Throughput-based quality adaptation for DASH in 5G mobile networks

Jesús Aguilar-Armijo, Cristian J. Vaca-Rubio, Gerardo Gómez,
M. Carmen Aguayo-Torres and José T. Entrambasaguas
E-mail {jaa, cjvr, ggomez, aguayo, jtem}@ic.uma.es
Dpto. de Ingeniería de Comunicaciones. Universidad de Málaga
Campus de Teatinos s/n, 29071 Málaga, España

***Abstract*—Video streaming in mobile networks is currently the most widely used service and its usage is expected to grow exponentially in the next years. Due to the changing conditions of the radio interface, techniques likes Dynamic Adaptive Streaming over HTTP (DASH) allows the user equipment to request the video coding rate that better matches the instantaneous network capacity. There are three types of algorithms to select the appropriate video coding rate based on different types of quality of service metrics: throughput-based, buffer-based and hybrid. In this paper we present three different versions of a throughput-based algorithm, comparing their performance in terms of mean and mode of the video quality index as well as the number of overlapping video chunks. We focus on the end-user quality of experience to evaluate which is the implementation that optimizes the performance.**

I. Introduction

The video streaming service is the most widely used service in mobile networks with a 60% traffic share, and it is expected to grow up to 75% in 2024 [1]. Under these circumstances, it is a key challenge for mobile network operators to offer a proper quality of experience (QoE) to users demanding video services.

Dynamic Adaptive Streaming over HTTP (DASH) is an adaptive bit rate technique that improves the performance of video streaming traffic by adjusting its video quality depending on the instantaneous network conditions in order to maximize the QoE of end users. DASH was defined by MPEG in 2012 in ISO/IEC 23009-1 [2], although it does not define the specific adaptation mechanism neither the metrics involved in the algorithm.

There are already some research works in this area. For example, in [3], an algorithm called Rate Adaptation for Adaptive HTTP Streaming (RAAHS) makes use of the client's playing time and segment recovery time to calculate the maximum bit rate that the client could afford for the next segment request.

Segment-Aware Rate Adaptation (SARA) is an algorithm used in [4] that focuses on the prediction of the time required to download the next video segment based on the segment size variation and buffer occupation.

In [5] metrics based on both consumed content and network information are used. The Radio Network Information Service (RNIS) provides the current network conditions information to improve the user QoE. This technique is not available for all video streaming services and it is mainly focused on live content viewed by many people at the same time.

Authors in [6] use Optimized Stall-Cautious Adaptive Bitrate Streaming Algorithm (OSCAR) to get high video quality avoiding stalls. This algorithm models the network estimating the stall probability with a random variable. It also has a window to prevent future variations of the bitrate.

Another buffer-based algorithm called Smooth Video Adaptation Algorithm (SVAA) is proposed in [7]. This algorithm makes the video quality increase smoothly when the network conditions improve and, oppositely, the video quality is abruptly decreased when the network conditions get worse.

In [8] a new algorithm called Fuzzy-Based DASH (FDASH) is designed for vehicular networks. This algorithm is based on the buffering time at the receiver's application layer and it is compared with other DASH algorithms like Adaptation Algorithm for Adaptive Streaming over HTTP (AAASH), Open Source Media Framework (OSMF), RAAHS, Serial Segment Fetching Time Methods (SFTM) and SVAA, trying to get the minimum number of video interruptions.

In this paper we propose a throughput-based algorithm, which provides accurate and reliable information of the instantaneous network conditions in order to select the best video quality. Different versions of the algorithm have been compared depending on the specific parameters involved in the algorithm: mode/mean of the quality index and number of overlapping video segments. The main advantage of this algorithm over the existing ones is its implementation simplicity. It does not require high computational cost or complex metrics. In addition, it offers a good performance, adapting quickly to variations in network conditions.

The rest of the paper is organized in the following sections. Section II explains the performance of the adaptive source based on DASH. The simulation platform used for the evaluation (WM-SIMA) is briefly described in section III. Section IV describes the configuration of the scenario under test. Subsequently, the results of these tests are shown in section V. Finally, the main conclusions and future work are drawn in section VI.

II. Fundamentals of DASH

DASH requires the video content to be divided into different segments, each one encoded with a different bit rate. These different bit rates are associated with different quality parameters such as resolution, frames per second or video codec. The DASH client decides and requests periodically the quality of the next segment to the server based on certain instantaneous quality of service (QoS) parameters. To maximize the QoE of the end user, the client requests a video segment that it is able to receive before the next segment request, with the maximum available resolution and without causing annoying rebufferings.

Proper adaptation of video streaming traffic to the network conditions is really important in mobile networks, as the radio conditions are typically changing during the connection. This implies that the algorithm need to be able to track them accordingly. Existing adaptation algorithms are divided into three main types: Throughput-based, buffer-based and hybrid [9], depending on the QoS metrics used. This paper is focused on throughput-based algorithms.

III. System model

The Wireless Mobile Simulator (WM-SIMA) [10] has been used to carry out the tests. The architecture of the simulator is organized in two levels: 1) system level (implemented in Matlab) and 2) link level (implemented in C++).

The adaptive source functionality is located at the transmitter (video server), which is responsible for generating the application-level packets. Packet generation is based on a previous packet modeling of real video streaming DASH traffic traces. Several options are possible for modelling the statistical distributions of the segment sizes; a shifted logistic distribution is used in the simulator because of its accurate adaptation, providing better results than Weibull or exponential distributions. For bigger segment sizes, assuming that the size of the successive video segments are independent is an accurate approximation for simplicity. Certainly, the correlation between two consecutive video segments can be measured and input as a parameter into the simulator.

The client periodically sends the selected quality index to the video server. Then, each new segment is generated following the aforementioned distribution with a predefined set of parameters depending on the quality index. Afterwards, packets are delivered to the Transmission Control Protocol (TCP) layer that implements the Reno TCP flavor, which divides incoming packets into TCP segments.

Table I shows the different qualities information:

Table I
Available qualities

Index	Resolution	Codec	Mean segment size
1	320x240	avc1.42c00d	47.000 bits
2	480x360	avc1.42c015	182.000 bits
3	854x480	avc1.42c01e	538.000 bits
4	1280x720	avc1.42c01f	808.000 bits
5	1920x1080	avc1.42c032	2.200.000 bits

At the receiver side of the simulator, after passing the data through lower layers, it reaches again the receiver TCP layer, which is responsible for sending back acknowledgements (ACKs) of correctly received TCP segments. Once the data arrives the application layer of the receiver, the instantaneous throughput is measured so that the DASH algorithm may select the next quality index for the next video segment. Then, it sends the request with this index to the adaptive source at the transmitter.

The following metrics are obtained at the receiver application layer:

- *Instantaneous throughput*: Number of bits received from a video segment divided by the time elapsed between the first and the last packet of that video segment.
- *Video segment delay*: It is measured as the time elapsed between the first packet of the video segment and the last one. The overlap between the different segments is an undesired behavior as it leads to consume content faster than it is downloaded, so the buffer would be reduced. If the buffer goes empty it will entail a rebuffering period which is harmful in terms of QoE.
- *Quality Index*: This metric indicates the quality used in the next video segment. A high quality index leads to a high resolution and a better QoE experience.

A proper selection of the quality index that improves the user QoE is essential in video streaming traffic. For this purpose, an algorithm based on the instantaneous throughput metric has been implemented. Tests have been performed with different versions of the algorithm, which differ in the choice of quality index. The version called $ic2$ chooses the quality whose mean throughput has reached the instantaneous throughput. The version called $ic3$ is more optimistic since it requests the index immediately superior to $ic2$. Finally, the version called $ic1$ is pessimistic since it chooses an immediately lower index than $ic2$.

The proposed algorithms measure the instantaneous throughput at the receiver's application layer; then, the measured throughput is compared with the mean throughput of twenty different video qualities and, depending on the version of the algorithm implemented, it chooses the next video segment quality index. Once the quality is chosen, it is translated to one of the five qualities available in the adaptive source and the client requests the next segment with the calculated index to the transmitter.

IV. Scenario configuration

Tests have been carried out in two different scenarios: 1) a UE moves away from the base station (BS), starting from an initial position located 20 meters from the base station to a final position located at 240 meters from it; 2) a similar scenario but in the opposite direction, so it would be approaching the BS.

The values of the parameters introduced in the simulator at the system level are listed in table II.

Table II
Parameter values

Parameter	Value
$Speed$	30 km/h
$Cellradius$	250 m
$Channelmodel$	UMa (Urban Macrocell)
P_{BS}	0.066 W
P_{OCI}	$6.212 \cdot 10^{-13}$ W
P_{RT}	$4.743 \cdot 10^{-16}$ W
$SINR_{20m}$	36,045 dB
$SINR_{240m}$	-8,561 dB

- $Speed$: Speed of the UE during the simulation.
- $Cellradius$: Radius of the macro cell.

- $Channelmodel$: Simulated channel model.
- P_{BS}: Transmission power of the BS.
- P_{OCI}: Interference power of non-simulated BSs that each user experiences at each resource element.
- P_{RT}: Thermal noise power per resource element whose bandwidth is 15 kHz.
- $SINR_{20m}$: Signal-to-interference-plus-noise ratio at 20 meters from the BS.
- $SINR_{240m}$: Signal-to-interference-plus-noise ratio at 240 meters from the BS.

The request time period at the application layer has been set to one second and the initial quality index is the minimum. The algorithm is not aware of the current bandwidth conditions at the start, so a conservative approach is to request a video segment with the minimum quality until receiving the first feedback from the client.

V. Results

Tests are focused on the maximization of the user QoE, which in the simulator is determined by the following three metrics:

- *Mean quality index*, $E[ic]$: It represents the average of all qualities requested during the connection. A high mean quality index leads to higher resolution for the user.
- *Mode quality index*, $M[ic]$: It represents the quality index that occurs most often. The higher the mode, the better the QoE will be.
- *Overlaps between video segments*: When one video segment overlaps with the next one, it consumes data from the buffer, since each video segment has a mean duration equal to the time between consecutive requests. An overlap does not necessarily mean that there is a rebuffering but it indicates that there is a high probability of rebuffering. This metric is averaged every 10 s, i.e. every 10 video segments.

A. UE moving away from the BS

The average power received per symbol is presented in Fig. 1, which gets worse as the UE moves away from the BS because the UE receives a lower power due to the pathloss. The different versions of the algorithm are represented with different colours ($ic3$ in red, $ic2$ in black and $ic1$ in green), representing the selected quality index along the time. It can be seen how the $ic3$ version tends to select a higher quality index whereas $ic1$ selects the lower quality, although the trend in all cases is always decreasing.

The instantaneous throughput with the $ic3$ version is presented in Fig. 2. Although there is a different video segment every second, many overlapping segments can be recognized, since this version of the algorithm tends to request higher quality. The first video segment has the lowest quality possible, and it is quickly received because of its low segment size and the good radio conditions; as a consequence, the instantaneous throughput is relatively high, causing that the next video segment has the best quality index possible.

The different packet sizes are due to the number of existing Modulation Coding Schemes (MCS), which decreases as the UE moves away from the BS.

Different realizations under the same conditions have been made to reduce the impact of the random factor of the logistic shifted distribution in the generation of the segment size. Therefore, the different statistics will be averaged to compare the different variants of the algorithm.

- The average of the mode quality index of all realizations has been named as "$E[M[ic]]$".
- The mean number of overlapping video segments in 10 requests for all realizations has been named as nOS.

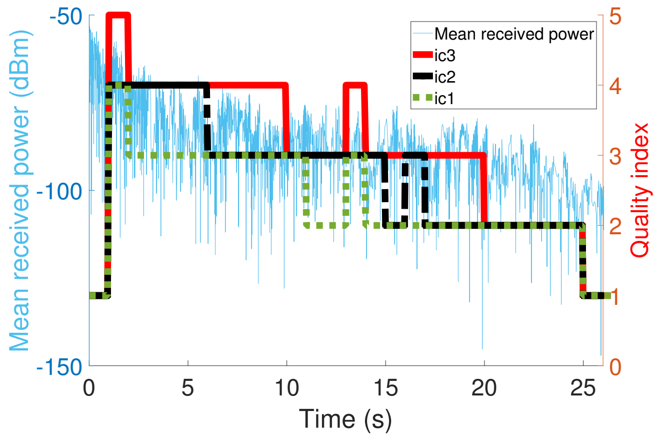

Figure 1. Algorithm variants in decreasing receiver power scenario

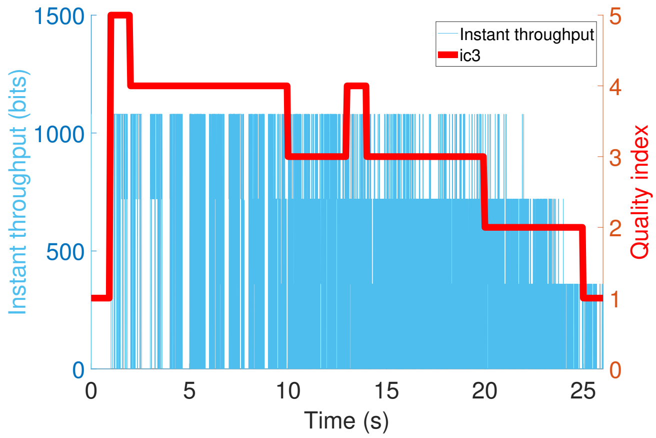

Figure 2. Decreasing instantaneous throughput

Table III
Decreasing index quality

Algorithm variant	$E[ic]$	$E[M[ic]]$	nOS
ic3	3	3.8	7,19
ic2	2.62	2.8	5.77
ic1	2.30	2	5.19

Given the results obtained in table III, there is an agreement between resolution and rebufferings. The higher the resolution the higher the user QoE; however, higher video qualities lead to a higher number of video segment overlappings, that is, a higher rebuffering probability and a poorer user QoE. On the one hand, the $ic3$ version achieves a better mean and mode of the index quality, that is, a higher resolution, at the expense of a higher number of overlappings. On the other hand, the

$ic1$ version achieves a lower mean and mode of the quality index, but also a lower number of rebufferings. Finally, the $ic2$ achieves intermediate results between $ic3$ and $ic1$.

B. UE moving closer to the BS

As the user approaches the BS, the UE receives more power over time, a higher MCS can be used and therefore more channel capacity is available.

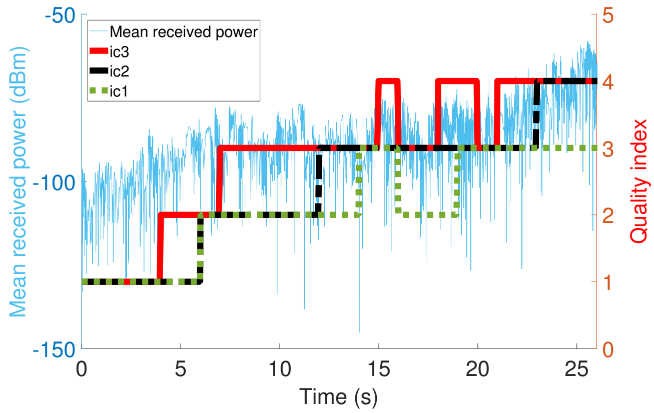

Figure 3. Algorithm variants in increasing receiver power scenario

It can be seen in Fig. 3 how the quality index increases as the radio conditions get better. The $ic3$ version always achieves higher (or at least the same) quality index as the other versions whereas $ic1$ always achieves the lower quality.

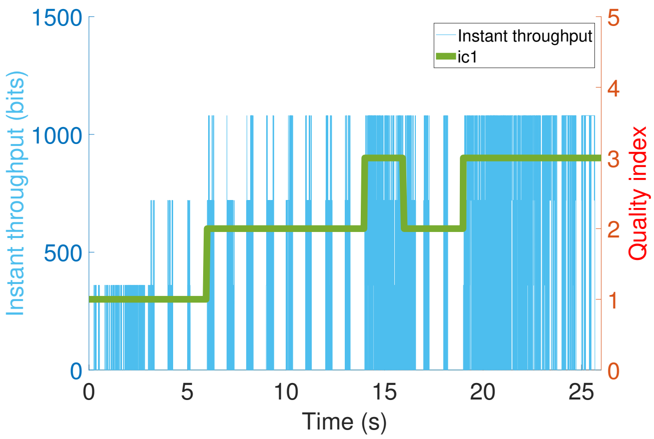

Figure 4. Increasing instantaneous throughput

The instantaneous throughput of the $ic1$ version is shown in Fig. 4. The size of the different segments is smaller than other versions because $ic1$ requests a lower quality index, leading to a lower video resolution but also resulting in a lower number of overlapping video segments.

In this scenario the $ic3$ version obtained higher mode and mean near to the third best quality although the number of overlapping segments is significantly higher than $ic2$ and $ic1$, for this reason it would be better have lower resolution but assure no rebuffering periods on the client.

VI. Conclusions

In order to get the best user QoE in adaptive video sources, a throughput-based algorithm has to balance two different

Table IV
INCREASING INDEX QUALITY

Algorithm variant	$E[ic]$	$E[M[ic]]$	nOS
ic3	2,94	3	6.77
ic2	2.50	2.8	4,88
ic1	2.14	2.1	2.81

QoE metrics: video resolution and rebuffering periods. It is important to deliver the best possible video resolution to the client but this also implies a higher rebuffering probability. Our proposed solution (located in the client's application layer) is intended to adequate the video quality (considering its mean throughput) to the instantaneous measured throughput. An optimistic version of the algorithm improves the video resolution at the expense of achieving a higher probability of rebuffering.

In the future, we are currently working on including in the simulator a buffer at the application layer of the receiver so that it is possible to compute the exact number of rebuffering periods during a simulation. Another potential future work is to include machine learning techniques so that the simulator may compute from a given instantaneous throughput, the number of rebufferings, mode and mean quality index.

Acknowledgements

This work has been supported by the Spanish Government (Ministerio de Economía y Competitividad) under grant TEC2016-80090-C2-1-R and by the Universidad de Málaga

References

[1] Ericsson, "Mobile traffic by application category," https://www.ericsson.com/en/mobility-report/reports/november-2018/mobile-traffic-by-application-category/, 2018, accessed: 2019-03-25.

[2] "Iso/iec 23009-1," https://www.iso.org/standard/65274.html, accessed: 2019-03-26.

[3] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate adaptation for adaptive HTTP streaming," in *Proceedings of the second annual ACM conference on Multimedia systems*, 2011, pp. 169–174.

[4] P. Juluri, V. Tamarapalli, and D. Medhi, "SARA: Segment Aware Rate Adaptation algorithm for dynamic adaptive streaming over HTTP," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 1765–1770.

[5] C. Ge, N. Wang, S. Skillman, G. Foster, and Y. Cao, "QoE-driven DASH video caching and adaptation at 5G mobile edge," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, 2016, pp. 237–242.

[6] A. H. Zahran, J. Quinlan, D. Raca, C. J. Sreenan, E. Halepovic, R. K. Sinha, R. Jana, and V. Gopalakrishnan, "OSCAR: An optimized stall-cautious adaptive bitrate streaming algorithm for mobile networks," in *Proceedings of the 8th International Workshop on Mobile Video*, 2016, pp. 1–8.

[7] G. Tian, Y. Liu, G. Tian, and Y. Liu, "Towards agile and smooth video adaptation in HTTP adaptive streaming," *IEEE/ACM Transactions on Networking (TON)*, vol. 24, no. 4, pp. 2386–2399, 2016.

[8] D. J. Vergados, K. Kralevska, A. Michalas, and D. D. Vergados, "Evaluation of HTTP/DASH Adaptation Algorithms on Vehicular Networks," in *2018 Global Information Infrastructure and Networking Symposium (GIIS)*, 2018, pp. 1–5.

[9] C. Midoglu, K. Kousias, C. Griwodz, and O. Alay, "Evaluation of DASH rate adaptation algorithms in operational mobile networks," *Poster at TMA PhD School*, 2017.

[10] I. M. Delgado Luque, F. J. Martín Vega, A. Carreras, G. Gómez, M. C. Aguayo-Torres, and J. Entrambasaguas, "WM-SIMA: Herramienta de simulacion para el enlace descendente LTE-Advanced," in *XXXII Simposium Nacional de la Unión Cientfica Internacional de Radio, Cartagena*, 2017.