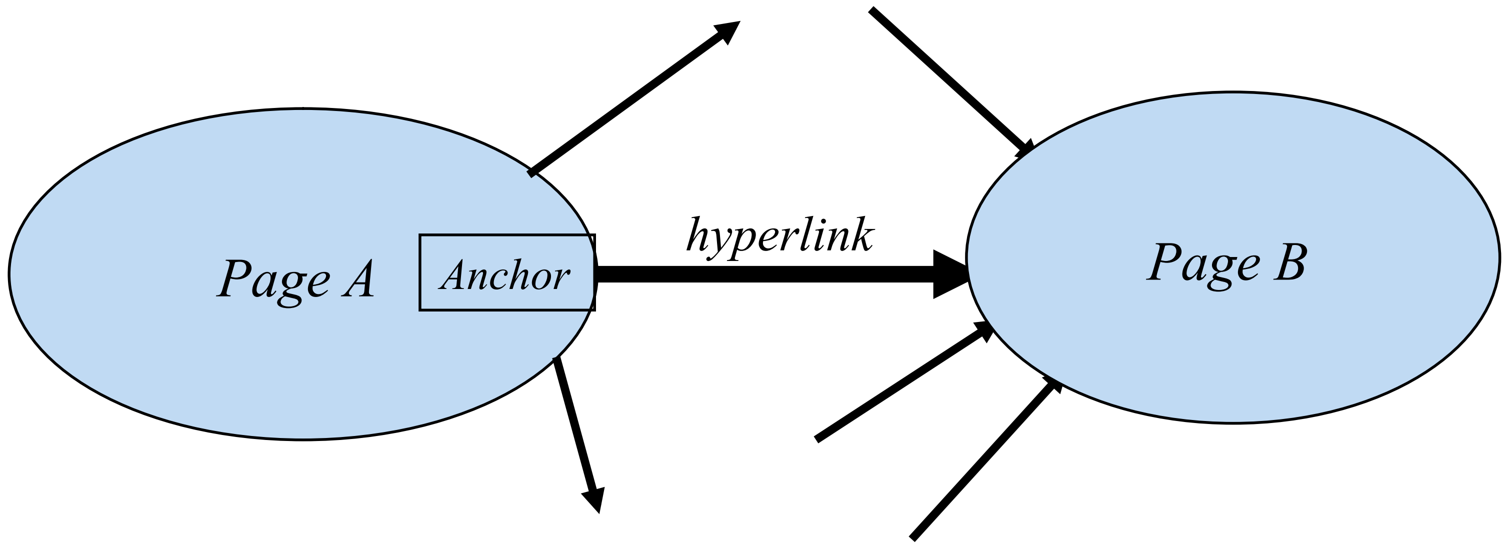
Multimedia Content Management: Link Analysis

Ralf Moeller
Hamburg Univ. of Technology

Today's lecture

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

The Web as a Directed Graph



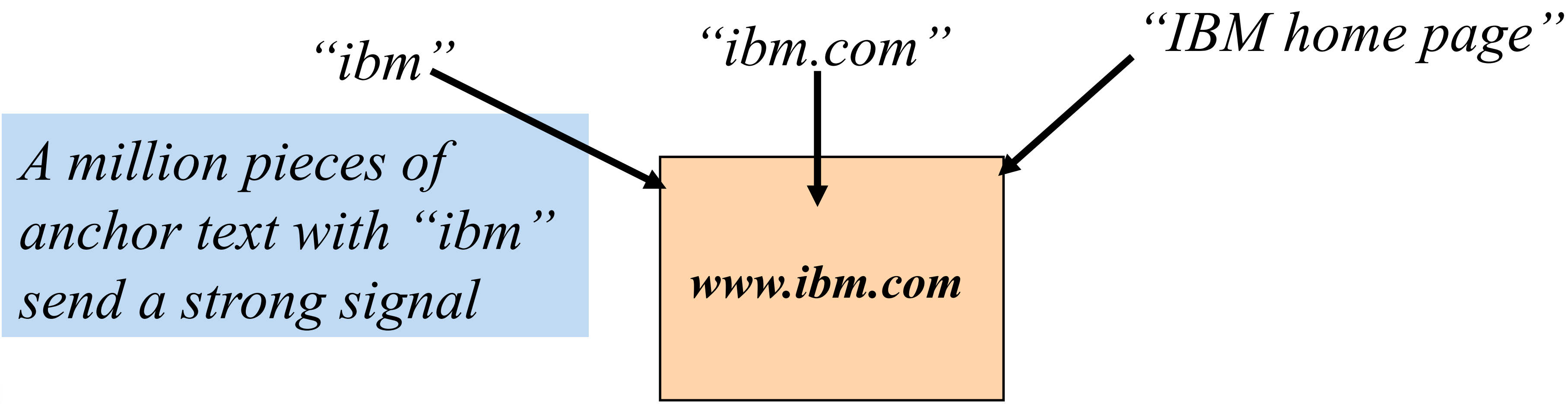
Assumption 1: *A hyperlink between pages denotes author perceived relevance (quality signal)*

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

Anchor Text

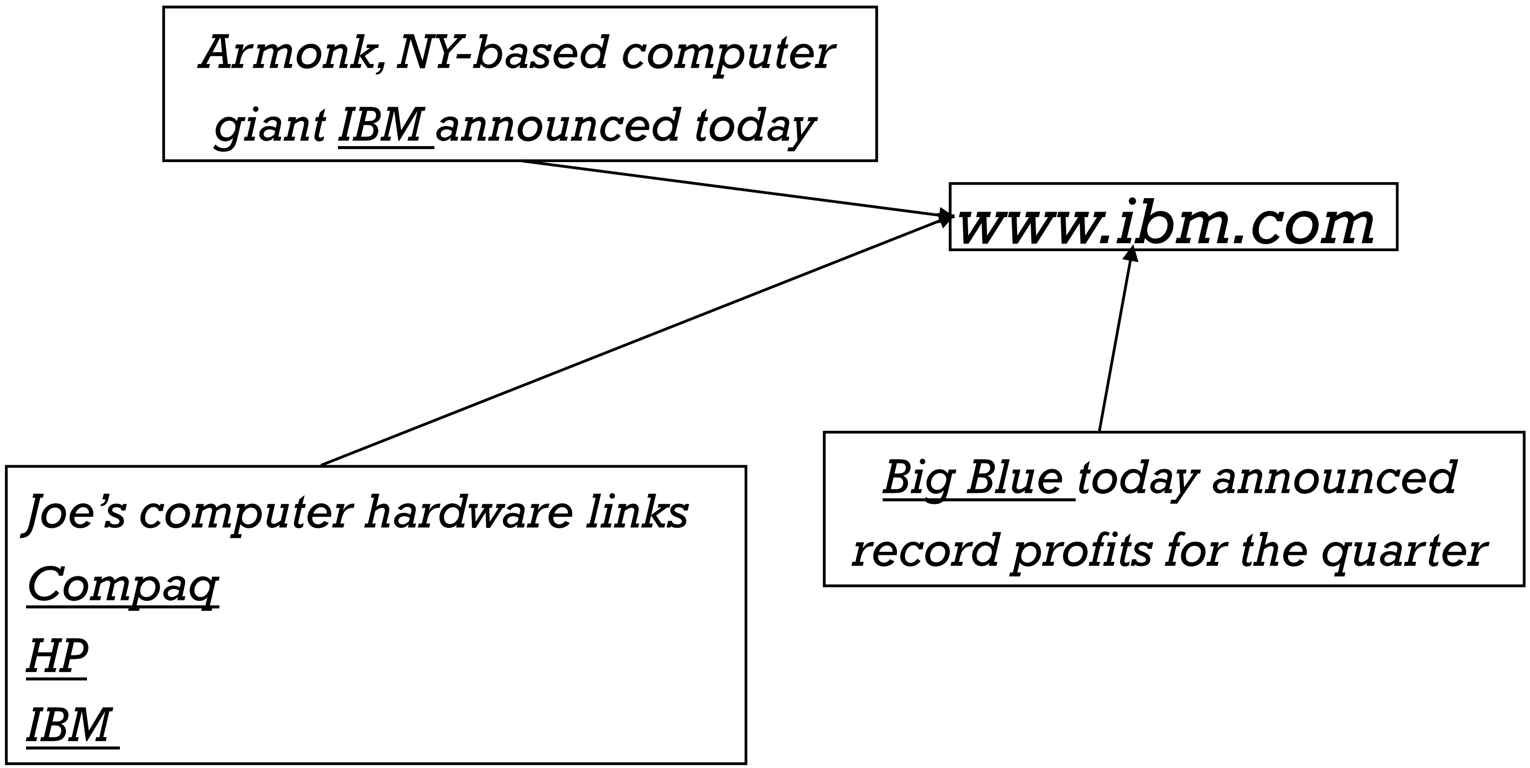
WWW Worm – McBryan [Mcbr94]

- 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.)



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 – *e.g.*, ***evil empire***.
- Can index anchor text with less weight.

Anchor Text

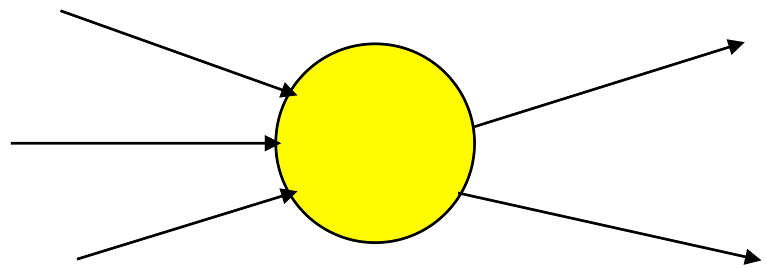
- Other applications
  - Weighting/filtering links in the graph
    - HITS [Chak98], Hilltop [Bhar01]
  - Generating page descriptions from anchor text [Amit98, Amit00]

Citation Analysis

- Citation frequency
- Co-citation coupling frequency
  - Cocitations with a given author measures "impact"
  - Cocitation analysis [Mcca90]
- Bibliographic coupling frequency
  - Articles that co-cite the same articles are related
- Citation indexing
  - Who is author cited by? (Garfield [Garf72])
- Pagerank (preview: Pinsker and Narin '60s)

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).



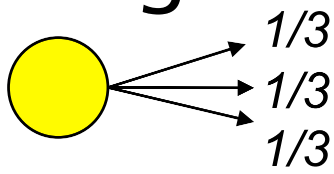
Query processing

- First retrieve all pages meeting the text query (say ***venture capital***).
- Order these by their link popularity (either variant on the previous page).

Spamming simple popularity

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

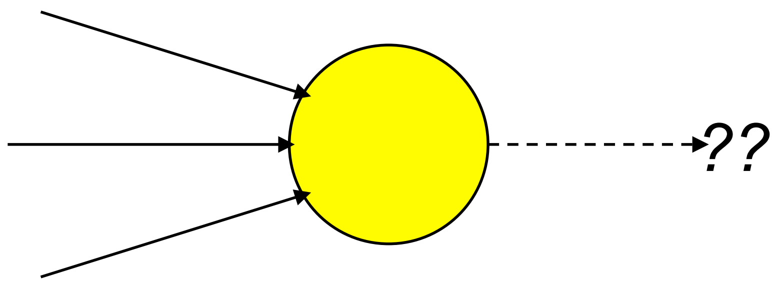
Pagerank scoring

- 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.

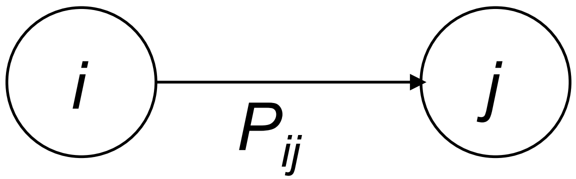
Result of teleporting

- 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

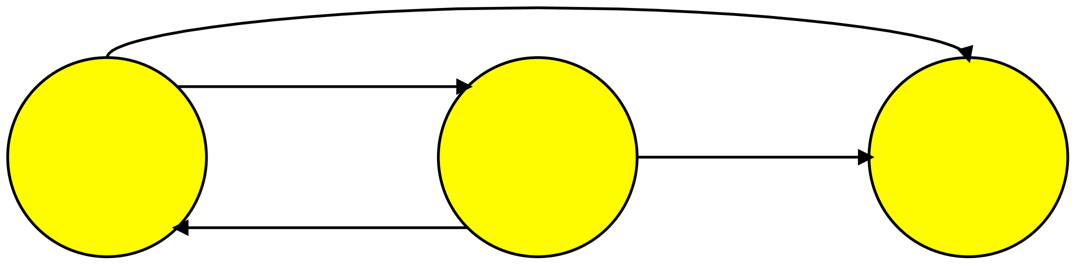
- A Markov chain consists of n states, plus an $n \times n$ transition probability matrix **P**.
- At each step, we are in exactly one of the states.
- For $1 \leq i,j \leq n$, the matrix entry P_{ij} tells us the probability of j being the next state, given we are currently in state i.

$P_{ii}>0$ is OK.



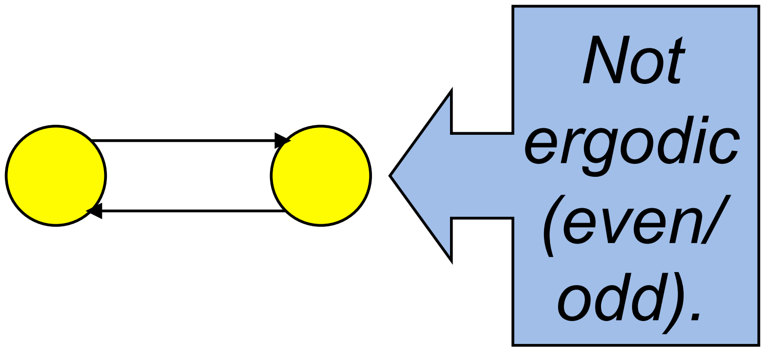
Markov chains

- 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 <u>ergodic</u> if
  - you have a path from any state to any other (reducibility)
  - returns to states occur at irregular times (aperiodicity)
  - For any start state, after a finite transient time T_0, <u>the probability of being in any state at a fixed time $T>T_0$ is nonzero.</u> (positive recurrence)



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*.

  $\quad\quad 1 \quad\quad i \quad\quad n$

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

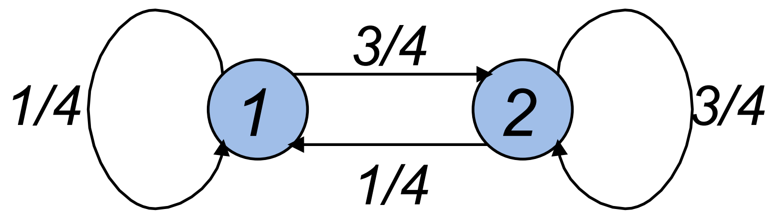
$$\sum_{i=1}^{n} x_i = 1.$$

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 prob. 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 we 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.

Eigenvalues & Eigenvectors

- **Eigenvectors** (for a square $m \times m$ matrix $\mathbf{S}$)

$$\mathbf{S}\mathbf{v} = \lambda \mathbf{v}$$

(right) eigenvector $\mathbf{v} \in \mathbb{R}^m \neq \mathbf{0}$

eigenvalue $\lambda \in \mathbb{R}$

Example

$$\begin{pmatrix} 6 & -2 \\ 4 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

- How many eigenvalues are there at most?

$$\mathbf{S}\mathbf{v} = \lambda \mathbf{v} \iff (\mathbf{S} - \lambda \mathbf{I})\,\mathbf{v} = \mathbf{0}$$

only has a non-zero solution if $|\mathbf{S} - \lambda \mathbf{I}| = 0$

this is a m-th order equation in λ which can have ***at most m distinct solutions*** *(roots of the characteristic polynomial) - <u>can be complex even though S is real.</u>*

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 summary

- Preprocessing:
  - Given graph of links, build matrix **P**.
  - From it compute **a**.
  - The entry a_i is a number between 0 and 1: the pagerank of page *i*.
- Query processing:
  - Retrieve pages meeting query.
  - Rank them by their pagerank.
  - Order is query-*independent*.

The reality

- Pagerank is used in google, but so are many other clever heuristics.

Pagerank: Issues and Variants

- How realistic is the random surfer model?
  - What if we modeled the back button? [Fagi00]
  - Surfer behavior sharply skewed towards short paths [Hube98]
  - Search engines, bookmarks & directories make jumps non-random.
- Biased Surfer Models
  - Weight edge traversal probabilities based on match with topic/query (non-uniform edge selection)
  - Bias jumps to pages on topic (e.g., based on personal bookmarks & categories of interest)

Topic-Specific Pagerank [Have02]

- Conceptually, we use a random surfer who teleports, with say 10% probability, using the following rule:
  - Selects a category (say, one of the 16 top level ODP categories) based on a query & user -specific distribution over the categories
  - Teleport to a page uniformly at random within the chosen category
- Sounds hard to implement: can't compute PageRank at query time!

ODP = Open Directory Project

Topic-Specific Pagerank [Have02]

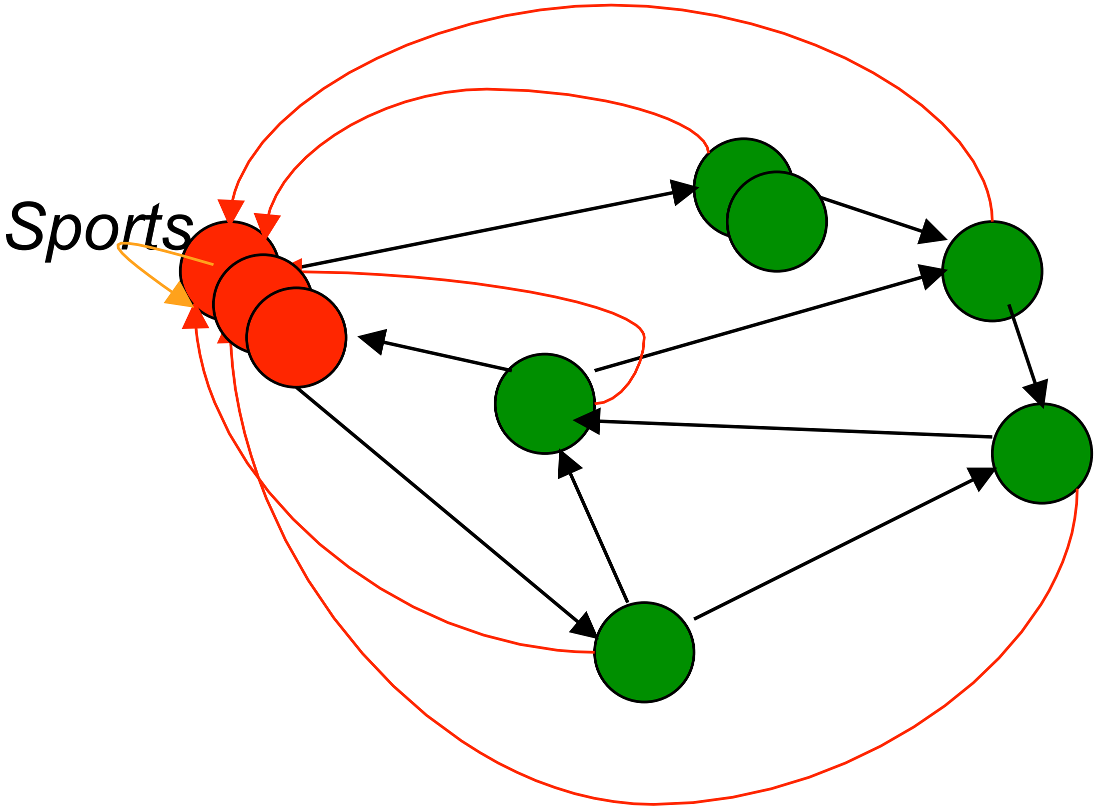
- Implementation
  - **Offline**: Compute pagerank distributions wrt to *individual* categories
    - Query-independent model as before
    - Each page has multiple pagerank scores – one for each ODP category, with teleportation only to that category
  - **Online**: Distribution of weights over categories computed by query context classification
    - Generate a dynamic pagerank score for each page – weighted sum of category-specific pageranks

Influencing PageRank ("Personalization")

- Input:
  - Web graph W
  - influence vector **v**

    $\mathbf{v}$: (page $\rightarrow$ degree of influence)
- Output:
  - Rank vector **r**:
    (page $\rightarrow$ page importance wrt **v**)
- $\mathbf{r} = \mathrm{PR}(W, \mathbf{v})$

Non-uniform Teleportation



Teleport with 10% probability to a Sports page

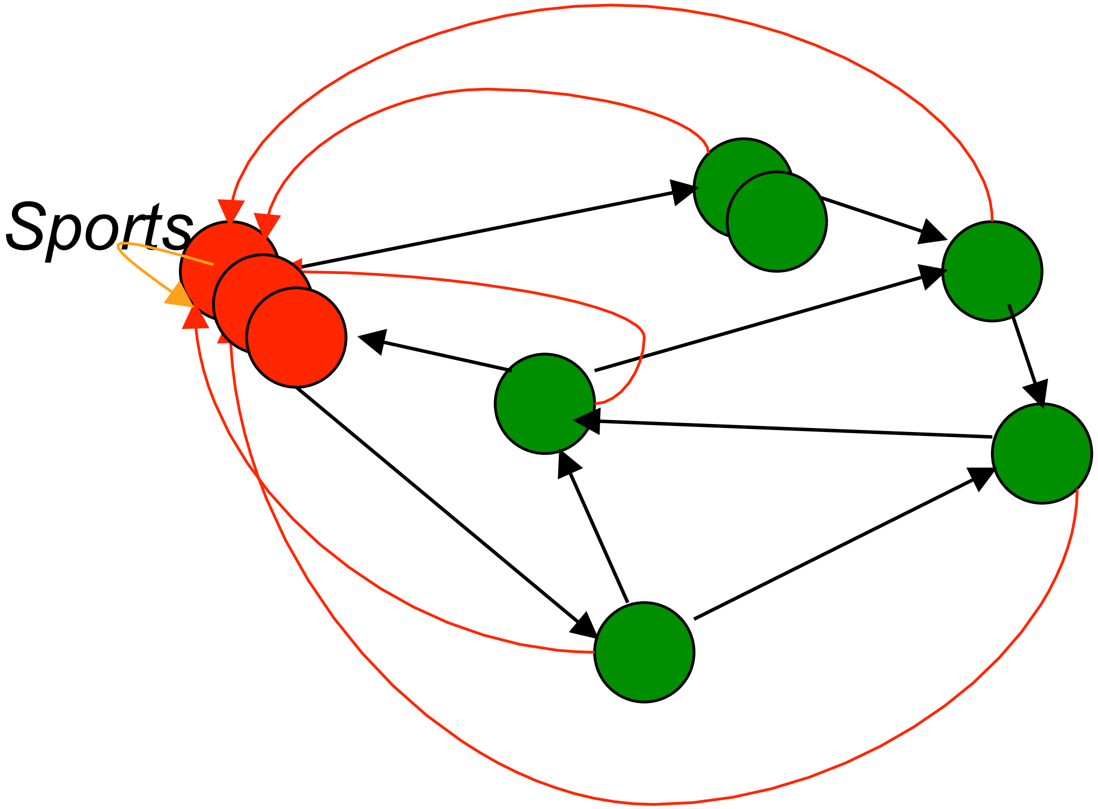
Interpretation of Composite Score

- For a set of personalization vectors $\{\mathbf{v}_j\}$

$$\sum_j [w_j \cdot PR(W, \mathbf{v}_j)] = PR(W, \sum_j [w_j \cdot \mathbf{v}_j])$$

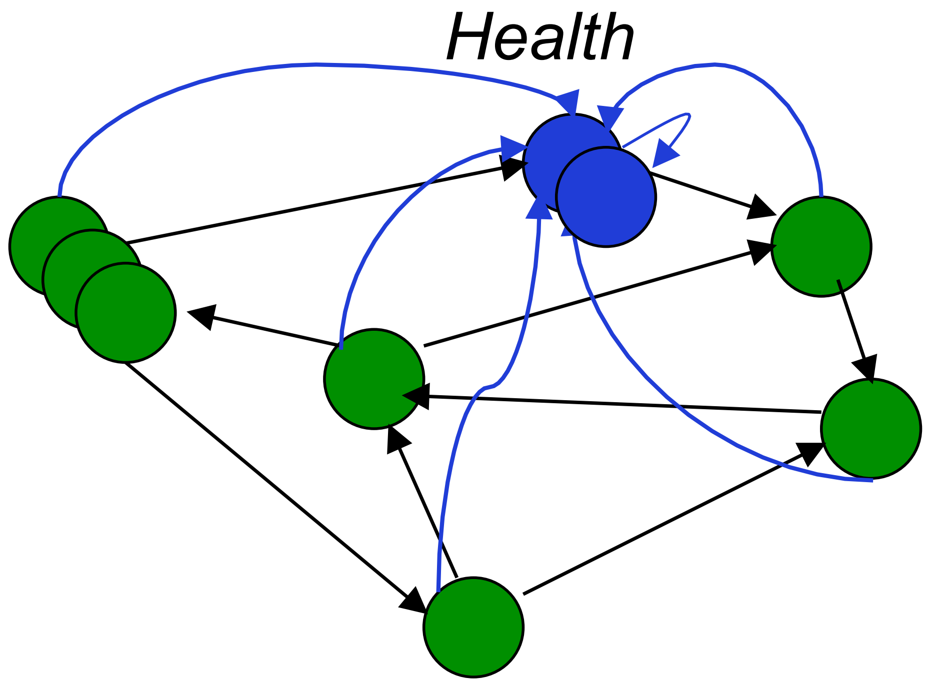
- Weighted sum of rank vectors itself forms a valid rank vector, because PR() is linear wrt $\mathbf{v}_j$

Interpretation



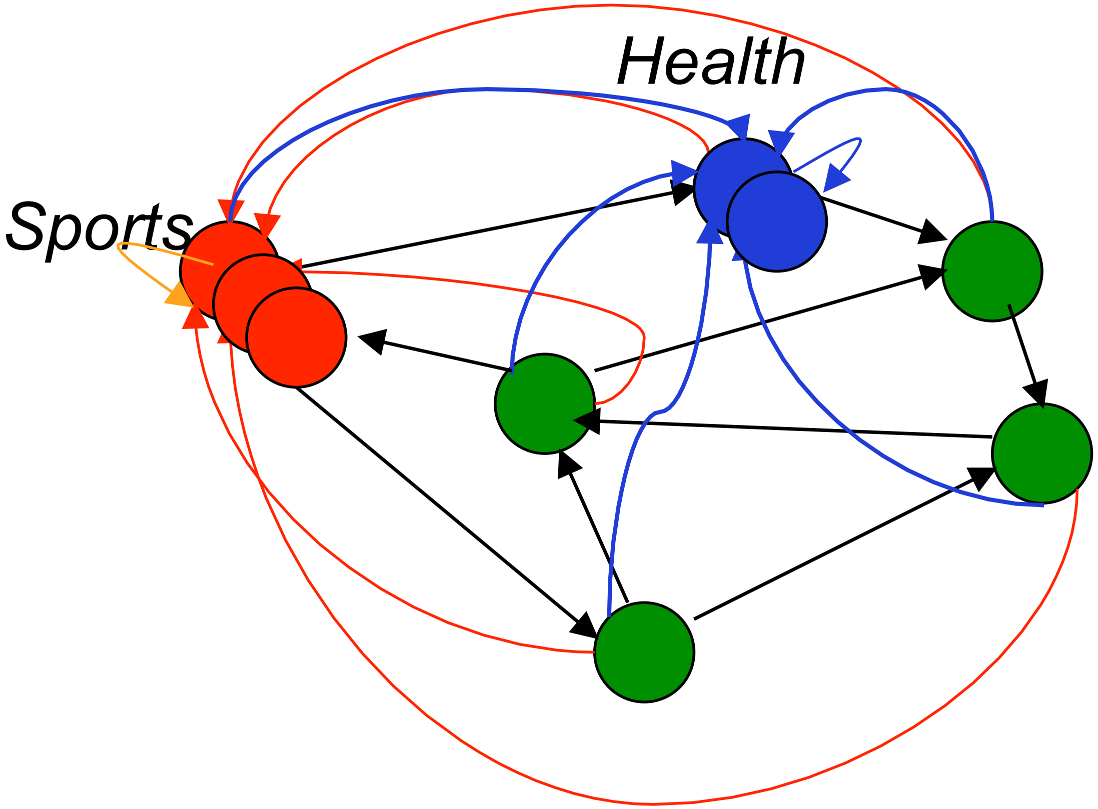
10% Sports teleportation

Interpretation



10% Health teleportation

Interpretation



pr = (0.9 PR_{sports} + 0.1 PR_{health}) gives you:
9% sports teleportation, 1% health teleportation

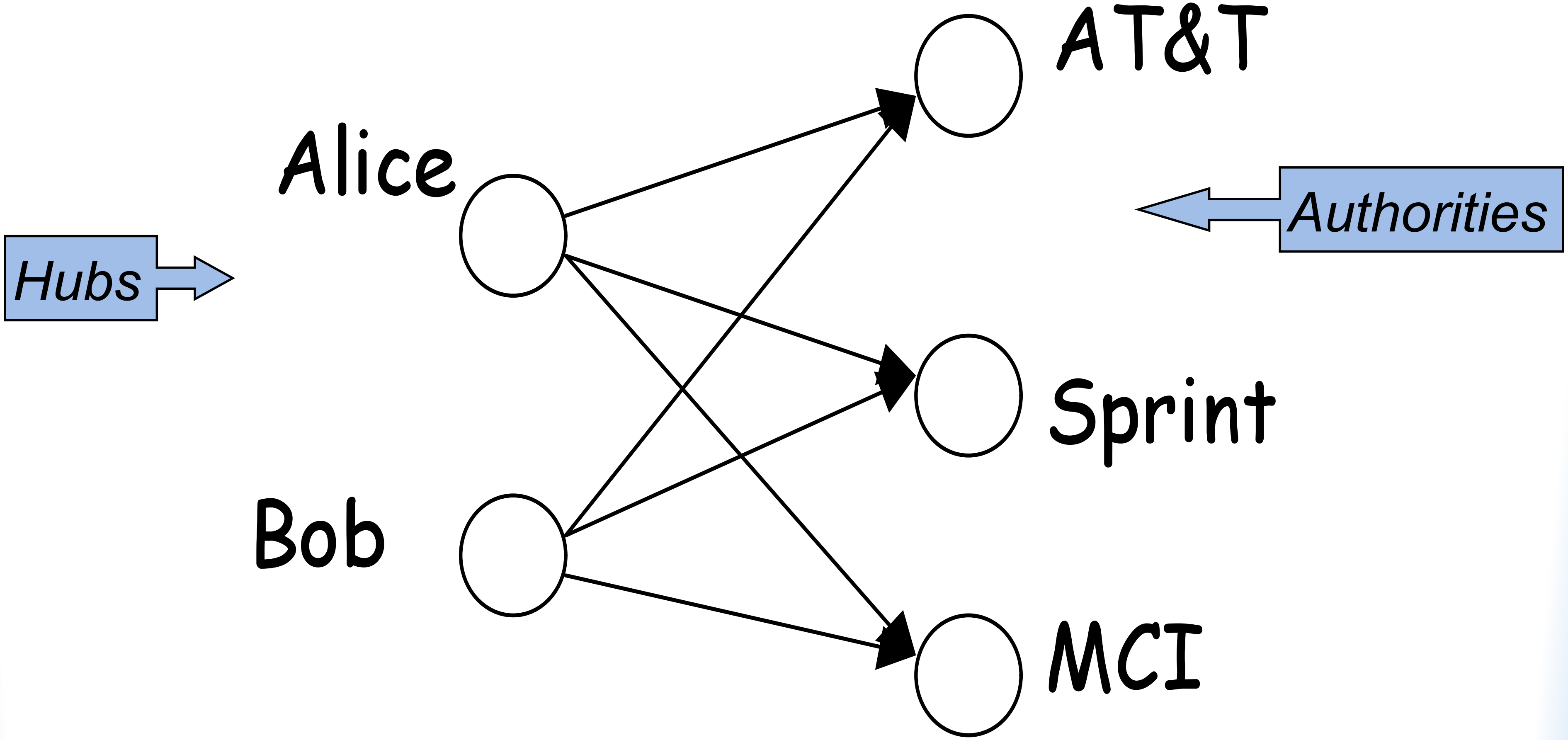
Hyperlink-Induced Topic Search (HITS) - Klei98

- 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.
- Best suited for "broad topic" queries rather than for page-finding queries.
- Gets at a broader slice of common *opinion*.

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



Long distance telephone companies

High-level scheme

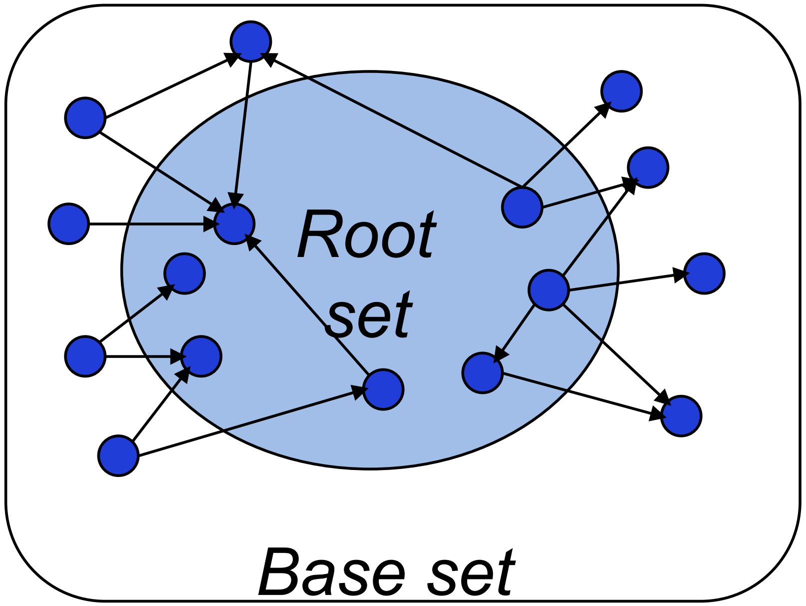
- Extract from the web a base set of pages that *could* be good hubs or authorities.
- From these, identify a small set of top hub and authority pages;

  →iterative algorithm.

Base set

- Given text query (say ***browser***), use a text index to get all pages containing ***browser.***
  - Call this the <u>root set</u> 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 <u>base set</u>.

Visualization



Assembling the base set [Klei98]

- Root set typically 200-1000 nodes.
- Base set may have up to 5000 nodes.
- How do you find the base set nodes?
  - Follow out-links by parsing root set pages.
  - Get in-links (and out-links) from a *connectivity server.*
  - (Actually, suffices to text-index strings of the form ***href="URL"*** to get in-links to *URL*.)

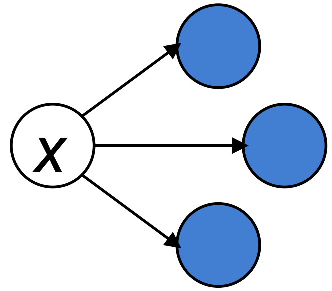
Distilling hubs and authorities

- 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.

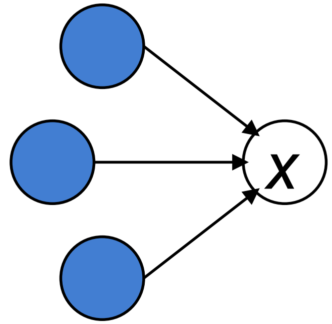
Iterative update

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

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



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



Scaling

- To prevent the *h()* and *a()* values from getting too big, can scale down after each iteration.
- Scaling factor doesn't really matter:
  - we only care about the *relative* values of the scores.

How many iterations?

- Claim: relative values of scores will converge after a few iterations:
  - In fact, suitably scaled, *h()* and *a()* scores settle into a steady state!
- We only require the <u>relative orders</u> of the *h()* and *a()* scores – not their absolute values.
- In practice, ~5 iterations get you close to stability.

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
- ,I,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

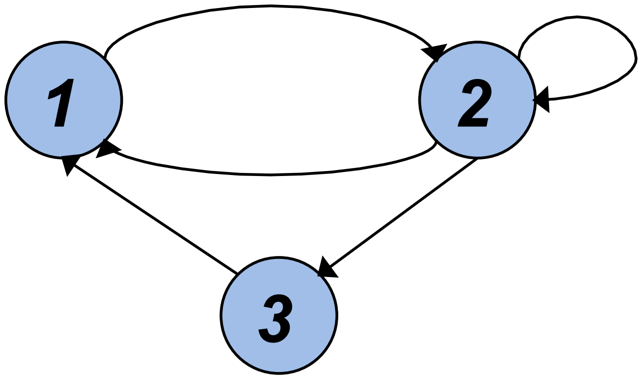
- The American School in Japan
- The Link Page
- ‰ªès—§ˆä“c¬ŠwZƒz[ƒƒy[ƒW
- Kids' Space
- ˆÀés—§ˆÀé¼•”¬ŠwZ
- ‹{é‹³ˆç‘åŠw•‘®¬ŠwZ
- KEIMEI GAKUEN Home Page (Japanese)
- Shiranuma Home Page
- fuzoku-es.fukui-u.ac.jp
- welcome to Miasa E&J school
- _“ÞìŒ§E‰¡•ls—§’†ì¼¬ŠwZ,Ìƒy
- http://www...p/~m_maru/index.html
- 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 <u>after</u> base set assembled
  - Iterative scoring is query-independent.
- Iterative computation <u>after</u> text index retrieval – significant overhead.

Proof of convergence

- $n \times n$ <u>adjacency matrix</u> **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**.*

Further, our algorithm is a particular, known algorithm for computing eigenvectors: the power iteration method.

Guaranteed to converge.

Issues

- Topic Drift
  - Off–topic pages can cause off–topic "authorities" to be returned
    - E.g., the neighborhood graph can be about a "super topic"
- Mutually Reinforcing Affiliates
  - Affiliated pages/sites can boost each others' scores
    - Linkage between affiliated pages is not a useful signal

Resources

- IIR Chap 21
- http://www2004.org/proceedings/docs/1p309.pdf
- http://www2004.org/proceedings/docs/1p595.pdf
- http://www2003.org/cdrom/papers/refereed/p270/kamvar-270-xhtml/index.html
- http://www2003.org/cdrom/papers/refereed/p641/xhtml/p641-mccurley.html