Online Learning for Adaptive Video Streaming in Mobile Networks

Optimizing Adaptive Video Streaming in Mobile Networks via Online Learning

Theodoros Karagkioules, Georgios S. Paschos, Nikolaos Liakopoulos, Atillio Fiandrotti, Dimitrios Tsilimantos and Marco Cagnazzo

***Abstract*—In this paper, we propose a novel algorithm for video rate adaptation in HTTP Adaptive Streaming (HAS), based on online learning. The proposed algorithm, named *Learn2Adapt (L2A)*, is shown to provide a *robust* rate adaptation strategy which, unlike most of the state-of-the-art techniques, does not require parameter tuning, channel model assumptions or application-specific adjustments. These properties make it very suitable for mobile users, who typically experience fast variations in channel characteristics. Simulations show that *L2A* improves on the overall Quality of Experience (QoE) and in particular the average streaming rate, a result obtained independently of the channel and application scenarios.**

***Index Terms*—Adaptive video streaming, online learning**

I. Introduction

VIDEO streaming accounts nowadays for more than 75% of the global Internet traffic, a percentage projected to reach a striking 82% by 2022 [1]. To facilitate this increasing demand for video consumption, HAS has been adopted as the main technology for video streaming over the Internet, gaining significant popularity as it allows video clients to seamlessly adapt to changing network conditions and video content to be distributed over existing web service infrastructures. In 2012, the Moving Picture Expert Group (MPEG) consortium created the Dynamic Adaptive Streaming over HTTP (DASH) standard [2], that has since become the dominant HAS method.

According to DASH, the video content is first encoded at multiple quality representations (e.g., multiple resolutions to meet the diverse display capabilities of different types of user devices and multiple bitrates to adapt to network characteristics) and is made available on an HTTP server. Each quality representation is organized in smaller and independently decodable files called segments; each segment typically accounting for a few seconds of video. A client desiring to access a video, initially fetches a manifest file from the server, that contains the description of the segments (available quality representations, bitrate of each segment, etc.). Then the client deploys a *rate adaptation algorithm*, that sequentially selects the appropriate bitrate for each segment, given network conditions. Based on the bitrate indication, the client then selects the corresponding quality representation and independently requests and downloads every segment at a finite-sized queue, known as the buffer. By controlling the bitrate for each segment, rate adaptation algorithms aim at matching the video download (or streaming) rate to the channel rate. If the video consumption (or playback) rate is larger than the download rate, the buffer will deplete, eventually leading to a re-buffering event (i.e. playback interruption). In essence, a rate adaptation algorithm is an optimization solution with the objective of maximizing the streamed video bitrate, while at the same time ensuring uninterrupted and stable (i.e. minimal bitrate switches) streaming.

Constant developments in cellular networking technology, such as 4G's Long Term Evolution (LTE) or the anticipated 5G, are changing the landscape of mobile high-bandwidth multimedia applications, that are becoming fast an integral part of the mobile clients' life. In particular, the demand for *mobile* video streaming has advanced at unprecedented growth rates over the last years and is expected to reach 79% of the global mobile data traffic by 2022 [3]. Nonetheless, cellular networks are typically characterized by throughput variation, caused by radio propagation effects, such as scattering, fast fading, path loss and shadowing or handover events; that occur when a data session is transferred to another cell. Such network conditions pose significant challenges on mobile streaming solutions, where optimal bitrate adaptation over fluctuating wireless channels remains an elusive task. This paper aims to offer a novel perspective on the mobile bitrate adaptation problem, under the scope of online optimization.

As the DASH standard does not specify a particular rate adaptation algorithm, a plethora of proposed solutions exists in both scientific literature and actual industry practices. A performance evaluation of recent rate adaptation algorithms in mobile networks [4], showed that fixed-rule schemes may require parameter tuning according to the considered network or user scenario, and thus cannot generalize well beyond a certain scope of usage. In an effort to overcome this limitation, some algorithms resort to learning techniques or control theoretic approaches to attain optimal bitrate adaptation. However, their practical implementation on mobile devices may be hindered by energy-demanding architectures [5] or by the complexity of exploring the complete optimization space [6]. In this work we propose a novel rate adaptation algorithm based on Online Convex Optimization (OCO) [7], that is independent of any parameter selection concerning the streaming environment and does not require computationally heavy operations.

OCO has emerged as a very effective online learning

T. Karagkioules (*corresponding author*), N. Liakopoulos and D. Tsilimantos are with the Mathematical and Algorithmic Sciences Lab, Paris Research Center, Huawei Technologies France, e-mail: firstname.lastname@huawei.com.

G. S. Paschos was with the Mathematical and Algorithmic Sciences Lab, Paris Research Center, Huawei Technologies France while research on this work was conducted, email: gpasxos@gmail.com.

T. Karagkioules, A. Fiandrotti and M. Cagnazzo are with Télécom Paris, LTCI, Institut Polytechnique de Paris, France, e-mail: firstname.lastname@telecom-paris.fr.

framework, that is also suitable for mobile deployment, in terms of resource requirements. According to OCO, an agent learns to make sequential decisions in order to minimize an adversarial loss function, unknown at decision time. OCO is "model-free", as no statistical assumption is required, while at the same time it provides tractable feasibility and performance guarantees [8]. Having already been proposed for problems with rapidly fluctuating environments, such as cloud resource reservation management [9] and dynamic network resource allocation [10], it constitutes an appealing candidate for HAS as well. However, the application of OCO in HAS is not a straightforward task. Given its discrete decision space (set of available quality representations) and instantaneous state-dependent constraints (finite-sized buffer queue), HAS optimization does not fall directly in the class of OCO problems.

This work provides multiple contributions towards formulating the HAS optimization problem under the OCO framework. First, we model the adaptive streaming client by a learning agent, whose objective is to maximize the average bitrate of a streaming session, subject to scheduling constraints of the buffer queue. In general, the choice of the objective function is made under the assumption that higher average bitrate typically corresponds to higher video quality, when comparing the same video in the same resolution. The constraints are chosen relative to the buffer queue, since re-buffering events can significantly influence QoE [11]. Second, we fulfill the OCO requirement that both the set of decisions and constraint functions must be convex by a) allowing the agent to make decisions on the quality representation of each segment, according to a probability distribution for the bitrate and by b) deriving a set of convex constraints associated with the upper and lower bound of the finite buffer. We achieve the latter by making a relaxation to an unbounded buffer that adheres to time-averaging constraints. Third, we model the channel rate evolution by an adversary, that decides the cost of each decision a posteriori. We eventually solve the HAS optimization problem by proposing *Learn2Adapt* (*L2A*), a novel rate adaptation algorithm based on the OCO theory. In our trace-based simulations, our proposed method proves to be *robust*, providing consistently better QoE, when evaluated against reference state-of-the-art rate adaptation algorithms in a wide spectrum of possible network and streaming conditions.

II. Related work

Video rate adaptation schemes can be broadly classified according to the module that implements the rate adaptation logic. According to the DASH standard, multimedia delivery requires a server-client architecture, thus the rate adaptation module can be hosted at either of the two components. Server-side rate adaptation methods, require no cooperation from the client and resort to traffic shaping methods at the server-side alone [12]. Such approaches may produce high overhead on the server and thus make scaling with the number of clients a real challenge. Additionally, network-assisted rate adaptation methods have also been proposed, that allow HAS clients to take network information into consideration for optimizing the rate adaptation process [13], [14]. Nonetheless, most of the proposed rate adaptation schemes, reside at the client-side, where bitrate decisions are made according to either network or application-level information, a combination of both, or even cross-layer metrics. In the following, we focus our analysis on such client-side approaches, as they are more relevant and thus comparable to the proposed method herein.

Primarily, heuristic approaches have been proposed for client-side rate adaptation, that can be mainly classified into three categories according to the considered input. First, throughput-based methods estimate the available channel rate to decide on the bitrate of the streamed video. For instance, Li et al. [15] propose, *PANDA*, a rate adaptation module that uses a moving average filter to estimate the available throughput and schedules the download of every segment, in a way that reduces bitrate oscillations, particularly in scenarios with multiple clients. A similar throughput-based strategy, called *FESTIVE* [16], focuses primarily on fairness amongst all clients. Second, buffer-based methods use application-level signals, such as the instantaneous buffer level to perform the adaptation. A notable such method comes from Huang et al. [17], who propose *BBA*; a mapping between instantaneous buffer values to video bitrate levels. Third, hybrid methods may use a combination of inputs. In that direction, Kim et al. [18] propose *XMAS*, a hybrid method that deploys a traffic shaping scheme, based on both throughput estimates and playback buffer levels, while Xie et al. [19] propose *piStream*; a physical-layer informed rate adaptation strategy. Lately, new Smartphone devices have enabled the fusion of multiple sensor readings to infer context in the mobile client's environment. To this end, Mekki et al. [20] solicit incorporating a user's inferred location into the decision process.

Recently, there has been a shift in the scientific literature, in regard to the methods used in the rate adaptation design; primarily towards optimization and control theoretic approaches. Most notably, Spiteri et al. [21] formulate rate adaptation as a utility maximization problem and devise *BOLA*, an online control algorithm, that makes use of the instantaneous buffer occupancy. Also in the direction of control-theoretic schemes, *MPC*, by Yin et al. [6], combines buffer occupancy and throughput predictions for optimal rate adaptation.

In regard to rate adaptation methods based on optimization and in particular on dynamic programming, Zhou et al. [22], propose *mDASH* and formulate the rate adaptation logic as a Markov Decision Process (MDP) where the buffer size, bandwidth conditions and bitrate stability are taken as Markov state variables. Similarly, Bokani et al. [23] model the rate adaptation logic as an MDP problem as well and incorporate mobility by including vehicular environments. Some of the main drawbacks of MDP-based solutions are computational load and the need to know the statistics of the network and video content in advance.

Model-free Reinforcement Learning (RL) approaches, such as Q-Learning (QL), have also been investigated for the design of rate adaptation methods. Claeys et al. [24] propose a QL-based HAS client, allowing dynamical adjustment of the streaming behavior to the perceived network state. While QL approaches provably converge to the optimal policy, provided that their parameters are chosen correctly, the convergence

speed becomes an issue when trying to cope with previously unseen channel or video content patterns.

Lately, Deep-Learning (DL) approaches have also been proposed for rate adaptation, presenting promising merits in both accuracy and convergence of bitrate decisions. *Pensieve* [25] is a rate adaptation DL framework that does not rely on pre-programmed models or assumptions about the environment, but instead gradually learns the best policy for bitrate decisions through observation and experience. Another DL approach is called *D-DASH* [5], that combines DL and RL mechanisms and achieves a good trade-off between policy optimality and convergence speed during the decision process. Nonetheless, the deployment of DL in mobile devices is typically associated with high computational and energy demands, especially during training phases and thus external (hardware) resources may be required to assist in the rate adaptation process.

Huang et al. [26] have recently explored combinatorial optimization for rate adaptation and have proposed *Hindsight*, a near-optimal, linear-time and linear-space greedy algorithm.

During our literature review, we have identified a requirement for a rate adaptation approach, that is not only scalable to the number of clients, but also light in computation and that does not rely on any modelling assumptions. We attempt to fulfill these prerequisites with our novel method proposed in Section IV. A more detailed survey on adaptive streaming solutions can be found in [27].

III. System Model

This section introduces the model for the media content and client operations used in the rest of this work. Moreover, the notation is summarized in Table I.

A. Media model

Let us assume that a video sequence of duration D seconds is stored on a server organized in the form of $T = \lceil D/V \rceil$ segments, each of constant playback duration V. Each segment is encoded at N quality representations at increasing *target bitrate* $r \in \{r_1, \dots, r_N\}$. For a given quality representation $n \in \{1, \dots, N\}$, the *actual size* of the t-th segment ($t \in \{1, \dots, T\}$) – denoted $S_{t,n}$ and measured in bits – is a function of the segment content. In the following, we will assume that the server is connected to the client across a channel of rate C_t and thus the t-th segment is downloaded across the channel in $\frac{S_{t,n}}{C_t}$ seconds.

B. Client model

The client issues a request to the server, for the t-th segment and then waits for that segment to be fully downloaded before requesting the $(t+1)$-th segment. We refer to the, typically variable, interval between two consecutive requests as a *decision epoch*. Since the content is downloaded in T segments, the total number of decision epochs is T and referred to as the *horizon*. At the beginning of the t-th epoch, the client selects the quality representation $x_t \in \mathcal{X} = \{1, \dots N\}$ for segment t, corresponding to the bitrate indication $r_{x_t} \in \{r_1, \dots, r_N\}$ of the deployed rate adaptation algorithm.

TABLE I: Notations

Notation	Definition	Units
D	Video content total duration	seconds
V	Segment duration	seconds
T	Streaming horizon	segments
N	Quality representations	scalar
x_t	Selected quality representation for segment t	scalar
r_{x_t}	Bitrate corresponding to quality x_t	kbps
$S_{t,n}$	File size of segment t in n-th quality	kbits
ω_t	Decision distribution	probability vector
ω^*	Benchmark distribution	probability vector
Q	Virtual queue	scalar
V_L	Cautiousness parameter	scalar
α	Step-size	scalar
β	Target switching rate	switches per epoch
γ	Switch counter	scalar
C_t	Channel rate at epoch t	kbps
B_t	Buffer level at epoch t	seconds
B_{max}	Maximum buffer level	seconds
Δ_t	Buffer delay	seconds

Let B_t represent the buffer level at the beginning of the t-th epoch, measured in seconds of buffered video at the client. The downloaded segments are stored in a buffer whose size may not exceed an upper bound B_{max}, that exists normally due to memory constraints of the mobile device. Upon completely downloading the t-th segment, B_t increases by V seconds.

However, due to the concurrent playback of the buffered segments, B_t will also decrease by the amount of time required to download the t-th segment, which is equal to $\frac{S_{t,x_t}}{C_t}$ seconds (as long as $B_t > 0$). So, the buffer level evolves between two consecutive epochs according to:

$$B_{t+1} = \left[B_t - \frac{S_{t,x_t}}{C_t}\right]^+ + V - \Delta_t, \tag{1}$$

where $[x]^+ \triangleq \max(0, x)$. A delay $\Delta_t = \left[B_t - \frac{S_{t,x_t}}{C_t} + V - B_{max}\right]^+$ is introduced to account for the upper bound B_{max} of the buffer size. In other words, if $B_t - \frac{S_{t,x_t}}{C_t} + V < B_{max}$, the $(t+1)$-th segment is requested immediately and $\Delta_t = 0$. Otherwise, the request for the $(t+1)$-th segment is delayed by Δ_t seconds, to allow the buffer to drop to B_{max}. This delay protects against *buffer overflow* incidents, which occur when the buffer surpasses B_{max} and creates the characteristic bursty traffic of HAS [28]. A *buffer underflow* occurs when the instantaneous buffer level drops below zero, causing a *stall* in the video playback, an event that significantly degrades the QoE [11].

In the next section we provide a framework which allows us to design a learning algorithm, that provably optimizes video quality subject to keeping the buffer asymptotically away from the two limits.

IV. Adaptive streaming problem formulation

This section provides an algorithmic solution based on the theory of OCO [7]. In order to cast the video streaming optimization problem as an OCO with budget constraints problem, we first propose a relaxation on the finite buffer queue and then we modify the formulation to convexify the decision space. In the following, we present our online-learning algorithm *Learn2Adapt* (*L2A*), based on gradient descent and we provide theoretical guarantees for its performance.

A. OCO formulation

We formulate the rate adaptation problem as a *constrained OCO* problem, where the goal is to minimize the cumulative losses $\sum_{t=1}^{T} f_t(x_t)$ (referring to the average bitrate of the downloaded segments) while keeping the cumulative constraint functions $\sum_{t=1}^{T} g_t^i(x_t),\ \forall i = 1, 2$, negative (referring to buffer underflow and overflow); see also the relevant literature [29], [30]. In the OCO framework, functions $f_t, g_t^i\ \forall i = 1, 2$ are chosen by an *adversary* and are unknown at decision time. We will relate these functions to the random evolution of the channel rate C_t, which in nature is not adversarial. Nevertheless, the adversarial setting is more general and includes any – potentially time-varying – distribution of C_t, which in turn bestows on our algorithm superior *robustness*. Next, we explain how these functions are used in our system.

Recall the set of quality representations $\mathcal{X}$, and let $x_t \in \mathcal{X}$ be the decision for the quality representation of the segment to be downloaded in epoch t. Consider the following functions:

$$\tilde{f}_t(x_t) \triangleq -r_{x_t} \tag{2}$$

$$\tilde{g}_t^1(x_t) \triangleq \frac{S_{t,x_t}}{C_t} - V, \tag{3}$$

$$\tilde{g}_t^2(x_t) \triangleq V - \frac{S_{t,x_t}}{C_t} - \frac{B_{max}}{T}, \tag{4}$$

where (2) captures the utility (higher bitrate yields smaller losses). (3)-(4) express the buffer displacement, which will be used to model the buffer underflow and overflow constraints, respectively. A high quality representation x_t combined with a low channel rate C_t will prolong download time $\frac{S_{t,x_t}}{C_t}$, which will result in high buffer consumption. Since C_t is unknown at decision time of x_t, it is impossible to know the values of $\tilde{g}_t^i(x_t),\ \forall i = 1, 2$. Our approach therefore, is to learn the best x_t based on our estimation of $\tilde{g}_t^i(x_t),\ \forall i = 1, 2$.

To cast the above problem as OCO with budget constraints, we propose the following steps:

- First, we provide a relaxation to the hard constraints of the buffer model.
- Second, we convexify the decision set by randomization. We associate a probability to each decision, and we learn the optimal probability distribution for deciding the quality representation to download at each epoch.

B. Buffer constraints

Here we explain how we use the cumulative constraint functions $\sum_{t=1}^{T} \tilde{g}_t^i(x_t),\ \forall i = 1, 2$ to model buffer underflow and overflow, respectively. The buffer evolves according to (1) and ensuring $0 \leq B_t \leq B_{max},\ \forall t$, involves in principle a very complicated control problem, which in the presence of unknown adversarial C_t is exacerbated.

To avoid computationally heavy approaches and to arrive at a simple (yet robust) solution, we thus seek an alternative approach. In that direction, we treat the buffer as an infinite queue, with the simpler (compared to (1)) update rule: $B_{t+1} = B_t + V - S_{t,x_t}/C_t$, where now no additional delay Δ_t is ever imposed on the system. By this, we allow instantaneous violation of the budget, but we utilize a penalty which aims to maintain the buffer on the $[0, B_{max}]$ range on average. In particular, using (3)-(4), we capture in $\tilde{g}_t^i(x_t),\ \forall i = 1, 2$ the instantaneous buffer displacement on both directions (measured in seconds) and by requiring the cumulative constraint $\sum_{t=1}^{T} \tilde{g}_t^i(x_t) \leq 0,\ \forall i = 1, 2$, we ensure that on average B_t remains in the non-negative regime below B_{max}. A benefit is that these constraints are in the realm of OCO theory, and therefore allow us to design a simple learning algorithm that provably satisfies them. Overall, our approach here is to apply a loosely coupled control to the buffer constraints, by tolerating instantaneous violations and ensuring that in the long-term only a few are experienced.

C. Convexification

To obtain a convex decision set, we use a convexification method based on randomization of the decision process [31]. Consider the probability simplex:

$$\Omega = \{\boldsymbol{\omega} \in \mathbb{R}^N : \boldsymbol{\omega} \geq 0 \wedge \|\boldsymbol{\omega}\|_1 = 1\},$$

where $\omega_n = \mathbb{P}(x = n)$ denotes the probability that we decide $x = n \in \{1, \ldots, N\}$ and Ω is a convex set. Thus, instead of learning directly the decision x_t, we learn the optimal probability $\boldsymbol{\omega}_t = (\omega_{t,n})_{n=1,\ldots,N}$ of picking x_t from the integer set $\mathcal{X}$. Given a decision $\boldsymbol{\omega}_t$, the actual quality representation will be chosen according to the expectation of the corresponding utility, i.e. $x_t \in \arg\min_{x \in \mathcal{X}} |r_x - \sum_{n=1}^{N} \omega_{t,n} r_n|$. The functions of interest become now random processes and we must appropriately modify them by taking expectations with respect only to $\boldsymbol{\omega}_t$ and not to the randomness of C_t:

$$f_t(\boldsymbol{\omega}_t) \triangleq -\mathbb{E}[r_{x_t}] = -\sum_{n=1}^{N} \omega_{t,n} r_n \tag{5}$$

$$g_t^1(\boldsymbol{\omega}_t) \triangleq \mathbb{E}\left[\frac{S_{t,x_t}}{C_t} - V\right] = \frac{\sum_{n=1}^{N} \omega_{t,n} S_{t,n}}{C_t} - V \tag{6}$$

$$g_t^2(\boldsymbol{\omega}_t) \triangleq \mathbb{E}\left[V - \frac{S_{t,x_t}}{C_t} - \frac{B_{max}}{T}\right] = V - \frac{\sum_{n=1}^{N} \omega_{t,n} S_{t,n}}{C_t} - \frac{B_{max}}{T}. \tag{7}$$

Given the loss function and constraints above, we formulate the constrained OCO problem, that we solve in Section V:

$$\min_{\boldsymbol{\omega} \in \Omega} \sum_{t=1}^{T} f_t(\boldsymbol{\omega}) \quad \text{s.t.} \quad \sum_{t=1}^{T} g_t^i(\boldsymbol{\omega}) \leq 0 \quad \forall i = 1, 2.$$

The following are true for functions (5)-(7) and our surrogate convex problem:

- The diameter of Ω, defined as the largest Euclidean distance between any two vectors, is $\sqrt{N}$.
- Functions f_t and $g_t^i,\ \forall i = 1, 2$ are smooth, bounded and have bounded gradients. Specifically, $\forall t, \boldsymbol{\omega}, i = 1, 2$:

$$|f_t(\boldsymbol{\omega})| \leq r_N,$$

$$|g_t^1(\boldsymbol{\omega})| \leq \max\left\{\left|\frac{S_{\min}}{C_{\max}} - V\right|, \left|\frac{S_{\max}}{C_{\min}} - V\right|\right\},$$

$$|g_t^2(\boldsymbol{\omega})| \leq \max\left\{\left|V - \frac{S_{\min}}{C_{\max}} - \frac{B_{max}}{T}\right|, \left|V - \frac{S_{\max}}{C_{\min}} - \frac{B_{max}}{T}\right|\right\},$$

$$\|\nabla f_t(\boldsymbol{\omega})\| \leq \sqrt{\sum_{n=1}^{N} r_n^2}, \quad \|\nabla g_t^i(\boldsymbol{\omega})\| \leq \sqrt{\sum_{n=1}^{N} \left(\frac{S_{t,n}}{C_{\min}}\right)^2},$$

where $C_t \in [C_{\min}, C_{\max}]$, $S_{t,j} \in [S_{\min}, S_{\max}]$ and $\nabla f_t(\boldsymbol{\omega}), \nabla g_t^i(\boldsymbol{\omega}),\ \forall i = 1, 2$ denote the gradients.

D. Regret metric

At every decision epoch $t = 1, 2, \dots T$ the following events occur in succession:

(a) the agent computes $\boldsymbol{\omega}_t \in \Omega$ according to an algorithm,
(b) the agent chooses $x_t \in \arg\min_{x\in\mathcal{X}} |r_x - \sum_{n=1}^{N} \omega_{t,n} r_n|$,
(c) an adversary decides C_t, and the loss function $\tilde{f}_t(\boldsymbol{\omega}_t)$ and the constraint functions $\tilde{g}_t^i(\boldsymbol{\omega}_t),\ \forall i = 1, 2$ are determined using (2)-(4), and then used to measure the actual loss and buffer displacement,
(d) the following forms of feedback are provided to the agent: (i) the value of C_t, (ii) the functions $f_t, g_t^i,\ \forall i = 1, 2$, (iii) the gradients $\nabla f_t(\boldsymbol{\omega}_t), \nabla g_t^i(\boldsymbol{\omega}_t),\ \forall i = 1, 2$.

The feedback above is used by the agent to eventually determine the gradient vectors $\nabla f_{t+1}(\boldsymbol{\omega}_{t+1}), \nabla g_{t+1}^i(\boldsymbol{\omega}_{t+1}),\ \forall i = 1, 2$. We now define the performance metric in our problem which consists of two parts: the *regret* of an algorithm and the i-th *constraint residual*, defined as:

$$R_T = \sum_{t=1}^{T} f_t(\boldsymbol{\omega}_t) - \sum_{t=1}^{T} f_t(\boldsymbol{\omega}^*) \quad \text{and} \quad V_T^i = \sum_{t=1}^{T} g_t^i(\boldsymbol{\omega}_t),$$

respectively. Here $\boldsymbol{\omega}^* \in \Omega$ is a benchmark distribution, that minimizes the losses in hindsight, with knowledge of the functions $f_t, g_t^i,\ \forall i = 1, 2$. This benchmark satisfies the cumulative constraints every K:

$$\begin{aligned} \boldsymbol{\omega}^* \in \arg\min_{\boldsymbol{\omega}\in\Omega} & \sum_{t=1}^{T} f_t(\boldsymbol{\omega}) \\ \text{s.t.} & \sum_{t=k}^{K+k-1} g_t^i(\boldsymbol{\omega}) \le 0, \\ & \forall k = 1, \dots, T-K+1, \quad \text{and} \quad \forall i = 1, 2. \end{aligned}$$

This benchmark is first explained in [29], where the authors prove that for any $K = o(T)$, a smart agent can learn to have no regret, while satisfying the adversarial constraints. In our case, picking $K = T^{1-\epsilon}$, for small $\epsilon > 0$, gives the best approximation of our algorithms' performance, allowing maximum freedom to the competing benchmark. If an algorithm achieves both $o(T)$ regret and $o(T)$ constraint residual, then it follows that as $T \to \infty$ we have (i) $R_T/T \to 0$, hence our algorithm has the same losses with (or "learns") the benchmark action, and (ii) $V_T^i/T \to 0,\ \forall i = 1, 2$, hence our algorithm ensures the average constraint. Since the benchmark action is the best *a posteriori* action, taken with knowledge of all the revealed values of C_t, learning it is both remarkable and very useful.

V. OCO SOLUTION

In this section, we propose a "no regret" algorithm to solve the constrained OCO problem defined in the previous section. We first provide the intuition behind the algorithm design and the introduction of a switching budget, that allows the control of the switching frequency for our algorithm. We then detail the proposed algorithm, and finally provide some performance bounds.

A. Learn to Adapt (L2A) algorithm

As a general note, a main challenge in such problems is that the constraints $g_t^i(\boldsymbol{\omega}_t),\ \forall i = 1, 2$ are not known when the decision of $\boldsymbol{\omega}_t$ is taken. The OCO approach to this issue is to predict such functions using a first order Taylor expansion of $g_t^i(\boldsymbol{\omega}_t),\ \forall i = 1, 2$ around $\boldsymbol{\omega}_{t-1}$ evaluated at $\boldsymbol{\omega}_t$ [7]:

$$\hat{g}_t^i(\boldsymbol{\omega}_t) \triangleq g_{t-1}^i(\boldsymbol{\omega}_{t-1}) + \langle \nabla g_{t-1}^i(\boldsymbol{\omega}_{t-1}), \boldsymbol{\omega}_t - \boldsymbol{\omega}_{t-1} \rangle,\ \forall i = 1, 2. \tag{8}$$

We recall that in (8), only $\boldsymbol{\omega}_t$ is unknown at t, whereas $\boldsymbol{\omega}_{t-1}$, $\nabla g_{t-1}^i(\boldsymbol{\omega}_{t-1})$ and $g_{t-1}^i(\boldsymbol{\omega}_{t-1}),\ \forall i = 1, 2$ are known via the obtained feedback.

Contrary to the standard (unconstrained) online gradient [7], our algorithm must combine the objective and the constraint functions. To this end, consider the regularized Lagrangian:

$$L_t(\boldsymbol{\omega}, \boldsymbol{Q}(t)) = \sum_{i=1}^{2} Q_i(t) \hat{g}_t^i(\boldsymbol{\omega}) + V_L \hat{f}_t(\boldsymbol{\omega}) + \alpha ||\boldsymbol{\omega}_t - \boldsymbol{\omega}_{t-1}||^2,$$

where $Q_i(t)$ is the Lagrange multiplier, $\hat{g}_t^i(\boldsymbol{\omega})$ is the prediction of the constraint function $g_t^i(\boldsymbol{\omega})$ from (8), V_L is a *cautiousness* parameter that controls the trade-off between regret and constraint residual, $\hat{f}_t(\boldsymbol{\omega})$ applies (8) to f_t, α is the step-size and $||\boldsymbol{\omega}_t - \boldsymbol{\omega}_{t-1}||^2$ is a regularization term that smooths the decisions. Parameters V_L and α are tuned for convergence and their choices are given below. We mention here, that the Lagrange multiplier $Q_i(t),\ \forall i = 1, 2$ is updated in a *dual ascent* approach, by accumulating the constraint deviations:

$$Q_i(t+1) = [Q_i(t) + \hat{g}_t^i(\boldsymbol{\omega}_t)]^+,\ \forall i = 1, 2.$$

We further compound the online optimization problem by introducing a switching budget. Let $\beta \in (0, 1]$ be the maximum allowed reconfiguration frequency measured in quality switches per epoch. The goal is to limit the number of changes within the horizon to at most βT[1]. This is a valuable property that allows stability control for the following algorithm (Algorithm 1), that takes a step in the direction of the sub-gradient of the regularized Lagrangian.

Algorithm 1 Learn2Adapt (L2A)

Initialize: $\boldsymbol{Q}(1) = 0$, $\boldsymbol{\omega}_0 \in S$, $t' = 1$
Parameters: cautiousness parameter V_L, step size α, maximum allowed switch rate β, switch counter $\gamma = 0$

```
for all t ∈ {1, 2, ..., T} do
    if γ/t ≤ β then
        ω_t = proj_Ω [ ω_{t-1} − {V_L ∇f_{j-1}(ω_{j-1}) + Σ_{i=1}^{2} Q_i(j) ∇g_{j-1}^i(ω_{j-1})} / (2α) ]
        t' = t + 1
        γ++
    else
        ω_t = ω_{t-1}
    end if
    Q_i(t+1) = [Q_i(t) + ĝ_t^i(ω_t)]^+, ∀i = 1, 2
end for
```

Here $\mathrm{proj}_{\Omega}[\cdot]$ denotes the Euclidean projection on set Ω.

[1]While β may allow a switch at a given epoch t, $\boldsymbol{\omega}_t = \boldsymbol{\omega}_{t-1}$ is still a valid decision.

B. Performance guarantees

The main contribution of this work is the formulation the rate adaptation problem in the constrained OCO framework and the proposal of *Learn2Adapt* (*L2A*). In the following we invoke the theorem from [29], to provide theoretical performance guarantees for the *Learn2Adapt* algorithm. We note here that although the following theoretical guarantees are derived for $\beta = 1$, in the numerical evaluation of Section VI we provide evidence that *L2A* performs well even for $\beta < 1$.

Theorem 1 (From [29])**.** *For $\beta = 1$, choose small $\epsilon > 0$, fix $K = o(T^{1-\epsilon})$, $V_L = T^{1-\epsilon/2}$, and $\alpha = V_L\sqrt{T}$. Then, the* Learn2Adapt (L2A) *algorithm guarantees:*

$$R_T = O(T^{1-\epsilon/2}), \qquad V_T^i = O(T^{1-\epsilon/4}),\ \forall i = 1, 2.$$

Effectively, this means that over time our algorithm learns the best a-posteriori distribution $\boldsymbol{\omega}^*$, which neatly satisfies the average constraints and minimizes the cumulative quality losses. We experimentally verify below that the corresponding choices x_t made by sampling this distribution have extremely well performing properties for video streaming adaptation.

VI. Experimental Evaluation

In this section we evaluate the performance of our proposed rate adaptation algorithm against two reference rate adaptation schemes, by experimenting with real mobile network traces and video sequences, for two separate streaming applications (Video on Demand (VoD) and live streaming) and under five video streaming performance metrics.

A. Experimental setup

a) Network scenarios: In our evaluation, we use *real* cellular network traces and in particular a data-set that includes 4G channel measurements for various mobility scenarios [32]. For our experiments, we have selected the *static*, *pedestrian* and *car* scenarios (operator A therein), as realistic cases for no, low and high mobility, respectively. The {*static*, *pedestrian*, *car*} scenario consist of $\{12, 26, 41\}$ traces with an average measurement duration of $\{17, 18, 23\}$ min, respectively. Cellular networks present significant challenges to the rate adaptation process, as they are typically characterized by rapid throughput fluctuation and short service outages; that may be caused by radio propagation effects, low-coverage areas or handover events. While these characteristics are realistically depicted in the selected traces of [32], taking a step further in our evaluation, we have designed a *synthetic* scenario consisting of 20 traces, that is characterized by abrupt and steep channel rate transitions. This so-called *markovian* scenario emulates two channel levels (states) $\{0.75, 23.0\}$ Mbps with a 0.05 state transition probability and is complementary to the real traces; to present the rate adaptation algorithms with an additional, even more demanding network scenario.

b) Video parameters: In [33] video sequences are encoded at multiple bitrates in conditions typical of Over-The-Top (OTT) video delivery. We used 3 sequences: *BBB*, *TOS* and *Sintel*, encoded in the H.264/AVC standard, at target bitrates $\{0.37, 0.75, 1.5, 3.0, 5.8, 12.0, 17.0, 20.0\}$ Mbps, corresponding to resolutions in $\{384 \times 216, 640 \times 360, 1024 \times 576, 1280 \times 720, 1920 \times 1080, 3840 \times 2160\}$, and organized in DASH segments with duration $V = 2$s.

c) Streaming scenarios: In our experiments, we consider a VoD streaming scenario and a live streaming scenario. For the VoD scenario, we considered a maximum buffer value of $B_{max} = 120$s (60 segments). For the live streaming scenario, we reduced the maximum buffer value to $B_{max} = 20$s (10 segments), according to the tighter latency requirements. All the figures below concern the case of VoD, while the results for the live streaming scenario are presented in Table III.

d) Algorithms: We compare our method *L2A*, for $\beta = 0.3$ and $\beta = 1$, against *RB*, a throughput-based method and *BB*, a buffer-based method, following the design principles and parameters selection found in [15] and [21], respectively.

RB [15, Section VI] is a throughput-based rate adaptation scheme based on a four-step adaptation model, where initially the available network bandwidth is estimated using a proactive probing mechanism, that is designed to minimize bitrate oscillations. Then, the throughput estimates are smoothed using noise-filters to avoid errors due to throughput variation and each segment download is scheduled according to inter-request times, that would drive the buffer to the maximum level.

BB BOLA is a buffer-based rate adaptation algorithm that uses Lyapunov optimization in order to indicate the bitrate of each segment. Practically, the algorithm is designed to maximize a joint utility function that rewards increases in the average bitrate and penalizes stalls. The implemented variant, called *BOLA-O*, mitigates bitrate oscillations by a form of bitrate capping when switching to higher bitrates.

These rate adaptation methods are widely used in research, each amongst the best performing methods of their class [4]. Regarding our method *L2A*, the presented results consider a cautiousness parameter of $V_L = T^{0.9}$ and step size of $\alpha = V_L\sqrt{T}$. We note here that according to DASH, in case of a stall, τ segments must be downloaded in order for the play-out to resume. For all algorithms we considered $\tau = 2$.

e) Video streaming performance metrics: We evaluate the performance of our proposed method based on the video streaming performance metrics presented in Table II. *Average bitrate* models the average bitrate $\bar{r} = \frac{\sum_{t=1}^{T} r_{x_t}}{T}$ of the received video segments in a session, normalized over the maximum average bitrate $\max_{m \in \mathcal{M}} \bar{r}^m$ obtained for that session by any adaptation method $m \in \mathcal{M}$, where $\mathcal{M}$ is the set of all evaluated methods. Streaming *stability* models the frequency of bitrate switching, while streaming *smoothness* is associated with the amplitude of the bitrate switches, i.e the absolute bitrate difference between sequential segments. Both *stability* and *smoothness* are normalized over the maximum attainable value for each respective metric, while $\mathbb{1}_{\{\mathcal{Y}\}}$ is an indicator vector; with ones at the positions that condition $\mathcal{Y}$ is true, and zeros otherwise. Additionally, we propose two metrics associated with a) the frequency of stalls and b) their severity (duration). With streaming *consistency* we measure the percentage of the user's allocated time-budget (typically equal to the video length D) that was spent actually consuming video content (as opposed to stalling), while streaming *continuity*

TABLE II: Video streaming performance metrics

Metric name	Element evaluated	Metric
Average bitrate	Average video bitrate	$\frac{\bar{r}}{\max_{m\in\mathcal{M}}\bar{r}^m}$
Stability	Bitrate switching frequency	$1-\frac{\sum_{t=2}^{T}\left(\mathbb{1}_{\{r_{x_t}\neq r_{x_{t-1}}\}}\right)}{T-1}$
Smoothness	Adaptation amplitude	$1-\frac{\sum_{t=2}^{T}\lvert r_{x_t}-r_{x_{t-1}}\rvert}{(r_N-r_1)(T-1)}$
Consistency	Stall duration	$1-\frac{\sum_{t=1}^{T}\mathbb{1}_{\left\{B_{t-1}<\frac{S_{t,x_t}}{C_t}\right\}}\left(-B_{t-1}+\sum_{k=0}^{\tau-1}\frac{S_{t+k,x_{t+k}}}{C_{t+k}}\right)}{D}$
Continuity	Frequency of stalls	$1-\frac{\sum_{t=1}^{T}\mathbb{1}_{\left\{B_{t-1}<\frac{S_{t,x_t}}{C_t}\right\}}}{\lceil\frac{T}{\tau}\rceil}$

expresses the percentage of segments that were downloaded while play-out remained uninterrupted, assuming $B_0 = 0$.

B. Results

In regard to the evaluation results, each point on Figure 1 corresponds to a score for each of the five performance metrics and is the average result over all traces for the considered network scenario. In particular, Figure 1(a) shows the performance of a static user (no mobility), Figure 1(b) shows the performance of a pedestrian user (low mobility), Figure 1(c) shows a user while being mobile in a car (high mobility) and Figure 1(d) corresponds to the artificial *markovian* scenario.

In Figure 1 *L2A*, our proposed method, registers significant improvement in average bitrate, almost up to 45% against *RB* and up to 20% against *BB*, for all studied real network scenarios and for both the cases of restricted ($\beta = 0.3$) and unrestricted ($\beta = 1$) switching. At the same time *L2A* offers consistent (i.e without interruptions) streaming with equivalent continuity to all the other methods, i.e. all methods experience a few brief stalls during periods of very poor channel quality. In regard to smoothness, all methods obtain equivalent scores. Nonetheless, in regard to the stability metric, *L2A*'s restricted switching variant ($\beta = 0.3$) achieves about 15% improvement in stability when compared to the case of unrestricted ($\beta = 1$) switching; a result that is anticipated from our algorithmic design. Additionally, *L2A*'s restricted switching variant ($\beta = 0.3$) improves on stability by 25% against *BB*. Comparing *L2A* and *RB* in stability, we observe equivalent performance, yet *RB* is overall more conservative, given the low average bitrate it obtained in all scenarios.

In the markovian network scenario depicted in Figure 1(d), *L2A* performs 50% better against *RB* and 25% better against *BB* in terms of average bitrate, while performing equally well, or even better (i.e. against *BB* in stability), in all other metrics. Thus *L2A* is *robust* against the channel fluctuations and doesn't require any assumption on the channel rate distribution.

In Figure 2(a), a sample path for the channel rate and the bitrate selection for each method is presented for a randomly-selected pedestrian-mobility trace, while, Figure 2(c) depicts the evolution of the buffer for the same trace[2]. From these plots, we can argue that *L2A* learns the volatile channel distribution, in order to re-actively provide the highest bitrate (which is the optimization objective) and to pro-actively protect the buffer from under-flowing (which is one of the optimization constraints). While this 'adaptive behavior' of *L2A* may come only at a marginal cost in smoothness (i.e. bitrate distance between consecutive decisions), Figure 1(b) shows that L2A achieves on average only 3% less smoothness than the other methods; a trade-off that is aligned with common HAS optimization principles.

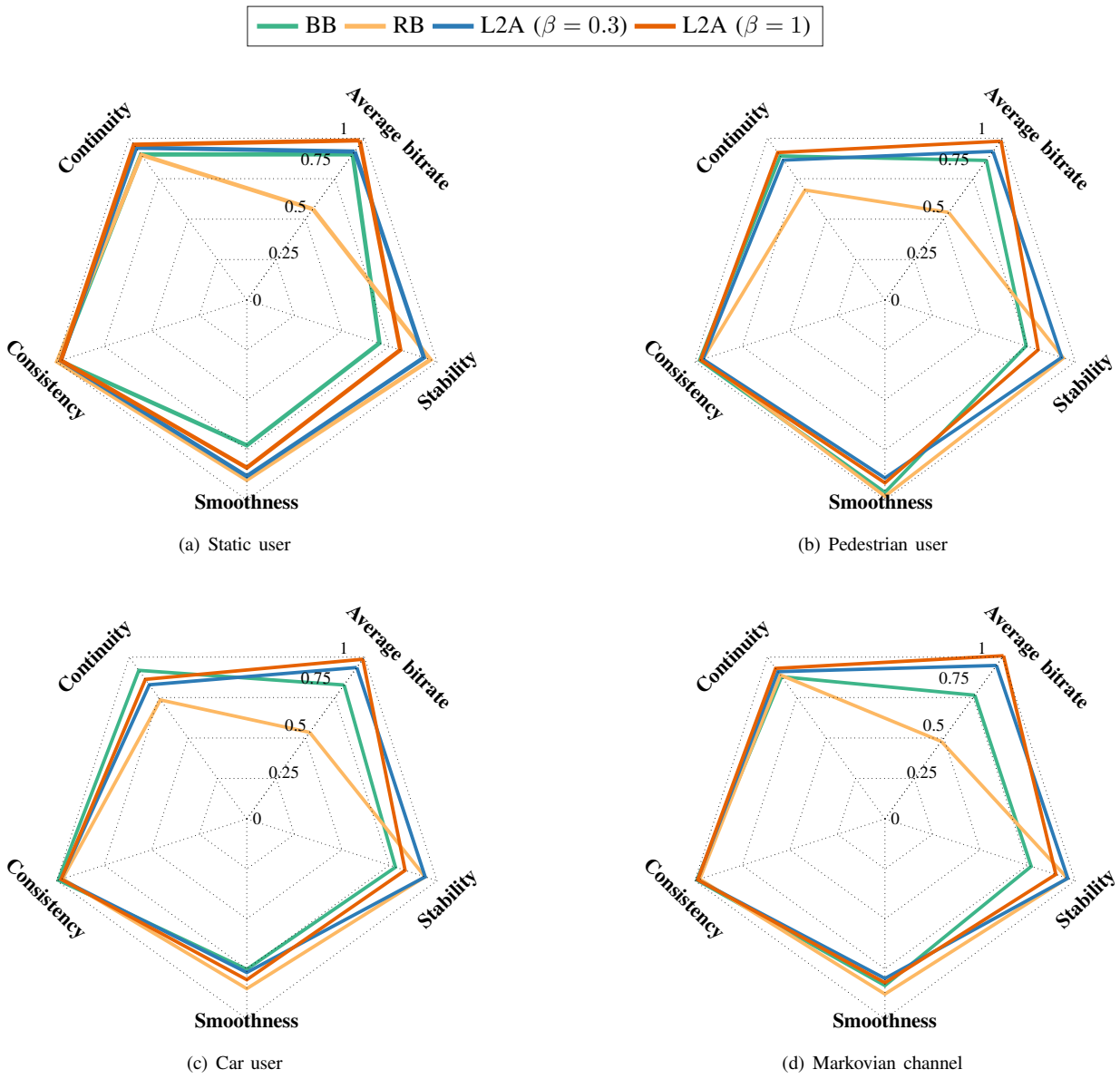
Similarly, we provide a bitrate selection sample path in Figure 2(b), for a randomly-selected markovian trace and its corresponding buffer evolution in Figure 2(d). Here, we observe: (i) that in terms of matching the bitrate to the channel rate at each decision epoch, *L2A* presents a more efficient channel utilization, (ii) a slightly unstable behavior for *BB*, especially at the beginning of the session and (iii) some stall events occurring for *RB*. *L2A* consistently manages to offer high bitrate, stable and uninterrupted streaming; even in the most demanding network scenarios.

To further investigate the *robustness* property of our method, we have synthesized an additional network profile, where we have concatenated *car* traces to extend the streaming session duration (horizon) in order to simulate longer, yet realistic scenarios. For these concatenated *car* traces, Figure 3(a) presents the regret rate R_T/T against the K-Slot benchmark of Section IV, for $K = T^{0.9}$. Here *L2A*, for both studied values of β $(0.3, 1)$, achieves better regret than any other method, significantly improving on the K-Benchmark in any streaming horizon, a result that is anticipated from Theorem 1.

Regarding the constraint residual V_T^i $\forall i = 1, 2$, we examine in particular the case of underflow ($i = 1$), as stalls are the most significant factors that can affect the streaming experience. Potential buffer overflows ($i = 2$) can be easily tackled by simply inducing a short delay before requests, according to (1). In Figure 3(b) we present the constraint residual rate for the concatenated *car* traces. *L2A* manages to respect the underflow constraint on average, given that the constraint residual rate V_T^1/T converges to 0.

In order to investigate the merits of *L2A* beyond VoD, we repeated the same cycle of experiments for the case of *live streaming*, where now $B_{max} = 20$s. In industrial live streaming applications, such small buffer values are commonly used, given the strict delay requirements. We present our results in Table III, which depicts that although *RB* achieves higher values in stability, it is not able to compete with the other methods in terms of average bitrate. On the contrary, *L2A*

[2]We note here that in Figure 2, while some markers have been omitted for clarity, the lines remain an accurate representation of the results.



(a) Static user

(b) Pedestrian user

(c) Car user

(d) Markovian channel

Fig. 1: Performance evaluation results - *L2A* improves average bitrate

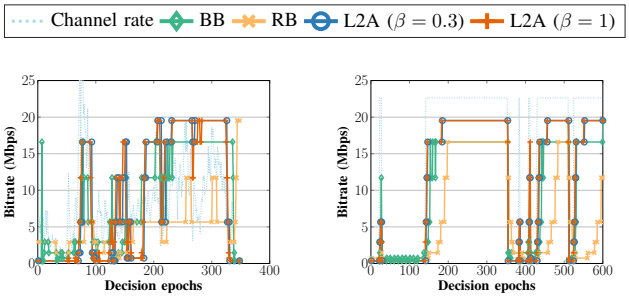
TABLE III: Live streaming ($B_{max} = 20s$) results (BB / RB / L2A ($\beta = 0.3$) / L2A ($\beta = 1$))

	Static	Pedestrian	Car	Markovian
Average bitrate	0.93 / 0.59 / 0.88 / **0.98**	0.94 / 0.58 / 0.93 / **0.96**	0.93 / 0.59 / 0.96 / **0.98**	0.91 / 0.69 / 0.97 / **1.00**
Stability	0.42 / **0.95** / 0.82 / 0.72	0.50 / **0.92** / 0.86 / 0.75	0.56 / **0.91** / 0.86 / 0.78	0.86 / **0.93** / 0.87 / 0.82
Smoothness	0.86 / 0.92 / 0.94 / **0.94**	0.86 / 0.92 / 0.94 / **0.95**	0.87 / 0.95 / 0.94 / **0.97**	0.96 / 0.95 / 0.92 / **0.98**
Consistency	0.87 / 0.81 / 0.92 / **0.92**	0.85 / 0.62 / 0.83 / **0.85**	0.88 / 0.62 / 0.80 / **0.85**	0.78 / 0.48 / 0.83 / **0.84**
Continuity	0.93 / 0.95 / 0.97 / **0.97**	0.93 / 0.93 / 0.97 / **0.97**	0.93 / 0.94 / 0.95 / **0.95**	0.92 / 0.88 / 0.94 / **0.94**

manages to provide up to 30% higher live streaming bitrate when compared to *RB* and equivalent bitrate to *BB*, while its switch-restricted instance shows up to 40% improvement in stability when compared to *BB*. Online learning methods have – by design – less dependency on the instantaneous buffer length and are also more reactive to throughput fluctuations, unlike throughput-based methods that are, normally, as efficient as their throughput estimation module.

VII. Conclusions

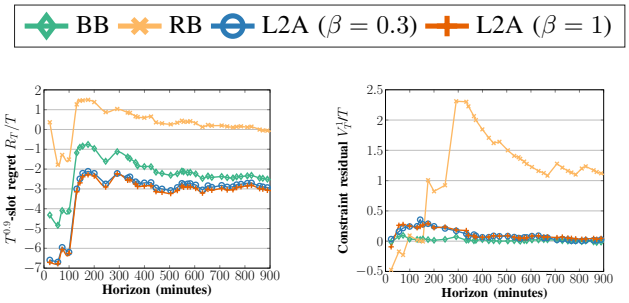
In this work we present *Learn2Adapt* (*L2A*), a novel rate adaptation algorithm for HAS, based on online learning. Overall, our proposed method performs well over a wide spectrum of streaming scenarios, due to its design principle; its ability to learn. It does so without requiring any parameter tuning, modifications according to application type or statistical assumptions for the channel. The *robustness* property of *L2A* allows it to be classified in the small set of rate adaptation algorithms for video streaming, that mitigate the main limitation of



(a) Bitrate selection pedestrian user (b) Bitrate selection markovian channel

(c) Buffer level pedestrian user (d) Buffer level markovian channel

Fig. 2: Sample paths for bitrate selection and buffer level



(a) $T^{0.9}$-slot regret R_T/T (b) Constraint residual V_T^1/T

Fig. 3: Convergence of regret and constraint residual

existing mobile HAS approaches; the dependence on statistical models for the unknowns. This is of significant relevance in the field of modern HAS, where OTT video service providers are continuously expanding their services to include more diverse user classes, network scenarios and streaming applications.

REFERENCES

[1] Cisco Visual Networking Index, "Forecast and Trends, 2017–2022," *White Paper*, Feb. 2019.

[2] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet," *IEEE MultiMedia*, vol. 18, April 2011.

[3] Cisco Visual Networking Index, "Global Mobile Data Traffic Forecast Update, 2017–2022," *White Paper*, Feb. 2019.

[4] T. Karagkioules, C. Concolato, D. Tsilimantos, and S. Valentin, "A Comparative Case Study of HTTP Adaptive Streaming Algorithms in Mobile Networks," in *Proc. of ACM Int. Conf. on NOSSDAV*, June 2017.

[5] M. Gadaleta, F. Chiariotti, M. Rossi, and A. Zanella, "D-DASH: A Deep Q-Learning Framework for DASH Video Streaming," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, Dec 2017.

[6] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP," in *Proc. ACM Int. Conf. on SIGCOMM*, Aug. 2015.

[7] M. Zinkevich, "Online Convex Programming and Generalized Infinitesimal Gradient Ascent," in *Proc. of ICML*, 2003.

[8] E. V. Belmega, P. Mertikopoulos, R. Negrel, and L. Sanguinetti, "Online convex optimization and no-regret learning: Algorithms, guarantees and applications," *arXiv e-prints*, 2018.

[9] N. Liakopoulos, G. Paschos, and T. Spyropoulos, "No Regret in Cloud Resources Reservation with Violation Guarantees," in *Proc. IEEE INFOCOM*, May 2019.

[10] T. Chen, Q. Ling, and G. B. Giannakis, "An Online Convex Optimization Approach to Proactive Network Resource Allocation," *IEEE Transactions on Signal Processing*, vol. 65, Dec 2017.

[11] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hofeld, and P. Tran-Gia, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys Tutorials*, vol. 17, Firstquarter 2015.

[12] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. C. Begen, "Server-based Traffic Shaping for Stabilizing Oscillating Adaptive Streaming Players," in *Proc. of ACM Int. Conf. on NOSSDAV*, June 2013.

[13] V. Krishnamoorthi, N. Carlsson, E. Halepovic, and E. Petajan, "BUFFEST: Predicting Buffer Conditions and Real-time Requirements of HTTP(S) Adaptive Streaming Clients," in *Proc. of ACM Int. Conf. on MMSys*, June 2017.

[14] N. Bouten, S. Latr, J. Famaey, W. Van Leekwijck, and F. De Turck, "In-Network Quality Optimization for Adaptive Video Streaming Services," *IEEE Transactions on Multimedia*, vol. 16, Dec 2014.

[15] 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 of Communication*, Apr. 2014.

[16] J. Jiang, V. Sekar, and H. Zhang, "Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Video Streaming With Festive," *IEEE/ACM Transactions on Networking*, vol. 22, Feb 2014.

[17] 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 *Proc. of ACM Conf. on SIGCOMM*, 2014.

[18] S. Kim and C. Kim, "XMAS: An Efficient Mobile Adaptive Streaming Scheme Based on Traffic Shaping," *IEEE Transactions on Multimedia*, vol. 21, Feb 2019.

[19] X. Xie, X. Zhang, S. Kumar, and L. E. Li, "piStream: Physical Layer Informed Adaptive Video Streaming over LTE," in *Proc. of ACM Int. Conf. on MobiCom*, 2015.

[20] S. Mekki, T. Karagkioules, and S. Valentin, "HTTP Adaptive Streaming with Indoors-Outdoors Detection in Mobile Networks," in *Proc. of IEEE INFOCOM Workshops*, May 2017.

[21] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in *IEEE INFOCOM*, April 2016.

[22] C. Zhou, C. Lin, and Z. Guo, "mDASH: A Markov Decision-Based Rate Adaptation Approach for Dynamic HTTP Streaming," *IEEE Transactions on Multimedia*, April 2016.

[23] A. Bokani, M. Hassan, S. Kanhere, and X. Zhu, "Optimizing HTTP-Based Adaptive Streaming in Vehicular Environment Using Markov Decision Process," *IEEE Transactions on Multimedia*, vol. 17, Dec 2015.

[24] M. Claeys, S. Latr, J. Famaey, T. Wu, W. Van Leekwijck, and F. De Turck, "Design of a Q-learning-based client quality selection algorithm for HTTP adaptive video streaming," in *In Proc. of Adaptive and Learning Agents Workshop, part of AAMAS*, May 2013.

[25] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with Pensieve," in *Proc. of ACM Int. Conf. on SIGCOM*, 2017.

[26] T.-Y. Huang, C. Ekanadham, A. J. Berglund, and Z. Li, "Hindsight: Evaluate video bitrate adaptation at scale," in *In Proc. of ACM MMSys*, Jun 2019.

[27] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP," *IEEE Communications Surveys Tutorials*, vol. 21, Firstquarter 2019.

[28] D. Tsilimantos, T. Karagkioules, and S. Valentin, "Classifying flows and buffer state for YouTube's HTTP adaptive streaming service in mobile networks," in *Proc. ACM Int. Conf. on MMSys*, June 2018.

[29] N. Liakopoulos, A. Destounis, G. Paschos, T. Spyropoulos, and P. Mertikopoulos, "Cautious Regret Minimization: Online Optimization with Long-Term Budget Constraints," in *Proc. of ICML*, June 2019.

[30] M. J. Neely and H. Yu, "Online Convex Optimization with Time-Varying Constraints," *arXiv e-prints*, Feb. 2017.

[31] S. Shalev-Shwartz, "Online learning and online convex optimization," *Foundations and Trends on Machine Learning*, Feb. 2012.

[32] D. Raca, J. J. Quinlan, A. H. Zahran, and C. J. Sreenan, "Beyond throughput: A 4G LTE dataset with channel and context metrics," in *Proc. of ACM Int. Conf. on MMSys*, June 2018.

[33] A. Zabrovskiy, C. Feldmann, and C. Timmerer, "Multi-codec DASH dataset," in *Proc. of ACM Int. Conf. on MMSys*, June 2018.