Improved caching for HTTP-based Video on Demand using Scalable Video Coding

Yago Sánchez, Thomas Schierl, Cornelius Hellge, Thomas Wiegand - *Fraunhofer HHI, Germany*
Dohy Hong - *N2N Soft, France*
Danny De Vleeschauwer, Werner Van Leekwijck, *Bell Labs - Alcatel Lucent, Belgium*
Yannick Lelouedec - *Orange Labs FT, France*

***Abstract*—HTTP-based delivery for Video on Demand (VoD) has been gaining popularity within recent years. Progressive Download over HTTP, typically used in VoD, takes advantage of the widely deployed network caches to release video servers from sending the same content to a high number of users in the same VoD service. However, due to the inherent heterogeneity of user demands, which may result in requesting the same video content in different resolutions or qualities, the caching efficiency is expected to decrease due to a higher variety in requested media files. The use of Scalable Video Coding allows different representations of the same content to be combined in a single file, whose parts, aka layers, are requested sequentially by a user up to the maximum desired quality. In this paper we show the benefits of using Scalable Video Coding to maintain the same set of possible video content representations, while at the same time maximizing the caching efficiency.**

***Index Terms*—Caching, HTTP Progressive Download, Video on Demand, SVC**

I. Introduction

HTTP-based video delivery of Video on Demand has been gaining popularity within recent years. Studies have shown that HTTP/TCP [1] is widely used to stream media to clients even though TCP imposes higher end-to-end delays on the communication [2]. HTTP is not affected by firewall and NAT traversal issues that exist in traditional streaming scenarios which typically rely on RTP over UDP. Further, UDP communication and traffic using ports other than the default HTTP port (port 80) are often blocked by firewalls. As a result, many content providers have resorted to using HTTP transport for media delivery, even though the end-to-end delay is increased and link efficiency is decreased [2] by the use of TCP.

Using HTTP for the file delivery can substantially reduce the load at the video server by re-using existing HTTP cache infrastructures on the Internet, thus relieving not only the server load but also reducing the overall uplink traffic towards the cache.

HTTP-based progressive download requires a file format and preparation which allows file access while the file is still being downloaded. The ISO base media file format [3] supports this using the file format fragment feature. Furthermore, the user selects the file that matches his capabilities as well as the available transmission bit rate constraints. Such signaling is typically out of scope of the actual progressive download scheme, but is an important feature of the VoD service. In this work we assume the presence of such a service feature. After selection, the requested file is typically transferred using the HTTP GET request.

However, viewers of a particular video content may request it in different representations at different bitrates, since user network access characteristics and device capabilities typically differ. The increased variety of files due to the presence of different coding versions may reduce the caching efficiency. In this work, we will evaluate and show the impact of multiple content representations on the caching efficiency. Therefore, we propose the use of Scalable Video Coding (SVC) [4] to address the impact on caching. Using SVC gives users the possibility to select an appropriate representation, while avoiding the decrease of caching efficiency at the same time, since those SVC representations are based on layers of a single SVC file. The AVC file format [5] allows the separation of SVC layers into file subsets. These file subsets are sequentially requested by the different users until the desired quality is reached. Using this approach the statistical dependency between these files should increase the efficiency of caching multiple representations of the same content. In this paper, we show the gain in terms of caching efficiency as well as reduction of the transmission bitrates at the content server and cache feeder links.

The remainder of this paper is organized as follows. Section II explains the main ideas about caching and its relation to the considered VoD service. In section III SVC is introduced as well as the benefits of using SVC for caching efficiency. Sections IV and V describe the simulation carried out and the results, respectively. The paper is concluded in section VI.

II. Caching in Http Progressive Download

A. *Traditional caching*

Figure 1 schematically shows a network over which a video library is offered by a Video on Demand service. The operator of the access network (i.e., the cloud in the figure), offers connectivity to its customers via access links and connects to the Internet (where the content library is offered on an origin server by a third party) over a “transit” link. In that way the customers of the access network operator can access video content, the video clips on the origin server. Usually the transit link is not owned by the access network operator serving the customers and therefore the operator tries to minimize the amount of traffic on that transit link. For that purpose the network operator deploys a proxy and a cache in its network. Indeed (temporarily) storing a video file in the cache and redirecting the user requests for video files, which are stored in that cache, saves traffic over the transit link. Since the cache is usually too small to host the complete video library and the content of the origin video library often changes, the video files that are stored in the cache at every moment need to be carefully selected. This is accomplished by an appropriate caching algorithm.

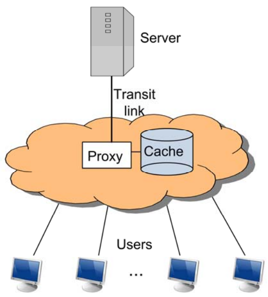

Figure 1: A typical network, hosting a cache, over which content is offered.

There are many different cache replacement algorithms that have been proposed over the last years that optimize the caching performance based on some special criteria, as summarized in [6]. Roughly explained, the choice of which files to cache certainly has to depend on the (measured) file popularity: caching a file X that is requested more frequently than file Y is more beneficial than vice versa (given the same file size), simply because caching file X saves more transmission bitrate than caching file Y at the same storage cost. Thus, some scores will be given to the cached files depending on their popularity, resulting in a ranking, which will determine the cache replacement policies.

The (streaming) video bit rate and file size should have less impact at least if the duration of all videos is about the same (here we assume that all files have a duration T). Storing a file with associated bit rate R (and hence associated file size R·T) saves a transport capacity, which depends on the file popularity and is proportional to R. For files with equal request rates, (i.e., popularities), caching one file with associated bit rate R is exactly the same (in terms of transport capacity decrease and storage cost increase) as caching two files with associated bit rate R/2.

In this paper we perform the decisions of our caching algorithm solely based on the popularity of files. In a dynamic environment (of frequently changing video libraries) the popularity of files is not known, but need to be measured. Based on these popularity estimates, the files are ranked and the ones with highest position in the ranking are cached, until the total caching space is consumed. When the space in the cache is consumed the files with lowest position in the ranking are replaced by new requested files. The ranking (and hence the contents of the cache) is updated at each user request.

We use the well-known Least-Recently Used (LRU) algorithm to rank the video objects in terms of their popularity. Note that although LRU does not explicitly measure the popularity (i.e., request rate), it still results in objects being ranked according to their popularity.

B. Effect of Multiple Representations on the Caching Efficiency

In this paper, we consider the scenario where users may request a certain video clip in one of a possible set of resolutions or quality versions. Hence, each video offered by the origin server must be encoded in a given number (N) of bit rates. These N versions can be encoded separately with AVC and offered side by side, a scenario we refer to as "Multi-Representation VoD (MR-VoD)", or can be embedded in a multi-layer representation which allows for further separation into file subsets (layers) using SVC, a scenario we refer to as "SVC-VoD". We discuss the impact of the former first and comment on the latter in Section III.

Compared to the scenario in which only one version is offered (which we refer to as the "Single-Representation VoD (SR-VoD)" scenario), in the MR-VoD scenario, the requests for a particular video clip are distributed over its N versions.

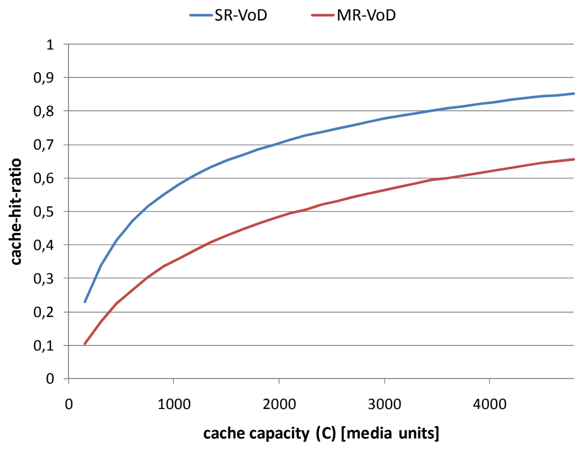

Figure 2: Caching efficiency reduction result of offering a higher variety of representations (e.g. 4) for each file.

If each of the versions associated with a video clip is requested with more or less equal probability, the ranking in the MR-VoD scenario is almost the same as in the scenario with only one version: instead of occurring only once, each video clip occurs N times in that ranking, but with high probability in a block of N consecutive ranks. A consequence of this is that if a certain version of a video is cached it is highly likely that all other versions need to be cached as well. Consequently, in order to attain the same cache-hit-ratio in the MR-VOD scenario as in the SR-VOD scenario, the cache should be able to store all N versions of the video instead of just one. Since storing N versions side by side requires more storage, a larger cache size/capacity is needed to attain the same hit ratio. Conversely, if the same cache capacity is used, a lower cache-hit-ratio results, as illustrated in Figure 2. Note that based on a similar reasoning (and as described in more detail in Section III) the SVC-VoD scenario could attain the same hit ratio with practically the same cache size. The main difference between MR-VoD and SVC-VoD is illustrated in Figure 3. It can be seen that by using SVC much more video clips at different representations can be stored in the cache, while with MR-VoD many files have to be removed from the cache to obtain additional space for the new incoming files or versions of them.

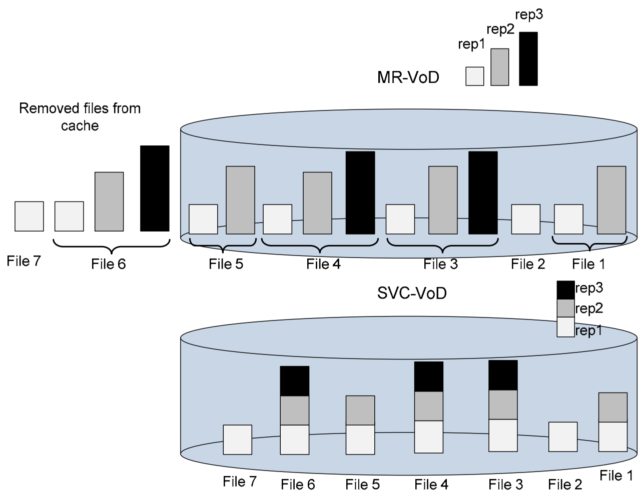

Figure 3: Caching performance comparison for MR-VoD and SVC-VoD

If not all of the N versions are requested with the same probability, the situation is not straightforward. The ranking associated with the MR-VoD scenario cannot simply be derived from the one in the SR-VoD scenario. For the most popular video clips, it will still be the case that if one version is cached, it is likely that all others are in the cache too, but for less popular video clips this will no longer be the case. The MR-VoD scenario will still perform worse (in terms of hit ratio) than the SR-VoD scenario, but the difference will be less pronounced than in the previous case.

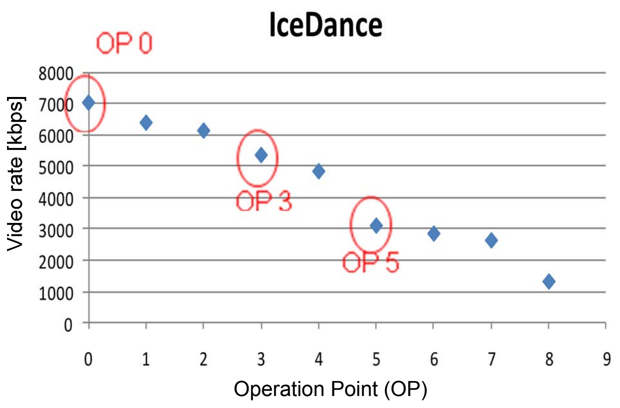

Figure 4: Quality scalable operation points (OPx) of a single SVC stream (IceDance sequence)

III. Scalable Video Coding and Impact on the Caching Efficiency

The scalable extension of H.264/AVC (SVC) [4] allows representing a video content in different spatial, temporal and quality resolutions within the same bit stream, where each SVC quality level is also referred to as a layer. The base layer provides the lowest quality and each additional decoded enhancement layer increases the quality up to a certain level. Due to inter-layer prediction in the coding process, decoding of a certain quality level of an SVC stream requires all dependent layers.

A. Multi-Bitrates with SVC

SVC allows for multiple Operation Points (OP) within the same bit stream. An OP refers to a valid substream at a certain quality level and a corresponding bit rate. One way is to encode multiple quality layers, which gives the flexibility to additionally incorporate different resolutions or support different decoder profiles. However, each integrated quality layer additionally reduces the coding efficiency of the SVC stream to some extent, but the overhead can be kept below 10% as shown in [7]. Another way is to extract multiple OPs from one or two quality layers, which keeps the overall coding overhead within an acceptable range.

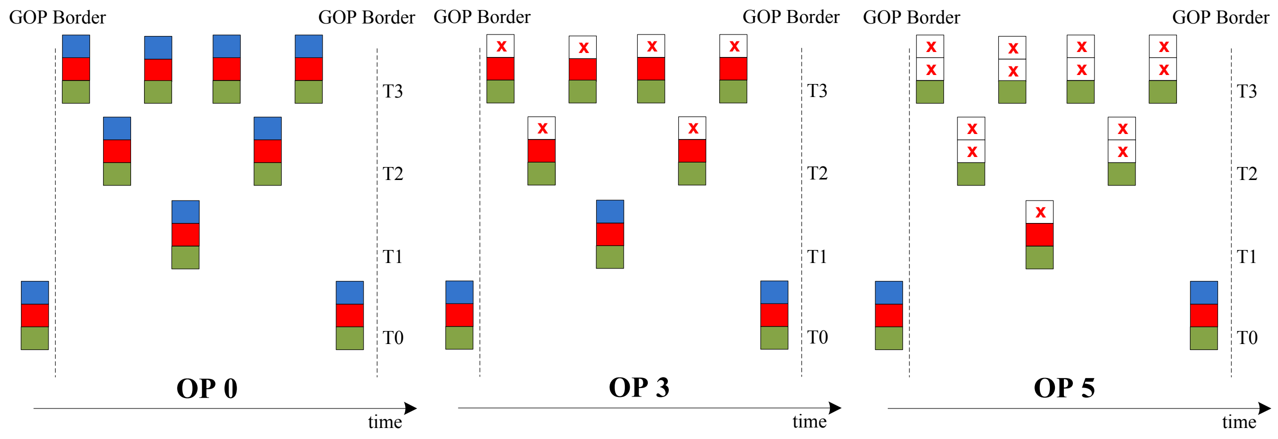

Figure 5: SVC coded pictures of temporal and quality layer combinations for OP 0, 3 and 5.

Figure 5 depicts three exemplary operation points of an SVC stream with a group of pictures (GOP) size of 8 with a hierarchical prediction structure and four temporal layers (T0-T3), which allows for an additional extraction of 4 different frame rates within each layer not taking quality layers into account. The depicted SVC stream is comprised of the base layer (green parts of the rectangle), and 2 quality enhancement layers (red, blue parts of the rectangle) for the coded pictures (rectangles in the figure). Enhancement layer packets can be dropped within a temporal layer, thereby reducing the quality of the affected frames and the overall bit rate. Figure 4 shows the bit rates of 9 OPs for the ITU-T test sequence "IceDance". It can be seen that dropping the quality layer parts Q1 (red) and Q2 (blue) of the coded pictures in temporal levels T2 and T3 reduces the video rate from 7Mbps to 5.3Mbps (OP3). Further, dropping the quality layer Q2 of all temporal levels results in a video rate of 3.1Mbps (OP5). In general, several OPs can be selected for bitrate optimization. However, the OPs must be selected in such a way, that users experience smooth quality degradation. The selection of OPs is out of the scope of this work.

To provide the same number of OPs with single layer coded streams would require encoding 9 independent streams. It is obvious, that in a cache based VoD system, which provides multiple bit rates per content, the use of SVC instead of single layer MR-VoD significantly reduces the amount of data which has to be transmitted within the network and stored within caches. The SVC overhead, resulting from the efficiency penalty of encoding the media in different layers, can be considered negligible in comparison to multiple encoded streams at different video rates, although the actual overhead has been taken into account in the simulations.

B. *Increased caching hit ratio with SVC*

When considering a VoD service with multiple available representations based on layers of SVC, first the amount of data that has to be transmitted to and stored in the cache is significantly reduced compared to the single layer case, and second, more clients request the same data (layers) since clients requesting different representations of a same video clip are expecting to receive a set of layers, where some layers are common for all of those requests, e.g. the base layer. Thus, the HTTP request for a certain quality results in a multiple HTTP request for each of the mentioned layers and all requests for a single content incorporate at least the base layer representation. Consequently, the probability of a cache-hit for files containing the lowest layers of SVC streams, which most of the users are interested in, is increased. The application of SVC not only reduces the load in the network or storage in the caches but also increases the cache hit ratio which further reduces the bit rate to be transmitted from the server to the VoD system.

IV. SIMULATION

A. *Empirical Data*

The simulations are based on real data statistics extracted from a real VoD service. The statistics have been measured within the time period of one month. The provided VoD service offers a wide variety of video clips of more than 5000 files among which the users can make their selection from. In these statistics an average of about 3400 requests per day is reported.

B. *Simulated Environment*

Different simulations have been carried out in order to measure the effectiveness of the proposed SVC-VoD solution. The simulated environment consists of about 8900 users connected to a video server via a unique proxy where the requested content is stored in the cache close to the proxy, as shown in Figure 1.

The cache replacement technique considered here is the Least-Recently Used (LRU) algorithm. The cache capacity (C), i.e. the storage size, is measured in media units and is varied from 150 to 4800 media units with a step equal to 150. The media units are equivalent to the size of a video clip of 90 minutes at 500 Kbps (1 media unit=337.5 MB).

Each of the video clips is offered at four different encoding bitrates: 500 Kbps, 1000 Kbps, 1500 Kbps and 2000 Kbps. Thus, in case of considering videos at multiple representations encoded with AVC (MR-VoD), the correspondent representations have sizes from 1 to 4 media units, whereas in case of considering SVC the layers have a size of 1 media unit for the base layer and 1.1 media units for each of the enhancement layers to take the SVC overhead into account.

Note that within this work only the storage of whole video files or whole video layer files is considered, and not the partial storage of segments of smaller duration than the duration of the original file.

In order to carry out the simulations, the aforementioned empirical data has been used. Thus the requests for video files are based on real life measurements. For a given video clip, available in different representations, a uniform distribution has been assumed for the requests of a particular representation. The total number of requests used for the simulations is 107450.

V. RESULTS

Figure 6 shows the average cache-hit-ratio for the simulation described above; where for the SVC case the average cache-hit-ratio for each of the layers is additionally shown. It has been already mentioned in Section II that increasing the number of possible representations for a given video, the variety of files available to be requested increases and thereby the number of requests for the same file decreases. Therefore, the cache-hit-ratio also decreases.

Since offering a video or file at different video rates is typically necessary due to the heterogeneity of the users, a solution has to be found to minimize the negative impact on the caching efficiency. This is where SVC plays an important role.

In Figure 6, it can clearly be seen how the use of SVC improves the performance of the system in terms of cache-hit-ratio compared to the use of MR-VoD. It is also noticeable that the cache-hit-ratio for the AVC case is even lower than for the highest layer when SVC is used, since the storage capacity at the cache runs out faster with the higher diversity in requested files due to using the MR-VoD approach. Furthermore, the caching performance for the base layer is significantly higher compared to the other files and layers as the number of request for this is higher than for the other layers or different representations when AVC is considered.

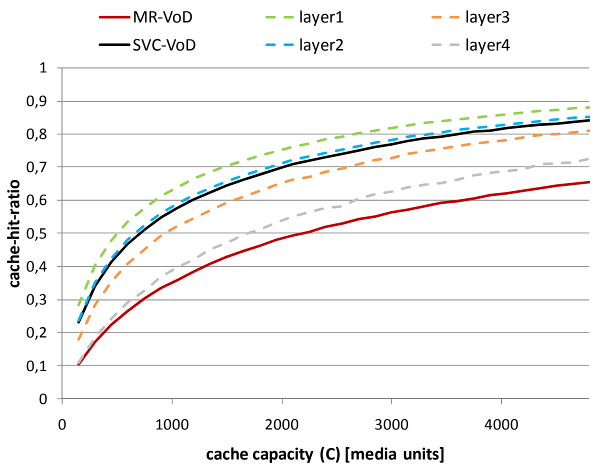

Figure 6: Average cache-hit-ratio for AVC with multiple representations and SVC

As a result of the increased caching efficiency, the average traffic in the "transit" link is significantly reduced, as shown in Figure 7.

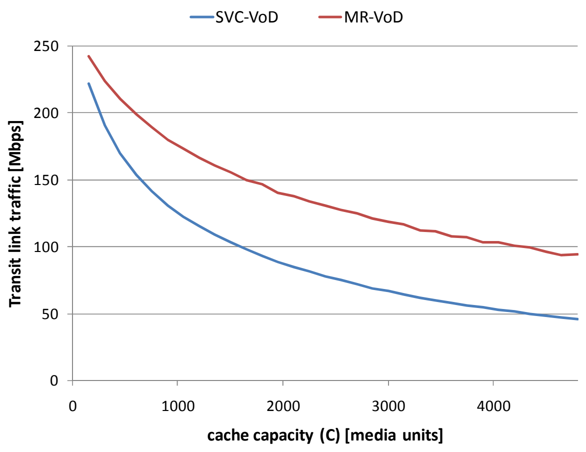

Figure 7: Average traffic through the "transit" link

Figure 8 shows the gain of using SVC in terms of saved throughput in the "transit" link for different values of N (the number of available representations for the video clips). The number of representations available to download for each file is varied from 2 to 8 as shown in the figure. It can be clearly seen that using SVC is a much better solution for VoD with multiple representations, since the cache is used more efficiently. Consequently, the amount of traffic in the transit link is reduced and thereby the costs for the operator providing this service.

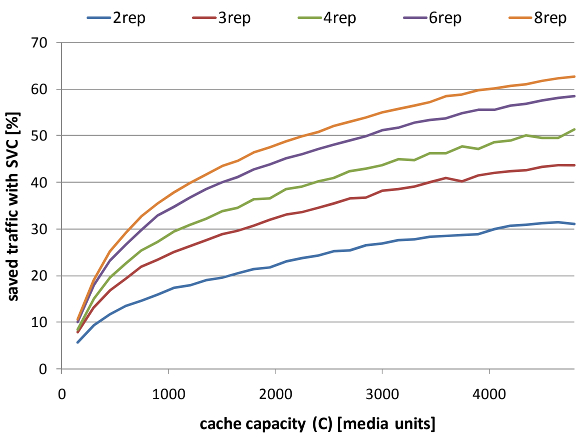

Figure 8: Saved traffic in the transit link with the used of SVC for different number of representations

VI. Conclusion

HTTP Progressive Download is a promising technique for video delivery in VoD services, since first transmission is not affected by firewall and NAT traversal issues that are typical in traditional streaming over UDP, and second network caches contribute to a reduction in the load on the servers and the transit links. However, providing a wide variety of files at different encoding rates, in order to satisfy all users with different capabilities, typically results in a sub-optimal performance of the network caches.

The adoption of SVC as a media codec enhances the efficiency of the network caches in comparison to the use of AVC at multiple encodings, significantly reducing the load on the video server.

Acknowledgment

The research leading to these results has received funding from the European Union's Seventh Framework Programme ([FP7/2007-2013]) under grant agreement n°248775, project OCEAN.

References

[1] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An Analysis of Live Streaming Workloads on the Internet", In Proc. of the 4th ACM SIGCOMM conference on Internet measurement, 2004.

[2] B. Wang, J. Kurose, P. Shenoy, D. Towsley, "Multimedia streaming via TCP: an analytic performance study". In Proc. of ACM International Conference on Multimedia, 2004: 908-915.

[3] ISO/IEC (2008) Information technology — Coding of audio-visual objects — Part 12: ISO base media file format. ISO/IEC 14496-12:2008 (3rd edition)

[4] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," IEEE Transactions on Circuits and Systems for Video Technology, vol. 17, no.9, pp 1103–1120, 2007.

[5] ISO/IEC (2004) Information technology — Coding of audio-visual objects — Part 15: Advanced Video Coding (AVC) file format. ISO/IEC 14496-15:2004.

[6] H. Bahn , K. Koh , S. H. Noh, and S. Lyul Min, "Efficient Replacement of Nonuniform Objects in Web Caches", IEEE Computer magazine, vol. 35 no. 6, p.65-73, June 2002.

[7] H. Schwarz and T. Wiegand, "Further results for an rd-optimized multi-loop SVC encoder", JVT-W071, JVT Meeting San Jose, USA, 2007, ftp://avguest@ftp3.itu.int/jvt-site/2007_04_SanJose/JVT-W071.zip