Chapter IR:IX

IX. Acquisition

- Crawling the Web
- Conversion
- Storing Documents

Crawling the Web

Web Technology

- Internet
- World Wide Web
- Addressing
- HTTP
- HTML
- Web Graph

Crawling the Web

Web Technology: Internet

Definition 1 (Internetwork, Internet [Wikipedia]**)**

The Internet is a global system of interconnected computer networks. A computer network connects computers (hosts) via a specific technology to allow data exchange.

Crawling the Web

Web Technology: Internet

Definition 1 (Internetwork, Internet [Wikipedia]**)**

The Internet is a global system of interconnected computer networks. A computer network connects computers (hosts) via a specific technology to allow data exchange.

1958 Founding of the Defense Advanced Research Project Agency DARPA in reaction to the Soviet Union taking the lead in the space race.

1969 First version of the ARPANET works with 4 hosts. The goal is to decentralize military networks, rendering them resilient to attacks.

1973 35 hosts connected, including trans-atlantic ones in England and Norway.

1989 More than 150,000 hosts. ARPANET is shut down. The NSFNET is now called INTERNET.

1991 Dawn of the World Wide Web.

1992 More than 1 million hosts. The Internet Society is founded.

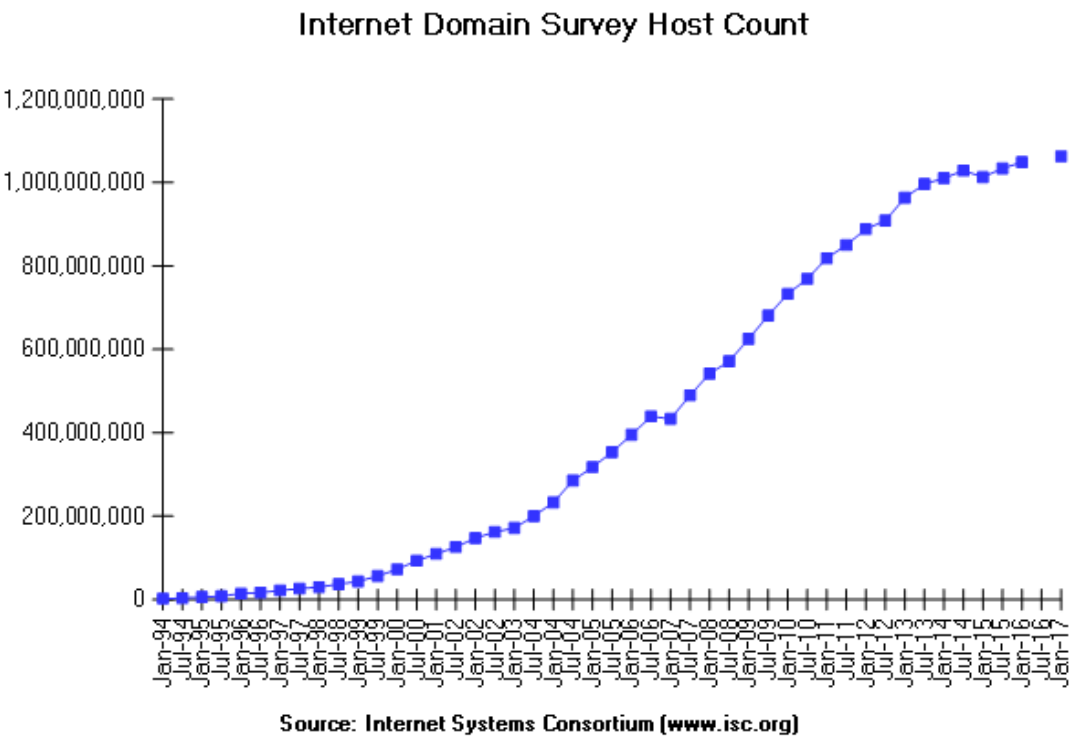
2013 More than 1 billion hosts.

Crawling the Web

Web Technology: Internet

Definition 1 (Internetwork, Internet [Wikipedia])

The Internet is a global system of interconnected computer networks. A computer network connects computers (hosts) via a specific technology to allow data exchange.



[Internet Systems Consortium, www.isc.org]

Crawling the Web

Web Technology: World Wide Web

Definition 2 (World Wide Web, WWW [Wikipedia])

The World Wide Web is a network of documents, located by Uniform Resource Locators (URLs), connected by hypertext links, and accessible via the Internet.

Crawling the Web

Web Technology: World Wide Web

Definition 2 (World Wide Web, WWW [Wikipedia]**)**

The World Wide Web is a network of documents, located by Uniform Resource Locators (URLs), connected by hypertext links, and accessible via the Internet.

1945 Vannevar Bush envisions Memex, the first hypertext-like system.

1963 Ted Nelson coins the term hypertext. In the following decades, many hypertext systems are proposed and developed.

1990 Tim Berners-Lee develops the first web client, invents HTML, and implements the first web server, calling his system "WorldWideWeb". CERN's phone book is the first application.

1993 CERN agrees to allow anyone to use Web protocol and code royalty-free.

1994 623 web servers. The World Wide Web Consortium W3C is founded.

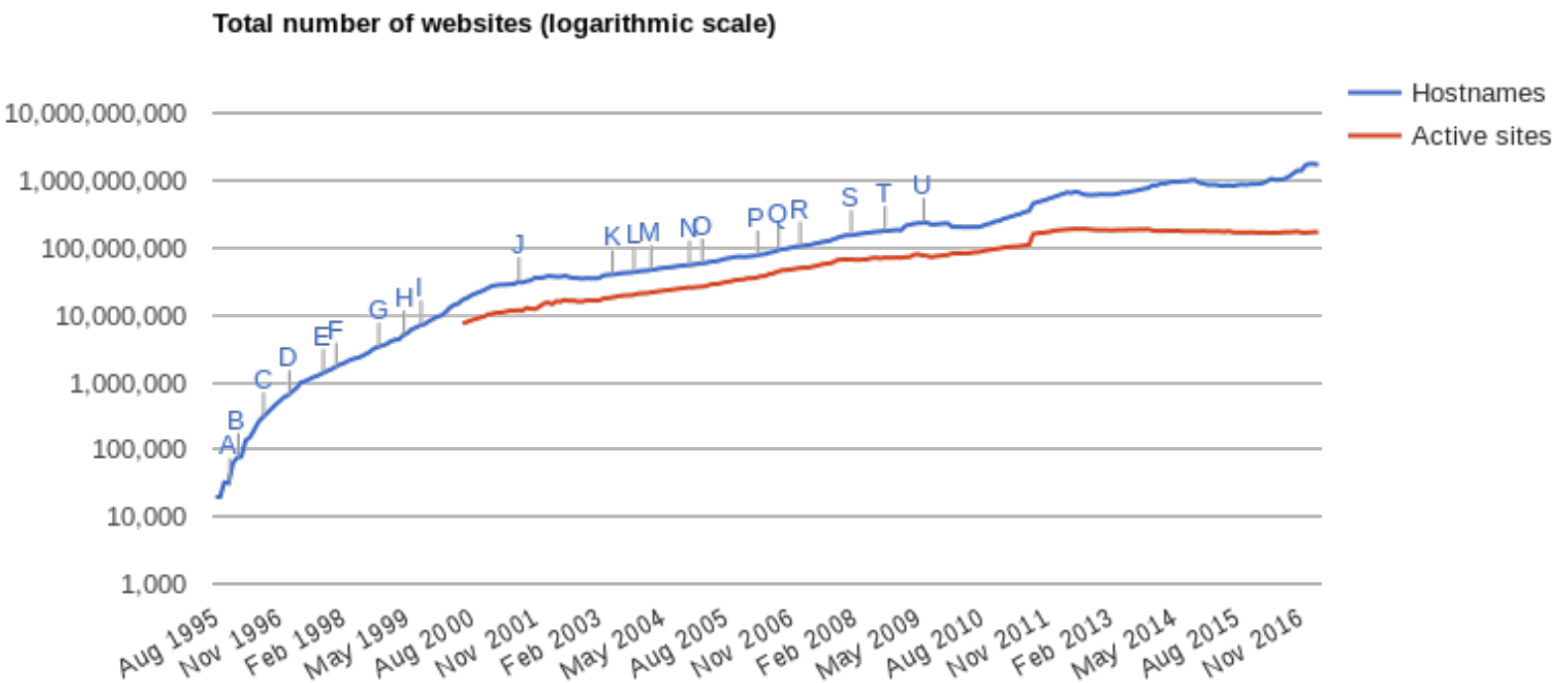
2017 1.760.630.795 Websites (unique Hostnames) [internetlivestats]
174.463.315 Websites (active)
6.271.146 Web-Servers (Web-Facing Computers)

Crawling the Web

Web Technology: World Wide Web

Definition 2 (World Wide Web, WWW [Wikipedia])

The World Wide Web is a network of documents, located by Uniform Resource Locators (URLs), connected by hypertext links, and accessible via the Internet.



[www.netcraft.com]

Crawling the Web

Web Technology: Addressing

The Internet Protocol (IP) is the principal communications protocol in the Internet. It delivers data packets between hosts, found solely based on their IP addresses. Two major versions are in use, IPv4 and IPv6:

- IPv4 addresses are 32 bit (4 byte) integers, denoted as sequence of 4 decimals, separated by points.

  `141.54.1.11/16`

- IPv6 addresses are 128 bit (32 byte) integers, denoted as sequence of 8 hexadecimals, separated by colons.

  `2001:0db8:85a3:08d3:1319:8a2e:0370:7344/64`

- IP addresses divide into address prefix and address suffix.
- The prefix (network ID) identifies the physical network.
- The suffix (host ID) identifies the computer within the netowrk ID's network.
- The prefix length is determined via CIDR notation (legacy: subnet mask).

Crawling the Web

Web Technology: Addressing

The domain name system (DNS) resolves hostnames to their hosts' IP addresses.

➜ DNS servers form a distributed database

First version:

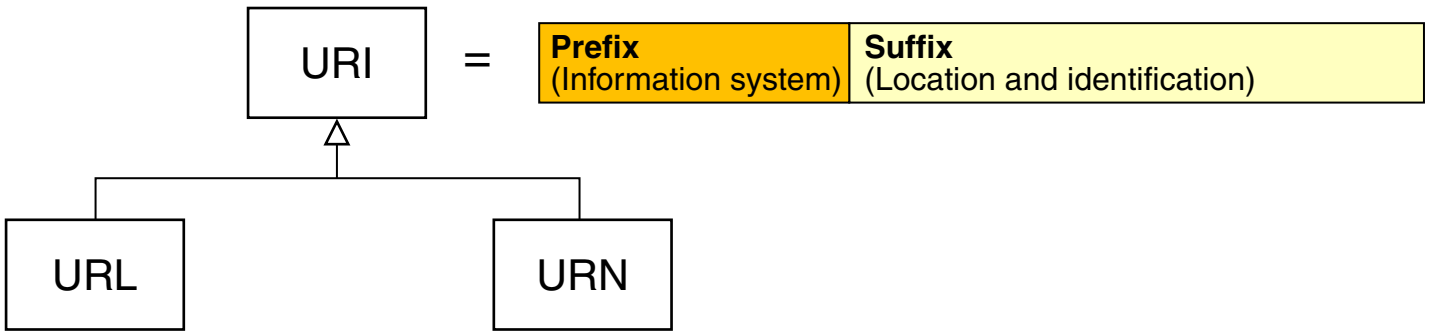
- All name-address pairs were collected in a central master file, which was distributed via FTP to other servers.
- Does not scale, since local organization becomes impossible.

Current version:

- Hierarchical organization via organizational partitioning (.com, .edu, .gov, .mil, etc.) as well as geographic partitioning (.de, .uk, .fr, etc.)
- The suffix following the last dot is called top level domain, TLD. List of current and new TLDs.

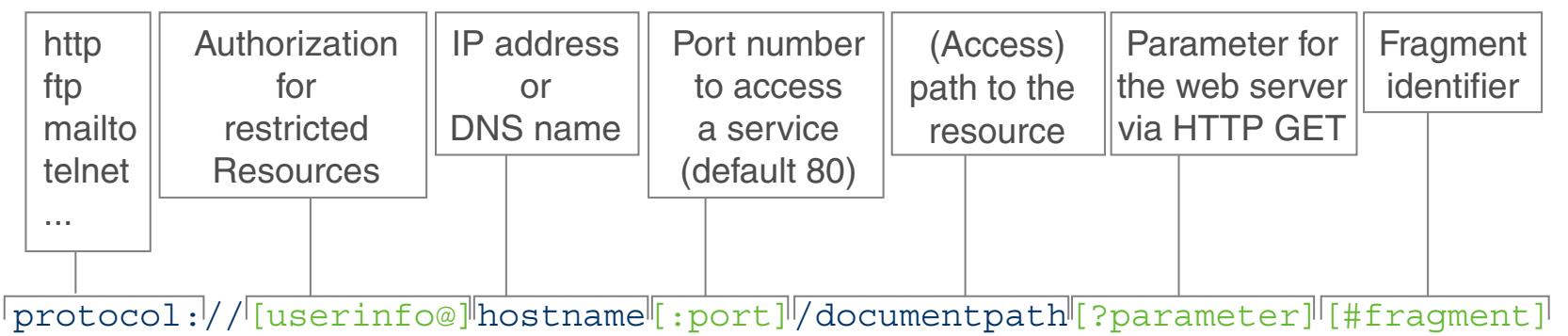
Crawling the Web

Web Technology: Addressing



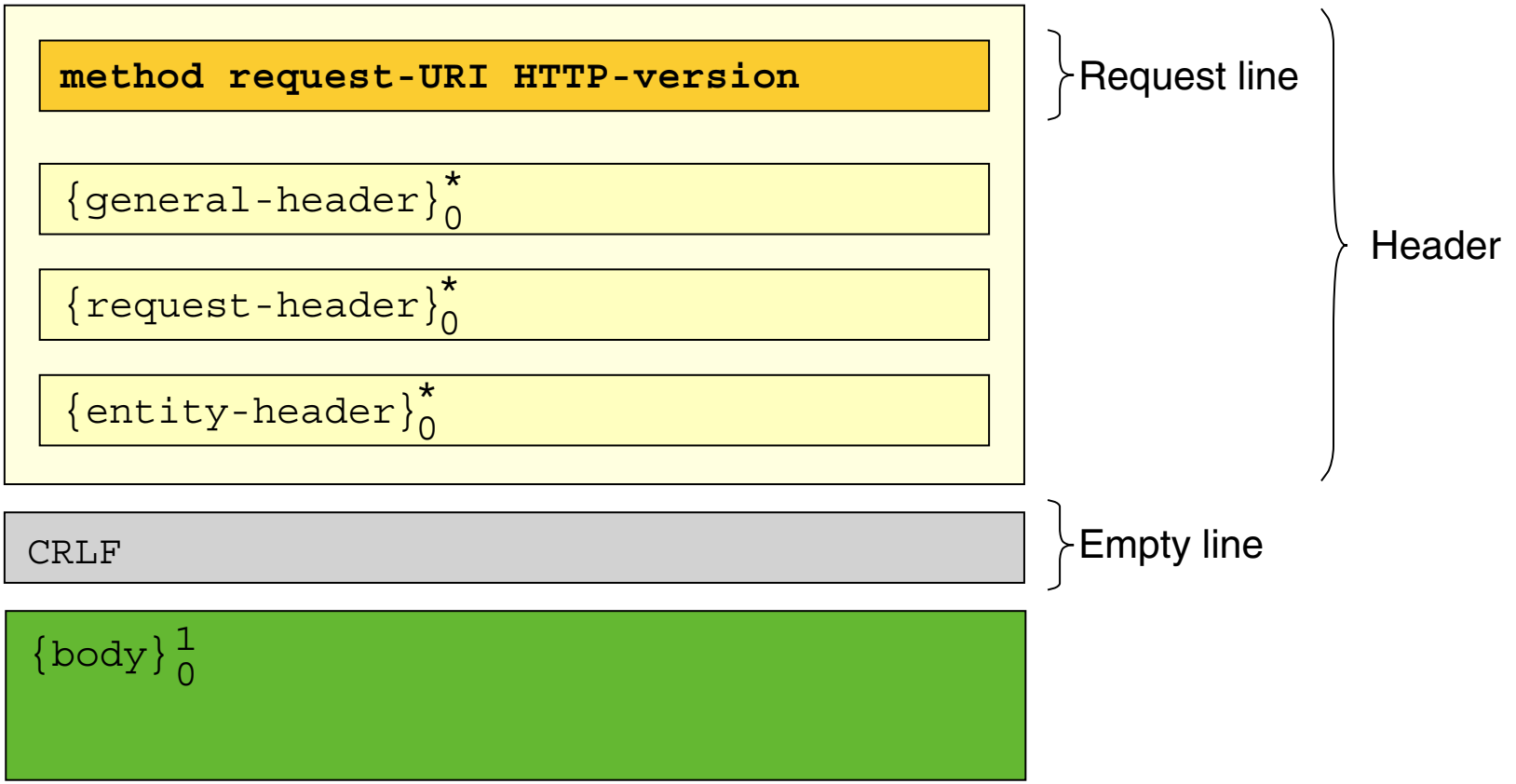
A Uniform Resource Identifier (URI) provides a simple and extensible means for identifying a resource. It can be further classified as a locator, a name, or both.

The term “Uniform Resource Locator” (URL) refers to URIs that, in addition to identifying a resource, provide a means of locating the resource by describing its primary access mechanism (e.g., its network “location”). [RFC 3986]



Crawling the Web

Web Technology: Hypertext Transfer Protocol (HTTP)

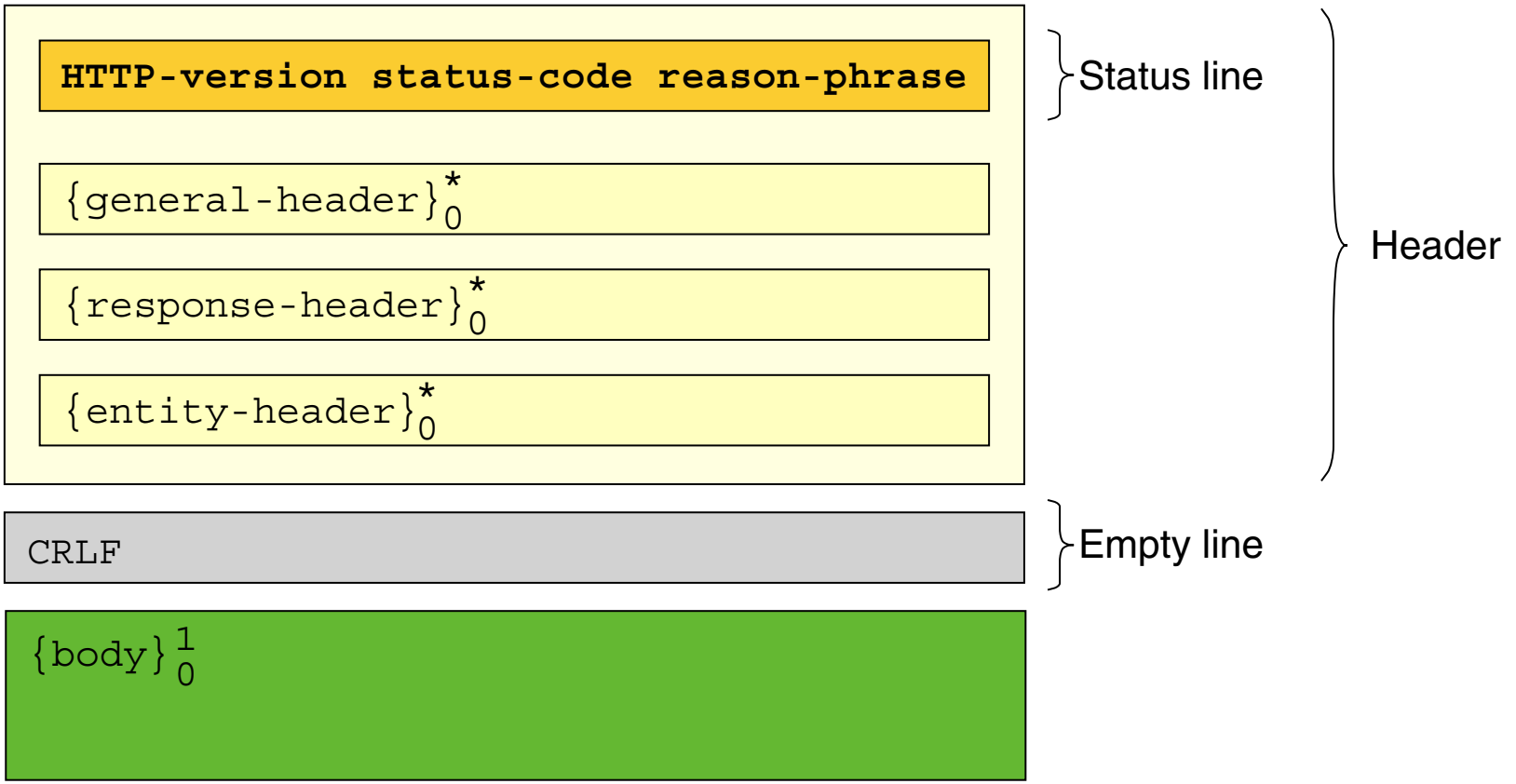


Example for a request line: `GET www.example.com/index.html HTTP/1.0`

Methods: GET, POST, HEAD, PUT, DELETE, OPTIONS, TRACE, CONNECT

Crawling the Web

Web Technology: Hypertext Transfer Protocol (HTTP)

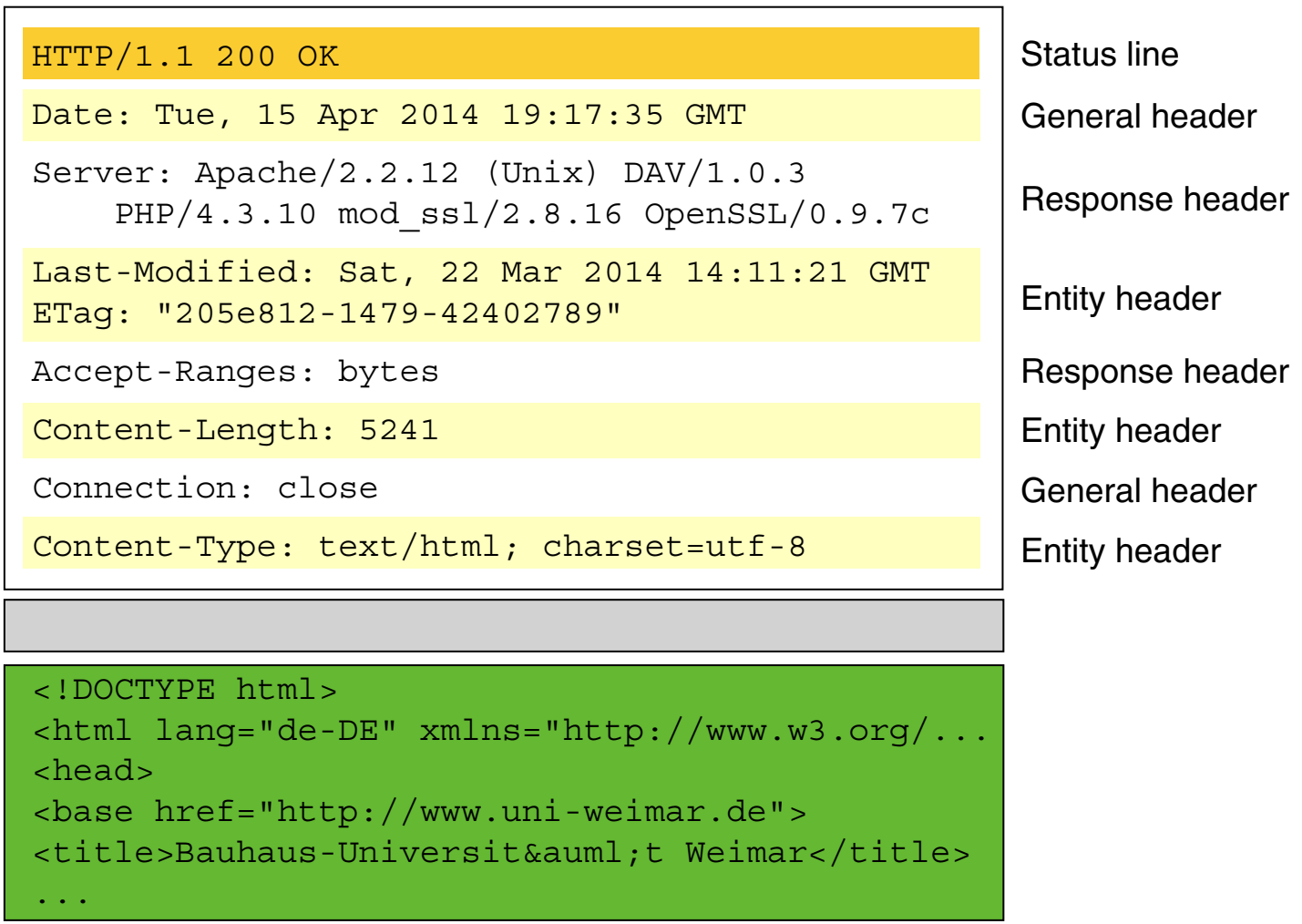


Example for a status line: `HTTP/1.0 200 OK`

Status codes: 200 OK, 301 Moved permanently, 304 Not modified, 404 Not found, 500 Internal server error, etc.

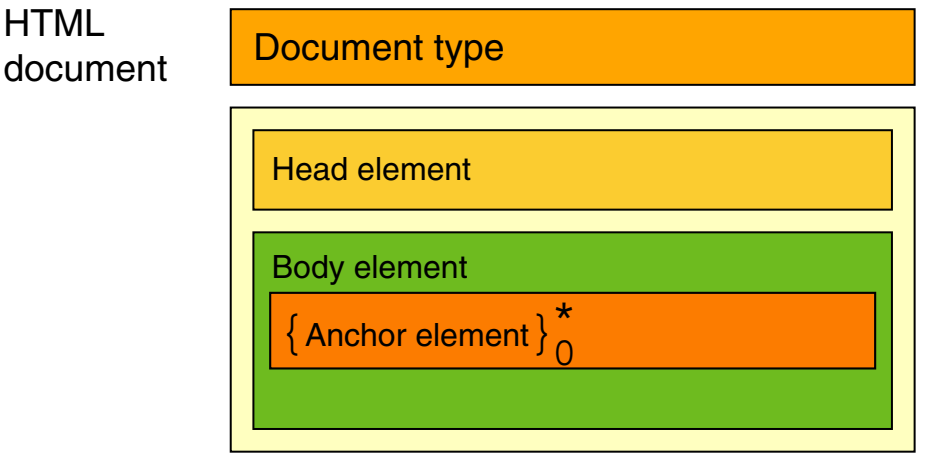
Crawling the Web

Web Technology: Hypertext Transfer Protocol (HTTP)



Crawling the Web

Web Technology: Hypertext Markup Language (HTML)



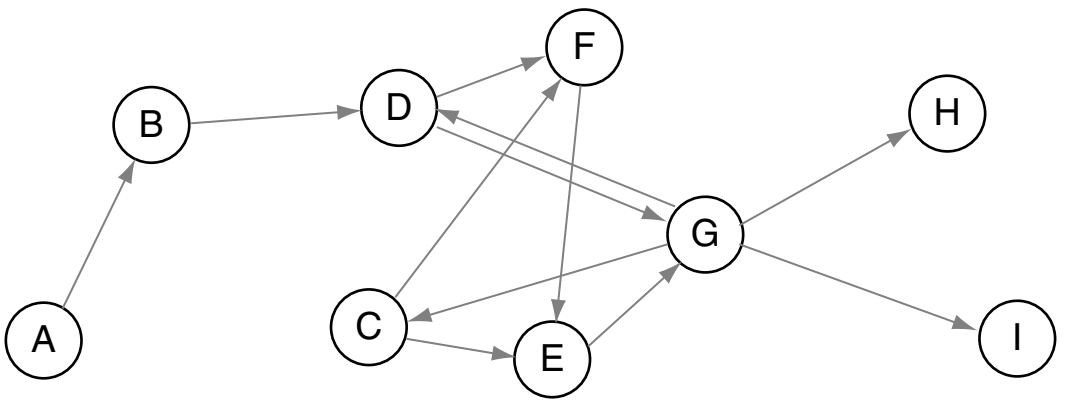

```
<!DOCTYPE html>

<html>
  <head>
    ...
  </head>
  <body>
    ...
    <a href="URI">link</a>
    ...
  </body>
</html>
```

- The `<html>` element represents the document root. [W3C REC 4.1]
- The `<head>` element represents meta data. [W3C REC 4.2]
- The `<body>` element represents document contents. [W3C REC 4.3]
- The `<a>` (anchor) element represents a hyperlink. [W3C REC 4.5.1]

`<a href="`*URI*`">`	Target defined by *URI*
`Ex.: <a href="#`*Identifier*`">`	Target element with id attribute within document
`Ex.: <a href="`*Path*`#`*Identifier*`">`	Target in relative document (i.e., same web server)

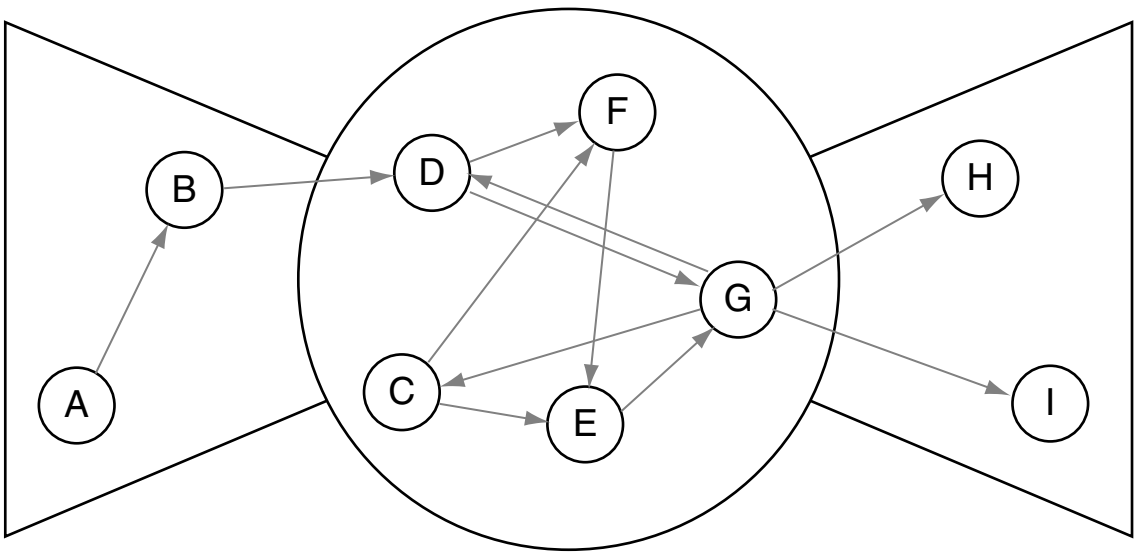
Crawling the Web

The Web Graph [Broder 2000]



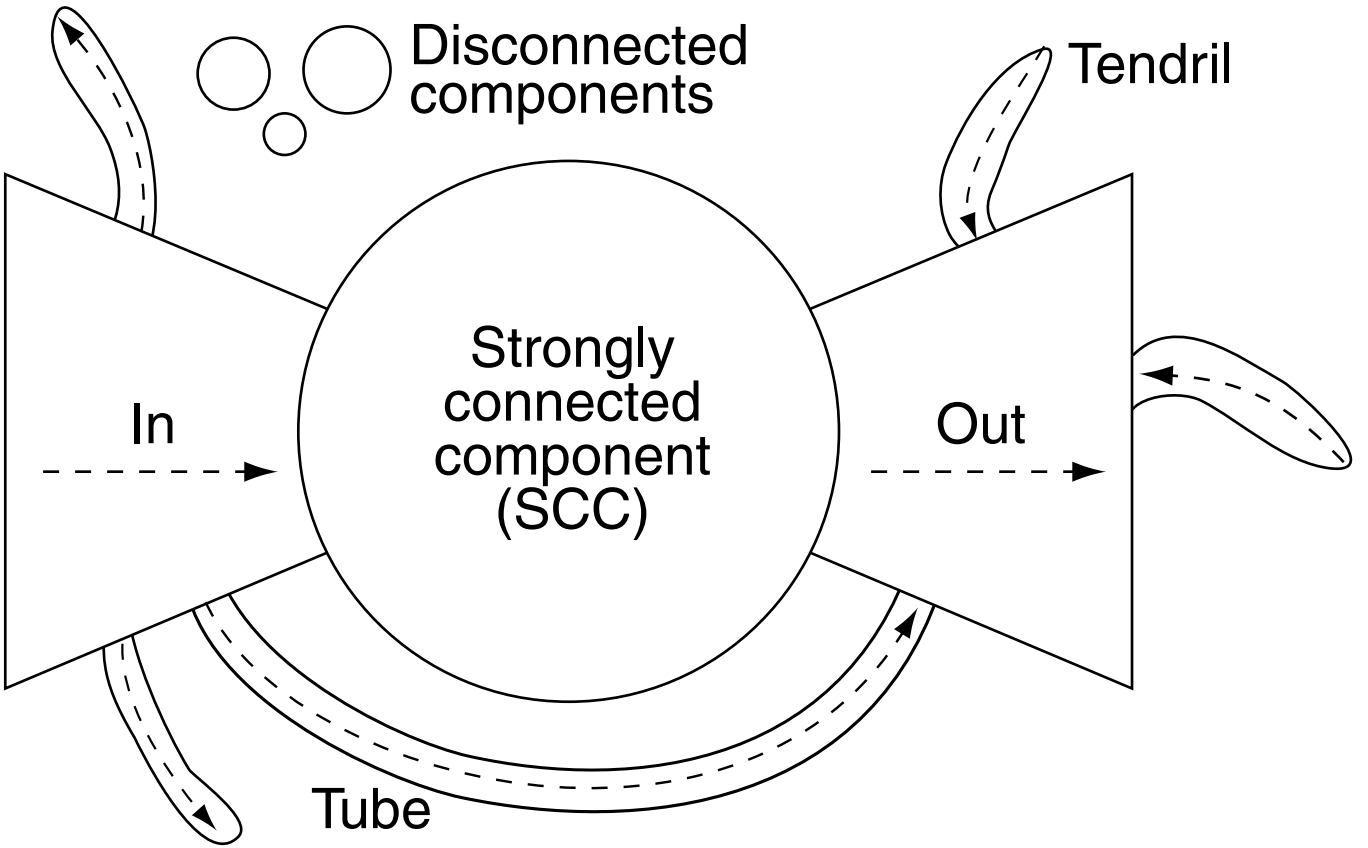
Crawling the Web

The Web Graph [Broder 2000]



Crawling the Web

The Web Graph [Broder 2000]



- The fraction of web pages with i in-links is proportional to $1/i^{2.1}$ (power law).
- SCC, In, Out, and the tendrils are of roughly equal size (SCC a little larger).
- Probability of a path between random pages is 0.24, the expected length 16.

Crawling the Web

Crawling Hypertext

The web can be crawled by traversing the web graph.

A basic crawling algorithm:

Algorithm: BFSCrawler
Input: seeds (list of URLs to start with),
store (document store)

```
1:  frontier ← seeds
2:  while frontier is not empty do
3:      url ← frontier.dequeue()
4:      page ← getURL(url)
5:      store.add(url, page)
6:      for each linkedUrl in page do
7:          if not store.has(linkedUrl) then
8:              frontier.enqueue(linkedUrl)
9:          end if
10:     end for
11: end while
```

Properties:

- Implements breadth-first search
- URLs are handled first in, first out
- Resilient to loops in the web graph
- Recrawls same pages if URL is different (e.g., http vs. https)
- Crawls site's pages in a row if linked consecutive on a page
- Does not recrawl pages

➜ Domain-specific adjustments necessary

Crawling the Web

Requirements [Manning 2008]

- **Selectivity**
  Strategies and policies for choosing web pages to download. This particularly includes the identification of malicious web pages and the prioritization of high-quality web pages.
- **Politeness**
  Adherence to explicit restrictions to download web pages. Prevention of unnecessary load on a web server.
- **Freshness**
  Strategies for recrawling web pages to obtain the most recent version as soon as possible.
- **Efficiency**
  Optimal utilization of available network bandwidth and hardware.
- **Scalability**
  Linear increase of efficiency by adding resources.
- **Extensibility**
  Modular architecture for easy integration and adaptation to new data formats and crawling strategies.

Crawling the Web

Selectivity

A crawler implements a selection policy determining which pages are crawled.

- Crawl seeds (Where to begin?)
  Initialization of a crawler frontier with previously collected URLs, so-called seeds.

- Crawl target (What to crawl?)
  Simple answer: Everything. More specifically: Every document for which a search engine's user might search ("Where was that document again?"). For web search engines, only few exceptions apply. In general, predicting universal non-usefulness of documents is difficult.

- Crawl priority (What first?)
  Web pages that are *predictably* more important to the search engine's users than others. Web sites may be judged as a whole. In particular, pages comprising high-quality content.

- Crawl filtering (What to avoid?)
  "Spider traps", and web pages from web sites whose owners harbor malicious intents toward the search engine, or its users, such as spam pages.

Remarks:

- Crawling is also called spidering, as in a spider wandering the web.
- A spider trap is a web page that (un)intentionally serves dynamically generated pages that comprise links to other dynamically generated pages ad infinitum. Examples: number pages referring to successive numbers, date pages referring to successive dates, hierarchy pages referring to deeper levels.

Crawling the Web

Selectivity: Malicious Pages (Black-hat SEO*, Spam) [Wikipedia]

Malicious pages try to manipulate the search engine in pursuit of goals that are at odds with those of a search engine. The only way to do so is at crawling time.

- Cloaking
  Serving different pages to a crawler than to a human user for a given URL.
- Mirroring
  Reuse of another web page's contents, either duplicating or paraphrasing it.
- Keyword stuffing
  Including keywords or text into a web page which did not occur organically during web page creation; possibly as hidden text.
- Doorway page
  Pages redirecting human users to another page, or asking them to click to proceed, otherwise comprising only little, specifically chosen content for indexing.
- Link farm
  Networks of pages/sites from different domains linking to each other to boost their predicted importance. They can be automatically generated or emerge manually via link exchange.

*SEO: Search Engine Optimization

Crawling the Web

Politeness

A crawler which crawls third party resources must implement a politeness policy to comply with their owner's wishes and needs.

- Server load conservation

  A web server hosts many pages and may have many users. Requesting all pages at once, or in a short time frame, may overload the server, affecting other users. Politeness dictates to leave significant headroom in terms of server resources to other users.

- Copyright compliance (Robots Exclusion Protocol)

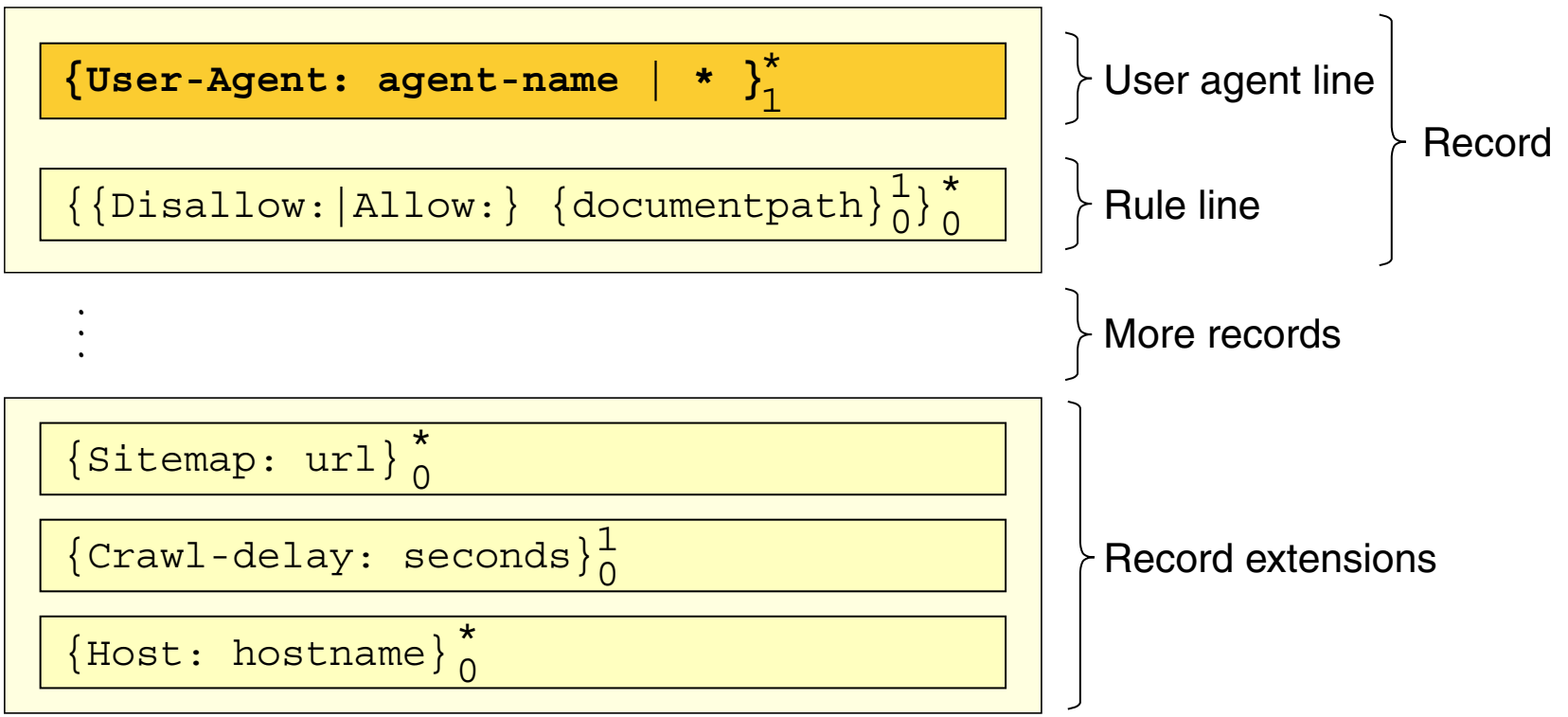
  Web site owners may not wish for parts or all of their pages to be indexed. Politeness dictates compliance with this wish. The robots exclusion protocol provides a standard interface to communicate such directives. Example:

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

User-agent:  FavoredCrawler
Disallow:

Sitemap:  http://www.example.com/sitemap.xml.gz
```

  FavoredCrawler may access everything, whereas every other crawler is disallowed to crawl anything, except for document paths starting with `/public/`.

Crawling the Web

Politeness: Robots Exclusion Protocol (robots.txt)



- The robots.txt must be at the document root of a domain:
  `www.example.com/robots.txt`
- The RFC has not been passed, but it is a de-facto standard.
- Extensions to records are interpreted only by some crawlers.

Remarks:

- The User-Agent line corresponds to the HTTP request header. A list of bots and the user agent lines they use. The `agent-name` is supposed to be just the name of the bot, not including extraneous information.
- The Sitemap extension was introduced as a kind of *Robots Inclusion Protocol*, telling crawlers with a standardized XML file where to look for pages, including ones that are not linked from anywhere else, and meta data such as last modification date, change frequency, priority, and alternative versions of a page, e.g., in different languages.
- The Host extension lets the crawler know which hostname is preferred for a site in cases the same site can be reached via multiple hostnames.
- Other means to disallow crawlers include adding response headers to HTTP requests:
  `X-Robots-Tag: [agent-name:] noindex, nofollow`,
  adding meta elements in a web page's header:
  `<meta name="robots" content="noindex,nofollow">`,
  adding the `rel="nofollow"` attribute to anchor elements:
  `<a href="URI" rel="nofollow">`,
  adding class attributes to arbitrary elements:
  `<div class="robots-noindex robots-nofollow">Text.</div>`,
  enclosing plain text with (crawler-dependent) structured comments:
  `<p>Do index this text. <!--googleoff: all-->Don't index this text.<!--googleon: all--></p>`,
  allowing for crawler-specific rules by replacing `robots` with agent names.

Crawling the Web

Politeness: Crawling Algorithm Revisited

A polite crawling algorithm:

Algorithm: PoliteCrawler
Input: seeds (list of URLs to start with),
store (document store)

```
 1: frontier ← seeds
 2: while frontier is not empty do
 3:     site ← frontier.nextSite()
 4:     url ← site.dequeue()
 5:     if site.permitsCrawl(url) then
 6:         page ← getURL(url)
 7:         store.add(url, page)
 8:         for each linkedUrl in page do
 9:             if not store.has(linkedUrl) then
10:                 site.enqueue(linkedUrl)
11:             end if
12:         end for
13:     end if
14: end while
```

Crawling the Web

Freshness

Previously crawled web pages may be revised or deleted. A crawler for a web search engine implements a recrawl policy to ensure that the most recent version of a web page is indexed.

- Update checking
  Web servers may report `Last-Modified: <date>` as a response header, indicating the last time the page under a given URL changed. A HTTP HEAD request will reveal the date without having to download the entire page again. Otherwise, update checking requires a recrawl, and a comparison of the most recent version with the previous one.
- Update prediction
  The crawler needs to learn to predict page updates to not waste resources on update checking, nor missing updates. Learning a web page's update probability is done by scheduling update checks, adjusting the recrawl delay based on whether a page changed since its last recrawl. Adjustments may be done via binary search, or time series analyses.

Examples:

- Stock market pages change frequently, and their most recent versions are what people will be searching for.
- News pages are updated as news unfold. People search for current events.

Crawling the Web

Freshness: Metric

We call a crawled web page fresh, if it corresponds to the version currently online, and stale otherwise.

A simple freshness metric (also called “freshness”) is the fraction of stale pages in a crawl at a given point in time. The smaller the fraction, the fresher the crawl.

Optimizing for this freshness metric is problematic:

- Why?

Crawling the Web

Freshness: Metric

We call a crawled web page fresh, if it corresponds to the version currently online, and stale otherwise.

A simple freshness metric (also called “freshness”) is the fraction of stale pages in a crawl at a given point in time. The smaller the fraction, the fresher the crawl.

Optimizing for this freshness metric is problematic:

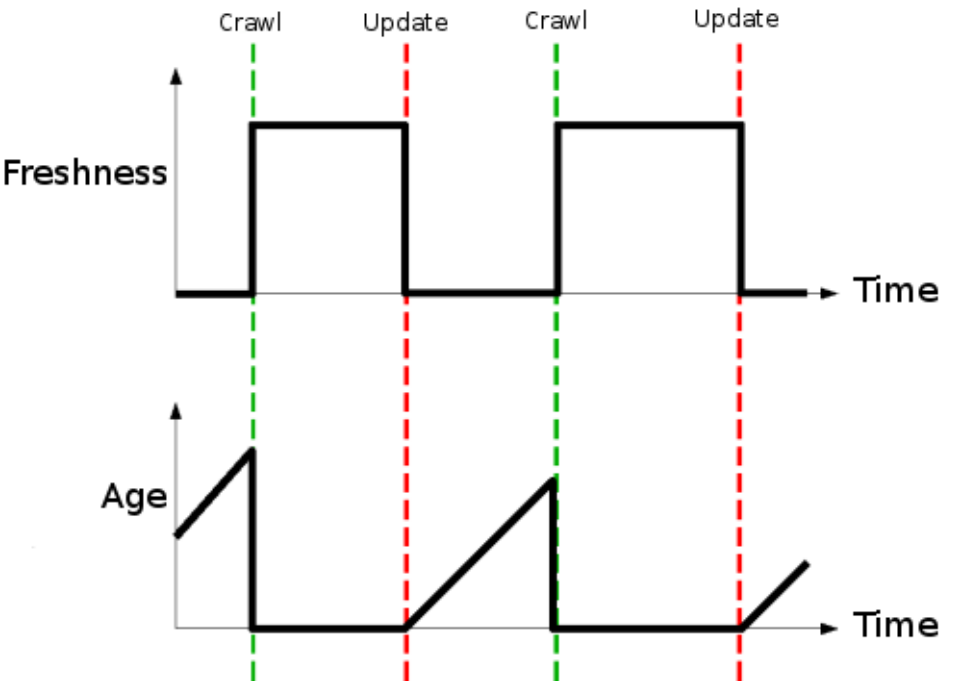
- Recrawling requires resources, and a crawler has only a limited amount.
- Allocating resources to recrawls will optimize freshness, if and only if, web pages are recrawled ordered from low to high update probability.
- Web pages with a change rate below the minimum recrawl delay (e.g., to conserve server load) will never be fresh.
- Hence, frequently changing web pages may not be crawled.
- Important pages often have a high change rate.

➜ Optimizing for this freshness metric contradicts the need for freshness.

Crawling the Web

Freshness: Age Metric

An idea for a better metric:



- Freshness is completely lost after an update.
- Age constantly grows after an update.
- This allows for measuring the urgency of an update.
- Problem: we do not know when an update occurs, so we have to guess.

Crawling the Web

Freshness: Age Metric [Cho and Garcia-Molina 2003]

Let λ denote the change rate of a page (i.e., λ changes per day). We calculate the expected age of a page at time t (e.g., t days) after it was last crawled as follows:

$$\text{Age}(\lambda, t) = \int_0^t P(\text{page changed at time } x)(t - x)dx,$$

where $(t - x)$ measures age at recrawl time t and change time $x < t$. Mutiplying by the change probability at time x, and integrating over it yields the expected value.

Crawling the Web

Freshness: Age Metric [Cho and Garcia-Molina 2003]

Let λ denote the change rate of a page (i.e., λ changes per day). We calculate the expected age of a page at time t (e.g., t days) after it was last crawled as follows:

$$\text{Age}(\lambda, t) = \int_0^t P(\text{page changed at time } x)(t - x)dx,$$

where $(t - x)$ measures age at recrawl time t and change time $x < t$. Mutiplying by the change probability at time x, and integrating over it yields the expected value.

Assuming that web page updates are governed by a Poisson process, the probability P at change rate λ can be expressed with $\lambda e^{-\lambda x}$, yielding:

$$\text{Age}(\lambda, t) = \int_0^t \lambda e^{-\lambda x}(t - x)dx$$

$$= \frac{\lambda t + e^{-\lambda t} - 1}{\lambda}.$$

For example, let $\lambda = 1/7$ (e.g., one change a week):

$$\text{Age}(1/7, 7) = 2.6$$

denotes the expexted age of a page when recrawling it at $t = 7$.

Crawling the Web

Freshness: Age Metric [Cho and Garcia-Molina 2003]

Let λ denote the change rate of a page (i.e., λ changes per day). We calculate the expected age of a page at time t (e.g., t days) after it was last crawled as follows:

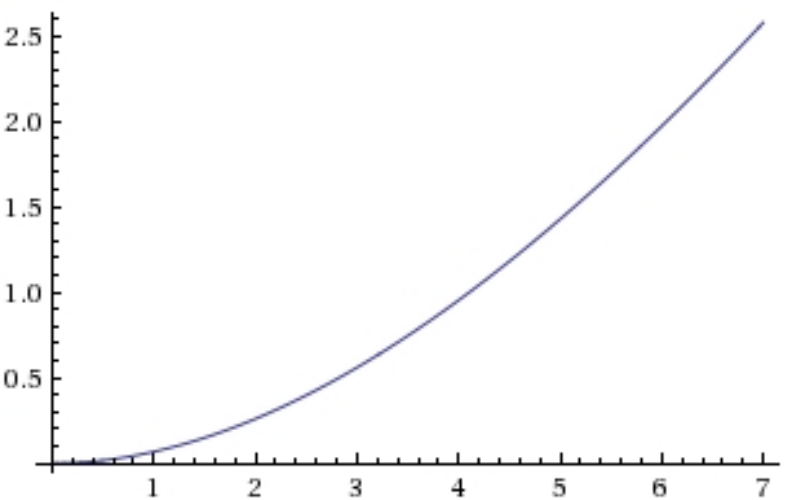
$$\text{Age}(\lambda, t) = \int_0^t P(\text{page changed at time } x)(t - x)dx,$$

where $(t - x)$ measures age at recrawl time t and change time $x < t$. Mutiplying by the change probability at time x, and integrating over it yields the expected value.

Assuming that web page updates are governed by a Poisson process, the probability P at change rate λ can be expressed with $\lambda e^{-\lambda x}$, yielding:

$$\text{Age}(\lambda, t) = \int_0^t \lambda e^{-\lambda x}(t - x)dx$$

$$= \frac{\lambda t + e^{-\lambda t} - 1}{\lambda}.$$



Crawling the Web

Freshness: Age Metric [Cho and Garcia-Molina 2003]

Observations:

- The second derivative of Age is $\lambda e^{-\lambda t}$.
- For $\lambda > 0$, the second derivative is always positive.
- Age increases at an accellerating rate.
- Age also increases with the change rate λ.
- The older a page and the more frequently it changes, the higher its Age.

➜ Optimizing for low overall Age prioritizes page recrawls more accurately.

Crawling the Web

Efficiency and Scalability

A crawler should utilize its resources efficiently, and be scalable linearly by adding more resources. Key bottlenecks:

- Internet connection utilization
  - Downloads incur communication costs: DNS lookup, IP connection, data transfer.
  - At 1MB/s bandwidth and 20KB web page size, 50 pages per second can be crawled.
  - 80ms connection latency + 20ms data transfer yields 100ms per page request.
  - Downloading 50 pages consecutively takes 5 seconds (underutilization).
  - 5 parallel connections are needed to maximize utilization.
  - This presumes the bandwidth of web servers is the same as ours. However, this is not the case; typically, the bandwidth of a crawler is faster than that of a web server.
- Politeness policy (e.g., crawl delays)
  - Recall that we want to maximize utilization of the crawlers internet connection, not that of the connections of web servers.
  - Downloads from a web site are restricted by crawl delays (say, 30 seconds).
  - At 50 pages/s download rate, we need pages from 1500 sites in the frontier.

In a distributed crawler, each crawl node is assigned a specific web site so that no two crawl nodes download simultaneously from the same site.

Remarks:

- Suppose sites are “more distant” with only 100KB/s bandwidth and 500ms latency. How many connections are needed?
  - 100KB/s bandwidth means 200ms for data transfer of 20KB
  - Results in 700ms per page request (200ms data transfer + 500ms latency)
  - 50 pages times 700ms yields 35 seconds
  - 35 connections are needed to transfer 50 pages per second
- Hash functions are used to assign hostnames to crawl nodes at random.

Crawling the Web

Extensibility

The web evolves at a rapid pace. In particular, web pages have turned from static HTML pages to dynamic pieces of software, compiled and executed within a browser. Crawlers must be adaptable to new technology.

Web pages/sites notoriously difficult to crawl (called the “deep web”):

- Private sites
  - No incoming links, or may require login with a valid account
  - Example: paywalled news pages (which still want to get indexed)
- Form results
  - Sites that can be reached only after entering some data into a form
  - Example: flight ticket selling (enter destination + dates etc.)
- Dynamic pages
  - Pages that generate content via JavaScript or other client technologies
  - Crawler needs to execute scripts (e.g., Google does so)
  - Significantly slows down the whole process