Rate adaptation with Bayesian attractor model for MPEG-DASH

Masayoshi Iwamoto
Graduate School of Information Science and Technology, Osaka University
Osaka, Japan
m-iwamoto@ist.osaka-u.ac.jp

Tatsuya Otoshi
Graduate School of Information Science and Technology, Osaka University
Osaka, Japan
t-otoshi@ist.osaka-u.ac.jp

Daichi Kominami
Graduate School of Economics, Osaka University
Osaka, Japan
d-kominami@econ.osaka-u.ac.jp

Masayuki Murata
Graduate School of Information Science and Technology, Osaka University
Osaka, Japan
murata@ist.osaka-u.ac.jp

***Abstract*—Recently, over-the-top video service providers focus on the quality of experience (QoE) as an important factor when they provide video content. Today, most video streaming service providers, such as Youtube and Netflix, provide video content to users with adaptive bitrate (ABR) control techniques for increasing the user QoE. To maximize the QoE under a fluctuating network condition, in this paper, we propose an ABR algorithm using the Bayesian attractor model, which models cognition and decision making of the human brain, as the name suggests, according to the Bayesian inference. Simulation results show that our proposed method achieves a higher video bitrate with less video quality switching to improve the user QoE compared to the methods even in the situation where network available bandwidth greatly fluctuates.**

***Index Terms*—Adaptive bitrate, video streaming, human brain, decision making**

I. INTRODUCTION

Most people nowadays carry mobile devices to access information on the Internet and use various services. Also, the amount of video traffic is increasing at a drastic pace. Cisco VNI [1] forecast that global mobile data traffic will grow seven-fold over five years from 2016 to 2021, and video traffic will account for 78% of the world's mobile data traffic by 2021. This increase in mobile traffic intensifies the degree of fluctuation in mobile traffic, and the range of fluctuation in the quality of service (QoS) level, which can be represented by the throughput, delay time, and packet loss rate, is thus increasing.

Although a QoS guarantee is an objective of network service providers, it faces many challenges because there are various factors destabilizing the QoS, such as the inherent variability in signal strength, interference, noise, and user mobility [2] in addition to the increase in mobile traffic. These factors make it harder to guarantee the QoS of mobile devices.

From the viewpoint of over-the-top video service providers, the quality of experience (QoE) is attracting attention as an important factor when they provide video content. There are several reasons for this. One reason is the diversification of user context in the use of mobile devices; i.e., many types of devices, services, and communications. The QoE is a concept of subjectively perceived quality that was introduced in [3], and techniques that maximize a mobile user's QoE are essential.

To maximize the user QoE, it is desirable for users to watch the highest-quality version of a video, which means the highest video bitrate, frame rate, encoding quality, and audio bitrate. However, it is not always possible to provide such a high-quality video because the bandwidth available on the network connection between the user's device and the video server is not always enough to accommodate the highest-quality video. If a user views a high-quality video on a poor network connection, the video freezes for *rebuffering* [4]. In addition, choosing a video with a bitrate that is too high may cause unnecessary decoding delays. Depending on the screen display resolution of the user's device, high video quality would not so much satisfy the user. These rebuffering and delays degrade the user's QoE [5].

Today, most video streaming service providers, such as Youtube and Netflix, provide video content to users with adaptive bitrate control techniques according to the user and QoS context. MPEG-DASH (Dynamic Adaptive Streaming over HTTP) [6] is one of the standards of HTTP Adaptive Streaming (HAS). Using DASH, the video player can dynamically switch among quality levels/representations, which means different bitrate levels, of the user's watching video while viewing in accordance with the QoS and the current quality of video. In DASH systems, an original video content is encoded into multiple encoded videos at different bitrates, and each encoded video is then partitioned into videos of a fixed length (generally a few seconds), which are called *chunks* or *segments* (where we use the term *segments*). Every finishing download of a segment, a client selects a next segment to download according to an adaptive bitrate (ABR) algorithm

978-1-7281-0554-3/19/$31.00

that is implemented generally in an application layer of the client.

Recent research has proposed various ABR algorithms for increasing the user QoE. Generally, ABR algorithms estimate the instantaneous network quality and use it as a decision criterion. However, as mentioned above, network conditions can fluctuate over time and are unstable for mobile devices, and the accurate estimation of network conditions is therefore difficult. This results in degrading the user QoE because client applications (1) cannot fully utilize network resources through ABR algorithms, (2) frequently switch the bitrate in response to fluctuating decisions made by an ABR algorithm, and (3) request a higher bitrate than the network bandwidth, which leads to video rebuffering.

To solve the above-mentioned problem of maximizing the QoE under a fluctuating QoS, in this paper, we propose an ABR algorithm using the Bayesian attractor model (BAM [7]), which models cognition and decision making of the human brain, as the name suggests, according to the Bayesian inference. The BAM is divided into two models; one is a cognitive process model where observed information is accumulated and the posterior probability of classification of the observation into predefined categories is obtained. The other is the decision-making process model that determines one classification according to its posterior probability. The BAM allows the control of a trade-off between accurate and rapid estimation for dynamically changing observed information.

In our proposal, the BAM is implemented in the client MPEG-DASH video streaming application. In the cognitive process model, the BAM perceives information available in the application layer and estimates the network and application conditions. In the decision-making process model, the BAM selects a video bitrate according to the results of the cognition process model. Here, the selection algorithm of a video bitrate that realizes a high user QoE is required. However, the factors that improve the QoE differ person by person. Therefore, in our proposal, we assume that this selection algorithm differs person by person, and the user can choose a selection algorithm that suits them.

The remainder of the paper is organized as follows. Section II provides existing research on QoE metrics, HAS techniques, and ABR algorithms. Section III gives a detailed description of the BAM. Section IV explains how to apply the BAM to an ABR and presents our proposed algorithm. Section V evaluates the performance of the proposed algorithm and compares it with that of two ABR algorithms, namely TCP-Like AIMD Based ABR algorithm [8] and BOLA [4] which is now part of an experimental algorithm integrated in dash.js [9]. In Section VI, we offer concluding remarks and refer to future challenges.

II. Related work

A. QoE metrics

The QoE is a measure of the degree of user satisfaction with a service. Past studies on the QoE of a video streaming service show that the QoE is strongly correlated with video player events (e.g., rebuffering, a change in video quality, and start-up delay). Some papers describe that the QoE relies on the start-up delay (e.g., [10], [11]) while other papers show that the QoE relies on rebuffering [10]–[12], the played bitrate [5], [13], and the bitrate change ratio [5], [12].

There are also studies that estimate the user's QoE using important factors of the QoE. Reference [13], for example, presents a user experience model that can quantitatively measure the QoE of the ABR video streaming service and designs the model with three factors of the QoE, the initial (start-up) delay, stalling (rebuffering), and variation of video quality. As a wide survey of the QoE for video streaming in real society, the authors of [12] developed a browser plug-in for YouTube, named YouSlow, and collected and analyzed information on video player events and the user's video abandonment. The results of YouSlow analysis show that the bitrate changes ratio (average amplitude of bitrate changes over playback time) and rebuffering ratio (average rebuffering time over playback time) are correlated to the user's video abandonment. Regarding the bitrate change ratio, it is reported that even when the bitrate was improved, a high bitrate change ratio led to the user abandoning the video. Although the reasons are not clarified in [12], this may be because users prefer the stability of the bitrate to higher video quality.

B. MPEG-DASH

HAS is widely used for video streaming services. For instance, it is implemented in Microsoft Silverlight Smooth Streaming (MSS) by Microsoft, HTTP Live Streaming (HLS) by Apple, and Adobe HTTP Dynamic Streaming (HDS) by Adobe Systems. As a standard for HAS, DASH [6] was issued by MPEG in 2012. DASH aims to provide a smooth video streaming service to users corresponding to network conditions and types of client device. In DASH systems, video content is encoded into multiple versions at different bitrates, and each encoded video is then partitioned into videos of fixed length $segments$. Segments are stored on the DASH server. When a DASH streaming session starts, the DASH server provides the Media Presentation Description (MPD) to the DASH client. The MPD is an index file that describes media metadata of the different audio and video bitrates available to the client. To play video content, the client first obtains the MPD and then requests segments in the desired bitrate according to MPD information, network conditions, and types of the client device. The MPD and segments are delivered using HTTP. Because the client sends HTTP requests for each segment, the video player can switch to video with different bitrates for each segment. In this way, ABR streaming is realized in DASH.

C. ABR algorithms

Various ABR algorithms have been proposed and they can be broadly classified into three categories according to the feedback information they use [14]: *throughput-based* [8], [15], *buffer-based* [4], [16], and *hybrid/control theory- based* [17], [18]. Because ABR algorithms work in the application layer of the client device, they generally decide the appropriate video bitrate for the next segment

to be downloaded, according to information available to the application layer of the client (e.g., playback buffer occupancy, and TCP throughput estimated by the application layer). Here, it is difficult to estimate accurate network conditions because network conditions can fluctuate over time and vary across environments. Inaccurate estimation can lead to inappropriate bitrate selections, resulting in lower video quality or frequent bitrate switching or rebuffering.

Each time the client sends an HTTP request, it has to select an appropriate video bitrate according to information available to it. This selection of bitrates is made by an ABR algorithm implemented in the client device. The general goals of the ABR algorithm are as follows [19].

1) Avoid playback interruptions due to buffer underruns (rebuffering).
2) Maximize the video quality.
3) Minimize the number of video quality shifts.
4) Minimize the time between the request for a new video by the user and starting to play the video.

However, there are trade-off relationships among these goals as the authors of [19] mentioned. For instance, it is always possible to minimize the number of interruptions by selecting the lowest video bitrate to achieve goal 1, but goal 2 then cannot be achieved. To achieve goal 2, the ABR algorithm can switch video bitrate by reacting to the smallest changes in the network bandwidth. This causes frequent video quality shifts, and goal 3 cannot be achieved. Goal 4 is also a trade-off with goal 2 because selecting the lowest video bitrate at the start minimizes the start-up time but degrades the video quality. It is therefore necessary for the ABR algorithm to maximize a multi-objective function for these multiple goals. However, factors for maximizing the user QoE differ among people. It is significant to provide appropriate ABR algorithms for person by person.

III. Bayesian attractor model

This section explains the Bayesian attractor model (BAM) proposed in [7] and our extension of the BAM. The BAM models a human's brain, which accumulates sensing information of the external field and makes a decision using the Bayesian inference framework.

The BAM has a decision state $\mathbf{z}$ as its internal state and updates $\mathbf{z}$ according to an internal generative model that has stable fixed points (*attractors*). Note that the authors of [7] used winner-takes-all dynamics for the generative model of the BAM. Internally, the BAM has several decision alternatives, and each alternative i corresponds to each attractor ϕ_i. Since $\mathbf{z}$ is a hidden variable, in the cognitive process model, the BAM estimates the posterior density function of $\mathbf{z}$ by using the Bayesian inference. In the decision-making process model, the BAM checks whether a probability density when $\mathbf{z} = \phi_i$ exceeds a threshold value.

The cognitive process model discriminates attractors by comparing the perceived information with past experience and memory. Past experience and memory are linked to K attractors. For more detail, the state vector of ϕ_i $(i = 1 \cdots K)$, is associated with past experience and memory by a feature vector μ_i. As mentioned above, the generative model of the BAM uses a nonlinear dynamics with these K attractors $(\phi_1 \cdots \phi_K)$. In the BAM, decision state $\mathbf{z}$ is updated by the following equation.

$$\mathbf{z}_t \quad = \quad \mathbf{z}_{t-\Delta} + \Delta g(\mathbf{z}_{t-\Delta}) + \sqrt{\Delta}\mathbf{w}_t, \tag{1}$$

where $\mathbf{z}$ is updated from one time step to the next and $g(*)$ denotes the attractor dynamics [20], Δ means the update interval of the dynamics, $\mathbf{w}_t$ is a white noise following the normal distribution $\mathcal{N}(0, \mathbf{Q})$, where $\mathbf{Q} = (q^2/\Delta) \cdot \mathbf{I}$ is the variance–covariance matrix of the noise, and q is a parameter representing dynamics uncertainty. If there is no noise in the dynamics (namely, $q = 0$), $\mathbf{z}$ is drawn into one of the fixed points ϕ_i by repeating the update. The dynamics uncertainty represents the amount of noise with which the decision maker expects the state variable to be changed, which is interpreted as the tendency for state variables to switch between fixed points.

In the BAM, it is assumed that an observation, denoted by a vector $\mathbf{x}_t$, are generated corresponding to one of the attractors, which is represented by Eq. (2).

$$\mathbf{x_t} = \mathbf{M} \cdot \sigma(\mathbf{z_t}) + \mathbf{v_t}, \tag{2}$$

where $\mathbf{M}$ is a feature matrix of $[\mu_1, \mu_2, ..., \mu_K]$, and a feature vector μ_i links ϕ_i and memory. $\sigma(*)$ is a sigmoid function that maps all values $z_j \in \mathbf{z}$ to values between 0 and 1. Owing to the winner-takes-all dynamics of $\mathbf{z}$, the fixed point ϕ_i is mapped to a vector $\sigma(\phi_i)$, where one element is approximately 1 and the other elements are approximately zero. The linear combination $\mathbf{M} \cdot \sigma(\phi_i)$ thus becomes almost μ_i. Note that μ_i is a feature vector of the same dimension as an observation values $\mathbf{x}$. v_t is a white noise following the normal distribution $\mathcal{N}(0, \mathbf{R})$, where $\mathbf{R} = r^2 \cdot \mathbf{I}$ is the variance–covariance matrix of the noise and r is a parameter representing sensory uncertainty. The sensory uncertainty represents the amount of noise in observations that the decision maker expects.

The BAM estimates the posterior density function of $\mathbf{z}$ from input sequences of $\mathbf{x}_t$. In the decision-making process model, the estimation of the decision state $\mathbf{z}$ according to the observation value $\mathbf{x}$ involves estimating $\mathbf{z}_t$ that gives the minimum variance of $\mathbf{x}_t$ in the Eq. (2). In [7], the unscented Kalman filter (UKF), one of a Bayesian filters, is used for this estimation. Although the UKF is developed for estimating a nonlinear generative model, due to the generative model of the BAM with strong nonlinearity such that a sigmoid function is included, it loses the accuracy of the estimation. Another algorithm that can handle a nonlinear/non-Gaussian system and can estimate the state with higher precision is therefore desirable. In this paper, the particle filter (PF) is adopted as an algorithm satisfying this condition.

Unlike the UKF, the PF supports a non-Gaussian state space model, such that a more accurate estimation can be expected in the BAM's internal model. Using the PF, the probability density function of $\mathbf{z}_t$ at time t, $P(\mathbf{z}_t|\mathbf{x}_t)$ is estimated and the

probability density $P(\mathbf{z}_t = \phi_i|\mathbf{x}_t)$ for each attractor ϕ_i is referred to as *confidence*. In the decision-making process model, when the confidence for the attractor ϕ_i, $P(\mathbf{z}_t = \phi_i|\mathbf{x}_t)$, exceeds the threshold λ, the attractor ϕ_i is finally adopted as the result of estimation. Additionally, if such ϕ_i does not exist, we will not do anything. If this threshold value is higher, estimation is more accurate but its speed is lower, and vice versa.

IV. Rate adaptation with Bayesian attractor model

A. Overview

The goal of the proposed method is to maximize the QoE of individual users in consideration of network and application conditions that change dynamically and the user's preference for video quality, by selecting appropriate bitrates of MPEG-DASH segments. In our method, the BAM runs in the client application and observes the network communication quality and video quality in the application layer. According to the observation, the BAM decides which feature vector is closest to the current observation among feature vectors designed in advance, and the video bitrate of the next segment to be downloaded is chosen according to the estimation result.

In the following sections, we explain how we realize our rate adaptive algorithm with the BAM, that is, we explain about the observation information that is input to the BAM, the design of attractors and feature-vectors in the BAM, and the bitrate control that achieves the purpose of our research.

B. Observation information

At first, as network and application conditions to be considered, we focus on the available bandwidth and the buffer occupancy. These are widely adopted metrics in ABR algorithms for DASH. An observation is performed every time the download of a segment is completed. dash.js [9] can acquire the playback buffer occupancy at the present moment. On the available bandwidth, as used in dash.js, we use the passive measurement method where an available bandwidth is calculated by dividing the segment size by the download time for it.

In our ABR algorithm, we prepare K sets of the playback buffer occupancy and the available bandwidth in advance, each of which equals μ_i. The observation information $\mathbf{x}_t$ input to the BAM at t is also a set of the buffer occupancy and the available bandwidth, and these pieces of information are acquired on the client device. From $\mathbf{x}_t$, the BAM estimates the current $\mathbf{z}_t$. When $\mathbf{z}_t$ is identified as one of the pre-specified conditions, which are represented by μ_i, our method selects an appropriate video bitrate according to the estimated condition.

C. Attractor and feature vector design

In this section, we explain how to design the attractor and feature vector of the BAM. The attractor design means to decide how many attractors are prepared, namely to decide the value of K. Since K is the number of network and application conditions we want to discriminate, we determine feature vectors. On the available bandwidth, we want to know whether it can accommodate bitrates that a client application can choose from a MPD file. Then, the number of the network condition is set to that of available encoded videos. On the buffer occupancy, we want to know if the current buffer is abundant or depleted. Then, the buffer occupancy is classified into three types, safe, transient, and risky, and the value of the buffer occupancy is represented by B_{safe}, $B_{transient}$, and B_{risky}, respectively. Thus, the number of the application conditions is three. Finally, K is calculated by multiplying the number of the network conditions and the application conditions.

D. Adaptive bitrate selection

When observation information $\mathbf{x_t}$ is input and $P(\mathbf{z}_t = \phi_i|\mathbf{x}_t)$ exceeds a threshold, the BAM refers μ_i as a current condition. Then, we design which bitrate of a next segment is to be selected and downloaded. This selection is independent of the BAM's cognition model and it is possible to realize bitrate selection suited to the preference of different users.

Bitrate selection algorithms aiming at improving the user QoE of video streaming services has to consider the average bitrate, rebuffering time, and switching frequency of the bitrate. Reference [12] reported that the more rebuffering occurs or the longer the rebuffering time is, the more users interrupt viewing. And similarly, the more frequently the video bitrate fluctuates, the more users interrupt viewing. In this paper, as an example of selection algorithms, we propose a bitrate selection algorithm with considering the preference of users who prefer less rebuffering and fewer changes in the bitrate. The selection algorithm is based on the following selection rules.

Rule 1: In case the buffer occupancy is low
: If the current bitrate is lower than the estimated available bandwidth, the current bitrate is kept. Otherwise, a two-level lower bitrate than the current one is selected (when there is no two-level lower bitrate than the current one, the lowest bitrate is selected).

Rule 2: In case the buffer occupancy is abundant
: If the current bitrate is lower than the estimated available bandwidth, a one-level higher bitrate than the current one is selected (when the current bitrate is the highest one, the current bitrate is kept). Otherwise, the current bitrate is kept.

Rule 3: In case it is neither buffered abundant nor exhausted
: If the current bitrate is equal or lower than the estimated available bandwidth, the current bitrate is kept. Otherwise, a one-level lower bitrate than the current one is selected (when the current bitrate is the lowest one, the current bitrate is kept).

An example of the feature vectors and bitrate selection rules when the number of encoded videos equals three is shown in Table I. In the table, T_1, T_2, and T_3 represent the available bandwidth (assuming $T_1 < T_2 < T_3$) corresponding to the bitrate of three encoded videos, respectively. For achieving less rebuffering and fewer changes in the bitrate, the buffer occupancy is put stress on.

TABLE I
EXAMPLE OF BAM ATTRACTORS, FEATURE VECTORS, AND BITRATE SELECTION RULES

Attractor	Available bandwidth	Buffer occupancy	Bitrate selection
ϕ_1	T_3	B_{safe}	Rule 2)
ϕ_2	T_2	B_{safe}	Rule 2)
ϕ_3	T_1	B_{safe}	Rule 2)
ϕ_4	T_3	$B_{transient}$	Rule 3)
ϕ_5	T_2	$B_{transient}$	Rule 3)
ϕ_6	T_1	$B_{transient}$	Rule 3)
ϕ_7	T_3	B_{risky}	Rule 1)
ϕ_8	T_2	B_{risky}	Rule 1)
ϕ_9	T_1	B_{risky}	Rule 1)

V. SIMULATION RESULTS

A. Simulation setup

We simulate our proposed method assuming a 5-minute movie in a situation where the available bandwidth changes dynamically. The 5-minute movie was encoded at five bitrates (0.5, 1.0, 1.5, 3.0, and 5.0 Mbps) and partitioned into 1-second segments.

For the available bandwidth to be observed in the simulation, referring to the benchmark provided by the DASH Industry Forum, the average value of available bandwidth is changed every 30 s from the start time and the average value thereof is switched to 9.0, 4.0, 2.0, 1.0, 2.0, 4.0, and 9.0 Mbps in order from the start time. Additionally, we add a different noise to each average value of available bandwidths. Each noise follows a normal distribution having an average of zero and standard deviation of $l_{noise}(\%)$ of each average value of the available bandwidth, where l_{noise} is defined as *noise level* hereafter. We change the value of the noise every second according to the distribution.

For example, we use the normal distribution where the standard deviation is $2.0 \cdot l_{noise}/100$ for a 2.0 Mbps segment. The set of the buffer occupancy embedded in each attractor, B_{risky}, $B_{transient}$, and B_{safe}, is 10, 30, and 50 s, respectively, and a set of the available bandwidth embedded in each attractor $\mathbf{T}$ corresponds to the set of bitrates available to the client; i.e., $T_1 = 0.5$, $T_2 = 1.0$, $T_3 = 1.5$, $T_4 = 3.0$, and $T_5 = 5.0$ (Mbps). Therefore, the number of the BAM's attractor K is equal to 15.

For parameters of the BAM, we set sensory uncertainty r to 0.5, dynamics uncertainty q to 0.5, and a threshold of confidence λ to 0.01.

B. Bitrate selection of proposal

We first verify the estimation of our proposed method described in Section IV by simulation. Figs. 1–3 show the result of the BAM's estimation of available bandwidth and buffer occupancy, and the transition of *confidence* for ϕ. In Fig. 3, the BAM adopts the most confident attractor among the attractors whose own confidence exceeds the threshold. the estimation of observation (Figs. 1 and 2) corresponds to the adopted attractor (Fig. 3). For example, when the confidence of ϕ_{13} exceeds the threshold at about time 60 (t), a set of available bandwidth and buffer occupancy is estimated to be that

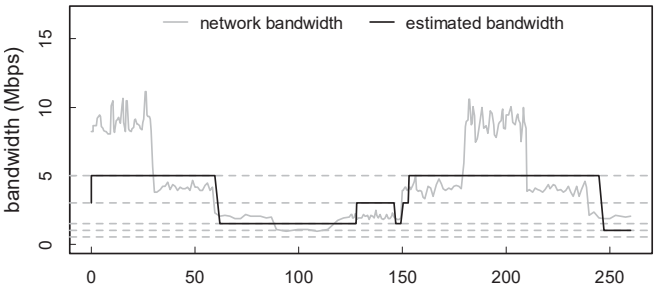


Fig. 1. Available bandwidth estimation using the BAM

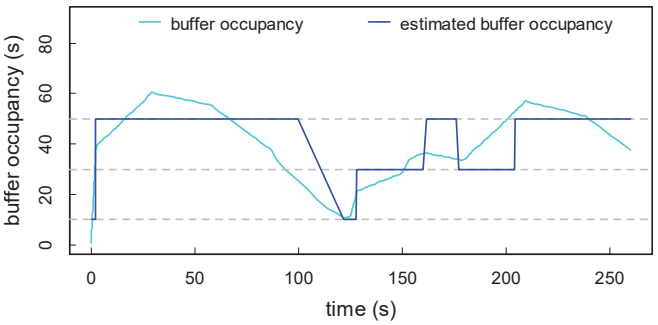


Fig. 2. Buffer occupancy estimation using the BAM

corresponding to ϕ_{13} (in this situation, available bandwidth is estimated to 3.0 Mbps, buffer occupancy is estimated to 50 s) by the BAM. These figures confirm that the estimated values are not unstable or affected by a fluctuation in observations. Meanwhile, the BAM tracks a large change in observation. It is confirmed that the state estimation appropriately performs.

We next simulate the bitrate selection of our proposed method. Fig. 4 presents that the BAM's bitrate choice is based on the estimated network and application conditions shown in Figs. 1 and 2. We can see that selected bitrates are not fluctuated and roughly follow the changes in the conditions.

C. Comparison

We now compare our proposed method with two ABR algorithms, namely, the *TCP-Like AIMD* based method (called as AIMD hereafter) [8] and *BOLA* [4]. The former method compares the segment transfer (fetch) time with the media

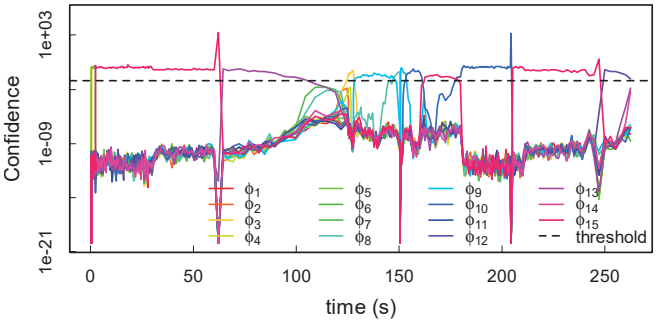


Fig. 3. Transition of the BAM confidence

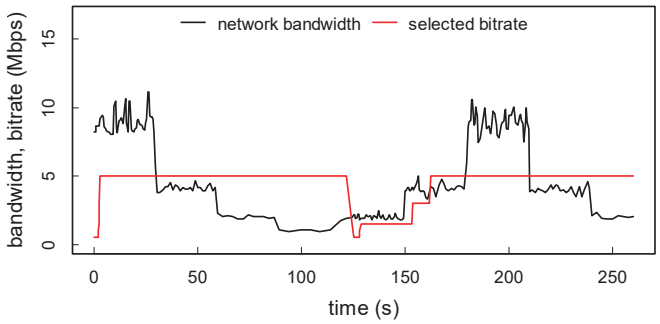
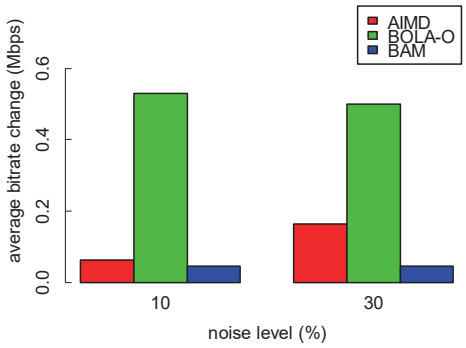


Fig. 4. Bitrate selection using the BAM

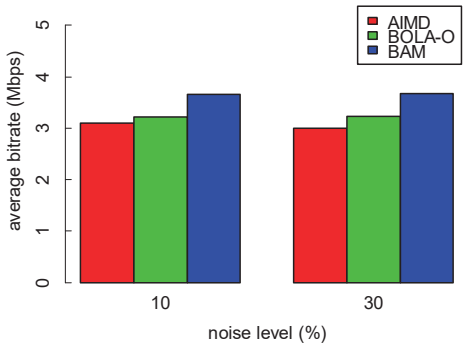
playback time contained in the segment. This method predefines two types of threshold time for switch-up and switch-down. When the fetch time exceeds the threshold for switch-up, the bitrate is step-wise switched up. When the fetch time falls below the threshold for switch-down, the bitrate is switched down aggressively to prevent rebuffering. With this method, we can compare the performance of our proposed method based on the BAM with that of bitrate control depending on only previous observations. Meanwhile, BOLA is an algorithm used in dash.js [9] that is a client-side reference implementation of MPEG-DASH, and a method expected to be widely used. We compare the performance of our proposed method with that of BOLA as BOLA is one of state-of-the-art ABR algorithms.

The simulation environment is the same as that in Section V-B. However, to evaluate the influence of noise added to the input value in terms of the robustness against noise, *noise level* is set to 10% and 30%. The average bitrate change and the average bitrate are evaluated. The average change of the bitrate is defined as the sum of the bitrate differences with the previous segment divided by the video playback time, and the evaluation results are shown in Figs. 5(a) and 5(b). First, the average bitrate change of the BOLA was larger than that of the other metods (Fig. 5(a)). Note that bitrate oscillations were a problem even in the paper that proposed BOLA [4], and the present paper proposes an improved algorithm BOLA-O that overcomes this problem. (Hereafter, the BOLA which does not mitigate the oscillations, is called BOLA-U and it is distinguished from BOLA-O) Our simulation result, however, despite mimicking the evaluation environment of the Ref. [4], BOLA-O is not taken advantage of, and the performance is almost the same as that of BOLA-U in our preliminary simulation. In both *AIMD* and the BAM, the average bitrate change is low, but the difference between *AIMD* and BAM becomes remarkable at *noise level*$= 30$, and it can be confirmed that the BAM achieves better performance.

Next, for the average bitrate (Fig. 5(b)), the BAM had the highest results at both *noise level* of $10, 30$. As a result, the BAM is superior to the compared methods in terms of average bitrate change and average bitrate, even though the minimization of average bitrate change and the maximization of average bitrate can not be realized simultaneously. The



(a) Average bitrate change



(b) Average bitrate

Fig. 5. Performance comparison

reason why the average bitrate change of the BAM is low is that our proposed method takes policy to positively keep the current bitrate according to the set of buffer occupancy and estimated available bandwidth, which is described in Rules in SectionIV-D. The reason why the average bitrate of the BAM is high is that our proposed method takes policy not to lower video bitrate if the buffer occupancy is abundant even in the situation where the network bandwidth getting lower (Rule 2 in Table I). This policy was set for the purpose of suppressing the fluctuation of the bitrate, but in the simulation environment this contributed also to improving the perfomance of average bitrate.

In addition to the attractor design policy, by the BAM's properly processing the observation and by the BAM's appropriately selecting which attractor the current situation is close to, that is, which policy should be selected, each policy works properly, and then high performance of our proposed method is realized.

Thus, from our computer simulation, we can conclude that the attractor design policy of suppression of the switching frequency of the bitrate can be realized with high accuracy under the condition where observation information greatly fluctuates.

VI. Conclusion

In this paper, for the estimation of network and application conditions, where QoS greatly fluctuates, we focused on the cognitive model of a human's brain, the Bayesian attractor model. We proposed an ABR algorithm using the BAM to realize the QoE maximization for individual users of MPEG-DASH application.

Our computer simulation showed that our proposed method can perform appropriate bitrate control, that is, it can control bitrate with less bitrate switching without greatly lowering the average bitrate compared to the two existing methods even in the situation where network available bandwidth greatly fluctuates. Our proposed method outperforms BOLA, with improvements in average bitrate of $11\%-12\%$ and improvements in average bitrate change of 91% in our computer simulation.

In the future, it is necessary to implement our proposed method in an actual video streaming application and evaluate its performance in more detail. In addition, although we evaluated the performance of our proposed method comparing with AIMD and BOLA, latest work propose other state-of-the-art algorithms. For example, paper [21] proposes an ABR algorithm using reinforcement learning and showed the algorithm outperforms BOLA. As a future task, we need to compare our proposed algorithms with such algorithms.

Acknowledgement

This research work was supported by JSPS KAKENHI Grant Numbers 18H04096 and the Ministry of Internal Affairs.

References

[1] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021," 2017. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html

[2] X. K. Zou, J. Erman, V. Gopalakrishnan, E. Halepovic, R. Jana, X. Jin, J. Rexford, and R. K. Sinha, "Can accurate predictions improve video streaming in cellular networks?" *in Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications - HotMobile '15*, pp. 57–62, 2015. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2699343.2699359 Accessed: Dec 1, 2018

[3] K. Brunnström, S. A. Beker, K. De, A. Dooms, S. Egger, M.-N. Garcia, T. Hossfeld, S. Jumisko-Pyykkö, C. Keimel, and M.-C. Larabi, "Qualinet white paper on definitions of quality of experience," 2014. [Online]. Available: http://hal.univ-nantes.fr/hal-00977812/document

[4] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in *Proceedings of IEEE INFOCOM 2016*, vol. 2016-July, 2016.

[5] M. Z. Shafiq, J. Erman, L. Ji, A. X. Liu, J. Pang, and J. Wang, "Understanding the impact of video quality on user engagement florin," *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, no. 1, pp. 367–379, 2014. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2637364.2591975

[6] "ISO/IEC 23009-1:2014 - Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats." [Online]. Available: https://www.iso.org/standard/65274.html Accessed: Dec 1, 2018

[7] S. Bitzer, J. Bruineberg, and S. J. Kiebel, "A bayesian attractor model for perceptual decision making," *PLoS Computational Biology*, 2015.

[8] C. Liu, "Rate adaptation for adaptive HTTP streaming," *Dl.Acm.Org*, pp. 169–174, 2011. [Online]. Available: https://dl.acm.org/citation.cfm?id=1943575 Accessed: Dec 3, 2018

[9] DASH Industry Forum, "Dash-Industry-Forum/dash.js," https://github.com/Dash-Industry-Forum/dash.js, accessed 2018-10-25.

[10] R. K. Mok, E. W. Chan, X. Luo, and R. K. Chang, "Inferring the QoE of HTTP video streaming from user-viewing activities," *in Proceedings of the first ACM SIGCOMM workshop on Measurements up the stack - W-MUST '11*, p. 31, 2011. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2018602.2018611Accessed: Dec 1, 2018

[11] T. D. Pessemier, K. D. Moor, W. Joseph, L. D. Marez, and L. Martens, "Quantifying the influence of rebuffering interruptions on the user ' s quality of experience during mobile," *Broadcasting, IEEE Transactions on*, vol. 59, no. 1, pp. 47–61, 2013.

[12] H. Nam, K. H. Kim, and H. Schulzrinne, "QoE matters more than QoS: Why people stop watching cat videos," *in Proceedings of IEEE INFOCOM 2016*, vol. 2016-July, 2016.

[13] Y. Liu, S. Dey, F. Ulupinar, M. Luby, and Y. Mao, "Deriving and validating user experience model for dash video streaming," *IEEE Transactions on Broadcasting*, vol. 61, no. 4, pp. 651–665, 2015.

[14] J. Kua, G. Armitage, and P. Branch, "A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1842–1866, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7884970/

[15] W. F. Estive, J. Jiang, V. Sekar, and H. Zhang, "Improving fairness , ef fi ciency , and stability in HTTP-based adaptive video streaming," vol. 22, no. 1, pp. 326–340, 2014.

[16] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: evidence from a large video streaming service," *in Proceedings of the 2014 ACM conference on SIGCOMM*, pp. 187–198, 2014.

[17] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran, "Probe and adapt: Rate adaptation for http video streaming at scale," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 719–733, 2014.

[18] A. Mansy, B. Ver Steeg, and M. Ammar, "SABRE: a client based technique for mitigating the buffer bloat effect of adaptive video flows," *in Proceedings of the 4th ACM Multimedia Systems Conference on - MMSys '13*, pp. 214–225, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2483977.2484004

[19] S. Varma, "Chapter 6 - flow control for video applications," in *Internet Congestion Control*, S. Varma, Ed. Boston: Morgan Kaufmann, 2015, pp. 173–203. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780128035832000062

[20] C. Zhang, "Storing heteroclinic cycles in hopfield-type neural networks," vol. 1, no. 1, pp. 1–11, 2009.

[21] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with Pensieve," *in Proceedings of the Conference of the ACM Special Interest Group on Data Communication - SIGCOMM '17*, pp. 197–210, 2017. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3098822.3098843, Accessed: Dec 1, 2018