Freshness

Crawling, session 6

Page Freshness

The web is constantly changing as content is added, deleted, and modified. In order for a crawler to reflect the web as users will encounter it, it needs to recrawl content soon after it changes.

This need for freshness is key to providing a good search engine experience. For instance, when breaking news develops, users will rely on your search engine to stay updated.

It’s also important to refresh less time-sensitive documents so the results list doesn’t contain spurious links to deleted or modified data.

HTTP HEAD Requests

A crawler can determine whether a page has changed by making an HTTP HEAD request.

The response provides the HTTP status code and headers, but not the document body. The headers include information about when the content was last updated.

However, it's not feasible to constantly send HEAD requests, so this isn't an adequate strategy for freshness.

Request

```
HEAD /csinfo/people.html HTTP/1.1
Host: www.cs.umass.edu
```

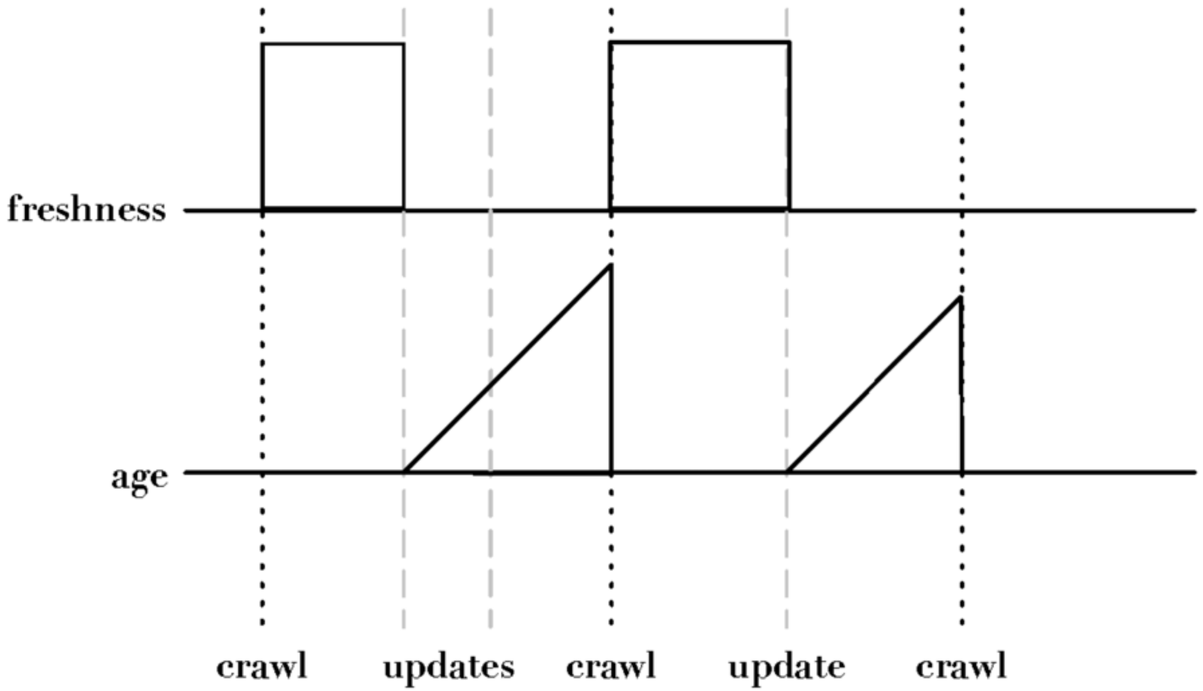
Response

```
HTTP/1.1 200 OK
Date: Thu, 03 Apr 2008 05:17:54 GMT
Server: Apache/2.0.52 (CentOS)
Last-Modified: Fri, 04 Jan 2008 15:28:39 GMT
ETag: "239c33-2576-2a2837c0"
Accept-Ranges: bytes
Content-Length: 9590
Connection: close
Content-Type: text/html; charset=ISO-8859-1
```

Freshness vs. Age

It turns out that optimizing to minimize freshness is a poor strategy: it can lead the crawler to ignore important sites.

Instead, it's better to re-crawl pages when the age of the last crawled version exceeds some limit. The *age* of a page is the elapsed time since the first update after the most recent crawl.



Freshness is binary, age is continuous.

Expected Page Age

The expected age of a page t days after it was crawled depends on its update probability:

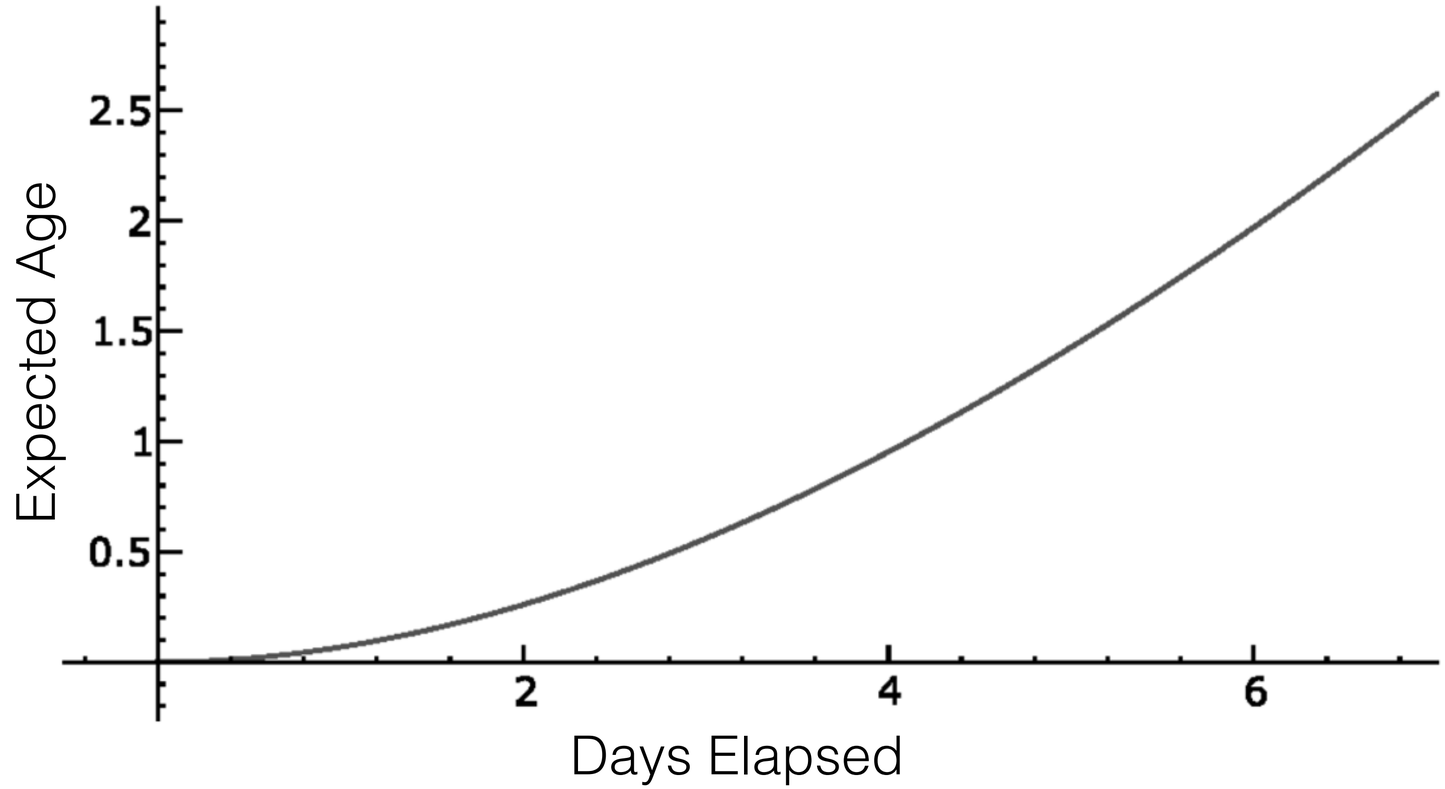
$$age(\lambda, t) = \int_0^t P(\textbf{page changed at time } x)(t - x)dx$$

On average, page updates follow a Poisson distribution – the time until the next update is governed by an exponential distribution. This makes the expected age:

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

Cost of Not Re-crawling

The cost of not re-crawling a page grows exponentially in the time since the last crawl. For instance, with page update frequency $\lambda = 1/7$ days:



Freshness vs. Coverage

The opposing needs of Freshness and Coverage need to be balanced in the scoring function used to select the next page to crawl.

Finding an optimal balance is still an open question. Fairly recent studies have shown that even large name-brand search engines only do a modest job at finding the most recent content.

However, a reasonable approach is to include a term in the page priority function for the expected age of the page content. For important domains, you can track the site-wide update frequency λ.

Wrapping Up

The web is constantly changing, and re-crawling the latest changes quickly can be challenging.

It turns out that aggressively re-crawling as soon as a page changes is sometimes the wrong approach: it's better to use a cost function associated with the expected age of the content, and tolerate a small delay between re-crawls.

Next, we'll take a look at what can go wrong with crawling.