A control-based algorithm for rate adaption in MPEG-DASH

Dimitrios J. Vergados¹, Angelos Michalas¹, Aggeliki Sgora¹,2, and Dimitrios D. Vergados²
¹ Department of Informatics Engineering, Technological Educational Institute of Western Macedonia, Greece
Email: djvergad@gmail.com, amichalas@kastoria.teikoz.gr
² Department of Informatics, University of Piraeus, Greece
Email: {asgora, vergados}@uni.pi.gr

Abstract—Although MPEG-Dynamic Adaptation Streaming over HTTP (MPEG-DASH) eliminates data packet loss, based on the rate adaptation strategy adopted by the user and the network conditions, still playback interruptions may not be avoided. In this paper an algorithm for rate adaption in MPEG-DASH is proposed that employs fuzzy logic to estimate the resolution of the next video segment that each client should obtain from the server. In this way, the buffering time at each client is kept above a target buffering time and thus buffer overflows and unnecessary bit rate fluctuations are avoided. Simulation results showed that the proposed rate adaptation strategy has a beneficial effect on the quality each user perceives.

I. INTRODUCTION

The increasing popularity of video streaming has lead several companies to develop their proprietary solutions, such as Apple, Microsoft, Abode etc, and several standards bodies, such as Third Generation Partnership Project (3GPP), Motion Picture Experts Group (MPEG), etc to establish their own task groups to develop standards for HTTP adaptive streaming (HAS).

In this context, in 2012, MPEG issued its standard for streaming multimedia over Internet, namely MPEG Dynamic Adaptive Streaming over HTTP (DASH) (ISO/IEC 23009-1) [1]. The basic concept of DASH is that users generally prefer to have the best possible resolution of undisrupted video playback in accordance with fluctuations of throughput over a TCP connection. The available throughput over an internet connection could change significantly due to the existence of multiple connections or the presence of interference and fading in channels.

However, the MPEG-DASH specification only defines the Media Presentation Description (MPD) and the segment formats. Since several aspects, such as the media-encoding format, the adaptation techniques, etc are outside the MPEG DASH’s scope, several research works that deal with these issues may be found in the literature [2]–[5].

In this paper a fuzzy control-based algorithm for rate adaption in MPEG-DASH is proposed that employs fuzzy logic to control the buffering time and the resolution in order to avoid buffer overflows and unnecessary bit rate fluctuations.

The rest of the paper is structured as follows. Section II presents preliminaries concerning fuzzy controllers while Section III presents the proposed algorithm for rate adaption. Section IV provides simulation results indicating the efficiency of the proposed model. Finally, conclusions are drawn in Section V.

II. PRELIMINARIES

The concept of fuzzy logic was introduced by Zadeh [6] and is used to make a decision from indeterminate and approximate information. One of the most important applications of the the fuzzy logic is the fuzzy logic controller. The major advantages of fuzzy logic controllers over the conventional controllers are [7]: 1) they do not need accurate mathematical model; 2) they can work with imprecise inputs; 3) they can handle nonlinearity; 4) they are more robust than conventional nonlinear controllers.

Although the fuzzy control application area is really wide, the basic steps of fuzzy controllers consists of:

- Fuzzification: Unlike classical (or crisp) set theory where sets have clear cut boundaries and no uncertainty is allowed, in fuzzy set theory the fuzzy sets have vague boundaries and thus it is able to deal effectively with uncertainties. A fuzzy set is an extension of the classical set theory in that an x can be a member of set A with a certain degree of membership. During the process of fuzzification, a membership is assigned to the crisp input obtained from different membership sets that are described by the means of membership functions, such as triangular, trapezoidal, sinusoidal etc. If x is some variable over some domain of discourse U and X is a fuzzy set over U, then μx(x) is defined to be the degree of membership of x in X. All the membership functions are graded between 0 and 1. The membership sets are assigned linguistic variable names that allow the designer to easily identify the membership set [8]. For example if U is identified with the parameter measuring temperature the fuzzy set X over U can be X = \{too − cold, cold, hot, too − hot\}.

- Fuzzy Rule (or knowledge) base: It associates the fuzzy output to the fuzzy inputs and is constructed to control the fuzzy output. A fuzzy rule is a simple IF-THEN rule with a condition and a conclusion. In case that a rule has multiple parts, fuzzy operators may be used to combine more than one inputs: AND = min, OR = max and NOT = additive complement. During this step the rule matrix is also built to describe
fuzzy sets and fuzzy operators in form of conditional statements.

- Fuzzy Inference Engine: It performs the fuzzy inference process, by computing the activation degree and the output of each rule. It uses all pieces described in previous sections: membership functions, logical operations and if-then rules. The most common types of inference systems are Mamdani and Sugeno [9].

- Defuzzification: The Defuzzification process is used to transform a fuzzy set to a crisp set. The input of this step is the aggregate output fuzzy set and the output is a crisp number. Several techniques may be applied, such as the maximum method, the centroid method, the center of gravity method etc.

III. THE PROPOSED ALGORITHM

The basic concept of DASH is that users generally prefer to have the best possible resolution of undisrupted video playback in accordance with fluctuations of throughput over a TCP connection. The available throughput over an internet connection could change significantly due to the existence of multiple connections or the presence of interference and fading in channels.

To this end, our model is based on clients implementing the MPEG-DASH standard to request streams of video with different resolutions from an HTTP video server. Each client uses the fuzzy controller rate adaptation algorithm to estimate the resolution of the next video segment obtained from the server.

A. The Proposed Algorithm

We assume that a video stream consisted of \( n \) segments of duration \( \tau \) is available at the server. Each segment is encoded in multiple resolutions of quality. The segment throughput is estimated at the client as:

\[
r_i = \frac{(b_i \times \tau)}{t_i^e - t_i^b}
\]

where \( b_i \) denotes the bit rate of segment \( i \), \( t_i^e \) and \( t_i^b \) denote respectively the time when the segment \( i \) has been started downloading and the time the whole segment has been received at the client.

Each client requests the next video segment with higher/lower or equal resolution to the last downloaded segment. The increase/decrease in resolution is determined by the fuzzy controller rate adaptation algorithm. The algorithm uses two crisp inputs, namely the buffering time denoting the time \( t_i \) that the last received segment waits at the client until it starts playing and the difference \( \Delta t_i = t_i - t_{i-1} \) of the last buffering time from the previous one. The proposed scheme tries on one hand to keep the buffering time at the client above the target buffering time \( T \) to avoid buffer under-runs and on the other hand to retain the difference between the current and the previous resolution close to zero in order to reduce consecutive changes of video resolution subject to continuous variations of the network throughput.

Specifically, the linguistic variables of the buffering time input are described as short, close and long, so as to denote the distance of the current buffering time from the target \( T \). Additionally, the linguistic variables for the differential of the buffering time input, signify that the rate between subsequent buffering times is falling, steady or rising, while the linguistic variables of the output described as reduce (\( R \)), small reduce (\( SR \)), No Change (\( NC \)), small increase (\( SI \)) and increase (\( I \)) declaring the factor of the increase/decrease of the resolution of the next segment.

The linguistic values for the input and output variables are represented by triangular fuzzy numbers. Figs. 1 and 2) depict the membership functions of each input, while Fig. 3 depicts the membership function of the output. The applied fuzzy rules are summarized in Table I.

<table>
<thead>
<tr>
<th>TABLE I: Fuzzy rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rising</td>
</tr>
<tr>
<td>Steady</td>
</tr>
<tr>
<td>Falling</td>
</tr>
</tbody>
</table>

Fig. 1: The membership functions for the buffering time

Accordingly, the rules that associate the fuzzy output to the fuzzy inputs could be expanded to the following if-then-else statements:

- Rule 1: if (short) and (falling) then \( R \)
- Rule 2: if (close) and (falling) then \( SR \)
- Rule 3: if (long) and (falling) then \( NC \)
- Rule 4: if (short) and (steady) then \( SR \)
- Rule 5: if (close) and (steady) then \( NC \)
- Rule 6: if (long) and (steady) then \( SI \)
- Rule 7: if (short) and (rising) then \( NC \)
- Rule 8: if (close) and (rising) then \( SI \)
- Rule 9: if (long) and (rising) then \( I \)

Then, the value of each rule is calculated as the minimum value among the two input functions that comprise it. Afterwards each output variable is calculated according to the
to the estimated channel throughput over the last period. Specifically,

$$b_k = f(t_i, \Delta t_{i-1}) \times r_d$$  \hspace{1cm} (8)$$

The term $r_d$ is the available connection throughput, estimated as the average segments throughput of the last $k$ segments downloaded during a specified period of time $d$. Consequently, the available connection throughput equals to $1/k \times \sum_{i=1}^{k} r_i$.

The final step of the algorithm tries to avoid unnecessary bit rate fluctuations. Since the videos are available on the server only at specific resolutions, $b_k$ is quantized to the highest available resolution $b_{k_0}$ that is lower than $b_k$. If $b_{k_0} > b_{k-1}$ and by selecting the new bit rate $b_k$, the buffer level is estimated to be less than $T$ for the next 60 sec, then the bit rate remains unchanged. Similarly, if $b_{k_0} < b_{k-1}$, but the old bit rate is estimated to produce a buffer level for the next 60 sec that is larger than $T$, then the bit rate remains unchanged. In all other cases the bit rate of the next segment is set to $b_{k_0}$.

IV. SIMULATION SETUP AND RESULTS

A. Simulation Setting

Our experiments were implemented using the simulation software ns-3 (http://www.nsnam.org).

The simulation setup consisted of several nodes, one acting as a DASH MPEG server, and the others acting as DASH MPEG clients. The bottleneck link is a 2Mbps link, with a propagation delay of 5ms. Each client node tries to download and play a video from the server using the proposed algorithm. The target buffering time $T$ was equal to 35 sec whereas the time period $d$, estimating the available connection throughput at the client, was set to 60 sec. The factors $N_2$, $N_1$, $Z$, $P_1$ and $P_2$ of the output membership function were set equal to 0.25, 0.5, 1, 1.5 and 2 respectively. The code of our experiments is available in https://github.com/djvergad/dash/tree/master/model.

The available video segments at the server have been produced using the trace files of several dash video streams given in http://www-itec.uni-klu.ac.at/ftp/datasets/mmsys12/BigBuckBunny/. The duration of each video segment $\tau$ was equal to 2 sec. The video segment resolutions used in the experiments were 45000, 89000, 131000, 178000, 221000, 263000, 334000, 396000, 522000, 595000, 791000, 1033000, 1245000, 1547000, 2134000, 2484000, 3079000, 3527000, 3840000, 4220000 bps.

B. Two clients attempting to access a DASH server

In this scenario we investigate the behavior of the algorithm when two clients compete for access to the server. In this case the target buffering time $T$ was set to 7sec. In Fig. 4 the evolution of the buffering time $t$ as well as the evolution of the differential of the buffering time $\Delta t$ can be seen for both clients. According to the obtained results, the algorithm manages to keep the playback buffering time above the target buffering time as well as the difference between the current and the previous resolution close to zero.
C. Multiple clients with varying target buffering times accessing a DASH server

In this scenario we evaluated the performance of the proposed algorithm for different numbers of clients and target buffering times.

Fig. 5 shows the average number of playback interruptions while Fig. 6 presents the average total interruption time among users. As it can be seen the algorithm tries to avoid buffer starvation by adjusting video resolution. Thus, in most cases the average number of playback interruptions as well as the average total interruption time is close to zero, except for the cases that there is a large number of clients requesting video or a small target buffering time is specified at the algorithm.

Fig. 7 depicts the average buffering time among clients throughout the simulation. From the obtained results, in can be observed that as the target buffering time increases, the average actual buffering time at the client increases as well. In case there are a few client connections, and a high target buffering time, then the largest playback buffering times are achieved at the clients.

Fig. 8 illustrates the average bitrate obtained by the clients. As it can be seen, the algorithm tries to adjust the download bit rate to the available throughput in an effort to provide the best possible resolution for each client. Accordingly, when there are few client connections, video of high resolution is provided. However as the number of connections increases, the video bitrate falls. In addition, the target buffering time also affects the bitrate; when the target buffering time increases, the average bitrate falls.

Finally, Fig. 9 presents the average minimum bitrate obtained by all clients. It can be seen that as the number of clients increases, the minimum rate decreases. However the minimum rate does not seem to be affected significantly by the target buffering time.

V. CONCLUSIONS

In this paper a novel MPEG-DASH fuzzy control-based adaptation scheme employing fuzzy logic is presented aiming
to efficiently adjust the video rate delivered to each user in accordance to the current network conditions. Each client watching a video stream obtained from the DASH server downloads the video segments by estimating their resolution with the aid of a fuzzy controller rate adaptation algorithm.

Simulation results showed that the proposed algorithm succeeds in avoiding buffer starvation by adjusting video resolution. Also, the results indicate the significance of the target buffering time value. The increase of the target buffer time has a beneficial effect on the users perceived quality. Thus if the flow is not ineffective, the algorithm can perform reasonably well for various congestion levels. However, when the buffer target is low, as would be the case in interactive video conferencing, the performance of the algorithm worsens in terms of interruptions and bitrate.

Directions for future work include the realization of further wide scale simulation trials in order to experiment with the applicability of the algorithm presented herewith, as well as, to incorporate other DASH adaptation schemes in ns-3 and test their performance against our algorithm’s performance.

ACKNOWLEDGEMENTS

This work is partly supported by UPRC (University of Piraeus Research Center).

REFERENCES