707.000
Web Science and Web Technology
„Link Analysis and Search“

Markus Strohmaier

Univ. Ass. / Assistant Professor
Knowledge Management Institute
Graz University of Technology, Austria

e-mail: markus.strohmaier@tugraz.at
web: http://www.kmi.tugraz.at/staff/markus
Midterm Course Feedback
Opinions, Criticism, Improvement Suggestions

• Course Contents
  – Prerequisites
  – Scope, Difficulty
  – Speed
  – Relevance / „Interestingness“

• Organisation
  – Mandatory Participation

• Breaks

• Lecture Style
  – Speed
  – Interaction

• Language
  – Comprehensibility

• Home Assignments
  – Effort
  – Degree of Difficulty
  – Comprehensibility

• Website Resources

• Bach. Thesis / Project

Other aspects?
Overview

Agenda

Architecture of search on the web including an overview of
• Crawling, indexing
• Link analysis
• Search Evaluation

Slides based on
• C. Gütl, Information Search and Retrieval, http://www.iicm.tugraz.at/isr/
Web based Retrieval: Challenges

*Working with an enormous amount of data*

10 billion pages a 500kB estimated in 01-2004
- 2 pages / person on the globe

20 times larger than the LoC print collection
- estimated in 2003

Furthermore there is a **Deep Web**
- 550 billion pages estimated in 2004
Web based Retrieval: Challenges

Example for the amount of web pages:
- Searching for ‘Star Trek’ yielded about 11 million of results on Google [Nov 2007]
- Ordinary users investigate 20-30 result list entries.

What web page is the most interesting?
How to store an index (inverted file) with this size?
Web based Retrieval: Challenges

The Web is highly dynamic

Study by Cho & Garcia-Molina (2002):
  - 40% of the web pages changed their dataset within a week
  - 23% of the .com pages changed on daily basis

Study by Fetterly et al. (2003):
  - 35% of the pages changed while the investigations
  - Larger web pages change more often
Web based Retrieval: Challenges

The Web is self-organized

No central authority (for the WWW) or main index

Everyone can add (even edit) pages

Pages disappear on regular basis
  – A US study claimed that in 2 investigated tech. journals 50% of the cited links were inaccessible after four years.

Lots of errors and falsehood, no quality control
Web based Retrieval: Challenges

The Web is hyperlinked

Based on HTML Markup tags and URIs
Pages are interconnected
  - Unidirectional links (in-link, out-link, self-link)
Network structures emerge from the links
  - Link analysis is possible
Common Architecture
History of Crawlers
[Witten 2007]

• World Wide Web Wanderer (1993)
  – Purpose not to index, but to measure its growth
• WebCrawler (1994)
  – First full-text index for entire web pages
• Lycos, Infoseek, Hotbot (1996)
• AskJeeves, Northern Light (1997)
• Others: OpenText, AltaVista
• Yahoo (What’s that acronym?)
  – Two Stanford PhD students

And then came Google (1998)
  – Another two Stanford PhD students (T. Winograd)
  – Who are now allowed to land their private air planes on a NASA airfield close to Mountain View
  

“Yet Another Hierarchical Officious Oracle“

officious
adj. eager to offer unwanted services; meddlesome; interfering; offering much unwanted advice
Crawler

_Crawlers, robots & spiders harvest sites_

Starting with a **root set** of URLs

**Following links**, that are found on the pages

**Applying filters** to the links
- e.g. only .at domains -> Austrian web pages
- e.g. based on link title & position (focused crawling)
Crawlers: Index Update

• Which sites should be updated and when?
• A page content might have changed since last visit
  – last modified dates are possibly inaccurate
• Different strategies are possible:
  – Refresh only portions ...
  – Prefer most popular sites ...

Ethical Questions:
• How much bandwidth is used?
  – Hit counts ...
• What does that mean for the server load?
• Let loose several spiders at once
  – Decrease of crawling time
  – Increase of load
Crawling: Robots.txt

Robots.txt is an option for webmasters to

- restrict crawler access
- point crawlers to interesting URLs
- identify crawlers (via hits on the robots.txt)
- see http://www.robotstxt.org/wc/robots.html

Example

User-agent: *
Disallow: /wp-admin/
Disallow: /netadmin/
Crawler: Google sitemaps

XML schema to identify interesting portions & updates of a web page
Integration into CMS possible

Example:
<url>
  <loc>http://www.semanticmetadata.net/</loc>
  <lastmod>2007-02-06T11:26:06+00:00</lastmod>
  <changefreq>daily</changefreq>
  <priority>1</priority>
</url>

What's a good crawler?
Crawler: Coverage, Freshness and Coherence
[Witten 2007]

Coverage:
• The percentage of pages that a crawler indexes

Freshness:
• The reciprocal of the time that elapses between successive visits to websites

Coherence:
• The overall extent to which the index corresponds to the web itself
Indexing Module

Takes each new uncompressed page
Extracts vital descriptors
  – terms, positions, links
Creates compressed version of page
Stores
  – Page in cache
  – Descriptors in index
Constructing a Full-text Index [Witten 2007]

<table>
<thead>
<tr>
<th>word</th>
<th>position in text</th>
</tr>
</thead>
<tbody>
<tr>
<td>be</td>
<td>2 6 ...</td>
</tr>
<tr>
<td>is</td>
<td>8 ...</td>
</tr>
<tr>
<td>not</td>
<td>4 ...</td>
</tr>
<tr>
<td>or</td>
<td>3 ...</td>
</tr>
<tr>
<td>question</td>
<td>10 ...</td>
</tr>
<tr>
<td>that</td>
<td>7 ...</td>
</tr>
<tr>
<td>the</td>
<td>9 ...</td>
</tr>
<tr>
<td>to</td>
<td>1 5 ...</td>
</tr>
</tbody>
</table>

(a) The beginning of the index.

(b) The text.

Figure 4.3 Making a full-text index.
Indexes

Content Index
Structure Index
Special Purpose Index
  – Document Formats (PDF, Doc, ...)
  – Media (Images, Video, ...)

[Diagram of indexing system with labels for Crawler Module, Indexing Module, Query Module, Repository, and Users]
Indexes

Content Index

- Inverted Document Index
  - term x -> <d11>, <d28>, <d31>, ...
  - term y -> <d10>, <d35>, <d36>, ...
- Index is a
  - quick lookup table
  - smaller than documents

Structure Index

- Hyperlink Information
- In-links, out-links & self-links
- Stored for ...
  - Later analysis
  - Later queries (who links to whom)
Ranking Module

- Orders set of relevant pages
  - Input from query module
- Employs **ranking algorithm**
  - Based on several aspects (terms, links, etc.)
  - Overall score is combination of
    - Content score (TF*IDF)
    - Popularity score (PageRank, HITS, etc.)
Popularity Ranking

- 2 Algorithms developed independently
  - PageRank, Brin & Page
  - Hypertext Induced Topic Search (HITS), Kleinberg

- Basic idea of popularity
  - Someone likes a page
  - Gives a recommendation (on another page)
  - Using a hyperlink
Popularity Ranking: Basic Idea

There are different types of people:

- Regarding their idea of recommendation
  - People giving a lot of recommendations (links)
  - People giving few recommendations (links)
- Regarding their state of recommendation
  - Recommended by a lot of people
  - Recommended by few people

Combinations are possible:

- Having no recommendation, but recommending a lot, ...
Popularity Ranking: Basic Idea

Think of .... people as pages recommendations as links

Therefore:
“Pages are popular, if popular pages link them”

“PageRank is a global ranking of all web pages, regardless of their content, based solely on their location in the Web’s graph structure.” [Page et al 1998]
A Tangled Web [Witten 2007]

**Figure 4.4** A tangled web.
Popularity Ranking: Basic Idea

Additional assumptions:

- **Hubs** are pages that point to highly ranked vertices
- **Authorities** are pages, which are pointed to by highly ranked vertices

*HITS*
PageRank: Original Summation Formula

Original summation formula

- PageRank of page $P_i$ is given by the summation of: all pages $P_j$ that link to $P_i$ (given by the set $B_{P_i}$) divided by the set of outbound links of $P_j$: $|P_j|$

$$r(P_i) = \sum_{P_j \in B_{P_i}} \frac{r(P_j)}{|P_j|},$$

Iterative formula, starting with rank $1/n$ for all n pages:

$$r_{k+1}(P_i) = \sum_{P_j \in B_{P_i}} \frac{r_k(P_j)}{|P_j|}$$
PageRank: Original Summation Formula
[Page et al 1998]

Figure 2: Simplified PageRank Calculation
PageRank: Original Summation Formula
[Amy N. Langville and Carl D. Meyer 2004]

\[ r_{k+1}(P_i) = \sum_{P_j \in B_{P_i}} \frac{r_k(P_j)}{|P_j|} \]

\[ r_1(P_2) = \frac{1}{6}/2 + \frac{1}{6}/3 = \frac{5}{36} \]

<table>
<thead>
<tr>
<th>Iteration 0</th>
<th>Iteration 1</th>
<th>Iteration 2</th>
<th>Rank at Iter. 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r_0(P_1) = 1/6 )</td>
<td>( r_1(P_1) = 1/18 )</td>
<td>( r_2(P_1) = 1/36 )</td>
<td>5</td>
</tr>
<tr>
<td>( r_0(P_2) = 1/6 )</td>
<td>( r_1(P_2) = \text{?} )</td>
<td>( r_2(P_2) = 1/18 )</td>
<td>4</td>
</tr>
<tr>
<td>( r_0(P_3) = 1/6 )</td>
<td>( r_1(P_3) = 1/12 )</td>
<td>( r_2(P_3) = 1/36 )</td>
<td>5</td>
</tr>
<tr>
<td>( r_0(P_4) = 1/6 )</td>
<td>( r_1(P_4) = 1/4 )</td>
<td>( r_2(P_4) = 17/72 )</td>
<td>1</td>
</tr>
<tr>
<td>( r_0(P_5) = 1/6 )</td>
<td>( r_1(P_5) = 5/36 )</td>
<td>( r_2(P_5) = 11/72 )</td>
<td>3</td>
</tr>
<tr>
<td>( r_0(P_6) = 1/6 )</td>
<td>( r_1(P_6) = 1/6 )</td>
<td>( r_2(P_6) = 14/72 )</td>
<td>2</td>
</tr>
</tbody>
</table>
Initial Problems

Rank sinks & cycles:
- Some pages get all of the score, other pages none
- Cycles just flip the rank
- Some nodes do not have outlinks: Dangling nodes

How many iterations?
- Will the process converge?
- Will it converge to one single vector?
Approach of Brin & Page

Notion of the random surfer

- Someone navigates through the web using hyperlinks
- If there are 6 links, there is a probability of 1/6 that s/he takes a specific link
- On dangling nodes (without out links) s/he can jump everywhere with equal chance
- Furthermore s/he can leave the link path with a given probability every time

Dealing with dangling nodes

replace all zero rows, \( 0^T \), with 
\( \frac{1}{n} e^T \), where \( e^T \) is the row vector of all ones and \( n \) is the order of the matrix.
Leaving the link structure:  
[Amy N. Langville and Carl D. Meyer 2004]

Introduction of the Google Matrix:

\[
G = \alpha S + (1 - \alpha) \frac{1}{n} ee^T
\]

Brin and Page suggested a damping factor \( \alpha = 0.85 \)

„That means, roughly five-sixths of the time a web surfer randomly clicks on hyperlinks (i.e. following the structure of the web) while one-sixth of the time this web surfer will go to the URL line and type the address of a new page to „teleport“ to.“

Every node is now directly connected to every other node.
The Google Matrix Step by Step

\[ H = \begin{pmatrix}
  P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \\
  0 & 1/2 & 1/2 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 \\
  1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
  0 & 0 & 0 & 0 & 1/2 & 1/2 \\
  0 & 0 & 0 & 1/2 & 0 & 1/2 \\
  0 & 0 & 0 & 1 & 0 & 0 \\
\end{pmatrix} \]

\[ G = 0.9H + (0.9 + 0.1)(1/6(1 1 1 1 1 1)) \]

\[ 0.9 \times \begin{pmatrix}
  0 & 1/2 & 1/2 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 \\
  1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\
  0 & 0 & 0 & 0 & 1/2 & 1/2 \\
  0 & 0 & 0 & 1/2 & 0 & 1/2 \\
  0 & 0 & 0 & 1 & 0 & 0 \\
\end{pmatrix} = \begin{pmatrix}
  0 & 9/20 & 9/20 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 \\
  3/10 & 3/10 & 0 & 0 & 3/10 & 0 \\
  0 & 0 & 0 & 0 & 9/20 & 9/20 \\
  0 & 0 & 0 & 9/20 & 0 & 9/20 \\
  0 & 0 & 0 & 9/10 & 0 & 0 \\
\end{pmatrix} \]
The Google Matrix
Step by Step

\[ G = 0.9H + 0.1 \left( \begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{array} \right) \times \frac{1}{6} \left( \begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{array} \right) \]
The Google Matrix
Step by Step

\[ G = 0.9H + 0.1 \left( \begin{array}{cccccc}
1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0
\end{array} \right) \times \frac{1}{6} \left( \begin{array}{cccccc}
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1
\end{array} \right) \]
The Google Matrix
Step by Step

The formula for the Google Matrix $G$ is given by:

$$G = 0.9H + (0.9 \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} + 0.1 \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}) \frac{1}{6} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

The matrix $G$ is constructed step by step as follows:

1. The initial step $H$ is a matrix with each row summing to 1.
2. The update step involves multiplying $H$ by $0.9$ and adding it to a matrix with $0.1$ in each row and $0.9$ in all other positions.
3. The result is normalized by dividing by the number of web pages.

The matrix $G$ is:

$$G = \begin{pmatrix}
0 & 9/20 & 9/20 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
3/10 & 3/10 & 0 & 0 & 0 & 3/10 \\
0 & 0 & 0 & 0 & 9/20 & 9/20 \\
0 & 0 & 0 & 9/20 & 0 & 9/20 \\
0 & 0 & 0 & 9/10 & 0 & 0
\end{pmatrix}$$

$$+ \begin{pmatrix}
1/60 & 1/60 & 1/60 & 1/60 & 1/60 & 1/60 \\
1/6 & 1/6 & 1/6 & 1/5 & 1/6 & 1/6 \\
1/60 & 1/60 & 1/60 & 1/60 & 1/60 & 1/60 \\
1/60 & 1/60 & 1/60 & 1/60 & 1/60 & 1/60 \\
1/60 & 1/60 & 1/60 & 1/60 & 1/60 & 1/60 \\
1/60 & 1/60 & 1/60 & 1/60 & 1/60 & 1/60
\end{pmatrix}$$

$$= \begin{pmatrix}
1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\
1/5 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\
19/50 & 19/60 & 1/60 & 1/50 & 19/60 & 1/60 \\
1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\
1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\
1/60 & 1/60 & 1/60 & 1/60 & 11/12 & 1/60
\end{pmatrix}$$
Result of the adaptations
[Amy N. Langville and Carl D. Meyer 2004]

Iterative Formula

\[ \pi^{(k+1)} = \pi^{(k)} G, \]

- Converges to a single PageRank vector

In our example:

\[ \pi^T = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \end{pmatrix} \]

\[ \pi^T = \begin{pmatrix} 0.03721 & 0.05396 & 0.04151 & 0.3751 & 0.206 & 0.2862 \end{pmatrix} \]

taken from “Google’s PageRank & Beyond”, Langville & Meyer
Retrieval Evaluation: Motivation

**Objectively** compare different
- Search engines
- Models & Weighting Schemes
- Methods & Techniques

**Scope**
- Academic
- Commercial & Industrial

**Axis**
- Runtime, Retrieval performance
Retrieval Evaluation

Approaches since first prototypes differ in:

- Test collections
- Experts assessing retrieval performance
- Metrics
  - What’s good? / What’s bad?

Overall problem:

- What is relevant?
Metrics: Precision & Recall

Within a document collection $D$ with a given query $q$

$|R|$ .. num. of relevant docs

$|A|$ .. num. of found docs

$|Ra|$ .. num. found & relevant
Metrics: Precision

\[
\text{Precision} = \frac{|Ra|}{|A|} = \frac{\text{found relevant docs}}{\text{found docs}}
\]

Gives % how many of the actual found documents have been relevant

Between 0 and 1

– Optimum: 1 ... all found docs are relevant
Metrics: Recall

\[
\text{Recall} = \frac{|Ra|}{|R|} = \frac{\text{found relevant docs}}{\text{relevant docs}}
\]

Gives % how many of the actual relevant documents have been found

Between 0 and 1

- Optimum: 1 ... all relevant docs are found
False Positives and False Negatives

[...] False Negatives

False Positives

No False Positives
No False Negatives

What principle ways of reducing FP/FN can you think of?
Examples: Precision & Recall

With a query only 1 document has been found, but this one is relevant (100 would be relevant):
- Precision & Recall?
  - Precision = 1
  - Recall = 0.01

With a query all documents of D have been found (5% of D would be relevant)
- Precision & Recall?
  - Precision = 0.05
  - Recall = 1
Recall vs. Precision Plot

Assumption:
- Result list is sorted by descending relevance
- User investigates result list linearly
  - Precision and Recall change

Approach:
- Map different states to graph
Recall vs. Precision Plot

**Result Set:**
01. d123 *  06. d9 *  11. d38  
02. d84        07. d511  12. d48  
03. d56 *      08. d129  13. d250  
04. d6         09. d187  14. d113  
05. d8         10. d25 * 15. d3 *

**Relevant Results:**
Rq={d3, d5, d9, d25, d39, d44, d56, d71, d89, d123} \(\rightarrow \sum 10\)
Recall vs. Precision Plot

11 Standard Recall Levels
{0%, 10%, 20%, … , 90%, 100%}

Recall = $\frac{|Ra|}{R} = \frac{1}{10}$

Precision = $\frac{|Ra|}{A} = \frac{1}{1}$
Recall and Precision

Recall = $\frac{|Ra|}{R} = \frac{2}{10}$

Precision = $\frac{|Ra|}{A} = \frac{2}{3}$
Recall and Precision

01. d123 * 06. d9 * 11. d38
02. d84 07. d511 12. d48
03. d56 * 08. d129 13. d250
04. d6 09. d187 14. d113
05. d8 10. d25 * 15. d3 *

Precision = ?
Recall = ?

Precision = $\frac{3}{6}$
Recall = $\frac{3}{10}$
Recall and Precision

Precision = ?
Recall = ?

Precision = \frac{4}{10} = 0.4
Recall = \frac{4}{10} = 0.4

Graph showing precision and recall values.

Markus Strohmaier 2008
Confusion Matrix

<table>
<thead>
<tr>
<th>Query</th>
<th>In Query (positiv)</th>
<th>Nicht in Query (negativ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevant</td>
<td>TP</td>
<td>FN</td>
</tr>
<tr>
<td>Nicht Relevant</td>
<td>FP</td>
<td>TN</td>
</tr>
</tbody>
</table>

Precision: \[ P = \frac{TP}{TP + FP} \]

Recall: \[ R = \frac{TP}{TP + FN} \]

Kombination im F-Measure

\[ F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \]

\[ \beta = 1 \Rightarrow F_1 = \frac{2PR}{P + R} \]
Problems

The Deep Web

What is the deep web?
⇒ Pages crawlers do not currently find.

Example: [http://www.aekstmk.or.at/](http://www.aekstmk.or.at/)

Communications of the ACM
Volume 50, Number 5 (2007), Pages 94-101
“Accessing the deep web”, Bin He, Mitesh Patel, Zhen Zhang, Kevin Chen-Chuan Chang
Any questions?

See you next week!