Search

Xin Liu
Searching the Web for Information

• How a Search Engine Works
  – Basic parts:
    1. Crawler: Visits sites on the Internet, discovering Web pages
    2. Indexer: building an index to the Web's content
    3. Query processor: Looks up user-submitted keywords in the index and reports back which Web pages the crawler has found containing those words

• Popular Search Engines: Google, Yahoo!, Bing, Walfram|alpha, etc.
• Text-based
• Knowledge based
• Natural Language

• Automated search
• Human intervention
Source


• How Internet Search Engines Work http://www.howstuffworks.com/search-engine.htm
System Anatomy

Source: The Anatomy of a large-scale Hypertextual web search engine by Sergey Brin and Lawrence Page
Repository

- **Repository**: Contains the actual HTML compressed 3:1 using open-source zlib.
  - Stored like variable-length data (one after the other)
  - Independent of other data structures
  - Other data structures can be restored from here

Repository: 53.5 GB = 147.8 GB uncompressed

```
<table>
<thead>
<tr>
<th>sync</th>
<th>length</th>
<th>compressed packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>sync</td>
<td>length</td>
<td>compressed packet</td>
</tr>
</tbody>
</table>
```

Packet (stored compressed in repository)

```
<table>
<thead>
<tr>
<th>docid</th>
<th>ecode</th>
<th>url</th>
<th>len</th>
<th>pagelen</th>
<th>url</th>
<th>page</th>
</tr>
</thead>
</table>
```
Index

• **Document Index:** Indexed sequential file (ISAM: Indexed sequential file access mode) with status information about each document.
  – The information stored in each entry includes the current document status, a pointer to the repository, a document checksum and various statistics.

• **Forward Index:** Sorting by docID. Stores docIDs pointing to hits. Stored in partially sorted indexes called “barrels”.

• **Inverted Index:** Stores wordIDs and references to documents containing them.
Lexicon

- **Lexicon:** Or list of words, is kept on 256MB of main memory, allocating 14 million words and hash pointers.
- Lexicon can fit in memory for a reasonable price.
- It is a list of words and a hash table of pointers.
Hit lists

• **Hit Lists**: Records occurrences of a word in a document plus details.
  – Position, font, and capitalization information

• Accounts for most of the space used.
  – Fancy hits: URL, title, anchor text, <meta>
  – Plain hits: Everything else
  – Details are contained in bitmaps:

<table>
<thead>
<tr>
<th>Hit: 2 bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>plain</strong>: cap:1</td>
</tr>
<tr>
<td><strong>fancy</strong>: cap:1</td>
</tr>
<tr>
<td><strong>anchor</strong>: cap:1</td>
</tr>
</tbody>
</table>
Crawlers

• When a crawler visits a website:
  – First identifies all the links to other Web pages on that page
  – Checks its records to see if it has visited those pages recently
  – If not, adds them to list of pages to be crawled
  – Records in an index the keywords used on a page
  – Different pages need to be visited in different frequency
Crawler

• How to find/update all pages?
• How fast to update?
  – News, blogs, regional news
  – Wiki
  – UCD homepage
  – Personal homepage
• More documents
  – Images, videos, maps
  – Offline
Building the index

• Like the index listings in the back of a book:
  – for every word, the system must keep a list of the URLs that word appears in.

• Relevance ranking:
  • Location is important: e.g., in titles, subtitles, the body, meta tags, or in anchor text
  • Some ignore insignificant words (e.g., a, an), some try to be complete
  • Meta tag is important.
Indexing the Web

• **Parser:** Must be validated to expect and deal with a huge number of special situations.
  – Typos in HTML tags, Kilobytes of zeros, non-ASCII characters.

• **Indexing into barrels:** Word > wordID > update lexicon > hit lists > barrels.

• **Sorting:** Inverted index is generated from forward barrels
Query Processor

• Ranking the results
  – PageRank: query-independent
  – Relevance: query-dependent
    • font, position and capitalization are used.

• Give users what they want, not what they type
  – Spelling correction
  – Delivering localized results globally

• All done in 0.25s!
DIGITAL CULTURE

YouTube Founders Have the Last Laugh

Morning Edition, October 13, 2006 - Everyone seems to have something to say about Google buying YouTube, in fact, the video-sharing site’s founders posted their own video about the purchase after it was announced.

E-mail this Page
Everyone seems to have something to say about Google buying YouTube. In fact, the video-sharing site's founders posted a description on their personal blog, saying they were 'very well aware' of the acquisition and 'very happy' about it. They called it a 'great move' for the company.

NPR's Al Letson talks to the founders about what they're doing now and how the deal could change the internet.

http://www.youtube.com/watch?v=wmQ0t2x08Dk

Note: NPR is not affiliated with YouTube or Google in any way.

Copyright 2006 NPR. All rights reserved.

Query Processors

• Gets keywords from user and looks them up in its index

• Even if a page has not yet been crawled, it might be reported because it is linked to a page that has been crawled, and the keywords appear in the anchor on the crawled page
Page ranking

• PageRank is a weighted voting system:
  – a link from page A to page B as a vote, by page A, for page B
  – Weighted by the importance of the voting pages.

• You can see the page rank in the browser.
  – Try: cnn.com, yahoo.com,
  – Compare: ucdavis.edu, cs.ucdavis.edu
Page Ranking

• Google's idea: PageRank
  – Orders links by relevance to user
  – Relevance is computed by counting the links to a page (the more pages link to a page, the more relevant that page must be)
    • Each page that links to another page is considered a "vote" for that page
    • Google also considers whether the "voting page" is itself highly ranked
Backlink

- Link Structure of the Web
- Every page has some number of forward links and back links.
- Highly back linked pages are more important than with few links.
- Page rank is an approximation of importance / quality
PageRank

- Pages with lots of backlinks are important.
- Backlinks coming from important pages convey more importance to a page.
- Ex: B, C, D→A
  \[ PR(A) = PR(B) + PR(C) + PR(D) \]
- If B→A, B→C
  \[
  PR(A) = \frac{PR(B)}{2} + \frac{PR(C)}{1} + \frac{PR(D)}{3}.
  \]

\[
PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)}
\]
Rank Sink

- Page cycles pointed by some incoming link

- Problem: this loop will accumulate rank but never distribute any rank outside

- Dangling nodes
Damping factor

\[ PR(A) = \frac{1 - d}{N} + d \left( \frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \cdots \right). \]

\[ R = \begin{bmatrix} (1 - d)/N \\ (1 - d)/N \\ \vdots \\ (1 - d)/N \end{bmatrix} + d \begin{bmatrix} \ell(p_1, p_1) & \ell(p_1, p_2) & \cdots & \ell(p_1, p_N) \\ \ell(p_2, p_1) & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ell(p_i, p_j) & \vdots \\ \ell(p_N, p_1) & \cdots & \ell(p_N, p_N) \end{bmatrix} \]

\[ \sum_{i=1}^{N} \ell(p_i, p_j) = 1 \]
Random Surfer Model

- Page Rank corresponds to the probability distribution of a random walk on the web graphs

- \((1-d)\) can be re-phrased as the random surfer gets bored periodically and jumps to a different page and not kept in a loop forever.
How to solve it?

• Iteration in practice.
• $D=0.85$ in the original paper, a tradeoff between convergence and the impact of random server
Convergence

- Calculate through iterations
- Does it converge?
- Is it unique?
- How fast is the convergence, if it happens?
Search

• How a Search Engine Works
  – Basic parts:
    1. Crawler
    2. Indexer
    3. Query processor

• Which part is the most challenging?
Challenges

• Personalized search
• Search engine optimization
  – Is PageRank out of date?
• Google bomb
• Data fusion
• Scalability and reliability
• Parallelization
• Security and privacy
• Complexity of network and materials
Look forward

• Organize the world's information and make it universally accessible and useful (Google).

• Compute information/knowledge (Walfram)

• Mobile search
• Voice search
Practical search tips

• Choosing the right terms and knowing how the search engine will use them

• Words or phrases?
  – Search engines generally consider each word separately
  – Ask for an exact phrase by placing quotations marks around it
  – Proximity used in deciding relevance
Logical Operators

• AND, OR, NOT
  – AND: Tells search engine to return only pages containing both terms
    Thai AND restaurants
  – OR: Tell search engine to find pages containing either word, including pages where they both appear
  – NOT: Excludes pages with the given word

• AND and OR are *infix operators*; they go between the terms

• NOT is a *prefix operator*; it precedes the term to be excluded
Google search tips

• Case does not matter
  – Washington = WASHINGtOn
• Automatic “and”
  – Davis and restaurant = Davis restaurant
• Exclusion of common words
  – Where/how/and, etc. does not count
  – Enforce by + sign; e.g., star war episode +1
  – Enforce by phrase search; e.g., “star war episode I”
• Phrase search
  – “star war episode I”
• Exclude
  – Bass ~music
• OR
  – Restaurant Davis Thai OR Chinese
• Domain
  – ECS15 site:ucdavis.edu
• Synonym search
  – ~food ~facts
• Numrange search
  – DVD player $50..$100
• Links:
What is your favorite tip?