iDASH: Improved Dynamic Adaptive Streaming over HTTP using Scalable Video Coding

Yago Sánchez, Thomas Schierl, Cornelius Hellge, Thomas Wiegand
Fraunhofer HHI, Germany

Danny De Vleeschauwer, Werner Van Leekwijck
Bell Labs - Alcatel Lucent, Belgium

Dohy Hong
N2N Soft, France

Yannick Le Louédec
Orange-FT, France

ABSTRACT

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 relieve video servers from sending the same content to a high number of users in the same access network. However, due to a sharp increase in the requests at peak hours or due to cross-traffic within the network, congestion may arise in the cache feeder link or access link respectively. Since the connection characteristics may vary over the time, with Dynamic Adaptive Streaming over HTTP (DASH), a technique that has been recently proposed, video clients may dynamically adapt the requested video quality for ongoing video flows, to match their current download rate as good as possible. In this work we show the benefits of using the Scalable Video Coding (SVC) for such a DASH environment.

Categories and Subject Descriptors
C.4 [**performance of systems**]: Modeling techniques.

General Terms
Performance, Experimentation, Verification.

Keywords
HTTP streaming, Video on Demand, Adaptive, SVC.

1. INTRODUCTION

HTTP-Streaming has been gaining popularity in recent years. Contrary to the past tendency of relying on RTP over UDP for multimedia communications due to the higher end-to-end delay imposed by TCP connections, many content providers have resorted to using HTTP transport for media delivery when the delay constraints allow it. In fact, [1] shows that HTTP/TCP is widely used to stream media to clients. The main reasons for that are first that HTTP is not affected by firewall and NAT traversal issues that exist in traditional streaming scenarios which typically rely on RTP over UDP and second that using HTTP for the file delivery can substantially relieve the load on the video server by re-using existing HTTP cache infrastructures on the Internet, therefore reducing the overall traffic at the cache feeder link.

Some additional evidence of the increasing interest of the market in HTTP-Streaming is the standardization processes lead by the standardization organization IETF [2], 3GPP [3], OIPF [4] and MPEG [5].

Dynamic Adaptive Streaming over HTTP (DASH) as defined in [5] refers to a video transport methodology where the clients adapt their requests based on some estimates of their available download rate at every time instant of the streaming service. For DASH a video is offered in a various (typically 4 to 10) versions. Each terminal can choose which version to download depending on its capabilities and the network congestion level. This choice is not only made at the beginning of the flow, but at frequently dispersed time instant during the streaming of the video, at which the DASH client can switch from one version to another (for example to alleviate the onset of congestion). Typically this is achieved by segmenting each version in chunks such that the segment boundaries are aligned in time. All the DASH clients need to do is to consecutively download the most appropriate chunks, based on the information obtained by monitoring recently downloaded chunks of the ongoing movie.

One possibility to provide DASH is to encode multiple representations of each of the videos with H.264/AVC [6] at the server and offer them side-by-side. Another is offering all these representations embedded in one file via Scalable Video Coding (SVC) [7]. Offering all these representations side-by-side does not only put a high burden on the storage requirements at the origin server, but might also result in a decrease in cache performance in comparison to SVC. In this paper we discuss the potential gain of using SVC to offer the different versions. Furthermore, as explained in [8], due to its layered nature, SVC provides flexibility to DASH, since it allows dividing media content both per SVC layers and per time intervals, and thus prioritizing very accurately the different elements of the media content according to their importance. Therefore, a higher responsiveness and better playback quality under adverse network conditions is obtained since a request for a time interval is diluted into multiple requests (HTTP_GET requests) performed subsequently, one for each of the layers, and when congestion is detected, requests for higher layers may be omitted. Conversely, for the AVC case a unique request is issued for the whole data of a given interval, having to wait longer for the requested data to be downloaded until witching to a lower representation can be performed.

This work describes the effects of congestion in the network, as well as the effects of caching multiple representations of the same content in DASH systems. Indeed the dynamic adaptation leads to a situation where the clients may request a different representation of a same video content. This paper aims to show the benefits of using SVC over AVC encoded streams in this respect. We will evaluate and show the improvement on the caching efficiency, which is a key factor for reducing the amount of transported data from the server, due to the use of SVC in an environment with requests for multiple content representations as a consequence of different client connectivity situations due to congestion.

The remainder of this paper is organized as follows. Section 2 summarizes some previous work. In section 3 SVC is introduced as well as how different representations can be obtained in order to preserve an efficient encoding. Sections 4 and 5 describe the simulation carried out and the obtained results, respectively. The paper is concluded in section 6.

2. RELATED WORK

There has been some previous work on the topic of combining caching and layered video codec, as [9] or [10], where the benefits of caching layered video codec have been shown. These previous works focused on a service where users selected a representation among a variety of possibilities based on their equipment capabilities.

Both studies focus on a system as the one depicted in Figure 1. This figure schematically shows a network over which a video library is offered by a Video on Demand (VoD) service. The operator of the access network (i.e., the cloud in the figure), offers connectivity to its customers via access links (e.g., DSL-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 this paper referred to as the cache feeder link. In that way the customers of the access network operator can access video content, in particular the movies on the origin server. The network operator deploys a proxy and a cache in its network to minimize the amount of transmitted data through the "transit" link relieving the server of having to send an extremely high amount of video data. Since the cache is usually too small to host the complete video library and the content library on the origin video server 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.

In [10] the advantage of using SVC for better usage of the cache resources/capability is shown. In this work the improvement on cache-hit-ratio as well as traffic within the cache feeder link are presented. In fact, it is shown that storing different representations of the same video side-by-side with independently encoded streams with AVC leads to a spoilage of the cache capacity, while showing that doing it similarly with layer of SVC lead to a much better solution.

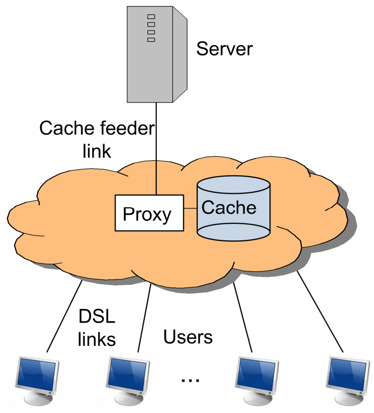

Figure 1: DASH network environment

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 [11][12]. Most algorithms make decisions based either on how recently an object has been requested (e.g., Least Recently Used - LRU) or on how frequently an object has been requested (e.g., Least Frequently Used - LFU) over a time period or a combination thereof. In [13] the chunk-based delivery (video files downloaded in smaller parts thereof, i.e. chunks/segments) is exploited in a caching context. In this work the chunks that will be consumed in a near future with a high probability are predicted, assuming that it is very likely that a user playing chunk *n* of a given video file at the current moment will play chunk *n*+*k* of the same video file *k* time instants later.

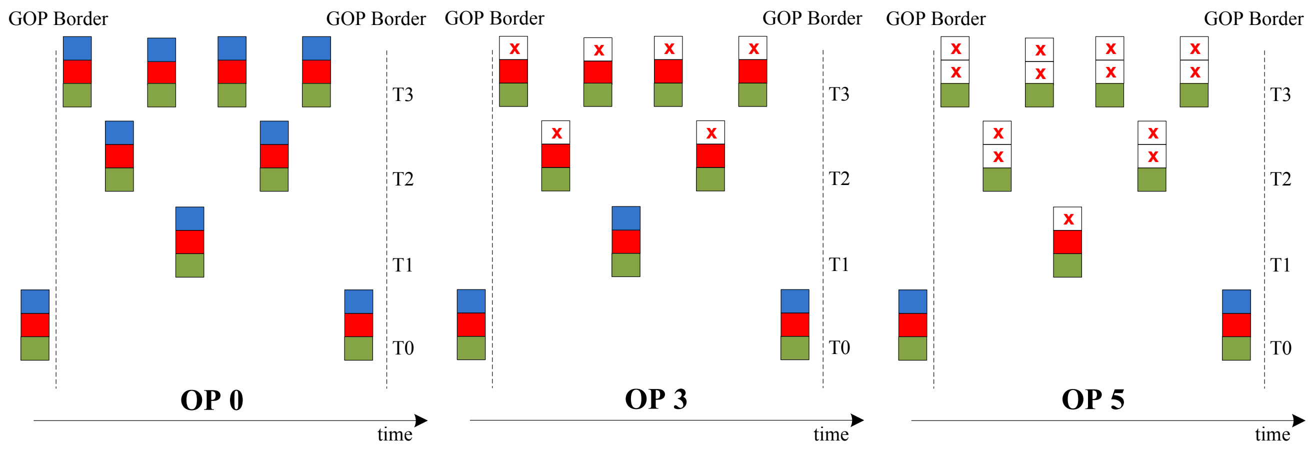

Figure 2: SVC coded pictures of temporal and quality layer combinations for OP 0, 3 and 5.

3. EFFICIENT ENCODING WITH SVC

The scalable extension of H.264/AVC (SVC) [7] provides features to represent different representations of the same video within the same bit stream by selecting a valid sub-stream. SVC is divided into layers, which correspond to different quality, spatial or temporal representations. It is composed by a base layer, which corresponds to the lowest representation, and one or more enhancement layers, which increase the quality, spatial and/or temporal representation when added to the base layer.

The main advantages of SVC are the possibility of serving a great number of users with different equipment capabilities with a single bit stream and the fact that it facilitates coping with congestion by applying on-the-fly adaptation, performed by adding or subtracting layers to match the capabilities of the network at every time instant.

SVC allows for multiple Operation Points (OP) within the same bit stream. An OP refers to a valid sub-stream at a certain quality level and a corresponding bit rate. In this work we only focus on quality scalability since we considered that all users have the same equipment capabilities and we take advantage of SVC in order to cope with congestion.

One way is to encode multiple quality layers either with Coarse-Grain Scalability (CGS) or Medium-Grain Scalability (MGS), and select each layer as an OP. Each integrated quality layer additionally reduces the coding efficiency of the SVC stream to some extent. However, if addressed properly the overhead can be kept below 10 % as shown in [14].

CGS and MGS differ basically in the fact that for the former the stream switching (from one OP to another OP) can only be performed on a layer basis, while the later has more flexibility. Hence, for CGS, the number of OPs is the same as the number of layers. However, when MGS is considered switching can be perform on a NAL unit basis, leading to a much higher number of possible OPs. As a result MGS can have more OP at less overhead than CGS.

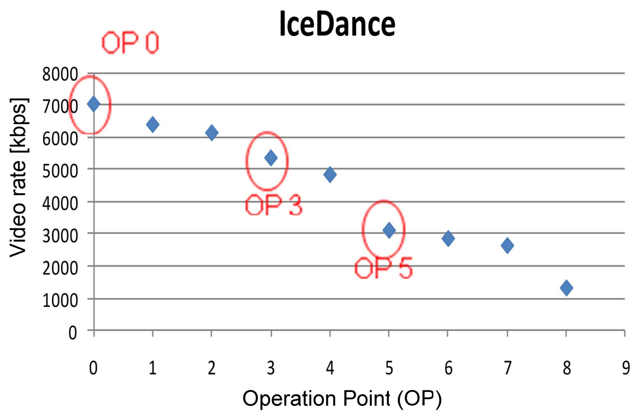

Figure 3: Operations points

Consequently, foreseeing that having so many layers as the number of desired OPs results in an increase in coding overhead, in order to achieve multiple OPs we encode a reduced number of quality layers with MGS and select a sub-stream of the original data not only selecting whole layers but an smaller part of them. This allows keeping the overall coding overhead within an acceptable range. Figure 2 illustrates how to obtain different OPs in this way. In the example, the SVC stream is comprised of the base layer (green parts of the rectangle), and two quality enhancement layers (red, blue parts of the rectangle) for the coded pictures (rectangles in the figure). The different OPs are obtained by dropping enhancement layer packets from the highest temporal levels. Figure 3 shows the bit rates of 9 OPs for the ITU-T test sequence “IceDance”. It can be seen that dropping the quality layer parts 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 and pictures from Q1 (red) of temporal levels T2 and T3 results in a video rate of 3.1Mbps (OP5). In general, several OPs can be selected for bit rate optimization. In Figure 3, we show different PSNR/bitrate combinations corresponding to the selection of different OPs. 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.

4. MODEL

4.1 User demand and network set up

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

Figure 4 shows more in detail the pattern of the requests during 31 days. It shows the number of request aggregated over one hour for each hour of the period of 31 days. Clear diurnal and weekly patterns can be observed. We define “busy hours” or peak hours as hours where the number of requests per hours is larger than a certain threshold.

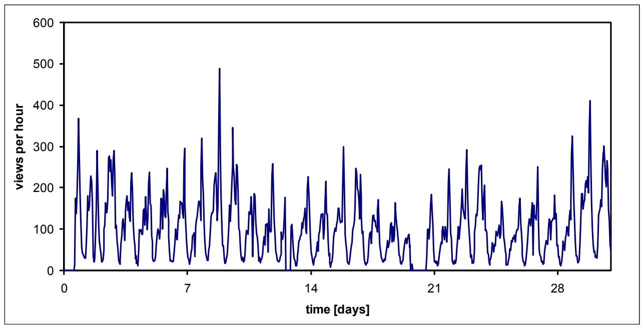

Figure 4: Requests statistics

The requests extracted from the real data are distributed among the users connected to the service, and congestion is simulated as described below in section 4.3. These users are connected over access links (e.g., DSL-links) to a proxy hosting a cache. The origin server is connected to the proxy hosting the cache by a transit link. The latter link is referred to in this paper as the cache feeder link. We consider two cases: only congestion on the access links and only congestion on the cache feeder link. The case where both the access links and the feeder link are simultaneously congested is outside the scope of the paper, since even though it could be easily handled and simulated the aim of this paper is to show the effects of congestion separately in cache feeder and access links for clarity. Both cases are detailed below.

4.2 Caching algorithms

The performance of the cache is here analyzed for two different caching algorithms (operating on chunks):

- LRU: where the most recently requested chunks are kept in the cache.
- CC: An algorithm described in [13] that takes into account the number of guaranteed hits of chunks (if the DASH client keeps on selecting the same version as it currently does), which uses an improved movie content scoring algorithm that combines the LRU and LFU basics.

In case of considering SVC there are *n* chunks per time interval, where *n* corresponds to the number of layers. In other words, the layers are transmitted and stored in the cache separately and therefore count as different objects for the cache-hit-ratio evaluation. In case of offering the *n* version side by side via AVC, each time interval has *n* independent versions, in the sense that if one version is cached and another is requested no cache hit can be counted.

4.3 Congestion Control

Clients (on the same access network) of a multimedia service typically share (transport and caching) resources with other multimedia clients and/or users downloading any type of data from the Internet, which produces some cross-traffic in the network causing congestion. This results in a temporarily reduced available download rate for the clients of the service.

These clients (DASH-clients) detect these variations in the connection rate available to them and adapt the bit rate at which they download their ongoing video stream, by requesting the following chunks/segments in an appropriate version. Therefore, every time a user requests a new chunk of a video an additional decision has to be made with respect to which version it will download. This choice depends on:

- the capability of the terminal of the user.
- the congestion state of the link between the server and the cache (feeder link) and congestion state between the cache and the end user (i.e., the access link (e.g., DSL-link)). If requesting the version that a user wants to download would congest the link, this request is downgraded as many times as needed to alleviate congestion.

The main contribution in this paper is to analyze the performance of the cache and transmitted rate, as well as the quality at the DASH-clients under different congestion situations within the network that are described below.

4.3.1 Modeling Congestion at the cache feeder link

Since the cache is not large enough to host the complete content library (and this content library regularly introduces new content) there is still considerable traffic on the feeder link towards the cache. Moreover, Figure 4 shows that the traffic demand is several times more important during peak hours than during off-peak hours. Therefore, the traffic on the cache feeder link due to cache misses and caches updates is expected to be higher during peak hours. Consequently, even though cache infrastructures are placed in the network to relieve the load on the server and reduce the overall traffic at the cache feeder link, this may not be sufficient at peak hours. A huge amount of data might be transmitted across the cache feeder link, resulting in congestion within this link.

This would disrupt the streaming service if nothing were done. Therefore, the DASH video clients of the users detect congestion, by e.g. noticing that the requested data is received with an additional delay, they adapt their requests, and this results in a lot of users switching to a lower quality, i.e., requesting the next chunk inline in a lower quality. In this paper the downgrading of the requests is done such that the congestion on the feeder link is alleviated.

If congestion disappears, the DASH video clients will try to grab more capacity, i.e., they will request future chunks in a higher quality. In this paper, the quality adaptation policy for switching-up is performed as follows. A waiting timer is set after switching to a lower quality. If during a time interval equal to this timer, no congestion is detected, DASH-clients switch to a higher quality. If this leads to congestion the quality is quickly downgraded again. If not, the clients keep on downloading the quality they have switched to.

The above behavior will approximate the behavior of real DASH clients very closely. We opted for simulating this coarse behavior without actually modeling a detailed DASH algorithm in each client for two reasons. First, DASH client algorithms are not standardized. Some implementations are known, and these aim for a behavior as we described above. Second, in this paper we are not interested in the time scales at which DASH clients see the packets trickling in, but we concentrate on the time scale at which cache decisions are made. Not simulating at the finest time scale allows us to simulate a month's worth of world time in a reasonable amount of computational time (simulations run typically for a couple of hours), but introduces some errors, which we argue, will be negligible.

4.3.2 Modeling congestion in the access links

In this subsection we consider congestion on the access link. On this access link other services run (i.e., a user may be downloading a large file, may be browsing the web, etc.) beside the DASH video streaming. This type of congestion can occur any time of the day and is not necessarily restricted to peak hours. The model for this type of congestion that we have simulated in this paper is shown in Figure 5. This figure illustrates a Markov-chain with four states corresponding to four possible download rates and selected OPs.

In fact we assume that the cross traffic on the access link (e.g., DSL link) which is the result of sharing this link with one or more DASH clients or any other client requesting data from the Internet is such that the DASH client requesting the version in the next slot, can be described by a Markov chain.

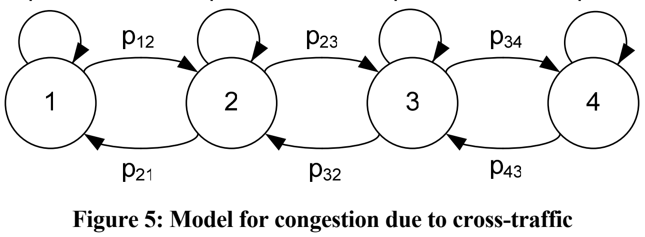
$$\begin{tikzcd}
1 \arrow[loop above, "p_{11}"] \arrow[r, bend left, "p_{12}"] & 2 \arrow[loop above, "p_{22}"] \arrow[l, bend left, "p_{21}"] \arrow[r, bend left, "p_{23}"] & 3 \arrow[loop above, "p_{33}"] \arrow[l, bend left, "p_{32}"] \arrow[r, bend left, "p_{34}"] & 4 \arrow[loop above, "p_{44}"] \arrow[l, bend left, "p_{43}"]
\end{tikzcd}$$

Figure 5: Model for congestion due to cross-traffic

As seen in Figure 5 this Markov chain consists of four states where the transition probabilities p_{ij} of the transition matrix $P=[p_{ij}]$ with

$|j-i|>1$ are set to zero, i.e. it is only possible to go from a state to its neighbor sates. The rest of the parameters (represented in the figure) have been set to values that lead to realistic situations.

The most important parameters to take into account to consider whether the selected values correspond to a realistic situation or not are the mean state sojourn time (mean duration of being in a state: $E[t_i]$) and average percentage of time in each of the states (p_i), which can be derived easily from the transition probabilities, as shown in Eq.(1) and Eq.(2).

$$E[t_i] = \sum_{t_i=0}^{\infty} (t_i+1) * p_{ii}^{t_i} * (1-p_{ii}) = \frac{1}{1-p_{ii}} \qquad (1)$$

$$p_i : \pi * P = \pi \qquad (2)$$

where, $\pi=\{p_1,p_2,p_3,p_4\}$is the left eigenvector of P (associated with eigenvalue 1), a.k.a. steady state vector, which fulfils

$$\sum_{i=1}^{4} p_i = 1 \qquad (3)$$

The simulation time step in the presented Markov-chain model corresponds to the selected chunk size, since the adaptation is performed by the DASH clients on a chunk basis.

4.4 Performance targets

In order to compare the system where the different version of a video are offered encoded in AVC side by side with the system in which the versions are embedded in one SVC stream, we consider three performance targets:

- The cache-hit-ratio: calculated on a chunk basis, or when SVC is considered on smaller objects, corresponding to each of the layers of each of the chunks. It represents the percentage of these objects that can be served from the cache and do not need to be transported over the (possibly congested) cache feeder link.
- Not downgrading clients: percentage of users that do not need to downgrade the requested quality for the video session, i.e. percentage of users that received the maximum quality for the whole service.
- D/R: this introduced measure represents the relation between the desired (D) rate and the received (R) video rate. High values (close to 100 %) correspond to users receiving the highest possible quality, while low values correspond to users receiving a much lower quality than that which would be desirable.

These parameters are either averaged over the complete duration or over the "peak/busy hours". Note that all these performance targets are defined in that way that higher values correspond to a better quality of service.

5. Results

The results presented in the following show the performance of the system comparing both multiple representations encoded with AVC (MR-VoD) offered side-by-side and multiple representations encoded with SVC (SVC-VoD). The rate distribution for the different video representations is summarized in the table:

Table 1: Rate distribution for the video representations

Codec	Rep. 1	Rep.2	Rep.3	Rep. 4
AVC	500 kbps	1000 kbps	1500 kbps	2000 kbps
SVC	500 kbps	1066 kbps	1633 kbps	2200 kbps

As aforementioned, in the performed simulations we studied separately two main cases for congestion: 1) DASH clients requesting too many data in peak hours (resulting in congestion in the cache feeder link) and 2) cross-traffic due to DASH clients sharing resources with other user downloading any type of data (resulting in congestion in the access link).

The videos that are considered for the following simulations have a duration of 90 minutes and are divided in chunks of 10 seconds length.

5.1 Congestion in the cache feeder link

We have simulated a cache feeder link with a bit rate capacity of 250 Mbps and have run the simulation with the data extracted from the deployed real VoD system. Whenever the needed data from the server exceeds the capacity of the cache feeder link, congestion arises, which results in many users adapting the video streaming quality. Table 2 and Table 3 summarize the performance observed in this simulation.

Table 2: Cache-hit-ratio for congestion in cache feeder link

Cache capacity (media units)	LRU		CC	
	AVC	SVC	AVC	SVC
500	52.8 %	57.0 %	59.9 %	64.3 %
1000	64.0 %	66.2 %	67.0 %	70.2 %
2000	73.8 %	73.9 %	74.1 %	75.8 %

The cache capacity is measured in media units, which are equivalent to the size of a video clip of 90 minutes at 500 kbps (1 media unit=337.5 MB).

The results shown in Table 2summarize the cache-hit-ratio for different values of cache capacity and the two different caching algorithms. It can be seen that there is a slight difference between the use of AVC and SVC, where the adoption of SVC as codec results in a small gain in terms of cache-hit-ratio in comparison with AVC. Furthermore, the CC algorithm outperforms the LRU algorithm by up to almost a 7 % for low cache capacity values (similar to what was observed in [13]).

Table 3: Percentage of users not downgrading quality

Cache capacity (media units)	LRU		CC	
	AVC	SVC	AVC	SVC
500	26.7 %	26.1 %	45.4 %	48.0 %
1000	47.5 %	43.5 %	58.9 %	61.0 %
2000	70.3 %	63.7 %	74.4 %	74.0 %

The results in Table 3 show that SVC and AVC perform similarly, although MR-VoD results in an almost equal but lower percentage of users downgrading their quality. This is the result of the slightly lower coding efficiency of SVC. Since we are considering that all users are requesting the same quality, this negative effect appears. However in the conclusion we will discuss what would happen in the reality where users may request for different representations due to congestion also in the access link or different equipment capabilities.

Table 4:Avg. D/R over the whole VoD service

Cache capacity (media units)	LRU		CC	
	AVC	SVC	AVC	SVC
500	63.6%	68.6 %	72.2 %	77.4 %
1000	76.3 %	79.5 %	81.0 %	84.8 %
2000	88.6 %	89.2 %	89.9 %	91.8 %

Table 4 shows the results of the obtained average D/R over the whole simulation time. The results for SVC are better than for AVC for both LRU and CC caching algorithms. The values are also higher for the presented CC algorithm than for LRU. The gain for SVC comes from the fact that switching to a higher quality only results in having to request the additional layers from the servers while re-utilizing the already cache video content (lower layers). However, when intended to perform that with MR-VoD, it is necessary to download the complete video data directly from the server, ignoring the lower representations of the video stored in the cache, resulting in a suboptimal usage of the available network resources.

Table 5: Avg. D/R over the peak hours (congested feeder link)

Cache capacity (media units)	LRU		CC	
	AVC	SVC	AVC	SVC
500	57.8 %	64.8 %	66.5 %	73.2 %
1000	69.3 %	75.1 %	74.8 %	79.9 %
2000	81.4 %	84.6 %	83.4 %	87.0 %

In Table 5 similar results as for Table 4 are presented but only focusing on the period of time when congestion happens. It can be seen how the D/R values are lower than the ones presented in Table 4, which is logical, but the results are coherent with the ones of Table 4. SVC-VoD outperforms MR-VoD and the usage of CC results in a better performance.

5.2 Congestion in the DSL-links

The results shown in Table 6 correspond to the case where the bottleneck is the access link, as a consequence of some cross-traffic produced by other users on the Internet. The transition probabilities can be found in the following transition matrix.

$$P = \begin{bmatrix} 0.996 & 0.004 & 0 & 0 \\ 0.004 & 0.992 & 0.004 & 0 \\ 0 & 0.004 & 0.992 & 0.004 \\ 0 & 0 & 0.004 & 0.996 \end{bmatrix}$$

The shown transition probabilities correspond to an $E[t_i]$ of approximately 40 minutes (for the selected chunk length of 10 s) and DASH clients that spend on average an equal percentage of time in each state of 25%.

In order to show the D/R value as done for the previous simulations, with congestion in the cache feeder link, we can compute this value easily based on the π vector. In fact, the components of this vector correspond to the average time receiving a video rate of 25 %; 50 %, 75 % and 100 %. Eq. (4) shows how to compute this value by multiplying the π with the column vector [0.25 0.5 0.75 1] that corresponds to the receiving video rate:

$$D/R = \pi * \begin{bmatrix} 0.25 & 0.5 & 0.75 & 1 \end{bmatrix}^T \quad (4)$$

For the presented transition matrix the D/R value is equal to 0.625 for both MR-VoD and SVC-VoD.

Table 6: Cache-hit-ratio for congestion in access links

Cache capacity (media units)	LRU		CC	
	AVC	SVC	AVC	SVC
500	30.9 %	45.6 %	42.9 %	56.6 %
1000	42.1 %	58.2 %	52.0 %	64.5 %
2000	54.6%	69.0%	61.5%	72.0%

It can be seen that these results show much higher differences between the use of AVC and SVC, since different versions of the requested videos are stored in the cache and this leads to a spoilage of the available storing capacity of the cache when a single layer codec is considered, whereas when SVC is used the available resources are much more efficiently used. A similar effect was observed in [10]. Furthermore, the hit-ratio increases due to the fact that many users make requests for the same data since, even though they may be interested in different version of the same video, their requests are split into multiple request, one associated with each layer that they are requesting. Since the layers built on top of each other, a user requesting layer *k*, needs to request layer 1 to *k*-1 too. In particular the base layer is requested by everyone.

Since the difference between both AVC and SVC are more disparate for this case we have conducted the simulations for a higher range of values for cache capacity (C) only focusing on the LRU caching algorithm, leading to the results shown in Figure 6.

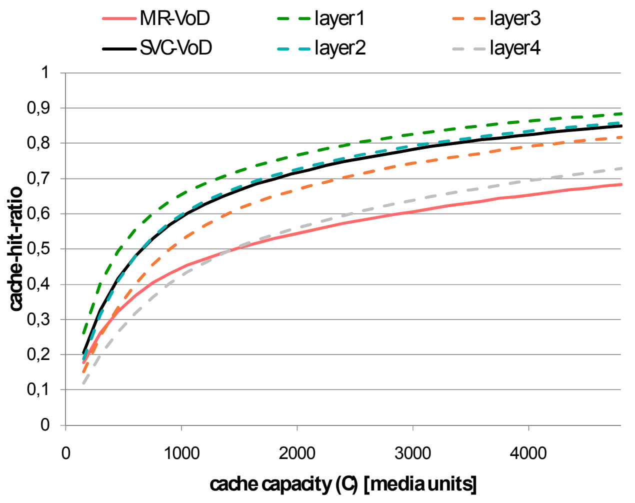

Figure 6: Congestion due to heavy cross traffic

In this figure, the cache-hit-ratio over cache capacity is depicted. 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 (layer 4) when SVC is used almost for all cache capacity values, 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 7 shows the results for a different set up of the simulation, for a situation with less heavy cross traffic, resulting in the DASH client residing in the highest state (4) more often. In this case the percentage of time in each state is unequal with p={9.1%, 9.5%, 19.1%, 62.3%}, as well as the mean state sojourn time $E[t_i]$=(approx.){2min, 2min, 10min, 40min}, which may be closer to that which may happen in the reality. The correspondent transition matrix is shown below:

$$P = \begin{bmatrix} 0.9 & 0.1 & 0 & 0 \\ 0.096 & 0.9 & 0.004 & 0 \\ 0 & 0.002 & 0.985 & 0.013 \\ 0 & 0 & 0.004 & 0.996 \end{bmatrix}$$

In this case, for the selected transition matrix the average D/R expected for the DASH-clients is equal to 0.84, noticeably higher than in the case before, which is expected for an environment with lighter cross-traffic.

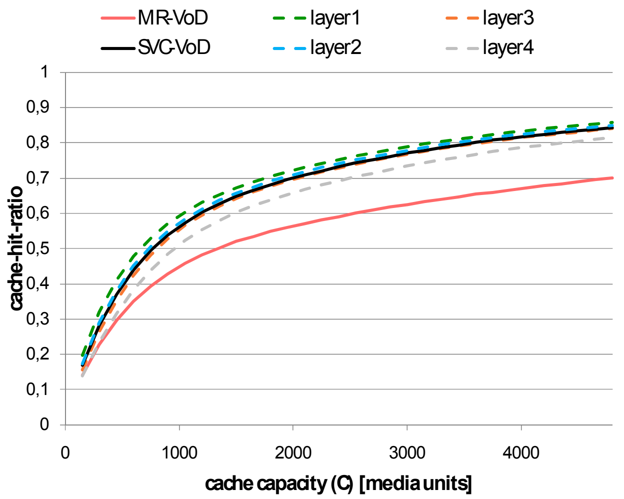

Figure 7: Congestion due to light cross traffic

Although the variety of versions requested for this set up is supposed to be lower than in the case before, the gains of SVC-VoD compared to MR-VoD are still noticeable. Due to this reduced variability the MR-VoD performs slightly better than before but still quite poorly when compared to SVC-VoD. It can be also clearly seen how the cache-hit-ratio for the base layer is reduced (layer 1) and the cache-hit-ratio for the highest layer is increased (layer 4). If we keep on reducing the congestion all lines would converge. An important interpretation of these results is that even though the network is not heavily congested, SVC-VoD outperforms MR-VoD and to make this effect negligible there should not be congestion in the access links, which is far away from the reality.

6. Conclusions

DASH 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 be able to cope with congestion within the network may result in a suboptimal performance of the network caches, especially if the congestion arises in the access links next to the clients, since they may request same videos but at different representations.

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. This would counter the effect of the low overhead of SVC that lead to a higher number of clients having to downgrade to a lower video quality, when the cache feeder link was considered as the single bottleneck. If both congestion cases (at cache feeder link and access links) are considered simultaneously, as would be expected in the reality, or clients with multiple equipment characteristics are considered, the worse performance of the cache with AVC would lead to more traffic within the cache feeder link, contributing to congestion faster than when SVC-VoD is considered leading to higher number of users downgrading quality. Thus, using SVC would lead a better experience for the users and a higher number of satisfied users.

7. ACKNOWLEDGMENTS

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.

8. 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] R. Pantos, and W. May, "HTTP Live Streaming" Draft V04, IETF, June 5, 2010, http://tools.ietf.org/html/draft-pantos-http-live-streaming-04.

[3] 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Transparent end-to-end Packet-switched Streaming Service (PSS); Protocols and codecs (Release 9); 3GPP TS 26.234 V9.3.0 (2010-06), Section 12: Adaptive HTTP Streaming.

[4] Open IPTV Forum – Release 2 Specification, HTTP Adaptive Streaming, Draft V0.06 - June 7, 2010.

[5] ISO/IEC JTC 1/SC 29/WG 11 (MPEG), "Dynamic adaptive streaming over HTTP", w11578, CD 23001-6, Guangzhou, China, Oct 2010.

[6] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE

Trans. Circuits Syst.Video Technol., vol. 13, no. 7, pp. 560–576, Jul. 2003.

[7] 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.

[8] T. Schierl, Y. Sanchez, R. Globisch, C. Hellge, and T. Wiegand, “Priority-based Media Delivery using SVC with RTP and HTTP streaming”, Multimedia Tools and Application, 2010

[9] F. Hartanto, J. Kangasharju, M. Reisslein, and K. Ross, “Caching video objects: layers vs. versions?”, Multimedia Tools and Applications, vol. 1, n. 2, 221-245, 2006.

[10] Y. Sánchez, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. De Vleeschauwer, W. Van Leekwijck, Y. Le Louedec. “Improved caching for HTTP-based Video on Demand using Scalable Video Coding”, Proceedings of the 8th Annual IEEE Consumer Communications and Networking Conference - Special Session IPTV and Multimedia CDN - (CCNC'2011 - SS IPTV)”, Las Vegas (NV), January 9-12, 2011.

[11] 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.

[12] S. Podlipnig, L. Böszörményi, “A survey of Web cache replacement strategies”, ACM Computing Surveys, vol. 35 (2003), pp. 331-373, 2003.

[13] D. Hong, D. De Vleeschauwer, F. Baccelli, “A Chunk-based Caching Algorithm for Streaming Video”, Proceedings of the 4th Workshop on Network Control and Optimization, Ghent (Belgium), November 29 – December 1, 2010.

[14] 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.