Cache'n DASH: Efficient Caching for DASH

Parikshit Juluri
University of Missouri - Kansas City
pjuluri@umkc.edu

Deep Medhi
University of Missouri - Kansas City
dmedhi@umkc.edu

ABSTRACT

HTTP-based video streaming services have been dominating the global IP traffic over the last few years. Caching of video content reduces the load on the content servers. In the case of Dynamic Adaptive Streaming over HTTP (DASH), for every video the server needs to host multiple representations of the same video file. These individual representations are further broken down into smaller segments. Hence, for each video the server needs to host thousands of segments out of which, the client downloads a subset of the segments. Also, depending on the network conditions, the adaptation scheme used at the client-end might request a different set of video segments (varying in bitrate) for the same video. The caching of DASH videos presents unique challenges. In order to optimize the cache hits and minimize the misses for DASH video streaming services we propose an Adaptation Aware Cache (AAC) framework to determine the segments that are to be prefetched and retained in the cache. In the current scheme, we use bandwidth estimates at the cache server and the knowledge of the rate adaptation scheme used by the client to estimate the next segment requests, thus improving the pre-fetching at the cache.

1. INTRODUCTION

Globally, IP video traffic is dominating all consumer download traffic. Popular video streaming services e.g, Netflix and YouTube predominately use MPEG-DASH for their video delivery. With DASH, each video file is encoded into multiple different *representations*, each varying in bitrate, resolution, and format. The *representations* are divided into smaller *segments* of fixed playback duration. The individual *segments* are stored as separate files with unique URLs on the web-server. A Media Presentation Description (MPD) file is created for each video. The MPD file has a list of all *representations* and the URLs for the individual segments for the video. At the client end, the DASH video player downloads the MPD file, and retrieves the video metadata. Based on the network conditions and current video buffer occupancy, the client requests the appropriate video segment.

With the increasing popularity of video services, the network and service providers need to employ caching and prefetching strategies to reduce access latencies. Caching of web objects has been widely deployed, however, the video objects are significantly larger in size than regular web objects. Hence, aggressive prefetching of videos could be very expensive, resulting in overloading the link between the cache and the content server. In order to maximize the byte-hits using prefetching it is necessary to better predict the requests. Although the segment numbers for DASH streams is known a priori, but the main challenge is to predict the correct bitrate for the next segment.

In this paper, we propose an Adaptation Aware Cache (AAC) framework for DASH videos that uses bandwidth measurement at the cache server and the knowledge of the adaptation scheme to predict the segment requests. The main objective is to maximize the byte-hits, and minimizing the unnecessary prefetches.

SIGCOMM'15, August 17–21, 2015, London, United Kingdom.

ACM ISBN 978-1-4503-3542-3/15/08..

DOI: http://dx.doi.org/10.1145/2785956.2790015

2. RELATED WORK

DASH is a fairly recent technology with the first open standard for MPEG-DASH being released as recent as November, 2011. Although the use of cache servers for video streaming has been studied earlier, not many consider DASH video streaming services. Our aim is to develop a cache framework that is optimized for serving DASH video streams. We identify the areas where DASH differs from other web requests, and we propose the techniques that could be used to improve the cache performance, thus improving the QoE. Other studies have proposed modifications to the cache servers to leverage the popularity of the video streams [1]. In [1], the cache server prefetches the *segments* based on the bandwidth between the client and the cache. However, the rate adaptation in this case is driven by the cache server, which is not the case with current DASH, that uses client-end adaptation.

An integrated prefetching and caching scheme for adaptive video streaming is presented in [2]. In this study, the authors propose prefetching the next segment of the current bitrate. However, they assume a high probability of the client requesting the same bitrate. In the current framework, we use bandwidth measurements at the cache server to predict the bitrates for the next segments.

3. OVERVIEW OF THE FRAMEWORK

In this section, we describe our **Adaptation Aware Cache** (AAC) framework as shown in Fig. 1. The aim of the AAC

framework is to predict future requests based on the path bandwidth measurements between the cache server and the client, the expected response to the variations in the throughput by the adaptation scheme used by the client. At the highest level, the AAC framework has two modules: the DASH Request Handler and the Cache Manager.

3.1 DASH Request Handler

The DASH request handler in AAC handles all the HTTP video requests received by the cache server. For DASH streaming, the HTTP requests consists of the video MPD file followed by the individual video segment requests. The **MPD parser** module parses the DASH MPD files to retrieve the list of bitrates. A mapping for the video ID and the list of available bitrates is created and stored in a hashmap in the *Segment Request Index*. Only requests for videos that are not already in the *Segment Request Index* need to be parsed. The **Bandwidth Estimator** module measures the path bandwidth of the active HTTP sessions. The current cache servers typically run on sophisticated hardware and are equipped to collect the path bandwidth measurements [3]. The bandwidth estimator is a critical component and there is a need for a good estimator.

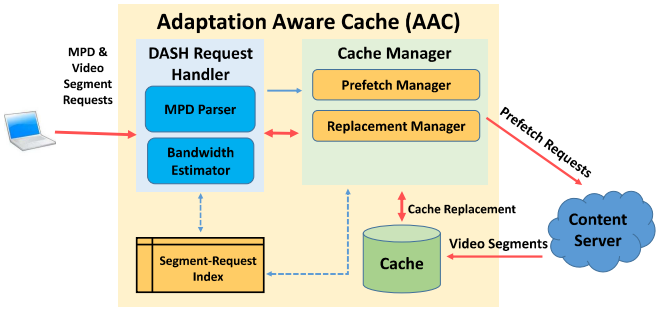

Figure 1: Adaptation Aware Cache Framework

3.2 Cache Manager Module

The cache manager module receives the video requests and the bandwidth measurement values form the *Bandwidth Estimator* module. As shown in Fig. 1, the *Cache Manager* consists of two modules: the Prefetch Manager and Replacement Manager. The cache manager interacts with the *Segment Request Index* to retrieve the video bitrate information. It also manages the requests to the cache. The cache in this case could also be a cluster of cache servers. The cache manager sends the prefetch and cache-miss requests to the content server.

Prefetch Manager: The DASH segment requests are based on a predefined order defined in the MPD file. However, the bitrate for each segment is selected by an adaptation algorithm employed by the client. Typical rate adaptation algorithms select the lowest bitrates for the initial set of segments. Later, based on the observed throughput, the algorithm ramps-up the bitrate and eventually settles at the appropriate bitrate. In Fig. 2 we plot the change in the bitrates requested by a typical rate adaptation algorithm. We found similar patterns with other adaption algorithms [4,5]. Based on these access patterns we can infer that prefetching the next segment of the current playback bitrate for the entire playback duration can result in unnecessary prefetches; especially during the ramp-up stage. Based on the bandwidth measurements collected at the cache server, the prefetch manager could predict if the client is expected to ramp-up the quality of the video. By making smart decisions on prefetches during the ramp-up phase avoids bitrate oscillations [3].

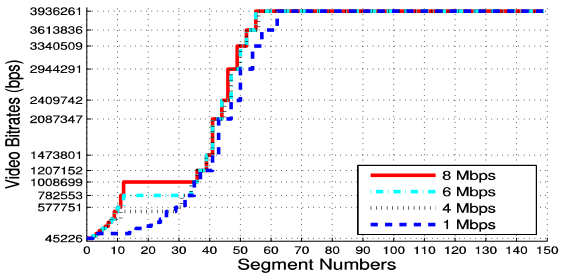

Figure 2: Bitrate Access Patterns for DASH video

Replacement Manager: The role of the replacement manager is to determine the segments that are to be replaced to accommodate new segments. The segment manager retrieves the information of the current active video streaming session from the *Segment Request Index* to determine which segments are most likely to be requested by the current video sessions and the least probable segments are replaced.

Segment Request Index stores two different hashmaps. The first hashmap is used to map the video ID with the available bitrates that are are retrieved from the DASH MPD file. The second hashmap is a mapping of the user ID with the corresponding bitrate measurements. Both these hashmaps are updated by the *Request Handler* and accessed by the *Cache Manager* modules, respectively.

4. SUMMARY AND FUTURE WORK

AAC is designed to improve the utility of the proxy cache by predicting the video segment download patterns. By using bandwidth measurements to improve the accuracy of prefetching, we can also avoid bitrate oscillations at the user end. Future work, beyond finishing the implementation and testing, focusses on optimizing the cache utility.

5. REFERENCES

[1] R. Rejaie, H. Yu, M. Handley, and D. Estrin, "Multimedia proxy caching mechanism for quality adaptive streaming applications in the internet," in *IEEE INFOCOM 2000*, vol. 2. IEEE, pp. 980–989.

[2] K. Liang, J. Hao, R. Zimmermann, and D. K. Yau, "Integrated prefetching and caching for adaptive video streaming over HTTP: an online approach," in *Proceedings of the 6th ACM MMSys*, 2015, pp. 142–152.

[3] D. H. Lee, C. Dovrolis, and A. C. Begen, "Caching in HTTP adaptive streaming: Friend or foe?" in *NOSSDAV*. ACM, 2014, p. 31.

[4] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *ACM SIGCOMM*, 2014, pp. 187–198.

[5] P. Juluri, V. Tamarapalli, and D. Medhi, "SARA: Segment Aware Rate Adaptation algorithm for Dynamic Adaptive Streaming over HTTP," in *ICC QoE-FI Workshop*, 2015.