Information Retrieval
(Web Search)

Fabio Aiolli
http://www.math.unipd.it/~aiolli

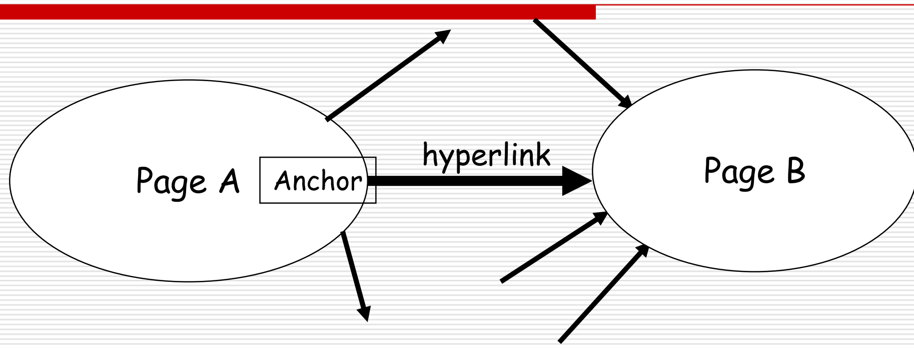
Dipartimento di Matematica Pura ed Applicata
Università di Padova

Anno Accademico 2008/2009

Web Search before Google

- Web Search Engines (WSEs) of the first generation (up to 1998)
  - Identified relevance with topic-relateness
  - Based on keywords inserted by web page creators (META tags)
  - Preprocessing (HTML tags removal, …), the only difference with standard text search
- Problems
  - Web pages are multimedia items and their relevance determined by non-testual content
  - Many Web pages, often use evocative (as opposed to descriptive) language

The Web as a Directed Graph



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

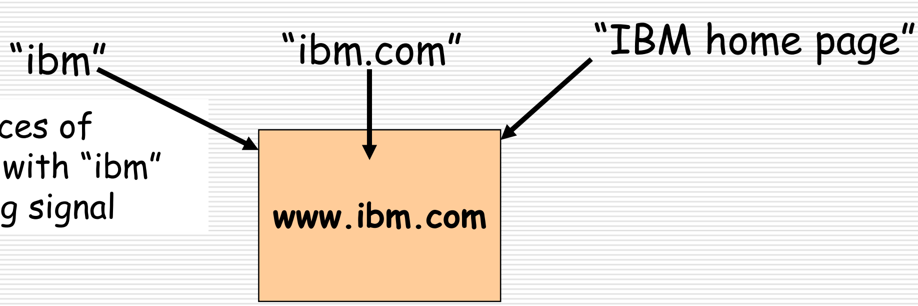
Assumption 2: The anchor of the hyperlink describes the target page (textual context)

Anchor Text

WWW Worm - McBryan [Mcbr94]

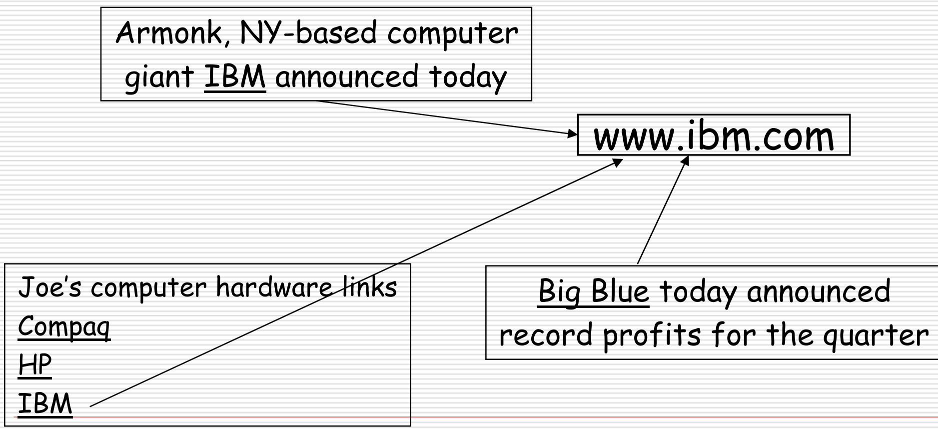
- For ***ibm*** how to distinguish between:
  - IBM's home page (mostly graphical)
  - IBM's copyright page (high term freq. for 'ibm')
  - Rival's spam page (arbitrarily high term freq.)

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*.



Indexing anchor text

- Can sometimes have unexpected side effects, e.g. derogatory phrases
- Can index anchor text with less weight.

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

Web Search after Google

- Web Search Engines (WSEs) of the second generation (from 1998 onwards)
  - Identify relevance with topic-relateness and authoritativeness
    - Independent by the particular format of the Web site
    - Relevance computation is more selective
- This has been possible by the development of Link-based Ranking Schemes (LRSs) algorithms which compute authoritativeness exploiting the hyperlink structure of the Web
- The Web can be seen as a network of recommendations, a social network. Social networks analysis has been applied in many contexts in the past, including epidemiology, espionage and scientific production

Spam Web Sites

- Spam Web Sites (SWSs) are Web pages designed to manipulate WSE ranking schemes, generally for commercial purposes
  - First Generation WSEs
    - Including deceptive self-description in the HTML META tag
    - Including "invisible words" (i.e. displayed in the same color as the background) or words typeset in tiny fonts, in order to deceive tfidf-based ranking schemes
  - Second Generation WSEs
    - LRSs would seem to be more robust, since SWSs are not authoritative, but naive LRSs may be fooled by artificially conferring authority onto SWSs
    - Adversarial IR to outwit companies specialized in promoting the rank of their customer (adaptive "enemies")

LRSs and Bibliometrics

- LRSs leverage on the body of literature within bibliometrics, the 80-years-old science of the quantitative analysis of scientific literature
- Bibliometrics studies the quality of scientific papers, journals, etc., in terms of their impact factors (IFs), i.e. a measure of the impact that it has had, obtained through a quantitative analysis of the bibliographic citations to it
- Many results are directly applicable by observing that a hyperlink from page p_i to page p_j can be seen as a bibliographic reference to paper p_j included in the bibliography of paper p_i

Link-based Ranking Systems (LRSs)

- LRSs rank a "base set" BS of Web pages
- Depending on what BS is, we have:
  - Query Dependent LRSs rank a set of Web pages that have previously been identified as being topic-related with the query
    - Based on both topic-relatedness and authoritativeness
    - Must be computed on-line
    - Best known algorithm: HITS[Kleinberg98] (Clever WSE)
  - Query Independent LRSs, in principle, rank the entire Web
    - Only based on authoritativeness
    - Can be computed off-line
    - At query time, it must be merged in some way with a query-dependent ranking based on topic-relatedness
    - Best known algorithm: PageRank[Brin&Page98] (Google WSE)

LRSs

- Preliminary steps to all LRSs are
  1. Identification of BS (necessary for QD LRSs only)
  2. The generation of the hyperlink graph G=<P,E>
- In Step 1, HITS obtains a base set BS of pages (loosely) topic-related to the query in the following way:
  - The query is fed to a standard text search system, and BS is initiated to a 'root set' consisting of the k top-ranked pages
  - All the pages pointing to pages in BS, and all the pages pointed to pages of BS, are added to BS
- Step 2 is obtained by considering all pages in BS as nodes in P, and all hyperlinks between pages of BS as edges in E, after discarding
  - 'nepotistic' hyperlinks (internal to the Web site)
  - 'duplicate' hyperlinks (only one link for any pair $\langle p_i,p_j \rangle$)
  - 'self-loops' (links from p_i to p_i)

Adjacency Matrix

- The input to any LRS is thus a $|BS| \times |BS|$ adjacency matrix W such that

  $W[i,j]=1$ iif there is a hyperlink from page p_i to p_j

- The output of any LRS is a vector $a=[a_1,..,a_{|BS|}]$ where a_i is the authoritativeness of page p_i

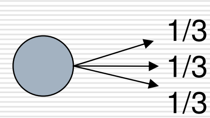
- Backward Neighbors, $B(j)=\{p_i \mid W[i,j]=1\}$
- Forward Neighbors, $F(i)=\{p_j \mid W[i,j]=1\}$

The InDegree Algorithm

- The InDegree algorithm [Marchiori97], consists in identifying the authoritativeness a_i of a page p_i with the in-degree of p_i, i.e. $|B(i)|$
- It corresponds to ranking Web pages according to their 'popularity' ('visibility')
- In matric notation $a = W^T \cdot 1$
- Main weakness: only the quantity of backward links, and not their quality, matters
- It can fooled easily by SWSs. To promote a page p_s, they only need to set up lots of dummy pages $p_1,..p_k$, containing pointers to p_s
- Not used in any current-day WSE

Pagerank scoring

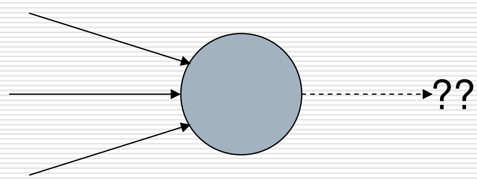
- Imagine a browser doing a random walk on web pages:
  - Start at a random page
  - At each step, go out of the current page along one of the links on that page, equiprobably



- "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

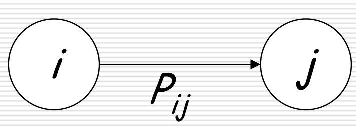
- At a dead end, jump to a random web page.
- At any non-dead end, with probability 10%, jump to a random web page.
  - With remaining probability (90%), go out on a random link.
  - 10% - a parameter.

Result of teleporting

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

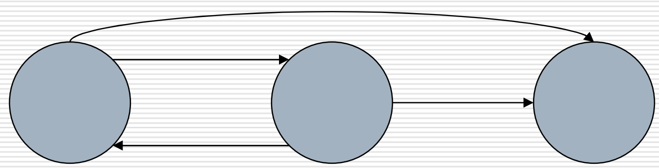
Markov chains

- 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*.



Markov chains

- Clearly, for all i, $\sum_j P_{ij} = 1$
- 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
  - you can be in any state at every time step, with non-zero probability.
- For any ergodic Markov chain, there is a unique long-term visit rate for each state.
  - *Steady-state 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 $\mathbf{x} = (x_1, \ldots x_n)$ tells us where the walk is at any point.
- E.g., (000…1…000) means we're in state *i*.

  1 i n

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

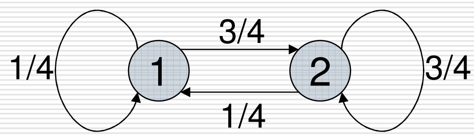
$$\sum_i x_i = 1$$

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 our current position is described by **a**, then the next step is distributed as **aP**.
- Whenever **a** is the steady state, it should be **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.
- Strict convergence is not necessary;
  - [Brin&Page98] reports acceptable convergence on 322M nodes in about 50 iterations

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*.

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!

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

Considerations on PageRank

- The ranking returned by PageRank can be used for doing prioritized crawling
- Without the teleporting factor, PageRank would be un-crackable by spammers

- The (undisclosed) ranking formula used by Google nowadays is a complex recipe (PageRank is the most important ingredient). Other ingredients include:
  - Text in the page
  - Anchor text
  - Query term proximity
  - URL length

HITS (Klimberg98]

- HITS may be seen as a modification of InDegree where a companion notion of the authority value (the hub value) is introduced.

- Authority Value a_i of p_i (how authoritative p_i is, 'seminal papers')
- Hub Value h_i of p_i (how good p_i is helping the user in locating authoritative pages, 'survey papers')

- They are defined in a mutual recursive manner
  - A page is a good hub when it points to many good authoritative pages $h_i = \sum_{j \in F(i)} a_j$
  - A page is a good authority when it is pointed by many good hubs $a_i = \sum_{j \in B(i)} h_j$

Equations

- Recasting equations in a matrix-vector form, we have
  - $h \leftarrow W a$
  - $a \leftarrow W^T h$

- Substituting these into one another, we obtain
  - $h = W W^T h$
  - $a = W^T W a$

- Eigenvectors equations!

Considerations

- The iterative updates, if scaled by an appropriate eigenvalues, are equivalent to the power iteration method for computing the eigenvectors of WW^T and W^TW respectively
- Thus the steady state is determined by the entries in W and hence the structure of the graph

- In computing these eigenvectors entries, we are not restricted to use the power iteration method

Problems

- The problem of HITS is that it is easily spammable: in fact, a spammer wishing to promote a page p_s only needs to set up a page p_t that points to many known authorities and to p_s

A variant: HubAvg

- A problem with HITS is that h_i monotonically grows not only with the authority, but also with the number $|F(i)|$ of the forward neighbors of p_i;
- Thus, the best hub is the one which points to all pages in BS!
- The HubAvg algorithm [Borodin+05] views h_i as the average authority value of the forward neighbors of p_i
  - $h_i = (\sum_{j \in F(i)} a_j)/|F(i)|$
  - $a_i = (\sum_{j \in B(i)} h_j)$

A variant: HubAvg

- It can be seen as a hybrid between HITS and PageRank
  - Authority and hubs to every page
  - Subdivides the hub score of a page amongst its forward neighbors
- Fairly easy to spam, although slightly more difficult than HITS