# Enforcing Bitrate-Stability for Adaptive Streaming Traffic in Cellular Networks

Albert Sunny, Rachid El-Azouzi, Afaf Arfaoui, Eitan Altman, Sudheer Poojary, Dimitrios Tsilimantos, Stefan Valentin

▶ **To cite this version:**

Albert Sunny, Rachid El-Azouzi, Afaf Arfaoui, Eitan Altman, Sudheer Poojary, et al.. Enforcing Bitrate-Stability for Adaptive Streaming Traffic in Cellular Networks. IEEE Transactions on Network and Service Management, IEEE, 2019, 16 (4), pp.1812-1825. 10.1109/TNSM.2019.2941450 . hal-02418530

**HAL Id: hal-02418530**
**https://hal.inria.fr/hal-02418530**

Submitted on 21 Dec 2019

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Enforcing Bitrate-Stability for Adaptive Streaming Traffic in Cellular Networks

Albert Sunny*, Rachid El-Azouzi†, Afaf Arfaoui†, Eitan Altman‡, Sudheer Poojary‖, Dimitrios Tsilimantos¶ Stefan Valentin§

***Abstract*—Video streaming over cellular network has become extremely popular in 4G and will be an integral part of future cellular networks. While most modern-day video clients continually adapt quality of the video stream, they neither coordinate with the network elements nor among each other. Consequently, a streaming client may quickly overload the cellular network, leading to poor Quality of Experience (QoE) for the users in the network. Motivated by this problem, we present D-VIEWS — a scheduling paradigm that assures video bitrate stability of adaptive video streams while ensuring better system utilization. D-VIEWS only needs to be aware of the set of video bitrates and requires no changes to the streaming client and other network functions. We also study, through simulations, the performance of proportional fairness scheduler and D-VIEWS in the presence of user arrival and departure events.**

***Index Terms*—Adaptive Video Streaming, DASH, Scheduling, Rate Control**

## I. Introduction

VIDEO traffic (e.g. TV video, video streaming, live video services) is expected to represent 90% of the Internet traffic by 2021 [1]. While significant progress has been made towards increasing the capacity of cellular networks, in recent years, users' *Quality of Experience (QoE)* has become a challenging and prominent issue. Researchers are exploring *HTTP Adaptive Streaming (HAS)* enabled architectures [2–6] as a means to balance network performance and QoE. *Dynamic Adaptive Streaming over HTTP (DASH)* has become a very popular HAS standard in recent years.

With DASH, each video is divided into multiple segments, and each segment is encoded into multiple bitrates/resolutions. Based on available capacity, DASH clients dynamically choose quality level on a segment-by-segment basis such that the visual quality is maximized and playout buffer under runs are minimized. Various proprietary pre-DASH technologies such as Microsoft's smooth streaming, Adobe's HTTP dynamic streaming, and Apple's live streaming follow nearly the same principles as DASH. Over time, DASH [7–9] has become the de-facto HAS solution of choice for content providers because it has a good balance between buffering delays and visual quality. This is achieved by employing an adaptation engine which chooses an appropriate bitrate for each segment taking into consideration network conditions such as estimated throughput and playout buffer level.

The major drawback of HAS protocols such as DASH is the potential instability in multi-user wireless networks under user dynamics [10]. Here, instability refers to persistent oscillations in the video bitrate of an adaptive stream. The root cause of this instability is the myopic reaction of DASH sources to bandwidth variations in multi-user wireless networks. From a network operator's point of view, this motivates us to design fair schedulers able to arbitrate multiple class of flows while ensuring stability for HAS flows.

### A. Our contributions

In this paper, we present D-VIEWS — a scheduling scheme that addresses the major challenges of allocating radio resources to multiple adaptive video streams in multi-user wireless networks. In particular, we present a low-complexity solution that

- enforces bitrate stability for DASH stream while ensuring good network resource utilization.
- assigns available resources fairly across multiple users.

D-VIEWS allocates radio resources to DASH users based on their channel state information and the set of predefined video bitrates at which DASH segments are encoded. The set of the bitrates could possibly be retrieved from a bitrate database that maps the source IP addresses of adaptive video streams to set of DASH video bitrates. D-VIEWS does not require (i) standardization of cross-layer interfaces; (ii) direct access to the application layer, which may violate user privacy; (iii) modification of existing base station schedulers, facilitating quicker deployments.

The remainder of this paper is organized as follows. Related works are presented in Section I-B. In Section II, we present a few challenges in allocating resources to DASH users. Section III presents in detail our design principles behind D-VIEWS. A method to compute the target rates used by D-VIEWS is presented in Section IV-A. Multiple resource block allocation is discussed in Sections V. In Section VI, we present evaluation details of D-VIEWS and the well-known proportional fairness scheduler. Finally, in Section VII, we present some discussions and conclude the paper.

### B. Related work

Current cellular networks incorporate radio resource management techniques that are designed to meet QoE require-

*Indian Institute of Technology, Palakkad, India Email: albert@iitpkd.ac.in
‡University of Cote d'Azur, INRIA Sophia Antipolis, France Email: eitan.altman@inria.fr
†LIA, University of Avignon, France. Email: rachid.elazouzi@univ-avignon.fr, afaf.arfaoui@alumni.univ-avignon.fr
‖Qualcomm India Pvt. Ltd., Bangalore, Karnataka Email: sudheer.poojary@gmail.com
¶Paris Research Center, Huawei Technologies, France Email: dimitrios.tsilimantos@huawei.com
§Darmstadt University, Darmstadt, Hessen, Germany Email: stefan.valentin@h-da.de

ments of traditional single-rate video streams and other HTTP-based traffic [11, 12]. Several studies in the literature have identified number of stalls, start-up delay and fraction of total playback time spent in buffering as some critical metrics that affect QoE of video traffic [13–16]. In addition to this, adaptive video streams also have QoE criteria such as number of video bitrate switches and average video quality. In fact, through extensive LTE simulations and experiments on a WiMAX base station prototype, the authors of [10] have established that adaptive video streams in wireless networks often suffer from performance issues such as bitrate unfairness among competing flows, instability due to frequent video bitrate switching and inefficient link utilization. Subsequently, several solutions have been proposed to mitigate these issues.

In [17], the authors present SAP — a DASH video traffic management solution that reduces playout stalls and seeks to maintain a consistent QoE for cellular users. SAP achieves this by leveraging both network and client state information to optimize the visual quality of individual video flows. Since SAP uses average video rate to predict stalls, it does not take into consideration number of switches between different video bitrates. If a video stream experiences frequent quality switches, QoE maybe low, even if its average video bitrate is high. The authors of [18] have established that variable temporal quality is indeed worse than maintaining a constant quality that is lower on the average. Also, it is not evident how SAP can be enhanced to perform traffic management when non-DASH flows are co-existent with DASH flows. By non-DASH flows, we mean different types of HTTP traffic such as long-lived file transfers, short-lived web traffic, delay intolerant traffic such as SSH, and traditional single-rate video streams.

MANE, proposed in [19], achieves fair playout buffer levels among HAS clients competing for same wireless resources by allocating radio resources according to video content characteristics, playout buffer levels and channel conditions. However, as in SAP, MANE also does not provide a means to control number of quality switches in adaptive video streams. The system architecture proposed in [19] requires MANE to be located close to *eNodeBs (eNBs)*, and requires *Channel State Information (CSI)* updates from eNBs. Based on these updates, it sets up rate values, i.e., *Guaranteed bitrate (GBRs)*, that eNBs will try to guarantee to their associated clients. Such a scheme requires tight co-ordination among HAS servers, video clients and the eNBs. While this may be possible in content distribution networks, it will be difficult to achieve in cellular networks.

Several researchers have also explored cross-layer schemes to improve QoE of adaptive video streams. Cross-layer allocation schemes that factor channel quality, video quality requirements, and encoding rate fluctuations of HAS video streams with the goal of minimizing transmission delays experienced by users were proposed in [20, 21]. The authors in [22] propose a cross-layer scheme to optimize aggregate utility of clients while maintaining stable video quality. AVIS presented in [10] is yet another cross-layer scheme which separates resource management of adaptive and regular video flows. While cross-layer schemes ensure good performance, they require co-ordination among video servers, clients and the eNBs. However, due to practical reasons such as scalability and operator policies, such co-ordination is often infeasible in cellular networks.

## II. System Model and Motivation

### A. DASH video traffic and QoE metrics

As per the MPEG-DASH specification published in early 2012 [9], within an MPEG-DASH server, each video is divided into multiple segments (each containing about 2-10 seconds of video). Each segment is then encoded into multiple bitrates/resolutions. Based on estimated throughput and media playout buffer occupancy, an adaptation engine within the MPEG-DASH client chooses a video bitrate, on a segment-by-segment basis, that ensures good video quality while avoiding playout interruptions.

When one wishes to quantify users' perception of an application or a service, one tries to identify measurable objects that allow one to predict the average score that would be given by users. These objects are known as Key Performance Indicators (KPI). The following are a few popular KPIs in adaptive video streaming:

***1. Average bitrate:*** sum of the video bitrate of the segments downloaded by the player divided by the total number of downloaded segments.

***2. Number of bitrate switching:*** number of times the video quality has changed during its playout.

***3. Buffering ratio:*** fraction of the total session time spent in re-buffering.

***4. Re-buffering rate:*** number of interruptions observed by a user watching a video.

***5. Startup delay:*** duration between initiation of a video session and the start of its playout.

### B. Network model

Long-Term Evolution (LTE) has adopted *Orthogonal Frequency Division Multiplexing (OFDM)* as the signal bearer and the associated access scheme *Orthogonal Frequency Division Multiple Access (OFDMA)* for downlink transmissions. In this scheme, the frequency dimension is divided into sub-carriers, whereas the time dimension is divided into $10\,ms$ radio frames. Each frame 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 known as a Resource Block (RB). In the LTE standard, a RB spans 12 OFDM carriers and 2 time slots, i.e., a total of 180 kHz for a duration of $1\,ms$. We note that each subcarrier can be modulated using schemes such as QPSK, QAM, 64-QAM. In the remainder of this paper, we denote the set of resource blocks available with an eNB as $\mathcal{K}$.

A scheduler is located at each eNB and is responsible for assigning RBs to its associated users every $1\,millisecond$. We will refer to this duration as a slot. A DASH segment, containing several seconds of video, is typically transferred over thousands of slots. The scheduler makes decisions at the end of every slot, whereas DASH clients make decisions

every few thousand slots. Assuming that average throughput[1] converges quickly (in a few hundred slots) under the given scheduling policy, the order-of-magnitude difference in the timescales of the above mentioned decisions creates a two-time scale process. Therefore, we can assume that a DASH segment, over its download duration, experiences the average throughput under the given scheduling policy.

### C. DASH video bitrate adaptation

DASH video bitrate adaptation algorithms aim to deliver the best possible video bitrate while trying to avoid playout buffer underflow. These algorithms can be classified into three main categories: buffer-based [23], throughput-based [24] and buffer–throughput-based algorithms [25]. While the algorithms in the first category make decisions based on playout buffer occupancy state, ones in the second category use historical TCP throughput measurements. The third category borrows and combines techniques used in the first and second categories.

Several studies have shown that throughput estimation is inherently unreliable and inaccurate over the HTTP layer [24, 26]. This will lead to undesirable bitrate switches and low quality for throughput-based algorithms. In [27], the authors investigated performance of a buffer-based solution by formulating it as a stochastic optimization problem with the objective of maximizing the QoE metrics. In comparison to Netflix's default algorithm, their approach was able to reduce the re-buffering rate by 10-20% while delivering similar average video quality. Their approach was also able to outperform alternative solutions such as FESTIVE [26], and the prediction method proposed in [24]. Therefore, we restrict our study to DASH flows with buffer-based video bitrate adaptation algorithms. However, we believe that with slight modifications, our approach can be used to efficiently allocate resources for any HTTP-based adaptive video streaming technology while maintaining good QoE.

As wireless network conditions are unpredictable, download duration of DASH segments form a *stochastic process*. Let $t_k$ denote the time instance when download of the $k^{\text{th}}$ segment of a DASH flow is completed. We assume that video bitrate for the $n^{\text{th}}$ segment is chosen as $l(n) \in \mathcal{L}$, where $\mathcal{L} = \{l_1, l_2, \ldots, l_m\}$ is the set of constant[2] video bitrates at which DASH segments are encoded. In the remainder of this paper, we refer to this set as the "set of DASH bitrates." For ease of presentation, we assume that all DASH flows in the network have the same set of DASH bitrates. However, our approach can be easily extended to cases when different flows use different sets of DASH bitrates.

Regardless of the encoded video bitrate, each segment contains $s$ seconds of video, and the DASH clients change video bitrate on a segment-by-segment basis, i.e., at the end of every segment download. The evolution of the playout buffer of a typical DASH client is given by the following recursive equation

$$b(t_n) = [b(t_{n-1}) + s - s \cdot l(n)/r(n)]^+ \tag{1}$$

where $[x]^+ = \max\{0, x\}$, $b(t_n)$ is the playout buffer level at the time instance $t_n$, $s \cdot l(n)$ represents the size of the $n^{\text{th}}$ segment in bits, and $r(n)$ is the average throughput in the time interval $[t_{n-1}, t_n]$. The choice of $l(n)$ in Equation (1) depends on the video bitrate adaptation algorithm used within the DASH client.

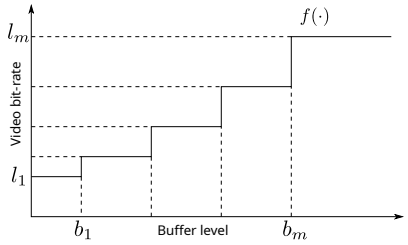



Fig. 1: A mapping of buffer level to discrete video bitrates.

Due to the discrete nature of set $\mathcal{L}$, function $f(\cdot)$ which maps playout buffer levels to video bitrate of the next requested segment takes the form of a step (or staircase) function (refer Fig. 1 for an illustration). Now, if the average throughput $r(n)$ is equal to any of the video bitrates in set $\mathcal{L}$, then we get a range of buffer sizes for which the video bitrate remains unchanged. On the other hand, if $r(n)$ takes values outside set $\mathcal{L}$, then there are no fixed points for Eqn. (1) and the video stream would experience repetitive quality switches.

### D. An initial experiment

DASH uses HTTP, which in turn uses TCP — a reliable transport protocol that ensures end-to-end lossless communication between devices. Thus, from a DASH application's point-of-view, packet losses on a link will manifest itself as a decrease in average throughput. So, a reasonable question is: would the congestion window dynamics of TCP affect the fixed points of Equation. (1)? To understand the impact, if any, we performed experiments on a real-world DASH flow over a wired network.

Our experimental setup consisted of a real-world DASH server and client (both operating on Ubuntu 16.04 LTS machines) connected via a wired link. The server was setup as per the instructions at [7]. For the client, we incorporated the adaptation algorithm presented in Sec. II-C into the MPEG-DASH client available at [8]. Throughput of the wired link was limited to $r$ using the *linux netem* tool. The results of this experiment is summarized in Table I. In the first column of this table, DASH bitrates[3] are shown in bold.

From Table I, we observe that average video bitrate of a DASH stream closely follows the average throughput. DASH achieves this by requesting segments in different video bitrates.

[1]For sake of brevity, we refer to "throughput toward user's end device" as just "throughput."

[2]Variable Bitrate (VBR) encoding allocates higher bitrate to complex segments of a media file and lower bitrates to simple segments. However, adding up the bitrates and dividing by the video duration (in seconds) gives the average bitrate of the media file. If the file size is large, then performance of VBR would be similar to Constant Bitrate (CBR) operating at the average bitrate.

[3]The set of quality levels used in this experiment is actually used by Youtube (refer https://support.google.com/youtube/answer/2853702)

TABLE I: Number of video bitrate switches and average video bitrate of a $3.2\,minute$ (96 segments of duration 2 seconds each) DASH video stream, over a rate-limited wired link, and set of quality levels $\mathcal{L} = \{0.2, 0.3, 0.48, 0.75, 1.2, 1.85, 2.85, 4.3, 5.3\}$ (in Mbps)

| Average throughput $r$ (in Mbps) | Number of video quality switches | Average video bitrate (in Mbps) |
|---|---|---|
| **0.2** | 1 | 0.20 |
| 0.25 | 37 | 0.24 |
| **0.3** | 51 | 0.28 |
| 0.4 | 60 | 0.38 |
| **0.48** | 64 | 0.46 |
| 0.6 | 63 | 0.57 |
| **0.75** | 46 | 0.73 |
| 0.9 | 56 | 0.86 |
| **1.2** | 24 | 1.15 |
| 1.5 | 60 | 1.44 |
| **1.85** | 5 | 1.75 |
| 2.35 | 66 | 2.20 |
| **2.85** | 7 | 2.66 |
| 3.6 | 33 | 4.02 |
| **4.3** | 7 | 3.95 |
| 4.8 | 9 | 4.76 |
| **5.3** | 6 | 4.79 |
| 5.5 | 8 | 4.81 |

Therefore, when average throughput is not equal to any of the DASH bitrates, video bitrate will oscillate between the greatest DASH bitrate below $r$ and lowest DASH bitrate above $r$. This observation is validated by values in the second column of Table I. We note that large number of quality switches happen when $r \in \{0.3, 0.48, 0.75\}\,Mbps$ because the first few video bitrate levels are closely spaced. However, when $r \in \{1.2, 1.85, 2.85, 4.3\}\,Mbps$, we observe significant reduction in number of quality switches.

Evolution of the playout buffer level and video bitrate during a $3.2\,minute$ (96 segments of durations 2 seconds each) video stream when $r = 1.5\,Mbps$ (not a DASH bitrate) and $r = 1.2\,Mbps$ (a DASH bitrate) are shown in Figs. 2 and 3, respectively. From, Fig. 2, we observe that playout buffer evolution when $r = 1.5\,Mbps$ and $r = 1.2\,Mbps$ are similar to each other. However, there is a significant difference in the video bitrate evolution at these rates. When $r = 1.5\,Mbps$ (not a DASH bitrate), DASH requests alternate between $1.2\,Mbps$ and $1.85\,Mbps$. On the other hand, when $r = 1.2\,Mbps$ (a DASH video bitrate), only a few video quality switches occur, and almost all segments are have video bitrate of $1.2\,Mbps$.

The results of this experiment demonstrate that stability of DASH (with respect to video bitrate switches) is indeed governed by the average throughput when *round-trip times (RTTs)* are at most a few milliseconds. Due to similar RTTs experienced by wired and last-hop wireless networks, we believe that the conclusion drawn from this experiment remain valid even for DASH flows over cellular networks.

## III. Scheduler Design

From Section II-D, we know that DASH stream of a user whose average throughput takes values in set $\mathcal{L}$ experiences minimal video quality switches. In this section, we achieve this objective by designing an appropriate scheduling policy.

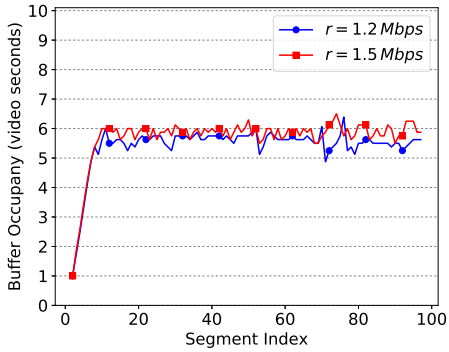
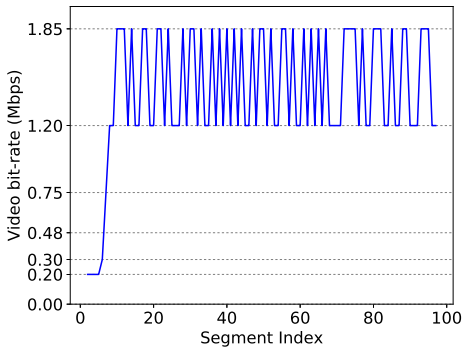
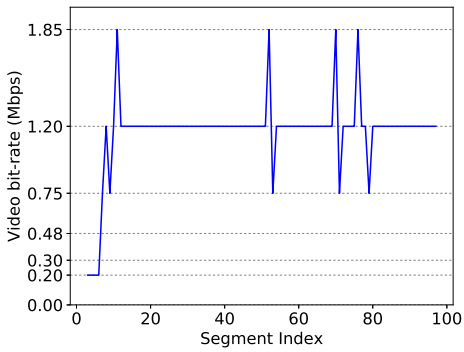



Fig. 2: Evolution of the DASH client's playout buffer level during a $3.2\,minute$ (96 segments of durations 2 seconds each) video.




(a) $r = 1.5\,Mbps$




(b) $r = 1.2\,Mbps$

Fig. 3: Video bitrates requested during a $3.2\,minute$ (96 segments of durations 2 seconds each) DASH video stream.

### A. Design principles

***1. Minimal video bitrate switching:*** The average throughput should take values only in set $\mathcal{L}$.

***2. Quick stabilization of average throughput:*** The scheduler makes decisions at the end of every slot, whereas DASH clients make decisions every few thousand slots. In order to maintain the time scale separation between the two decision processes, the average throughput should stabilize within a few hundred slots.

***3. Practicality and stand-alone independent operation:*** Due to practical reasons such as scalability, scheduler customization and operator policies, co-ordination and co-operation among content providers, clients and eNBs is often infeasible in cellular networks. Therefore, the scheduler should be able

to function with just information about set of video bitrates used by different content providers to encode their adaptive video streams. To facilitate this, network operators can setup a lookup table at schedulers that maps source IP address of adaptive video streams to set of video bitrates. Alternatively, based on the origin (IP address) of the stream, the scheduler can query the set of video bitrates through APIs.

### B. Utility-based scheduling

We begin with a standard utility-based scheduler. The goal is to adapt such a scheduler to handle DASH users with minimal modifications. Let us consider a network of $\mathcal{N} = \{1, 2, \ldots, n\}$ users. With each user $i \in \mathcal{N}$, we associate a utility function which has the following form

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

where $\gamma_i$ is the long-term average throughput of user $i$. $U_i(\gamma_i)$ captures different fairness criteria such as proportional fairness, minimum potential delay fairness and max-min fairness for suitable choice of parameter $\alpha_i$ [28].

Utility-based schedulers are typically used to maximize aggregate utility of the long-term average throughput of users subject to capacity constraints determined by channel statistics. It has been shown in [29–31] that this optimization problem is equivalent to solving the following problem in every time-slot $t$.

$$P_1 : \max \sum\nolimits_{i \in \mathcal{N}} U_i^{'}(\gamma_i(t)) \cdot \Gamma_i(t) \tag{2}$$

where $U_i^{'}(\cdot)$ is the derivative of function $U_i(\cdot)$, $\Gamma_i(t)$ is the instantaneous channel capacity of user $i$ in time-slot $t$, and $\gamma_i(t)$ is the average throughput of user $i$ till time $t$, i.e., $\gamma_i(t) = \frac{1}{t}\sum_{\tau=0}^{t-1} \Gamma_i(\tau) \cdot \mathbf{1}_{\{\text{user } i \text{ is scheduled in slot } \tau\}}$. Here, $\mathbf{1}_{\{\text{user } i \text{ is scheduled in slot } \tau\}} \in \{0, 1\}$ is equal to 1 only if user $i$ is scheduled in slot $\tau$. If we impose the constraint that only one user can be scheduled in any slot, the optimal solution of problem $P_1$ is

$$i^*(t) = \arg\max\nolimits_{i \in \mathcal{N}} U_i^{'}(\gamma_i(t)) \cdot \Gamma_i(t) \tag{3}$$

We note that $i^* : \mathbb{N} \rightarrow \mathcal{N}$ maps every slot to a user in the network. Therefore, we denote it as a function of $t$. The restriction that only one user can be scheduled in any slot will be relaxed in Sec. V.

***Example** 3.1:* Consider a network with 2 users who have independent and identically distributed (i.i.d) *ON-OFF* channels with *ON* capacity $10\,Mbps$ and *ON* probability $0.5$. Let $\alpha_i = 1$ (proportional fairness), and DASH video bitrates $\mathcal{L} = \{0.2, 0.3, 0.48, 0.75, 1.2, 1.85, 2.85, 4.3, 5.3\}$ (in Mbps). From the *Bhatia-Davis inequality*, we know that if a random variable $X \in [0, b]$, then $Var(X) \leq b^2/4$. Now, if $X$ is a Bernoulli random variable such that $P\{X = b\} = P\{X = 0\} = 1/2$, we have $Var(X) = b^2/4$, i.e., *ON-OFF* channels with *ON* probability $0.5$ achieve the largest possible variance among all channels with bounded capacity.

When both users have their respective channel in *ON* state, they are given equal fraction of air-time ($0.5$ each); this happens with probability $0.25$. When only one user has its channel in *ON* state, the user with *ON* channel is given full air-time; once again this happens with probability $0.25$. Therefore, the expected throughput of each user is $10 \times (0.5 \times 0.25 + 1 \times 0.25) = 3.75\,Mbps$. Since average throughput achieved by both users is not equal to any of the video bitrates in set $\mathcal{L}$, these users will experience video playout that frequently switches between video bitrates $2.85\,Mbps$ and $4.3\,Mbps$ — an undesirable user experience.

### C. VIEWS: Virtual Penalty Weighted Scheduling

We recall that our primary objective is to assign resources to DASH flows while avoiding video bitrate oscillations. To that end, in this section, we propose VIEWS — a scheduler capable of guiding average throughput of a DASH user to a desired target rate. We achieve this by modifying the utility-based scheduling rule as follows

$$i^*(t) = \arg \max_{1 \leq i \leq n} U_i^{'}(\gamma_i(t)) \cdot \Gamma_i(t) \cdot \phi_i(r_i, \gamma_i(t)) \tag{4}$$

where $\phi_i(r_i, \gamma_i(t)) = \exp(\frac{-\beta(\gamma_i(t) - r_i)}{r_i})$, $\beta$ denotes the growth rate of *penalty function* $\phi_i(\cdot, \cdot)$, and $r_i$ is the target rate for user $i$. We call the scheduler given by Equation (4) as the *Penalty Weighted (PW)* scheduler. For large values of $\beta$, $\phi_i(r_i, \gamma_i(t))$ is a large positive value when $r_i > \gamma_i(t)$, i.e., users with average throughput less than their respective target rate are imposed a high penalty, in turn increasing their chances of being scheduled. On the other hand, we have $\lim_{\gamma_i(t) \rightarrow \infty} \phi_i(r_i, \gamma_i(t)) = 0$, i.e., for large values of $\beta$, users with average throughput greater than their respective target rate are assigned a penalty very close to zero. Thus, these users are less likely to be scheduled, in turn bringing their average throughput closer to their respective target rate. If $\beta = 0$, then $\phi_i(r_i, \gamma_i(t)) = 1$ and the *PW* scheduler becomes the *utility-based* scheduler. We will use this property to propose a hybrid scheduler in Section IV-B.

Now, consider a network with a single DASH user who has an *ON-OFF* channel with $ON$ capacity $10\,Mbps$, and *ON* probability $0.5$. The *PW* scheduler, irrespective of the penalty, always schedules user 1. Consequently, user 1 achieves an average throughput of $0.5 \times 10 = 5\,Mbps$; not a DASH bitrate. To circumvent this problem, we add a *virtual user* (user 2) to the system and slightly tweak the *PW* scheduler as follows

$$i^*(t) = \begin{cases} 1 & U_i^{'}(\gamma_1(t)) \cdot \Gamma_1(t) \cdot \phi_i(r_1, \gamma_1(t)) \\ & \quad \geq (1 - \epsilon) \cdot U_i^{'}(\gamma_1(t)) \cdot \Gamma_1(t) \\ 2 & \text{otherwise} \end{cases} \tag{5}$$

where $\epsilon$ is a *small positive real number*. We term the scheduling scheme given by Equation (5) as ***VI**rtual **PE**nalty **W**eighted **S**cheduling (**VIEWS**)*. To see how *VIEWS* works, consider the following example.

***Example** 3.2:* Let us set $r_1$, i.e., target throughput of user 1, as $2.85\,Mbps$ (one of the DASH bitrates). Now, if $\gamma_1(t) > 2.85 + \frac{\epsilon}{\beta}$, then $\phi_i(r_1, \gamma_1(t)) < 1 - \epsilon$ (we have used the approximation $e^{-\epsilon} \approxeq 1 - \epsilon$ for small values of $\epsilon$), and user 1 is not scheduled, in turn reducing its average throughput. On the other hand, if $\gamma_1(t) \leq 2.85 + \frac{\epsilon}{\beta}$, then $\phi_i(r_1, \gamma_1(t)) \geq 1 - \epsilon$,

**Algorithm 1 VIEWS**: **VI**rtual **P**Enalty **W**eighted **S**cheduling

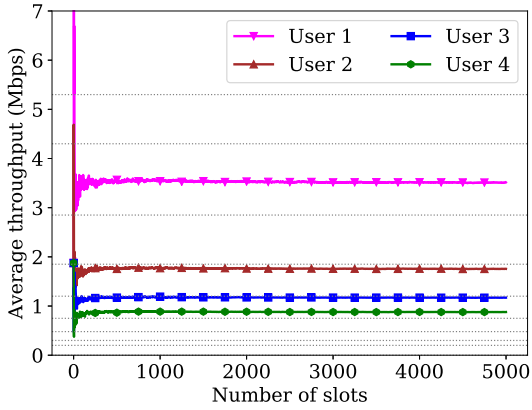
**Input:** set of video bitrates $\mathcal{L}$, target rate vector $\mathbf{r} = [r_1, r_2, \ldots, r_n]$, and $\epsilon > 0$.
**Output:** user allocation for each time slot $t \geq 1$, i.e., $\{i^*(t), t \geq 1\}$

for all $i \in \mathcal{N}$, initialize $\gamma_i(0) = 0$
**for** time slot $t \geq 0$ **do**
  obtain the instantaneous channel capacity vector $\{\Gamma_i(t), i \in \mathcal{N}\}$
  **if** $\max_{1 \leq i \leq n} U_i^{'}(\gamma_i(t)) \cdot \Gamma_i(t) \cdot \phi_i(r_i, \gamma_i(t)) \geq \max_{1 \leq i \leq n}(1-\epsilon) \cdot U_i^{'}(\gamma_i(t)) \cdot \Gamma_i(t)$ **then**
    $i^*(t) = \arg\max_{1 \leq i \leq n} U_i^{'}(\gamma_i(t)) \cdot \Gamma_i(t) \cdot \phi_i(r_i, \gamma_i(t))$
  **else**
    $i^*(t) = n+1$ — the virtual user
  **end if**
  **for** $i \in \mathcal{N}$ **do**
    $\gamma_i(t+1) = (t \cdot \gamma_i(t) + \Gamma_i(t) \cdot \mathbf{1}_{\{i^*(t)=i\}})/(t+1)$
  **end for**
**end for**

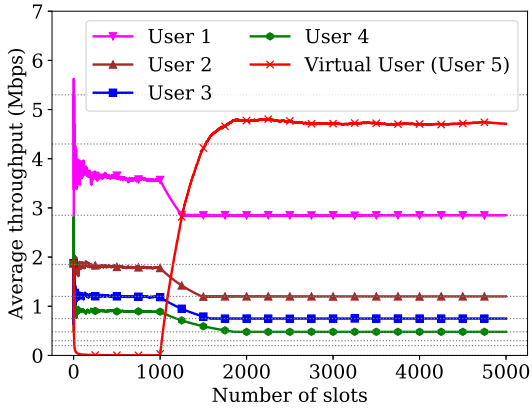
and user 2 is not scheduled allowing user 1 to increase its average throughput.

The inclusion of the virtual user can be interpreted as a means to prevent users whose average throughput is higher than their respective target rate from being scheduled. *VIEWS*, generalized to a $n$ user network, is presented as Algorithm 1. In Algo. 1, if average throughput of every user is equal to their respective target rate, then $\phi_i(r_i, \gamma_i(t)) = 1$, and the virtual user does not come into play. From the decision rule of Algo. 1, it is easy to see that, for any $i \in \mathcal{N}$, if $\gamma_i(t) > r_i$, then $i^*(t) \neq i$, i.e., users whose average throughput is higher than their respective target rate are throttled. Further, if $\gamma_i(t) \leq r_i$ for some $i \in \mathcal{N}$, then $i^*(t) \neq n+1$, i.e., the virtual user always relinquishes its air time to users whose average throughput is below their respective target rate. We note that when there are non-DASH user in the network, there is no need for the virtual user, and resources not allocated to DASH users can be reallocated to non-DASH users (refer Section V).

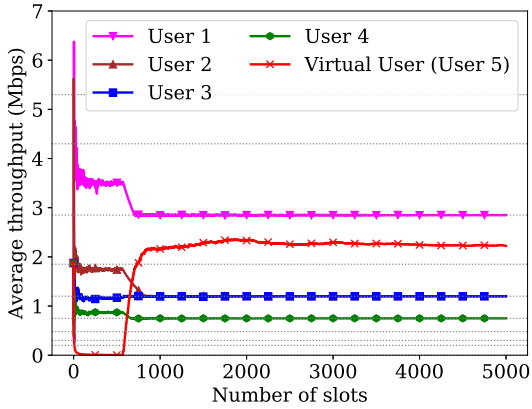
Figs. 4a, 4b and 4c present the evaluation results of the *utility-based scheduler* and *VIEWS* in a network with 4 users who have independent *ON-OFF* channels. User $i$ has *ON* capacity $\frac{C}{i}$ and *ON* probability 0.5. Let $\alpha_i = 1 \, \forall 1 \leq i \leq n$. Then, as expected, the *utility-based scheduler* is able to achieve proportional fairness when there is heterogeneity in the network (refer Fig. 4a). However, average throughput of some users lies outside set $\mathcal{L}$. Consequently, these users would experience incessant video quality switches. In Fig. 4b, the target rate vector is $[2.85, 1.2, 0.75, 0.48]\, Mbps$, i.e., all users are throttled. As a consequence of this, the virtual user gets a large fraction of the air-time. On the other hand, in Fig. 4c, the target rate vector is $[2.85, 1.2, 1.2, 0.48]\, Mbps$, i.e., users 1 and 2 are throttled, whereas average throughput of users 3 and 4 are pulled-up. Since the air-time lost by users 1 and 2 is used up by users 3 and 4, and not by the virtual user, the red line (with the marker $\times$) in Fig. 4c hovers around $2.5\, Mbps$.




(a) *Utility-based scheduler*




(b) *VIEWS*: $r_1 = 2.85\, Mbps$, $r_2 = 1.2\, Mbps$, $r_3 = 0.75\, Mbps$, $r_4 = 0.48\, Mbps$, $\beta = 1000$, $\epsilon = 0.001$.




(c) *VIEWS*: $r_1 = 2.85\, Mbps$, $r_2 = 1.2\, Mbps$, $r_3 = 1.2\, Mbps$, $r_4 = 0.75\, Mbps$, $\beta = 1000$, $\epsilon = 0.001$.

Fig. 4: 4 user network, $\alpha_i = 1 \, \forall 1 \leq i \leq 4$, user $i \in \{1, 2, 3, 4\}$ has an *ON-OFF* channel with *ON* capacity $\frac{15}{i}\, Mbps$, and *ON* probability 0.5.

We would like to note that throughput of the virtual user is merely a proxy for number of scheduler slots unused by the bitrate constrained DASH flows. Therefore, channel statistics of the virtual user is irrelevant for *VIEWS*.

## IV. Obtaining Target Rates

*VIEWS* can guide average throughput of users to a target rate vector. If target rates are too high, it may be infeasible

and average throughput of users may take values outside set $\mathcal{L}$. On the other hand, low target rates may be achievable, but may result in system under-utilization. Thus, choosing the right target rate vector is a challenging problem.

### A. Optimal target rates

In this section, we present a means to obtain an optimal target rate vector $\mathbf{r}$. Let $\mathcal{C}_i$ denote the set of possible instantaneous rates of user $i$'s channel. Then, the joint channel state space is given as $\mathcal{C} = \mathcal{C}_1 \times \mathcal{C}_2 \times \ldots \times \mathcal{C}_n$. The problem of optimal target rate computation can be formally stated as the following *Mixed Integer Nonlinear Program (MINLP)*

$$P_2 : \max_{\{a_i(\mathbf{c})\, i\in\mathcal{N}\, \mathbf{c}\in\mathcal{C}, y_{ij}\, i\in\mathcal{N}\, 1\le j\le m\}} \sum_{i\in\mathcal{N}} U_i\left(\sum_{j=1}^{m} l_j \cdot y_{ij}\right)$$

Subject to:

$$\sum\nolimits_{\mathbf{c}\in\mathcal{C}} \pi(\mathbf{c}) \cdot a_i(\mathbf{c}) \cdot \mathbf{c}[i] = \sum\nolimits_{j=1}^{m} l_j y_{ij} \quad \forall i \in \mathcal{N} \tag{6}$$

$$\sum\nolimits_{i\in\mathcal{N}} a_i(\mathbf{c}) \le 1 \quad \forall \mathbf{c} \in \mathcal{C} \tag{7}$$

$$\sum\nolimits_{j=1}^{m} y_{ij} = 1 \quad \forall i \in \mathcal{N}, \mathbf{c} \in \mathcal{C} \tag{8}$$

$$y_{ij} \in \{0,1\} \quad \forall i \in \mathcal{N}, 1 \le j \le m \tag{9}$$

$$a_i(\mathbf{c}) \ge 0 \quad \forall i \in \mathcal{N}, \mathbf{c} \in \mathcal{C} \tag{10}$$

where $\mathbf{c}[i] \in \mathcal{C}_i$ denotes the $i^{\text{th}}$ element of vector $\mathbf{c}$, $\{l_1, l_2, \ldots, l_m\}$ is the set of $m$ DASH bitrates, $a_i(\mathbf{c})$ is the fraction of resource allocated to user $i$ when joint channel state is $\mathbf{c}$, $\pi(\mathbf{c})$ is the probability of occurrence of joint channel state $\mathbf{c}$, and $y_{ij} \in \{0,1\}$ is an indicator variable that takes the value 1 if any only if user $i$'s DASH segments are always encoded at bitrate $l_j$. Constraints (8) and (9) together force average throughput of each user to take values only within set $\mathcal{L}$. Constraint (6) ensures existence of a schedule that achieves an average throughput of $\sum_{j=1}^{m} l_j y_{ij}$ for user $i$.

Let $\{a_i^*(\mathbf{c})\, i \in \mathcal{N}\, \mathbf{c} \in \mathcal{C}, y_{ij}^*\, i \in \mathcal{N}\, 1 \le j \le m\}$ be an optimal solution of problem $P_2$, then target rate $r_i$ for user $i \in \mathcal{N}$ can be computed as $r_i = \sum_{j=1}^{m} l_j \cdot y_{ij}^*$. It can be easily shown that $P_2$ is a *NP-hard* problem. While heuristic-based algorithms are known to aid in the solution of such problems, applying these algorithms to solve $P_2$ would require knowledge of channel statistics. Due to exponentially growing cardinality (with respect to the number of users) of joint channel state space $\mathcal{C}$, $P_2$ also suffers from state space explosion.

### B. Dynamic rate inference

In this section, rather than computing an optimal target rate vector, we explore the possibility of dynamic throttling of average throughput to sub-optimal target rates. We do this by first observing the evolution of users' average throughput under the *utility-based scheduler* for a sufficiently long duration. From these observations, for each user, we infer the best possible video bitrate that does not cause video quality switches and set it as their target rate. Let

$$I_{ij}(t) = \begin{cases} 1 & \text{if } j = \arg\max\{l_k | l_k \le \gamma_i(t), 1 \le k \le m\} \\ 0 & \text{otherwise} \end{cases}$$

i.e., $I_{ij}(t) \in \{0,1\}$ is a binary valued variable that takes value 1 if $l_j$ is the largest bitrate less than average throughput $\gamma_i(t)$. We note that $\sum_{j=1}^{m} I_{ij}(t) = 1$. Next, for user $i$, we define a random time $T_i$ as follows

$$T_i = \left[\min\left\{t \middle| \max_{1\le j\le m} \sum_{\tau=1}^{t} I_{ij}(\tau) \ge \zeta t,\, t \ge t_{min}\right\}\right]^* \tag{11}$$

where $\zeta \in [0,1]$ and $[\cdot]^* = min\{\cdot, t_{max}\}$.

$T_i \in [t_{min}, t_{min}+1, \ldots, t_{max}-1, t_{max}]$ is the first time when $\max_{1\le j\le m} \frac{1}{t}\sum_{\tau=1}^{t} I_{ij}(t)$ exceeds the threshold $\zeta$. In other words, we are looking for a time $T_i$ when there exists a bitrate which was the largest bitrate less than average throughput for at least $\zeta t$ slots. We note that there is a possibility of this never happening; in which case the value of $T_i$ is set to $t_{max}$. In Equation (11), we ensure that we make a decision only after observing the first $t_{min}$ slots because we would like to discard the transients of the *utility-based scheduler*. We note that $t_{min}$ and $t_{max}$ are the minimum and maximum number of slots we observe before computing the target bitrates and enforcing penalty. The actual values of $t_{min}$ and $t_{max}$ can be chosen by the network operator based on their operational and QoE requirements. Given $T_i$, target rate $r_i$ of user $i$ is chosen as follows

$$r_i = l_{j^*(i)} \quad \text{where} \quad j^*(i) = \arg\max_{1\le j\le m} \frac{1}{T_i}\sum_{\tau=1}^{T_i} I_{ij}(\tau)$$

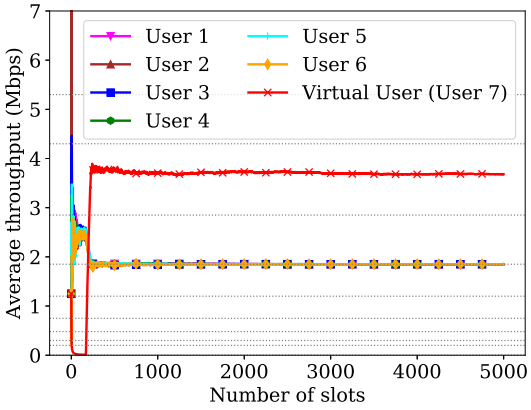
As soon as we have $r_i$, we can switch on penalty function for this user so that average throughput of user $i$ is guided to $r_i$. We achieve this by manipulating growth rate of the penalty function proposed in Section III-C. Let $T = \max_{1\le i\le n} T_i$, i.e., $T$ is the time when all users have obtained their target rates. We propose the following dynamic growth rate for penalty function $\phi_i(\cdot,\cdot)$

$$\beta_i(t) = \begin{cases} 0 & \text{if } t \le T \\ \beta & \text{otherwise} \end{cases}$$

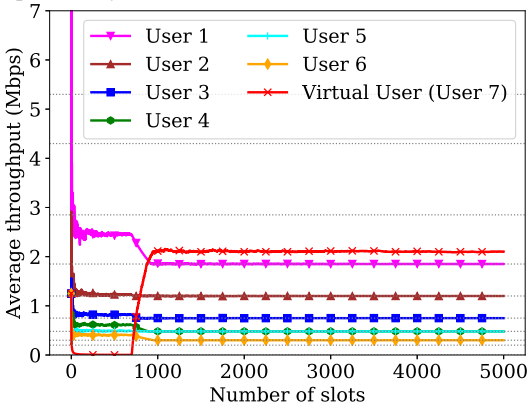
As before, for a system with $n$ users, we add user $n+1$ as a virtual user. Now, for this system, we define the ***D**ynamic **VI**rtual **PE**nalty **W**eighted **S**cheduling (**D-VIEWS**)*. D-VIEWS is identical to *VIEWS* with the exception of penalty function's growth rate. *VIEWS*, uses a time invariant growth rate. Whereas in D-VIEWS, penalty growth rate of user $i$ at time $t$ is $\beta_i(t)$ — a function of $t$.

It is easy to see that when $t \ge T$, D-VIEWS is identical to *VIEWS*. Further, when $t < T$, then $i^*(t) \ne n+1$. We activate the virtual user only after all users have acquired their target rates. If activated earlier, the virtual user will overwhelm other users and end up forcing them to low rates.

Evaluation results of D-VIEWS in a network with 6 homogeneous (independent and identically distributed channels) and 6 heterogeneous (independent but not identically distributed channels) users are presented in Fig. 5a and Fig. 5b, respectively. In the homogeneous case, each user is assigned $1.85\,Mbps$ (the highest bitrate less than $\frac{15}{6}(1-(1-0.5)^5) = 2.50\,Mbps$) as their target rate. At $t = 150$, penalty function is switched on for all users, and average throughput of each user quickly drops to $1.85\,Mbps$. D-VIEWS ensures fairness by

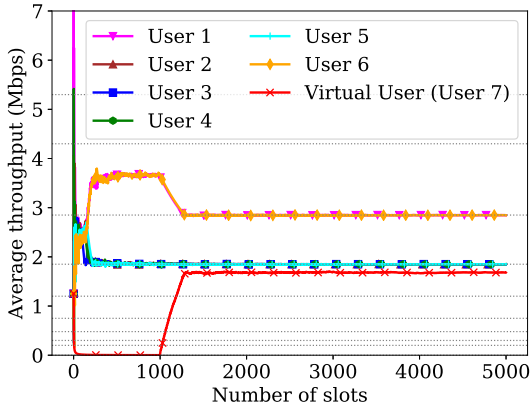



(a) Each user has an *ON-OFF* channel with *ON* capacity 15 $Mbps$, and *ON* probability 0.5.

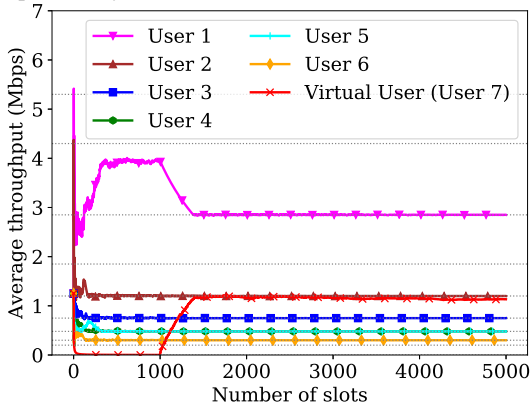



(b) User $i \in \{1, 2, 3, 4, 5, 6\}$ has an *ON-OFF* channel with *ON* capacity $\frac{15}{i}$ $Mbps$, and *ON* probability 0.5.

Fig. 5: D-VIEWS: 6 user network, $\alpha_i = 1 \, \forall 1 \leq i \leq 6$, $t_{min} = 100$, $t_{max} = 1000$, $\zeta = 0.9$, $\beta = 1000$, $\epsilon = 0.001$.




(a) Each user has an *ON-OFF* channel with *ON* capacity 15 $Mbps$, and *ON* probability 0.5.




(b) User $i \in \{1, 2, 3, 4, 5, 6\}$ has an *ON-OFF* channel with *ON* capacity $\frac{15}{i}$ $Mbps$, and *ON* probability 0.5.

Fig. 6: *D-VIEWS v2*: 6 user network, $\alpha_i = 1 \, \forall 1 \leq i \leq 6$, $t_{min} = 100$, $t_{max} = 1000$, $\zeta = 0.9$, $\beta = 1000$, $\epsilon = 0.001$.

forcing users with identical channel statistics to the same target rate. However, each user had to sacrifice about 0.65 $Mbps$ of throughput resulting in an under-utilized system.

In the heterogeneous case, D-VIEWS activates penalty and the virtual user at $t = 750$, and average throughputs converge to the target rate vector $[1.85, 1.2, 0.75, 0.48, 0.48, 0.3]$ $Mbps$ (refer Fig. 5b). The throughput loss of users with lower average rate is less, resulting in a better resource utilization compared to the homogeneous case. We note that user 6 who has an average channel rate of 1.25 $Mbps$ gets assigned a target rate of 0.2 $Mbps$, whereas users 1 who has an average channel rate of 7.5 $Mbps$ has a target of 2.75 $Mbps$; indicating that users with better average channel rates get better target rates.

We recollect that in D-VIEWS, penalty is enabled only after all users have acquired their target rates, i.e., at time $T = \max_{1 \leq i \leq n} T_i$. Alternatively, we could active penalty for user $i$ immediately after its target rate is acquired, i.e., at time $T_i$. Then, penalty function growth rate for user $i$ is given by

$$\beta_i(t) = \begin{cases} 0 & \text{if } t \leq T_i \\ \beta & \text{otherwise} \end{cases}$$

We refer to D-VIEWS with the above policy as *D-VIEWS v2*. We would like to note that even with the above policy, the virtual user is enabled only after all users have acquired their target rates, i.e., at time $T = \max_{1 \leq i \leq n} T_i$. Performance of *D-VIEWS v2* in a network with 6 homogeneous and 6 heterogeneous users is presented in Fig. 6a and Fig. 6b, respectively.

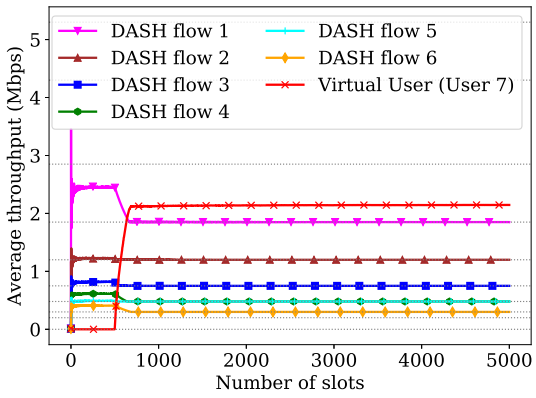
For the homogeneous case, average throughput of users 1 and 6 settles down to 2.85 $Mbps$, whereas average throughput of others settles down to 1.85 $Mbps$ (refer Fig. 6a). This happens because penalty function was enabled individually and not at the same time. In this evaluation run, users $2 - 5$ turn on their penalty before users 1 and 6. After the penalty is turned on, throughput of users $2 - 5$ rapidly decreases to 1.85 $Mbps$. This, in turn, gives more air-time for users 1 and 6; allowing their average throughput to converge to a higher target rate. Due to the individual activation of penalty, we have better resource utilization. However, we lose out on fairness.

In the heterogeneous case, penalty function is activated last for user 1. Thus, most of the air-time lost by users $2 - 6$ is consumed by user 1, allowing its average throughput to reach 2.85 $Mbps$ (refer the curve with marker ▼ in Fig. 6b). Whereas, with simultaneous penalty activation, user 1 had achieved an average throughput of just 1.85 $Mbps$ (refer Fig. 5b).

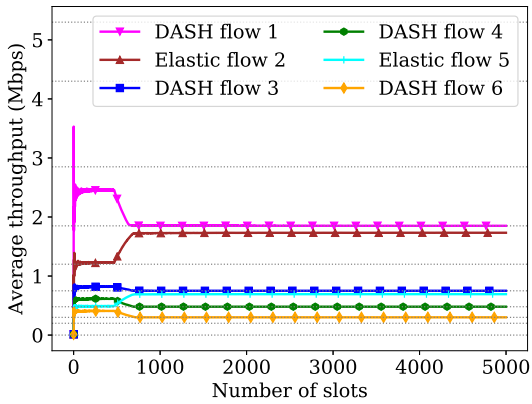
**Algorithm 2** Scheduling multiple resource blocks

**Input:** set of video bitrates $\mathcal{L}$, $\epsilon > 0$, set of DASH users $\{1, \ldots, n\}$, set of non-DASH users $\{n+1, \ldots, u\}$
**Output:** user-resource block allocation for each time slot $t \geq 1$, i.e., $\{i_k^*(t), k \in \mathcal{K}, t \geq 1\}$

1: for all $1 \leq i \leq u$, initialize $\gamma_i(0) = 0$
2: **for** each time slot $t \geq 0$ **do**
3: obtain the instantaneous channel capacity vector $\{\Gamma_i^k(t), k \in \mathcal{K}, 1 \leq i \leq u\}$
4: set $\hat{\gamma}_i = \gamma_i(t) \cdot (1 - \frac{1}{t+1})$ for all $1 \leq i \leq u$
5: **for** $k \in \{1, 2, \ldots, |\mathcal{K}|\}$ **do**
6: **if** $\max_{1 \leq i \leq n} U_i^{'}(\hat{\gamma}_i) \cdot \Gamma_i^k(t) \cdot \phi_i(r_i, \hat{\gamma}_i) \geq \max_{1 \leq i \leq n}(1-\epsilon) \cdot U_i^{'}(\hat{\gamma}_i) \cdot \Gamma_i^k(t)$ **then**
7: $i_k^*(t) = \arg\max_{1 \leq i \leq n} U_i^{'}(\hat{\gamma}_i) \cdot \Gamma_i^k(t) \cdot \phi_i(r_i, \hat{\gamma}_i)$
8: **else**
9: **if** no non-DASH users in the network **then**
10: $i_k^*(t) =$ virtual user indexed as $n+1$
11: **else**
12: **if** $\max_{1 \leq i \leq n} U_i^{'}(\hat{\gamma}_i) \cdot \Gamma_i^k(t) \cdot \phi_i(r_i, \hat{\gamma}_i) < \max_{n+1 \leq i \leq u} U_i^{'}(\hat{\gamma}_i) \cdot \Gamma_i^k(t)$ **then**
13: $i_k^*(t) = \arg\max_{n+1 \leq i \leq u} U_i^{'}(\hat{\gamma}_i) \cdot \Gamma_i^k(t)$
14: **else**
15: $i_k^*(t) = \arg\max_{1 \leq i \leq n} U_i^{'}(\hat{\gamma}_i) \cdot \Gamma_i^k(t) \cdot \phi_i(r_i, \hat{\gamma}_i)$
16: **end if**
17: **end if**
18: **end if**
19: set $\hat{\gamma}_p = \hat{\gamma}_p + \frac{\Gamma_p^k(t)}{t+1}$, where $p = i_k^*(t)$
20: **end for**
21: **for** $1 \leq i \leq u$ **do**
22: $\gamma_i(t+1) = \hat{\gamma}_i$
23: **end for**
24: **end for**




(a) 6 DASH users




(b) 4 DASH users (1, 3, 4, and 6), 2 non-DASH users (2 and 5)

Fig. 7: *Algo. 2*: 6 user network, $\alpha_i = 1 \forall 1 \leq i \leq 6$, $t_{min} = 100$, $t_{max} = 1000$, $\zeta = 0.9$, $\beta = 1000$, $\epsilon = 0.001$ , 100 resource block, in each resource block user $i \in \{1, 2, 3, 4, 5, 6\}$ has an *ON-OFF* channel with *ON* capacity $\frac{0.15}{i}$ $Mbps$, and *ON* probability $0.5$ in each resource block.

## V. Allocating Multiple Resource Blocks to Co-existing DASH and Non-DASH Users

In this section, we consider the problem of allocating a set of $\mathcal{K}$ resource blocks to co-existing DASH and non-DASH users. Let DASH users be indexed as $\{1, 2, \ldots, n\}$ and regular (non-DASH) users be indexed as $\{n+1, n+2, \ldots, u-1, u\}$. Now, each resource block $k \in \mathcal{K}$, can be allocated to any of the $u$ users in set $\mathcal{N}$. Then, user to whom $k^{\text{th}}$ resource block should be allocated in $t^{\text{th}}$ slot is determined using Algorithm 2.

Algo. 2 is an extension of D-VIEWS that allocates multiple resource blocks to co-existing DASH and non-DASH users. While D-VIEWS assigns resources given up by DASH users to the virtual user, Algo. 2 reuses these resources by allocating them to non-DASH users (refer steps $9-13$). Consequently, when there are non-DASH user in the network, no virtual user is added to the network. The user scheduled in each resource block is decided by considering one resource block at a time. However, when computing schedule for a resource block, the schedule computed for the previous resource blocks, in the current time-slot, is also taken into consideration (refer step 2).

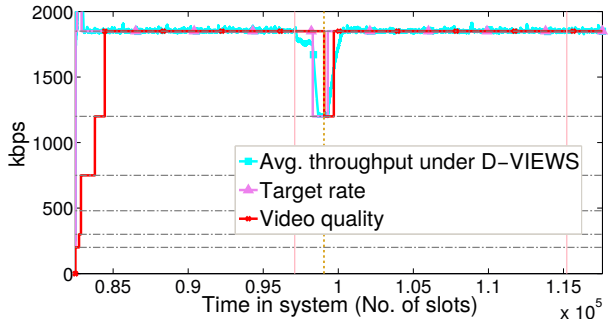
Evaluation results of a scheduler based on Algo. 2 in a heterogeneous network with only DASH users, 100 resource blocks and proportional fairness (all $\alpha_i = 1$) is presented in Fig. 7a. From Fig. 7a, we can see that acquisition and subsequent convergence to target rates are comparable to the single RB case presented in Fig. 5b.

Fig. 7b presents performance of Algo. 2 in a network with co-existing DASH and non-DASH users (users downloading a large file). DASH users are 1, 3, 4 and 6, whereas the non-DASH users are 2 and 5. Algo. 2 reallocates slots unused by DASH users to non-DASH users. Therefore, the virtual user is not present in Fig. 7b. From Fig. 7b, we can see that Algo 2 is able to ensure that average throughput of each DASH users takes values only in set $\mathcal{L}$. Algo. 2 also ensures that air-time given up by DASH users, in order to attain their respective target bitrates, is shared with non-DASH users in the network. This, in turn, improves average throughput of non-DASH users (refer the curve corresponding to users 2 and 5 in Fig. 7b).

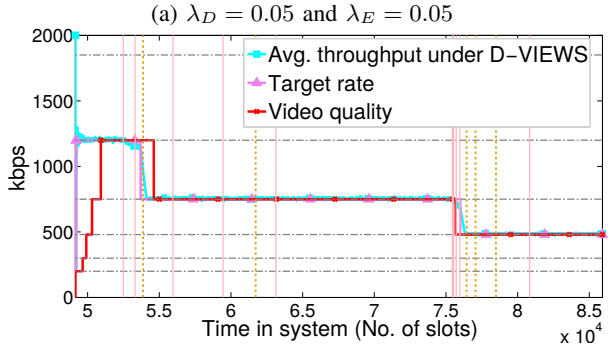
## VI. Simulation Results

### A. Simulation setup

In order to evaluate D-VIEWS under different scenarios, we conduct extensive simulations with Vienna LTE-A sim-

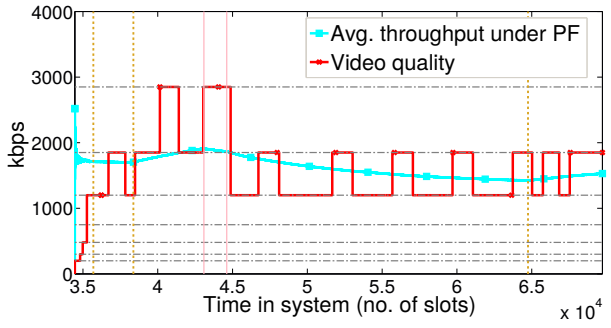



(a) $\lambda_D = 0.05$ and $\lambda_E = 0.05$

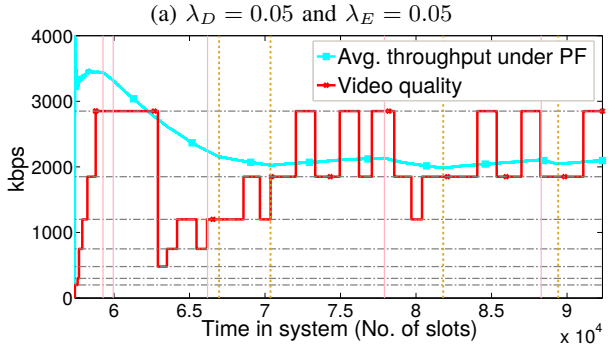



(b) $\lambda_D = 0.13$ and $\lambda_E = 0.05$

Fig. 8: Impact of D-VIEWS on a DASH user in the presence of user arrival and departures events; vertical pink lines and dashed golden lines denote user arrival and departure events, respectively.




(a) $\lambda_D = 0.05$ and $\lambda_E = 0.05$




(b) $\lambda_D = 0.13$ and $\lambda_E = 0.05$

Fig. 9: Impact of PF scheduler on a DASH user in the presence of user arrival and departures events; vertical pink lines and dashed golden lines denote user arrival and departure events, respectively.

ulator — a MATLAB based 3GPP-compliant LTE system-level simulator. For comparison, we also report performance of Proportional Fairness scheduler (PF) [32]. The basic scenario is a LTE downlink with a single base station and multiple DASH (D) and non-DASH (E) users.

We simulate a total of 100 users entering the system according to a Poisson process with rate $\lambda_j$, $j \in \{D, E\}$. Each arriving DASH user requests a video whose duration follows an Exponential distribution with mean $\mu_D = 80\,sec$. The set of DASH bitrates are chosen as $\mathcal{L} = \{0.2, 0.3, 0.48, 0.75, 1.2, 1.85, 2.85, 4.3, 5.3\}\,Mbps$. This set is chosen based on the *Media Presentation Description (MPD)* file of videos on YouTube. Furthermore, DASH users adapted their video bitrate according to a buffer-based strategy, whereas non-DASH users download a file whose size followed an Exponential distribution with mean $\mu_E = 10\,Mbits$. For simulations presented in this section, $\lambda_E$ was chosen $0.05\ users/sec$.

### B. Handling user dynamics and mobility

Till now, we had considered the set of users in the network to be fixed. However, due to mobility of users, this set is often dynamic. To handle such scenarios we reset D-VIEWS every $t_{reset}$ number of slots. Upon reset, average throughput of users is set to zero, growth rates of the penalty functions are set to zero, and the virtual user is disabled. Such a reset mechanism coupled with quick stabilization of average throughput, ensured by the design of D-VIEWS, improves network resource utilization when there are user arrival and departure events. The choice of reset interval $t_{reset}$ depends on operator's requirements. Large values of $t_{reset}$ will reduce number of quality switches experienced by DASH users, whereas smaller values allow the scheduler to react quickly to changes in the system, albeit at a cost of large number of quality switches. For all simulation, $t_{reset}$ was chosen as $2\,sec$ (2000 scheduler slots).

### C. Performance of PF and D-VIEWS

We consider a scenario where six resource blocks (bandwidth of $1.4\,Mbps$) are shared by users in the LTE simulator. We compare the impact of PF and D-VIEWS on different QoE metrics for different DASH user arrival rates.

Figs. 8a and 8b show the impact of D-VIEWS on average throughput and video quality of a DASH flow. It is clear that D-VIEWS successfully throttles average throughput of the user to target rates chosen from set $\mathcal{L}$, ensuring that there are limited video quality switches even when the set of users in the network is dynamic. On the other hand, for PF scheduler (refer Figs. 9a and 9b) we observe frequent fluctuations in video quality. The reason for this is the greedy nature of DASH adaptation algorithm that tends to set a playout bitrate which is not sustainable at the throughput achieved under the PF scheduler.

In Fig. 10, for different DASH user arrival rates, we plot the ratio of number of video bitrate switches to number of downloaded DASH segments. From Fig. 10, we can see that D-VIEWS ensures a much lower switching rate among

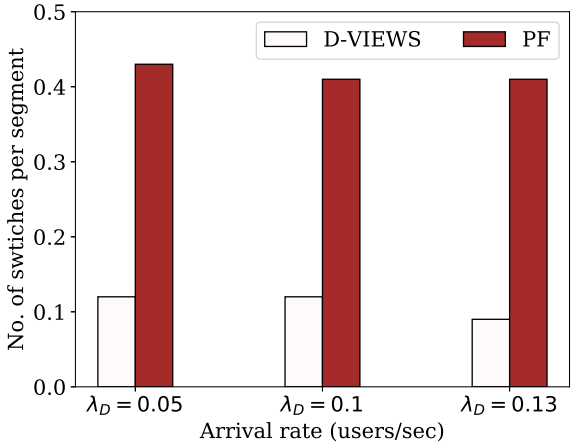



Fig. 10: **Number of video bitrate switches per segment** for different DASH flow arrival rates.

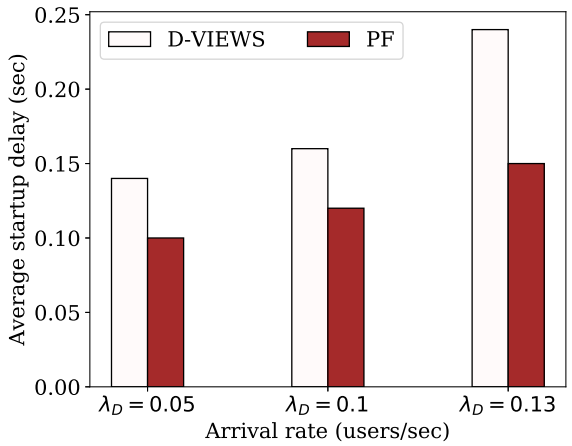



Fig. 12: **Average startup delay** for different DASH flow arrival rates.

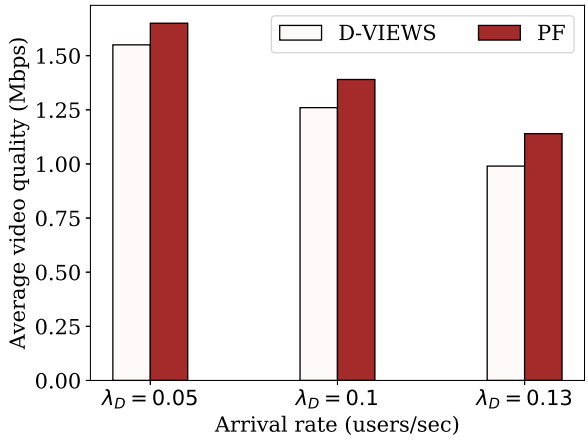



Fig. 11: **Average video bitrate** for different DASH flow arrival rates.

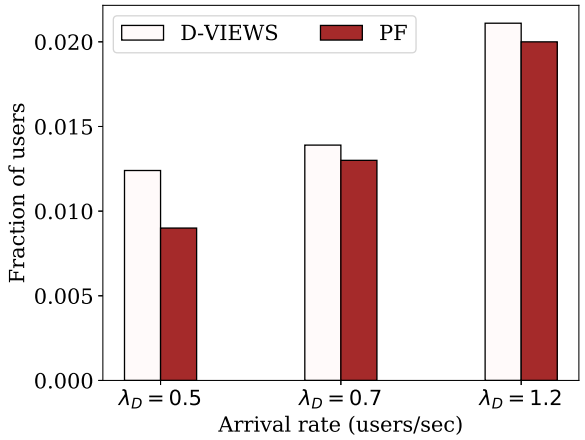



Fig. 13: **Fraction of users that experienced video stalls** for different DASH flow arrival rates.

competing DASH flows than PF scheduler. In fact, with D-VIEWS, switching rate reduces by as much as 70% while average video bitrate decreases by not more than 12% (refer Fig. 11). A drop in average video quality is expected because DASH flows often have to sacrifice some of their air-time to achieve the designated target rate. However, this will not result in under-utilization of the systems because these slots are eventually reallocated to non-DASH users in the network. It is worth noting that, during these simulations, video stalls did not occurred for any DASH client because average throughput was always higher $0.2\,Mbps$ (the lowest video bitrate).

Next, we compare D-VIEWS and PF in terms of average startup delay. From Fig. 12, we can see that D-VIEWS has a higher startup delay compared to PF. This happens because DASH flows are allocated lesser resources under D-VIEWS compared to PF. In Fig. 13, we plot fraction of user that experienced video stalls for different DASH flow arrival rates. We observe that this fraction changes by less than 5% when D-VIEWS is used instead of PF. This is a further validation for the operation of D-VIEWS scheduler which is designed to negatively affecting other QoE metrics.

### *D. Efficiency and fairness*

To study the trade-off between efficiency and fairness, we use two metrics which are critical for networks performance where multiple users share resources. These metrics are important to evaluate the performance improvement that can be attained by any cellular resource allocation solution.

***1. Aggregate rate:*** It is the expected sum of rates delivered to users of the network in a slot. D-VIEWS attains good aggregate rate, because RBs unused by DASH flows are utilized to serve non-DASH flows. In fact, the values achieved by D-VIEWS are very close to that of PF scheduler (refer Fig. 14).

***2. Fairness:*** We use Jain's fairness index [33] 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). Jain's fairness index is defined by $\mathcal{J}(\vec{J}) = \frac{(\sum_{i=1}^{n} J_i)^2}{n \sum_{i=1}^{n} J_i^2}$, where $n$ is the number of users in the system.

In order to understand the degree of fairness attained by D-VIEWS, we consider a scenario where an arriving user is either

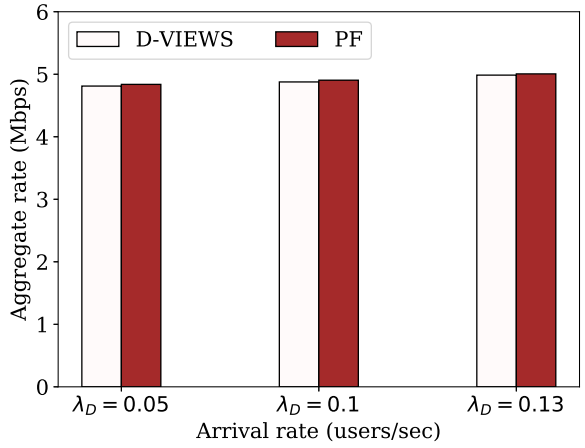



Fig. 14: **Aggregate rate** for different DASH flow rates and $\lambda_E = 0.05\ users/sec$.

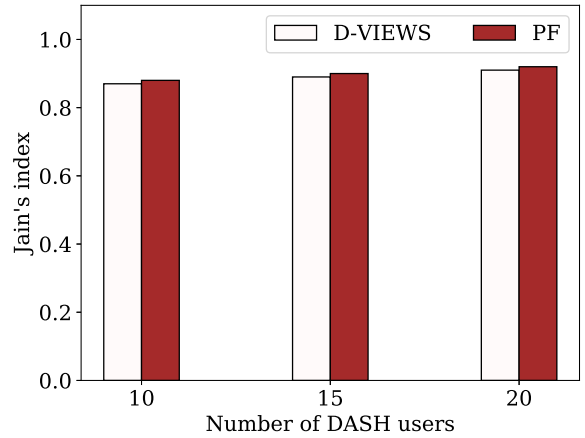



Fig. 15: **Jain's index** for different number of DASH flows in the systems.

close to eNB (a high data rate user) or close to cell edge (a low data rate user) with equal probability. For this scenario, Jain's fairness index under D-VIEWS and PF is shown in Fig. 15, which indicates that D-VIEWS, like PF scheduler, is able to perform fair resource allocation across users in the presence of heterogeneous channel conditions.

## VII. Discussions and Conclusion

In recent years, several researchers have designed schedulers for video streaming using cross-layer approaches, which require coordination among content providers, clients and eNBs. However, due to practical reasons such as scalability, network operator's policies and different video adaptation algorithms used by DASH players, tight co-ordination between content providers and network operators is often infeasible. Motivated by this, we proposed ***D**ynamic **VI**rtual **PE**nalty **W**eighted **S**cheduling (**D-VIEWS**)*. D-VIEWS is capable of enforcing bitrate stability for DASH streams without necessitating any modification to the video delivery mechanism or other network elements.

Design of D-VIEWS allows it to 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. Imagine a slice dedicated to adaptive streaming videos. The mechanism of our scheduler can be used within this slice to dynamically allocate resources. Further, by feeding information about users' throughput 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 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 setting.

In this paper, we have assumed cellular last hop to be the bottleneck, i.e., users' queue at the scheduler have infinite backlog. However, bottlenecks can also occur in the WAN path. Then, we will have to consider scenarios when users have finite and different queue lengths. An interesting future research direction would be to enhance D-VIEWS to account for size of users' queue when allocating resources. We recollect from our discussions in Section IV-B, that aggregate utility achieved by our scheduler depends on the set of bitrates available for the video streaming service. A natural question that arises is: how does the set of video bitrates impact the difference in aggregate utility obtained by D-VIEWS and utility-based scheduler? It would also be interesting to investigate sensitivity of D-VIEWS to various parameters such as set of video bitrates, number of users sharing the 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. Argyriou, D. Kosmanos, and L. Tassiulas, "Joint time-domain resource partitioning, rate allocation, and video quality adaptation in heterogeneous cellular networks," *IEEE Trans. Multimedia*, vol. 17, no. 5, pp. 736–745, May 2015.

[7] Instructables. (2017) Making Your Own Simple MPEG-DASH Server. [Online]. Available: https://www.instructables.com/id/Making-Your-Own-Simple-DASH-MPEG-Server-Windows-10/

[8] epiclabsDASH. (2017) dash.js — a reference client implementation for the playback of MPEG DASH via JavaScript and compliant browsers. [Online]. Available: https://github.com/Dash-Industry-Forum/dash.js/

[9] International Organization for Standardization, "ISO/IEC 23009-1:2012," Tech. Rep., 2012. [Online]. Available: https://www.iso.org/standard/57623.html

[10] 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.

[11] S. Thakolsri, W. Kellerer, and E. Steinbach, "QoE-based cross-layer optimization of wireless video with unperceivable temporal video quality

fluctuation," in *IEEE Int. Conf. Communications (ICC)*, June 2011, pp. 1–6.

[12] 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

[13] S. Thakolsri, S. Khan, E. Steinbach, and W. Kellerer, "QoE-driven cross-layer optimization for high speed downlink packet access," *Journal of Communications*, vol. 4, no. 9, p. 669, 2009.

[14] P. Ameigeiras, J. J. Ramos-Munoz, J. Navarro-Ortiz, P. Mogensen, and J. M. Lopez-Soler, "QoE oriented cross-layer design of a resource allocation algorithm in beyond 3G systems," *Computer Communications*, vol. 33, no. 5, pp. 571 – 582, 2010.

[15] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang, "Understanding the impact of video quality on user engagement," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 362–373, August 2011.

[16] 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 *Proc. ACM SIGCOMM*. New York, NY, USA: ACM, 2013, pp. 339–350.

[17] A. H. Zahran, J. J. Quinlan, K. K. Ramakrishnan, and C. J. Sreenan, "SAP: Stall-aware pacing for improved DASH video experience in cellular networks," in *Proc. 8th ACM on Multimedia Systems Conference*, ser. MMSys'17. New York, NY, USA: ACM, 2017, pp. 13–26.

[18] C. Yim and A. C. Bovik, "Evaluation of temporal variation of video quality in packet loss networks," *Signal Processing: Image Communication*, vol. 26, no. 1, pp. 24–38, 2011.

[19] S. Cicalò, N. Changuel, V. Tralli, B. Sayadi, F. Faucheux, and S. Kerboeuf, "Improving QoE and fairness in HTTP adaptive streaming over LTE network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 12, pp. 2284–2298, Dec 2016.

[20] S. Colonnese, F. Cuomo, T. Melodia, and I. Rubin, "A cross-layer bandwidth allocation scheme for HTTP-based video streaming in LTE cellular networks," *IEEE Commun. Lett.*, vol. 21, no. 2, pp. 386–389, Feb 2017.

[21] S. Colonnese, F. Cuomo, L. Chiaraviglio, V. Salvatore, T. Melodia, and I. Rubin, "CLEVER: a cooperative and cross-layer approach to video streaming in HetNets," *IEEE Trans. Mobile Comput.*, vol. PP, no. 99, pp. 1–1, 2017.

[22] 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.

[23] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *Proc. ACM Conf. SIGCOMM*. ACM, 2014, pp. 187–198.

[24] G. Tian and Y. Liu, "Towards agile and smooth video adaptation in dynamic HTTP streaming," in *Proc. 8th Int. Conf. Emerging Networking Experiments and Technologies*, ser. CoNEXT '12, 2012, pp. 109–120.

[25] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over HTTP," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 325–338, Aug. 2015. [Online]. Available: http://doi.acm.org/10.1145/2829988.2787486

[26] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in HTTP-based adaptive video streaming with FESTIVE," *IEEE/ACM Trans. Netw.*, vol. 22, no. 1, pp. 326–340, Feb 2014.

[27] W. Huang, Y. Zhou, X. Xie, D. Wu, M. Chen, and E. Ngai, "Buffer state is enough: Simplifying the design of QoE-aware HTTP adaptive video streaming," *IEEE Trans. Broadcast.*, pp. 1–12, 2018.

[28] 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.

[29] 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.

[30] 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.

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

[32] 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.

[33] 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.