

Dynamic Adaptive Streaming over HTTP Dataset

Stefan Lederer, Christopher Müller, and Christian Timmerer

Alpen-Adria-Universität Klagenfurt

Universitätsstraße 65-67

9020 Klagenfurt am Wörthersee, Austria

+43 (0) 463 2700 3600

{firstname.lastname}@itec.uni-klu.ac.at

ABSTRACT

The delivery of audio-visual content over the Hypertext Transfer Protocol (HTTP) got lot of attention in recent years and with dynamic adaptive streaming over HTTP (DASH) a standard is now available. Many papers cover this topic and present their research results, but unfortunately all of them use their own private dataset which – in most cases – is not publicly available. Hence, it is difficult to compare, e.g., adaptation algorithms in an objective way due to the lack of a common dataset which shall be used as basis for such experiments. In this paper, we present our DASH dataset including our DASHEncoder, an open source DASH content generation tool. We also provide basic evaluations of the different segment lengths, the influence of HTTP server settings, and, in this context, we show some of the advantages as well as problems of shorter segment lengths.

Categories and Subject Descriptors

D.5.1 [Multimedia Information System]: Video

General Terms

Algorithms, Measurement, Standardization, Documentation.

Keywords

Dynamic Adaptive Streaming over HTTP, DASH, Dataset, Encoder, Content Generation Tool

1. INTRODUCTION

Online video portals like YouTube or Netflix deploy progressive download or adaptive video on demand systems and count millions of users watching their content every day. Real-time entertainment produces already nearly 50% of the U.S. peak traffic nowadays, with almost 30% of the total U.S. downstream traffic produced by Netflix [1]. Due to these enormous infrastructure and bandwidth needs those costs became a huge problem for content service providers (CSP).

HTTP streaming and especially adaptive HTTP streaming got much attention in the past few years. Proprietary systems like Microsofts' Smooth Streaming [2], Adobes' HTTP Dynamic Streaming [3] or Apples' HTTP Live Streaming [4] provide similar approaches by leveraging existing content delivery network (CDN) and proxy cache infrastructures which costs are significant lower than dedicated streaming server like the Flash Media Server or other competing products.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MMSys'12, February 22-24, 2012, Chapel Hill, North Carolina, USA.
Copyright 2012 ACM 978-1-4503-1131-1/12/02...\$10.00.

But leveraging existing and cheap HTTP infrastructures is not the only reason for adaptive HTTP streaming. Due to the increasing availability of mobile high speed Internet connections like WLAN/3G/4G and the huge smartphone boom in the recent years, mobile video streaming is becoming more and more important. As shown in Figure 1, the annual growth of mobile video will be more than 100% over the next years until a total share of up to 66% of the total mobile traffic in 2014. However, basic progressive HTTP download/streaming is not suitable for these mobile environments with considerable high bandwidth fluctuations. The video stream has to adapt to the varying bandwidth capabilities in order to deliver the user a continuous video stream without stalls at the best possible quality for the moment, which is achieved by dynamic adaptive streaming over HTTP [5][6].

In this context, the emerging ISO/IEC MPEG standard Dynamic Adaptive Streaming over HTTP (DASH) [7] may get a lot of attention in the upcoming years and probably will become an important as well as company-independent adaptive HTTP streaming standard for all types of devices like PCs, Smartphones, Set-Top Boxes, etc. Hence, it is important to provide easy content generation tools and test content for researchers and developers. Unfortunately, there is currently no common test content dedicated to DASH available, showing the basic possibilities of the standard. In this paper we present our easy-to-use DASH content generation tool called DASHEncoder, which enables the user to encode and multiplex DASH-compliant content in segmented and continuous file structure with the associated Media Presentation Description (MPD). Furthermore, we provide a DASH dataset comprising full-length sequences of various genres encoded at different representations in terms of bitrate, resolution, and quality.

The remainder of this paper is organized as follows. Section 2 describes related work. The content generation is described in Section 3 and Section 4 provides details about the actual dataset. In Section 5 we use the dataset to evaluate the influence of the segment size and various Web server configurations to the streaming performance. The paper concludes with Section 6 including future work.

2. Related Work

A detailed evaluation of the media formats of current adaptive streaming systems and a low overhead container format for adaptive streaming is described in [8]. [9] analyzes the quality improvements when using adaptive HTTP streaming over multiple heterogeneous network interfaces.

Microsoft Smooth Streaming is presented in [10] with two test sequences of about 10 minutes of the open source movie Big Buck Bunny. They offer the clip in different resolutions up to 720p and 1080p respectively, with 8 quality levels from about 230 kbit/s up to 3 Mbit/sec. The content generation for Microsoft

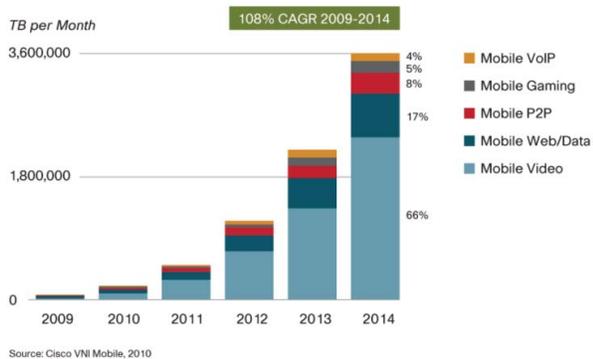


Figure 1: Mobile Data Volume Growth [5].

Smooth Streaming is done by the IIS Transform Manager, which performs a fully automated transcoding of content to Microsoft Smooth Streaming compatible fragmented MP4 files at different quality levels.

Adobe offers test videos for their Dynamic HTTP Streaming together with the standard installation of their Flash Media Server [11]. This includes sample encodings of approx. 2 minutes ranging from 150 kbit/s to 1500 kbit/s which comes together with the Flash Media Server. For content generation they also offer a command line based packaging tool called File Packager which performs the segmentation and packaging of the raw video/audio streams. The encoding of the videos has to be done separately, e.g., using the Adobe Encoder or x264 as an open source alternative [3].

Currently, the only available DASH content is the adapted 3GPP Adaptive HTTP Streaming (AHS) test content of Ericsson Labs which is available in [12]. They offer two clips, Mountain Bike (HVGA resolution) and Snowboard (VGA resolution) with a length of approx. 4 and 5 minutes respectively. This test content is offered at different bitrates ranging from 200 kbit/s to 1200 kbit/s but it maintains the same resolution. However, this content uses segments with a length of 10 seconds which is a restriction when testing, e.g., high bandwidth fluctuations or adaption algorithms like in [9].

Finally, MP4Box from GPAC [13] supports the generation of fragmented MP4 files and basic MPDs for separate representations. At the time of writing of this paper, MP4Box supports only one representation which is also used as the basis for our DASHEncoder introduced in the next section.

3. Content Generation

For this dataset we developed a new content generation tool – on top of GPAC’s MP4Box – for DASH video on demand content called DASHEncoder. Using our DASHEncoder the user does not need to encode and multiplex each quality level of the final DASH content separately. The DASHEncoder steps are depicted in Figure 2. It generates the desired representations (quality/bitrate levels), fragmented MP4 files, and MPD file based on a given configuration file (e.g., as shown in Figure 3) or by command line parameters respectively.

Given the set of parameters the user has a wide range of possibilities for the content generation, including the variation of the segment size, bitrate, resolution, encoding settings, URL, etc., which is shown by the example of a DASHEncoder configuration file in Figure 3. Due to this easy batch processing of multiple encodings to one final DASH source it is also possible to use this application fully automatically, e.g., on user-driven social video

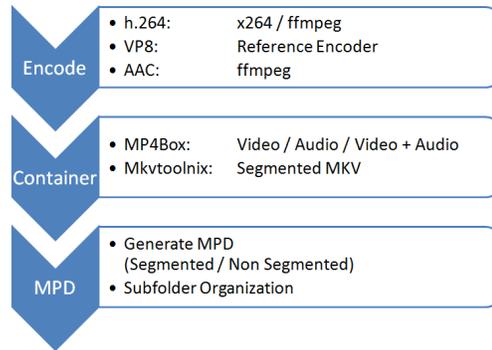


Figure 2: DASHEncoder Content Generation Steps.

```

# x264 Options
#=====
bitrate : 200|400|600|1000
statistics : stat.temp
gop : 48
profile : baseline
preset : slow
input : /home/stefan/foreman_cif.y4m
passes : 1
const-filesize : 0
# MP4Box Options
#=====
fragment-size : 2000
segment-size : 2000
rap-aligned
segment-name : foreman
folder-prefix : foreman
# MPD Options
#=====
mpd-name : foreman.mpd
url-root : http://localhost/foreman_stats/
set-base-url
  
```

Figure 3: DASHEncoder Configuration File Example.

platforms like YouTube, without any time consuming manual work of system administrators.

The DASHEncoder is available open source at [14] with the aim that other developers will join this project. The content generated with the DASHEncoder is compatible with [15] which can be used as a decoder and player respectively.

We developed the DASHEncoder in a very generic way which allows us to add new encoding and multiplexing tools with very low effort and with the usage of the existing parameters and configuration files respectively. Even other video codecs and/or container formats can be integrated in the same way. For example, in addition to the currently integrated AVC/MP4 tools presented in this paper, it is also possible to generate fragmented WebM files which use the VP8 video encoder and a subset of the Matroska container format. Furthermore, it shall be also possible to integrate, e.g., the Adobe packaging tool for HTTP Dynamic Streaming in DASHEncoder and use the same encoded AVC files for the generation of Adobe HTTP Dynamic Streaming compliant content.

For the current dataset we used our DASHEncoder with x264 [16] for the video encoding and MP4Box [13] for the multiplexing as well as for the MPD generation on representation level. Based on these separate MPDs our software generates a combined MPD file containing all representations. This MPD file fits to the generated file structure of the DASHEncoder and the provided URL of the configuration file so that one has no need to edit this file after the content generation process.

Table 1. Segment Length Quality Comparison of the Big Buck Bunny Sequence (300 kbit/sec., 480x360).

Segment Length (GOP)	1 sec. (24)	2 sec. (48)	4 sec. (96)	6 sec. (144)	10 sec. (240)	15 sec. (360)
PSNR	35,83	36,51	36,98	37,14	37,31	37,31

Table 2: Overview of our DASH dataset.

Name	Source Quality	Length	Genre
Big Buck Bunny [17]	1080p YUV	09:46	Animation
Elephants Dream [18]	1080p YUV	10:54	Animation
Red Bull Playstreets	1080p 6 Mbit AVC	01:37:28	Sport
The Swiss Account [19]	1080p 6 Mbit AVC	57:34	Sport
Valkaama [20]	1080p 6 Mbit AVC	01:33:05	Movie
Of Forest and Men [21]	SD	10:53	Movie

In order to retrieve the best suitable bitrate vs. resolution configuration we analysed the PSNR values of the encoded content of several overlapping bitrate configurations. For this we decoded the encoded videos with ffmpeg and upscaled the lower resolution encodings with a bilinear scaling filter to the resolution of the source video. Based on this we calculated the PSNR value of each encoding compared to the original source video. Using this metric we retrieved the best bitrates to change the resolution of the encoded content to provide a continuous quality increase through low to high bitrate levels.

For the subsequent multiplexing and segmentation step it is necessary to maintain fixed I-frame positions of the video which is achieved by restricting the GOP-size of the encoder to the desired segment size of the content. As a consequence, smaller segment sizes have a disadvantage because of their high number of I frames which need more bits in the overall bitstream. Furthermore, smaller GOP sizes also provide a lower encoding performance and quality which is also shown in [1]. Table 1 shows the PSNR values for different segment sizes and provides evidence that this needs to be considered in the evaluation process for the segment sizes of DASH streaming systems.

The time needed for the content generation of one representation in 1080p resolution based on a YUV source video and using 2 Xeon 2,33 GHz CPUs corresponds to the video playback time on average, however lower resolution representations can be generated significantly faster. In this process the time needed is mainly dependent on the x264 encoder and the settings used therefore, the multiplexing and MPD generation in contrast is done in less than 5 % of the total time.

4. DASH Dataset

4.1 Introduction

During our research on adaptive video streaming and DASH we always had the problem that there is no DASH test content available to perform objective comparisons of streaming algorithms and systems. In general there is hardly any DASH content available at all, except the two test sequences from [12].

Some papers used common test sequences like Foreman et al. for their video streaming tests, which are in general quite short for those kind of experiments, in the case of Foreman the length is 300 frames. One approach is to concatenate the same sequence over and over again in order to have a longer sequence for the evaluation. This approach has the disadvantage that the video characteristics are quite the same over the whole experiment which is often not the case in the real-world. In such cases one

Table 3. Target Bitrate Levels for each Genre.

#	Animation	Sport	Movie
1	50 kbit/s, 320x240	100 kbit/s, 320x240	50 kbit/s, 320x240
2	100 kbit/s, 320x240	150 kbit/s, 320x240	100 kbit/s, 320x240
3	150 kbit/s, 320x240	200 kbit/s, 480x360	150 kbit/s, 320x240
4	200 kbit/s, 480x360	250 kbit/s, 480x360	200 kbit/s, 480x360
5	250 kbit/s, 480x360	300 kbit/s, 480x360	250 kbit/s, 480x360
6	300 kbit/s, 480x360	400 kbit/s, 480x360	300 kbit/s, 480x360
7	400 kbit/s, 480x360	500 kbit/s, 854x480	400 kbit/s, 480x360
8	500 kbit/s, 480x360	700 kbit/s, 854x480	500 kbit/s, 854x480
9	600 kbit/s, 854x480	900 kbit/s, 854x480	600 kbit/s, 854x480
10	700 kbit/s, 854x480	1,2 Mbit/s, 854x480	700 kbit/s, 854x480
11	900 kbit/s,1280x720	1,5 Mbit/s,1280x720	900 kbit/s,1280x720
12	1,2 Mbit/s,1280x720	2,0 Mbit/s,1280x720	1,2 Mbit/s,1280x720
13	1,5 Mbit/s,1280x720	2,5 Mbit/s,1280x720	1,5 Mbit/s,1280x720
14	2,0 Mbit/s,1280x720	3,0 Mbit/s,1920x1080	2,0 Mbit/s,1920x1080
15	2,5 Mbit/s,1920x1080	4,0 Mbit/s,1920x1080	2,5 Mbit/s,1920x1080
16	3,0 Mbit/s,1920x1080	5,0 Mbit/s,1920x1080	3,0 Mbit/s,1920x1080
17	4,0 Mbit/s,1920x1080	6,0 Mbit/s,1920x1080	4,0 Mbit/s,1920x1080
18	5,0 Mbit/s,1920x1080		5,0 Mbit/s,1920x1080
19	6,0 Mbit/s,1920x1080		6,0 Mbit/s,1920x1080
20	8,0 Mbit/s,1920x1080		

usually faces a tradeoff between high and low motion scene, fade-in and fade-out, and also quality changes. It is also not always possible to achieve a continuous constant bitrate of an encoded video, especially at higher bitrates due to, e.g., low motion scenes. Therefore, full-length sequences are needed in order to provide more realistic results.

Our DASH dataset comprises the sequences as shown in Table 2 and located at [22]. All animation and movie sequences are licensed under Creative Commons-BY 2.0, so they are free to share and modify. Both sport sequences are free to use for research purposes.

4.2 Representations, Segment Lengths and MPDs

Table 3 shows the representation levels of our dataset. We decided to distinguish the quality level configuration between types of video and to offer at least two sequences with the same configuration. This is useful because of the different scene characteristics (high motion in sport, low detail in Animation, etc.) of the chosen video types. It is also important to mention that the sport and movie genres have only a maximum quality of 6 Mbit/sec. due to fact that our source video files are encoded at 6 Mbit/sec. Another limitation arises at the animation videos due to the lack of achieving the target bitrate of the x264 encoder at higher bitrates for this type of content. Therefore, the higher bitrate encodings are finally encoded at a lower bitrate than desired (the MPDs entries represent the bitrate of the encoded videos).

Additionally, the DASH dataset is encoded and multiplexed using different segment sizes inspired by commercial products ranging from 2 seconds (i.e., Microsoft Smooth Streaming [2]) to 10 seconds per fragment (i.e., Apple HTTP Streaming [4]) with some steps in between and at the lower end. Thus, we end up with the segments sizes of 1, 2, 4, 6, 10 and 15 seconds. The length of the segments is a very important decision when providing content for adaptive streaming. Shorter segments may provide a larger overhead in term of requests, which is shown in the evaluation section, and result in more quality switches, with the cost of a lower encoding efficiency. On the other hand, longer segments are not very handy in environments with high bandwidth fluctuations like mobile networks, because the client cannot adjust the video

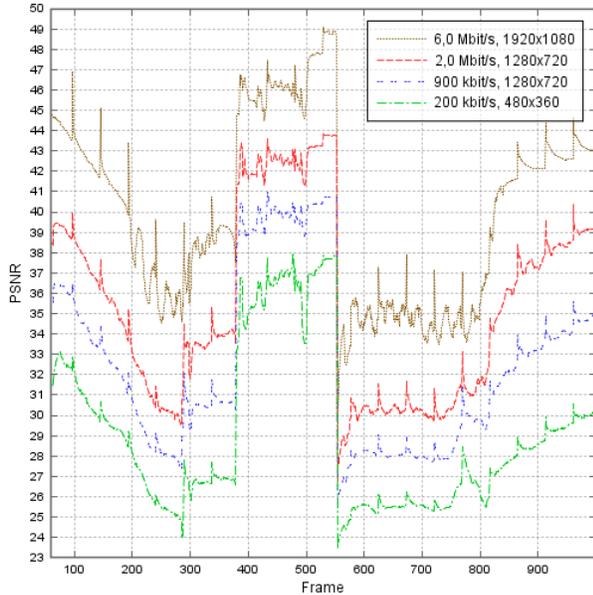


Figure 4: PSNR Value Example of different Representations.

stream as fast as it would be needed in case of a significant bandwidth drop.

In this dataset we offer any content in a version with separate segments and the corresponding MPD. Additionally, we also offer an unsegmented version of the videos and the corresponding MPD for the movies of the animation genre, which allows for byte-range requests. The provided MPDs of the dataset are based on the 2nd Draft International Standard (DIS) of DASH [7] and are also available in the version compatible with the current implementation of the DASH VLC Plugin [15].

4.3 PSNR Statistics

The average bitrate of the streamed video is a good metric for DASH clients and their stream switching algorithms. But the bitrates do not necessarily correspond to the quality of the video, especially because of the differences in encoding efficiency between high- and low-motion scenes. Usually the encoder compensates those differences by adjusting the bitrate over the whole video to get the best overall quality. However, due to the constant bitrate of DASH representations, which is more or less constant all over the video (or at least the current DASH-Period), this is not possible in our scenario. Thus, it can be useful to consider the PSNR as a metric for stream switching algorithms in certain cases.

Therefore, we also provide PSNR values on frame level of the Big Buck Bunny encodings. Those values are available for each frame of each representation and each segment length version of this video, which sums up in more than 1.7 million PSNR values. One can find them together with the dataset in comma separated value (CSV) as well as Microsoft Excel files [22]. As an example, Figure 4 shows the PSNR values of 4 representations of about 850 frames of the 2-second segment length version of our Big Buck Bunny encoding.

4.4 Dataset Usage

Our dataset can be used for many kinds of evaluations in the context of MPEG-DASH like e.g. the comparison of different stream switching algorithms, the caching performance of CDNs

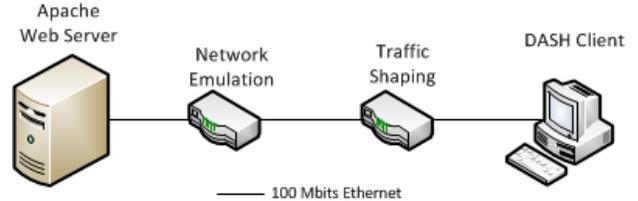


Figure 5: Network Emulation Environment.

and proxies, new applications on top of DASH, etc. The main advantage of our dataset is that it now enables the objective comparison of research results across different institutions working in this field. The dataset and a short compendium of its characteristics can be found in [22] where it is possible either to stream the video directly from our Web server or download the files via FTP.

5. Evaluation

5.1 Introduction

Together with the dataset we provide some basic evaluations of the streaming characteristics of the different segment length versions of the videos and their impact on the average bitrate of the streamed video. Therefore, we evaluate the streaming performance of each segment length version in terms of average bitrate and interpolate a graph to find the optimal segment length for the given network scenario.

During our first tests we realized different Web server configurations on our hosting provider's sites. Some of them allow persistent connections, as defined in HTTP 1.1 [23], and some restrict the reuse of connections for multiple requests by closing the connection after the response. This is signaled by the *Connection: close* field of the HTTP 1.1 response header. Other providers limit the Keep-Alive-Time to 1 second, which is quite short in comparison to the 15 second standard configuration of the Apache Web server [24]. Due to this difference in today's available Web server configurations we also decided to analyze the impact thereof w.r.t. the streaming performance and segment lengths.

5.2 Evaluation Setup

As a first step, we built a test environment (shown in Figure 5 and similar to [25]) emulating standard Internet connections to show the impact of the HTTP server configuration on DASH. Therefore, we used a standard *Apache Web server* as source of our DASH streaming scenario which has a wide range of configuration possibilities regarding connections. The *DASH client* is a modified version of the DASH VLC plugin [15], with some minor changes regarding logging. Additionally, we used two Ubuntu nodes which act as gateways to emulate the typical network characteristics of a last mile Internet connection (e.g. ADSL). One node – *network emulation* – emulates the Internet service provider characteristics by adding a delay of 150 ms using NetEm (packet loss/packet reordering are currently not considered to keep this example evaluation simple). The other node – *traffic shaping* – simulates scenarios of fluctuating bandwidth conditions.

5.3 System Parameters

For our evaluations we used a standard installation of the Apache Web server and enabled/disabled the *KeepAlive On/Off* setting in the *httpd.conf* to show the impact of persistent and non-persistent connections. As already mentioned we simulated the network by

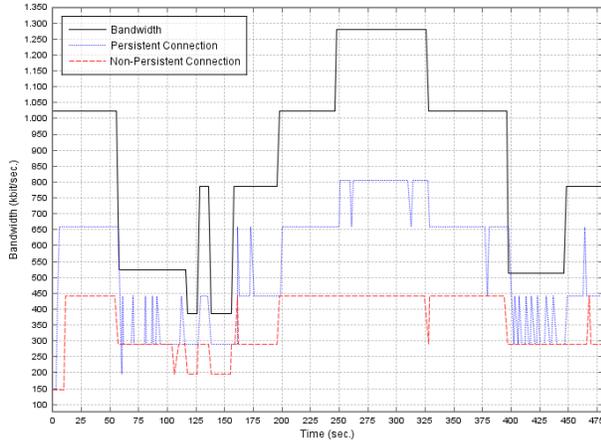


Figure 6: Non-/Persistent Connections - 2 sec. segments.

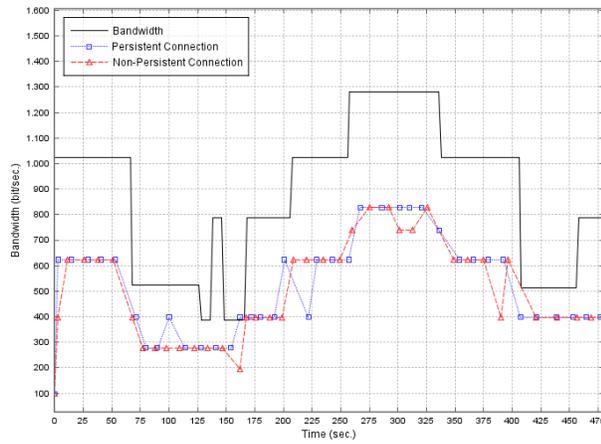


Figure 7: Non-/Persistent Connections - 15 sec. segments.

adding a delay of 150 ms and introducing bandwidth fluctuations. The bandwidth varies over time with a minimum of about 386 kbit/sec. and a maximum of 1536 kbit/sec. for the downlink. For our evaluation we used the Big Buck Bunny movie of our dataset, which is available in 6 different segment lengths of 1, 2, 4, 6, 10, and 15 seconds, each with 21 representations.

5.4 Results

Our evaluations showed a significant performance drop by streaming from a Web server which does not allow persistent connection, especially for short segments lengths like 2 seconds or less. One can see this in Figure 6, which shows the comparison between the DASH streaming performance of the 2-second segment length version of Big Buck Bunny with and without persistent connections.

In our evaluation the streaming using persistent connections corresponds with the findings from [25] and [26] that for TCP-based streaming one needs about the double bandwidth of the media bitrate to achieve a sufficient performance. Streaming without persistent connections results in a significantly higher bandwidth demand and the average media bitrate in our evaluation of 2-seconds segment in Figure 6 is about 34% lower than using persistent connections. The average bitrate in the worst scenario with 1-second segments drops by 46% in comparison to a Web server with persistent connections enabled as shown in Figure 8.

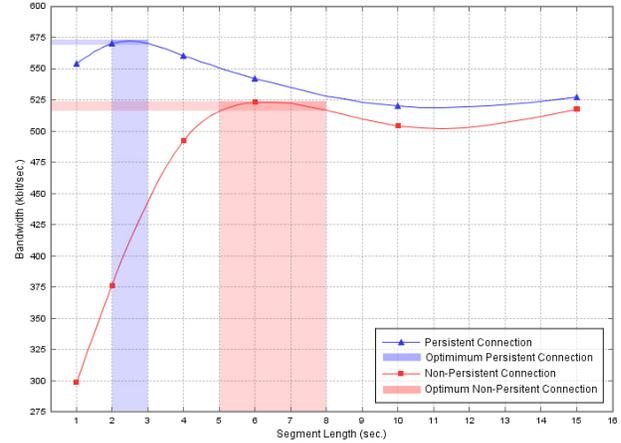


Figure 8: Segment length/Bandwidth Evaluation.

Segment sizes greater than 6 seconds are not influenced so much by the connection settings of the Web server. Due to the lower amount of segments and, thus, the smaller overhead produced by reconnects, the difference between both configurations reduces by 2% for the 15-seconds segment length version of Big Buck Bunny (cf. Figure 7). One can see the pretty equal media bitrate need of both evaluations as well as the lower amount of requests shown by squares and triangles in the plot. The requests are not shown in Figure 6, due to the high frequency of the requests (i.e., every two seconds) compared to one request every 15 seconds shown in Figure 7.

Finally, we evaluated the optimal segment size of our network configuration scenario for both cases, with and without persistent connections. Therefore, we analyzed the performance of the 1, 2, 4, 6, 10 and 15 seconds segment length of Big Buck Bunny of our dataset and interpolated a graph showing the performance of the segment sizes in term average media bitrate. As shown in Figure 8 the optimal segment size for this network setting would be between 2 and 3 seconds using persistent connections and between 5 and 8 without using them. The bitrate of the optimal segment lengths of both configurations differs by about 50 kbit/sec.

6. Conclusion and Future Work

In this paper we provide, to the best of our knowledge, the first DASH dataset comprising different genres, most of them in HD quality. In comparison to other sources for video test content, which mostly offer clips with only a few seconds of length, this dataset is especially designed for DASH-based evaluations and consists of videos between 10 and 90 minutes.

Each video of this dataset is provided in different segment lengths as well as in different representations ranging from 50 kbit/sec. up to several MBit/sec. One can use this dataset to compare DASH-based streaming clients/systems in an objective way with a common content basis. Additionally, we provide PSNR values for every frame of the Big Buck Bunny representations, which provide even more evaluation possibilities for researchers.

We also developed a convenient content generation tool called DASHencoder, which we used for the encoding of this dataset. This fully configurable and extensible DASH encoding tool is offered as open source at [14].

Finally, we showed an evaluation of the optimal segment size for a given network configuration based on our dataset. We also showed the influence of enabled versus disabled persistent connections of Web servers to the streaming performance of DASH. Therefore, we build a simulation environment for performing tests to get the optimal segment size for those different scenarios.

Based on this dataset we plan to do research on stream switching algorithms and DASH-based systems in general. We also plan to update and extend our dataset in the future, like adding alternative encodings or audio tracks to the existing videos. Additionally, we will include versions with different sub-segment lengths and other media formats.

7. ACKNOWLEDGMENTS

This work was supported in part by the EC in the context of the ALICANTE (FP7-ICT-248652) and SocialSensor (FP7-ICT-287975), and QUALINET (COST IC 1003) projects. Special thanks to the Red Bull Media House for providing us the Red Bull Playstreets video. They own the rights of the content but the usage for scientific purposes is permitted. Special thanks also to the team of Valkaama and to the producer Tim Baumann for providing us this content.

8. REFERENCES

- [1] Schonfeld, E., Tech Crunch, 2011. Netflix Now The Largest Single Source of Internet Traffic In North America (posted on May 17th, 2011), <http://techcrunch.com/2011/05/17/netflix-largest-internet-traffic/> (last access: Dec. 2011)
- [2] Microsoft Smooth Streaming, <http://www.iis.net/download/smoothstreaming> (last access: Aug. 2011).
- [3] Adobe HTTP Dynamic Streaming, http://help.adobe.com/en_US/HTTPStreaming/1.0/Using/index.html, (last access: Dec. 2011)
- [4] Pantos, R., May, W. 2010. HTTP Live Streaming, IETF draft (Jun. 2010) <http://tools.ietf.org/html/draft-pantos-http-live-streaming-04> (last access: Dec. 2010).
- [5] Stockhammer, T. 2011, Dynamic Adaptive Streaming over HTTP – Design Principles and Standards, In Proceedings of the second annual ACM conference on Multimedia systems (MMSys11), ACM, New York, NY, USA
- [6] Sodagar, I., The MPEG-DASH Standard for Multimedia Streaming Over the Internet, IEEE Multimedia, vol. 18, no. 4, pp. 62-67, Oct.-Dec. 2011
- [7] ISO/IEC DIS 23009-1.2, Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats.
- [8] Riiser, H., Halvorsen, P., Griwodz, C., Johansen, D. 2010. Low overhead container format for adaptive streaming. Proceedings of the First Annual ACM SIGMM Conference on Multimedia Systems (MMSys), ACM, New York, NY, USA
- [9] Evensen, K., Kaspar, D., Griwodz, C., Halvorsen, P. 2011. Improving the Performance of Quality-Adaptive Video Streaming over Multiple Heterogeneous Access Networks, In Proceedings of the second annual ACM conference on Multimedia systems (MMSys11), ACM, New York, NY, USA
- [10] Smooth Streaming Showcase, <http://www.iis.net/media/experiencesmoothstreaming>, (last access: Dec. 2011).
- [11] Flash Media Server, <http://www.adobe.com/products/flashmediaserver/>, (last access: Dec. 2011)
- [12] Ericsson AHS Content at ITEC/Alpen-Adria-Universität Klagenfurt, http://www-itec.uni-klu.ac.at/dash/?page_id=6 (last access: Dec 2011)
- [13] Le Feuvre, J., Concolato, C., Dufourd, J.-C., Bouqueau, R., Moissinac, J.-C. 2011. Experimenting with Multimedia Advances using GPAC, Proceedings of ACM Conference on Multimedia (ACM MM) 2011, ACM, New York, NY, USA
- [14] DASH Encoder at ITEC/Alpen-Adria-Universität Klagenfurt, http://www-itec.uni-klu.ac.at/dash/?page_id=282 (last access: Dec. 2011)
- [15] Müller, C., Timmerer, C. 2011. A Test-Bed for the Dynamic Adaptive Streaming over HTTP featuring Session Mobility, In Proceedings of the second annual ACM conference on Multimedia systems (MMSys11), ACM, New York, NY, USA
- [16] X264, <http://www.videolan.org/developers/x264.html>, (last access: Dec. 2011)
- [17] Big Buck Bunny Movie, <http://www.bigbuckbunny.org> (last access: Dec. 2011)
- [18] Elephants Dream Movie, <http://www.elephantsdream.org>, (last access: Dec. 2011)
- [19] The Swiss Account, <http://t11.com/2011/07/18/the-swiss-account-2> (last access: Dec. 2011)
- [20] Valkaama Movie, <http://www.valkaama.com>, (last access: Dec. 2011)
- [21] Of Forest and Men, <http://www.offorestsandmen.org>. (last access: Dec. 2011)
- [22] DASH Dataset at ITEC/Alpen-Adria-Universität Klagenfurt, http://www-itec.uni-klu.ac.at/dash/?page_id=207 (last access: Dec. 2011)
- [23] Fielding, R. et al, RFC 2068 - Hypertext Transfer Protocol -- HTTP/1.1, <http://www.w3.org/Protocols/rfc2616/rfc2616.html> (last access: Dec. 2011).
- [24] Apache HTTP Server Project, <http://httpd.apache.org/> (last access: Dec. 2011).
- [25] Kuschnig, R., Kofler, I., Hellwagner, H. 2010. An Evaluation of TCP-based Rate-control Algorithms for Adaptive Internet Streaming of H.264/SVC, Proceedings of the First Annual ACM SIGMM Conference on Multimedia Systems (MMSys), ACM, New York, NY, USA
- [26] Wang, B., Kurose, J., Shenoy, P., and Towsley, D. 2008. Multimedia streaming via TCP: An analytic performance study. ACM Trans. Multimedia Comput. Commun. Appl. 4, 2, Article 16 (May 2008)