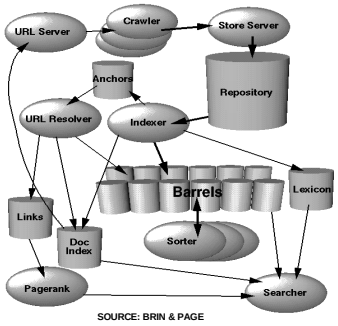
Search Engine Architecture

- **Spider**
  - Crawls the web to find pages. Follows hyperlinks. Never stops
- **Indexer**
  - Produces data structures for fast searching of all words in the pages
- **Retriever**
  - Query interface
  - Database lookup to find hits
    - 300 million documents
    - 300 GB RAM, terabytes of disk
  - Ranking

Google Search Engine Architecture



SOURCE: BRIN & PAGE

URL Server- Provides URLs to be fetched
Crawler is distributed
Store Server - compresses and stores pages for indexing
Repository - holds pages for indexing (full HTML of every page)
Indexer - parses documents, records words, positions, font size, and capitalization
Lexicon - list of unique words found
Barrels hold results of distribution sort
Anchors - keep information about links found in web pages
URL Resolver - converts relative URLs to absolute
Sorter - generates Doc Index
Doc Index - inverted index of all words in all documents (except stop words)
Links - stores info about links to each page (used for Pagerank)
Pagerank - computes a rank for each page retrieved
Searcher - answers queries

Crawlers (Spiders, Bots)

- **Retrieve web pages for indexing by search engines**
- **Start with an initial page P_0. Find URLs on P_0 and add them to a queue**
- **When done with P_0, pass it to an indexing program, get a page P_1 from the queue and repeat**
- **Can be specialized (e.g. only look for email addresses)**
- **Issues**
  - Which page to look at next? (Special subjects, recency)
  - Avoid overloading a site
  - How deep within a site to go (drill-down)?
  - How frequently to visit pages?

Spiders

- **243 active spiders registered 1/01**
  - http://info.webcrawler.com/mak/projects/robots/active/html/index.html
- **Inktomi Slurp**
  - Standard search engine
- **Digimark**
  - Downloads just images, looking for watermarks
- **Adrelevance**
  - Looking for Ads.

Robot Exclusion

- **You may not want certain pages indexed.**
- **Some crawlers conform to the Robot Exclusion Protocol. Compliance is voluntary.**
- **They look for file robots.txt at highest directory level in domain. If domain is www.ecom.cmu.edu, robots.txt goes in www.ecom.cmu.edu/robots.txt**
- **A specific document can be shielded from a crawler by adding the line: <META NAME="ROBOTS" CONTENT="NOINDEX">**

Robots Exclusion Protocol

- **Format of robots.txt**
  - Two fields. User-agent to specify a robot
  - Disallow to tell the agent what to ignore
- **To exclude all robots from a server:**

```
User-agent: *
Disallow: /
```

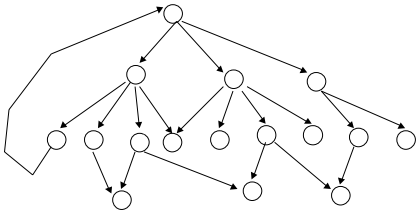
- **To exclude one robot from two directories:**

```
User-agent: WebCrawler
Disallow: /news/
Disallow: /tmp/
```

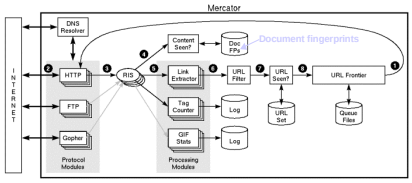
- **View the robots.txt specification at** http://info.webcrawler.com/mak/projects/robots/norobots.html

Web Crawling Strategy

- **Starting location(s)**
- **Traversal order**
  - Depth first
  - Breadth first
  - Or ???
- **Cycles?**
- **Coverage?**



Mercator Spider Structure



1. Remove URL from queue
2. Simulate network protocols & REP
3. Read in RewindInputStream (RIS) mode
4. Has this doc been seen before? (Uses checksums and fingerprints)
5. Extract links
6. Should we download new URL?
7. Has new URL been seen before?
8. Add URL to frontier

URL Frontier (priority queue)

- **Most crawlers do breadth-first search from seeds.**
- **Politeness constraint: don't hammer servers!**
  - Obvious implementation: "live host table"
  - Will it fit in memory?
  - Is this efficient?
- **Mercator's politeness:**
  - One FIFO subqueue per thread.
  - Choose subqueue by hashing host's name.
  - Dequeue first URL whose host does NOT have an outstanding request.

Fetch Pages Module

- **Need to support ftp, gopher, http.**
- **Need to fetch multiple pages at once.**
- **Need to cache as much as possible (DNS, robot exclusion rules).**
- **Need to be defensive!**
  - Need to time out http connections.
  - Watch for "crawler traps" (e.g., infinite URL names.)
  - See section 5 of Mercator paper.
  - Use URL filter module

Duplicate Detection

- **URL-seen test: has this URL been seen before? (to save space, store a "hash")**
- **Content-seen test: same doc, different URL.**
  - Supress link extraction from mirrored pages.
- **What to save for each doc?**
  - 64 bit "document fingerprint"
  - Minimize number of disk reads upon retrieval.

Synch vs. Asynch I/O

- **Problem: due to network/host latency, want to GET multiple URLs at once.**
- **Google: single-threaded crawler + asynchronous I/O.**
- **Mercrator: multi-threaded crawler + synchronous I/O. (easier to code?)**
- **Lucene:**

Cache, Cache, Cache

- **Read 600 URLs off URL frontier on disk.**
- **Cache robot exclusion rules.**
- **Cache document locally for re-processing.**
- **Cache DNS results**

- **Checkpointing: write snapshots to disk!**

Crawling Strategies

- **Priority queue instead of FIFO.**
  - How to determine priority?
  - Google: PageRank.
    - How many links point to this page?
    - What is the "rank" of pages that point to this page?
  - Location (e.g., does end w/ .edu? Does it have 'home' in it? Is the page on a 'good' site?)

- **Focused Crawling: find pages relevant to a particular topic.**
  - Intuition: focused crawlers will be more efficient, provide faster updates, and more relevant results.

Focused Crawling

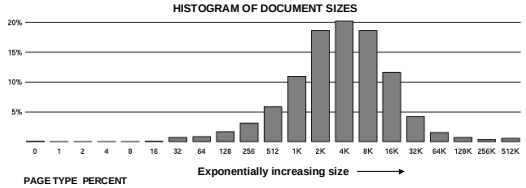
- **Classifier: is crawled page P relevant to topic?**
  - Algorithm that maps page to relevant/irrelevant.

- **Distiller: is crawled page P likely to lead to relevant pages?**
  - Algorithm that maps page to likely/unlikely.
- **Distiller determines priority of following links off of P!**

Conclusions

- **Writing a trivial spider is, well, trivial.**
- **Challenge is writing a spider that is efficient and stable.**
- **Google has shown that pageranking works.**
- **Focused crawling is a "hot" direction.**
- **Project requires you to apply techniques learned to mp3 crawling.**

Mercator Statistics



PAGE TYPE	PERCENT
text/html	69.2%
image/gif	17.9%
image/jpeg	8.1%
text/plain	1.5
pdf	0.9%
audio	0.4%
zip	0.4%
postscript	0.3%
other	1.4%

Crawling the MM Web

- **Crawling the MM Web is tricky.**
  - Most Web pages do not contain links to streaming media
  - Efficient MM crawling heuristics differ greatly from Web crawling heuristics
- **Unique discovery rate > 200K /day**

How big is the Multimedia Web?

- **Currently singingfish.com has largest collection of streaming media URLs: 6+Million URLs**
- **75% of MM URLs can be identified by file extension**
  - Dropping as sites move to dynamic pages produced by MM content management systems.