

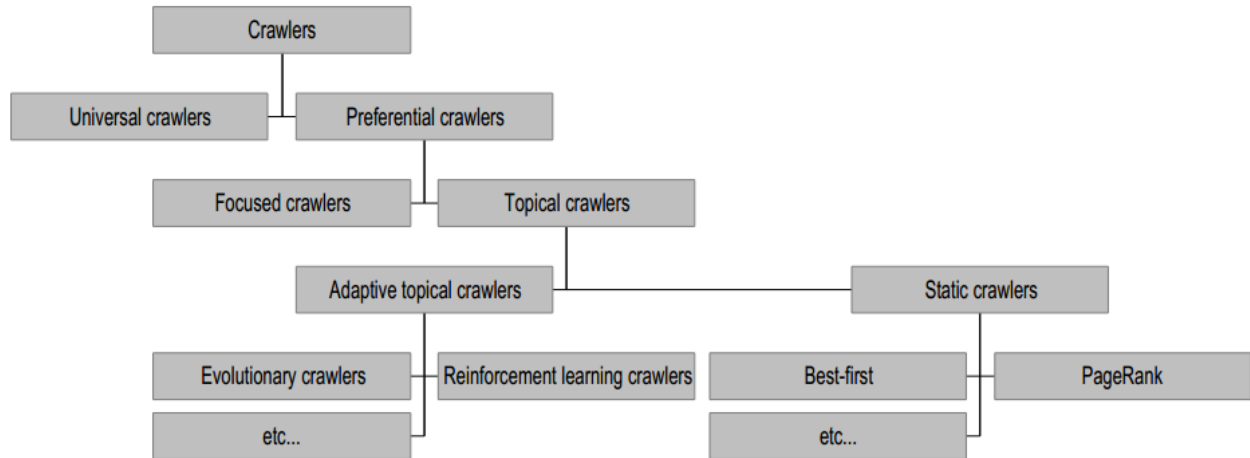
# WEB CRAWLER

## **Many names**

- Crawler
- Spider
- Robot (or bot)
- Web agent
- Wanderer, worm, ...
- And famous instances: googlebot, scooter, slurp, msnbot, ...

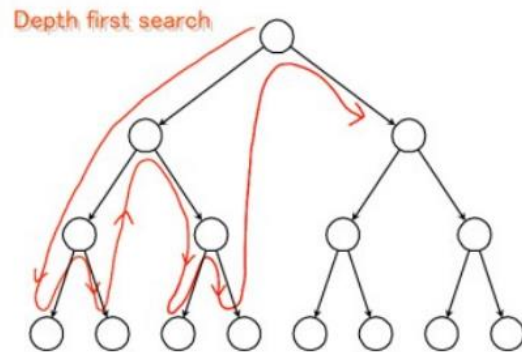
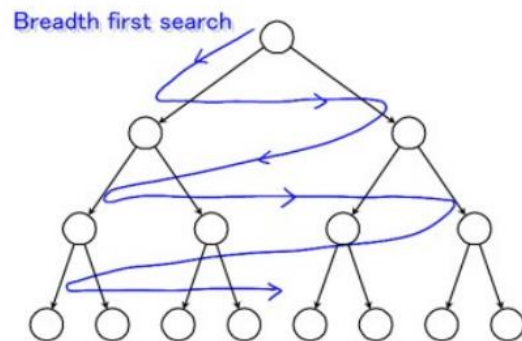
## **Motivation for crawlers**

- Support universal search engines (Google, Yahoo, MSN/Windows Live, Ask, etc.)
- Vertical (specialized) search engines, e.g. news, shopping, papers, recipes, reviews, etc.
- Business intelligence: keep track of potential competitors, partners
- Monitor Web sites of interest
- Evil: harvest emails for spamming, phishing...



## Graph traversal (BFS or DFS?)

- Breadth First Search
  - Implemented with QUEUE (FIFO)
  - Finds pages along shortest paths
  - If we start with “good” pages, this keeps us close; maybe other good stuff...
- Depth First Search
  - Implemented with STACK (LIFO)
  - Wander away (“lost in cyberspace”)



## **Implementation issues**

- Don't want to fetch same page twice!
  - Keep lookup table (hash) of visited pages
  - What if not visited but in frontier already?
- The frontier grows very fast!
  - May need to prioritize for large crawls
- Fetcher must be robust!
  - Don't crash if download fails
  - Timeout mechanism
- Determine file type to skip unwanted files
  - Can try using extensions, but not reliable
  - Can issue 'HEAD' HTTP commands to get Content-Type (MIME) headers, but overhead of extra Internet requests

## **More implementation issues**

- Fetching
  - Get only the first 10-100 KB per page
  - Take care to detect and break redirection loops
  - Soft fail for timeout, server not responding, file not found, and other errors

## **More implementation issues: Parsing**

- HTML has the structure of a DOM (Document Object Model) tree
- Unfortunately actual HTML is often incorrect in a strict syntactic sense
- Crawlers, like browsers, must be robust/forgiving
- Fortunately there are tools that can help

– E.g. tidy.sourceforge.net

- Must pay attention to HTML entities and unicode in text

- What to do with a growing number of other formats?

– Flash, SVG, RSS, AJAX...

### **More implementation issues**

- Stop words

– Noise words that do not carry meaning should be eliminated (“stopped”) before they are indexed

– E.g. in English: AND, THE, A, AT, OR, ON, FOR, etc...

– Typically syntactic markers

– Typically the most common terms

– Typically kept in a negative dictionary

- 10–1,000 elements

- E.g. [http://ir.dcs.gla.ac.uk/resources/linguistic\\_utils/stop\\_words](http://ir.dcs.gla.ac.uk/resources/linguistic_utils/stop_words)

– Parser can detect these right away and disregard them

More implementation issues

Conflation and thesauri

- Idea: improve recall by merging words with same meaning

3. We want to ignore superficial morphological features, thus merge semantically similar tokens

– {student, study, studying, studious} => studi

4. We can also conflate synonyms into a single form using a thesaurus

– 30-50% smaller index

– Doing this in both pages and queries allows to retrieve pages about ‘automobile’ when user asks for ‘car’

- Thesaurus can be implemented as a hash table

### **More implementation issues**

- Stemming
  - Morphological conflation based on rewrite rules
  - Language dependent!
  - Porter stemmer very popular for English
- <http://www.tartarus.org/~martin/PorterStemmer/>
- Context-sensitive grammar rules, eg:
  - “IES” except (“EIES” or “AIES”) --> “Y”
- Versions in Perl, C, Java, Python, C#, Ruby, PHP, etc.
- Porter has also developed Snowball, a language to create stemming algorithms in any language
- <http://snowball.tartarus.org/>
- Ex. Perl modules: `Lingua::Stem` and `Lingua::Stem::Snowball`

### **More implementation issues**

- Static vs. dynamic pages
  - Is it worth trying to eliminate dynamic pages and only index static pages?
  - Examples:
    - <http://www.census.gov/cgi-bin/gazetteer>
    - <http://informatics.indiana.edu/research/colloquia.asp>
    - <http://www.amazon.com/exec/obidos/subst/home/home.html/002-8332429-6490452>
    - <http://www.imdb.com/Name?Menczer,+Erico>
    - <http://www.imdb.com/name/nm0578801/>
  - Why or why not? How can we tell if a page is dynamic? What about ‘spider traps’?
  - What do Google and other search engines do?

### **More implementation issues**

- Relative vs. Absolute URLs

- Crawler must translate relative URLs into absolute URLs
- Need to obtain Base URL from HTTP header, or HTML Meta tag, or else current page path by default

– **Examples**

- Base: <http://www.cnn.com/linkto/>
- Relative URL: intl.html
- Absolute URL: <http://www.cnn.com/linkto/intl.html>
- Relative URL: /US/
- Absolute URL: <http://www.cnn.com/US/>

More implementation issues

- URL canonicalization
- All of these:
  - <http://www.cnn.com/TECH>
  - <http://WWW.CNN.COM/TECH/>
  - <http://www.cnn.com:80/TECH/>
  - <http://www.cnn.com/bogus/../TECH/>
- Are really equivalent to this canonical form:
  - <http://www.cnn.com/TECH/>
- In order to avoid duplication, the crawler must transform all URLs into canonical form
- Definition of “canonical” is arbitrary, e.g.:
  - Could always include port
  - Or only include port when not default :80

More on Canonical URLs

- Some transformation are trivial, for example:

❏ <http://informatics.indiana.edu>

❏ <http://informatics.indiana.edu/>

❏ <http://informatics.indiana.edu/index.html#fragment>

❏ <http://informatics.indiana.edu/index.html>

❏ <http://informatics.indiana.edu/dir1/../../dir2/>

❏ <http://informatics.indiana.edu/dir2/>

❏ <http://informatics.indiana.edu/%7Efil/>

❏ <http://informatics.indiana.edu/~fil/>

❏ <http://INFORMATICS.INDIANA.EDU/fil/>

❏ <http://informatics.indiana.edu/fil/>

#### More on Canonical URLs

Other transformations require heuristic assumption about the intentions of the author or configuration of the Web server:

- Removing default file name

❏ <http://informatics.indiana.edu/fil/index.html>

❏ <http://informatics.indiana.edu/fil/>

– This is reasonable in general but would be wrong in this case because the default happens to be ‘default.asp’ instead of ‘index.html’

- Trailing directory

❏ <http://informatics.indiana.edu/fil>

❏ <http://informatics.indiana.edu/fil/>

– This is correct in this case but how can we be sure in general that there isn’t a file named ‘fil’ in the root dir?

#### More implementation issues

- Spider traps

– Misleading sites: indefinite number of pages

dynamically generated by CGI scripts

– Paths of arbitrary depth created using soft directory

links and path rewriting features in HTTP server

– Only heuristic defensive measures:

- Check URL length; assume spider trap above some threshold, for example 128 characters

- Watch for sites with very large number of URLs
- Eliminate URLs with non-textual data types
- May disable crawling of dynamic pages, if can detect

More implementation issues

- Page repository
  - Naïve: store each page as a separate file
  - Can map URL to unique filename using a hashing function, e.g. MD5
  - This generates a huge number of files, which is inefficient from the storage perspective
  - Better: combine many pages into a single large file, using some XML markup to separate and identify them
- Must map URL to {filename, page\_id}
- Database options
  - Any RDBMS -- large overhead
  - Light-weight, embedded databases such as Berkeley DB