Introduction to

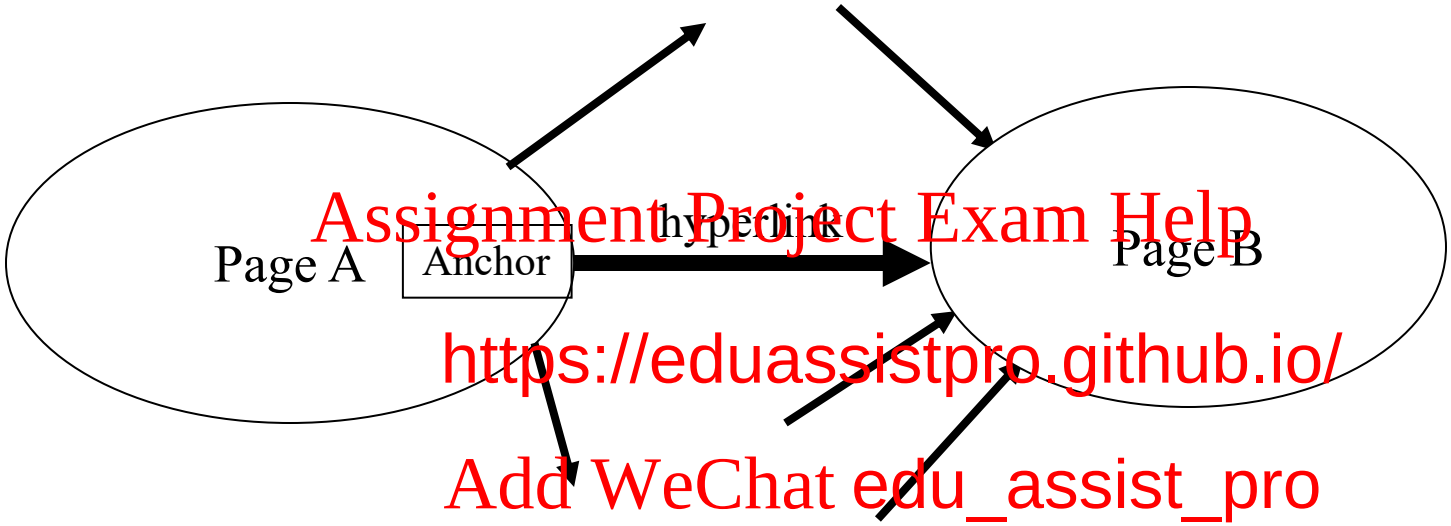
Informa I

Lecture 18: Li

Today's lecture

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

The Web as a Dire ph



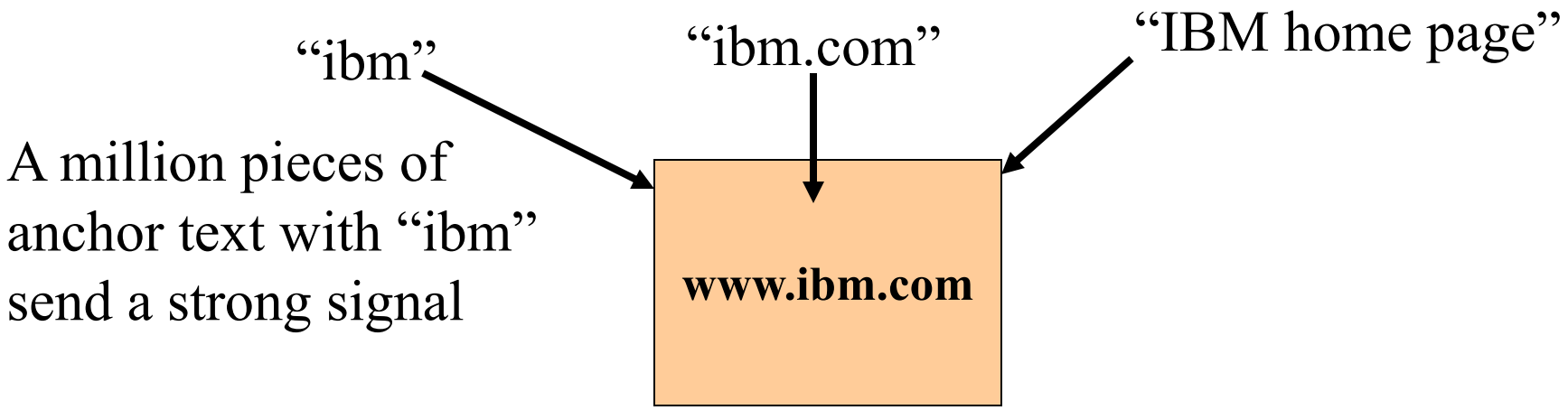
Assumption 1: A hyperlink between pages denotes author perceived relevance (quality signal)

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

Anchor Text

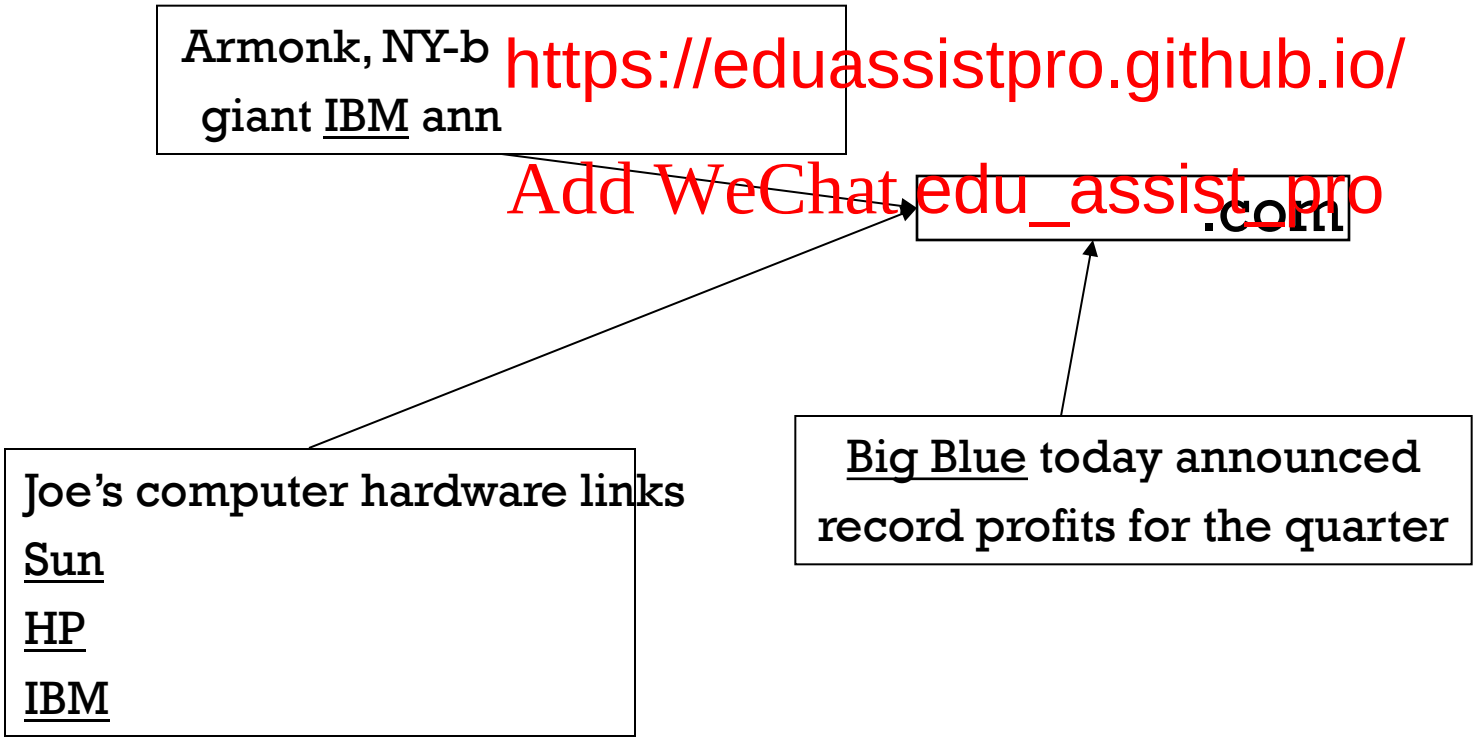
WWW Worm - McBryan [

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



Indexing anchor te

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



Indexing anchor te

- Can sometimes have unexpected side effects - *e.g.*, ***evil empire***.
- Can score anchor text with weight depending on the authority of t ite
  - E.g., if we were to assume th rom cnn.com or yahoo.com is authoritative, t e anchor text from them

Anchor Text

- Other applications
  - Weighting/filtering links in the graph
  - Generati s from anchor text

Citation Analysis

- Citation frequency
- Co-citation coupling frequency
  - Cocitations with a given author measures "impact"
  - Cocitation
- Bibliographic coupling fr
  - Articles that co-cite the same articles are related
- Citation indexing
  - Who is this author cited by? (Garfield 1972)
- Pagerank preview: Pinsker and Narin '60s

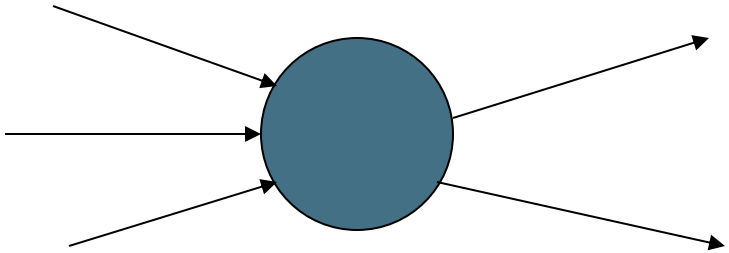
Query-independen ng

- First generation: using link counts as simple measures of popularity.

- Two basic sugg

  - Undirected
    - Each page gets a score = the 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 s
- More nuanced – use link co easure of static goodness (Lecture 7), with text match score

Spamming simple ity

- *Exercise*: How do you spam each of the following heuristics so your page gets a high score?

1. Each page gets a static score = the number of in-links plus the

2. Static score of a page = nu s in-links.

Ideas of Pagerank

- Inlinks as votes
  - www.stanford.edu has 23,400 inlinks
  - www.joe-schmoe.com has 1 inlink
- Web pages are
  - www.joe-schm
  - vs. www.stanford.edu → p2
- Are all inlinks equal?
  - Recursive question!

Pagerank scoring

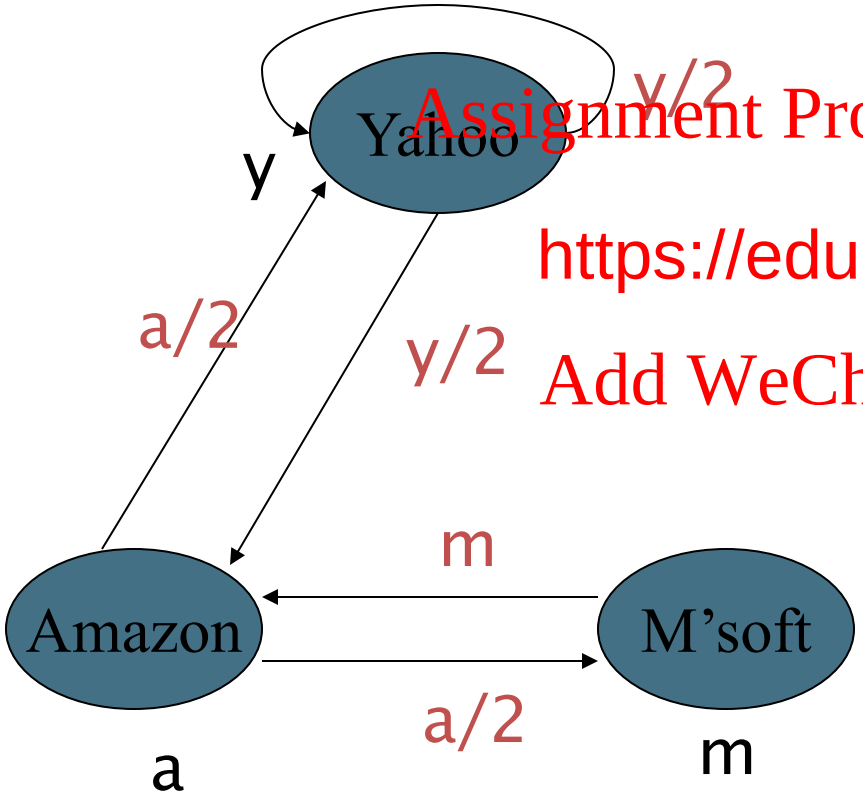
- Imagine a browser doing a *random walk* on web pages:
  - Start at a random page
  - At each step, go out along one of the links on that page, equally likely
- "In the steady state" each page has a long-term visit rate - use this as the page's score.

Example – the Sim "w" Model

The web in 1839

$$y = y/2 + a/2$$

$$a = y/2 + m$$

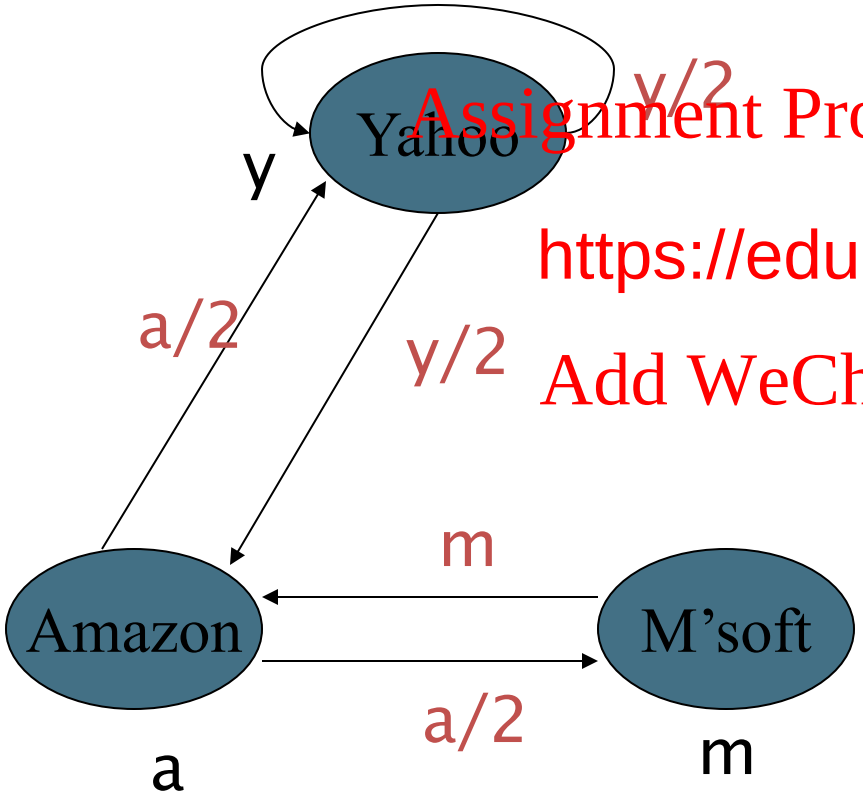


Solving the flow eq

- 3 equations, 3 unknowns, no constants
  - No unique solution
  - All solutions equivalent modulo scale factor
- Additional co ness
  - y+a+m = 1
  - y = 2/5, a = 2/5, m = 1/5
- Gaussian elimination method works for small examples, but we need a better method for large graphs

Example – the Sim "w" Model

The web in 1839



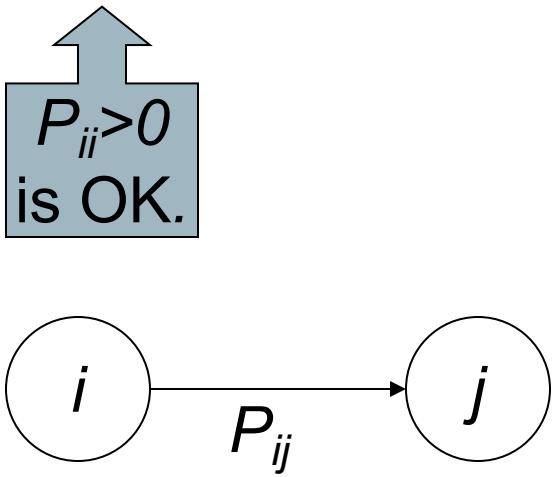
$$y_{new} = y_{old}/2 + a_{old}/2$$
$$a_{new} = y_{old}/2 + m_{old}$$
$$m = a_{old}/2$$

	1/3	5/12	...	2/5
	1/2	1/3	...	2/5
m =	1/3	1/6	1/4	... 1/5

Matrix-based characterization of the computation is simpler and more useful for the general case.

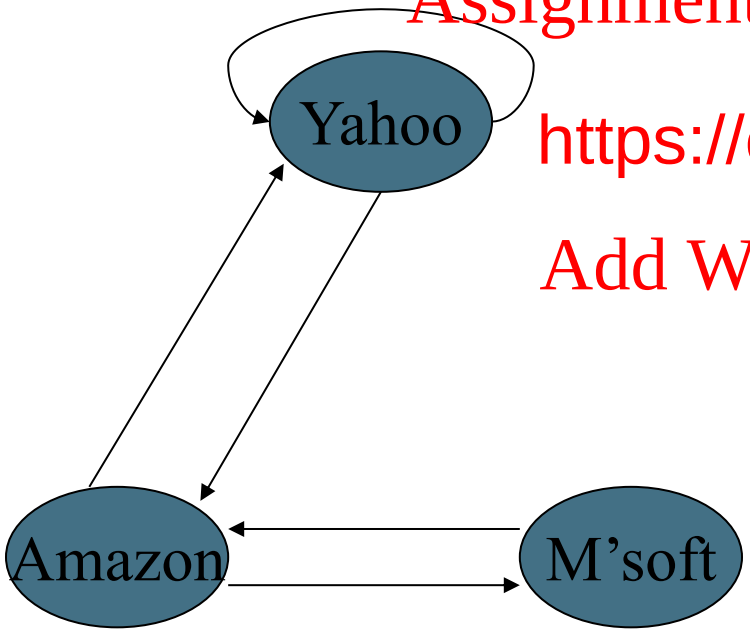
Markov chains

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



Markov chains

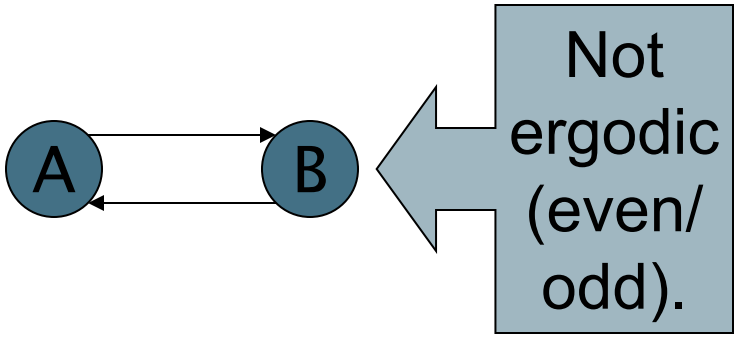
- Clearly, for all i, $\sum_{j=1}^{n} P_{ij} = 1.$
- Markov chains are abstractions of random walks.



	a	m
	0	1/2
m	0	1	0

Ergodic Markov ch

- A Markov chain is ergodic if
  - you have a path from any state to any other
  - For any sta transient time T_0, the probab t a fixed time $T>T_0$ is nonzero.



Ergodic Markov ch

- For any ergodic Markov chain, there is a unique long-term visit rate for each state.
  - *Steady-sta* *ution.*
- Over a long time-perio t each state in proportion to this rate.
- It doesn't matter where we start.

Probability vectors

- A probability (row) vector $\mathbf{x} = (x_1, \ldots x_n)$ tells us where the walk is at any point.
- E.g., (000…1…000) means we're in state i.

1 i

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

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

Change in probabil or

- If the probability vector is **x** = $(x_1, \ldots x_n)$ at this step, what is it at the next step?

- Recall that r tion prob. Matrix **P** te next from state *i*.

- So from **x**, our next state is distributed as **xP**.

How do we compu ector?

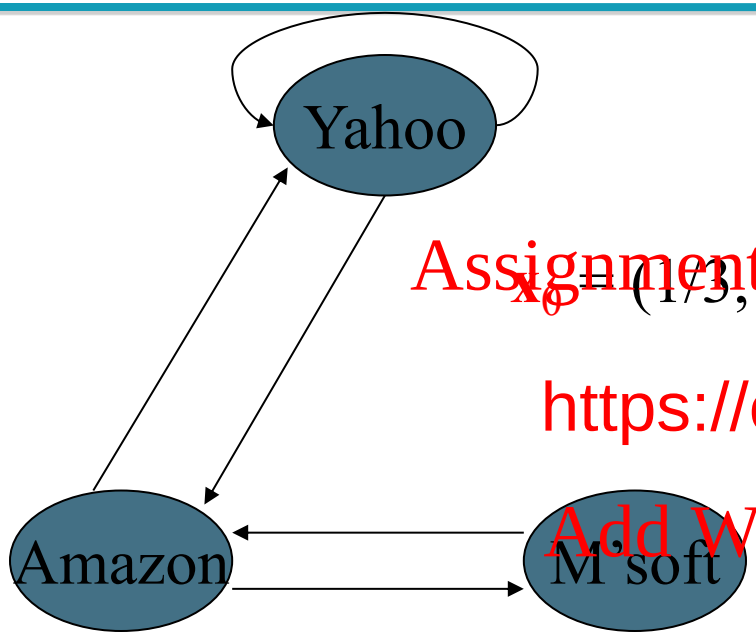
- Let **a** = $(a_1, \ldots a_n)$ denote the row vector of steady-state probabilities.
- If our current position is described by **a**, then the next step is di
- But **a** is the steady state, so
- Solving this matrix equatio .

  - 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 compu

- Recall, regardless of where we start, we eventually reach the steady state **a**.
- Start with any distribution (say **x**=(*1/n, 1/n, …, 1/n*)).
- After one ste
- after two steps at **xP**^{2}, then o on.
- "Eventually" means for "lar **a**.
- Algorithm: multiply **x** by increasing powers of **P** until the product looks stable.

Power Iteration: Ex



$x_0 = (1/3, 1/3, 1/3)$

P

	y	a	m
y	1/2	1/2	0
a	1/2	0	1/2
m	0	1	0

$$y_{new} = y_{old}/2 + a_{old}/2$$
$$a_{new} = y_{old}/2 + m_{old}$$
$$m_{new} = a_{old}/2$$

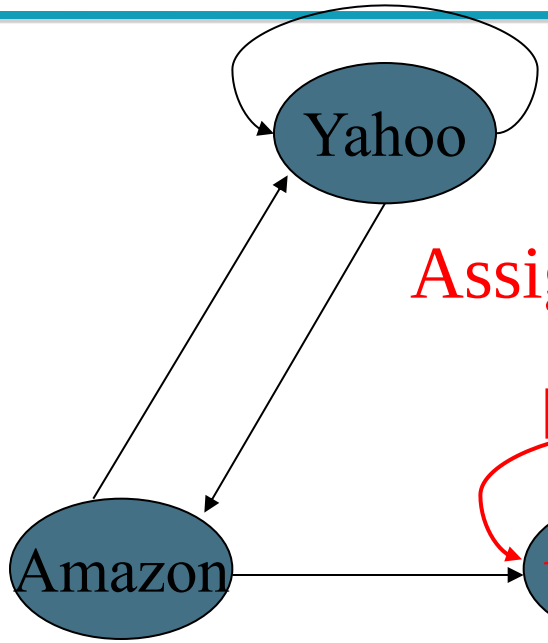
	x_0	x_1	x_2	x_3		x_t
y	1/3	1/3	5/12	3/8		2/5
a =	1/3	1/2	1/3	11/24	. . .	2/5
m	1/3	1/6	1/4	1/6		1/5

Spider traps

- A group of pages is a spider trap if there are no links from within the group to outside the group
  - Random surfer gets trapped

- Spider traps v eeded for the random walk t

Microsoft becomes a trap



	y	a	m
y	1/2	1/2	0
a	1/2	0	1/2
m	0	0	1

$$\begin{pmatrix} y \\ a \\ m \end{pmatrix} = \begin{pmatrix} 1/3 & 1/3 & 1/4 & 5/24 & & 0 \\ 1/3 & 1/6 & 1/6 & 1/8 & \cdots & 0 \\ 1/3 & 1/2 & 7/12 & 2/3 & & 1 \end{pmatrix}$$

Random teleports

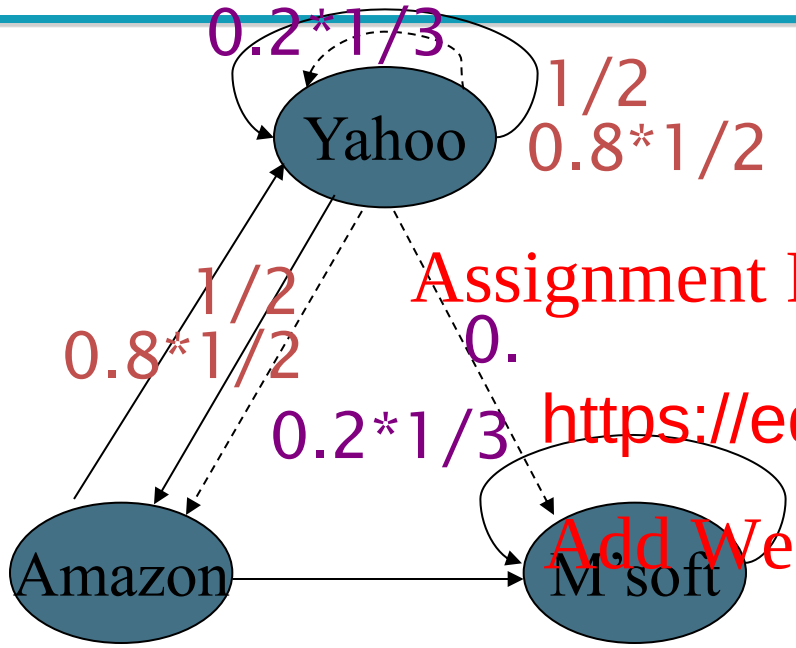
- The Google solution for spider traps
- At each time step, the random surfer has two options:

  - With probab em

  - With probability 1-β, jump t uniformly at random

  - Common values for β are in the range 0.8 to 0.9
- Surfer will teleport out of spider trap within a few time steps

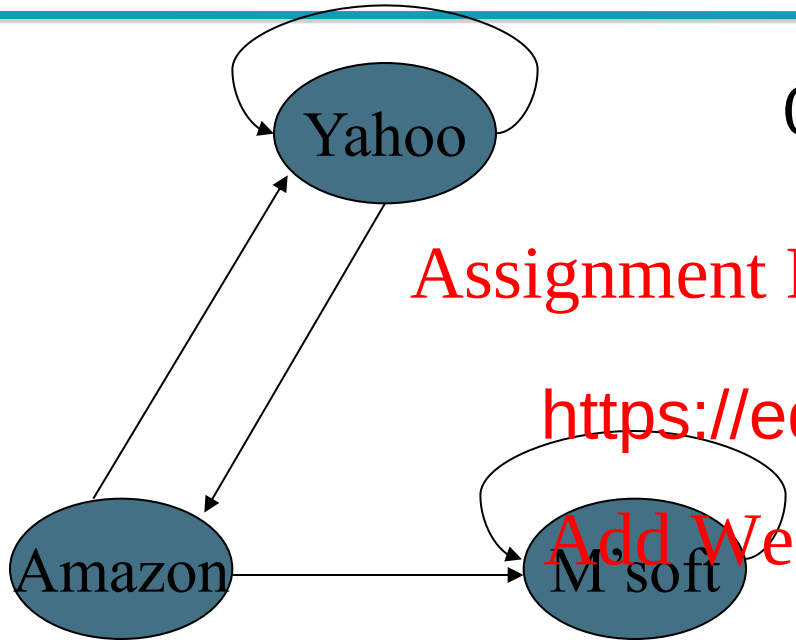
Random teleports ( )



$$0.8 \begin{bmatrix} & & \\ & & \\ 0 & 0 & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

y	7/15	7/15	1/15
a	7/15	1/15	7/15
m	1/15	1/15	13/15

Random teleports ( )



$$0.8 \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

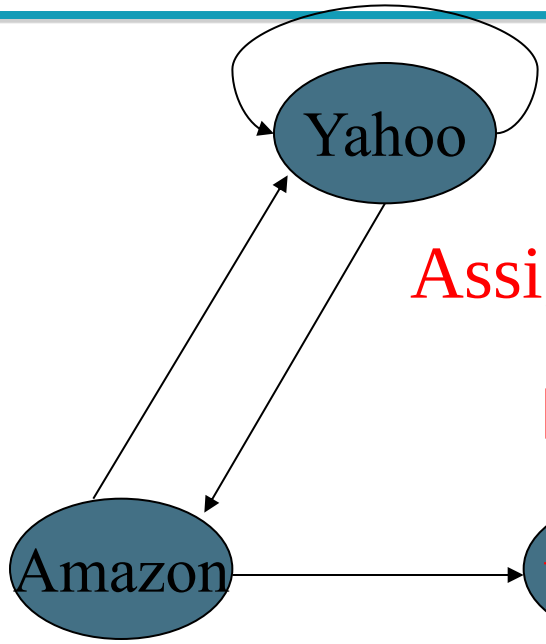
$$\begin{bmatrix} 5 & 7/15 & 1/15 \\ 5 & & \\ 1/15 & & 13/15 \end{bmatrix}$$

$$\begin{matrix} y \\ a \\ m \end{matrix} = \begin{matrix} 0.33 & 0.33 & 0.28 & 0.26 \\ 0.33 & 0.20 & 0.20 & 0.18 \\ 0.33 & 0.47 & 0.52 & 0.56 \end{matrix} \quad \ldots \quad \begin{matrix} 7/33 \\ 5/33 \\ 7/11 \end{matrix}$$

Dead ends

- Pages with no outlinks are "dead ends" for the random surfer
  - Nowhere to go on next step
- Especially co Engines
  - URLs that have not yet been

Microsoft become end



$$0.8 \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1/2 \\ 0 & 0 & 0 \end{bmatrix} + 0.2 \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

$$\begin{bmatrix} 5 & 7/15 & 1/15 \\ 5 & & \\ & 1/15 & 1/15 \end{bmatrix}$$

Non-stochastic!

$$\begin{matrix} y \\ a \\ m \end{matrix} = \begin{matrix} 0.33 & 0.33 & 0.262 & 0.216 & & 0 \\ 0.33 & 0.20 & 0.182 & 0.143 & \ldots & 0 \\ 0.33 & 0.20 & 0.129 & 0.111 & & 0 \end{matrix}$$

Dealing with dead-

- Teleport
  - Follow random teleport links with probability 1.0 from dead-ends
  - Adjust matri
    - How?
- (Suggested by Google) pru p agate
  - Preprocess the graph to eliminate dead-ends
  - Might require multiple passes
  - Compute page rank on reduced graph
  - Approximate values for deadends by propagating values from reduced graph

Q: Why approximate values and why errors are insignificant?

Pagerank summary

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

$$\tilde{\mathbf{P}} = \frac{1}{n}$$

1: the pagerank of page *i*.

The reality

- Pagerank is used in google, but is hardly the full story of ranking
  - Many sophisticated features are used
  - Some addres
  - Machine lear heavily used
- Pagerank still very useful for t awl policy

Pagerank: issues and variants

- How realistic is the random surfer model?
  - (Does it matter?)
  - What if we modeled the back button?
  - Surfer behavior sharply skewed towards short paths
  - 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 Pager

- Goal – pagerank values that depend on query *topic*
- Conceptually, we use a random surfer who teleports, with say 10% probability, using the following rule:

  - Selects a topic (say, one of the 16 top level ODP categories) based on a query-specific distribution over the categories
  - Teleport to a page uniformly at random within the chosen topic

only randomly teleport to a subset of pages

- Sounds hard to implement: can't compute PageRank at query time!

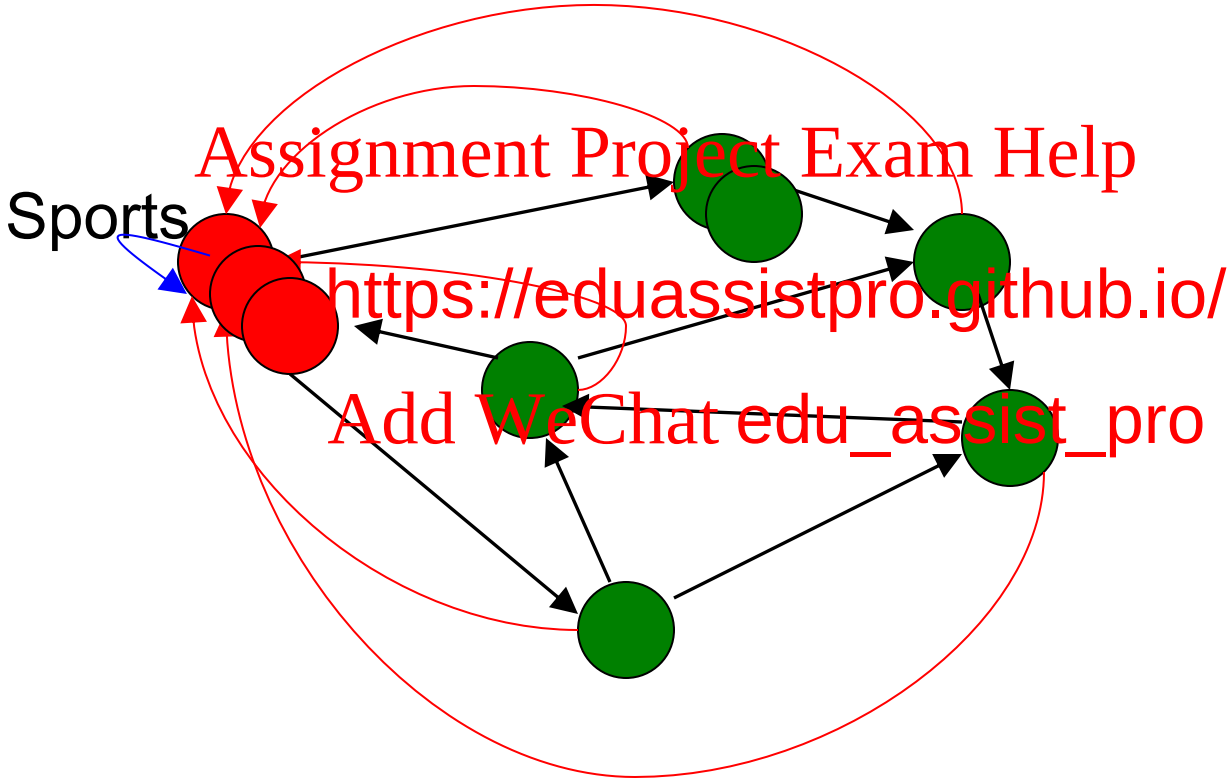
Topic Specific Pager

- **Offline**:Compute pagerank for *individual* topics
  - Query independent as before
  - Each page has multiple pagerank scores – one for each ODP category, wit at category
- **Online**: Query (distribution of weights over) topics
  - Generate a dynamic pagerank scor e – weighted sum of topic-specific pageranks

Influencing PageRan "personalization")

- Input:
  - Web graph *W*
  - Influence v topic
    v : (page)
- Output:
  - Rank vector **r**: (page → page importance wrt **v**)
- **r** = PR(*W* , **v**)

Non-uniform Tele n



Teleport with 10% probability to a Sports page

Interpretation of Co Score

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

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

Given a user's p , express as a combination of the "basis" v

PageRank as a Line m [Jeh & Widom, KDD 2003]

[Optional]

- Preference vectors **u** ➔ specifies the random teleport probability distribution
  - A column vector of n dimensions (n: # of vertices in G)
  - sum up to 1.

- Personalized y-state distribution over the vertic

  - Also a column vector of n dimensions
- v is determined by u via a linear system

$$\mathbf{v} = (1-\beta)\mathbf{P}^\top \mathbf{v} + \beta\mathbf{u}$$

[Optional]

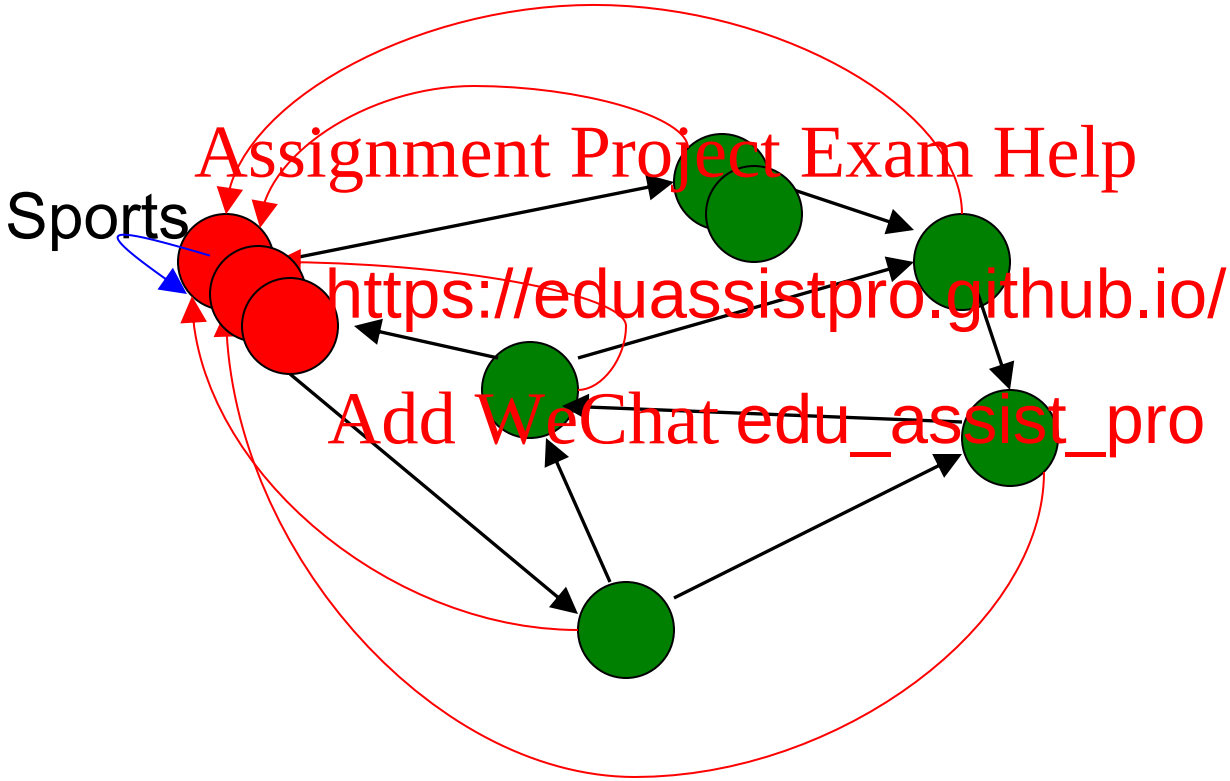
Linearity of PageR

- For two preference vectors $\mathbf{u}_1$ and $\mathbf{u}_2$, let $\mathbf{v}_1$ and $\mathbf{v}_2$ be the corresponding personalized page rank vectors, then we can prove that

$$\lambda\mathbf{v}_1 + (1-\lambda)\mathbf{v}_2 = (\quad)\beta(\lambda\mathbf{u}_1 + (1-\lambda)\mathbf{u}_2)$$

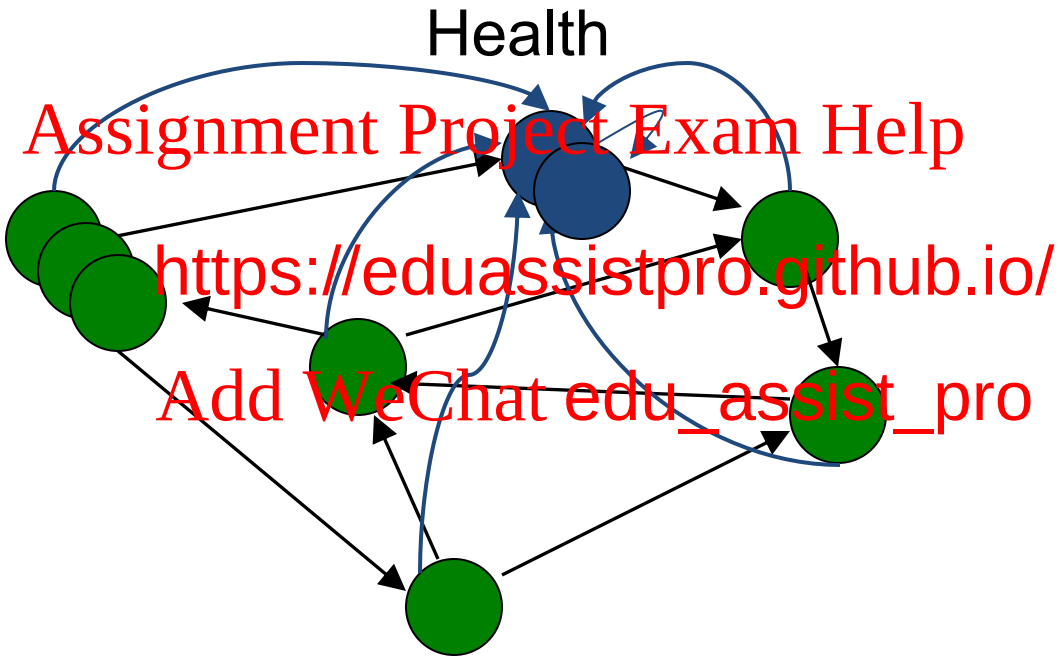
- Implication:
  - Personalized pagerank vectors induced by a linear combination of preference vectors can be computed as the same linear combination of corresponding personalized pagerank vectors.

Interpretation



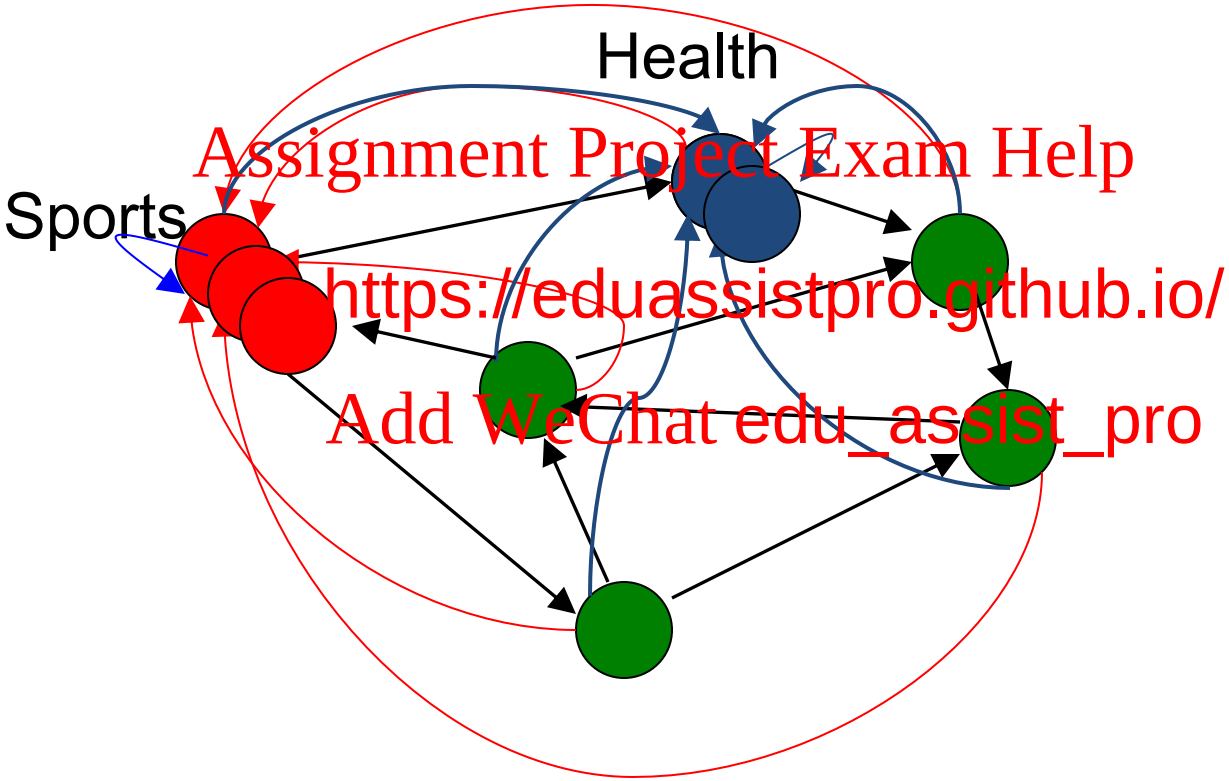
10% Sports teleportation

Interpretation



10% Health teleportation

Interpretation



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

Resources

- IIR Chap 21
- http://www2004.org/proceedings/docs/1p309.pdf
- http://www2 cs/1p595.pdf
- http://www2 refereed/p270/ kamvar-270-xhtml/index.ht
- http://www2003.org/cdrom fereed/p641/ xhtml/p641-mccurley.html
- Glen Jeh and Jennifer Widom: sScaling Personalized Web Search. KDD 2003.