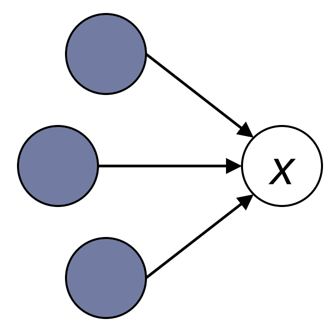
Iterative update

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

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



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



Japan Elementary Schools

Hubs

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

Authorities

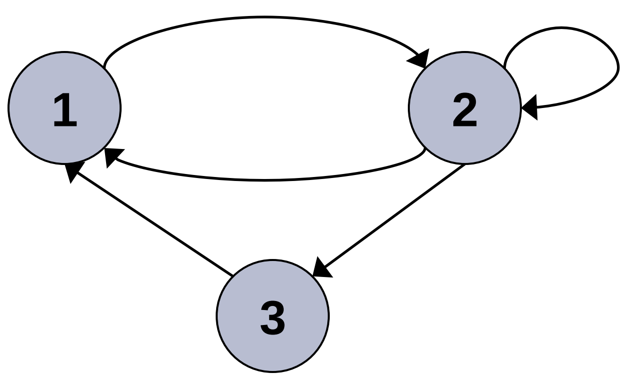
- The American School in Japan
- The Link Page
- ‰ªès—§ˆä“c¬ŠwZƒz[ƒƒy[ƒW
- Kids' Space
- ˆÀés—§ˆÀé¼•”¬ŠwZ
- ‹{é‹³ˆç‘åŠw•‘®¬ŠwZ
- KEIMEI GAKUEN Home Page (Japanese)
- Shiranuma Home Page
- fuzoku-es.fukui-u.ac.jp
- welcome to Miasa E&J school
- “ÞìŒ§E‰¡•ls—§’†ì¼¬ŠwZ‚Ìƒy
- http://www...p/~m_maru/index.html
- 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.



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

Hub/authority vectors

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

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

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

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 to compute the eigenvectors.

Resources

- *Introduction to Information Retrieval*, chapters 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