NEWCAST: Anticipating resource management and QoE provisioning for mobile video streaming

Imen Triki, Rachid El-Azouzi, Majed Haddad

NEWCAST: Anticipating Resource Management and QoE Provisioning for Mobile Video Streaming

Imen Triki, Rachid El-Azouzi and Majed Haddad
LIA/CERI, University of Avignon,
Agroparc, BP 1228, 84911, Avignon, France

***Abstract*—The knowledge of the future capacity variations in wireless networks using smartphones becomes more and more possible by exploiting the rich contextual information from smartphone sensors through mobile applications and services. It is entirely likely that such contextual information, which may include the traffic, mobility and radio conditions, could lead to a novel agile resource management not yet thought of. Inspired by the attractive features and potential advantages of this agile resource management, several approaches have been proposed during the last period. However, agile resource management also comes with its own challenges, and there are significant technical issues that still need to be addressed for successful rollout and operation of this technique. In this paper, we propose an approach (called NEWCAST) for anticipating throughput variation for mobile video streaming services. The solution of the optimization problem realizes a fundamental trade-off among critical metrics that impact the user's perceptual quality of the experience (QoE) and system utilization. Both simulated and real-world traces collected from [1] are carried out to evaluate the performance of NEWCAST. In particular, it is shown that NEWCAST provides the efficiency, computational complexity and robustness that the new 5G architectures require.**

I. Introduction

Since the smartphones introduction, mobile networks are witnessing an exponential traffic growth. The large penetration of devices such as smartphone with wireless connectivity provides new capabilities to support various applications and services. Due to their smaller form factor, these truly mobile devices allow the users to access the wireless networks while undergoing different types of mobility, posing new challenges to wireless protocols. This creates regimes where wireless networks are pushed to operate close to their performance limits. The evolution of multimedia services in the Internet and the increasing consumer demand for high definition (HD) contents have led operators and industry to rethink the way networks are dimensioned. Many studies in the literature have identified the critical metrics that impact the user's perceptual QoE [2]. Recent works developed approaches for understanding how some quality metrics influence user engagement [3], [4]. Authors in [3] quantified the user engagement and identified critical metrics that affect the QoE, e.g., buffering ratio, rate of buffering, start-up delay, rendering quality and average bit rate. [4] showed that stalls (or rebuffering) events have a significant impact on the QoE in the sense that the time spent on rebuffering during a video session can significantly reduce the user engagement. [5] provided guidance to operators for improving user engagement in real time using only network-side measurements. Another aspect is related to variations in the temporal quality of the video. Indeed, authors in [6] suggested that temporal variability in quality can be considered as worse as a constant quality with a lower average bitrate.

Recently, Dynamic Adaptive Streaming over HTTP (DASH) or equivalent (Apple HLS, Adobe HDS, etc.) have been proposed as an emerging standard on video delivery in order to minimize video interruption and rebuffering time. In DASH, each video file is divided into multiple small segments according to the size of GOPs (Group of Pictures) and each segment is encoded into multiple quality levels [7]. The segment size is usually between two and ten seconds of video. Various adaptation methods have been proposed to support adaptive HTTP streaming [8]. Their common goal is to make the client chose the most suitable level of the future segment in order to deliver the highest possible QoE. In the literature, adaptive algorithms have been classified in three main classes: buffer-based [9], throughput-based [10] and buffer–throughput-based algorithms [11]. While the first class makes the decision based on the playback buffer occupancy state, the second class makes use of the historical TCP throughput of last few fragments to estimate the current bandwidth and instantaneously adapt the segment quality. Several techniques have been used to estimate the resources from past observations [12]. In an environment with constant throughput, past observations are relevant to predict the future capacity. However, in a highly variable environment, this approach cannot provide accurate estimation of the future capacity. Unfortunately, current environments are characterised by their significant variability, essentially due to mobility and heterogeneity of wireless networks.

Recent studies open a promising possibility to accurately predict available resources over a medium horizon. For instance, context acquisition can target the monitoring of contextual situations as soon as they are created. The output can describe the contexts encountered as well as the likelihood of encountering similar contexts in the future [13]. This rises the opportunity to efficiently design the client-side bitrate adaptation by exploiting the knowledge of future capacity variations [14]. In this paper, we design a QoE-driven optimization scheme that realizes the trade-off between system utilization and content resolution under constraints on rebuffering events.

A. Related works

Many strategies have been proposed to adapt the video quality in order to reduce the interruption of playback buffer [9], [15]. Authors in [9] developed a rate adaptation method to enhance DASH performance over multiple content distribution servers. In [16], a method based on customers subjective perception was designed to analyze the impact of bitrate distribution on user's psychology and QoE. Further works in [17] investigated the impact of temporally changing quality by introducing new quality profiles such as quality hoping and switching rate. [11] proposed a predictive control algorithm that can optimally combine throughput and buffer occupancy information. [18] developed a suite of techniques that can guide the trade-offs between stability, fairness, and efficiency leading to a general framework for robust video adaptation. Empirical results in [19] showed that humans appear to be more forgiving on buffer stalls than they are on video quality variations. Long buffer freezing events are even not rated worse than short buffer freezing towards high video quality levels. In [20], authors address the resource management issue in DASH QoE provisioning, while considering user preferences on rebuffering and cost of video delivery.

Although there is a rich literature on methods for optimizing QoE parameters for video streaming services, anticipating (future) capacity variations have been presented in only few papers. The main idea of this paper is inspired from the paper [21] which designed a QoE-driven optimization problem that exploits the knowledge of the future capacity. The authors developed a cross-layer transport protocol that minimizes system utilization while avoiding rebuffering events. Their approach allows to reduce the system utilization, but the model only holds for classical video streaming as it ignores important visual quality metrics related to adaptive HTTP streaming such as video resolution and bitrate distribution. In this paper, we assume that the video player can adapt its video quality across segments. The choice of the video quality level can be based on several factors such as the system utilization, the risk of stalls and the switch in video quality.

B. Contributions and Organization

This study provides important insights in the design and optimization of adaptive video streaming by exploiting the knowledge of future capacity variations. The main objective is to design the client-side bitrate adaptation that performs under diverse opening regimes with high throughput variability. This framework allows to extend the model developed in [21] by balancing different objectives such as system utilization, video quality, rebuffering events, robustness, etc. In short, we can summarize the main contributions of this paper as follows

- We provide a general optimization framework for stored video delivery that accounts for heterogeneous client preferences, QoE models, capacity variations and video quality,
- Under the constraint of no rebuffering events, we formally obtained the optimal solution where the transmission schedule is of a threshold type and the coding strategy is of an ascending type,
- Under the hypothesis of tolerating one buffer stall during the streaming session, we evaluate the system performances by forcing stalls at different moments of the video for different network and user preferences,
- We propose an efficient mechanism, which we call NEWCAST, that performs close to the optimal solution, and we evaluate its performance and robustness using realistic traces.

The rest of the paper is organized as follows: In Section II, we introduce the system model and formulate the optimization problem. In Section III, we discuss the properties of the optimal solution. Then, in Section IV, we propose an algorithmic approach to implement optimal and sub-optimal solutions of our optimization problem. In Section V, simulation results are presented and discussed. Section VI concludes the paper.

II. Problem formulation

We consider a video streaming server where each video file is divided into N small segments according to the size of GOPs. Each segment is encoded with L different quality levels. Let l_j be the video quality level j, and b_j its associated bitrate level, where $b_i < b_j$ for $i < j$. For each segment, there is the same number of frames denoted by S. Assume that, while streaming a video, the client requests segments from the server such that only one quality level can be selected at time. Let $b(t)$ be the bitrate level used at time t. In order to characterize this bitrate level with respect to the best quality level, we define $\gamma(t)$ as $\gamma(t) = \frac{b_L}{b(t)}$. At the client side, we assume that the playback buffer is large enough to avoid buffer overflows and that the playback frame rate holds the same for all quality levels, denote it by λ (in fps). To avoid rebuffering events, a prefetching stage is introduced at the beginning of the streaming session; before starting the video, the media player stores frames until a certain number, called start-up threshold and denoted by Q_0, is reached.

In our problem modeling, we attempt to optimize the system cost and the video quality while taking into account user's perception metrics like rebuffering. This optimization problem will exploit the knowledge of future capacity variations over a finite window of the horizon (see Fig. 11). Let $c(t)$ and $r(t)$ be the average network capacity and the actual used bit rate at time t such that $0 \leq r(t) \leq c(t)$. Next, we define the cost function of our model, which is the sum of two terms: the average system utilization cost and the average quality of the video. As in [21], we define the network utilization cost as

$$\sigma = \frac{1}{T} \int_0^T \frac{r(t)}{c(t)} dt \tag{1}$$

where $\frac{r(t)}{c(t)}$ designs the proportion of resources used at time t, and T defines the video lecture length in (s).

During the session, the number of frames streamed with video quality level j is given by

$$\int_0^T \frac{\delta_{\{b(t)=b_j\}} r(t)\lambda}{b(t)} dt = \int_0^T \frac{\gamma_j(t) r(t)\lambda}{b_L} dt \tag{2}$$

where

$$\gamma_j(t) = \begin{cases} \gamma(t) & \text{if } b(t) = b_j \quad j \in [1 \dots L] \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Denote by w_j the weight associated to video quality level j, where $w_i < w_j$ for $i < j$. The value of w_j represents the user's perception on the video quality level j. Hence the average quality of the video is given by

$$\rho = \frac{\sum_{j=1}^{j=L} w_j \int_0^T \gamma_j(t) r(t)\lambda dt}{b_L \times (N \times S)} = \frac{\sum_{j=1}^{j=L} w_j \int_0^T \gamma_j(t) r(t) dt}{S_L} \tag{4}$$

where S_L is the video total size in bits when it is coded with the highest bitrate level b_L, i.e., $S_L = \frac{b_L \times N \times S}{\lambda}$.

To better explain the meaning of this term, consider a video file of 4 frames and 2 bitrate levels: $b_1 = 6000kbps$ and $b_2 = 3000kbps$. Let $\lambda = 30fps$ and $c(t) = 200kbps$ for 4 lapses of time. Obviously, when 4 frames are coded with b_1 we use all the available resources since b_1 gives $200kbpf$. However when the 2 first frames are coded with b_2 and the 2 other frames are coded with b_1, we use only the three first capacities, since b_2 gives $100kbpf$. In the first case, $\rho = w_1 \times (1 \times 200 + 1 \times 200 + 1 \times 200 + 1 \times 200)/800$ which gives $w_1 \times 1$, but in the second case $\rho = (w_2 \times (2 \times 200) + w_1 \times (1 \times 200 + 1 \times 200) + 0 \times 200)/800$ which gives $w_2 \times 0.5 + w_1 \times 0.5$; In both cases we fall in with the proportion of each bitrate level multiplied by its corresponding weight, which is clearly what we need to express through the QoE.

It goes without saying that higher QoE comes at higher system utilization cost. However, it may happen that a user wishes to reduce his utilization of the system in return of a lower video quality, or that a content provider wishes to reduce delivery cost at the expand of a minimal video quality that guarantees clients engagement. Another interesting example is when an operator prefers saving network resources for further usage. To cover such situations, we define a positive constant parameter π that allows to balance between system utilization cost (at the network side) and QoE (at the client side). Our optimization cost function can then be defined as

$$\mathcal{F} = \sigma - \pi \times \rho.$$

We start with the case where there are no rebuffering events during the streaming session. Thus, when minimizing our cost function $\mathcal{F}$, we are constrained to ensure that the playback buffer does not fall empty. Thereafter, at the end of Section V-B, we will study the case where we tolerate a rebuffering event during the streaming session.

Let $u(t)$ and $l(t)$ be the *cumulative* number of arrival frames and the *cumulative* number of frames watched by the user till time t under the no-rebuffering events assumption. Then, the buffer underflow constraint can be defined as $u(t) \geq l(t)\ \forall t \leq T$. Given the network bitrate $r(t)$ and the corresponding streamed bitrate level $b(t)$, we compute the network frame rate as $\frac{\lambda\ r(t)}{b(t)}$.

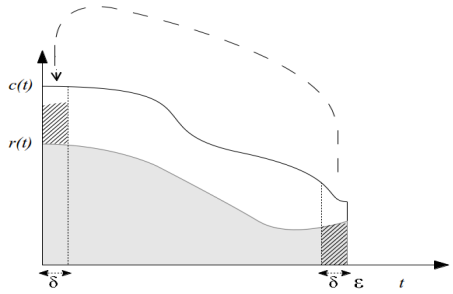

Fig. 1. Sketch of proof of the threshold strategy.

Let (r, γ) denote the video transmission strategy, where r defines the transmission schedule and γ characterizes the bitrate level strategy. We summarize the optimization problem without rebuffering, as follows

$$\min_{(r,\gamma)} \mathcal{F}(r,\gamma) = \frac{1}{T}\int_0^T \frac{r(t)}{c(t)} dt - \pi \times \frac{\sum_{j=1}^{j=L} w_j \int_0^T \gamma_j(t) r(t) dt}{S_L}$$
$$s.t \begin{cases} \int_0^t \frac{\lambda\ c(t)\gamma_1}{b_L} \geq l(t) & \forall t \leq T \\ \int_0^t \sum_{j=1}^{j=L} \frac{\lambda\ r(t)\gamma_j(t)}{b_L} \geq l(t) & \forall t \leq T \\ \int_0^T \sum_{j=1}^{j=L} \frac{\lambda\ r(t)\gamma_j(t)}{b_L} = l(T) \end{cases} \tag{5}$$

where the first constraint ensures the existence of at least one solution that defines a mono-coded video using the lowest bitrate level b_1.

III. Properties of optimal solution without rebuffering events

A. The threshold scheme for transmission schedule

Before introducing the properties of optimal solution, let us define the threshold strategy based on data transmission schedule.

Definition 1. *Giving the network capacity c, we define the threshold transmission schedule by*

$$r_{th}(t) = \begin{cases} c(t) & \text{if } c(t) \geq \alpha \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

Note that in [21], the video is encoded using one quality level. However, considering multiple quality levels with variable size makes our optimization problem different to the one in [21], and the threshold demonstration too.

Proposition 1. *Assume that there exists a feasible solution that satisfies the constraints in (5), then there exists an optimal strategy $(r_{th}, \gamma_{r_{th}})$ of optimization problem (5), where r_{th} is a threshold transmission schedule.*

Proof. Let $c(t)$ and $r(t)$ be the network capacity and the user transmission bitrate (in bps) on a given time interval $[0, \epsilon]$ as depicted in Fig. 1. Without loss of generalityand for the sake of illustration, we choose an interval where $c(t)$

is monotonically decreasing as shown in Fig. 1. As we have $r(t) \leq c(t) \forall t \in [0, \epsilon]$, there $\exists\ (\delta, \beta) \in [0, \frac{\epsilon}{2}] \times [0, 1]$ such that $\forall\ t \in [0, \delta]$

$$c(t) \geq c(t + \epsilon - \delta) \tag{7}$$

and

$$\int_0^{\delta} \frac{r(t) + \beta r(t + \delta - \epsilon)}{c(t)} dt \leq \delta \tag{8}$$

where inequality (7) derives from the decreasing pace of c, and relation (8) derives from the fact that some data at the end can be transmitted beforehand. (According to the figure above, the hatched area on the right can be entirely shifted to the left, which gives a value of β equal to 1). On the other hand, we have

$$\int_0^{\epsilon} \frac{r(t)}{c(t)} dt = \int_0^{\delta} \frac{r(t) + \beta r(t + \epsilon - \delta)}{c(t)} dt + \int_{\delta}^{\epsilon - \delta} \frac{r(t)}{c(t)} dt + \int_{\epsilon - \delta}^{\epsilon} \frac{r(t)}{c(t)} dt - \int_{\epsilon - \delta}^{\epsilon} \frac{\beta r(t)}{c(t - \epsilon + \delta)} dt \tag{9}$$

Using Inequality in (7), we obtain

$$\int_0^{\epsilon} \frac{r(t)}{c(t)} dt \geq \int_0^{\delta} \frac{r(t) + \beta r(t + \epsilon - \delta)}{c(t)} dt + \int_{\delta}^{\epsilon - \delta} \frac{r(t)}{c(t)} dt + \int_{\epsilon - \delta}^{\epsilon} \frac{r(t)}{c(t)} dt) - \int_{\epsilon - \delta}^{\epsilon} \frac{\beta r(t)}{c(t)} dt \tag{10}$$

Obviously, if

$$\int_0^{\delta} \frac{r(t) + \beta r(t + \delta - \epsilon)}{c(t)} dt = \delta,$$

then all the given capacities in $[0, \delta]$ will be used, i.e, all the white surface in Fig 1 will be filled. In that case, we define a new transmission schedule $r'(t)$ such that

$$r'(t) = \begin{cases} c(t) & \text{t} \in [0, \delta] \\ r(t) & \text{t} \in [\delta, \epsilon - \delta] \\ (1 - \beta) r(t) & \text{t} \in [\epsilon - \delta, \epsilon] \end{cases} \tag{11}$$

which gives

$$\int_0^{\epsilon} \frac{r(t)}{c(t)} dt \geq \int_0^{\epsilon} \frac{r'(t)}{c(t)} dt$$

Otherwise, if

$$\int_0^{\delta} \frac{r(t) + \beta r(t + \delta - \epsilon)}{c(t)} dt < \delta, \tag{12}$$

then β will be equal to 1 since we aim at swapping as much data as possible from the worst capacities to the best ones. Therefore, to completely use the highest available capacities, we must repeat this same operation on $[0, \epsilon - \delta]$ considering a new transmission function $r'(t)$ verifying

$$\begin{cases} \int_0^{\delta} \frac{r'(t)}{c(t)} dt = \int_0^{\delta} \frac{r(t) + \beta r(t + \delta - \epsilon)}{c(t)} dt \\ r'(t) = r(t)\ \forall\ t \in [\delta, \epsilon - \delta] \end{cases} \tag{13}$$

In both cases, inequality (10) remains correct, which means that transmitting data using the highest capacity values is less expensive in terms of network utilization cost. As we perform this operation on all the future window, we end up having all the highest capacities fully used and all the lowest one unused, which is clearly a threshold transmission schedule as defined in Definition 1.

Let us now focus on the impact of the threshold strategy on the second term of the cost function and the rebuffering constraints. We need the following conditions to be satisfied

1) the bitrate levels used at the intervals concerned by the swapping-operation must be the same,
2) data swapping should not interrupt a segment transmission schedule, otherwise it is abandoned,
3) swapping data should not violate the stall constraints, otherwise it is abandoned.

It is somehow intuitive that sending data beforehand does not affect, in any way, the quality of video lecture since, as we have assumed, the playback buffer is large enough, and hence there is no risk that packets will be rejected as we transmit much data at earlier times. Therefore, any swapping to earlier higher capacity regions will be performed without violating the stall constraints. However, when it is a matter of a non decreasing capacity-function, each swapping to later higher capacity regions has to be checked whether or not it meets the stall constraints. As we only change the bitrate levels' order without varying the amount of data of each bitrate level, we ensure that the same QoE defined in ρ will be maintained. Thereby, the resulting strategy $(r_{th}, \gamma_{r_{th}})$ improves the performance of the problem in (5), which completes the proof. □

In practice, the decision on α is made progressively in function of time and the future capacities and doesn't follow the previous demonstration, since if not, it becomes complicated to generate such a threshold scheme. In section IV, we design an approach to build a threshold strategy for the transmission schedule.

B. Ascending bitrate level strategy

In this section we study the proprieties of the bitrate level strategy under a threshold based transmission schedule. Precisely, we analyze the impact of video levels' order on the setting of α.

Definition 2. *We say a bitrate level strategy is* **ascending** *if the quality level of segments increases during the session, i.e., for all* $0 \leq t \leq t' \leq T$

$$b(t) \leq b(t')\ \textit{i.e.,}\ \gamma(t) \geq \gamma(t')$$

Proposition 2. *Assume that there exists a threshold-based solution* (r_{th}, γ) *that satisfies constraints in (5), then there exists a threshold-based ascending bitrate level solution* (r'_{th}, γ') *that optimizes problem in (5).*

Proof. Pick a suite of N quality levels in a non-ascending order (a level per segment) that allows a smooth streaming of the video over the future horizon, then, according to this suite, set a threshold solution (r_{th}, γ) such that the first constraints violation will occur at $t = s_n$ if we set a threshold upper than α.

Suppose that b_1 and b_2 are two bitrate levels used respectively on $[\tau, \tau + \delta]$ and $[\tau', \tau' + \delta']$ as depicted in Fig. 2, such that

$$\tau + \delta < s_n, \quad \tau' > s_n, \quad b_1 > b_2$$

and

$$\int_{\tau}^{\tau + \delta} r_{th}(t) dt = \int_{\tau'}^{\tau' + \delta'} r_{th}(t) dt$$

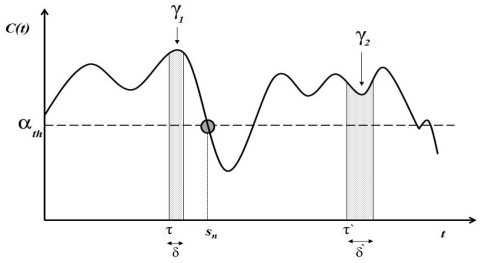

Fig. 2. Ascending bitrate level strategy.

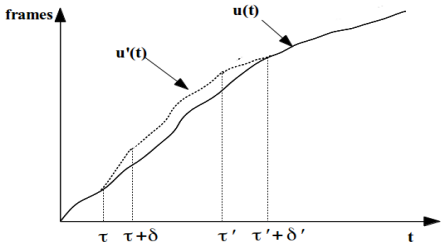

Fig. 3. Impact of bitrate levels switching on the cumulative number of arrival frames u.

Denote by $fr_{th}(t)$ the network frame rate at time t. As $b_1 > b_2$, the number of transmitted frames during $[\tau', \tau' + \delta']$ is greater than the number of frames transmitted on $[\tau, \tau + \delta]$. Therefore, $\exists\ \beta > 0$ such that

$$\int_{\tau'}^{\tau'+\delta'} fr_{th}(t)dt = \int_{\tau}^{\tau+\delta} fr_{th}(t)dt + \beta \tag{14}$$

Suppose that we switch between b_1 and b_2 on the two intervals in question. Then, the number of cumulative received frames at s_n will be increased by β. Let u and u' be respectively the cumulative number of arrival frames function before and after switching, therefore,

$$u'(s_n) = u(s_n) + \beta \tag{15}$$

If we assume that $u'(s_n)$ is large enough to allow increasing the threshold without violating the constraints at $t = s_n$ and later, then the cost function will be reduced, if not, the threshold remains the same without changing the system performance. In fact, as explained in previous sections, sending data beforehand will simply add more flexibility toward constraints at early times, since the buffer is assumed to be big enough. We show by the following that the streaming is still possible with the same threshold once we switch levels' order. Let fr'_{th} be the network frame rate function after switching. We have

$$fr'_{th}(t) \ > fr_{th}(t)\ \forall t \in [\tau, \tau + \delta[\tag{16}$$

$$fr'_{th}(t) \ < fr_{th}(t)\ \forall t \in [\tau', \tau' + \delta'[\tag{17}$$

$$\int_{\tau}^{\tau+\delta} fr'_{th}(t) - fr_{th}(t)\ dt = \int_{\tau'}^{\tau'+\delta'} fr_{th}(t) - fr'_{th}(t)\ dt = \beta \tag{18}$$

We further define u' as

$$u'(t) = \begin{cases} u(t) & \text{t} < \tau \\ u(\tau) + \int_{\tau}^{t} fr'_{th}(s)\ ds & \text{t} \in [\tau, \tau + \delta[\\ u(t) + \beta & \text{t} \in [\tau + \delta, \tau'[\\ u(\tau') + \beta + \int_{\tau'}^{t} fr'_{th}(s)\ ds & \text{t} \in [\tau', \tau' + \delta'[\\ u(t) & \text{t} \geq \tau + \delta' \end{cases} \tag{19}$$

Now, we show that, for any $t \in [0, T]$, $u'(t) \geq u(t)$ and the control $u'(t)$ satisfies the stall constraints (see Fig. 3). Actually, the cumulative watched frames function l will remain the same as the frame rate λ holds the same for all bitrate levels. Obviously, $\forall\ t \notin [\tau', \tau' + \delta'[$, we have $u'(t) \geq u(t)$. However, for $t \in [\tau', \tau' + \delta'[$, we have

$$u'(t) - u(t) = \beta - \int_{\tau'}^{t} fr_{th}(s) - fr'_{th}(s)\ ds, \tag{20}$$

which is clearly positive according to (17) and (18). Finally, streaming the segments in an ascending bitrate levels' order may result in higher thresholds which reduces better the cost function and never regresses the problem performance. Note that a simple switching between levels' order does not impact the QoE term since the proportion of each quality level is kept the same. □

IV. Algorithmic approaches

In this section, we present our optimal approach as well as our proposed heuristic to algorithmically solve optimization problem (5). We start by the case where no rebuffering events may occur during the streaming session, then, to generalize our heuristic, we assume that the user's predicted capacity is insufficient to ensure playing the video without stalls at the lowest quality level.

1) Algorithm for optimal threshold-based solution with ascending bitrate levels: The principle of this algorithm is illustrated in Fig. 4 and can be summarized in three major points: (i) We first look at all the possible values of $\alpha \in [\alpha_{min}, \alpha_{max}]$ that satisfy the constraints in (5) while associating to each one the highest possible video quality, (ii) suppose that we obtain a set of M possible thresholds $\{\alpha_i, i = 1, 2, \ldots, M;\ \alpha_i < \alpha_j\ ,\ i < j\}$. Therefore, for each threshold and its corresponding video quality, we compute the resulting cost function $\mathcal{F}$, (iii) the optimal solution corresponds to the one that minimizes $\mathcal{F}$. The accuracy of this algorithm increases with M at the expand of increasing complexity. In Section IV-4, we present an heuristic approach to determine the footstep $\alpha_{i+1} - \alpha_i$ which directly impacts the value of M.

2) Approach for optimal thresholds: To obtain optimal thresholds with the lowest complexity we propose to try all the capacity values in an ascending way, c_{min} will correspond to the lowest threshold, the highest threshold, however, will be defined when a constraint violation appears for the first time. Fig.7 illustrates the example used for the simulation section.

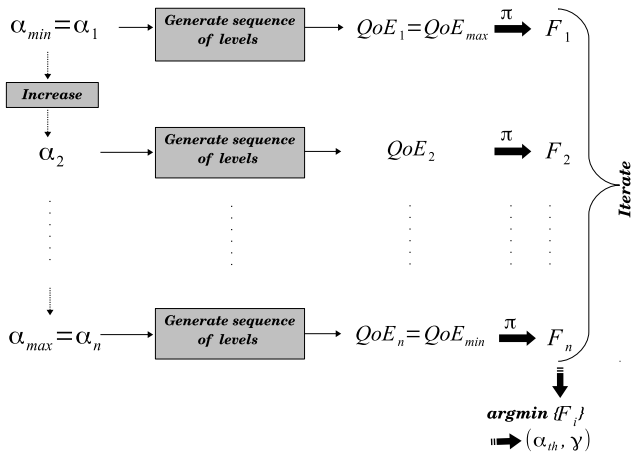

Fig. 4. Optimal threshold-based solution with ascending bitrate levels.

3) Approach for optimal ascending bitrate levels: To maximize the QoE with an ascending bitrate level strategy, we construct a tree of N levels, where each level corresponds to a segment of the video. Each node of a tree level i corresponds to a quality level that can be used for the i^{th} segment such that its parent-node (if it exists) is a level of a worse or equal quality, its child-nodes (if they exist) are levels of a better or equal quality. We construct the tree level by level. At each level, we remove the nodes that cause a constraint violation so that to minimize the number of nodes at the bottom of the tree. At each level, we compute the partial-QoE till reaching the end of the tree. The optimal suite of bitrate levels corresponds to the path that maximizes the QoE. The key shortcoming of this algorithm is its complexity that may reach up to $\mathcal{O}((L+1)^N)$. Clearly, this algorithm is very consuming in processing time and, thus, is very hurdling for online streaming services. In the next section, we propose a heuristic that gives sub-optimal strategy with polynomial complexity.

4) Heuristic for aNticipating qoE With threshold sCheme And aScending biTrate levels (NEWCAST): Let γ_α and $\mathcal{F}_\alpha$ be, respectively, the ascending bitrate level strategy and the cost function under r_α-based transmission schedule. The main steps of the proposed heuristic are described in Algorithm 1, where INVEST represents the approach for generating sub-optimal thresholds and AWARE represents the heuristic for setting sub-optimal ascending bitrate levels.

5) Heuristic for generating thresholds: INcrease with VariablE foot STep (INVEST): The increase on α is performed by adding a variable footstep at each iteration depending on the dynamic of the network capacity. The idea behind this approach is that setting a constant small footstep over the whole algorithm can be judicious at some iterations, but may lead to higher complexity. On the other hand, setting a constant high footstep can be reasonable at some iterations, but may result on lower accuracy. To overcome this hurdle, we fix the amount of data that we wish to abandon at each step (denoted by Q). Then, we adjust the value of α which gives the corresponding variable footstep (see Fig. 5). The main steps of the proposed approach can be found in [22].

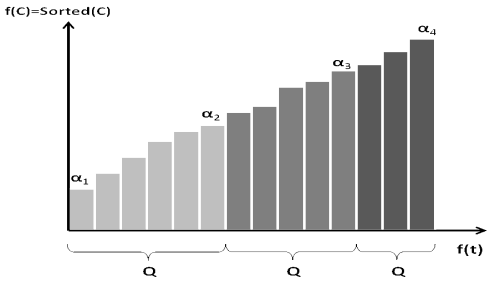

Fig. 5. INVEST: INcrease with VariablE foot STep.

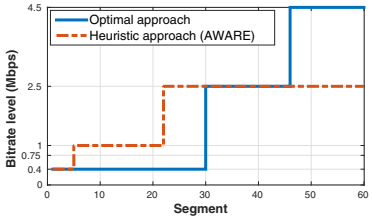

Fig. 6. Comparative example between optimal approach and AWARE.

6) Heuristic for Anticipating qoe With Ascending bitRate lEvels (AWARE): Here we propose a fast heuristic to maximize the QoE. For space limitation, we put the algorithm description in [22]. Our simulation results show that the outcoming solution of AWARE approaches at 98% the optimal solution in terms of QoE. The principle of the proposed heuristic is as follows: We start by assigning the lowest level to all video segments. Then, we keep increasing the levels progressively starting by the end of the video as long as the constraints are satisfied. Once a constraint is violated, we choose the previous level of the segment. The number of loops is equal to $L-1$. To reduce at maximum the startup delay, which is a prominent key QoE factor (but not included in our optimization problem), we set by default the buffering-cache segments to the lowest video quality and use a greedy[1] transmission instead of using a threshold-based transmission. As can be seen in Fig. 6, an inherent advantage of this algorithm is that it ensures a progressive increase of the quality levels instead of an aggressive increase as in the optimal solution, which is quite more appreciable for the user's perception.

A. Approaches under rebuffering events

So far, we have assumed that no rebuffing events are acceptable for the video streaming. To go further with the analysis, we adapt our approach to the case where K stalls happen when using the lowest quality and maximum system utilization cost. In such cases, we proceed as follows: We first spot the K segments where the stalls may happen, then, we divide the video into $K+1$ parts. At each part, we

[1]A greedy transmission uses all the available network capacities.

Algorithm 1: NEWSCAST: aNticipating qoE With threshold sCheme And aScending biTrate levels

Data: $c, VideoProperties, L, w, Q$

$\alpha \leftarrow c_{min}$; $i \leftarrow 1$;

$[PossibleTransmission, r_\alpha, \gamma_\alpha] =$ ***AWARE***$(c, \alpha, videoProperties, L)$

while $PossibleTransmission$ **do**

  $\mathcal{F}_\alpha = computeObjFunction(c, r_\alpha, \gamma_\alpha, w)$

  $i = i + 1$

  $\alpha =$ ***INVEST***(c, i, Q)

  $[PossibleTransmission, r_\alpha, \gamma_\alpha] =$ ***AWARE***$(c, \alpha, videoProperties, L)$

end

$\mathcal{F}^*_{\alpha^*} = min\{\mathcal{F}_\alpha\}$

$\alpha_{th} = \alpha^*$

return $(\alpha_{th}, \gamma_{\alpha_{th}})$

independently run NEWCAST as if we had different streaming sessions. This approach keeps the same results at the first parts of the video (just before the K^{th} stall), while at the same time, tends to enhance performance at the last part since it may result in lower system utilization or higher QoE (depending on the value of π).

V. Implementation and Simulation Results

A. Simulation tools and setup

All the simulations have been performed using Matlab server R2015b on a Dell PowerEdge T420 Intel Xeon running Ubuntu 14.04. The network capacity is randomly generated around a given mean throughput and the streaming session is configured according to some DASH and Youtube parameters [23] [24]. To the best of our knowledge, there is no explicit way to compute the weight that can be accorded to each video level. In [16], authors explore a QoE estimation model in which each video segment is assigned a QoE_{seg} that varies logarithmically with its bitrate level and its motion factor. In [25], however, authors assign a MOS (Mean Opinion Score) factor to reflect the user's satisfaction toward the streamed video quality. In this paper, we assign the weights proportionally to the corresponding bitrate levels as follows

$$w_i = \frac{b_i}{\sum_{i=1}^{L} b_i},$$

where b_i is the i^{th} bitrate level and w_i is its corresponding weight. All the parameters are listed in Table I. To give more accuracy to our simulation results, we explore the values of α in an optimal way. Our heuristic approach (INVEST) is discussed in [22].

Window Size	3 min 10 s
Mean throughput	2 Mbps
Capacity Time Slot	1 s
Video Length	3 min
Segment Length	1s
Video frame rate	30 fps
Playback cache	4s
Bitrate levels Mbps	[0.4 0.75 1 2.5 4.5]
Levels weights	[0.09 0.17 0.22 0.55 1]

TABLE I
Simulations parameters.

B. Simulation results

Fig. 7 outlines the dynamic of the capacity used for all the simulation section and its correspondent threshold α. It will be impossible to send the video with the lowest quality if α exceeds its maximum value. When we set α to c_{min}, we have the maximum cost in network utilization and the maximum QoE as if we did not apply a threshold strategy. The latter case will be referred to as a benchmark from now on.

The running of NEWCAST shows a variation on the system performance for π between 4.50 and 4.70. In the following, we focus our analysis on three values of π: low, medium and high. Denote by α_{th} the resulting threshold α after running NEWCAST.

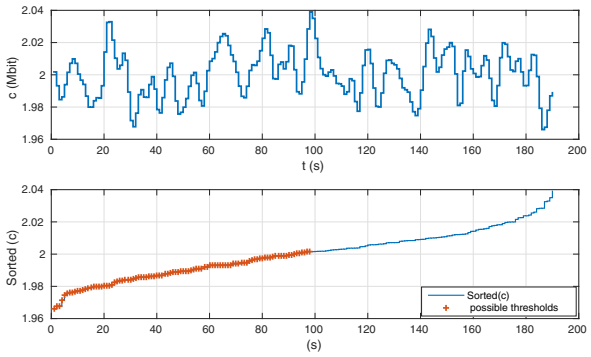

Fig. 7. Network capacity and threshold α.

Fig. 8 depicts the variation of α_{th} as function of π. A low value of π results in a high α_{th} as it prioritizes the system cost. A high value of π, however, implies that the system uses almost all the future resources and gives a low threshold as it accords more importance to the QoE. A medium π results in a medium threshold which balances between QoE and system cost.

In Fig. 9 we plot the playback buffer state evolution over time and its correspondent streamed bitrate levels for the three aforementioned values of π. When π is small, many silent times are noticed and the buffer state evolves with a high slope (mainly at the beginning and the middle of the video). This is due to the bad quality of the segments being streamed. Notice that the player streams as much frames as the bitrate level decreases. More flexibility is noticed for a medium value of π, with shorter silent times and better QoE. For a big value of π, no silent times are noticed since almost all the network resources are being used. The buffer state evolves gradually with a low slope, given the fact that segments of high-order level are being streamed.

Let us now study the impact of enforcing a stall on the performance of our heuristic. In Fig. 10, we plot the initial

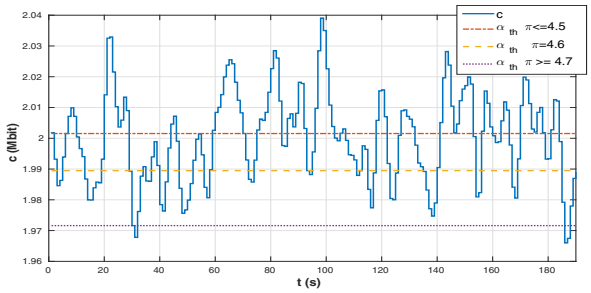

Fig. 8. Variation of α_{th} as function of π.

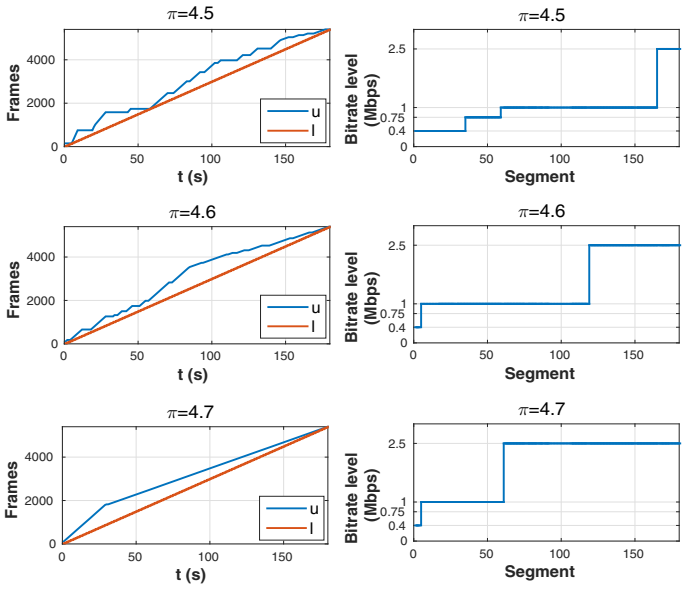

Fig. 9. Playback buffer state evolution and corresponding bitrate levels for different π.

playback buffer state variation over the time for the same three values of π. We compute the resulting $\mathcal{F}$ before and after enforcing a stall at multiple emplacements of the video. As illustrated in Fig. 10, for $\pi = 4.5$, high fluctuations are noticed around the original $\mathcal{F}$ mainly at the beginning of the video. The lowest values of $\mathcal{F}$ are obtained at moments where the initial buffer state is critical, i.e., no much flexibility regarding the stall constraint and a low quality level. At these moments, the critical state of the buffer leads to set a low threshold to avoid a constraint violation. Enforcing a stall at this moment, results in increasing the threshold at the right part of the video, which reduces the overall system cost. Now, by increasing π, we observe a decreasing on the resulting function $\mathcal{F}$; a stall enforcement certainly enhances the quality at the beginning of the video, but it condemns the flexibility and the QoE for the rest of the video. The degradation in global QoE induces a reduction in global system cost that outweighs the resulting $\mathcal{F}$. To sum it up, a stall enforcement would be only interesting for low values of π, since it may reduce the system cost. A judicious choice of the stall's emplacement would be at the moments where the initial buffer state is critical.

C. Robustness under prediction errors

One key limitation of our work is that there is still no accurate approach to predict well the network capacity over up to ten seconds of the future. In order to evaluate the robustness of our approach, we use the HSDPA dataset in [1]. It consists of 30 minutes of continuous measurements of throughput of a moving devices in Telenor's 3G/HSDPA wireless mobile network. We use traces of the Ljabru-Jernbanetorget trajectory as it has the least variance in throughput spatial variability. A temporal mapping of the streaming throughput is performed by supposing that the user is moving at a speed of 50Kmph. Assuming the same configuration as in Table I, we evaluate the robustness of NEWCAST through the difference between performance at the optimal solution using one realization throughput ($\mathcal{P}_{real}$) and performance at the optimal solution using mean-throughput ($\mathcal{P}_{mean}$), where mean-throughput is the average throughput of all throughput realizations. The average error rate $\mathcal{P}_{error}$ can be expressed as

$$\mathcal{P}_{error} = \left| \frac{\mathcal{P}_{real} - \mathcal{P}_{mean}}{\mathcal{P}_{mean}} \right|.$$

Results in Fig. 12 show an average error rate less than 15% for both the system cost and the QoE. We also observe that the system cost is less sensitive for small values of π, whereas the QoE is less sensitive for high values of π. In general, it is clear that our scheme performs pretty well even with errors in the prediction of the future capacity.

VI. Conclusion

In this paper, we have developed a new scheme, which we called NEWCAST, for optimizing the delivery of video streaming. NEWCAST approach leverages on the knowledge of future capacity. It is designed to achieve better system utilization while guaranteeing the highest possible QoE to the end-user in terms of video quality and rebuffering events. From an implementation point of view, results have shown the possibility to use NEWCAST as an online protocol (well suited for dynamic adaptive streaming over HTTP). Moreover, NEWCAST has proven to be highly robust to prediction errors. Interesting future direction consists in incorporating errors on throughput prediction in the utility function in order to improve the robustness of our approach.

Acknowledgement

This work has been carried out in the framework of IDEFIX project, funded by the ANR under the contract number ANR-13-INFR-0006.

References

[1] Dataset: Hsdpa-bandwidth logs for mobile http streaming scenarios. http://home.ifi.uio.no/paalh/dataset/hsdpa-tcp-logs/.

[2] J. De Vriendt, D. De Vleeschauwer, and D. Robinson, "Model for estimating qoe of video delivered using http adaptive streaming," in *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, May 2013, pp. 1288–1293.

[3] F. Dobrian, A. Awan, D. Joseph, A. Ganjam, J. Zhan, V. Sekar, I. Stoica, and H. Zhang, "Understanding the impact of video quality on user engagement," in *ACM SIGCOMM*, 2011.

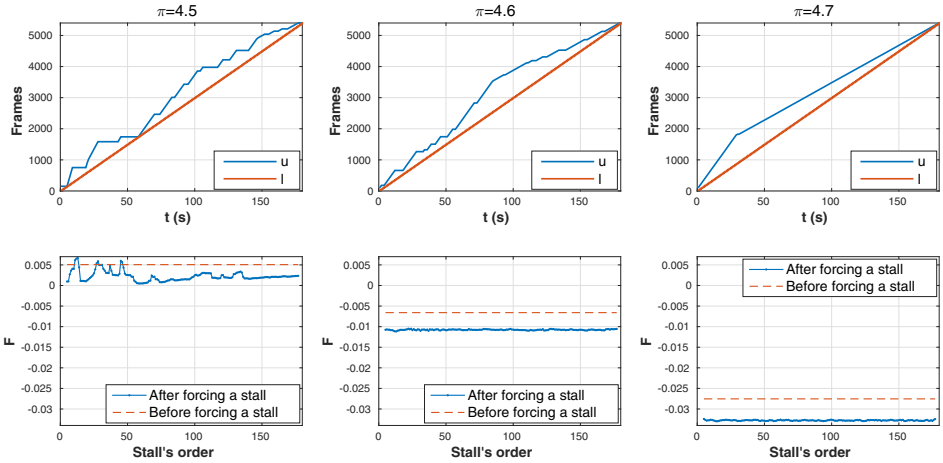

Fig. 10. System performance with buffer stall enforcement.

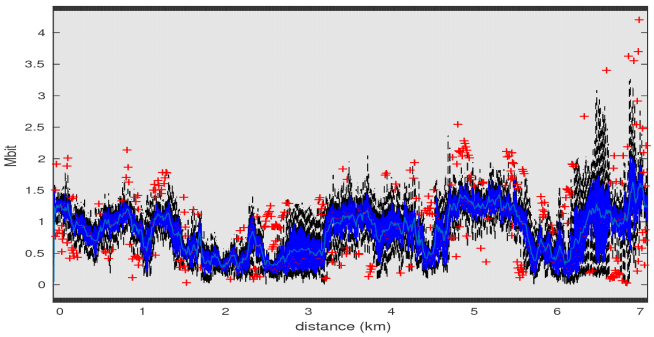

Fig. 11. Experimental spatial variations of the capacity on the tramway Ljabru-Jernbanetorget trajectory.

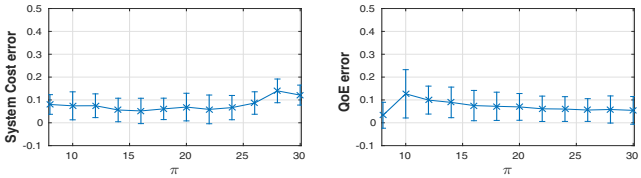

Fig. 12. Average error rate on the system performance under throughput prediction errors.

[4] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang, "Developing a predictive model of quality of experience for internet video," in *ACM SIGCOMM*, 2013.

[5] M. Z. Shafiq and J. Erman and L. Ji and A. X. Liu and Jeffrey Pang and Jia Wang, "Understanding the impact of network dynamics on mobile video user engagement," in *SIGMETRICS*, 2014.

[6] C. Yim and A. C. Bovik, "Evaluation of temporal variation of video quality in packet loss networks," in *Signal Processing: Image Communication*, 2011.

[7] T. Stockhammer, "Dynamic adaptive streaming over http: Standards and design principles," in *Second Annual ACM Conference on Multimedia Systems*, 2011.

[8] T. C. Thang, H. T. Le, A. T. Pham, and Y. M. Ro, "An evaluation of bitrate adaptation methods for HTTP live streaming," *IEEE Journal on Selected Areas in Communications*, 2014.

[9] 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 *ACM Conference on Special Interest Group on Data Communication (SIGCOMM)*, 2014.

[10] G. Tian and Y. Liu, "Towards agile and smooth video adaptation in dynamic http streaming," in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, 2012.

[11] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," *SIGCOMM Comput. Commun. Rev.*, pp. 325–338, 2015.

[12] A. Jain and A. Terzis and N. Sprecher and P. Szilagyi and H. Flinck, "Mobile Throughput Guidance Signaling Protocol draft-flinck-mobile-throughput-guidance-00," *IETF*, April 2014.

[13] Expert Working Group on 5G Challenges Research Priorities, and Recommendations, "Networld 2020 etp," *White paper*, Aug. 2014.

[14] 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*, 2015.

[15] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate adaptation for adaptive http streaming," in *ACM Multimedia Syst.*, 2011.

[16] Y. Shen, Y. Liu, Q. Liu, and D. Yang, "A method of qoe evaluation for adaptive streaming based on bitrate distribution," in *IEEE International Conference on Communications, Workshops Proceedings*, 2014.

[17] S. Egger, B. Gardlo, M. Seufert, and R. Schatz, "The impact of adaptation strategies on perceived quality of http adaptive streaming," in *Proceedings of the 2014 Workshop on Design, Quality and Deployment of Adaptive Video Streaming*, 2014.

[18] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive," in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, 2012.

[19] A. K. Moorthy, L. K. Choi, A. C. Bovik, and G. de Veciana, "Video quality assessment on mobile devices: Subjective, behavioral and objective studies." *J. Sel. Topics Signal Processing*, pp. 652–671, 2012.

[20] V. Joseph and G. de Veciana, "NOVA: QoE-driven optimization of DASH-based video delivery in networks," in *INFOCOM, Proceedings IEEE*, April 2014.

[21] Z. Lu and G. de Veciana, "Optimizing stored video delivery for mobile networks: The value of knowing the future." in *INFOCOM*, 2013.

[22] Imen Triki, Rachid El-Azouzi and Majed Haddad, "Anticipating resource management and qoe provisioning for mobile video streaming," 2015. [Online]. Available: http://arxiv.org/abs/1512.05705

[23] Parametres, debits et resolutions de l'encodeur d'evenements en direct. https://support.google.com/youtube/answer/2853702?hl=fr.

[24] S. Lederer, C. Müller, and C. Timmerer, "Dynamic adaptive streaming over http dataset," in *Proceedings of the 3rd Multimedia Systems Conference*, 2012.

[25] A. E. Essaili, D. Schroeder, D. Staehle, M. Shehada, W. Kellerer, and E. G. Steinbach, "Quality-of-experience driven adaptive http media delivery." IEEE, 2013, pp. 2480–2485.