Extreme Web Caching for Faster Web Browsing

Ali Raza, Yasir Zaki
NYU Abu Dhabi, UAE
{ali.raza,
yasir.zaki}@nyu.edu

Thomas Pötsch
University of Bremen,
Germany
tpoetsch@uni-bremen.de

Jay Chen,
Lakshmi Subramanian
NYU and CTED, NYUAD
{jchen,lakshmi}@cs.nyu.edu

ABSTRACT

Modern web pages are very complex; each web page consists of hundreds of objects that are linked from various servers all over the world. While mechanisms such as caching reduce the overall number of end-to-end requests saving bandwidth and loading time, there is still a large portion of content that is re-fetched – despite not having changed. In this demo, we present *Extreme Cache*, a web caching architecture that enhances the web browsing experience through a smart pre-fetching engine. Our extreme cache tries to predict the rate of change of web page objects to bring cacheable content closer to the user.

CCS Concepts

•**Networks** → *Network architectures;*

1. INTRODUCTION

There have been a number of recent projects that explore how caching can effect the performance of web browsing. Xia et. al.[2] found that layout and source code of web pages that block subsequent objects to load are highly cacheable. The authors have also shown that a large amount of content (especially HTML) does not change significantly over the course of days or months.

Motivated by their findings, we analyzed Alexa's top 20 web pages over the course of three months to see how web pages change over time. We periodically (every 10 minutes) downloaded pages and observed that a large portion of the page objects does not change frequently over time (such as CSS, Javascripts, Images, HTML, and etc.). Moreover, even when a certain object changes, we observe that for text-based objects the changes are relatively small. In many cases, the changes are only limited to a few lines of code within the entire object. We also observed that these minor changes are mostly related to advertising or randomly generated (tracking) IDs and are probably not of major importance to the user.

Most existing caching solutions rely on HTTP max-age headers option [1] to identify the freshness of a cached object. To confirm whether the max-age option is actually set correctly by the content provider and if it reflects the correct number (i.e. timestamp) on when an object would change, we compared the max-age option of every object and of every version of our recorded set of Alexa's top 20 web pages. Since there were 10 min between consecutive page requests, we can tell whether an object has changed or not by comparing the MD5 hash of the objects in both versions. We find that the max-age option does not reflect the reality of when objects change. This causes either a large number of unnecessary requests or stale objects in the cache.

In order to cope with the findings mentioned above, we developed a novel caching architecture called *Extreme Cache* that aims to (a) estimate the correct rate of change for every object within a web page and then overwrites the max-age option which was set by the content provider and (b) bring web content closer to the user. Hence, unnecessary web requests can be avoided and content can be served from a closer edge cache.

2. KEY FINDINGS

We looked into how objects change over time within several popular web sites and have discovered that a large number of objects did not change over time. In addition, even though some objects might have changed over time, the ratio of the change to the overall object size is very small and does not require to fetch the entire object again. Moreover, caching policies are not dictated properly by the content owners. The key observations that we have made in our analysis can be summarized as follows:

- Web publishers do not configure the cache-control headers in HTTP correctly/carefully. For the top 20 web pages from Alexa, we found that a large number of objects violate this, i.e. changing either before or after the specified max-age.

SIGCOMM '15 August 17-21, 2015, London, United Kingdom
© 2015 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-3542-3/15/08.
DOI: http://dx.doi.org/10.1145/2785956.2790032

- Changes in Javascript, HTML, and CSS objects are not significant and it is not required to fetch the whole object again. Sending only diffs can tremendously speed up page load times.

Figure 1 shows the correlation between the max-age and the change rate of all objects for the recorded top 20 Alexa web sites. The x-axis represents the actual max-age set by the content provider that is communicated through the HTTP headers; the y-axis represents the actual time the object took to change. If the max-age value is a true mean of the object's change rate, then the data should be highly correlated along a 45 degree line. The figure shows that the max-age generally does not correlate with the object's actual rate of change.

For the same dataset, Figure 2 shows the percentage of objects that stays the same after a certain time has elapsed in comparison to the first version of the website. The idea is to show how similar a certain web page stays after a certain time. The x-axis indicates the times at which the similarity is computed and the y-axis represents the similarity index calculated as number of similar objects/total number of objects. It can be seen that for a long period of time, around five hours, the similarity index is above 65%. For certain web pages, the similarity index remains above 90% even after a day.

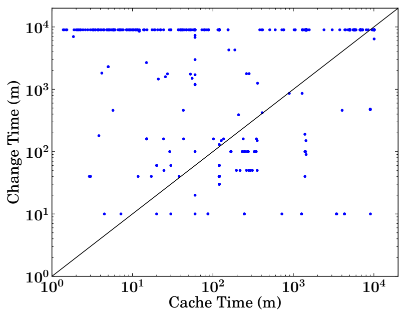

Figure 1: Per object max-age vs. change rate

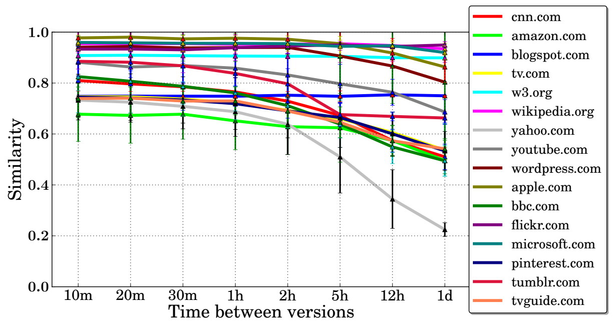

Figure 2: Page similarity over time

3. ARCHITECTURE

Our extreme cache architecture aims at enhancing the hit rate of legacy caching systems and to reduce unnecessary content requests. Our solution is easy to deploy since it neither requires any additional changes to the network nor to the client browsers and web servers.

The extreme cache architecture is implemented at different locations in the network and consists of the following two components: a cloud controller and an edge cache. Figure 3 shows the basic framework of the extreme cache. As can be seen in the figure, the edge cache is close to the client and has a backhaul connection to the Internet so as to accommodate cache misses or non-cacheable content. The cloud controller is connected to both, the Internet (for pre-fetching) and to the edge cache so as to populate, prioritize, and update its cache. The edge cache contains the most recent version of each object even before it is requested by a user. The cloud controller is responsible for pre-fetching the web pages, estimating the objects rate of change, and populating/updating the edge cache.

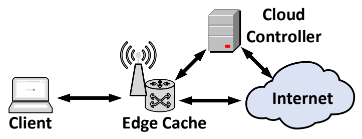

Figure 3: Extreme cache architecture

3.1 Cloud Controller

The functionality of the cloud controller is to periodically request objects of web pages from the Internet and then comparing the objects with the previous (cached) version. This enables the cloud controller to estimate the rate at which the objects change and will help to decide when and how frequently it needs to update the edge cache. If possible, the updates are done in terms of delta updates (i.e. only sending the changes (diffs) instead of the whole object) in order to save bandwidth. Further, it overrides the max-age option with the predicted change rate (one option for the estimated max-age of an object is to use to use the moving average of the object's change rate). The cloud controller can control several edge caches and prioritizes objects based on their expected benefits to the cache hit rate.

3.2 Edge Cache

The edge cache is a regular caching proxy that does not have any additional intelligence. It is responsible for serving the client requests with content stored in its cache. In case of a cache hit, it delivers the cached objects; otherwise it requests the content from the origin. The edge cache also records its client's access patterns and updates the cloud controller to help it prioritize and decide on what to pre-fetch.

4. REFERENCES

[1] R. Fielding, M. Nottingham, and J. Reschke. Hypertext transfer protocol (http/1.1): Caching, June 2014. RFC7234.

[2] X. S. Wang, A. Krishnamurthy, and D. Wetherall. How much can we micro-cache web pages? In *Proceedings of the 2014 Conference on Internet Measurement Conference*, IMC '14, pages 249–256, New York, NY, USA, 2014. ACM.