Introduction to

Information Retrieval

CS276
Information Retrieval and Web Search
Chris Manning, Pandu Nayak
and Prabhakar Raghavan
Link analysis
Today’s lecture – hypertext and links

- We look beyond the content of documents
  - We begin to look at the hyperlinks between them

- Address questions like
  - Do the links represent a conferral of authority to some pages? Is this useful for ranking?
  - How likely is it that a page pointed to by the CERN home page is about high energy physics

- Big application areas
  - The Web
  - Email
  - Social networks
Links are everywhere

- Powerful sources of authenticity and authority
  - Mail spam – which email accounts are spammers?
  - Host quality – which hosts are “bad”?
  - Phone call logs
- The Good, The Bad and The Unknown
Example 1: Good/Bad/Unknown

- The Good, The Bad and The Unknown
  - Good nodes won’t point to Bad nodes
  - All other combinations plausible
Simple iterative logic

- **Good** nodes won’t point to **Bad** nodes
  - If you point to a **Bad** node, you’re **Bad**
  - If a **Good** node points to you, you’re **Good**
Simple iterative logic

- **Good** nodes won’t point to **Bad** nodes
  - If you point to a **Bad** node, you’re **Bad**
  - If a **Good** node points to you, you’re **Good**
Simple iterative logic

- **Good** nodes won’t point to **Bad** nodes
  - If you point to a **Bad** node, you’re **Bad**
  - If a **Good** node points to you, you’re **Good**

Sometimes need probabilistic analogs – e.g., mail spam
Example 2:
In-links to pages – unusual patterns 😊

Spammers violating power laws!
Many other examples of link analysis

- Social networks are a rich source of grouping behavior
- E.g., Shoppers’ affinity – Goel+Goldstein 2010
  - Consumers whose friends spend a lot, spend a lot themselves
Our primary interest in this course

- Link analysis for most IR functionality thus far based purely on text
  - Scoring and ranking
  - Link-based clustering – topical structure from links
  - Links as features in classification – documents that link to one another are likely to be on the same subject

- Crawling
  - Based on the links seen, where do we crawl next?
The Web as a Directed Graph

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

**Hypothesis 2:** The text in the anchor of the hyperlink on page A describes the target page B
Assumption 1: reputed sites

This is the companion website for the following book:

Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval

You can order this book at [CUP](#) or your local bookstore or on the internet. The best search.

The book aims to provide a modern approach to information retrieval from a computer science

University and at the University of Stuttgart.

We'd be pleased to get feedback about how this book works out as a textbook, what is missing

comments to: informationretrieval(at)yahoogroups(dot)com
Assumption 2: annotation of target
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

![Diagram](www.ibm.com)
Indexing anchor text

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

Armonk, NY-based computer giant IBM announced today

www.ibm.com

Joe's computer hardware links
Sun
HP
IBM

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 (more) the anchor text from them
  - Increase the weight of off-site anchors (non-nepotistic scoring)
Anchor Text

- Other applications
  - Weighting/filtering links in the graph
  - Generating page descriptions from anchor text
  - Web document classification
Web document classification

- Suppose we know the classes of some web pages
- How do we use these as features for learning the classes of neighboring pages?
- Basic approach – use known class labels together with in- or out-link info
(Important) Aside:
Connectivity servers

Getting at all that link information.
Inexpensively.
Connectivity Server

- Support for fast queries on the web graph
  - Which URLs point to a given URL?
  - Which URLs does a given URL point to?

Stores mappings in memory from
  - URL to outlinks, URL to inlinks

- Applications
  - Crawl control
  - Web graph analysis
    - Connectivity, crawl optimization
  - Link analysis
Boldi and Vigna 2004

- Webgraph – set of algorithms and a java implementation
- Fundamental goal – maintain node adjacency lists in memory
  - For this, compressing the adjacency lists is the critical component
Adjacency lists

- The set of neighbors of a node
- Assume each URL represented by an integer
- E.g., for a 4 billion page web, need 32 bits per node
- Naively, this demands 64 bits to represent each hyperlink
- Boldi/Vigna get down to an average of ~3 bits/link
  - Further work achieves 2 bits/link
Adjacency list compression

- Properties exploited in compression:
  - Similarity (between lists)
  - Locality (many links from a page go to “nearby” pages)
  - Use gap encodings in sorted lists
  - Distribution of gap values
Main ideas of Boldi/Vigna

- Consider lexicographically ordered list of all URLs, e.g.,
  - www.stanford.edu/alchemy
  - www.stanford.edu/biology
  - www.stanford.edu/biology/plant
  - www.stanford.edu/biology/plant/copyright
  - www.stanford.edu/biology/plant/people
  - www.stanford.edu/chemistry
Each of these URLs has an adjacency list

Main idea: due to templates, the adjacency list of a node is similar to one of the 7 preceding URLs in the lexicographic ordering

Express adjacency list in terms of one of these

E.g., consider these adjacency lists

- 1, 2, 4, 8, 16, 32, 64
- 1, 4, 9, 16, 25, 36, 49, 64
- 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144
- 1, 4, 8, 16, 25, 36, 49, 64

Encode as (-2), remove 9, add 8
Gap encodings

- Given a sorted list of integers $x, y, z, \ldots$, represent by $x, y-x, z-y, \ldots$

- Compress each integer using a code
  - $\gamma$ code - Number of bits $= 1 + 2 \lfloor \lg x \rfloor$
  - $\delta$ code: ...
  - Information theoretic bound: $1 + \lfloor \lg x \rfloor$ bits
  - $\zeta$ code: Works well for integers from a power law Boldi Vigna DCC 2004
Main advantages of BV

- Depends only on locality in a canonical ordering
  - Lexicographic ordering works well for the web
- Adjacency queries can be answered very efficiently
  - To fetch out-neighbors, trace back the chain of prototypes
  - This chain is typically short in practice (since similarity is mostly intra-host)
  - Can also explicitly limit the length of the chain during encoding
- Easy to implement one-pass algorithm
Back to link analysis
Citation Analysis

- Citation frequency
- Bibliographic coupling frequency
  - 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
  - Asked: which journals are authoritative?
The web isn’t scholarly citation

- Millions of participants, each with self interests
- Spamming is widespread
- Once search engines began to use links for ranking (roughly 1998), link spam grew
  - You can join a *link farm* – a group of websites that heavily link to one another
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 long run” 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.
Introduction to Information Retrieval

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 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.

$P_{ij}$
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

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

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 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 \( \mathbf{P} \) tells us where we go next from state \( i \).

- So from \( \mathbf{x} \), our next state is distributed as \( \mathbf{xP} \)
  - The one after that is \( \mathbf{xP}^2 \), then \( \mathbf{xP}^3 \), etc.
  - (Where) Does the converge?
How do we compute this vector?

- Let $\mathbf{a} = (a_1, \ldots, a_n)$ denote the row vector of steady-state probabilities.
- If our current position is described by $\mathbf{a}$, then the next step is distributed as $\mathbf{aP}$.
- But $\mathbf{a}$ is the steady state, so $\mathbf{a} = \mathbf{aP}$.
- Solving this matrix equation gives us $\mathbf{a}$.
  - So $\mathbf{a}$ is the (left) eigenvector for $\mathbf{P}$.
  - (Corresponds to the “principal” eigenvector of $\mathbf{P}$ with the largest eigenvalue.)
  - Transition probability matrices always have largest eigenvalue 1.
Pagerank summary

- Preprocessing:
  - Given graph of links, build matrix $P$.
  - From it compute $a$ – left eigenvector of $P$.
  - 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.
  - But this rank order is query-independent ...
The reality

- Pagerank is used in web search 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 hope

Mobile telecom 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 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.
Visualization

Get in-links (and out-links) from a connectivity server
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 $x$:

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

\[ a(x) \leftarrow \sum_{y \rightarrow x} h(y) \]
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!
  - proof of this comes later.
- In practice, ~5 iterations get you close to stability.
# Japan Elementary Schools

<table>
<thead>
<tr>
<th><strong>Hubs</strong></th>
<th><strong>Authorities</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>schools</td>
<td>The American School in Japan</td>
</tr>
<tr>
<td>LINK Page-13</td>
<td>The Link Page</td>
</tr>
<tr>
<td>“ú¬¡JSwZ</td>
<td>%wè£— § ¨”cŠwZfz[ffy][fW</td>
</tr>
<tr>
<td>ã‰ëŠwZfz[ffy][fW</td>
<td>Kids' Space</td>
</tr>
<tr>
<td>100 Schools Home Pages (English)</td>
<td>“Åô£— § “Åô¼”¬ŠwZ</td>
</tr>
<tr>
<td>K-12 from Japan 10/met and Education</td>
<td>%ëª“ç¬Šw•®¬ŠwZ</td>
</tr>
<tr>
<td><a href="http://www...iglobe.ne.jp/~IKESAN">http://www...iglobe.ne.jp/~IKESAN</a></td>
<td>KEIMEI GAKUEN Home Page (Japanese)</td>
</tr>
<tr>
<td>]„ÔŠU”N.P ’gr”Œé</td>
<td>Shiranuma Home Page</td>
</tr>
<tr>
<td>OŠ—”— § ÖŠ—”Œ=ŠwZ</td>
<td>fuzoku-es.fukui-u.ac.jp</td>
</tr>
<tr>
<td>Koulutus ja oppilaitokset</td>
<td>welcome to Miasa E&amp;J school</td>
</tr>
<tr>
<td>TOYODA HOMEPAGE</td>
<td>_”PIŒ § E%¡hlš— § Ô’¼¬ŠwZ[fy</td>
</tr>
<tr>
<td>Education</td>
<td><a href="http://www...p/~m_maru/index.html">http://www...p/~m_maru/index.html</a></td>
</tr>
<tr>
<td>Cay’s Homepage(Japanese)</td>
<td>fukui haruyama-es HomePage</td>
</tr>
<tr>
<td>–γ”–ŠwZfzf[ffy][fW</td>
<td>Torisu primary school</td>
</tr>
<tr>
<td>UNIVERSITY</td>
<td>goo</td>
</tr>
<tr>
<td>Ï‰£ŠwZ DRAGON97-TOP</td>
<td>Yakumo Elementary,Hokkaido,Japan</td>
</tr>
<tr>
<td>Å‰žŠwZ.T”N.P ’åf[ffy][fW</td>
<td>FUZOKU Home Page</td>
</tr>
<tr>
<td>êµ” é¼Å© Yå¥ÉVå”¼ Yå¥ÉVå¼</td>
<td>Kamishibun Elementary School...</td>
</tr>
</tbody>
</table>
Things to note

- Pulled together good pages regardless of language of page content.
- Use *only* link analysis after base set assembled
  - iterative scoring is query-independent.
- Iterative computation after text index retrieval - significant overhead.
Proof of convergence

- \( n \times n \) 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.

\[
\begin{pmatrix}
1 & 0 & 1 & 0 \\
2 & 1 & 1 & 1 \\
3 & 1 & 0 & 0
\end{pmatrix}
\]
Hub/authority vectors

- View the hub scores $h()$ and the authority scores $a()$ as vectors with $n$ components.
- Recall the iterative updates

\[
\begin{align*}
h(x) &\leftarrow \sum_{x \rightarrow y} a(y) \\
a(x) &\leftarrow \sum_{y \rightarrow x} h(y)
\end{align*}
\]
Rewrite in matrix form

- \( h = Aa \).
- \( a = A^t h \).

Recall \( A^t \) is the transpose of \( A \).

Substituting, \( h = AA^t h \) and \( a = A^t A a \).

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
- The WebGraph framework I: Compression techniques (Boldi et al. 2004)