



Web Crawling

Nguyen Ngoc Thao
nnthao@fit.hcmus.edu.vn

Content outline

- An introduction to Web crawlers
- Types of crawlers
- Crawler: Ethics and Conflicts

Web crawlers: A definition

- **Web crawlers** (a.k.a. *spiders* or *robots*) are programs that automatically download Web pages.
 - They browse the Web by following hyperlinks to virtually move from one page to the next.
- Why do we need crawlers?
 - The Web is evolving at rapid rates.
 - Applications need to stay current as pages and links are added, deleted, moved or modified.

*Spiders are the only web developers
in the world that enjoy finding bugs.*



Web crawlers: Applications



- Crawlers collect pages for search engines to build their indexes.



- Business intelligence: organizations collect information about their competitors and potential collaborators



- Web sites and pages of interest are monitored for site maintenance



- Spammers harvest email addresses or collect personal information for phishing or other identity theft attacks.

Web crawlers: A basic algorithm

- A crawler starts from a set of **seed pages** (URLs) and then uses the links within them to fetch other pages.
- The links in these pages are, in turn, extracted and the corresponding pages are visited.
- The process repeats until a sufficient number of pages are visited or some other objective is achieved.

Web crawlers: A basic algorithm

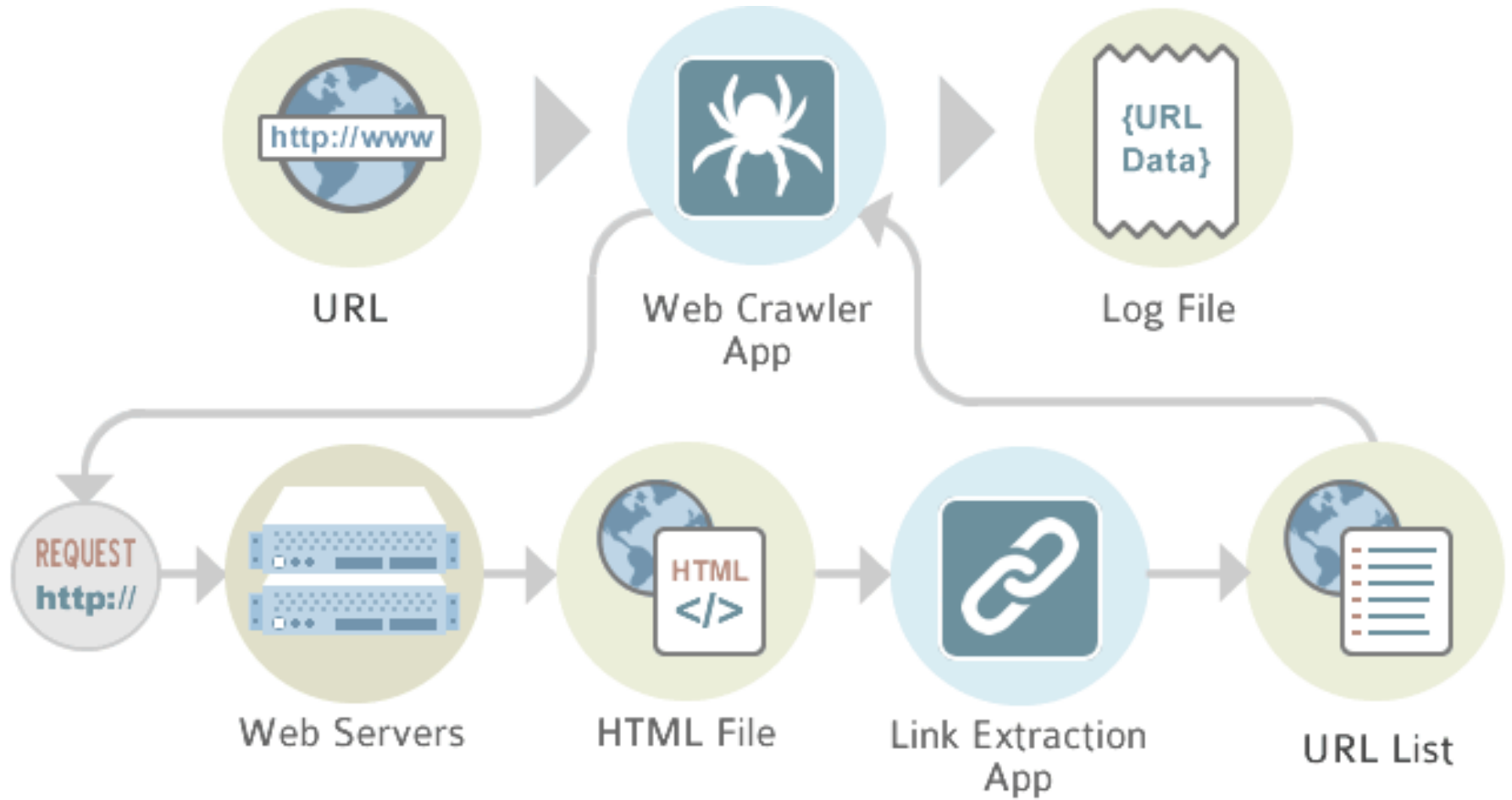
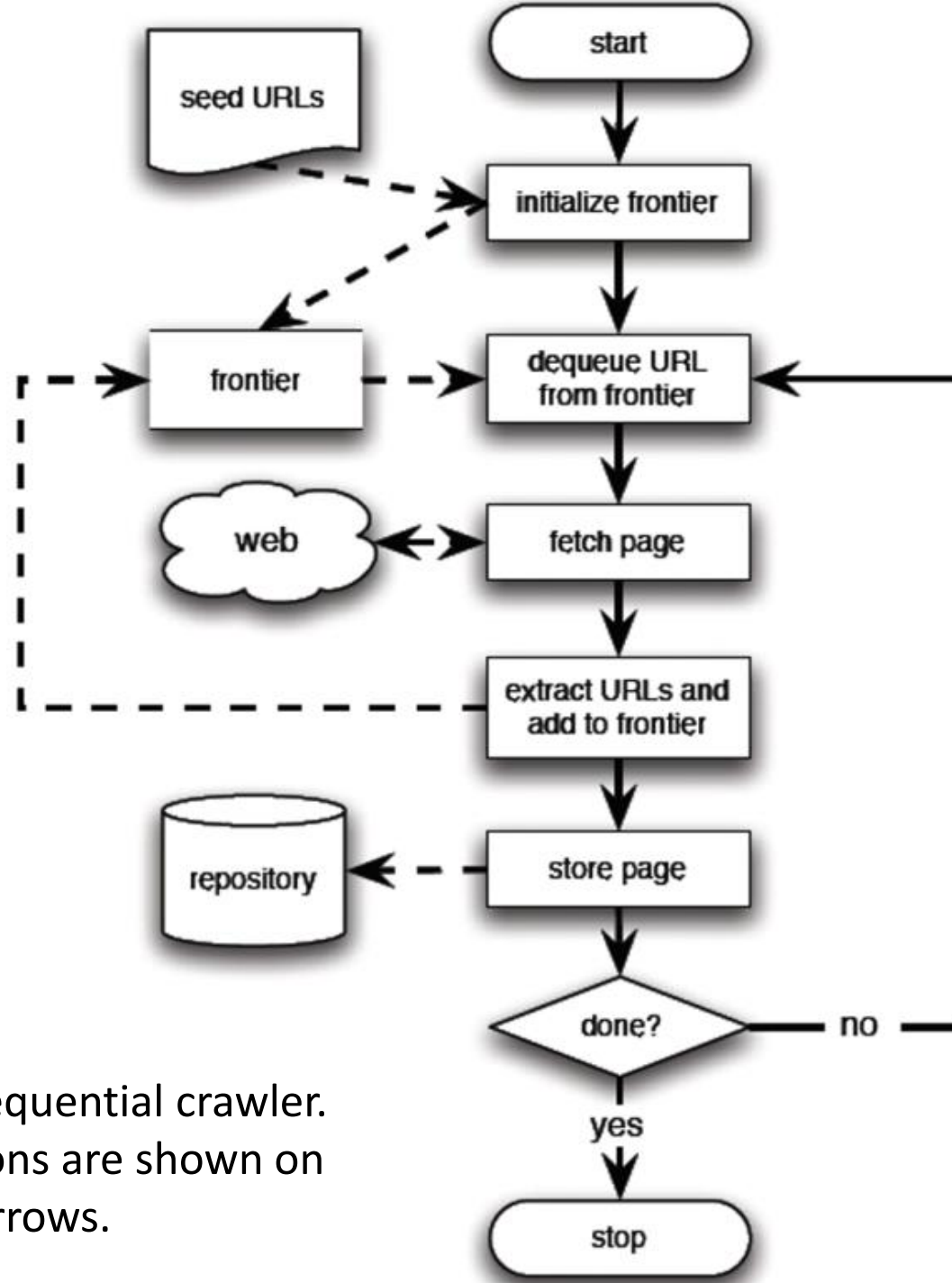


Image credit: SEOPressor Connect



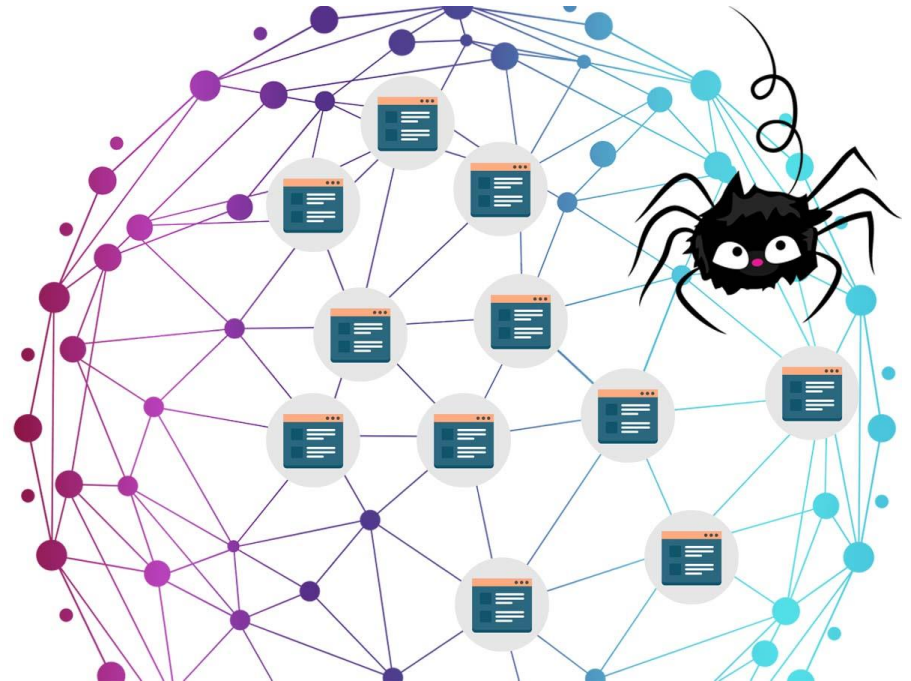
Flow chart of a basic sequential crawler.
The main data operations are shown on
the left, with dashed arrows.

Web crawlers: A basic algorithm

- The Web can be seen as a large graph with pages as its nodes and hyperlinks as its edges.
- A crawler is essentially a graph search algorithm.

A crawler starts from a few of the nodes (seeds) and then follows the edges to reach other nodes.

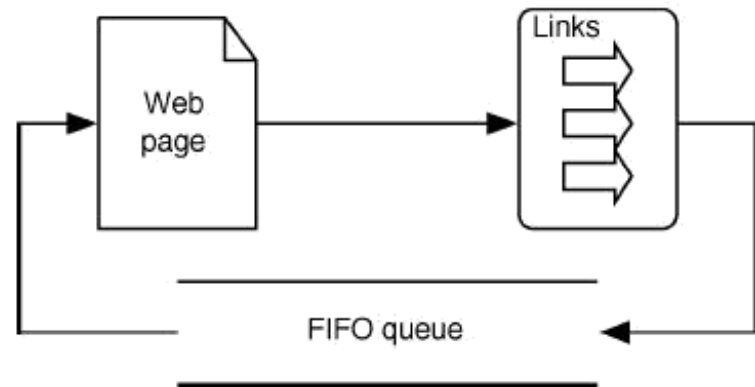
The process of fetching a page and extracting the links within it is analogous to expanding a node in graph search



Web crawlers: Frontier

- The frontier is the main data structure, which contains the URLs of unvisited pages.
 - Typical crawlers store the frontier in the main memory for efficiency.
- The crawler design must determine two factors:
 - Which URLs have low priority and thus get discarded when the frontier is filled up.
 - The order in which new URLs are extracted from the frontier.

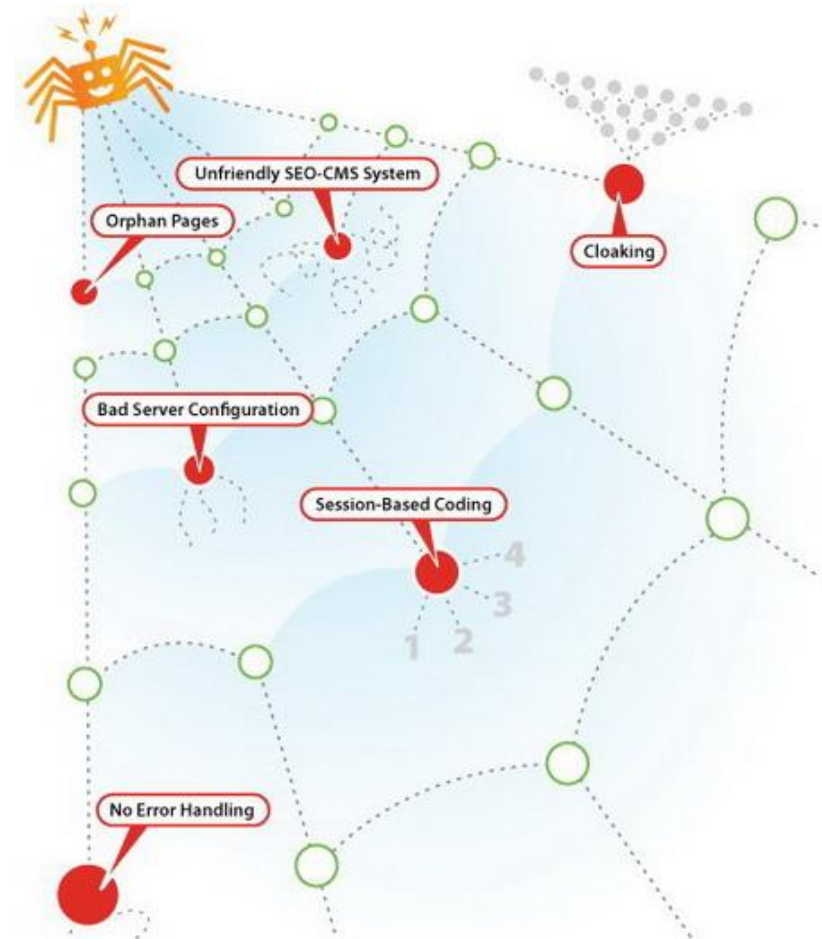
The frontier of a breadth-first crawler may be implemented as a first-in-first-out (FIFO) queue.



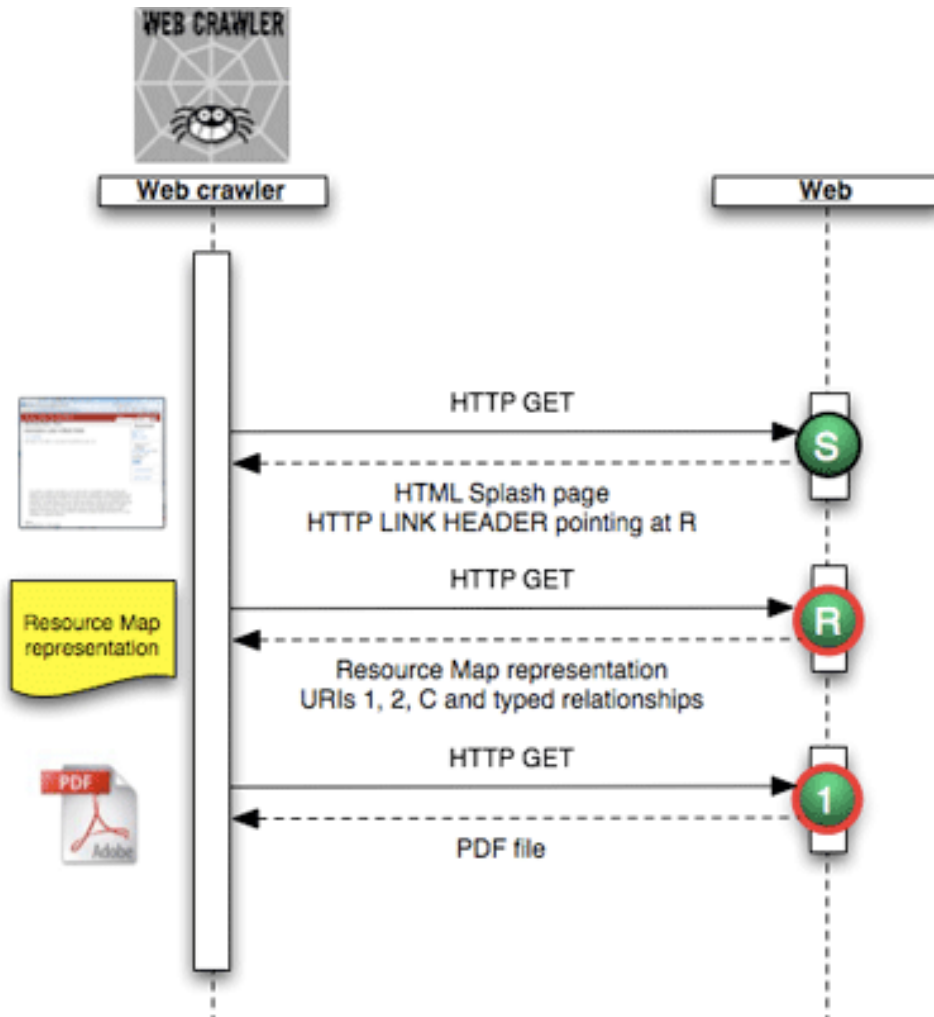
Web crawlers: Crawl history

- The crawl history is a time-stamped list of URLs fetched by the crawler tracking its path through the Web.
- It may be stored on disks for later analysis and evaluation.
 - E.g., to see if the most relevant or important resources are found early in the crawl process
- It may be an in-memory data structure for fast look-up, to avoid revisiting pages or wasting space in the frontier.

Implementation issues



Fetching pages



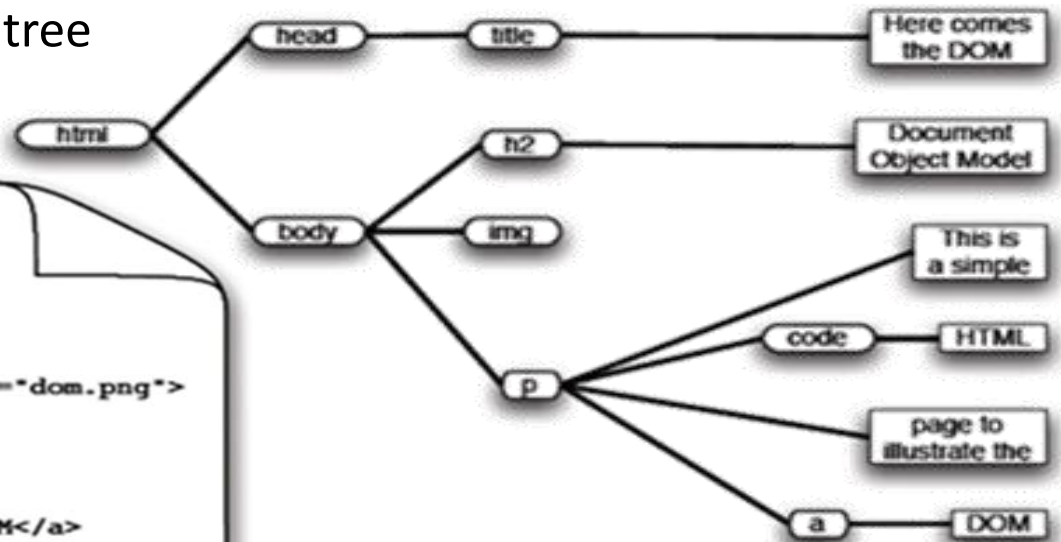
- A crawler acts as a Web client.
- It sends an HTTP request to the server hosting the page and reads the response.
- The client needs a timeout to prevent waiting for responses from slow servers or reading huge pages.

Parsing the content

- The crawler extracts information to facilitate the indexing in a search engine or the crawling process.
- Parsing may imply simple URL extraction from hyperlinks, or more involved analysis of the HTML code.

Illustration of the DOM (or tag) tree built from a simple HTML page.

```
<html>
<head>
  <title>Here comes the DOM</title>
</head>
<body>
  <h2>Document Object Model</h2>
  
  <p>
    This is a simple
    <code>HTML</code>
    page to illustrate the
    <a href="http://www.w3.org/DOM/">DOM</a>
  </p>
</body>
</html>
```



Parsing the content

- HTML standards are laxly enforced by common browsers.
 - Missing required tags, tags improperly nested, missing close tags, misspelled or missing attribute names and values
 - Reserved characters for tag syntax, e.g., double quotes and "
- Robust crawlers often apply some tools, such as [tidy](#), to clean up the HTML content prior to parsing.
- A growing portion of Web pages are written in formats other than HTML, which is good for human interaction but not the crawlers.
 - E.g., HTML5, CSS3, JavaScript, and SVG for web graphic animation

Stop word removal and stemming

- Stop words are very common terms but with little meaning.
 - A [list](#) of articles, preposition and pronouns, e.g., an, from and they
- Stemming conflates morphological variants of terms into common roots (stems).



- Both stop-word removal and stemming help increase the discrimination of pages, and thus facilitating topical crawlers.

Link extraction and Canonicalization

- We extract URLs from anchor (<a>) tags and href attributes.

HTML Code	Visit W3Schools.com!
Display	Visit W3Schools.com!

- Filtering may be necessary to exclude certain file types that are not to be crawled.

Send an HTTP HEAD request
and inspect the content-type
response header

protocol **status code**

HTTP/1.1 200 OK

```
Transfer-Encoding: chunked
Date: Sat, 28 Nov 2009 04:36:25 GMT
Server: LiteSpeed
Connection: close
X-Powered-By: W3 Total Cache/0.8
Pragma: public
Expires: Sat, 28 Nov 2009 05:36:25 GMT
Etag: "pub1259380237;gz"
Cache-Control: max-age=3600, public
Content-Type: text/html; charset=UTF-8
Last-Modified: Sat, 28 Nov 2009 03:50:37 GMT
X-Pingback: http://net.tutsplus.com/xmlrpc.php
Content-Encoding: gzip
Vary: Accept-Encoding, Cookie, User-Agent
```

HTTP headers as Name: Value

Link extraction and Canonicalization

- A URL must be converted into its canonical form before being added to the frontier.

Description and transformation	Example and canonical form
Default port number Remove	http://cs.indiana.edu:80/ http://cs.indiana.edu/
Root directory Add trailing slash	http://cs.indiana.edu http://cs.indiana.edu/
Guessed directory* Add trailing slash	http://cs.indiana.edu/People http://cs.indiana.edu/People/
Fragment Remove	http://cs.indiana.edu/faq.html#3 http://cs.indiana.edu/faq.html
Current or parent directory Resolve path	http://cs.indiana.edu/a/../../b/ http://cs.indiana.edu/b/
Default filename* Remove	http://cs.indiana.edu/index.html http://cs.indiana.edu/
Needlessly encoded characters Decode	http://cs.indiana.edu/%7Efil/ http://cs.indiana.edu/~fil/
Disallowed characters Encode	http://cs.indiana.edu/My File.htm http://cs.indiana.edu/My%20File.htm
Mixed/upper-case host names Lower-case	http://CS.INDIANA.EDU/People/ http://cs.indiana.edu/People/

Spider trap

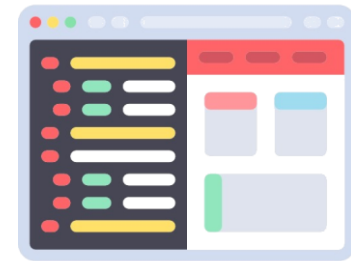
- The URLs of dynamically created links are modified based on the sequence of actions taken by the browsing user (or crawler).
 - E.g., <http://foo.com/A/B/A/...>, the two products, A and B, have dynamic pages pointer to each other.



Page A



Page B



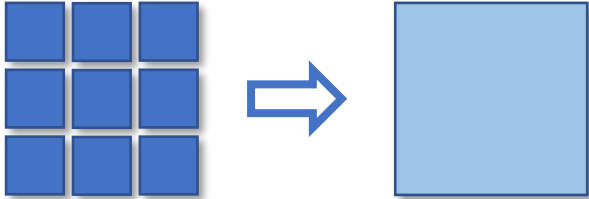






Same page A, but
with a different URL

Spider trap

- A crawler could go inside the spider trap forever without fetching any new content.
- The server may create an entry in a database every time the user clicks on certain dynamic links.
 - The database may be filled to capacity → the site will be disabled.
 - This is a type of **DoS attack** carried out unwittingly by the crawler.
- **Heuristic approach:** Limit the URL sizes (e.g., 256 characters) or the number of pages requested from a given domain.

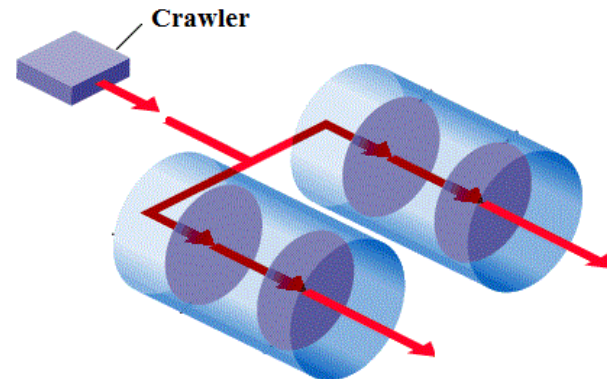
Page repository

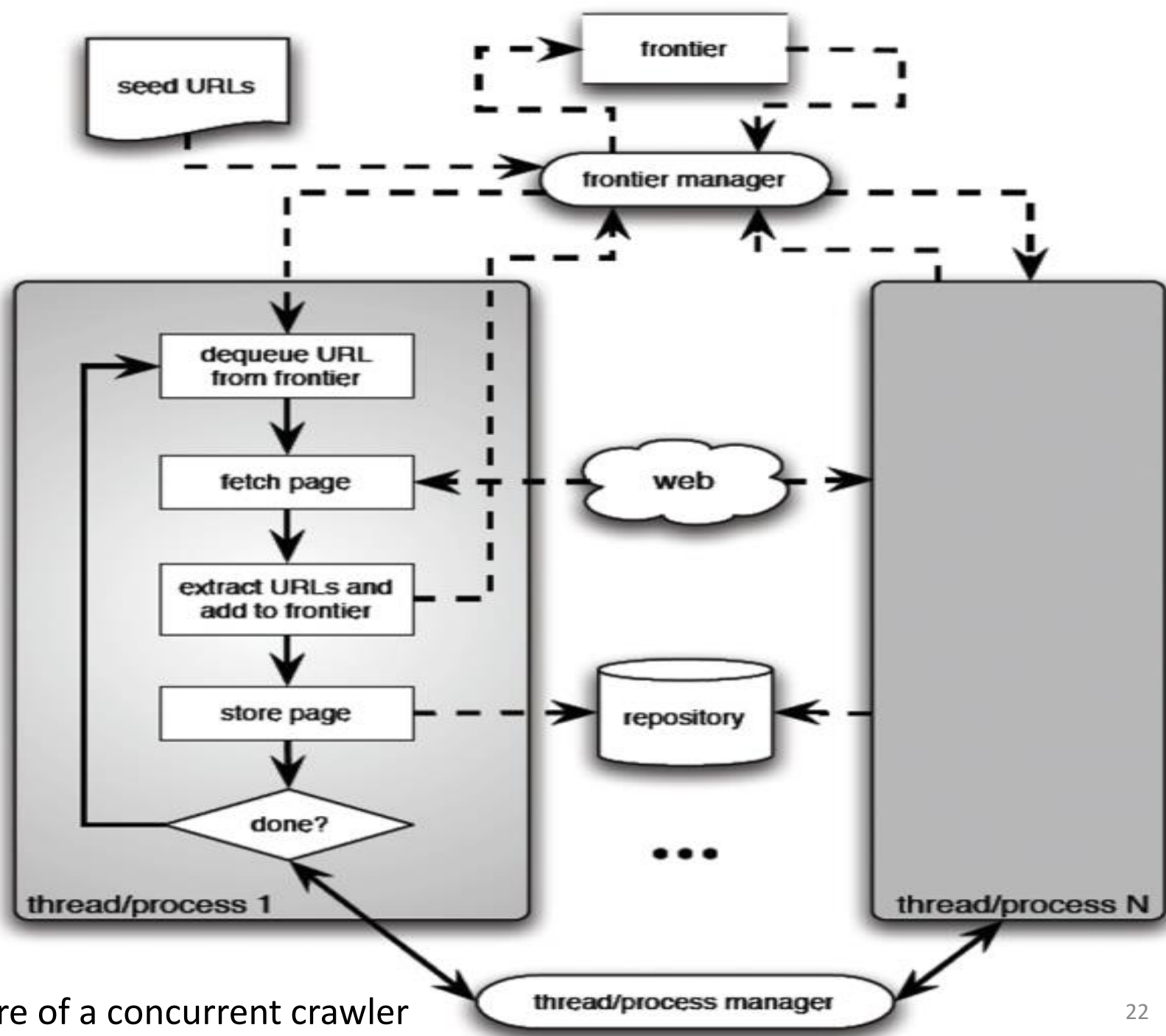
- A page repository may store the crawled pages as separate files
- Large scale crawlers may involve significant time and disk space overhead to manage a very large number of small individual files.

Combine multiple small files into a larger file	Use a database to store and index the pages
	<div> mongoDB</div> <div> <i>cassandra</i></div> <div> Elasticsearch</div> <div> amazon DynamoDB</div> <div> APACHE HBASE</div> <div> redis</div>

Concurrency

- A crawler spends three main resources: network, CPU, and disk.
 - Each is a bottleneck with limits imposed by bandwidth, CPU speed, and disk seek/transfer times.
- The most straightforward way to speed-up a crawler is through concurrent processes or threads.
 - The concurrent design can speed-up a crawler by a factor of 5 or 10 → still insufficient for a commercial search engine.





Architecture of a concurrent crawler

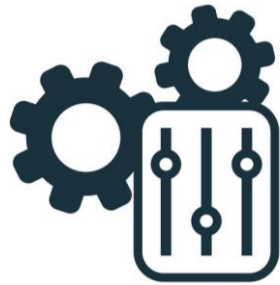
Types of crawlers

Crawling strategies

- Crawlers vary according to **how they choose which URLs to process next**.
- **Breadth-first crawlers** consider the URLs to crawl following their order of entering the frontier.
- **Preferential crawlers** assign each unvisited link a priority based on an estimate of the value of the linked page.
 - Topological properties (e.g., the indegree of the target page), content properties (e.g., the similarity between a user query and the source page), or any other combination of measurable features

Universal crawlers

- Universal crawlers differ from the concurrent breadth-first crawlers along two major dimensions:



Performance

Fetch and process hundreds of thousands of pages per second

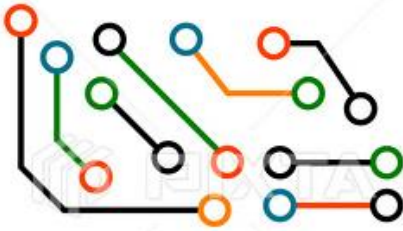


Policy

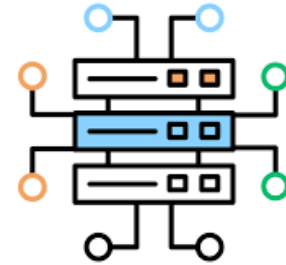
Cover important pages while maintaining their freshness

Universal crawlers: Scalability

- There are issues in meeting those two requirements.



Asynchronous socket



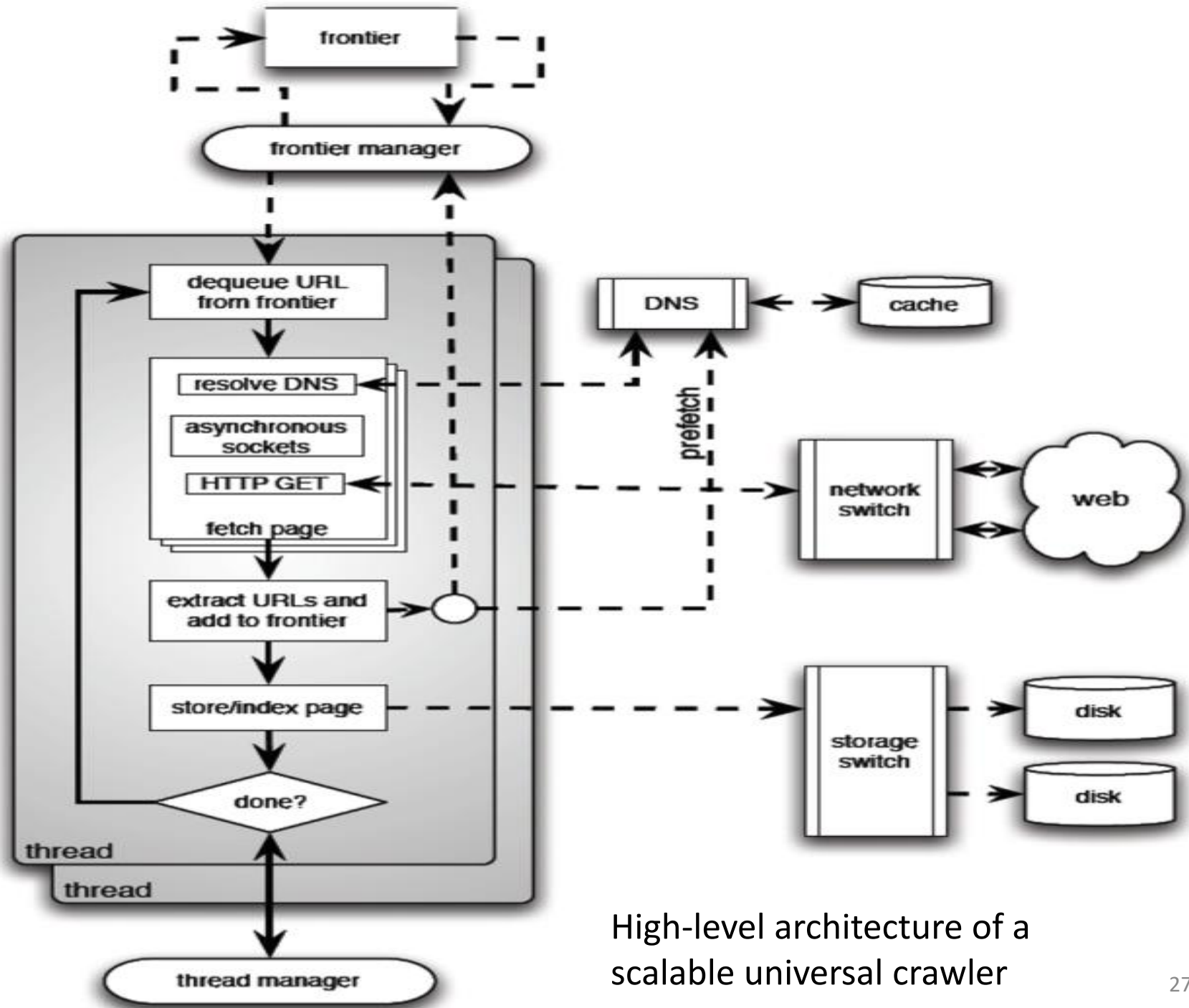
Frontier manager



Connections to DNS servers



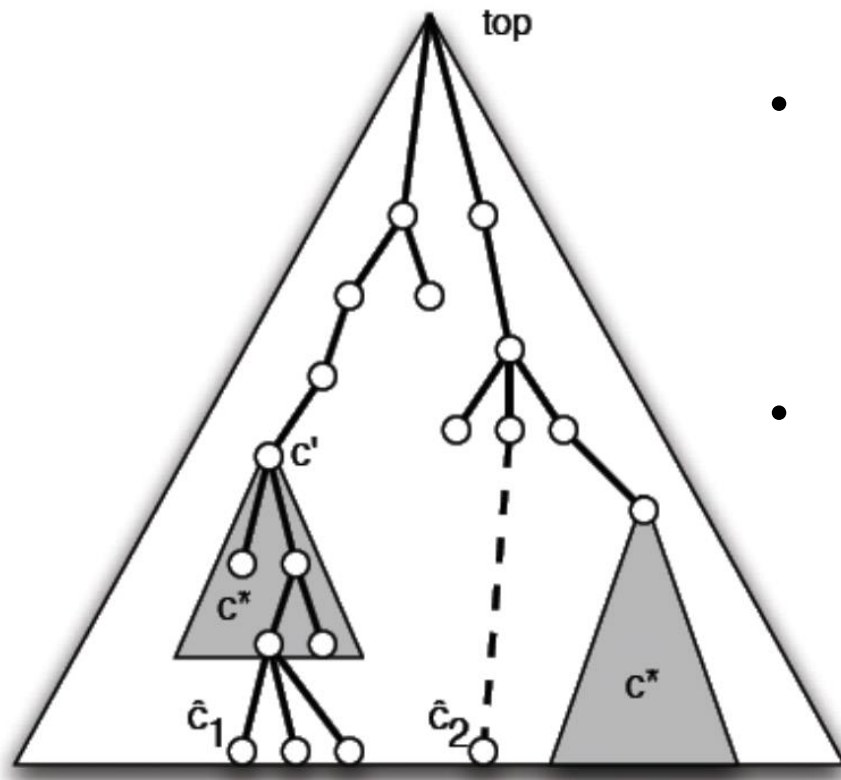
Multiple network resources



High-level architecture of a scalable universal crawler

Focused crawlers

- Focused crawlers attempt to bias the crawler towards pages in certain categories in which the user is interested.



- **“Soft” focused strategy:** prioritize the crawled page with highest score

$$R(p) = \sum_{c \in C^*} \Pr(c|p)$$

- **“Hard” focused strategy:** accept links from a page classified in the leaf category \hat{c}_1 , while discarding the links from a page in \hat{c}_2

A taxonomy for a focused crawler. The areas in gray denote the categories of interest c^* .

Topical crawlers

- Sometimes, labeled (positive and negative) pages may be insufficient to build classifiers guiding crawling.
- Instead, there is a small set of seed pages and a description of a topic of user interest.
 - The topic can consist of one or more example pages (possibly the seeds) or even a short query.
- Topic crawlers preferentially explore the Web by comparing features from visited pages with cues in the target topic.

Scrapers vs. Crawlers

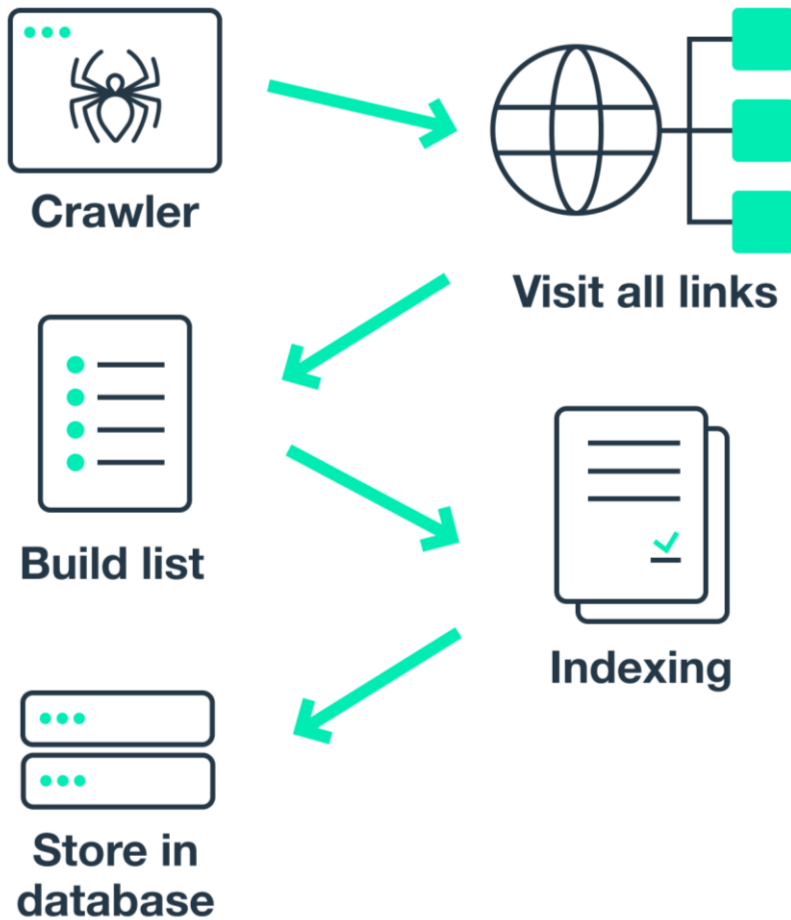


- Web scraping also locates the target data from web pages, yet it knows the exact data element needed to be extracted.

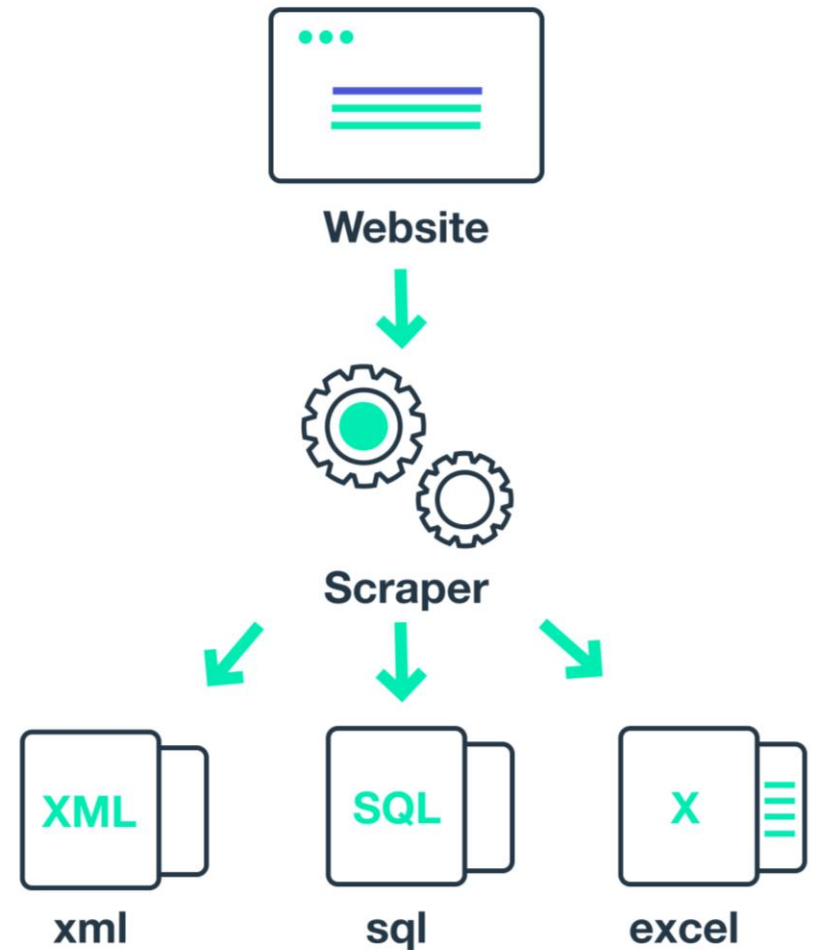
Web scraping	Web crawling
Extract data from downloaded web pages	Refer to download pages from the web
Done at any scale	Mostly done at a large scale
Deduplication is not necessarily	Deduplication is an essential part

- Both scraping and crawling go hand in hand in the whole process of data gathering.

Web Crawler



Web Scrapping





Crawler: Ethics and Conflicts



Web server overload

- Crawlers may put a significant strain on the resources of web servers, mainly on their network bandwidth.
- A crawler must control its requests so that any server does not receive requests at more than some maximum rate.
 - A concurrent crawler handles the workload by the frontier manager.
 - This practice also helps crawler avoid spider traps.



Crawler identity acknowledgement

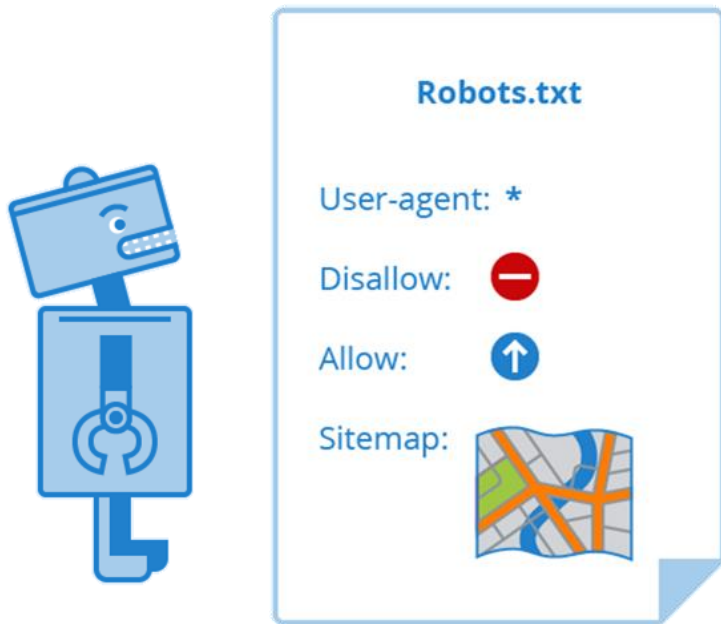
- A crawler must disclose its nature using the User-Agent HTTP header.
 - Name, version number, email contact, and a pointer to where Web administrators may find information about the crawler

```
Host: www.██████████.com
Accept-Language: en-US
Cache-Control: no-cache
Connection: keep-alive
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
From: googlebot(at)googlebot.com
User-Agent: Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)
Accept-Encoding: gzip,deflate,br
```

More information at <https://www.searchdatalogy.com/blog/googlebots-http-headers/>

Robot Exclusion Protocol (REP)

- Web server administrators use REP to communicate which files may not be accessed by a crawler.



An optional file, **robots.txt**, in the root directory of the Web server states access policies for crawlers.

Crawlers must parse this file before sending requests to the server.

- Compliance with the protocol is an ethical issue *only*.

REP: robot.txt file

- Crawlers are identified by their User-agent field.
- Web authors can indicate if a page may (not) be indexed, cached, or mined by a crawler using special meta-tags.

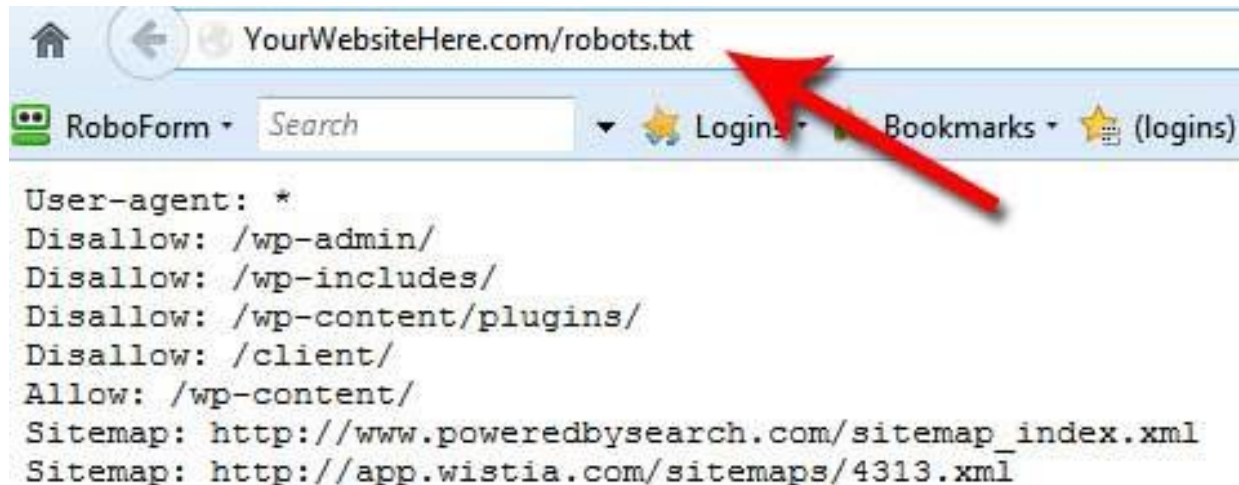


Image credit: wakeupcoders.medium.com/

References

- Bing Liu. 2007. Web Data Mining-Exploring Hyperlinks, Contents, and Usage Data. Springer Series on Data-Centric Systems and Applications. Chapter 8.