Dynamic DASH Aware Scheduling in Cellular Networks

Rachid El-Azouzi, Albert Sunny, Liang Zhao, Eitan Altman, Dimitrios Tsilimantos, Francesco de Pellegrini, Stefan Valentin

Dynamic DASH Aware Scheduling in Cellular Networks

Rachid El-Azouzi*, Albert Sunny†, Liang Zhao‡, Eitan Altman§,
Dimitrios Tsilimantos¶, Francesco De Pellegrini* and Stefan Valentin‖
*CERI/LIA, University of Avignon, Avignon, France; †Indian Institute of Technology, Palakkad, India
‡Fuden, University, Shangai, China; §INRIA, Sophia Antipolis, France;
¶France Research Center, Huawei Technologies Co. Ltd. France;
‖Darmstadt University, Darmstadt, Hessen, Germany

***Abstract*—Dynamic Adaptive Streaming over HTTP (DASH) has become the standard choice for live events and on-demand video services. In fact, by performing bitrate adaptation at the client side, DASH operates to deliver the highest possible Quality of Experience (QoE) under given network conditions. In cellular networks, in particular, video streaming services are affected by mobility and cell load variation. In this context, DASH video clients continually adapt the streaming quality to cope with channel variability. However, since they operate in a greedy manner, adaptive video clients can overload cellular network resources, degrading the QoE of other users and suffer persistent bitrate oscillations.**

In this paper, we tackle this problem using a new eNB scheduler, named `Shadow-Enforcer`, which ensures minimal number of quality switches as well as efficient and fair utilization of network resources. Our scheduler works well under dynamic scenarios and mobility, and requires minimal information, i.e., just the set of video bitrates supported by DASH video clients. It consists of the cascade of a virtual scheduler, `Shadow`, and the actual scheduler, `Enforcer`, piloted by the virtual one. Extensive simulations demonstrate the efficiency, fairness and the smooth response to channel variations of the proposed solution.

***Index Terms*—DASH video streaming, quality of experience, cellular networks, fairness, scheduling**

I. Introduction

According to recent statistics, video streaming dominates today's traffic share and is expected to reach 90% by 2020 [1]. Since a large part of such multimedia traffic is served to wireless devices such as smartphones, tablets and laptops over cellular networks, mobile operators have been forced to deliberate on their current resource management solutions. In fact, while in recent years significant progress has been made towards increasing the capacity of cellular networks, users' *Quality of Experience (QoE)* – which is key in multimedia service provisioning – has in turn become a challenging and prominent issue in mobile networks, and extensive studies are being carried out on how to achieve better user QoE.

To efficiently trade-off network performance and QoE, researchers are exploring *HTTP Adaptive Streaming (HAS)* enabled architectures [2]–[5]. In fact, *Dynamic Adaptive Streaming over HTTP (MPEG-DASH)* has become the standard choice for live streaming events and on-demand streaming services, including those provided by contents providers such as Amazon, YouTube and Netflix. In a typical DASH system, the video file is encoded at different bitrates called representations. Each representation is split into video segments of equal duration. The video is then stored on a web server and streamed to the client via a series of HTTP requests. DASH does not include specifications on how the bitrate adaptation should be performed, rather, it delegates this task to the client. However, as a matter of fact, DASH clients are designed to perform video streaming at the highest possible QoE while being aware of network congestion.

Current cellular networks incorporate radio resource management techniques which are conceived to meet the QoE requirements of traditional single-rate video streams and elastic data flows [6]. On the other hand, adaptive streams have multiple possible QoE requirements. Furthermore, they are greedy because the client always seeks to download the maximum number of segments at the highest quality possible. Thus, the presence of adaptive video streams requires careful resource management in order to take into account such factors and avoid possible drawbacks, namely, (i) instability due to unnecessary switching of video bitrate (ii) unfair resources allocation to DASH traffic against other traffic types (iii) inefficient network utilization.

Researchers have explored different approaches to handle the above mentioned issues by developing cross-layer mechanisms to improve QoE of adaptive video streams. The authors of [7] propose a cross-layer scheme to optimize the total utility of all clients while maintaining stable video quality and supporting user and device-specific needs. AVIS presented in [8] is a cross-layer scheme that can separate resource management of adaptive video flows from that of regular video flows. While cross-layer schemes ensure good performance, they require coordination among the video server, the client and the eNB. But, due to various practical reasons, namely scalability and operator policies, such level of coordination is often not feasible in current cellular networks. Furthermore, differentiating regular and adaptive video sources is becoming difficult, because more-and-more video content providers are adopting strong encryption, which prevents the

usage of network monitoring techniques such as *deep packet inspection* [9]. In [10], the authors proposed a new scheduler called D-VIEWS, to enforce bitrate stability for each DASH flow in the networks. D-VIEWS performs well in static scenarios but in dynamic scenarios the scheduler tends to generate quality switches due to a reset mechanism employed to handle variations in resources utilization due to user arrival/departure and mobility. In fact, after every reset duration, the scheduler would run in the default mode, acquire target rates and then introduce a penalty. This periodic reset forces user's target rate to adapt at each user arrival/departure, with the unintended consequence that all video streams would incur several quality switches due to the arrival/departure process.

A. Our Contributions

In this paper, we design the `Shadow-Enforcer` scheduler — a new scheduling scheme that effectively solves the major challenge of allocating radio resources to multiple adaptive video streaming while being fair to other traffic types. It is also capable of reducing the occurrences of quality switches compared to state of the art solutions. This mechanism contains two parts: (a) the `Shadow` scheduler which is a *virtual* scheduler, i.e., it allows us to keep track of the average user throughput that a standard fairness scheduler can achieve; the virtual scheduler covers the well-known class of α-fair measures [11], (b) the `Enforcer` scheduler, used to perform the actual allocation of resource blocks (RBs) to active DASH users; it receives as input the target rate provided by the `Shadow` scheduler.

The main principles behind the design of our scheduler are:
1. Minimal video switching rate: the scheduler shall ensure that the average throughput of the users takes values only in the set of available DASH bitrates.
2. Quick reaction in the event of arrivals/departures and mobility: the scheduler shall be capable of reacting to changing network conditions, arrival/departure events and user mobility.
3. Quick convergence to the optimal target bite-rate: the proposed scheduling algorithm should converges quickly to a stable solution.
4. Practicality and stand-alone independent operation: the proposed scheduler shall not require coordination of video client and eNB's scheduler: as such, it shall be able to function with just information about the video bitrates supported by adaptive streams from different content providers.

II. Motivation

In this section, we describe the DASH standard and resource allocation schemes on LTE cellular networks.

A. Adaptive Streaming Traffic

Traditional non-adaptive streaming may suffer startup delays and buffering during the video session, which causes intolerable video freezing events on the users' playout application. This is typically due to capacity outage, an event relatively common in wireless access channels. The first HTTP adaptive streaming solution was proposed by Move Networks in 2006 [12], [13]. Thereafter, it has garnered significant attention of the multimedia community through commercial solutions such as, Microsoft Silverlight Smooth Streaming (MSS), HTTP Live Streaming (HLS) [14] by Apple and HTTP Adaptive Dynamic Streaming (HAS) [15] by Adobe Systems Inc.

Over time, DASH [16] has become the de-facto HAS solution of choice for content providers. This is due to the fact it attains efficient trade-off between smooth streaming and visual quality. In DASH, each video file is divided into multiple small segments and each segment is encoded into multiple quality levels. Based on the available capacity, the client dynamically chooses the quality level of the segment such that the visual quality is maximized at a low risk of depleting the playback buffer. This is done assessing current network conditions, e.g., the measured or estimated throughput or media playout buffer (play out buffer fill level). Based on this information, an adaptation engine chooses the appropriate bit-rate for the next segment. This strategy can ensure good video quality while avoiding playback interruptions.

The drawback of HTTP Adaptive Streaming (HAS) protocols such as DASH is the potential instability in the presence of multiple video streaming flows. Such instability appears as persistent oscillation of the video bitrate. The root cause of the instability is the fact that each DASH source reacts myopically to variations in channel conditions generated by the presence of other clients who compete for same wireless resource. From an infrastructure point of view, this motivates us to design fair schedulers able to arbitrate multiple flows and induce adaptive rate throttling at the RAN level, while HAS flows are rate-limited in a stable manner.

B. QoE Metrics

The notion of QoE has emerged since the late 90s. It has gained significant attention because the de facto notion of quality at the time, e.g., the Quality of Service (QoS), was unable to fully express the performance of modern communication systems, which are increasingly complex. To quantify the users' perception of the quality of video streaming applications, QoE metrics have been defined and used by industry and the scientific community. They are known as QoE Key Performance Indicators (KPI), i.e., user-based metrics capturing the user's perception of a service. The following are a few popular KPIs used by adaptive video streaming:
1. Average bitrate: the average bitrate is the mean quality of the streamed video, and is denoted by the bitrate in bits per second (bps).
2. Number of bitrate switching: number of times the video quality has changed during its playback.
3. Buffering ratio: the ratio of the time the video player spends in stalling to the total play time (including the stalling time).
4. Initial delay or Startup delay: duration (measured in seconds) between the time that a user initiates a video session and the time that the media player starts playing video frame.

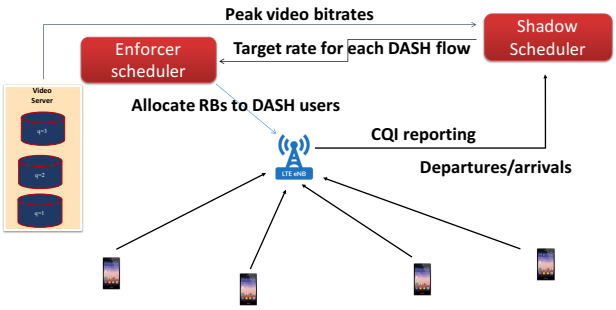


Fig. 1: Shadow-Enforcer scheduling architecture

III. Problem Formulation

LTE downlink transmission resources, through the use of OFDMA, cover both time and frequency. The frequency dimension is divided into sub-carriers, whereas the time dimension is divided into $10\,ms$ radio frames. Each frames is further subdivided into ten $1\,ms$ subframes, each of which is split into two $0.5\,ms$ time slots. The smallest unit of resource which can be allocated to a user is referred to as the *Resource Block (RB)*. In the remainder of the paper, we denote the set of resource blocks available with a tagged eNB as $\mathcal{C}$.

Without loss of generality, we consider two classes of traffic or service types: DASH traffic (D) and non-DASH traffic (E). Non-DASH traffic can consist of regular, i.e., non-adaptive video flows, elastic traffic, e.g., FTP transfers and real-time traffic, e.g., audio and video calls. We assume that an LTE downlink associated with a tagged eNB delivers traffic to mobile users belonging to both classes. As introduced before, with DASH, each video is divided into multiple segments, and each segment is encoded into multiple bitrates/resolutions at the server. Let $\mathcal{L} = \{l_1, l_2, \ldots, l_m\}$ denote the set of video bitrates supported by the DASH flows. Flows of class $k \in \{D, E\}$ arrive at a rate λ_k and have sizes that follow a given distribution function $G_k(.)$, with mean value μ_k. We denote $\mathcal{N}_D$ and $\mathcal{N}_E$ as the set of DASH and non-DASH traffic flows, respectively.

IV. Shadow-Enforcer Scheduler for DASH Traffic

We recall that the primary objective of our scheduler is to assign resources in such a way so as to avoid DASH video quality oscillations. To achieve this, we use two schedulers in conjunction so as to obtain efficient, stable and fair resource allocation among users. The first scheduler, referred to as `Shadow`, computes the target rates for the DASH flows. Whereas, the second scheduler, referred to as `Enforcer`, is responsible for allocating RBs to the active users following the target rate provided by the first scheduler. Fig. 1 illustrates the whole `Shadow-Enforcer` scheme formed by the cascade of these two schedulers.

A. `Shadow` Scheduler

In this section, we present a technique to obtain the optimal target rate vector for all DASH flows. Let $\mathcal{Z}_i = \prod_{c \in \mathcal{C}_D} \mathcal{Z}_{i,c}$ denote the collection of channel rate of user i across the resource block. Here, $\mathcal{Z}_{i,c}$ denotes the set of possible rates for user'i channel on resource block c, and C_D denotes the set of resource blocks allocated to DASH flows. Then, the set of joint channel rates is given as $\mathcal{Z} = \mathcal{Z}_1 \times \mathcal{Z}_2 \times \cdots \times \mathcal{Z}_{n_D}$ and n_D is the number of DASH flows in the system. We remark that `Shadow` is a *virtual* scheduler, since the actual resource allocation is performed by the subsequent module, i.e., the `Enforcer` scheduler.

The `Shadow` scheduler is designed as a standard utility-based scheduler. We choose such a scheme because it can be easily integrated into existing eNBs with minimal modifications. The utility of each user $i \in \mathcal{N}_D$ is assumed to belong to the well-known class of fairness measures called α-fairness [18]. Specifically, we have

$$u_i(r_i) = \begin{cases} \log r_i & \text{if } \alpha_i = 1 \\ \frac{{r_i}^{1-\alpha_i}}{1-\alpha_i} & \text{otherwise} \end{cases}$$

where r_i is the long-term average throughput of user i. We recall that α-fairness is a general fairness measure that satisfies the four axioms from [11]. If $\alpha_i = 1$, α-fairness becomes the well-known *proportional fairness*; when all $\alpha_i \to \infty$, the measure becomes *max-min fairness*.

The scheduling decision variable $a_i(c, \mathbf{z})$, under joint channel state $\mathbf{z}$, represents the fraction of frames where resource block c is allocated to user i. The overall throughput $r_i(\mathbf{a})$ of user i is the sum of its throughput over all the resource blocks, i.e.,

$$r_i(\mathbf{a}) = \sum_{\mathbf{z} \in \mathcal{Z}} \sum_{c \in \mathcal{C}_D} \pi(\mathbf{z}) a_i(c, \mathbf{z}) \cdot z_{i,c} \tag{1}$$

where $\pi(\mathbf{z})$ is the probability of occurrence of the joint channel state $\mathbf{z}$, $z_{i,c}$ is the instantaneous channel capacity of user i on resource block c and vector $\mathbf{a} = (\mathbf{a}_1, \ldots, \mathbf{a}_{n_D})$. Thus, the optimization problem of the utility-based scheduler is formulated as shown below:

$$P_1 : \max_{\mathbf{a}} \sum_{i \in \mathcal{N}_D} u_i(r_i)$$

Subject to:

$$\sum_{i \in \mathcal{N}_D} a_i(c, \mathbf{z}) \leq 1 \qquad \forall c \in \mathcal{C}_D,\ \forall \mathbf{z} \in Z \tag{2}$$

$$a_i(c, \mathbf{z}) \geq 0 \qquad \forall i \in \mathcal{N}_D, c \in \mathcal{C}_D, \mathbf{z} \in Z \tag{3}$$

In the above formulation, Inequalities (2) and (3) capture the scheduling constraint. It has been shown in [19]–[21] that the above optimization problem can be solved by an online algorithm as follows: the eNB obtains the instantaneous channel quality (CQI) for each user in every resource block in time slot t. Then, it keeps track of the average throughput for each user i, and allocates resource blocks to users as follows

$$i_c^*(t) = \arg \max_{i \in \mathcal{N}_D} {u_i}'(r_i(t)) \cdot z_{i,c}(t)$$

where $z_{i,c}$ is the instantaneous capacity of user i on resource block c in time-slot t, $i_c^*(t)$ is the user allocated to resource block c at time slot t, and $r_i(t)$ is the average throughput of

user i till time t. The value of the average rate $r_i(t)$ is updated as follows

$$r_i(t+1) = (1-\beta)\cdot r_i(t) + \beta \sum\nolimits_{c\in\mathcal{C}} z_{i,c}(t)\cdot \mathbb{I}_{\{i_c^*(t)=i\}}$$

where $\mathbb{I}_{\{i_c^*(t)=i\}}$ is the indicator function for the event that user i is allocated resource block c at time slot t by the `Shadow` scheduler. Here, β is the memory of the averaging filter. In this type of schedulers, β can be also interpreted as the time-scale over which the scheduler aims to achieve the target for each DASH flow. Finally, the `Shadow` scheduler computes the *target rates*, for the DASH flows, from the average rate as below

$$\bar{r}_i(t) = \begin{cases} 0 & \text{if } r_i(t) = 0 \\ \max\{l_j \mid l_j \le r_i(t),\ 1 \le j \le m\} & \text{otherwise} \end{cases} \tag{4}$$

B. `Enforcer` Scheduler

The goal of the `Enforcer` scheduler is to allocate resource blocks of the eNB across a set of users streaming DASH flows. Specifically, it takes as input the target rate of each DASH flow provided by the virtual `Shadow` scheduler (refer to (4)). Since the `Shadow` scheduler runs continuously in the background, the target rate is updated in each time slot. This ensures that the whole system is reactive to user mobility and arrival/departure events. We recall that the shadow scheduler strives to attain fairness among DASH flows. The task of the `Enforcer` scheduler, conversely, is to reduce the average bitrate switch rate perceived by DASH users while maintaining high average bitrate allocated to different DASH flows.

Let $\bar{r}_i$ be the target bitrate input to `Enforcer`. The goal of the `Enforcer` is to ensure that the average throughput of DASH flow i matches $\bar{r}_i$. This result can be achieved by solving the following optimization problem

$$P_2 : \min_{\mathbf{a}} \sum_{i\in\mathcal{N}_\mathcal{D}} (\gamma_i(\mathbf{a}) - \bar{r}_i)^2$$

Subject to:

$$\sum_{i\in\mathcal{N}_D} a_i(c,\mathbf{z}) \le 1 \qquad \forall c\in\mathcal{C},\ \forall \mathbf{z}\in\mathcal{Z}$$

$$a_i(\mathbf{c},\mathbf{z}) \ge 0 \qquad \forall i\in\mathcal{N}, c\in\mathcal{C}, \mathbf{z}\in\mathcal{Z}$$

where

$$\gamma_i(\mathbf{a}) = \sum_{\mathbf{z}\in\mathcal{Z}} \sum_{c\in\mathcal{C}_D} \pi(\mathbf{z}) a_i(c,\mathbf{z})\cdot z_{i,c}$$

Optimization problem P_2 can be solved by following the following allocation policy

$$\bar{i}_c(t) = \arg\max_{i\in\mathcal{N}_D} (\bar{r}_i - \gamma_i(t))\cdot z_{i,c}(t)$$

where $z_{i,c}(t)$ is the instantaneous capacity of user i on resource block c in time-slot t, $\bar{i}_c(t)$ is the user allocated to resource block c in time-slot t, and $\gamma_i(t)$ is the average throughput of user i till time t. Further, the value of $\gamma_i(t)$ evolves as follows

$$\gamma_i(t+1) = (1-\beta)\cdot\gamma_i(t) + \beta\sum\nolimits_{c\in\mathcal{C}} z_{i,c}(t)\cdot\mathbb{I}_{\{\bar{i}_c(t)=i\}} \tag{5}$$

Algorithm 1 `Shadow-Enforcer` scheduling policy

Input: set of DASH flows $\mathcal{N}_D$; set of DASH video bitrates $\{l_0 = 0\} \cup \{l_1, l_2, \ldots, l_m\}$
Output: User-resource block allocation for each time slot $t \ge 1$, i.e., $\{\bar{i}_c(t), c\in\mathcal{C}\}$

for all $i\in\mathcal{N}_D$, initialize $r_i(0) = \gamma_i(0) = 0$
for time slot $t \ge 0$ **do**
  obtain $\mathbf{z}(t) = \{z_{i,c}(t), c\in\mathcal{C}, i\in\mathcal{N}_D\}$ — the instantaneous channel capacity vector
  $\bar{r}_i(t) = \max\{l_j \mid l_j \le r_i(t),\ 0\le j\le m\}$, for all $i\in\mathcal{N}_D$
  for each resource block $c\in\mathcal{C}$ **do**
    if $\max_{i\in\mathcal{N}_D}(\bar{r}_i(t) - \gamma_i(t)) \ge 0$ **then**
      $\bar{i}_c(t) = \arg\max_{i\in\mathcal{N}_D}(\bar{r}_i(t) - \gamma_i(t))\cdot z_{i,c}(t)$
    end if
    $i_c^*(t) = \arg\max_{i\in\mathcal{N}_D} {u_i}'(r_i(t))\cdot z_{i,c}(t)$
  end for
  for $i\in\mathcal{N}_D$ **do**
    $r_i(t+1) = (1-\beta)\cdot r_i(t) + \beta\sum_{c\in\mathcal{C}} z_{i,c}(t)\cdot\mathbb{I}_{\{i_c^*(t)=i\}}$
    $\gamma_i(t+1) = (1-\beta)\cdot\gamma_i(t) + \beta\sum_{c\in\mathcal{C}} z_{i,c}(t)\cdot\mathbb{I}_{\{\bar{i}_c(t)=i\}}$
  end for
end for

where $\mathbb{I}_{\{\bar{i}_c(t)=i\}}$ is the indicator function for the event that user i is allocated resource block c at time slot t by the `Enforcer`.

Optimization problems P_1 and P_2 are both solved using the well-known gradient algorithm. For `Shadow` , the utility is the sum of the α-fair utilities of the DASH users, whereas for the `Enforcer`, the utility is the sum square error of the average throughput of the DASH flows, namely $\sum_{i\in\mathcal{N}_D}(\gamma_i(\mathbf{a}) - \bar{r}_i)^2$. Both utilities are concave and differentiable everywhere in the domain. Thus, the result in [19] ensures that for small β, the average throughput of `Shadow` and `Enforcer` converge to the optimal solution of problem P_1 and P_2, respectively. The `Shadow-Enforcer` scheduling policy, which is a cascade of `Shadow` and `Enforcer` is presented as Algorithm 1. In Algo. 1, we handle the condition in (4) by introducing quality level l_0 with a video bitrate of 0. Since `Shadow` and the `Enforcer` work in an open loop setup, their cascade is also expected to exhibit a convergent behavior.

In the basic formulation of our problem, `Enforcer` forces all DASH flows to attain the target rate. However, for some configurations, the scheduler may not achieve the target. In fact, it may happen that the scheduler dictates to a DASH flow a rate higher than the rate requested by the DASH client. As such, there may be spare resource blocks and the system may be under-utilized. In the next section, we will present a solution that circumvents this issue by allocating the spare resource blocks to non-DASH traffic.

V. Resource Allocation for Heterogeneous Traffic

Let us consider the case when DASH and non-DASH traffic coexist, and share the resource blocks. In case of under-utilization of the resource blocks by the `Shadow-Enforcer` scheduler, we need a mechanism in place to allocate these unused resource blocks, opportunistically, to non-DASH users. The modified `Shadow-Enforcer` scheduling

Algorithm 2 Modified `Shadow-Enforcer` scheduling policy for DASH and non-DASH flows

Input: set of DASH flows $\mathcal{N}_D$; set of DASH video bitrates $\{l_0 = 0\} \cup \{l_1, l_2, \ldots, l_m\}$; set of non-DASH flows $\mathcal{N}_E$
Output: User-resource block allocation for each time slot $t \geq 1$, i.e., $\{\bar{i}_c(t), c \in \mathcal{C}\}$

1: let $\mathcal{N} = \mathcal{N}_D \cup \mathcal{N}_E$; initialize $r_i(0) = \gamma_i(0) = 0 \, \forall i \in \mathcal{N}$
2: **for** time slot $t \geq 0$ **do**
3: obtain $\mathbf{z}(t) = \{z_{i,c}(t), c \in \mathcal{C}, i \in \mathcal{N}\}$ — the instantaneous channel capacity vector
4: **for** resource block $c \in \mathcal{C}$ **do**
5: $i_c^*(t) = \arg\max_{i \in \mathcal{N}} {u_i}'(r_i(t)) \cdot z_{i,c}(t)$
6: **if** $i_c^*(t) \in \mathcal{N}_E$ **then**
7: $\bar{i}_c(t) = i_c^*(t)$
8: **else**
9: $\bar{r}_i(t) = \max\{l_j \mid l_j \leq r_i(t), \ 0 \leq j \leq m\}$, for all $i \in \mathcal{N}_D$
10: **if** $\max_{i \in \mathcal{N}_D} (\bar{r}_i(t) - \gamma_i(t)) \geq 0$ **then**
11: $\bar{i}_c(t) = \arg\max_{i \in \mathcal{N}_D} (\bar{r}_i(t) - \gamma_i(t)) \cdot z_{i,c}(t)$
12: **else**
13: $\bar{i}_c(t) = \arg\max_{i \in \mathcal{N}_E} {u_i}'(r_i(t)) \cdot z_{i,c}(t)$
14: **end if**
15: **end if**
16: **end for**
17: **for** $i \in \mathcal{N}$ **do**
18: $r_i(t+1) = (1-\beta) \cdot r_i(t) + \beta \sum_{c \in \mathcal{C}} z_{i,c}(t) \cdot \mathbb{I}_{\{i_c^*(t) = i\}}$
19: $\gamma_i(t+1) = (1-\beta) \cdot \gamma_i(t) + \beta \sum_{c \in \mathcal{C}} z_{i,c}(t) \cdot \mathbb{I}_{\{\bar{i}_c(t) = i\}}$
20: **end for**
21: **end for**

policy for DASH and non-DASH traffic is presented as Algorithm 2.

In addition to the set of DASH flows $\mathcal{N}_D$ and DASH video bitrates, Algo. 2 also receives as input the set of non-DASH flows $\mathcal{N}_E$. For each resource block $c \in \mathcal{C}$, Algo. 2 first determines which user is to be allocated to resource block c as $i_c^*(t) = \arg\max_{i \in \mathcal{N}} {u_i}'(r_i(t)) \cdot z_{i,c}(t)$, where $z_{i,c}(t)$ is the instantaneous capacity of user i on resource block c in time-slot t. We note that the maximum is taken over the set of all users $\mathcal{N} = \mathcal{N}_D \cup \mathcal{N}_E$. If $i_c^*(t) \in \mathcal{N}_E$, no further action is required, and resource block c is allocated to user $i_c^*(t)$ (Algo. 2 line no. 7). On the other hand, if $i_c^*(t) \in \mathcal{N}_D$, then we first compute the target rate $\bar{r}_i(t)$ and enforce it (Algo. 2 line no. 9-11). However, if the average throughput of all the DASH flows is above their target rate, the RB is not allocated to DASH flows. Instead, Algo. 2 reallocates this RB to the non-DASH traffic (Algo. 2 line no. 13), in turn improving the system utilization.

VI. Simulation Results

A. Simulation Setup

Extensive simulations have been conducted to evaluate the performance of the proposed algorithms. For comparison, we have reported the performance of the Proportional Fairness scheduler (PF) [22]. The simulations are performed with a 3GPP-compliant LTE system-level simulator. The basic scenario is a LTE downlink with a single eNB and multiple users, including DASH flows and non-adaptive flows. We implemented DASH video streaming on top of the HTTP protocol as well as the adaptation algorithm in the clients.

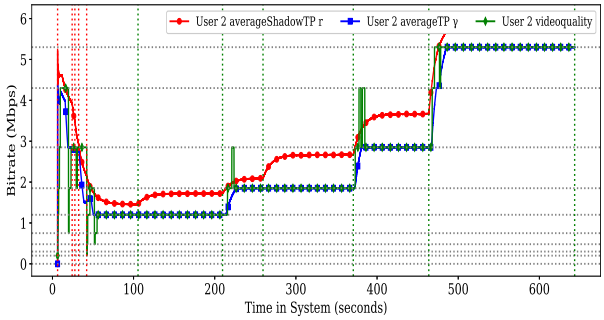


Fig. 2: `Shadow-Enforcer` dynamics for a typical user: average throughout of the `Shadow` scheduler (red curve), average throughput of the `Enforcer` scheduler (blue curve) and resulting video bitrate (green curve).

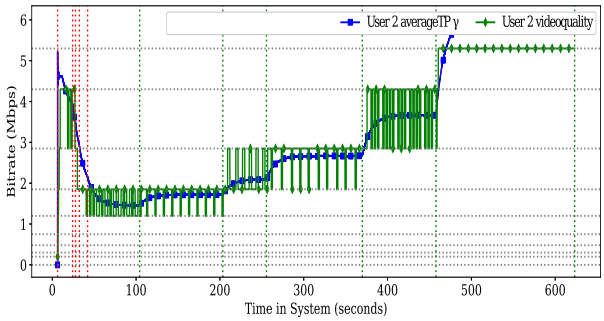


Fig. 3: PF dynamics for a typical user: average throughput of the shadow scheduler (red curve), average throughput of the Enforcer (blue curve) and resulting video bitrate (green curve).

The duration of the DASH flows was exponentially distributed with mean $1/\mu_D = 192$ seconds. The set of available DASH bitrates is $\mathcal{L} = \{0.2, 0.3, 0.48, 0.75, 1.2, 1.85, 2.85, 4.3, 5.3\}$ Mbps[1]. Furthermore, DASH users adapt their video bitrate according to a buffer-based strategy while non-adaptive users download a fixed file size following a general distribution. In all our simulations, we have chosen β as 10^{-4}.

The system operates in a time-slotted fashion with a slot duration of $1\ ms$ and a simulation time horizon of $10^7\ sec$. We assume that users have independent Rayleigh fading channels. We consider two scenarios for the channel state: the *homogeneous scenario*, where users are statistically homogeneous and have three possible channel states, and the *edge scenario*, where users are either close to the eNB, i.e., high rate users, or are placed at the cell edge, i.e., low rate users.

B. Time Scheduling Scenario

In this set of experiments, we assume that users are assigned the whole set of resource blocks, i.e., all DASH flows share same frequency using a time-division multiplexing scheme.

[1]Based on the Media Presentation Description (MPD) file of videos in the YouTube platform

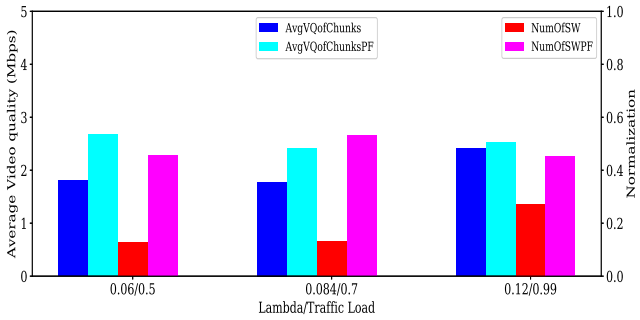
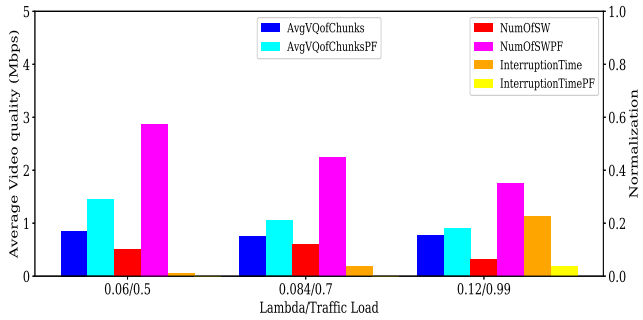
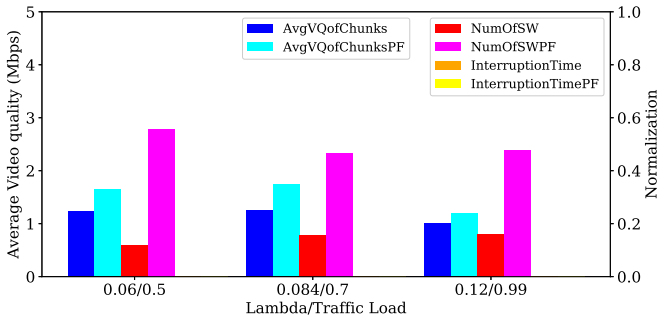


Fig. 4: Average quality (AvgVQofChunks and AvgVQofChunksPF), bitrate switching rate (NumofSW and NumofSWPF) for different values of traffi loads under `Shadow-Enforcer` and PF schedulers.



(a) Basic `Shadow-Enforcer` and PF scheduling with lowest bitrate $l_1 = 0.75$Mbps



(b) `Shadow-Enforcer` with minimum-rate variant and PF scheduling with lowest bitrate $l_1 = 0.75$Mbps

Fig. 5: Average quality (AvgVQofChunks and AvgVQofChunksPF), bitrate switching rate (NumofSW and NumofSWPF) and probability of starvation (InterruptionTime and InterruptionTimePF) for different values of traffic loads under the `Shadow-Enforcer` and the PF scheduler, respectively.

This simple scheme, applied in the homogeneous scenario, lets us isolate and study the response of the scheduler to channel variation. QoE performance is described in terms of switching rate and average streaming quality under both schedulers, namely `Shadow-Enforcer` and PF.

In our first experiment, DASH flows arrive to the network at a rate $\lambda = 0.1\ flows/sec$. In Fig. 2 we have depicted the average throughput of `Shadow` and `Enforcer` schedulers, and the video quality in $Mbit/s$. The dynamics represents the performance experienced by a typical user, i.e., user 2 in our simulation: it is clear that `Shadow-Enforcer` succeeds in maintaining the average throughput close to a specific target rate belonging to the video quality set (blue line in Fig. 2). Multiple available DASH bitrate levels are reported as horizontal gray dotted lines. The red line represents the virtual throughput dynamics under `Shadow`. The target rate for `Enforcer` is obtained by choosing the max bitrate level accordingly, i.e., the closest grey dotted line below the red line. Note how, for any change of the virtual average throughput to the next bitrate level, the real average throughput follows and switches quickly to this target level.

Fig. 2 provides an intuition of how various events influence our scheduler. The vertical red dotted lines mark flow arrivals, while the vertical dotted green lines represent flow terminations. The video bitrate remains unchanged for most of the time, except in proximity of these events. Besides, when some DASH flow terminates, the average throughput of the tagged user, as well as its video quality, increases and quickly stabilizes to the next video bitrate. Under the PF scheduler (see Fig. 3), conversely, we observe frequent fluctuations of the video bitrate. The greedy policy of DASH clients, in fact, tends to set a playout bitrate not sustainable at the available throughput, causing oscillations in the video bitrate.

In Fig. 4, `Shadow-Enforcer` has been tested under increasing load conditions. In particular, we have measured the video quality switching rate as the ratio between the number of switches occurred and the total number of chunks transmitted. We observe that the switching rate is greatly reduced compared to PF, i.e., by $56 - 77.5\%$ overall. However, the average video quality only decreases by $3 - 10\%$. It is worth noting that, during this simulation, no stall occurred for any DASH user because the average rate of the shadow scheduler has always been higher than the lowest bitrate, i.e., $l_1 = 0.2$ Mbps.

Next, we have evaluated the stall time of `Shadow-Enforcer`. To do this, we raised the minimum bitrate to a relatively large value, i.e., $l_1 = 0.75$ Mbps. As shown in Fig. 5a, the stall time (orange) is higher compared to the PF scheduler (yellow). In fact, the basic `Shadow-Enforcer` chooses the highest bitrate below the reference throughput of the `Shadow` scheduler. Thus, it sets to 0 the target rate whenever the average throughput falls below the lowest bitrate level. Overcoming this issue is simple: it is sufficient to set the average throughput of the `Shadow` scheduler as the target for `Enforcer` whenever it falls below the minimum bitrate l_1 (this variant is not reported in the pseudocode for ease of presentation). Fig. 5b repeats the test with this minimum-rate variant, and we note that the difference in stall time due to the two schedulers is almost indistinguishable.

To study the trade-off between efficiency and fairness, we use two metrics which are critical for network performance when multiple DASH users share the channel with other types of traffic. These metrics are important to evaluate the performance improvement that can be attained by any cellular resource allocation solution.

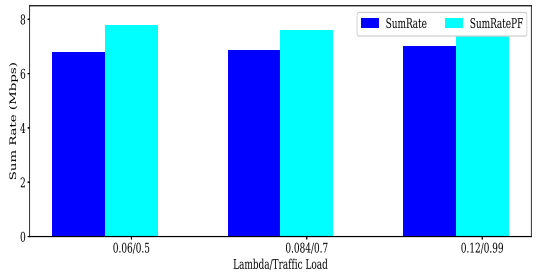


Fig. 6: Total throughput of `Shadow-Enforcer` and PF schedulers for different traffic loads.

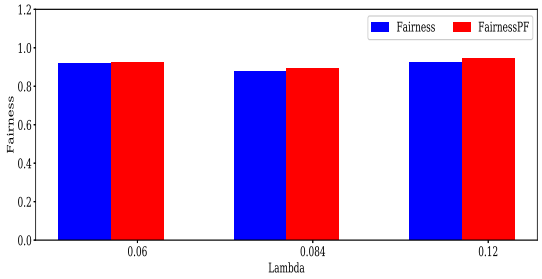


Fig. 7: Jain's index for the `Shadow-Enforcer` and PF schedulers for various traffic loads.

1) *Sum-Rate Efficiency:* it is measured by the sum of the rates delivered to the users of the network.
2) *Fairness:* we use Jain's fairness index [17] to measure whether users are receiving a fair share of system resources. Let $J_i = \frac{r_i}{R_i}$ where R_i is the average throughput of user i if it was allocated the full transmission (depends on its channel state statistic), and r_i is the average rate allocated to user i (depends on the scheduling policy). The Jain's fairness index is defined by

$$\mathcal{J}(\vec{J}) = \frac{(\sum_{i=1}^{n} J_i)^2}{n \sum_{i=1}^{n} J_i^2} \tag{6}$$

where n is the number of active users in the system.

Fig. 6 presents the total average throughput for different values of λ. By design, the total throughput under `Shadow-Enforcer` scheduler is smaller that the one under PF. We will show in the next section that a modified version of `Shadow-Enforcer` outperforms PF in terms of channel utilization by allocating unused resource blocks to non-adaptive flows.

In order to compute the degree of fairness attained by our scheme, we have considered the edge scenario by adding a set of DASH edge users, having different arrival rate and channel statistics. The Jain's fairness index under `Shadow-Enforcer` and PF is depicted in Fig 7, indicating that the `Shadow-Enforcer` scheduler provides fair resource allocation in presence of heterogeneous channel conditions across users.

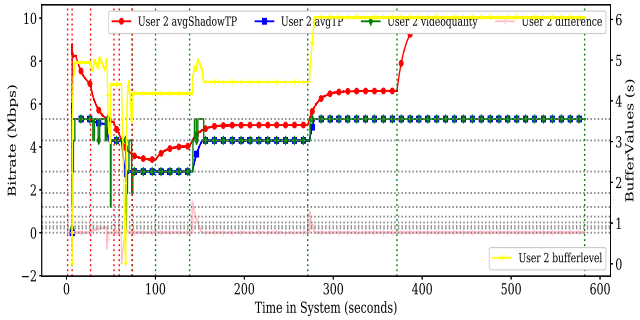


Fig. 8: User 2 under `Shadow-Enforcer` scheduling; the vertical dotted red (green resp.) lines denote DASH flow arrivals (departures resp.); avgShadowTP — average throughout of the shadow scheduler; avgTP — average throughput of the user; 25 RBs and $\lambda = 0.06$

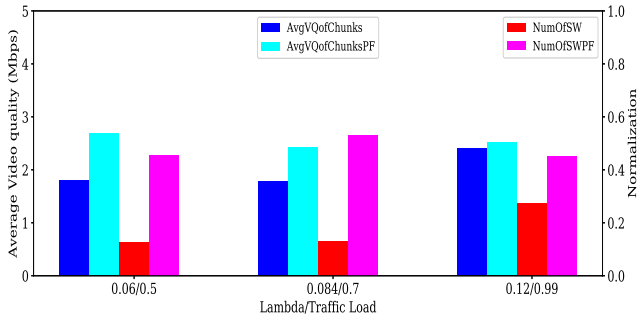


Fig. 9: Average video quality and switching rate under `Shadow-Enforcer` and PF scheduler; $\lambda = 0.1$.

C. Time-Frequency Scheduling Scenario

Next, we assume that DASH users can be simultaneously allocated across multiple resource blocks, to exploit the capabilities of LTE to multiplex flows simultaneously in time and frequency. We assume that 25 RBs are available for DASH flows. Fig. 8 shows the results for a tagged DASH client; the user is the one requesting video with the longest duration. Despite this user experiencing several channel state variations, we observe that our scheme successfully prevents unnecessary video quality fluctuations by stabilizing the video bitrate (yellow line).

We also compare the average quality and average switching rate. The minimum rate l_1 is $1.2\,Mbps$ for this experiment. From Fig. 9, we observe that our scheduler outperforms PF by reducing the switching rate by 86.6%. However, this major improvement is attained with a moderate 8% reduction of the average quality.

Finally, the efficiency and fairness of the scheme are presented in Fig. 10 and Fig. 11 under multiple RBs allocation. As expected, Fig. 10 confirms that `Shadow-Enforcer` attains better channel utilization, since RBs unused by DASH flows are utilized to serve non-DASH ones. From Fig. 10, we can see that even in this scenario, the fairness achieved by our scheme is very close to that of the PF scheduler.

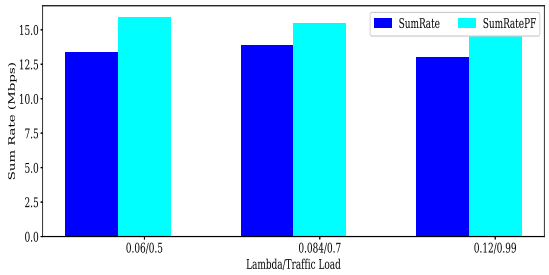


Fig. 10: Total average throughput under `Shadow-Enforcer` and PF schedulers, and 25 RBs.

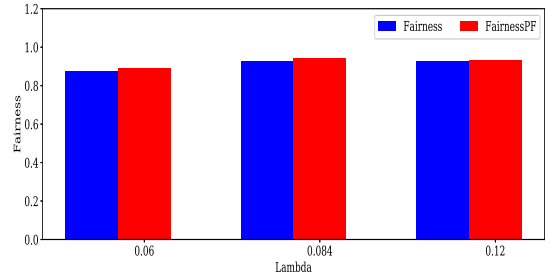


Fig. 11: Jain's index for the `Shadow-Enforcer` and PF schedulers; 25 RBs and 20 DASH users, in the edge scenario.

VII. Discussions and Conclusion

In recent years, numerous researchers have designed schedulers for video streaming using cross-layer approaches. These schemes require coordination among the content provider, the client and cellular networks. However, due to various practical reasons such as scalability and the need for scheduler customization, such level of coordination is often infeasible. Motivated by this practical issue, we proposed `Shadow-Enforcer`, a scheduler capable of ensuring bitrate stability for DASH flows without modifying legacy video delivery mechanism and other network elements.

`Shadow-Enforcer` can also be used within the radio-access component of the upcoming 5G network, in which the slicing concept allows for flexible and dynamic service of diverse traffic types. Furthermore, by feeding information about the throughput of users back into the *Radio-Access Network (RAN)* multi-tenant cell slicing controller, we can ensure that the portion of slice unused by DASH flows can be redistributed to other slices, in turn, ensuring better utilization of the radio resources. Such a joint allocation has to be performed vertically (a PHY-MAC cross-layer approach) as well as horizontally through the RAN controller, in a dynamic manner.

As discussed in the numerical section, the performance obtained by our scheduler depends on the set of peak video bitrates available for the video streaming service. A natural question that arises is: how does the set of peak video bitrates impact QoE metrics, fairness and efficiency? It would also be interesting to investigate the sensitivity of `Shadow-Enforcer` to various parameters such as set of peak video bitrates, number of users sharing network resources, and different video bitrate adaptation methods.

References

[1] Cisco Systems, "Global mobile data traffic forecast update, 2016-2021," *White Paper*, 2017.

[2] C. Ge, N. Wang, G. Foster, and M. Wilson, "Toward QoE-assured 4K video-on-demand delivery through mobile edge virtualization with adaptive prefetching," *IEEE Trans. Multimedia*, vol. 19, no. 10, pp. 2222–2237, Oct 2017.

[3] K. T. Bagci, K. E. Sahin, and A. M. Tekalp, "Compete or collaborate: Architectures for collaborative DASH video over future networks," *IEEE Trans. Multimedia*, vol. 19, no. 10, pp. 2152–2165, Oct 2017.

[4] A. Bentaleb, A. C. Begen, R. Zimmermann, and S. Harous, "SDNHAS: An SDN-enabled architecture to optimize QoE in HTTP adaptive streaming," *IEEE Trans. Multimedia*, vol. 19, no. 10, pp. 2136–2151, Oct 2017.

[5] J. Samain, G. Carofiglio, L. Muscariello, M. Papalini, M. Sardara, M. Tortelli, and D. Rossi, "Dynamic adaptive video streaming: Towards a systematic comparison of ICN and TCP/IP," *IEEE Trans. Multimedia*, vol. 19, no. 10, pp. 2166–2181, Oct 2017.

[6] A. Galanopoulos and A. Argyriou, "Adaptive video streaming over LTE unlicensed," *CoRR*, vol. abs/1608.00239, 2016. [Online]. Available: http://arxiv.org/abs/1608.00239

[7] Y. Im, J. Han, J. H. Lee, Y. Kwon, C. Joe-Wong, T. Kwon, and S. Ha, "FLARE: Coordinated rate adaptation for HTTP adaptive streaming in cellular networks," in *IEEE Int. Conf. Distributed Computing Systems (ICDCS)*, June 2017, pp. 298–307.

[8] J. Chen, R. Mahindra, M. A. Khojastepour, S. Rangarajan, and M. Chiang, "A scheduling framework for adaptive video delivery over cellular networks," in *Proc. 19th Annu. Int. Conf. Mobile Computing and Networking*, ser. MobiCom '13. New York, NY, USA: ACM, 2013, pp. 389–400.

[9] W. Pan, G. Cheng, H. Wu, and Y. Tang, "Towards QoE assessment of encrypted youtube adaptive video streaming in mobile networks," in *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, June 2016, pp. 1–6.

[10] A. Sunny, R. El-Azouzi, E. Altman, S. Poojary, S. Valentin, and D. Tsilimantos, "D-VIEWS: Enforcing bitrate-stability for adaptive streaming traffic in cellular networks," in *Technical report, University of Avignon*, 2018.

[11] T. Lan, D. Kao, M. Chiang, and A. Sabharwal, "An axiomatic theory of fairness in network resource allocation," in *Proc. IEEE INFOCOM*, March 2010, pp. 1–9.

[12] Move Networks, 2010. [Online]. Available: http://www.movenetworkshd

[13] D. F. Brueck and M. B. Hurst, "Apparatus, system, and method for multibitrate content streaming," in *US Patent 7,818,444.*, 2010.

[14] R. Pantos and W. May, "HTTP live streaming," *IETF, Informational Internet-Draft 2582,*, Sep 2011.

[15] Adobe Systems Inc., "HTTP dynamic streaming," 2013. [Online]. Available: http://www.adobe.com/products/hds-dynamic-streaming.html.

[16] ISO/IEC, "Dynamic adaptive streaming over HTTP (DASH)," 2012.

[17] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer system," *DEC Research Report TR-301*, Sep 1984.

[18] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw*, vol. 8, no. 5, pp. 556–567, Oct 2000.

[19] A. L. Stolyar, "On the asymptotic optimality of the gradient scheduling algorithm for multiuser throughput allocation," *Operations Research*, vol. 53, no. 1, pp. 12–25, Jan. 2005.

[20] G. Song and Y. Li, "Cross-layer optimization for OFDM wireless networks-part I: theoretical framework," *IEEE Trans. Wireless Commun.*, vol. 4, no. 2, pp. 614–624, March 2005.

[21] G. Song and Y. Li, "Cross-layer optimization for OFDM wireless networks-part II: algorithm development," *IEEE Trans. Wireless Commun.*, vol. 4, no. 2, pp. 625–634, March 2005.

[22] R. Kwan, C. Leung, and J. Zhang, "Proportional fair multiuser scheduling in lte," *IEEE Signal Processing Letters*, vol. 16, no. 6, pp. 461–464, June 2009.