A SURVEY OF HTTP CACHING IMPLEMENTATIONS ON THE OPEN SEMANTIC WEB

Kjetil Kjernsmo, http://folk.uio.no/kjekje/#cache-survey,

Department of Informatics, University of Oslo, Norway. @KKjernsmo

MOTIVATION AND TAKE-AWAY

The HTTP standard and Internet infrastucture has good facilities, we shouldn' speculate on whether they can be used.

QUESTIONS TO BE ANSWERED

- How prevalent is support for caching?
- Are certain implementations better than others?
- How long may resources be cached?
- Does it matter whether it is linked data, a SPARQL endpoint, etc?
- Can lifetimes be estimated any other way?

CACHING IN HTTP AND THE INTERNET

From **RFC 7234**:

The goal of caching in HTTP/1.1 is to significantly improve performance by reusing a prior response message to satisfy a current request.

CACHING VS. CONDITIONAL REQUESTS

Caching (defined in RFC 7234)

HTTP headers where the server says how long the server thinks the response will remain valid

Conditional Requests (defined in RFC 7232)

A mechanism where a client asks the server: "is the response I have still valid?"

MOST RELEVANT HTTP HEADERS – CACHING

Cache-Control

For fine-grained control, values including max-age and no-store.

Expires

A time for when the response should not be used

MOST RELEVANT HTTP HEADERS – CONDITIONAL REQUESTS

RESPONSE HEADERS

ETag
: An opaque validator for the response

Last-Modified
: A timestamp giving the time of modificaton of the resource

REQUEST HEADERS

If-None-Match
: Check the opaque validator

If-Modified-Since
: Give a timestamp to check if the resource has changed.

FRESHNESS LIFETIME

Number of seconds that the response may be used without contacting the origin server.

HEURISTIC FRESHNESS LIFETIME

RFC7234 allows for heuristics to be used to estimate freshness lifetime.

Also suggests a useful heuristic based on Last-Modified.

We might also use RDF data or machine learning to estimate.

METHODOLOGY

Gather as many different hosts as we could and send HTTP requests to them and record relevant data.

Classify resources into

1. Dataset descriptions
2. SPARQL Endpoints
3. Vocabularies
4. Generic information resources

GATHERING HOSTS

To decide where to go, we used:

- **Linked Open Vocabularies**
- **prefix.cc**
- **SPARQLES survey**
- **Billion Triples Challenge 2014**

Got a list of 3117 unique hosts, and did 7745 HTTP requests

WHY ISN'T IT BIGGER?

- We only included valid data
- **Important assumption:** Cache headers are set mostly on a per-host basis
- Not the RDFa Web
- Others do not report per-host statistics, but LODstats has 4442 error-free datasets

POSSIBLE BIASES

- Biases may have been introduced by the coverage and data reduction.
- Biases due to discarding momentarily dysfunctinal parts of the Semantic Web was not investigated.
- Other possible biases are considered in a companion technical report.

ANALYSIS

1. Distribution of freshness lifetime
    1. Standards-compliant
    2. Simple heuristic
2. Dublin Core properties
3. Do certain server implementations provide better caching support than others?
4. Cache (re)validation

SUCCESSFUL RESPONSES

We had 2965 successful responses

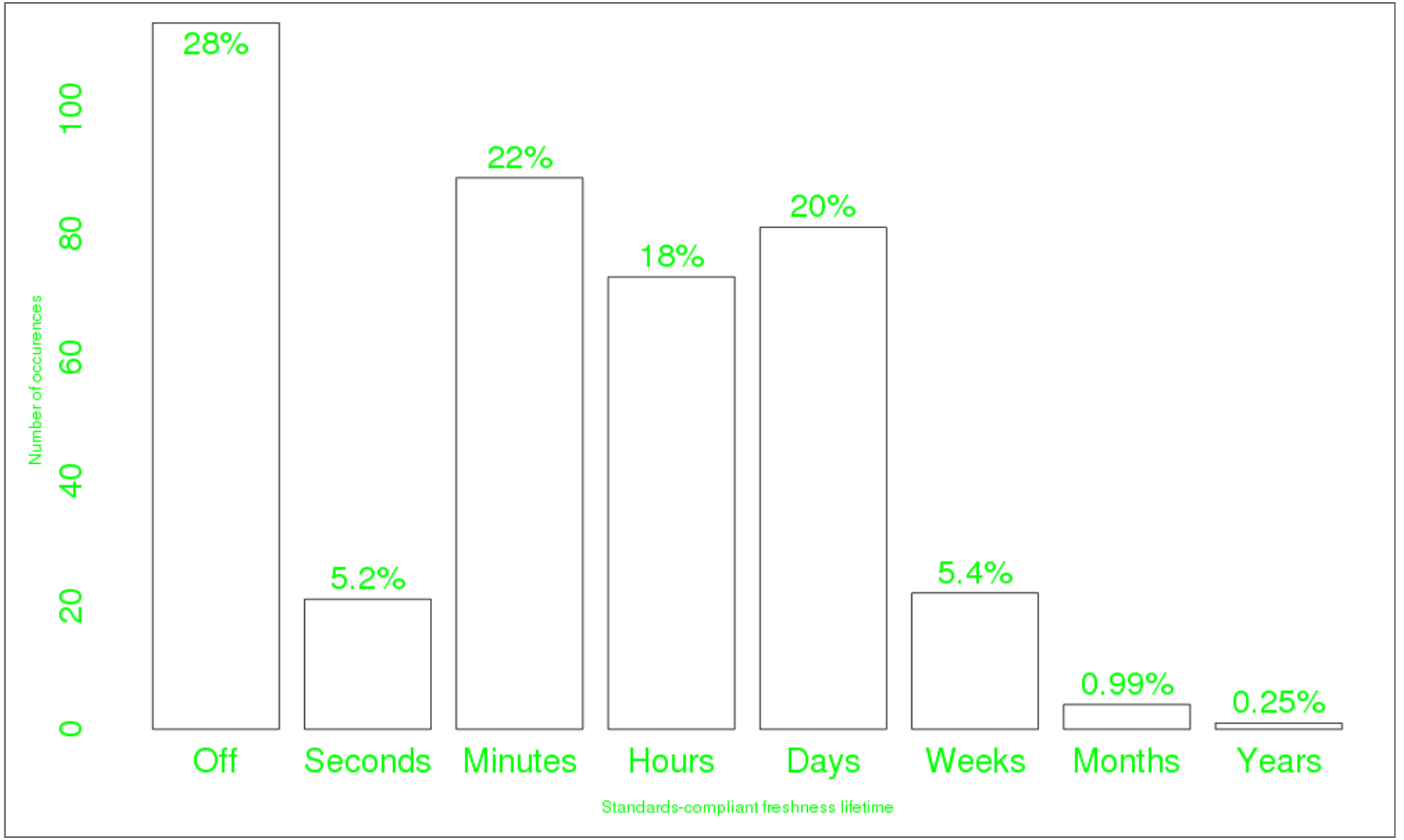
Successful response defined as

- Successful HTTP Response (after redirects)
- Valid RDF media type or SPARQL result
- Must parse into an RDF Model

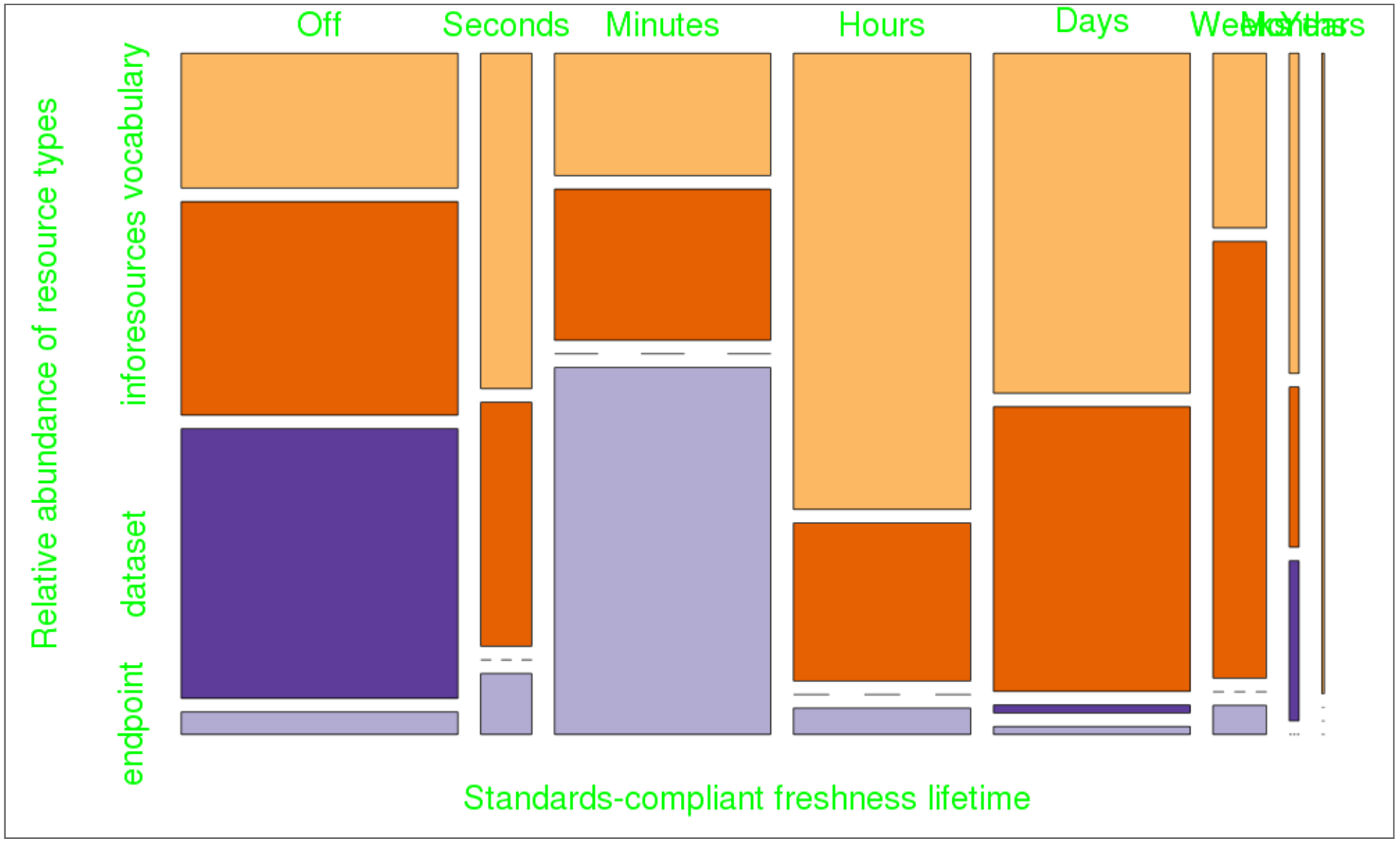
STANDARDS-COMPLIANT CACHING IN NUMBERS

- **405** resources returned parsable cache headers
- **114** did so to prohibit caching
- **3** contained conflicting headers
- Usually Cache-Control and Expires both occurred, former most common.
- **269** used Cache-Control for other purposes than freshness lifetime

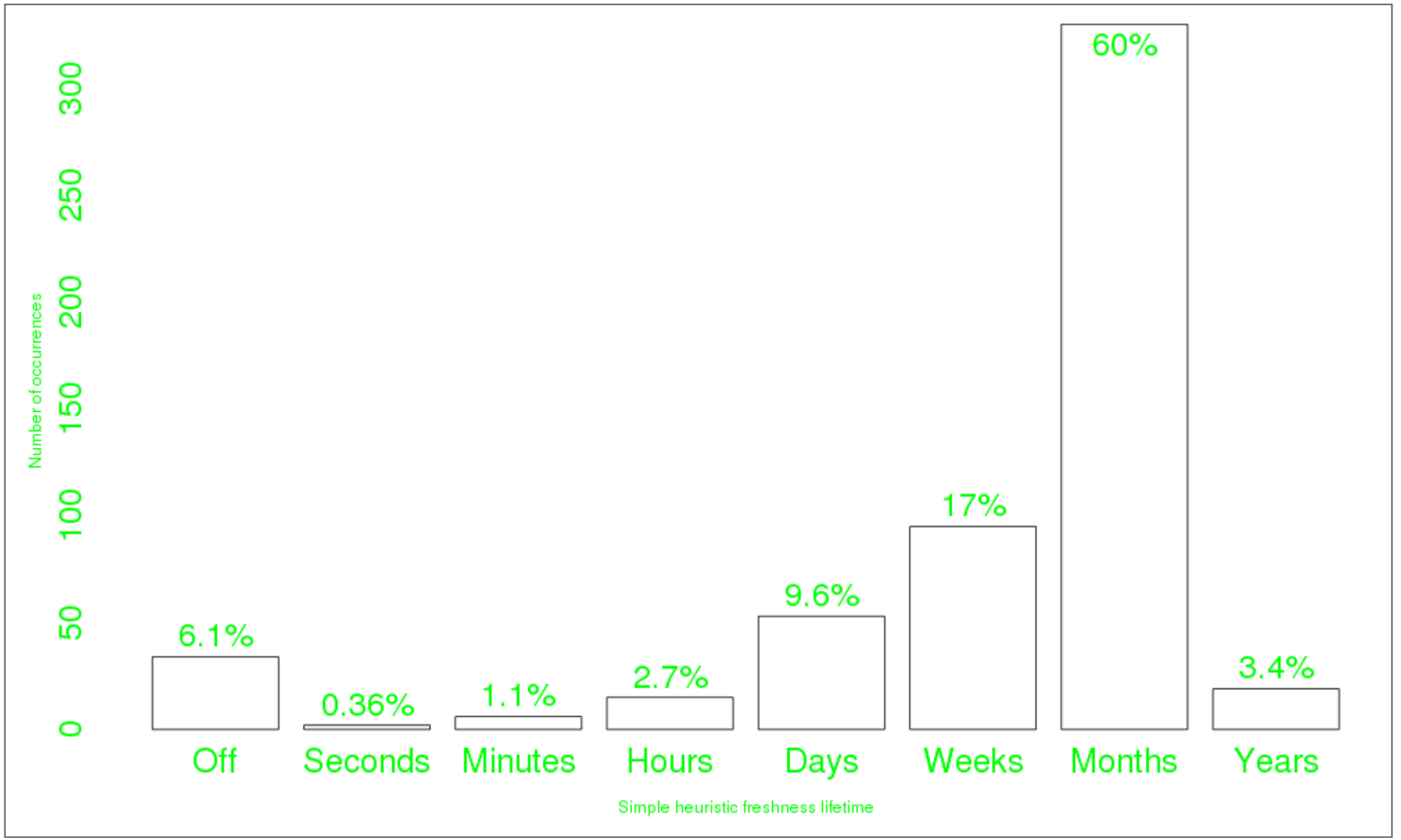
STANDARDS-COMPLIANT FRESHNESS LIFETIME



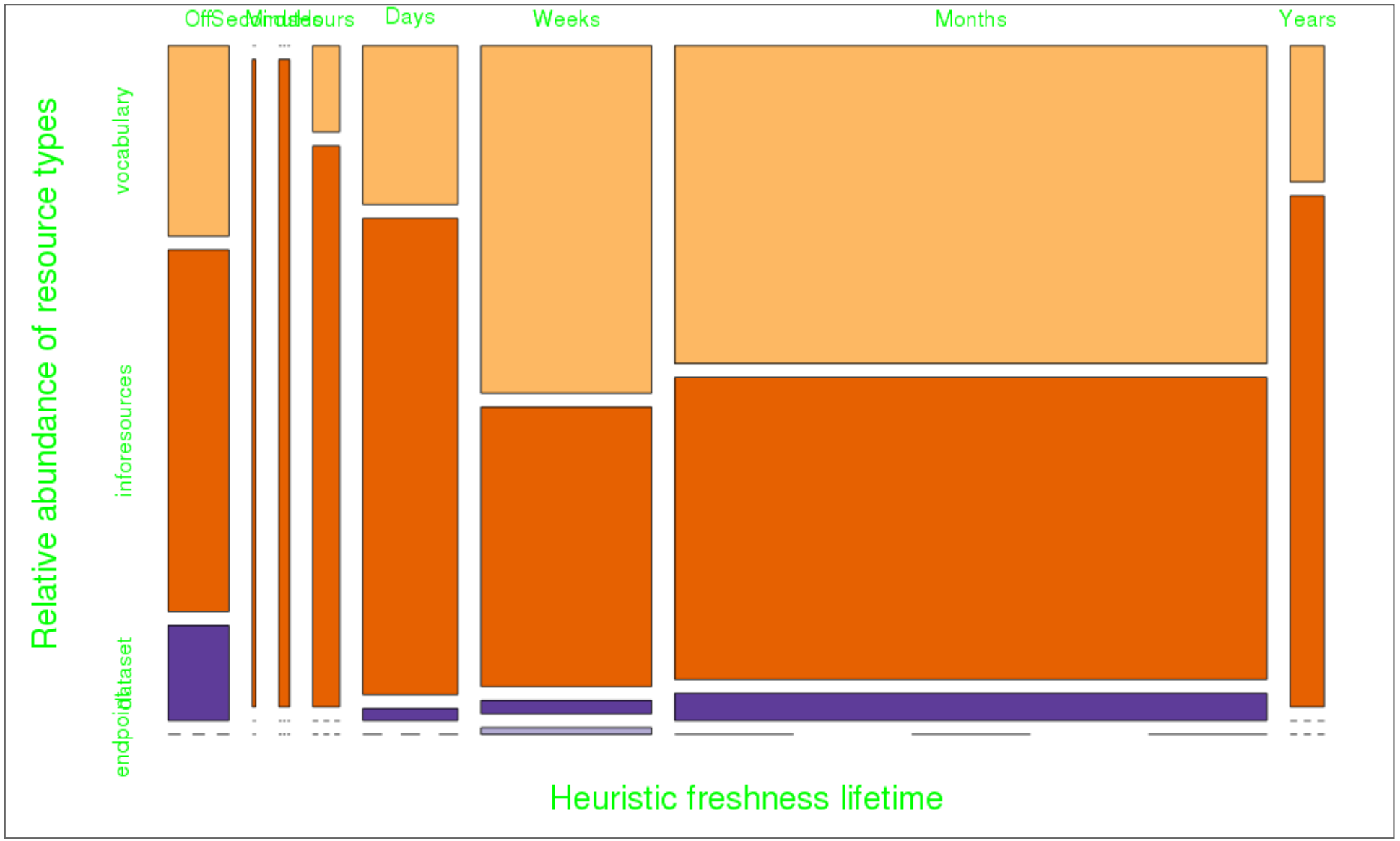
STANDARDS-COMPLIANT FRESHNESS LIFETIME



SIMPLE HEURISTIC FRESHNESS LIFETIME



SIMPLE HEURISTIC FRESHNESS LIFETIME



DUBLIN CORE PROPERTIES

Predicate	Number of occurences
dct:modified	2687
dct:valid	21
dct:accrualPeriodicity	2
dct:date	36
dct:created	389
dct:issued	1475

CACHE VALIDATION

IN OUR EXCERPT FROM BTC-2014:

- 1733 had ETag
- 690 had Last-Modified
- with great overlap
- 911 were still verified as fresh

CACHE VALIDATION

IN INITIAL RESPONSES:

- 1260 had ETag
  - 606 for vocabularies
  - 117 for datasets
  - 12 for endpoints
  - 525 for unclassified

CONDITIONAL REQUEST SUPPORT

Another 1822 requests to check if conditional requests were actually supporte

Found 85 faulty implementations

SERVER-HEADERS

DFE/largefile
git_frontend
nginx/1.3.9
thin 1.6.0 codename Greek Yogurt
Oracle-Application-Server-10g/10.1.3.4.0 Oracle-HTTP-Server
Oracle-Application-Server-10g/10.1.3.4.0 Oracle-HTTP-Server
TwistedWeb/8.2.0
RDF::Endpoint/0.07
Jetty(6.1.26)
nginx/1.6.1
Jigsaw/2.3.0-beta3
Apache/2.2.9 (Win32) PHP/5.2.6
Apache/2.4.10 (Unix) mod_fcgid/2.3.9

GFE/2.0
RDF::LinkedData/0.70
Apache/2.2.17 (Unix) mod_wsgi/3.3 Python/2.6.6
Virtuoso/07.10.3211 (Linux) i686-generic-linux-glibc212-64 VDB
Apache/2.2.24 (Unix) mod_ssl/2.2.24 OpenSSL/0.9.8y
Apache/2.2.22 (Fedora)
INSEE
GitHub.com

CONTINGENCY TABLE TEST

We can do a statistical test to see if certain headers occur significantly more frequently.

Pearson's χ^2 test with simulated p-value (based on 10000 replicates)

For standards-compliant caching

Supports that some server may be better than others at p-value = 0.0001

For other usable features

Supports that some server may be better than others at p-value = 0.0001

CONCLUSIONS

- **We found moderate uptake for HTTP caching and conditional requests**
- Many resources change slowly, but standard-compliant cache doesn't reflect this
- Errors are common, but not in caching headers
- Possible to compute heuristic freshness from headers or Dublin Core in many cases
- Conditional requests seldomly supported on SPARQL endpoints, but standards-compliant freshness lifetimes have been seen

HALL OF FAME

- DBPedia
- Dydra
- RDF::LinkedData
- Graphity
- Callimachus
- UniProt

FUTURE WORK

- Learn change frequency of resources (but please respect the standards!)
- Do curated collections (e.g. LOV, SPARQLES) have different characteristics
- Understand what the differences between implementations are based on fingerprinting
- User support systems to help estimate freshness lifetime
- SPARQL caching based on prefetching results for query answering on proxy
- Understanding the actual impact of caching of data

RECOMMENDATIONS

- You should set cache headers, on a best-effort basis!
- Framework authors should make it easy to be accurate
- If headers can't be set, set dct:valid and dct:modified
- DBMS authors should make it much cheaper to retrieve a modification time of a subgraph than to retrieve the graph itself.
- A change periodicity predicate should be standardized in VoID
- For now, caches are key-value databases, accept that for short-term impact

THANKS