A Learning-Based Algorithm for Improved Bandwidth-Awareness of Adaptive Streaming Clients

Jeroen van der Hooft†, Stefano Petrangeli†, Maxim Claeys†, Jeroen Famaey§ and Filip De Turck†
† Ghent University - iMinds, Department of Information Technology
Gaston Crommenlaan 8/201, B-9050 Ghent, Belgium
§ University of Antwerp - iMinds, Department of Mathematics and Computer Science
Middelheimlaan 1, B-2020 Antwerp, Belgium
E-mail: jeroen.vanderhooft@intec.ugent.be

***Abstract*—HTTP Adaptive Streaming (HAS) is becoming the de-facto standard for Over-The-Top video streaming. A HAS video consists of multiple segments, encoded at multiple quality levels. Allowing the client to select the quality level for every segment, a smoother playback and a higher Quality of Experience (QoE) can be perceived. Although results are promising, current quality selection heuristics are generally hard coded. Fixed parameter values are used to provide an acceptable QoE under all circumstances, resulting in suboptimal solutions. Furthermore, many commercial HAS implementations focus on a video-on-demand scenario, where a large buffer size is used to avoid play-out freezes. When the focus is on a live TV scenario however, a low buffer size is typically preferred, as the video play-out delay should be as low as possible. Hard coded implementations using a fixed buffer size are not capable of dealing with both scenarios. In this paper, the concept of reinforcement learning is introduced at client side, allowing to adaptively change the parameter configuration for existing rate adaptation heuristics. Bandwidth characteristics are taken into account in the decision process, thus allowing to improve the client's bandwidth-awareness. Focus in this paper is on actively reducing the average buffer filling, evaluating results for two heuristics: the Microsoft IIS Smooth Streaming heuristic and the QoE-driven Rate Adaptation Heuristic for Adaptive video Streaming by Petrangeli et al. We show that using the proposed learning-based approach, the average buffer filling can be reduced by 8.3% compared to state of the art, while achieving a comparable level of QoE.**

I. Introduction

Over the last years, delivery of multimedia content has become more prominent than ever. Particularly, video streaming applications are responsible for more than half of the internet traffic [1]. To enable video streaming over the best-effort Internet, the concept of HTTP Adaptive Streaming (HAS) has recently been introduced. In HAS, video content is temporally divided into segments with a typical length of 1 to 10 seconds, each encoded at multiple quality levels. Segments are dynamically requested by the HAS client, equipped with a rate adaptation heuristic to select the best quality level based on criteria such as the perceived bandwidth and the video player's buffer filling level. The HAS approach comes with several advantages. For the provider, video delivery is cheaper because no dedicated network elements are required. Better scalability is guaranteed, since quality selection is performed by clients in a distributed way. For the end user, a smoother playback experience is generally perceived, as the client can adapt the requested bit rate to the perceived bandwidth and device characteristics. Because of these advantages, major players such as Microsoft, Apple and Adobe massively adopted the adaptive streaming paradigm. However, two issues exist. Current implementations are generally hard coded, using parameters and threshold values optimized for specific network conditions. This prevents true adaptation to changing network environments and makes it harder to deal with a vast range of network setups and corresponding bandwidth variations. Furthermore, many commercial HAS implementations focus on a video-on-demand scenario, where a large buffer size is used to avoid play-out freezes. When the focus is on a live TV scenario however, a low buffer size is typically preferred as the video play-out delay should be as low as possible. Current implementations generally use a large buffer size (i.e. 10 to 30 seconds or more), and are therefore not capable of dealing with both scenarios.

In this paper, we propose to introduce the concept of reinforcement learning (RL) at client side, so that the client is able learn the most appropriate parameter configuration under different network conditions. RL is a machine learning technique, in which an agent can learn about its environment by performing a number of actions. Every time an action is taken, the agent perceives feedback through a numerical reward from the environment. The agent's goal is to learn which action should be taken in a given environmental state, in order to maximize the cumulative numerical reward [2]. In the proposed solution, the action set is defined by possible parameter configurations for the rate adaptation heuristics, while the agent's environment is defined by certain properties of the perceived bandwidth. In this way, the client's bandwidth-awareness can be improved. The proposed learning-based approach is applied to two existing rate adaptation algorithms: the widely used Microsoft IIS Smooth Streaming (MSS) algorithm [3] and the QoE-driven Rate Adaptation Heuristic for Adaptive video Streaming (QoE-RAHAS) algorithm by Petrangeli et al. [4]. Focus here is on actively reducing the average buffer filling level, while providing the user with an acceptable QoE at all times.

The remainder of this paper is structured as follows. The concept of HAS is presented in Section II, while the concept of RL is discussed in Section III. The proposed learning-

based approach is presented in Section IV, defining all RL components involved. A detailed evaluation is presented in Section V. Related work is discussed in Section VI, before coming to final conclusions in Section VII.

II. HTTP Adaptive Streaming

In this section, the concept of HAS is discussed and the most important features of the MSS and QoE-RAHAS rate adaptation heuristics are briefly described. In addition, an explanation is given as to why adaptively changing the parameter configuration should indeed be beneficial.

A. General Concept

HAS represents the third generation of HTTP-based streaming solutions. An overview of the general concept is shown in Figure 1. The video content is temporally segmented and encoded at different quality levels. The segment duration generally varies between 1 to 10 seconds, depending on the implementation. A manifest file is maintained by the HAS server, which contains information concerning the segments and the available quality levels. Based on this information, the client requests the next segment s_i at quality q_i to the HAS server upon arrival of the previous segment. The client decodes all segments and plays back the sequence of chunks in linear order. The main advantage of HAS is that the client can decide at which quality level the next segment is requested. A quality selection heuristic is used for this purpose, basing its decision on criteria such as the perceived bandwidth and buffer filling. In this way, the client can adapt to network conditions and provide the user with a better video streaming experience.

Many rate adaptation heuristics have recently been proposed. Well-known examples are Microsoft's IIS Smooth Streaming (MSS) [3], Apple's HTTP Live Streaming (HLS) [5] and Adobe's HTTP Dynamic Streaming (HDS) [6]. As most of these implementations tend to use the same architecture, the Motion Picture Expert Group (MPEG) proposed Dynamic Adaptive Streaming over HTTP (DASH), a standard that defines the interfaces and protocol data for adaptive video streaming [7]. The rate adaptation heuristics are however still implementation specific.

In this paper, our proposed learning-based approach is evaluated for two existing rate adaptation heuristics. The first algorithm is the MSS rate adaptation heuristic [3], as this heuristic is widely used and source code is freely available [8]. The second considered algorithm is the QoE-RAHAS heuristic [4], proposed by Petrangeli et al. This algorithm is based on a dedicated model for the QoE and has shown promising results. Both of these algorithms are briefly described below.

B. Considered Rate Adaptation Heuristics

In the MSS heuristic [3], the next quality level is selected based on the current buffer filling and the perceived bandwidth. The most important parameters are the buffer size and the panic, lower and upper thresholds, which actively steer the buffer filling towards a value between the lower and upper threshold: a lower quality level is selected when the buffer filling drops below the lower threshold, and a higher quality level is selected when the buffer filling exceeds the upper threshold. When the buffer filling is lower than the panic threshold, the rate adaptation heuristic immediately selects the lowest quality level, in an attempt to avoid buffer starvation. In this way, an attempt is made to deliver a higher average quality level and to actively avoid play-out freezes.

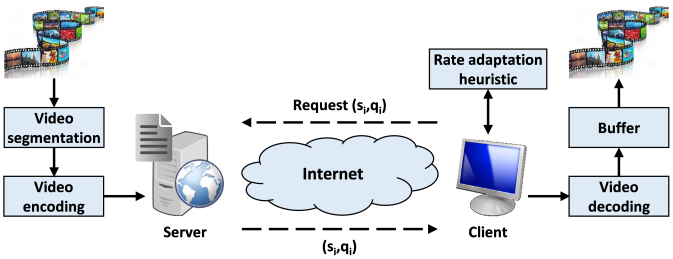


Figure 1: HTTP Adaptive Streaming concept.

The goal of the QoE-RAHAS rate adaptation heuristic is to maximize the user's QoE [4]. This is achieved by intelligently selecting the next quality level, pursuing a high average quality level while limiting the number of quality switches and avoiding play-out freezes. Quality selection is based on a utility function, designed to be a metric for the QoE (cfr. Subsection V-B). The most important parameters are the buffer size, the panic threshold and a target level for the buffer filling.

In Section IV, a self-learning HAS client is proposed that dynamically changes the parameter configuration for both heuristics. An extensive analysis revealed that performance for the heuristics highly depends on the perceived bandwidth. When the perceived bandwidth is fixed, a small buffer size and filling is sufficient to provide the user with an acceptable QoE. This is because the same quality level can be selected for a certain amount of time, while little variations occur in the buffer filling. Buffer starvation is less likely to occur, so that little to no play-out freezes are observed. When the perceived bandwidth is highly variable however, a large buffer size is required to prevent buffer starvation and a decrease in the selected quality level. A higher average buffer filling is observed in this case, corresponding to an increased play-out delay. In the proposed learning-based approach, certain characteristics of the perceived bandwidth are taken into account, enabling the client to adapt the parameter configuration and buffer filling to these different network conditions.

III. Reinforcement Learning

RL is an area of machine learning in which an agent can only interact with its environment through a set of specified actions. The agent does not need any a priori knowledge of this environment, and evaluates its actions based on an assigned, numerical reward. The agent's goal is to learn the optimal action to take in a given environmental state, in order to maximize a cumulative numerical reward [2]. The basic RL model is typically formulated as a Markov decision process (MDP), formally described as a 4-tuple (S, A, P, R) where S is a finite set of states, A is a finite set of actions, P is a state transition probability matrix and R is a reward function.

A well-known RL algorithm is Q-learning, a model-free technique introduced by Watkins [9]. A Q-table is used, where rows correspond to the state set S and columns correspond to the action set A. For each state-action combination (s, a) a Q-value $Q(s, a)$ is stored, which reflects the quality of performing action a when the environment is in state s. The Q-values are updated every time an action a is taken in a state s, resulting in a reward $r = R_a(s, s')$ and a new state $s' \in S$:

$$Q(s,a) = (1-\alpha)\ Q(s,a) + \alpha\ \left[r + \gamma\ max_{a'} Q(s',a')\right] \quad (1)$$

In this equation, $\alpha \in [0; 1]$ and $\gamma \in [0; 1]$ are the learning rate and the discount factor respectively. The former determines to what extent the agent learns from newly acquired information, the latter determines the importance of future rewards. In this formulation, rewards are only accounted to the last action performed. To account rewards to actions taken further in the past, eligibility traces can be applied through an eligibility trace-decay parameter $\lambda \in [0; 1]$. These traces record which states have recently been visited and indicate the degree to which each state-action combination is eligible for undergoing learning changes when a new reward is perceived.

One of the challenges during the learning process is finding the right balance between exploration and exploitation. Although complex exploration methods have successfully been applied, simpler methods are shown to be efficient in practice. One of these is the ϵ-greedy approach [9], where the action with the highest current Q-value is selected with probability $1 - \epsilon$, while a random action is selected with probability ϵ. The parameter ϵ is application dependent, so it has to be fine-tuned to find a near-optimal value. Most important drawback is that all actions have an equal chance of being selected when the agent is exploring, while one might expect the estimated next-to-best action to have a higher chance of being selected than the worst action. To overcome this issue, the Softmax exploration method was proposed by Sutton and Barto [2]. In this method, a Boltzmann distribution is most commonly used to rank the Q-values for a specific state. The selection probability $P(a|s)$ for every action is calculated as follows:

$$P(a|s) = \frac{\exp[\beta Q(s,a)]}{\sum_{a'} \exp[\beta Q(s,a')]} \quad (2)$$

In this equation, β is a strictly positive parameter called the Softmax inverse temperature. Low values for β cause the actions to be nearly equiprobable, while high values lead to a higher selection probability for actions with a higher Q-value.

IV. RL-Based HAS Optimization

In this section, the proposed RL-based approach is presented. The environmental state elements, the defined reward function and the agent's actions are described in detail.

A. State Definition

The state of the environment is defined using two specific properties of the perceived bandwidth. The first property is the average available bandwidth, as it strongly affects the quality decision and thus the QoE of the video stream. Boundaries are based on the available quality bit rates: when video content is offered at N quality levels, $N + 1$ bandwidth levels are considered. The second property is a metric capable of discriminating between a variable and a fixed bandwidth scenario. A number of metrics were evaluated over a range of bandwidth samples (e.g. the standard deviation and the median deviation), eventually leading to the use of the mean absolute difference. Defined boundaries need to provide the right granularity, taking into account the considered experimental setup. Both state elements are more closely defined in Subsection V-A.

B. Reward Definition

As for the reward, a trade-off is made between the perceived QoE and the buffer filling. We propose a reward function that is the linear combination of two terms. The first term is defined as the Mean Opinion Score (MOS, cfr. Subsection V-B) obtained over a certain time window. The second term represents the average buffer filling level over the same time window. The reward r is defined as follows:

$$r = \psi\, \text{MOS}_w - (1-\psi)(\text{buffer}_w) - c \quad (3)$$

The trade-off between a high MOS and low buffer filling is reflected by the parameter ψ. A value close to 1 means that the only goal of the client is to pursue a high QoE, while a value close to 0 means that the client is driven to select the parameter configuration leading to the lowest buffer filling. The rationale behind this reward is that, in a variable bandwidth scenario, a significant increase in terms of the QoE is obtained when the average buffer filling is increased. Using an appropriate value for ψ, the agent should learn to use a parameter configuration that leads to a higher average buffer filling. In a fixed bandwidth scenario however, the possible increase in terms of the MOS is relatively small. Using the same value for ψ, the agent is now expected to learn to use a parameter configuration that leads to a low buffer filling. Note that the reward component MOS_w will differ significantly for different levels of the available bandwidth, thus illustrating the importance of incorporating the average available bandwidth in the environmental state. In our design we make sure rewards are negative at all times, by subtracting a constant value c. This is because the learning phase starts with an all-zero Q-table, and exploration would therefore be limited when positive rewards are considered.

C. Action Definition

The action set consists of several parameter configurations, either for the MSS or for the QoE-RAHAS algorithm. For the MSS algorithm, the most important parameters are the buffer size and the panic, lower and upper thresholds. Preliminary results showed that the buffer size and threshold values have a significant impact on the average QoE and buffer filling (cfr. Subsection V-C). Therefore, each action corresponds to setting a new value for the buffer size and threshold values. As the buffer size is expressed as a multiple of the segment size, considered parameter configurations are defined in the

Quality level	Bit rate [kpbs]
1	300
2	427
3	608
4	866
5	1233
6	1636
7	2436

Table I: Bit rates for the *Big Buck Bunny* video trace.

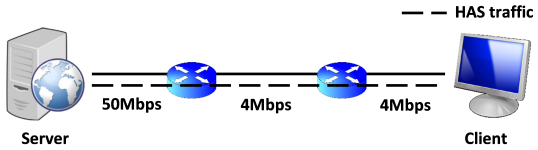


Figure 2: Simulated network topology.

experimental setup. For the QoE-RAHAS algorithm, the most important parameters are the buffer size, the panic threshold and the buffer target. Preliminary results showed that the buffer size, the panic threshold and buffer target have a significant impact on the average QoE and buffer filling (cfr. Subsection V-C). Every action thus corresponds to setting a new value for the buffer size, the panic threshold and buffer target. Again, the considered parameter configurations are defined in the experimental setup.

D. Algorithm Parameters

The proposed approach makes use of the following parameters. First, the decision interval D, indicating how often a new parameter configuration will be selected. This interval is expressed as the number of video segments that is downloaded between two successive configuration changes. Second, the reward window W, indicating how many segments are taken into account when evaluating the average MOS, buffer filling and average available bandwidth. Note that W should be equal to or lower than D, as the reward needs to reflect the performance for the current parameter configuration only. Third, the trade-off parameter ψ in the reward function, which will influence the average buffer filling level and the QoE. Finally, the following Q-learning parameters should be considered: the learning rate α, the discount factor γ, the eligibility trace decay λ and the Softmax inverse temperature β. In Section V-D, the optimal values for these parameters are discussed.

V. Evaluation and Discussion

A. Experimental Setup

To evaluate the performance of the MSS and QoE-RAHAS rate adaptation heuristics and our learning-based approach, a simple network topology was modelled using the NS-3 network simulator [10]. It consists of a single HAS client, streaming the *Big Buck Bunny* video trace from a dedicated HAS server, as shown in Figure 2. The video trace consists of 299 segments, each 2 seconds of length and encoded at seven different quality levels, as shown in Table I.

Based on these quality levels and the segment size, the states of the environment and the actions for the learning-based approach can be defined. Seven quality levels are defined, resulting in eight considered bandwidth levels. As for the mean absolute difference, seven levels are distinguished to provide the right granularity. Thus, 56 different states are distinguished in the environmental mode. Note that the perceived bandwidth at any given point in time is calculated using an exponentially weighted moving average over past bandwidth samples.

Action	BS [s]	PT [s]	LT [s]	UT [s]
1	6	2	3	3
2	8	2	4	5
3	10	2	6	6
4	12	2	6	8

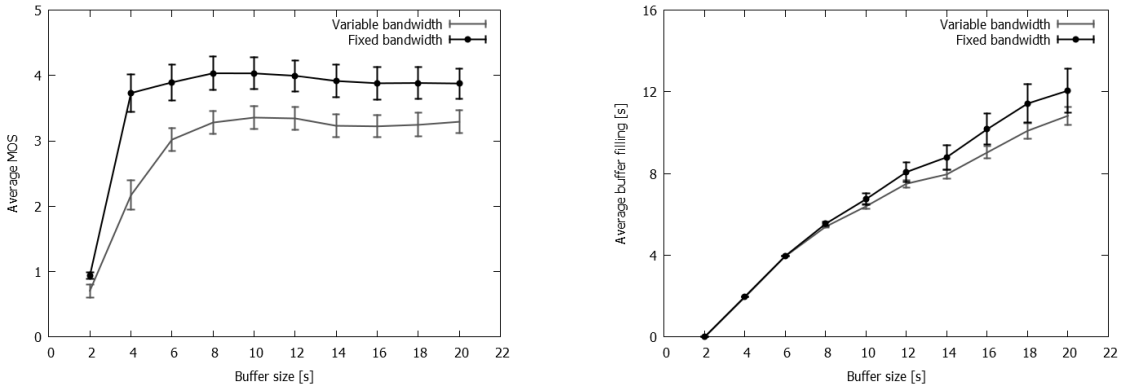
Table II: Defined actions for the MSS algorithm.

Action	BS [s]	PT [s]	BT [s]
1	4	0	2
2	6	2	4
3	8	2	6
4	10	2	8

Table III: Defined actions for the QoE-RAHAS algorithm.

Defined actions for the MSS and QoE-RAHAS algorithms are shown in Tables II and III. For the MSS heuristic, the buffer size (BS) and the panic, lower and upper thresholds (PT, LT and UT respectively) are defined for every action. Possible values for the buffer size are defined as a multiple of the segment size. A buffer size higher than 12 seconds is not considered, as preliminary results showed that the average QoE does not improve when a larger buffer is used. The lowest considered buffer size is 6 seconds, as a smaller buffer would lead to an unacceptable QoE [11]. For the QoE-RAHAS heuristic, the buffer size (BS), the panic threshold (PT) and the buffer target (BT) are defined. Even lower values for the buffer size are considered, because preliminary results indicated that even for a buffer size of 4 seconds, an acceptable QoE can be achieved when the available bandwidth is fixed (cfr. Subsection V-C). The maximum buffer size is limited to 10 seconds, as results again showed that the QoE does not improve when a larger buffer is used.

To evaluate the performance of the rate adaptation heuristics under different network conditions, several realistic bandwidth traces were generated. Variable bandwidth traces were constructed using the approach suggested by Claeys et al. [12], resulting in traces with an average bandwidth of 1550kbps and a standard deviation of 463kbps. Fixed bandwidth traces on the other hand, simply consist of a uniformly selected value for the available bandwidth, ranging from 350kbps to 3277kbps. To evaluate the performance of the traditional MSS and QoE-RAHAS algorithms, 50 episodes of the video trace were streamed using 50 variable and 50 fixed bandwidth traces. When the learning-based approach is introduced, the learning phase of the agent consists of 3000 episodes of the video trace, using randomly selected bandwidth traces simulating either a variable or a fixed bandwidth scenario. Afterwards, results for the same 50 variable and fixed bandwidth traces are evaluated for comparison reasons.



(a) Average MOS

(b) Average buffer filling

Figure 3: Impact of the buffer size on the average MOS and buffer filling in the QoE-RAHAS algorithm, using the most appropriate panic threshold and buffer target.

B. Evaluation Metrics

A first evaluation criterion is based on the observed QoE, for which several metrics exist. One of these is the Mean Opinion Score (MOS), which is an average score ranging from 1 (bad QoE) to 5 (excellent QoE) that was first introduced in the domain of telephony [13]. The MOS is a subjective score, thus requiring the use of human test subjects. De Vriendt et al. however proposed an estimation of the MOS for HAS services, which is based on two objective factors: the average requested quality level and its standard deviation [14]. The estimated MOS is computed as a linear combination of these two factors, for which parameters were fine-tuned based on results from subjective measurements. Mok et al. however showed that freezes also have a large impact on the average MOS [15]. In their research, estimations are calculated using three discrete levels of freeze frequency and length. Using interpolation on these levels, Claeys et al. proposed the following continuous function to measure the impact of freezes ϕ [12]:

$$\phi = \frac{7}{8} max\left(\frac{ln(F_{freq})}{6} + 1, 0\right) + \frac{1}{8}\left(\frac{min(FT_{avg}, 15)}{15}\right) \quad (4)$$

In this equation, F_{freq} and FT_{avg} represent the frequency of freezes and the average length of freezes respectively. An estimation of the MOS is now possible through the combination of the normalized, average quality level μ, its standard deviation σ and the impact of freezes ϕ:

$$\text{MOS}_{est} = \max(5.67\,\mu - 6.72\,\sigma - 4.95\,\phi + 0.17, 0) \quad (5)$$

All coefficients have been tuned by De Vriendt et al. [14] and Claeys et al. [12]. Note that the theoretical range of the MOS estimation is $[0; 5.84]$, although in practice a range of $[0; 5.06]$ is observed. For the defined reward function, the constant c should thus simply comply to $c \leq 5.84$.

The second considered evaluation criterion is the average buffer filling, as the goal in this paper is to actively lower this value. In the following section, results are reported considering both the average MOS and the average buffer filling.

C. Rationale

In Section IV it was pointed out that, by selecting an appropriate value for the trade-off parameter ψ in the reward function, the agent's behaviour should be different in a variable and a fixed bandwidth scenario. To illustrate this idea, Figure 3 shows the average MOS and buffer filling as a function of the buffer size used by the traditional QoE-RAHAS algorithm. In a fixed bandwidth scenario, the average MOS is already acceptable for a buffer size of 4 seconds, and does not increase significantly when a larger buffer is used. This is not the case in a variable bandwidth scenario, where a significant increase is observed between results for a buffer size of 4 and a buffer size of 10 seconds. As the average buffer filling increases linearly as a function of the buffer size, one can indeed expect the agent to learn to use a configuration with a high buffer size when the available bandwidth is variable, and a configuration with a low buffer size when the available bandwidth is fixed.

D. Parameter Configuration Details

The presented approach comes with a number of algorithm-specific and Q-learning parameters. An extensive analysis was conducted, evaluating the impact of parameter values on the algorithm's performance. The set of evaluated algorithm parameter configurations is presented in Table IV. The best results are obtained for a decision window of $D = 15$ and a reward window of $W = 10$, which is explained as follows. When the decision interval is too small, resulting behaviour will largely consist of transient effects. Indeed, changing the buffer size from 6 to 12 seconds in MSS for instance, the buffer filling level will slowly increase by lowering the selected quality level. In this case, the algorithm will not be able to make reliable and funded decisions. When the decision interval is too high, the algorithm's responsiveness to changing network conditions decreases, again leading to a lower overall performance. As for the reward window, the number of considered video segments should be lower than for the decision window. In this way, results for the previous configuration and transient effects are discarded, so that only significant

Parameter	Evaluated values
Decision interval D	5, 10, 15, 20
Reward window W	5, 10, 15, 20
Trade-off parameter ψ	$0.05\,i, i \in [0; 20]$

Table IV: Evaluated algorithm parameter configurations.

Parameter	Evaluated values
Learning rate α	0.1, 0.3, 0.5, 0.7, 0.9
Discount factor γ	0.1, 0.3, 0.5, 0.7, 0.9
Eligibility trace-decay λ	0.1, 0.5, 0.9
Softmax inverse temperature β	0.1, 0.5, 1.0, 5.0

Table V: Evaluated Q-learning parameter configurations.

measurements are considered in the reward function. For the trade-off parameter ψ, finally, the value of 0.75 turns out to render the best results.

As for the Q-learning parameters, the set of evaluated configurations is presented in Table V. Preliminary evaluations showed that the best results are obtained when low values for the learning rate and discount factor are used. As the properties of the available bandwidth are continuously changing, a low learning rate is recommended to slowly converge to the optimal solution. This also explains why a low discount factor is preferred: future rewards cannot actively be controlled by the agent, and its impact should thus be limited. For this reason, a configuration with $\alpha = 0.1$ and $\gamma = 0.1$ turns out to render the best results. Note that a low discount factor causes the system to be rather insensitive to the eligibility trace-decay, as the decay is strongly accelerated in this case. As such, a value of 0.5 for λ is decided upon. For the Softmax inverse temperature, finally, the best configuration is more likely to be selected when higher values are used. For this reason, evaluations showed that a configuration with $\beta = 5$ tends to lead to the best results.

E. Detailed Results

Using the most appropriate parameter configuration, results were thoroughly evaluated. In the learning phase, up to 3000 episodes of the *Big Buck Bunny* video trace were simulated. Afterwards, average results were evaluated both over 50 variable and fixed bandwidth traces. In this way, the average MOS and buffer filling can fairly be compared with results for the traditional MSS and QoE-RAHAS heuristics.

1) MSS Heuristic

Results for the traditional MSS heuristic and the proposed Q-learning algorithm are presented in Figure 4. Using the learning-based approach, an average MOS of 3.55 and a buffer filling of 4.90 seconds are observed in a fixed bandwidth scenario. To achieve a comparable MOS with a fixed parameter configuration, a buffer size of at least 8 seconds should be used. Compared to results for this configuration, the proposed approach achieves a significantly lower buffer filling (-11.5%), while the average MOS is only slightly affected (-2.3%). In a variable bandwidth scenario, an average MOS of 2.73 and a buffer filling of 5.72 seconds are observed. Compared to results with a fixed buffer size of 8 seconds, the proposed approach comes with a comparable, yet somewhat higher buffer filling ($+6.7\%$) and average MOS ($+0.8\%$). These results lead us to conclude that, using the proposed learning-based approach, the agent indeed learns to actively reduce the buffer filling when the perceived bandwidth is fixed, while increasing the buffer filling when high variations in the perceived bandwidth occur. This was the intended target, yet on average the buffer filling is reduced by merely 2.5%, while the loss in terms of the MOS is limited to 1.0%.

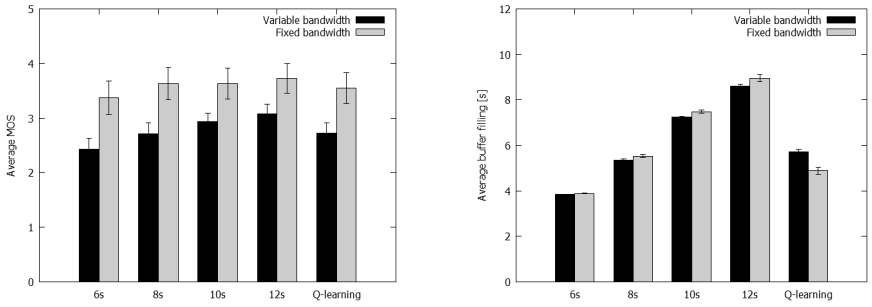
2) QoE-RAHAS Heuristic

Results for the traditional QoE-RAHAS heuristic and the proposed Q-learning algorithm are presented in Figure 4 as well. Using the learning-based approach, an average MOS of 3.72 and a buffer filling of 2.88 seconds are observed in a fixed bandwidth scenario. To again achieve a comparable MOS with a fixed parameter configuration, a buffer size of at least 6 seconds should be used. Compared to results for this configuration, the proposed approach again achieves a significantly lower buffer filling (-27.4%), while the average MOS is only slightly affected (-4.5%). In a variable bandwidth scenario, an average MOS of 2.86 and a buffer filling of 4.38 seconds are observed. Compared to results with a fixed buffer size of 6 seconds, the proposed approach comes with a higher buffer filling ($+10.9\%$) and a comparable MOS (-5.2%). Again this leads us to conclude that the agent indeed learns to actively reduce the buffer filling when the perceived bandwidth is fixed, while increasing the buffer filling when high variations in the perceived bandwidth occur. On average, the observed buffer filling is reduced by 8.3%, while the loss in terms of the MOS corresponds to 4.8%.

While actions are chosen at random when the agent is first introduced to the environment, parameter configurations are more intelligently selected as the learning phase progresses. To illustrate the agent's behaviour once the learning phase is completed, Figure 5 shows the selected actions for the QoE-RAHAS heuristic over three different episodes of the video trace. Network conditions are dynamically changing, simulating scenarios where the perceived bandwidth is either variable or fixed. When the perceived bandwidth is variable, a configuration with a higher buffer size is preferred by the agent. In this way, an attempt is made to increase the average buffer filling, actively avoiding buffer starvations and preventing lower quality levels from being selected. When the perceived bandwidth is fixed however, configurations with a lower buffer size are favoured by the agent. In this way, the average buffer filling can actively be reduced, while still providing the user with a comparable level of QoE.

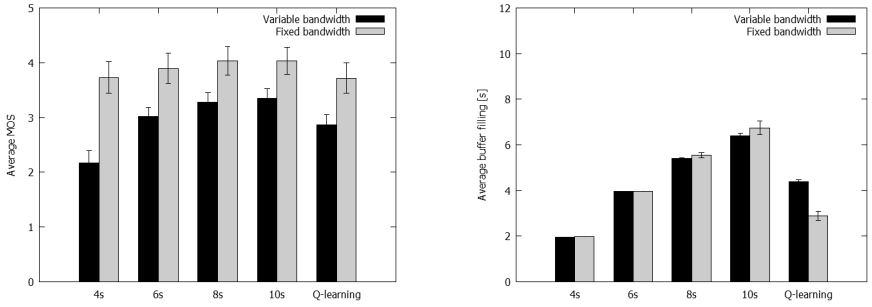
3) Comparison

While the Q-learning approach seems promising for both heuristics, Figure 4 shows that results for QoE-RAHAS are in fact superior to those for MSS. When the perceived bandwidth is fixed, the average MOS is 4.8% higher, while the average buffer filling is 41.2% lower. The same trend is observed for a



(a) Average MOS [MSS]

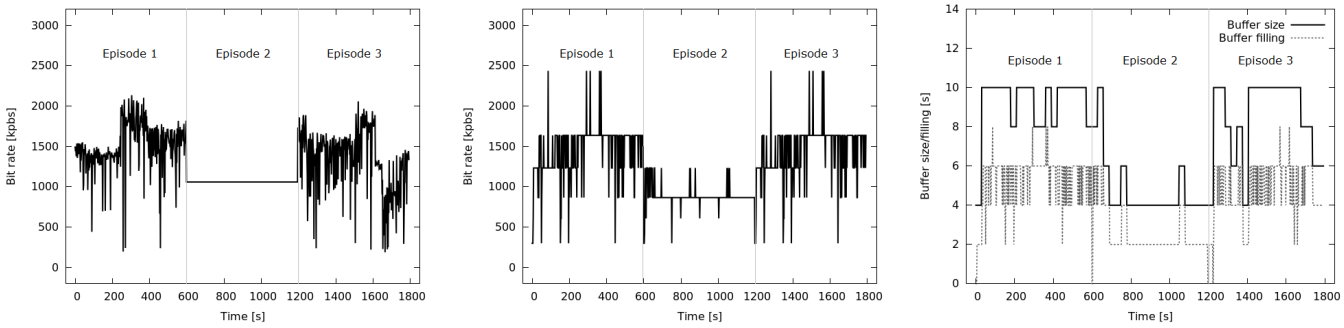
(b) Average buffer filling [MSS]



(c) Average MOS [QoE-RAHAS]

(d) Average buffer filling [QoE-RAHAS]

Figure 4: Results for the MSS and the QoE-RAHAS algorithms, using fixed parameter configurations and the proposed Q-learning solution. A label of "xs" corresponds to a parameter configuration with a fixed buffer size of x seconds and most appropriate threshold values.



(a) Available bandwidth over time

(b) Selected quality level over time

(c) Resulting buffer size/filling over time

Figure 5: The agent's decision making under changing network conditions, for the proposed QoE-RAHAS approach. Note that three separate episodes of the video trace are streamed, explaining the empty buffer each 598 seconds.

variable bandwidth scenario, with an increase for the average MOS of 4.9% and a decrease of 23.4% for the average buffer filling. This indicates that the QoE-RAHAS approach is more suitable to provide both a high QoE and a low average buffer filling. It is worth noticing that results for the Q-learning approach for MSS are even outperformed by the traditional QoE-RAHAS algorithm, when a fixed buffer size of 6 seconds is used: both a higher MOS ($+10.6\%$ and $+8.9\%$) and a lower average buffer filling (-31.0% and -19.0%) are observed in a variable and a fixed bandwidth scenario respectively.

There are three reasons for this significant difference between the two Q-learning approaches, which are in fact all related to the quality selection process. First, the average MOS is higher because decisions in the QoE-RAHAS heuristic are based on the considered QoE model. Taking into account the requested quality level over a moving window allows the next quality level to be requested in a more reasoned way. Second, the quality level is not necessarily increased one step at a time. This means that the heuristic can react faster to variations in the perceived bandwidth, again leading to a higher average MOS. Third, the heuristic is capable of dealing with a lower buffer filling. This is because the heuristic tries to respect the panic threshold at all times, requesting a quality level which will most likely not cause buffer starvation. In this way, a lower average buffer filling is achieved, while freezes are less likely to occur. Based on this analysis, we conclude that the proposed Q-learning approach for the QoE-RAHAS algorithm most closely fulfils the intended objectives of this paper.

VI. Related Work

Few examples of RL techniques in the area of HAS exist in literature. Early research focused on adaptive streaming techniques that target server or network side solutions, in order to provide a certain Quality of Service (QoS) in adaptive streaming services. Fei et al. studied the issues of QoS provisioning in adaptive multimedia delivery in mobile networks [16]. Call admission control and bandwidth adaptation are formulated as a constrained MDP and solved using Q-learning. Charvillat et al. presented a dynamic adaptation agent which considers both user behaviour and context information [17]. The generic approach is used to solve a ubiquitous streaming problem in mobile networks. McClary et al. proposed a transport protocol, in which artificial neural networks are used to adapt the audio transmission rate in mobile ad-hoc networks [18]. Considered QoS variables are the perceived throughput, delay and jitter.

More recently, the concept of RL has been introduced in the HAS quality selection process. Menkovski et al. proposed the use of the SARSA(λ) technique to select the most appropriate quality level, based on the estimated bandwidth, the buffer filling and the position in the video stream [19]. Even though convergence is shown with respect to the QoE, performance is not compared with respect to other existing HAS implementations. Claeys et al. proposed the use of Q-learning to select the next quality level, based on the estimated bandwidth and the buffer filling [12]. Results show that the client is able to outperform deterministic algorithms such as MSS in several network environments. In a multi-client scenario, Petrangeli et al. suggested an approach in which each client learns to select the most appropriate quality level, maximizing a reward based both on its own QoE and on the QoE perceived by other clients [20]. To this end, a coordination proxy estimates all perceived rewards and generates a global signal that is sent periodically to all clients. Without explicit communication among agents, the algorithm is able to outperform both MSS and the algorithm proposed by Claeys et al. in a multi-client scenario. While the above works all propose a learning-based rate adaptation heuristic, we introduce a learning-based layer on top of pre-existing rate adaptation heuristics. Furthermore, our focus is on both the QoE and the average buffer filling, while the focus in [12] and [20] is on the QoE only.

VII. Conclusions

In this paper, the concept of RL was introduced in two existing rate adaptation heuristics to adaptively change the parameter configuration according to certain bandwidth conditions. Using the proposed learning-based approach, the buffer filling for the MSS heuristic is on average reduced by 2.5%, while the loss in terms of the MOS is limited to 1.0%. For QoE-RAHAS, the average buffer filling is reduced by 8.3%, while the loss in terms of the MOS is limited to 4.8%. Although these results could definitely be improved, we showed that the agent is capable of intelligently changing its parameter configuration according to network conditions. Future work will focus on extending the approach to a multi-client scenario, where multi-agent RL techniques can be applied.

Acknowledgement

This work was partly funded by FLAMINGO, a Network of Excellence project (ICT-318488) supported by the European Commission under its Seventh Framework Programme. Jeroen van der Hooft and Maxim Claeys are funded by grant of the Agency for Innovation by Science and Technology in Flanders (IWT).

References

[1] Sandvine. (2013) Global Internet Phenomena Report. [Online]. Available: http://www.sandvine.com/news/global_broadband_trends.asp

[2] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[3] Microsoft. (2010) IIS Smooth Streaming Technical Overview. [Online]. Available: http://www.microsoft.com/en-us/download/details.aspx?id=\17678

[4] S. Petrangeli, J. Famaey, M. Claeys, and F. De Turck, "A QoE-Driven Rate Adaptation Heuristic for Enhanced Adaptive Video Streaming," Ghent University - iMinds, Department of Information Technology, Tech. Rep., 2014. [Online]. Available: http://users.ugent.be/~spetrang/QoE-RAHAS.pdf

[5] Apple. (2009) Internet Draft - HTTP Live Streaming. [Online]. Available: http://tools.ietf.org/html/draft-pantos-http-live-streaming-01

[6] Adobe. (2010) Adobe HTTP Dynamic Streaming. [Online]. Available: http://help.adobe.com/en_US/HTTPStreaming/1.0/Using/index.html

[7] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP: Standards and Design Principles," in *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, 2011, pp. 133–144.

[8] Microsoft. [Online]. Available: https://slextensions.svn.codeplex.com/\svn/trunk/SLExtensions/AdaptiveStreaming/

[9] C. Watkins, "Learning from Delayed Rewards," Ph.D. dissertation, King's College, 1989.

[10] G. Carneiro. (2010) NS-3: Network Simulator 3. [Online]. Available: http://www.nsnam.org/tutorials/NS-3-LABMEETING-1.pdf

[11] N. Bouten, S. Latré, J. Famaey, W. Van Leekwijck, and F. De Turck, "Minimizing the Impact of Delay on Live SVC-Based HTTP Adaptive Streaming Services," in *2013 IFIP/IEEE International Symposium on Integrated Network Management*, 2013, pp. 1399–1404.

[12] M. Claeys, S. Latré, J. Famaey, T. Wu, W. Van Leekwijck, and F. De Turck, "Design and Optimisation of a (FA)Q-learning-Based HTTP Adaptive Streaming Client," *Connection Science*, vol. 26, no. 1, pp. 25–43, 2014.

[13] International Telecommunication Union. (1996) ITU-T P.800. Methods for Subjective Determination of Transmission Quality - Series P: Telephone Transmission Quality; Methods for Objective and Subjective Assessment of Quality. [Online]. Available: http://electronics.ihs.com/document/abstract/DFGPCAAAAAAAAAAA

[14] J. De Vriendt, D. De Vleeschauwer, and D. Robinson, "Model for Estimating QoE of Video Delivered Using HTTP Adaptive Streaming," in *2013 IFIP/IEEE International Symposium on Integrated Network Management*. IEEE, 2013, pp. 1288–1293.

[15] R. Mok, E. Chan, and R. Chang, "Measuring the Quality of Experience of HTTP Video Streaming," in *2011 IFIP/IEEE International Symposium on Integrated Network Management*, 2011, pp. 485–492.

[16] F. Yu, V. Wong, and V. Leung, "Efficient QoS Provisioning for Adaptive Multimedia in Mobile Communication Networks by Reinforcement Learning," *Mobile Network Applications*, vol. 11, no. 1, pp. 101–110, 2006.

[17] V. Charvillat and R. Grigoras, "Reinforcement Learning for Dynamic Multimedia Adaptation," *Journal of Network and Computer Applications*, vol. 30, no. 3, pp. 1034–1058, 2007.

[18] D. McClary, V. Syrotiuk, and V. Lecuire, "Adaptive Audio Streaming in Mobile Ad Hoc Networks Using Neural Networks," *Ad Hoc Networks*, vol. 6, no. 4, pp. 524–538, 2008.

[19] V. Menkovski and A. Liotta, "Intelligent Control for Adaptive Video Streaming," in *2013 IEEE International Conference on Consumer Electronics*, 2013, pp. 127–128.

[20] S. Petrangeli, M. Claeys, S. Latré, J. Famaey, and F. De Turck, "A multi-agent q-learning-based framework for achieving fairness in http adaptive streaming," in *2014 IEEE Network Operations and Management Symposium*, 2014, pp. 1–9.