Ranking linked data
PageRank, topic-specific PageRank and HITS

Web and Media Search

Ranking Web data (part 2)

1. Web page segments
2. Text pre-processing
3. Terms weighting
4. Ranking text data
5. Ranking linked data
   1. Links and anchors
   2. PageRank
   3. Topic-specific PageRank
   4. Hubs and Authorities
5. Ranking linked data

- Links are inserted by humans.
- They are one of the most valuable judgments of a page's importance.
- A link is inserted to denote an association. The anchor text describes the type of association.

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

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

  - Armonk, NY-based computer giant IBM announced today
  - Joe's computer hardware links
    - Compaq
    - HP
    - IBM

  - www.ibm.com

  - Big Blue today announced record profits for the quarter

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
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

Incoming and outgoing links

- The popularity of a page is related to the number of incoming links
  - Positively popular
  - Negatively popular
- The popularity of a page is related to the popularity of pages pointing to them
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).

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?
The random surfer

- The PageRank of a page is the probability that a given random "Web surfer" is currently visiting that page.
- This probability is related to the incoming links and to a certain degree of browsing randomness (e.g. reaching a page through a search engine).

![Diagram of PageRank with nodes A, B, and C and transition probabilities 0.59, 0.32, and 0.40]

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 \( i \).

![Diagram of a Markov chain with states i and j and transition probability P_{ij}]
Transitions probability matrix

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>P_{ab}</td>
<td>P_{ac}</td>
<td>P_{ad}</td>
</tr>
<tr>
<td>B</td>
<td>P_{ba}</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>P_{cb}</td>
<td>0</td>
<td>P_{cd}</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>P_{db}</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

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.

The PageRank of Web page \( i \) corresponds to the probability of being at page \( i \) after an infinite random walk across all pages (i.e., the stationary distribution).

PageRank

- The rank of a page is related to the number of incoming links of that page and the rank of the pages linking to it.

\[
\text{PR}(A) = (1 - d) + d \left( \frac{\text{PR}(B)}{\text{OL}(B)} + \frac{\text{PR}(C)}{\text{OL}(C)} + \ldots \right)
\]
PageRank: Formalization

- The RandomSurfer model assumes that the pages with more inlinks are visited more often.

The rank of a page is computed as:

\[ p_i = (1 - d) + d \sum_{j=1}^{N} \left( \frac{L_{ij}}{c_j} \right) p_j \]

where \( L_{ij} \) is the link matrix, \( c_j \) is the number of links of page, and \( p_j \) is the PageRank of that page.
Example

- Consider three Web pages:

\[ p_i = (1 - d) + d \sum_{j=1}^{N} \left( \frac{L_{ij}}{c_j} \right) p_j \]

- The transition matrix \( \frac{L_{ij}}{c_j} \) is:

\[
\begin{pmatrix}
0 & 0.5 & 0.5 \\
1 & 0 & 0 \\
1 & 0 & 0
\end{pmatrix}
\]

Example: computation

- Reducing the PageRank to its matrix notation:

\[ p_i = (1 - d) + d \sum_{j=1}^{N} \left( \frac{L_{ij}}{c_j} \right) p_j \quad p = (1 - d)e + d \cdot LD^{-1}p \]

- We get \( p = A \cdot p \)

- The computation is reduced to:

\[ p_k \leftarrow A p_{k-1}; \quad p_k \leftarrow \frac{N}{e^T p_k} p_k \]

- Corresponding to the computation of the principal eigenvector whose eigenvalue is 1.
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 ~10% probability, using the following rule:**
  - Selects a category (say, one of the 16 top level 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 - Implementation

- **offline**: Compute pagerank distributions wrt individual categories
  - Query independent model as before
  - Each page has multiple pagerank scores – one for each 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

Non-uniform Teleportation

10% Sports teleportation
10% Health teleportation

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 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*.

The hope

```
<table>
<thead>
<tr>
<th></th>
<th>Hubs</th>
<th>Authorities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>AT&amp;T</td>
<td>Sprint</td>
</tr>
<tr>
<td>Bob</td>
<td>MCI</td>
<td></td>
</tr>
</tbody>
</table>
```

*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 and root 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.
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 $x$:

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

  $$a(x) \leftarrow \sum_{y \rightarrow x} h(y)$$
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 relative orders of the $h()$ and $a()$ scores - not their absolute values.**

- In practice, ~5 iterations get you close to stability.

Summary

1. Web page segments
2. Text pre-processing
3. Terms weighting
4. Ranking text data
5. Ranking linked data
   1. Links and anchors
   2. PageRank
   3. Topic-specific PageRank
   4. HITS: Hubs and Authorities