QoE-fair Adaptive Streaming of Free-viewpoint Videos over LTE Networks

Ahmed Hamza
Simon Fraser University
Burnaby, BC, Canada

Hamed Ahmadi
Simon Fraser University
Burnaby, BC, Canada

Saleh Almowuena
Simon Fraser University
Burnaby, BC, Canada

Mohamed Hefeeda
Simon Fraser University
Burnaby, BC, Canada

ABSTRACT

Free-viewpoint video (FVV) applications enable viewers to interactively change their viewing point and watch a scene from different angles. Each FVV is composed of multiple streams representing the scene and its geometry from different vantage points. In addition, virtual views can be synthesized from captured views to provide a smoother and more immersive experience to users. Delivering FVV streaming services over cellular networks, while achieving quality-of-experience (QoE) fairness and minimizing fluctuations in perceived quality, is very challenging due to the large bandwidth requirements, the complex relationship between the bitrates of the transmitted streams and the quality of rendered virtual views, and the time-varying channel conditions. In this paper, we formulate FVV adaptive streaming as a multi-objective QoE-fairness problem and propose a heuristic algorithm to solve it efficiently. Our experiments show that the proposed algorithm achieves high QoE-fairness and provide users with high and stable qualities. It reduces quality variations by up to 32 % on average while saving up to 18 % of cellular bandwidth, compared to state-of-the-art approaches.

KEYWORDS

LTE; 5G; adaptive streaming; DASH; free-viewpoint video.

1 INTRODUCTION

Video streaming over cellular networks has become one of the most prevalent mobile services. According to a recent study by Cisco, video traffic will account for nearly 75 % of total mobile data traffic by 2020 [5]. Recent technology advances have paved the way for new video applications such as interactive 360-degree videos and free-viewpoint videos (FVV). Unlike 360-degree videos which only provide rotational movement around the center point of the camera, FVV enables viewers to move to any viewpoint that is located between cameras positioned at different locations around the scene. For example, in sports events the cameras can be placed around the field and viewers can experience the game from different perspectives. This provides a richer experience by enabling viewers to watch the scene from their view angle of interest and/or to move around obstructions to get a better view of occluded objects. Non-captured views, known as *virtual views*, are synthesized using a technique known as *depth-image-based rendering* (DIBR) [8]. DIBR generates a virtual view using two reference views, where each reference view consists of image and depth streams, as shown in Figure 1.

For video delivery, HTTP adaptive streaming (HAS) [24] has recently emerged as a simple and effective method. In HAS, videos are encoded in several bit rate versions and each version is split into small chunks called *segments*. Segments are stored on the content providers' servers or within a content distribution network (CDN). A streaming client adaptively requests video segments based on the network conditions. During each segment download, the client estimates the available bandwidth based on the download throughput and decides on the best version to request for the next segment. This adaptive mechanism enables the streaming client to provide the user with the highest *quality-of-experience* (QoE) given the network conditions. Integrating HAS with free-viewpoint videos is therefore a promising solution for deploying interactive multi-view video streaming services [12] [26] [9].

Future mobile networks such as LTE-Advanced and 5G networks will be able to support the FVV systems by providing more bandwidth capacity. In LTE networks, each cell is managed by a base station, known as eNodeB. Users within a cell share the radio channel and its radio resources. An eNodeB scheduler is responsible for distributing the radio resources between active users and quickly adapting to changing channel conditions using channel state feedbacks. In a multi-user video streaming environment where users compete for the available resources, achieving efficiency and QoE-fairness becomes critical. Most commercial eNodeB schedulers allocate resources by employing variations of the *proportional fair* [16] scheduling policy. While this type of fairness seems to perform well when all users follow the same utility, it tends to be inefficient in scenarios where users follow different utilities [27], which is typically the case with video streaming. Moreover, proportionally fair schedulers mainly focus on the network (bitrate) utility and are oblivious to application layer QoE, which may not be ideal for video streaming applications in which the relationship between rate and QoE varies within a single video stream as well as across streams [2] [25]. This problem becomes more challenging with free-viewpoint videos where the quality of synthesized virtual views depends on the encoding configuration of the components of the reference views. This makes the rate-utility relationship in these videos more complex. Moreover, the relationship between rate and perceived quality in FVV content varies from one viewpoint position to another. Therefore, changing the viewpoint would result in a different rate-quality relationship which translates to variations in the perceived quality. Recent studies have shown that such variations adversely affect the user's overall QoE [20].

ThematicWorkshops'17, October 23–27, 2017, Mountain View, CA, USA

ACM ISBN 978-1-4503-5416-5/17/10…$15.00
https://doi.org/10.1145/3126686.3126738

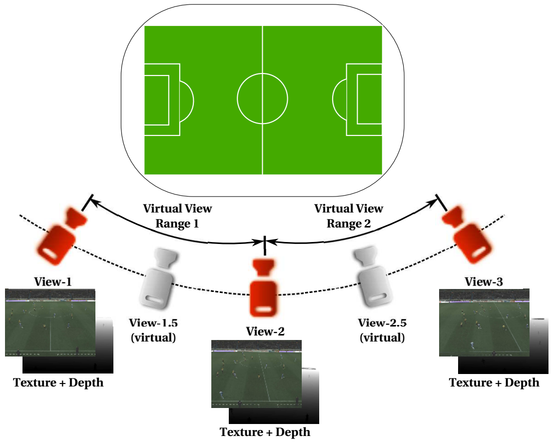

Figure 1: Free-viewpoint video using multi-view-plus-depth content representation.

In this paper, we present a network-assisted rate adaptation approach. We propose a QoE-fair radio resource allocation algorithm for DASH-based FVV systems. Our algorithm runs on a *media-aware network element* (MANE) connected to the eNodeB scheduler. The objective of the proposed algorithm is to maximize users' perceived quality and to achieve QoE-fairness among them. Unlike previous works for achieving QoE-fairness [29] [7] [4], our algorithm takes quality variations into consideration. Furthermore, current QoE-fairness approaches are not directly applicable to free-viewpoint videos since they only optimize the resource allocation based on the quality of a single view. To the best of our knowledge this is the first work that addresses the problem of QoE-fair radio resource allocation for FVV streaming systems in mobile networks.

2 RELATED WORK

2.1 Fairness in Wired Networks

Jian et al. [15] proposed an algorithm to achieve HAS fairness using a combination of harmonic bandwidth estimation, stateful and delayed bitrate update, and randomized request scheduling. In [17], Li et al. presented an approach to determine when the TCP download throughput can be taken as an accurate indicator of the fair-share bandwidth and backs off when congestion is encountered. Both [15] and [17] attempt to achieve fairness in terms of bandwidth share and do not take the difference between the rate-quality relationships of different videos into consideration. And although they may work for wired networks, they are not suitable for cellular networks with dynamic links. Moreover, when multiple HAS clients compete for the available bandwidth, user-driven approaches such as [15] and [17] will generally yield suboptimal results. Therefore, network-assisted streaming approaches which rely on active cooperation between video streaming applications and the network are more efficient. Cofano et al. [6] evaluated several network-assisted strategies of HTTP adaptive streaming in software-defined networks (SDNs) in terms of fairness, average video quality and quality variations. Mansy and Ammar [19] utilized the concept of maximal fairness, because QoE max-min fair allocations might not exist due to discrete QoE values. They proposed a QoE-based progressive algorithm, referred to as QPA in this paper, to achieve maximal fairness. These network-assisted schemes demonstrate the advantage of joint client-network adaptation. However, they are not designed for dynamic cellular networks, nor can they handle complex FVV content.

2.2 Fairness in Wireless Networks

In [28], De Vleeschauwer et al. proposed a method to adaptively set up the guaranteed bitrate of each video flow in an LTE network with heterogeneous traffic. However, the utility function used by the algorithm is not content-aware since it is not based on a video quality metric. In [29], a QoE continuum model which considers both cumulative playback quality and playback smoothness using an exponential weighted moving average is presented. Based on this model, a quality adaptation algorithm is proposed that can guarantee both QoE and fairness between multiple clients in a cellular network by exploiting the nature of human perception and video source. This algorithm tends to adjust the instantaneous quality proportional to the channel quality. However, it results in undesired quality variations in case of temporary fluctuations in channel quality. In [7], El Essaili et al. presented a QoE-based resource allocation method for HAS, which optimizes a utility function that combines the perceived quality and a penalty for quality switches. Two rate adaptation approaches (reactive and proactive) are discussed to adapt the user's application rates to the data rates chosen by the scheduler. However, in their solution fairness across users is not considered as an objective in the optimization. Cicalò et al. [4] studied the problem of QoE-fair resource allocation for HAS over cellular networks and formulated the problem as a multi-objective optimization problem in terms of maximizing the average quality and minimizing QoE differences between users. However, the given formulation fails to consider quality variation. They proposed an iterative quality-fair adaptive streaming algorithm (QFAS) to solve that formulated problem.

In this paper, we compare the performance of our algorithm against QFAS [4] and QPA [19] since they perform well in achieving fairness and high video quality in the case of 2D videos. We modify QPA to take into consideration the user's channel conditions in the case of cellular networks. We also modify both algorithms to utilize rate-utility models for FVV content, described in Section 5.1.

3 SYSTEM MODEL AND OPERATION

We consider a streaming system that supports FVV content, as shown in Figure 2.

3.1 Wireless Network Model

An LTE wireless access network with an eNodeB serves free-viewpoint videos to a set $\mathcal{K} = \{1, \ldots, K\}$ of *user equipments* (UEs). The LTE downlink channel is divided into 10 ms frames, each further divided into 1 ms sub-frames [11]. The sub-frames are transmitted using *orthogonal frequency-division multiplexing* (OFDM) which divides available radio resources into a grid in both time and frequency domains. A *resource block* is the smallest unit that can be allocated by the eNodeB in LTE. Each resource block spans

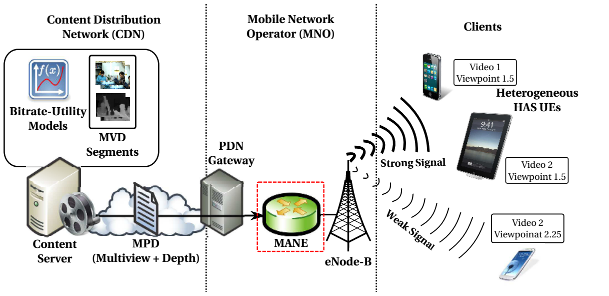

Figure 2: System model for a HAS-based FVV streaming system. Videos stored in MVD format and encoded at multiple bitrates. Resource allocation algorithm runs on MANE to ensure QoE fairness and achieve high and stable session qualities.

0.5 ms (i.e., half a sub-frame) in the time domain and 12 OFDM sub-carriers (180 kHz) in the frequency domain. UEs are dynamically allocated non-overlapping sets of resource blocks depending on their channel conditions. The channel quality in the LTE downlink is measured by the UE and sent to the eNodeB in the form of so-called *channel quality indicators* (CQIs). To accommodate the time-varying radio channel conditions of UEs, LTE uses adaptive modulation and coding. The *modulation and coding scheme* (MCS) used for each UE is based on the reported CQI value by the device.

3.2 FVV Content Model

Each UE is receiving an FVV in the *multi-view-plus-depth* (MVD) representation format, where a video is composed of a set $\mathcal{W} = \{w_1, \ldots, w_W\}$ of equidistant captured (reference) views and their associated depth maps. Depth maps can either be captured using depth cameras or estimated using a depth estimation technique [22]. In the following we refer to the texture and depth video streams corresponding to each captured view as the *component streams* of the view. Neighboring reference views bound a *virtual view range* (a set of virtual view positions). The number of virtual view positions in each virtual view range is equal to E. Therefore, the total number of possible viewpoint positions that a UE can request is $(W-1)E + W$. Component streams of each captured view are encoded at L bitrates (representations) and divided into a number of segments of duration τ seconds each. A manifest file, known as a *media presentation descriptor* (MPD), is generated for the FVV with information about the captured views and depth maps as well as the parameters of the cameras used to capture the views. For a given virtual view range, we refer to a combination of representations for the reference views of the range as an *operating point*. To support the rate adaptation process and to enable the client to choose the best operating point for a virtual view range, the MPD file also includes virtual view quality models for each virtual view range and each segment index, similar to the model in [12]. These models provide an estimate for the quality of a virtual view given the qualities of the components of the reference views. In addition, the MPD file includes a link to a file which contains for each segment index and each virtual view range the parameters of a rate-utility model. This information is needed by the resource allocation

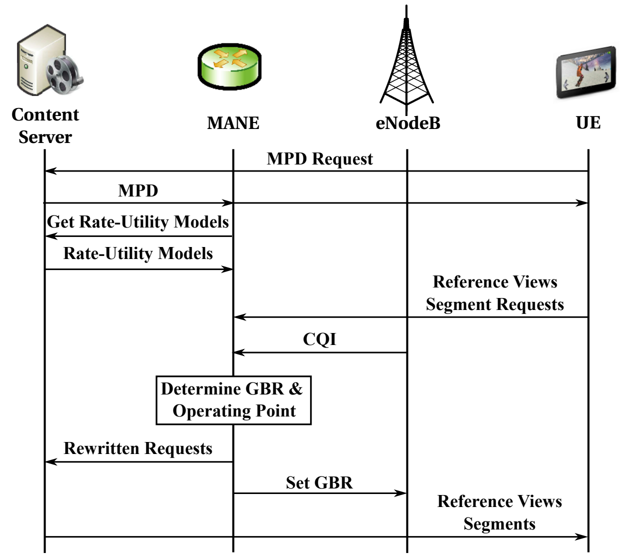

Figure 3: Sequence diagram using HTTP or HTTPS with CDN and mobile network collaboration.

algorithm to achieve QoE-fairness and is stored in a separate file since it is not utilized by the streaming client and would increase the download time of the MPD file. We describe the details of the rate-utility models and how they are generated in Section 5.1.

3.3 System Operation

FVV content, including segments of component streams and a manifest file, are hosted on servers within a content distribution network (CDN). These servers are accessible by the core network of the mobile network operator (MNO) via the packet data gateway (P-GW) which connects the core network to the Internet. Alternatively, the MNO may host the FVV content on their own mobile content distribution network to provide fast and efficient delivery and establish implicit trust between the components of the mobile network infrastructure and the CDN [30]. The manifest (MPD) file for the free-viewpoint videos contains an XML element that provides a URL from which a rate-utility models file for the virtual view ranges and segment indices can be retrieved. An FVV streaming client runs on each UE and keeps track of the user's viewpoint position and available network bandwidth. The client issues segment requests to the content server for the texture and depth component streams of the two reference views bounding the virtual view range to which the target view position belongs. Decisions for the operating points to be requested from the server are based on a virtual view quality-based rate adaptation method similar to the one presented in [12]. Communication between the streaming client and the content server can either be over non-secure HTTP or secure HTTP (HTTPS). A media-aware network element (MANE) is connected to the eNodeB and is able to intercept HAS requests from UEs. We distinguish between two scenarios. In the first scenario, we assume that either the client requests are sent using HTTP or they are sent using HTTPS but with a common certificate and encryption keys used by both the CDN and the MANE. The communication between different network components is shown in Figure 3. When the client issues a request for reference views, the MANE intercepts the request and uses the segment index and view ids to lookup the parameters of the corresponding rate-utility model. In the second scenario, the client communicates with the content server over a secure protocol but no certificates or keys are shared between the CDN and MANE. This requires collaboration between the streaming clients and the MANE through a separate control plane. This is achieved using an approach similar to [3], which manages streaming sessions over HTTPS in SDN networks.

4 PROBLEM STATEMENT

PROBLEM 1. *Given the channel conditions of multiple FVV streaming sessions, determine the optimal number of resource blocks to be assigned to each streaming session and the corresponding GBR such that the: (i) average QoE for each session is maximized, (ii) QoE difference across all streaming sessions is minimized, and (iii) quality fluctuation within each session is minimized.*

Let T_s be the scheduling time interval. At each time instant nT_s, $n \in \mathbb{N}$, the MANE needs to determine the users GBR values $R_k[n]$ which achieve QoE-fairness between UEs in the following scheduling window, where $k = 1, \dots, K$. We denote by $\mathcal{V} = \{v_1, \dots, v_K\}$ the set of videos being streamed by UEs. Let m_k be the MCS value chosen by the eNodeB for user k based on the reported CQI by the UE of that user, where $m \in [1, M]$. The per-resource block capacity c_m is a non-decreasing quantity of the MCS mode m such that $c_1 \leq c_2 \leq \cdots \leq c_M$. Let Π be the total number of resource blocks available for UEs within a scheduling window. This number may vary from one scheduling window to another and can be dynamically computed as in [28]. We denote by r_k the total bitrate for the representations in the chosen operating point for video v_k. For two reference views where the component streams are encoded into L representations, the number of possible operating points equals the number of all possible representation combinations for the four components (i.e., L^4). Let the set of bitrates corresponding the operating points be $\mathcal{R}_k = \{r_{k,1}, \dots, r_{k,L^4}\}$. We denote by $r_{k,\min}$ and $r_{k,\max}$ the minimum operating point bitrate and maximum operating point bitrate, respectively. Let $s_k[n]$ and $\lambda_k[n]$ be the segment index and virtual view range index requested by user k at time n, respectively. In the following, we drop the scheduling window index n for brevity. Given a rate-utility model U_{k,s_k,λ_k}, the problem can be formulated as follows

$$\max \quad \sum_{k=1}^{K} U_{k,s_k,\lambda_k}(r_k[n]) \tag{1a}$$

$$\min \quad \sum_{k=1}^{K} \sum_{j>k} \Delta(U_{k,s_k,\lambda_k}(r_k[n]), U_{j,s_j,\lambda_j}(r_j[n])) \tag{1b}$$

$$\min \quad \sum_{k=1}^{K} \Gamma(U_{k,s_k[n-1],\lambda_k[n-1]}(r_k[n-1]), U_{k,s_k,\lambda_k}(r_k[n])) \tag{1c}$$

$$\text{s.t.} \quad \sum_{k=1}^{K} \lceil \frac{R_k[n]\tau}{c_{m_k}} \rceil \leq \Pi \tag{1d}$$

$$U_{k,s_k,\lambda_k}(r_{k,\min}) \leq U_{k,s_k,\lambda_k}(r_k[n]) \leq U_{k,s_k,\lambda_k}(r_{k,\max}), k \in \mathcal{K}, \tag{1e}$$

where c_{m_k} is the resource block capacity for UE k given MCS m_k.

Similar to [4], the Δ function in (1b) is a utility-fairness metric defined as:

$$\Delta(U_i, U_j) = \begin{cases} 0 & \text{if} \quad U_i = f_U(r_{i,\min}) \wedge U_j < U_i \\ 0 & \text{if} \quad U_i = f_U(r_{i,\max}) \wedge U_j > U_i \\ |U_i - U_j| & \text{otherwise,} \end{cases} \tag{2}$$

where $\wedge$ denotes the AND operator. This metric takes into consideration that U_i, U_j are constrained to their minimum and maximum values. If one of the videos achieves its maximum utility, the available resources should be used to increase the utilities of other videos. When resources are scarce, if the i-th video is at its minimum utility value, decreasing its rate is not possible. It is therefore necessary to decrease the rate of the other videos. We note that unlike other formulations in previous works (e.g., [4]), our formulation has an explicit objective for minimizing quality changes for each user, Eq (1c). The Γ function in (1c) is a measure for the change in quality. This enables us to evaluate quality changes between the current allocation window and the previous window for each user. It has been shown that QoE degradations caused by a change in quality where the quality is increased is much smaller than a change causing a decrease in quality of the same scale [18]. Therefore, we define Γ as follows

$$\Gamma(U_i, U_j) = \begin{cases} |U_i - U_j| & \text{if} \quad U_i < U_j \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

The problem in Eq. (1) is a mixed integer non-linear programming (MINLP), and therefore NP-hard, and has multiple (potentially conflicting) objectives. As the optimal solution is computationally expensive, in the following we propose a heuristic algorithm to efficiently solve this problem.

5 PROPOSED SOLUTION

To solve the FVV streaming radio resource allocation problem, our proposed QoE-fair allocation algorithm utilizes virtual view rate-utility models for estimating the quality of the views synthesized at the client. Radio resource blocks are incrementally allocated across users based on the estimated perceived quality for each user as well as the perceived qualities for the previous scheduling window. In this section, we describe how to generate the rate-utility models and present the proposed algorithm.

5.1 Rate-Utility Models for FVV

A QoE-aware resource allocation algorithm requires access to the relation between the video bitrate and the perceived quality. Unlike 2D, where there is only one video stream, each free-viewpoint video has multiple video streams corresponding to the components of the different views. The relation between bitrate and quality for virtual views is therefore complicated by the fact that changes in the bitrates of the component streams do not equally contribute to the quality of the synthesized virtual view. We consider the following parametric rate-utility model:

$$U_{k,i,j}(r_k) = f(r_k; \boldsymbol{\alpha}_{k,i,j}), \tag{4}$$

where k is the UE index, i is the segment index, j is the virtual view range index, r_k is the total bitrate for the requested components for segment i of the video streamed by user k, and $\boldsymbol{\alpha}_{k,i,j} \in \mathcal{A} \subset \mathbb{R}^{N_\alpha}$

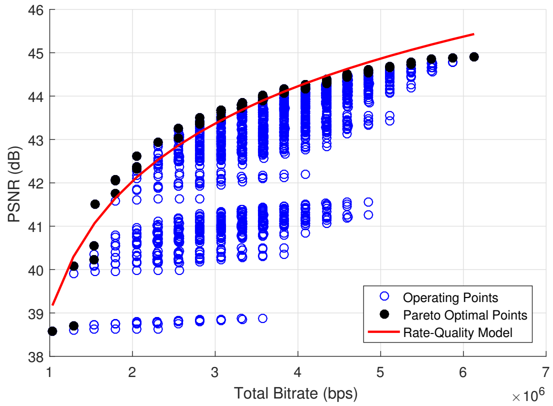

Figure 4: Operating points for a virtual view generated from two reference views and their associated depth maps, each component having 6 representations.

is a time varying and content-dependent vector of N_α parameters. To determine the model that best represents the relationship between the operating point bitrate and the quality of the virtual views for a free-viewpoint video, we generate HAS content based on the system model described in Section 3.2 for three MVD videos. For each segment index and each virtual view range, we generate a scatter plot for all operating points. We show an example plot for the Kendo MVD video sequence in Figure 4, where the components of views 1 and 3 are used to synthesize views 1.5, 2, and 2.5 and each component is encoded into 6 constant bit rate representations (250, 500, 750, 1000, 1250, 1500 kbps). Similar plots were obtained for other videos. Each point in the figure designates an operating point with a total bitrate equal to the sum of the components bitrates and the corresponding average quality for the synthesized virtual views. The quality of the virtual views is measured in *peak signal-to-noise ratio* (PSNR) using views synthesized from the original uncompressed components as references. As can be seen in the figure, the filled points represent the Pareto-optimal points which provide the maximal quality for a given bitrate. The rate-utility relationship for each virtual view range can therefore be obtained offline by applying a curve fitting method on this set of points. In our evaluation, we chose the following logarithmic model

$$U_{k,i,j}(r_k) = f(r_k; \boldsymbol{\alpha}_{k,i,j}) = \alpha_1 \log(\alpha_2 r_k + \alpha_3), \tag{5}$$

where parameters $\alpha_1, \alpha_2, \alpha_3$ are the elements of $\boldsymbol{\alpha}_{k,i,j}$.

5.2 Quality-fair FVV Rate Allocation

To achieve QoE-fairness and minimize fluctuations in perceived quality, we propose the *Quality-fair Free-viewpoint Video Resource Allocation* (QFVRA) algorithm presented in Algorithm 1. In Algorithm 1, $\varphi_m(\cdot)$ is a function that maps a number of radio resources to the corresponding data rate given the modulation and coding scheme m. Assuming that a feasible solution is achievable, i.e., the sum of minimum bitrate resource blocks for all users is less than or equal to Π, the algorithm proceeds as follows. For the first scheduling window, each user is initially assigned a number of resource blocks that corresponds to the bitrate of the minimum operating point of the video being streamed and the UE's reported channel condition (lines 2 to 4). In line 5, the algorithm then uses the function PRA to iteratively add resource blocks to users. The users are sorted based on their estimated perceived quality and the user receiving minimal quality is allocated an additional resource block.

Algorithm 1: QFVRA

Input: Set $\mathcal{K}$ of UEs, where $|\mathcal{K}| = K$
Input: Vectors $\mathbf{r}_{\min} = (r_{1,\min}, \ldots, r_{K,\min})$ and $\mathbf{r}_{\max} = (r_{1,\max}, \ldots, r_{K,\max})$ with minimum and maximum operating point bitrates, respectively, of videos streamed by UEs
Input: Vector of users' channel conditions $\mathbf{m} = (m_1, \ldots, m_K)$
Input: Number of radio resources in a scheduling window Π
Input: Set of vectors $\mathcal{A} = \{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_K\}$, where each vector $\boldsymbol{\alpha}_k \in \mathcal{A}$ represents the values of N_α parameters for the utility function of UE k and $\mathcal{A} \subset \mathbb{R}^{N_\alpha}$
Input: Allocation vector of previous scheduling window $\mathbf{p} = (p_1, \ldots, p_K)$
Input: Vector $\mathbf{q}' = (q'_1, \ldots, q'_K)$ for estimated qualities in previous scheduling window
Input: Quality gain thresholds β_1 and β_2
Output: Allocation vector $\mathbf{x} = (x_1, x_2, \ldots, x_K)$
Output: Estimated qualities vector $\mathbf{q} = (q_1, q_2, \ldots, q_K)$ based on allocation

1 **if** *First scheduling window* **then**
2   **for** $i \leftarrow 1$ **to** K **do**
3     $x_i \leftarrow \lceil \frac{r_{i,\min}\tau}{c_{m_i}} \rceil$
4     $q_i \leftarrow f(\varphi_{m_i}(x_i); \boldsymbol{\alpha}_i)$
5   $\mathbf{x} \leftarrow \mathsf{PRA}(K, \Pi, \mathbf{m}, \mathbf{r}_{\min}, \mathbf{r}_{\max}, \mathcal{A}, \mathbf{x})$
6 **else**
7   **for** $i \leftarrow 1$ **to** K **do**
8     $b_i \leftarrow \lceil f^{-1}(q'_i; \boldsymbol{\alpha}_i)\tau / c_{m_i} \rceil$
9     $x_i \leftarrow b_i$
10   **if** $\sum_{i=1}^{K} x_i < \Pi$ **then**
11     $\mathbf{x} \leftarrow \mathsf{PRA}(K, \Pi, \mathbf{m}, \mathbf{r}_{\min}, \mathbf{r}_{\max}, \mathcal{A}, \mathbf{x})$
12   **for** $i \leftarrow 1$ **to** K **do**
13       $q_i \leftarrow f(\varphi_{m_i}(x_i); \boldsymbol{\alpha}_i)$
14   **if** $(m_i^{instability} > \gamma)$ *OR* $(\beta_1 > q_i - q'_i)$ **then**
15         $x_i \leftarrow b_i$
16         $q_i \leftarrow f(\varphi_{m_i}(x_i); \boldsymbol{\alpha}_i)$
17   **else if** $(\beta_2 < (q_i - q'_i))$ **then**
18         $x_i \leftarrow \lceil f^{-1}(\beta_2; \boldsymbol{\alpha}_i)\tau / c_{m_i} \rceil$
19         $q_i \leftarrow \beta_2$
20   **else if** $\sum_{i=1}^{K} x_i > \Pi$ **then**
21     $\mathbf{x} \leftarrow \mathsf{PRA}(K, \Pi, \mathbf{m}, \mathbf{r}_{\min}, \mathbf{r}_{\max}, \mathcal{A}, \mathbf{x})$
22   **for** $i \leftarrow 1$ **to** K **do**
23       $q_i \leftarrow f(\varphi_{m_i}(x_i); \boldsymbol{\alpha}_i)$
24   **else**
25   **for** $i \leftarrow 1$ **to** K **do**
26       $q_i \leftarrow f(\varphi_{m_i}(x_i); \boldsymbol{\alpha}_i)$
27 **return** $\mathbf{x}, \mathbf{q}$

The estimated quality that each user gets is calculated based on the rate-utility model given in Eq. (4) for the virtual view range that the user's current viewpoint belongs to. This process is repeated until either all users reach data rates equivalent to the bitrates of the maximum operating points of their respective videos or all resource blocks within the scheduling window are allocated. It should be noted that if adding a resource block to a user causes their data rate to exceed the maximum operating point bitrate, this user is skipped and is no longer considered by the algorithm within this scheduling window.

The progressive resource allocation algorithm described above ensures fairness among users within the scheduling window in terms of perceived quality. However, due to the time-varying channel conditions of users, running this algorithm for each scheduling window independently without considering the qualities achieved from previous allocations may not result in a stable and smooth perceived quality for each user. To minimize quality variation across scheduling windows, QFVRA first attempts to maintain the same quality values that the users obtained in the previous scheduling window. This is done by finding the corresponding data rates for these quality values using $f_U^{-1}(U, \boldsymbol{\alpha}_{k,s_k,\lambda_k})$, which is the inverse function of f_U in Eq. (4), and calculating the number of needed resource blocks for each user to achieve these data rates in the current scheduling window given the users' new channel conditions at the beginning of the window (line 8). The total number of needed resource blocks is then compared against the capacity of the scheduling window to determine how to proceed. If the total number of resource blocks required to maintain the qualities in the previous scheduling window exceeds the number of available resource blocks within the current window (line 20), a *draining* process is utilized. Similar to progressive filling, users are first sorted based on an estimate of their perceived qualities given the current allocation. However, instead of adding resource blocks, the algorithm iteratively removes one resource block from the user with the highest estimated quality value in each iteration until the capacity of the scheduling window is reached.

When the qualities in the previous scheduling window are achievable and a number of remaining resource blocks are available, distributing these blocks among users is not trivial. If a user is suffering from temporary changes in channel conditions, assigning more resource blocks to them will soon be followed by reclaiming those resource blocks in the following scheduling window causing variations in the perceived quality. To maintain long term QoE smoothness and avoid fluctuations in the perceived quality, QFVRA considers the stability of the channel condition for each user and only allocates more resource blocks to those users with stable conditions. Therefore, the scheduler needs to estimate the conditions of users channels. This is done by utilizing the CQI history in order to predict the stability of the channel conditions in the next scheduling window. Another issue which arises when increasing the resource blocks allocated to users is that when the allocation does not result in a noticeable change in quality, it is more efficient to assign these resource blocks to users who are suffering from bad channel conditions so that they can enjoy a quality comparable to other users. On the other hand, when the allocation results in huge difference in quality, it is better to limit the amount of increase in quality. For example, if channel conditions drop in the near future, the impact of quality switch would be reduced. If channel conditions were good and allow more quality improvements, the quality would be gradually increased and the user would not suffer from abrupt changes.

To overcome these issues, QFVRA maintains a set of users which are eligible for quality improvement. This set includes users which satisfy two conditions: (i) they have relatively stable channel conditions; and (ii) additional resources allocated to them through progressive filling will result in noticeable improvement in perceived quality. After running the PRA algorithm to calculate a QoE-fair allocation starting from a resource allocation providing the qualities of the previous window, QFVRA checks both the stability of each user's channel conditions as well as the quality difference between the two consecutive windows (line 14). We use the standard deviation of each user's channel conditions over a window of H previous CQI reports, $m_i^{\text{instability}}$ in Algorithm 1, to estimate the stability by comparing it against a stability threshold γ. Parameter β_1 defines a quality gain threshold corresponding to a just-noticeable-difference. Users that do not satisfy these conditions are assigned a number of resource blocks that maintain their perceived quality in the previous window (lines 15 to 16). Parameter β_2 defines a second threshold that limits the amount of quality change between consecutive scheduling windows (line 17).

6 EVALUATION

6.1 Setup

We simulate an LTE cellular network using OPNET Modeler and its LTE module [31]. We implement the proposed QFVRA algorithm and use the channel conditions obtained from OPNET as input to the algorithm. We also implement the QFAS [4] and QPA [19] algorithms to compare the performance of our algorithm against them. Table 1 shows the configuration of the simulated network. Other parameters are set to the default values of the OPNET LTE module. In practical scenarios, mobile operators usually install base stations in crowded areas to serve most users with strong signals. Accordingly, in our simulations, mobile users are randomly distributed within each cell such that the majority of the users, about 90 % of them, are densely populated within $\frac{1}{3}$ of cell radius and the rest are sparsely scattered around the rest of the cell area. We evaluate multiple scenarios where the number of users is 10, 20, 30, 40, and 50. We run each scenario 5 times and report the average of the results.

Time is divided into scheduling windows with a duration of one second. The simulator runs the resource allocation algorithm at the beginning of each scheduling window. We configure users to move following the random way-point model in which mobility speed is randomly chosen between 0 and 5 m/s. We configure the mobile devices to send CQI reports to the associated base station every 100 ms. We choose this reporting interval to ensure that we do not miss any channel condition changes, and at the same time we do not receive unnecessary frequent reports. For QFVRA, we use a sliding window of 20 CQI reports to assess the stability of the UEs channels and we set the stability threshold γ to 1.0. Quality gain thresholds β_1 and β_2 are set to 0.5 dB and 1.0 dB, respectively.

Table 1: Mobile Network Configuration.

Physical Profile	LTE 20 MHz FDD
eNodeB Antenna Gain	15 dBi
UE Antenna Gain	−1 dBi
Max. Downlink Bitrate	6000 kbps
Cell Radius	5×5 Km
Transmission Power	0.0558 W
Propagation Model	Pedestrian (ITU-R M.1225)
Mobility Model	Random Way-point (0 − 5 m/sec)
Simulation Time	10 minutes

We use three MVD FVV sequences from the MPEG 3DV ad-hoc group data sets [13] [21] which have different characteristics: Kendo, Balloons, and Café. The resolution for Kendo and Balloons is 1024×768 and the resolution of the Café sequence is 1920×1080. The Kendo and Balloons sequences have moving cameras while the cameras in Café are fixed. We extend the length of the video sequences from 10 to 360 seconds by repeating the frame sequence. For each video, we choose three cameras from the set of captured views and we allow three virtual views within each virtual view range, for a total of 6 supported virtual view positions. The video streams for the texture and depth components of each camera are then encoded using a CBR configuration for the H.264/AVC encoder with bitrate values of 250, 500, 750, 1000, 1250, and 1500 kbps. We use the GPAC framework [1] to generate one second segments for the different representations of each component. For each segment index, we generate virtual view quality models for all supported virtual view positions, as discussed in Section 3.2. We also generate rate-utility models for each virtual view range. Our evaluation is based on the following metrics:

- *Quality of Experience.* To the best of our knowledge, there is no comprehensive quantitative measure to evaluate QoE for adaptive video streaming. However, low video quality and frequent and/or high-amplitude quality switches are believed to result in low QoE [10] [23]. Therefore, in order to assess QoE, we use two metrics: video quality and quality switches. We measure the *average video quality* (AVQ) for user j as

$$Q_{\text{AVQ}} = \frac{\sum_{i=1}^{T} q_i^j}{T}, \tag{6}$$

where where q_i is the video quality in the scheduling window i, and T is the total number of scheduling windows during the user's session. For quality switches, we measure the average amplitude of quality switches over time for each user. Since, it is observed that the QoE degradations caused by upward video quality switches are much smaller than downward switches of the same scale [18], we only measure the rate of *downward video quality switches* (DVQS) as

$$Q_{\text{DVQS}} = \frac{\sum_{i=2}^{T} I(q_i^j, q_{i-1}^j)}{T}, \tag{7}$$

$$I(q_i^j, q_{i-1}^j) = \begin{cases} q_{i-1}^j - q_i^j & \text{if } q_i^j < q_{i-1}^j \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

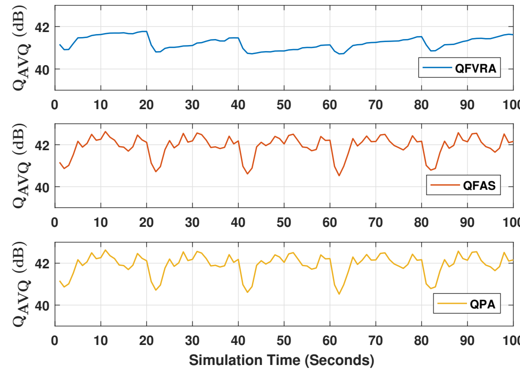

Figure 5: Average video quality over time (20 users).

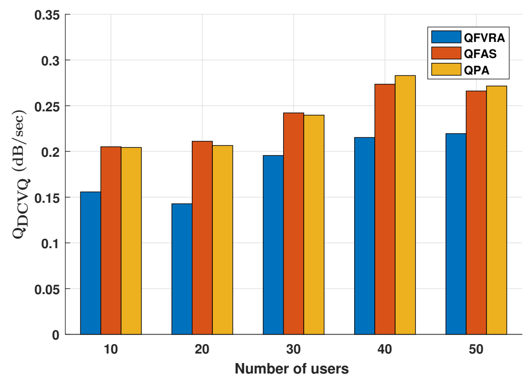

Figure 6: Average rate of downward video quality switches.

- *Network Utilization.* We measure the percentage of saved resource blocks (SRB) in each scheduling window as

$$U_{\text{SRB}} = \left(1 - \frac{\sum_{i=1}^{N} a_i}{\Pi}\right) * 100, \tag{9}$$

where a_i is the number of assigned resource blocks to user i.

- *Fairness.* We measure QoE-fairness based on Jain's Index [14] in scheduling window i as

$$F_{\text{QoE}} = \frac{\left(\sum_{i=1}^{N} q_i^j\right)^2}{N \sum_{i=1}^{N} (q_i^j)^2}, \tag{10}$$

where N is the number of admitted users in the network.

- *Running Time.* The time required to compute the allocation.

6.2 Results

Quality of Experience. Figure 5 shows the average video quality over a 100-second period of simulation for the 20-user scenario. Note that in each scenario run, users randomly choose to watch one of the three FVV videos. The figure demonstrates how our proposed algorithm avoids potentially damaging upward quality switches to achieve a better quality of experience. The other two algorithms increase the video quality at every possibility, regardless of the stability of channel conditions. Although this approach provides higher video qualities, it results in fluctuating perceived quality, and consequently less QoE, especially when users are experiencing unstable channel conditions. QFVRA, however, minimizes

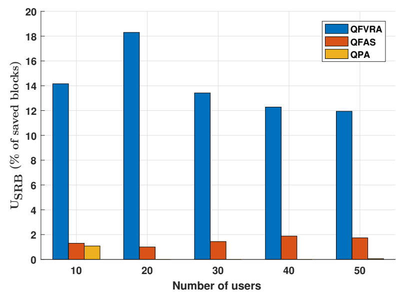

Figure 7: Percentage of saved blocks.

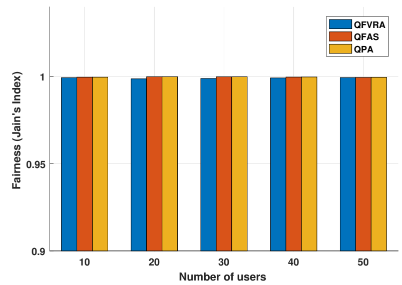

Figure 8: Fairness across users.

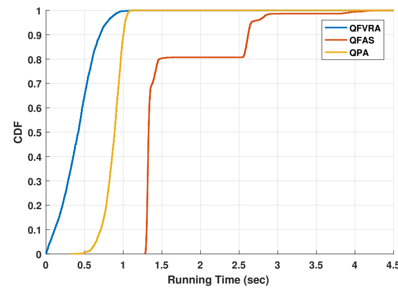

Figure 9: Average running time.

the impact of quality switches. Figure 6 shows that our algorithm outperforms QFAS and QPA by achieving at least 12 % and up to around 32 % less rates of downward quality switches. As can be seen, the amount of improvement varies for different scenarios. This amount is affected by the channel conditions and the rate-distortion complexity of video segments watched by users. Note that since quality levels for virtual views are different from those of reference views, we report the amplitude of quality switches per second instead of just reporting their frequencies.

Network Utilization. Increasing video quality blindly in each scheduling window without considering the previous qualities that users have been experiencing as well as possible variations in channel conditions has another drawback, namely *bandwidth underutilization*. Figure 7 shows that our algorithm is able to save around 12 to 18 percent of the total available resource blocks while providing almost the same average video quality and significantly less quality variations. These resource blocks can be provisioned for other traffic in the same base station. In the same figure, it can be seen that QPA has saved some resource blocks in cases of 10 and 50 users although based on its design it assigns to all the available resource blocks to the users. This observation is due to the fact that these scenarios are at the extreme ends of resource block availability. As the number of users decreases, QPA is more likely to provide them with the maximum possible qualities. On the other hand, as the system gets more and more crowded, available resource blocks might not even be sufficient to serve users with minimum possible qualities. Therefore, no resource blocks would be assigned to users for some scheduling windows. Similar to QPA, QFAS also tends to exhaust all the available resource blocks. However, since it solves the problem in continuous domain and chooses the nearest rate-distortion points in discrete domain, a small portion of resource blocks, around 1.5 %, remain unassigned.

Fairness. Figure 8 represents the average Jain's Index across users for different numbers of FVV clients. As can be seen, the three algorithms achieve fairness and their performance is not influenced by the number of clients in the cell.

Running Time. Figure 9 shows the empirical cumulative distribution function of the average running time for each scheduling window for the 40-user scenario. Our simulations have been conducted on a PC with a CPU of 2.7 GHz and 16 GB of memory. The running time for each scheduling window in QFAS and QPA is expected to be virtually the same. However, it is observed that a number of scheduling windows in QFAS take more time and this number increases as the number of clients increase. This is due to the running time of the Newton's method utilized in QFAS, which is not quite constant and depends on the complexity of derivative of the input function. For QFVRA, the running time for each scheduling window can vary depending on how far the next allowable solution is from the solution in the previous scheduling window. In the worst case, the performance of our algorithm would reach that of QPA. This only occurs if the channel conditions for all users experience frequent abrupt changes all the time, which is an extreme case. It is clear that our algorithm's execution time is noticeably faster than that of QFAS and QPA. For example, QFVRA runs 65 % of the scheduling windows in less than 0.5 seconds while this is around 0 % and 1 % for QFAS and QPA, respectively. In almost all scheduling windows, the running time of our algorithm was less than the scheduling interval. Therefore, it can easily run in real-time.

7 CONCLUSIONS

We presented a QoE-fair resource allocation algorithm for adaptive streaming of free-viewpoint videos over cellular networks. The proposed algorithm utilizes virtual view rate-quality models to allocate the radio resources such that the differences in users' perceived qualities are minimized, i.e., fairness in terms of QoE is achieved. It also minimizes the frequency and amplitude of quality switches by taking into account the qualities observed by the clients as a result of previous allocations and avoiding unnecessary quality increases when the user's channel conditions are unstable. We simulated an LTE network and evaluated the performance of our algorithm and compared it against the closest algorithms in the literature. Results show that our algorithm achieves a high level of fairness, and it reduces the rate of quality switches by up to 32 % compared to others. In addition, it saves up to 18 % of the radio resource blocks of the cellular network while achieving comparable average quality to others.

ACKNOWLEDGMENTS

This work is partially supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada, and by the Qatar National Research Fund (grant #[NPRP8-519-1-108]).

REFERENCES

[1] GPAC multimedia framework. http://www.gpac.io.

[2] N. Changuel, B. Sayadi, and M. Kieffer. Control of distributed servers for quality-fair delivery of multiple video streams. In *Proceedings of the ACM International Conference on Multimedia (MM'12)*, pages 269–278, 2012.

[3] J. Chen, M. Ammar, M. Fayed, and R. Fonseca. Client-driven network-level QoE fairness for encrypted 'DASH-S'. In *Proceedings of the ACM Workshop on QoE-based Analysis and Management of Data Communication Networks*, Internet-QoE '16, pages 55–60, 2016.

[4] 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 Transactions on Circuits and Systems for Video Technology*, 26(12):2284–2298, December 2016.

[5] Cisco visual networking index: Global mobile data traffic forecast update, 2014-2019. White Paper, February 2016. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html.

[6] G. Cofano, L. De Cicco, T. Zinner, A. Nguyen-Ngoc, P. Tran-Gia, and S. Mascolo. Design and experimental evaluation of network-assisted strategies for HTTP adaptive streaming. In *Proceedings of the ACM International Conference on Multimedia Systems (MMSys'16)*, pages 3:1–3:12, 2016.

[7] A. E. Essaili, D. Schroeder, E. Steinbach, D. Staehle, and M. Shehada. QoE-based traffic and resource management for adaptive HTTP video delivery in LTE. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(6):988–1001, June 2015.

[8] C. Fehn. Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV. In *Proceedings of SPIE Stereoscopic Displays and Virtual Reality Systems XI*, pages 93–104, 2004.

[9] Z. Gao, S. Chen, and K. Nahrstedt. OmniViewer: Enabling multi-modal 3D DASH. In *Proceedings of the ACM International Conference on Multimedia (MM'15)*, pages 801–802, October 2015.

[10] M. N. Garcia, F. D. Simone, S. Tavakoli, N. Staelens, S. Egger, K. Brunnström, and A. Raake. Quality of experience and HTTP adaptive streaming: A review of subjective studies. In *Proceedings of the International Workshop on Quality of Multimedia Experience (QoMEX'14)*, pages 141–146, September 2014.

[11] A. Ghosh, J. Zhang, J. G. Andrews, and R. Muhamed. *Fundamentals of LTE*. Prentice Hall, 1st edition, 2010.

[12] A. Hamza and M. Hefeeda. Adaptive streaming of interactive free viewpoint videos to heterogeneous clients. In *Proceedings of the ACM International Conference on Multimedia Systems (MMSys'16)*, pages 10:1–10:12, 2016.

[13] ISO/IEC. Description of Exploration Experiments in 3D Video Coding. Doc. N11095, ISO/IEC JTC1/SC29/WG11 (MPEG), January 2010.

[14] R. Jain, D. Chiu, and W. Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Technical Report TR-301, Digital Equipment Corporation, 1984.

[15] J. Jiang, V. Sekar, and H. Zhang. Improving fairness, efficiency, and stability in HTTP-based adaptive video streaming with FESTIVE. In *Proceedings of the International Conference on Emerging Networking Experiments and Technologies (CoNEXT'12)*, pages 97–108, 2012.

[16] F. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8(1):33–37, 1997.

[17] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran. Probe and adapt: Rate adaptation for HTTP video streaming at scale. *IEEE Journal on Selected Areas in Communications*, 32(4):719–733, April 2014.

[18] Y. Liu, S. Dey, D. Gillies, F. Ulupinar, and M. Luby. User experience modeling for DASH video. In *Proceedings of the International Packet Video Workshop (PV'13)*, pages 1–8, December 2013.

[19] A. Mansy, M. Fayed, and M. Ammar. Network-layer fairness for adaptive video streams. In *IFIP Networking Conference*, pages 1–9, May 2015.

[20] R. K. Mok, E. W. Chan, X. Luo, and R. K. Chang. Inferring the QoE of HTTP video streaming from user-viewing activities. In *Proceedings of the ACM Workshop on Measurements Up the Stack (W-MUST'11)*, pages 31–36, 2011.

[21] Nagoya University FTV test sequences. http://www.fujii.nuee.nagoya-u.ac.jp/multiview-data/mpeg/mpeg_ftv.html.

[22] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3):7–42, 2002.

[23] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia. A survey on quality of experience of HTTP adaptive streaming. *IEEE Communications Surveys Tutorials*, 17(1):469–492, 2015.

[24] I. Sodagar. The MPEG-DASH standard for multimedia streaming over the Internet. *IEEE MultiMedia*, 18(4):62–67, April 2011.

[25] G. M. Su, Z. Han, M. Wu, and K. J. R. Liu. A scalable multiuser framework for video over OFDM networks: Fairness and efficiency. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(10):1217–1231, October 2006.

[26] T. Su, A. Sobhani, A. Yassine, S. Shirmohammadi, and A. Javadtalab. A DASH-based HEVC multi-view video streaming system. *Journal of Real-Time Image Processing*, pages 1–14, 2015.

[27] G. Tychogiorgos, A. Gkelias, and K. K. Leung. Utility-proportional fairness in wireless networks. In *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'12)*, pages 839–844, September 2012.

[28] D. D. Vleeschauwer, H. Viswanathan, A. Beck, S. Benno, G. Li, and R. Miller. Optimization of HTTP adaptive streaming over mobile cellular networks. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM'13)*, pages 898–997, April 2013.

[29] Z. Yan, J. Xue, and C. W. Chen. QoE continuum driven HTTP adaptive streaming over multi-client wireless networks. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME'14)*, pages 1–6, July 2014.

[30] F. Z. Yousaf, M. Liebsch, A. Maeder, and S. Schmid. Mobile CDN enhancements for QoE-improved content delivery in mobile operator networks. *IEEE Network*, 27(2):14–21, March 2013.

[31] Y. Zaki, T. Weerawardane, C. Görg, and A. Timm-Giel. Long term evolution (LTE) model development within OPNET simulation environment. In *Proceedings of OPNETWORK 2011*, Bethesda, MD, USA, August 2011.