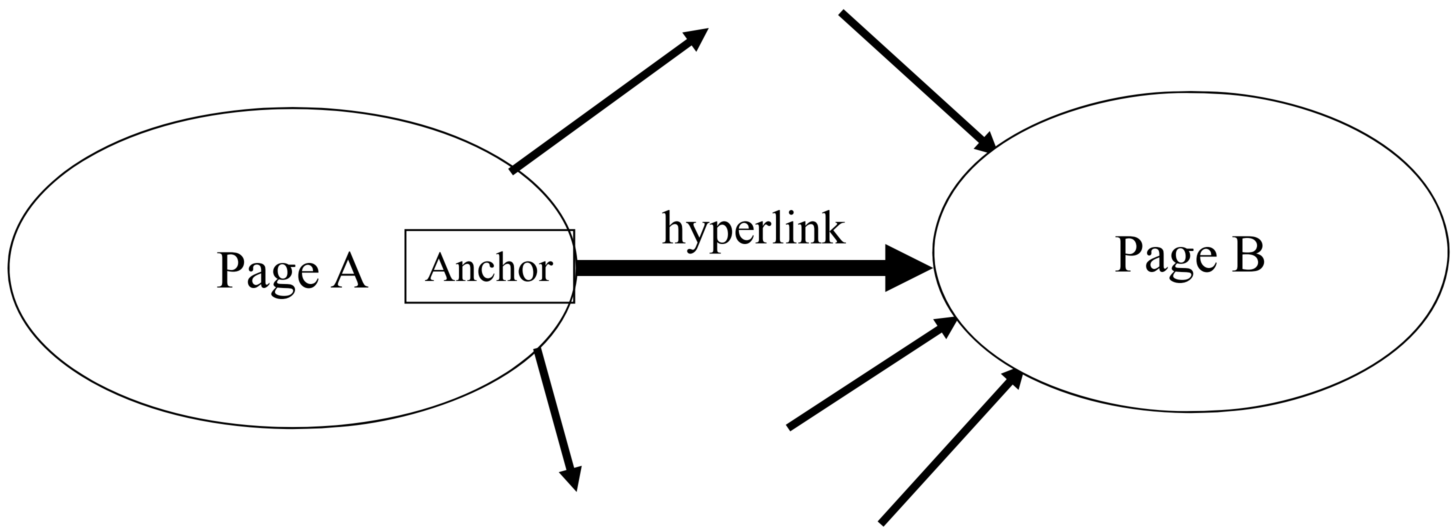
Link Analysis

Reference: Introduction to Information Retrieval
by C. Manning, P. Raghavan, H. Schutze

The Web as a Directed Graph

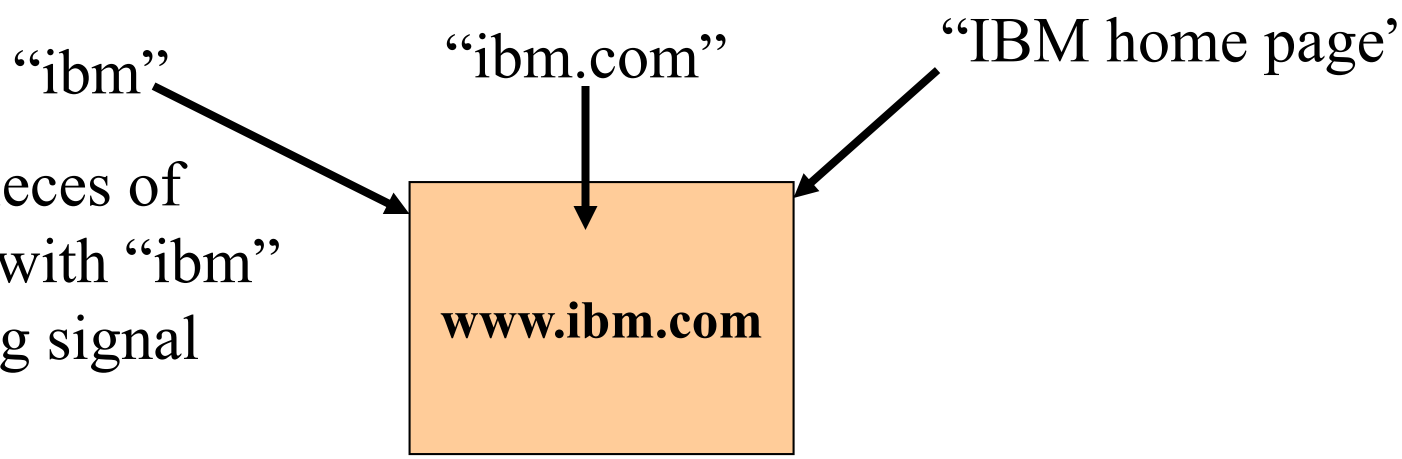


Assumption 1: A hyperlink between pages denotes
a conferral of authority (quality signal)

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

Anchor Text

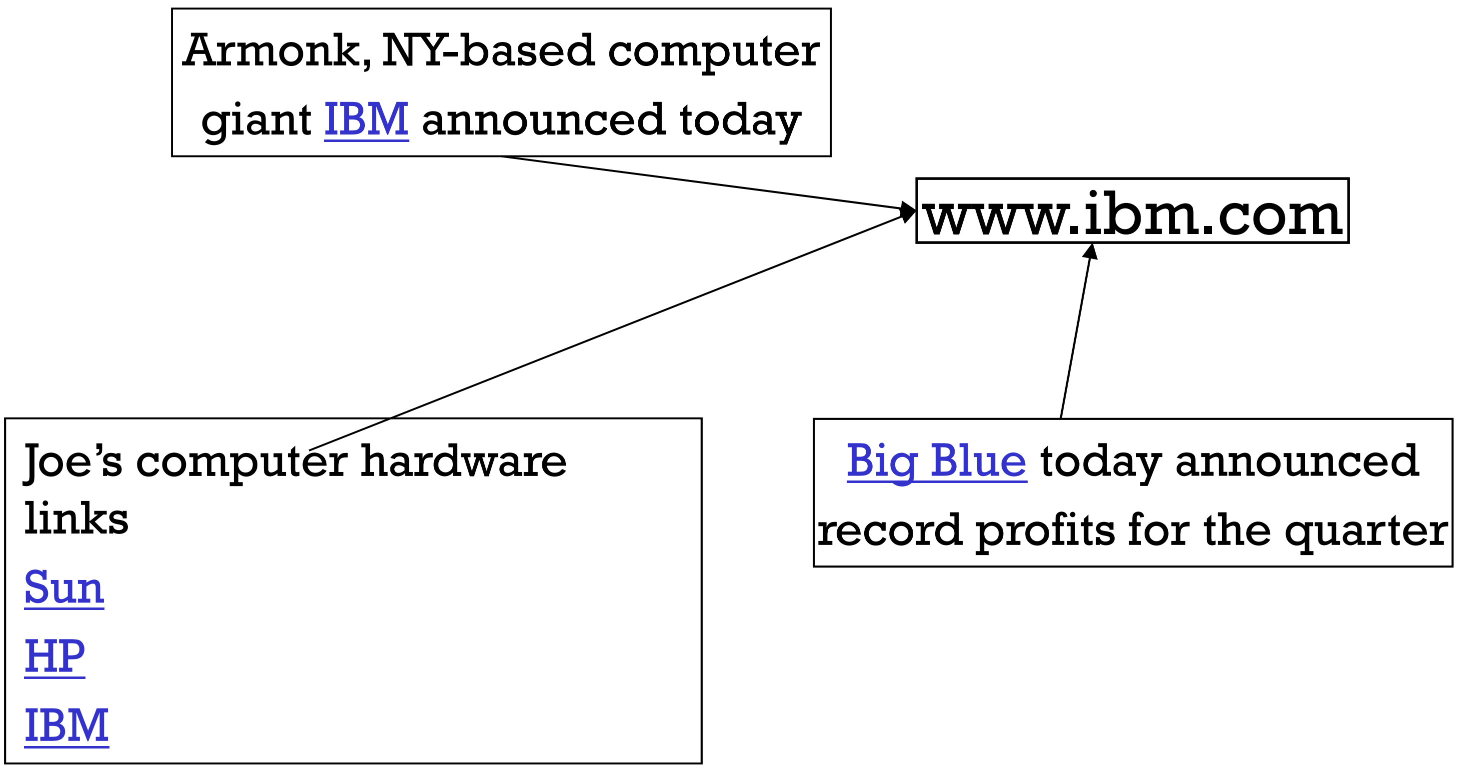
- For ***ibm*** how to distinguish between:
  - IBM’s home page (mostly graphical)
  - IBM’s copyright page (high term freq. for ‘ibm’)
  - Rival’s spam page (arbitrarily high term freq.)



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

Indexing anchor text

- When indexing a document *D*, include (with some weight) anchor text from links pointing to *D*.



Indexing anchor text

- Can sometimes have unexpected side effects - *e.g.*, ***evil empire***.
- Can score anchor text with weight depending on the authority of the anchor page's website
  - E.g., if we were to assume that content from cnn.com or yahoo.com is authoritative, then trust the anchor text from them

PageRank

- Ranking derived from the link structure

The Internet: How Search Works

https://www.youtube.com/watch?v=LVV_93mBfSU

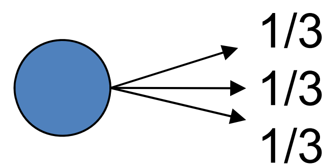
2’35’’

PageRank

- Ranking derived from the link structure
- Assign web page a numerical score, known as PageRank
- Given a query, search engine computes a composite score that combines a set of features, together with the PageRank score
- This composite score is used to provide a ranked list of web pages for the query

Pagerank scoring

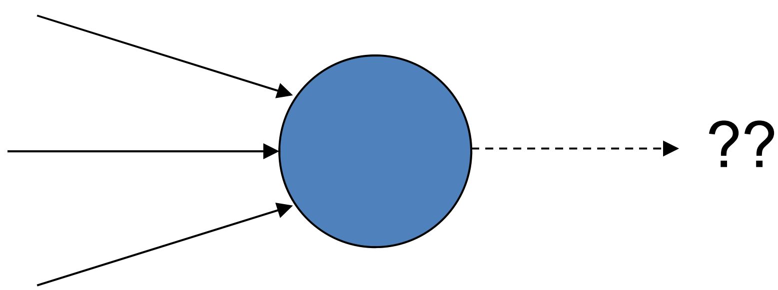
- Imagine a browser doing a random walk on web pages:
  - Start at a random page



  - 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.



  - At a dead end, jump to a random web page.

Teleporting

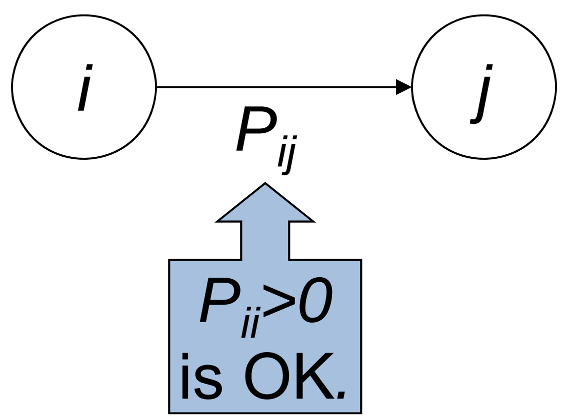
- At any non-dead end, with probability 10%, jump to a random web page.
  - With remaining probability (90%), go out on a random link.
  - 10% - a parameter.

Result of teleporting

- Now cannot get stuck locally.
- There is a long-term rate at which any page is visited
- How do we compute this visit rate?
- Use the theory of Markov chain to claim that when the surfer follows this combined process for long time, the score of a page is called the PageRank score.

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 transition probability matrix entry P_{ij} tells us the probability of j being the next state, given we are currently in state i.

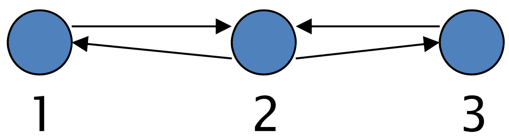


Markov chains

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

Adjacency Matrix

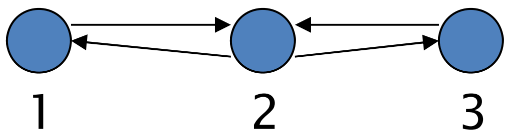
- Given three web pages with a web graph of three nodes 1, 2 and 3.



- The adjacency matrix A is defined as:
  if there is a hyperlink from page i to page j,
  then A_{ij} =1, otherwise, A_{ij} = 0

$$A = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Constructing Transition Probability Matrix



$$A = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

The steps for producing the transition probability matrix P:

1. If a row of A has no 1's, then replace each element by 1/N where N is the number of nodes
2. For all other rows:
   - Divide each 1 by the number of 1's in its row

Let G be the matrix after the above two operations.

$$G = \begin{pmatrix} 0 & 1 & 0 \\ 1/2 & 0 & 1/2 \\ 0 & 1 & 0 \end{pmatrix}$$

Suppose that the teleport probability is α

3. P = (1-α)G + αE where E is a matrix with all entries 1/N

For example, If α = 0.5

$$P = \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}$$

Probability vectors

- A probability (row) vector **x** = $(x_1, \ldots x_n)$ tells us where the walk is at any point.
- E.g., (0 0 0…1…0 0 0) 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 **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**
  - The one after that is $\mathbf{xP}^2$, then $\mathbf{xP}^3$, etc.
  - (Where) Does the process converge?

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.

Power Iteration for Random Walk

- Given the initial distribution x_0, and the transition probability matrix P, the distribution after one step is:

  $x_1 = x_0 P$

- Suppose $x_0 = (\ 1\ \ 0\ \ 0\ )$

$$\vec{x_0}P = \begin{pmatrix} 1/6 & 2/3 & 1/6 \end{pmatrix} = \vec{x_1}.$$

- After two steps, it is:

$$\vec{x_1}P = \begin{pmatrix} 1/6 & 2/3 & 1/6 \end{pmatrix} \begin{pmatrix} 1/6 & 2/3 & 1/6 \\ 5/12 & 1/6 & 5/12 \\ 1/6 & 2/3 & 1/6 \end{pmatrix}$$

$$= \begin{pmatrix} 1/3 & 1/3 & 1/3 \end{pmatrix} = \vec{x_2}.$$

Power Iteration for Random Walk

- Continuing in this fashion gives a sequence of probability vectors.

$\vec{x}_0$	1	0	0
$\vec{x}_1$	1/6	2/3	1/6
$\vec{x}_2$	1/3	1/3	1/3
$\vec{x}_3$	1/4	1/2	1/4
$\vec{x}_4$	7/24	5/12	7/24
...	...	...	...
$\vec{x}$	5/18	4/9	5/18

- We can see that the distribution converges to the steady state.
- The steady state probability is the PageRank value.

Another Method

- Let **a** = *(a_1, … 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.

Pagerank Application

- Pagerank values are independent of user queries
- Pagerank is used in Google and other engines, but is hardly the full story of ranking
  - Many sophisticated features are used
  - Some address specific query classes
  - Machine learned ranking heavily used
- Pagerank still very useful for things like crawl policy

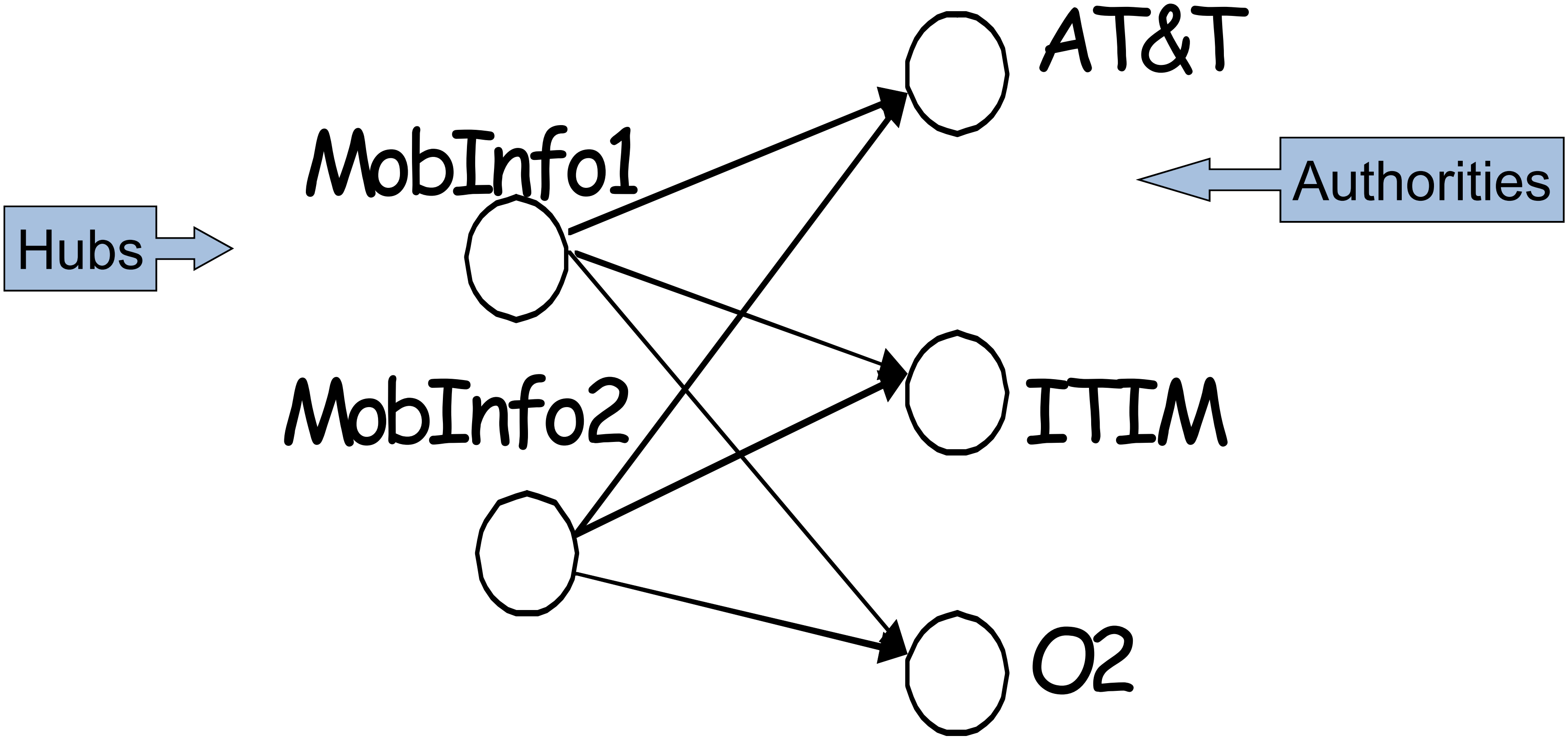
Hyperlink-Induced Topic Search (HITS)

- 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.
- 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 Basic Idea



Mobile telecom companies

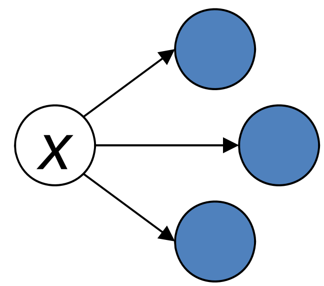
Distilling hubs and authorities

- Compute, for each page *x*, 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)*; ⇦ Key
- After iterations
  - output pages with highest *h()* scores as top hubs
  - highest *a()* scores as top authorities.

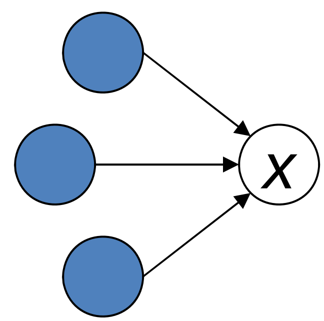
Iterative update

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

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



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

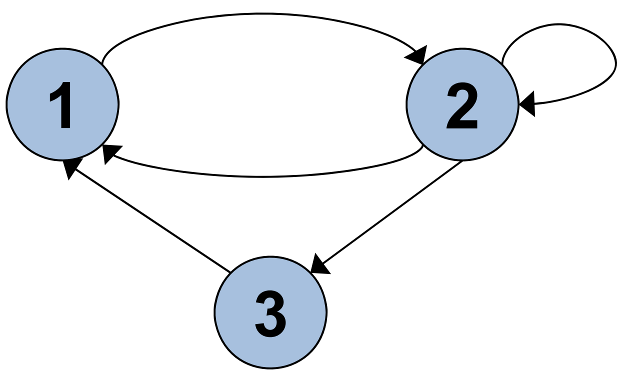


Scaling

- 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.

Adjacency Matrix

- $n \times n$ 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.



	1	2	3
1	0	1	0
2	1	1	1
3	1	0	0

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

Iterative Update Scheme:

- **h←Aa**
- **a←A^T^h** → Recall A^T is the transpose of A.

Analytical Solution:

- Substituting these into one another:
  h←AA^T^h and **a←A^T^Aa**
- As a result: $\boldsymbol{h} = (\boldsymbol{1}/\boldsymbol{\lambda_h})\boldsymbol{A}\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{h}$

$$\boldsymbol{a} = (\boldsymbol{1}/\boldsymbol{\lambda_a})\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{A}\boldsymbol{a}$$

  where λ_h and λ_a denote the eigenvalue of AA^T and A^TA respectively.

- Can be solved by **power iteration method**

Key Consequences

- The iterative update scheme is equivalent to the power iteration method for computing the eigenvalues of AA^T and A^TA
- Provided that the principal eigenvalue of AA^T is unique, the iterative update scheme can compute *h* and *a* by settling a steady-state values
- We can also use any method for computing the principal eigenvector of a matrix.

Eigenvector Method

- Assemble the target subset of web pages, form the graph induced by their hyperlinks and compute AA^T and A^TA
- Compute the principal eigenvectors of AA^T and A^TA to form the vector of hub scores *h* and authority scores *a*
- Output the top-scoring hubs and top-scoring authorities.