IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 29, 2019, accepted September 2, 2019, date of publication September 10, 2019,
date of current version September 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2940292

Edge Computing Assisted Joint Quality Adaptation for Mobile Video Streaming

WAQAS UR RAHMAN, (Member, IEEE), CHOONG SEON HONG, (Senior Member, IEEE), AND EUI-NAM HUH, (Member, IEEE)
Department of Computer Science and Engineering, Kyung Hee University, Yongin 17104, South Korea

Corresponding authors: Choong Seon Hong (cshong@khu.ac.kr) and Eui-Nam Huh (johnhuh@khu.ac.kr)

This work was supported in part by the MSIT (Ministry of Science and ICT), South Korea, under the ICT Consilience Creative Program under Grant IITP-2019-2015-0-00742, and in part by the "Service Mobility Support Distributed Cloud Technology" supervised by the Institute for Information and Communications Technology Planning and Evaluation (IITP) under Grant 2017-0-00294.

ABSTRACT Most studies on adaptive streaming over Hypertext Transport Protocol (HTTP) have focused on improving the quality of experience (QoE) of clients by running the rate adaptation algorithm on the client side. In a cellular environment, this leads to inefficient resource utilization because of the lack of coordination between the competing clients. In cellular networks, the key challenge for HTTP adaptive streaming (HAS) is to optimize the conflicting video quality objectives. Edge cloud-assisted adaptive streaming presents an opportunity to optimize the quality of experience in cellular networks by moving the adaptation intelligence from the client to the edge cloud. HAS algorithms select the video quality based on the estimated throughput and playback buffer level. In this paper, we first present a joint throughput estimation method for HAS by taking advantage of mobile edge computing. Next, we present an optimized solution for multi-access edge computing (MEC)-assisted HAS by using edge cloud capabilities. Due to the non-deterministic polynomial-time hardness of the problem, we design a heuristic rate adaptation algorithm to jointly enhance the quality metrics of the competing clients. Our extension simulation results show that the proposed edge cloud-assisted rate adaptation algorithm outperforms the existing strategies under different client-side and server-side settings. Furthermore, we show that the proposed algorithm is promising under slow-moving and fast-moving environments.

INDEX TERMS HTTP adaptive streaming, quality of experience, multi-access edge computing (MEC), quality adaptation algorithm, fairness.

I. INTRODUCTION

Multimedia contents account for a majority of the traffic over the Internet. According to *Cisco's* Visual Networking Index, the global mobile data traffic is expected to reach 82% by 2022 [1]. The most common solution for managing the traffic demands is to use Hypertext Transport Protocol (HTTP). HTTP adaptive streaming (HAS) solutions include Apple's HTTP Live Streaming, Adobe's HTTP Dynamic Streaming, Microsoft's ISS Smooth Streaming, and Dynamic Adaptive Streaming over HTTP (DASH) developed under MPEG and standardized by the International Organization for Standardization and International Electrotechnical Commission [2].

In HAS, the video content is fragmented into multiple segments and each segment is encoded into different video rates. The rate adaptive algorithm on the client side selects an appropriate segment that depends on the received metadata and system conditions, such as the throughput and the occupancy of the playback buffer. The rate adaptive algorithm attempts to maximize the quality of the video by meeting conflicting objectives in a manner that improves the user's viewing experience. The potential objectives include selecting the highest feasible set of video bit rates, avoiding needless video bit rate changes, assigning equitable video rates among the competing video clients, and preserving the buffer level to avoid any interruption in the playback [3]–[8].

In cellular networks, video streaming can be subject to low video quality and playback rebuffering because of bandwidth limitations or unstable networks. The rate adaptation algorithms strive to select the highest feasible video rate to maximize the bandwidth utilization. However, in an unstable network, it leads to higher frequency of vide bitrate switches

The associate editor coordinating the review of this manuscript and approving it for publication was Dharm Singh Jat.

and higher risk of playback interruption due to buffer underflow. The standard throughput estimation methods cannot accurately estimate the bandwidth fluctuations in the presence of competing video clients [8]–[10]; this is because the throughput per segment cannot estimate the bandwidth share. The client downloads the video as fast as possible to fill the playback buffer by selecting a video rate less than the available bandwidth. Once the buffer is full, the client enters the ON–OFF phase. During the OFF state, the client waits to have sufficient space in the buffer to download the next segment. The ON state means that the client requests the next segment. During the steady-state phase, when multiple streams compete for the network resources, the available bandwidth is estimated unfairly and incorrectly by HTTP clients unless downloading of the segment saturates the end-to-end bandwidth [8]. Moreover, because of the lack of coordination among multiple clients, the client-based video quality selection results in unfair allocation of the video quality.

Recently, the edge computing paradigm has been proposed as a promising approach to provide better performance compared to cloud computing [11]–[13]. Multi-Access Edge Computing (MEC) [11] brings computation and storage capabilities to the edge of the mobile network by deploying servers within the radio access network (RAN). The MECs have real-time access to the application and RAN information. The user experience could be enriched by transferring the video quality adaptation intelligence at the edge cloud. In a cellular network, the HTTP clients are oblivious of the bottlenecks in the radio channel and the competing clients. The MEC presents an opportunity to enhance the user experience by centrally adapting the video quality.

The understanding of edge computing-assisted rate adaptation strategies is still limited. In this paper, we present an edge computing-assisted rate adaptation method for a single cell with multiple clients. The contributions of this research are as follows:

- We present a joint throughput estimation method in mobile video streaming using edge computing facilities that assist the quality adaptation algorithm by fairly assigning video rates and reducing unnecessary quality fluctuations.
- We design an integer non-linear programming (INLP) optimization model that jointly optimizes the viewing experience of the competing clients in a cellular network with edge computing capabilities.
- We design a heuristic algorithm to efficiently solve the rate-selection optimization problem.
- Through extensive simulations, we prove that the proposed algorithm is promising under different client/server settings and client speeds. Our proposed algorithm outperforms the existing adaptation algorithms in terms of video quality and smoothness while protecting the playback buffer from draining. Furthermore, the proposed scheme efficiently uses network resources, and the competing streams achieve equitable video rates.

The rest of this paper is organized as follows. Section II reviews the existing video streaming algorithms. Section III presents the MEC-assisted HTTP adaptation system. Section IV explains the proposed throughput estimation method. The optimization problem is presented in Section V, and Section VI provides the details of the proposed heuristic rate adaptation algorithm. Section VII provides the simulation results. Finally, Section VIII concludes the paper.

II. RELATED WORK

A. THROUGHPUT ESTIMATION

The available throughput, T, is calculated by the client as the ratio of the segment size divided by the download time. The download time is computed from the instant when the HTTP request is sent to the instant when the last byte of the requested chunk is received. Based on the current and past observations, the throughput of the upcoming segments is estimated. Currently, several methods have been proposed to estimate the throughput. The running average of the throughput, T^E, is calculated as follows [14]:

$$T^E(i+1) = \begin{cases} \delta \times T(i) + (1-\delta) \times T^E(i) & i > 1 \\ T(i) & i = 1 \end{cases} \quad (1)$$

where the weight coefficient δ is bounded between 0 and 1. Dubin *et al.* [15] use the median of the throughput of the last several segments to estimate the throughput of the next segment. Rahman and Chung [16] show that the McGinely dynamic indicator offers a stable response to the throughput fluctuations, while maintaining a stable playback buffer. To estimate the throughput of the next segment, [17] and [18] use the harmonic mean of the throughput of the last 3 and the last 20 downloaded segments respectively. Conventionally, the client runs the estimation method to predict the throughput. In this paper, we propose a joint throughput estimation method based on HAS by taking advantage of mobile edge computing (MEC).

B. RATE ADAPTATION

Multiple rate adaptation algorithms for enhancing quality of experience (QoE) in DASH have been proposed. Traditionally, rate adaptive algorithms run at the client end to select the video quality. Certain researchers [19] and [20] have proposed rate adaptation algorithms that select video rates based only on the estimated throughput. Many methods have been proposed to incorporate the information concerning the playback buffer for selecting the video rate [21]–[23]. The algorithms divide the playback buffer into multiple predefined thresholds: B_1, B_2, B_3, B_{max} such that $B_1 < B_2 < B_3 < B_{max}$. The algorithms increase or decrease the video rate aggressively or conservatively based on whether the buffer level increases above the next higher buffer threshold or decreases below the next lower buffer threshold, respectively. In [16] and [24], the authors propose algorithms

that consider the size of the upcoming segments in addition to the throughput and the buffer occupancy to predict the segment download time.

Recently, several studies have focused on the rate adaptation algorithms by considering multi-client scenarios. Zhao *et al.* [25] focus on maximizing the video quality subject to the stability of the servers' queues. The problem of optimal cache resource allocation for streaming over mobile networks is investigated by [26]. Their research focuses on energy-efficient caching along with QoE optimization. However, it does not consider the constraints imposed by the limited amount of radio resources.

A recent addition to the DASH [2] standard is the server and network-assisted DASH (SAND-DASH) [27], which specifies the means for the clients, the servers, and the DASH-aware network elements to collaborate with one another. Li *et al.* [28] propose the maximization of the utility function to improve the QoE of the clients over a cellular network using the SAND-DASH collaboration mechanism. Yan *et al.* [29] present Prius, a hybrid-edge cloud and client-rate adaptation framework that adds a layer of intelligence to the edge cloud for jointly optimizing the rate adaptation in a cellular network. Mehrabi *et al.* [30] design an INLP problem to jointly optimize the QoE while fairly allocating bit rates among the clients and balancing the utilized resources among multiple edge servers.

III. MULTI-ACCESS EDGE COMPUTING ASSISTED HAS

Client-side rate adaptation algorithms have been widely implemented in modern streaming systems. The cellular links are highly dynamic, and the underlying TCP is unfair; therefore, it is unlikely that a single client will accurately capture the bandwidth share. The throughput information of multiple clients can be collected intuitively; the throughput can be jointly estimated, and the video bit rates can be selected.

One possible solution is to shift the intelligent adaptation from the client to the base station. This allows the central controller to jointly optimize the video rate selection. The MEC computational capabilities and storage support allows the joint adaptation of multiple clients. However, the competing clients might not be able to support all the available bit rates because of limited display resolution and power support.

We follow the MEC system architecture proposed in [29]. Fig. 1 illustrates the edge computing–assisted HAS system for adaptive video streaming over a cellular network. The HTTP server stores the video data of different views encoded into multiple quality levels. The HTTP client downloads the video stream divided into segments of fixed duration. All the segments were of equal duration and were given in seconds by the time constant τ. HAS operates by monitoring a network in real time and adjusting the quality of the video stream accordingly without resetting the TCP connection. The edge servers were deployed by the service provider within the RAN adjacent to the base stations to improve the quality of the mobile services.

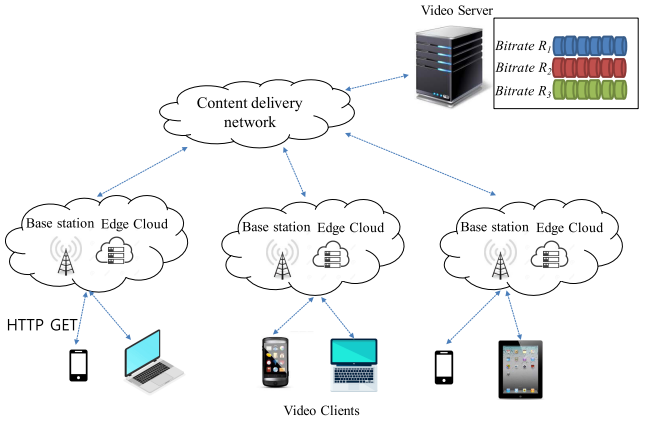

FIGURE 1. Streaming architecture for multi-access edge computing assisted video streaming.

The edge cloud can access the RAN information available in the base station and is computationally powerful. The adaptation module at the edge cloud can access the channel knowledge of multiple clients for joint adaptation. The HAS server shares the MPD w with the edge cloud and the clients so that they have knowledge of the video representations stored at the server. The clients first select a video rate R_c based on constraints such as the display, the buffer level, and the battery level. Then, the client requests the segment at R_c. In conventional client-side adaptation, the cellular network forwards the client request to the server. Unlike the conventional client-side video rate adaptation, the edge could intercept the request from the client. The adaptation algorithm at the edge cloud overwrites the client's request based on the joint optimization of the clients and R_c. The client periodically shares playback information including the buffer level and QoE status. The 3rd Generation Partnership Project standardized the QoE reporting process for the clients by using the HTTP POST request carrying the XML formatted metadata [31]. The rate adaptation results are then sent to the server for streaming the segments. This does not require any modifications in the video server.

A. QUALITY OF EXPERIENCE AND FAIRNESS

Some of the potential video quality objectives include selecting the highest feasible set of video bit rates, avoiding needless video bit rate changes, and preserving the buffer level underflow to avoid interrupting the playback. The video bit rate and playback interruption because of the buffer underflow affects the user experience the most [32]. Moreover, one long playback interruption is preferred to multiple short ones [33]. However, there is a tradeoff between selecting a high video rate and the risk of playback interruption. The high video rate improves the user experience but increases the risk of playback interruption. The average video rate of the k^{th} client, Q_k, can be obtained using:

$$Q_k = \frac{\sum_{n=1}^{S} r_{ik}(n)}{S} \tag{2}$$

where r_{ik} is the i^{th} video rate assigned to the k^{th} client, n is the segment index and S is the number of segments downloaded by the client. The stall event plays a critical role in determining the QoE; therefore, we sacrificed the video rate to mitigate the risk of playback interruption. In the paper, we designed the optimization problem that ensures that the sum of the selected video rates of the competing client is less than the available resources at the base station.

Frequent bit rate switches inversely affect the QoE. Unlike the case for smooth switching, the QoE for down switching is impaired by abrupt switching [7]. For an instantaneous switch, the minimum quality level is on average 30% better than that of gradual video rate switches [34]. Experiments have shown that there was little noticeable difference between the qualities of the high-quality and mid-quality switches during video playback [34]. In [35], the researchers suggest that the user experience improves when the video rate was increased aggressively because the increase makes the users believe that the provider was attempting to maximize their QoE. In addition, a long spell of good quality videos improves the user experience [36]. The bit rate switching metric of the client k is given by:

$$QS_k = \frac{\sum_{n=1}^{S} r_{ik}(n) - r_{ik}(n-1)}{S} \tag{3}$$

Multiple clients competing at the bottleneck must be able to achieve equitable video rates. LTE base stations use the proportional fairness policy to allocate radio resources to multiple competing clients [37]. Traditionally, each client selects the video rate according to the observed throughput. However, because of the lack of coordination between the competing clients, the client-based rate adaptation heuristics may unfairly allocate the video rates among the clients. As part of our MEC-assisted optimized solution, we obtain fairness using the following relationship:

$$F_k = \frac{\sum_{k=1}^{C} r_{ik} - R_{avg}}{C} \tag{4}$$

where R_{avg} is the average of the bit rates of the other active competing clients. For each competing client, the minimization of F_k should satisfy the available resource blocks at the base station.

IV. THROUGHPUT ESTIMATION METHOD

The conventional quality adaptation algorithms run at the client end and select the video rate based on the knowledge of their estimated throughput. MECs are computationally powerful and can access the application layer information; therefore, the quality of the clients can be jointly adapted by overlaying a layer of intelligence on top of the clients. The proposed estimation method collects the information of multiple clients at the MEC and jointly estimates the throughput of the clients for the upcoming segment. As explained earlier, when multiple streams compete for the bandwidth, clients estimate the segment throughput inaccurately. The proposed scheme keeps monitoring the throughput of the n^{th} segment $T_k(n)$ of the k^{th} client and compares it with the estimated throughput of the $(n+1)^{th}$ segment. If the throughput is inaccurately estimated, the propose scheme continuously modifies the estimated throughput according to Eq. (5):

$$T_k^E(n+1) = \begin{cases} T_k(n) & ^n = 1 \\ T_k^R(n) + e \times \mu & n > 1 \end{cases} \tag{5}$$

where e and μ are given by:

$$e = abs\,(T_k^R(n) - T_k(n)) \tag{6}$$

$$\mu = \log(\frac{T_k^R(n) - T_k(n)}{T_k(n)})\, where\ -1 \leq \mu \leq 1 \tag{7}$$

Eq. (6) gives the difference between $T_k^{RR}(n)$ and the observed throughput. Eq. (7) shows the extent of variability in $T_k^R(n)$ in relation with the observed throughput. A high value of μ indicates that there is a significant difference between $T_k(n)$ and $T_k^{RR}(n)$ because of throughput fluctuations or the inaccurate estimation. The client must be sensitive to the large differences between the estimated and the observed throughputs. We select video bit rate based on the estimated throughput; therefore, an inaccurate estimation would lead to a low video quality or an increase in the risk of playback interruptions. If the value of μ is large, Eq. (5) aggressively adjusts the estimated throughput. However, a small μ value requires a small adjustment to the estimated throughput. In a cellular network, the channel conditions can vary significantly during the download of segments. Therefore, it is reasonable to leverage both the throughput history and the current throughput to interpret the trend of the throughput fluctuations. For this, we use a simple exponential smoothing function to capture the channel characteristics as:

$$T_k^R(n+1) = \begin{cases} T^E(n) & n=1 \\ (1-\alpha) \times T_k^E(n+1) + \alpha \times T_k^R(n) & n>1 \end{cases} \tag{8}$$

where α $(0 < \alpha \leq 1)$ is the smoothing factor. The value of α is set to 0.8 to remove the outliers from the dataset. The MEC has the information of the estimated throughput of all the clients. Therefore, the throughput is jointly estimated. The HTTP streams compete for the bandwidth resource at the bottleneck. Fig. 2 shows the example of unfair throughput estimation because of the ON–OFF scheduling pattern. As explained above, after the playback buffer is full, the client enters the ON–OFF scheduling pattern. Let us suppose that the available throughput is 3 Mbps. As shown in Fig. 2, all three clients are downloading from time 0 to time t. Each client observes a throughput of 1 Mbps over the download of the segments. In the next downloading cycle, Client 3 enters the OFF state. Clients 1 and 2 overestimate the throughput. Similarly, in the next downloading cycle, Clients 1 and 2 enter the OFF state. Client 3 overestimates the throughput. The ON–OFF scheduling pattern not only results in unfair and inaccurate throughputs, but it also leads to unnecessary fluctuations in the observed throughput. The clients consider

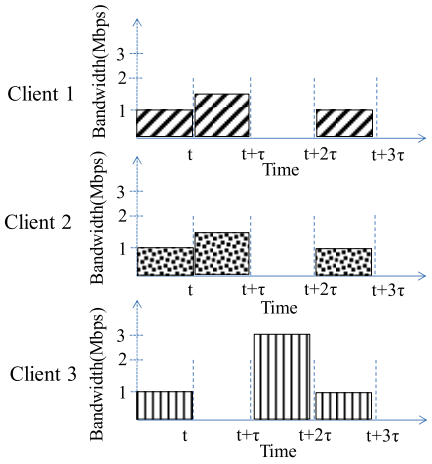

FIGURE 2. Example of unfair throughput prediction resulting from the ON–OFF scheduling pattern.

the throughput for selecting the video bit rate; this leads to unfair throughput sharing and unnecessary video bit rate fluctuations.

Our proposed scheme jointly estimates the throughput of the competing streams. First, the client calculates the average throughput of the competing clients as follows:

$$T^{sum} = \sum_{k=1}^{C} T_k^R \tag{9}$$

$$T^{avg} = \frac{T^{sum}}{C} \tag{10}$$

where C is the number of competing clients. However, in a cellular environment, the throughput is also affected by factors such as the path loss, fading, and shadowing. Therefore, a client that is closer to the base station would experience a higher throughput as compared with the user at the edge of the cell. Therefore, the throughput for the next segment is estimated as follows:

$$\overline{T}_k(n+1) = T_{avg} \times (1 + \frac{T_k^R(n+1) - T_{avg}}{T_{avg}}) \tag{11}$$

V. RATE ADAPTATION PROBLEM

For the rate adaptation problem, a video is fragmented into τ second segments. The set of representations available for the video stream is denoted by R where $R = \{r_1, r_2, r_3, \ldots, R_N\}$. The problem is to select the bit rate $r_i \epsilon\ R$, for all clients, such that the QoE is maximized. The variable x_{ik} defines the number of clients streaming the i^{th} video rate stored at the server. In the next section, we provide a detailed description of the joint optimization problem and the fairness of QoE. We define three weighing parameters $0 \leq \delta, \beta, \varphi \leq 1$ $(\delta + \beta + \varphi = 1)$ to control the respective video rates, the video rate switches, and the fairness. In addition, we add a problem constraint to avoid the playback buffer underflow. The descriptions of the parameters involved in the system have been summarized in Table 1.

TABLE 1. Description of parameters.

Notation	Description
r_{ik}	Discrete video bit rate from the i^{th} segment of the k^{th} client
R	Discrete set of available bit rates at the server
R_{avg}	Average of video rates of the competing clients
R_{next}	Video rate selected for the next segment
R_{prev}	Video rate selected for the previous segment
R_c	Video rate suggested by the client
B_k	Playback buffer occupancy of the k^{th} client
δ_S	Switching threshold
δ_F	Fairness threshold
C	Number of clients
S	Number of Segments
N	Available video rates
$\overline{T}_k$	Estimated throughput for the k^{th} client
Q_k	Average video bit rate for the k^{th} client
QS_k	Accumulated bit rate switching for the k^{th} client
F_k	Fairness value by allocating bit rates to the k^{th} client
W	Available resource blocks on the base station
W_k	Resource blocks assigned to the k^{th} client by the base station

A. PROBLEM FORMULATION

We model the joint optimization for each client i with the following INLP formulation.

$$\text{Maximize } U_k = \delta^* Q_{k-} \beta^* QS_{k-\varphi}{}^* F_k \tag{12}$$

$$\text{Subject to } \sum_{k=1}^{C} x_{ik} \times W_k \leq W \tag{13}$$

$$0 \leq x_{ik} \tag{14}$$

$$0 \leq B_k \leq B_{max} \tag{15}$$

$$r_{ik} \leq R_c, \quad \forall 1 \leq k \leq S \tag{16}$$

$$r_{ik} \in R, \quad \nabla \leq k \leq S \tag{17}$$

The objective function (12) aims to jointly maximize the user experience of the k^{th} client and equitably allocate video rates among the clients. The constraint (13) ensures that the total resources distributed among the clients by the base station do not exceed the available resources at the base station. W denotes the available resources blocks at the base station. The scheduler is responsible for resource scheduling and resource allocation to the video clients. It is important to note that in our network model, the observed throughput would be affected by the presence of competing clients. The constraint (14) specifies that a video rate can be allocated to multiple clients. The constraint (15) ensures that the client does not experience buffer underflow. Constraint (16) guarantees that the video rate selected for the k^{th} client at the MEC does not exceed R_c. Finally, the constraint (17) specifies that the bit rates allocated to the clients belong to the set of video rates available at the server.

B. NON-DETERMINISTIC POLYNOMIAL-TIME HARDNESS

The optimization problem formulated in the previous section belongs to the non-deterministic polynomial-time (NP)–hard

problems because of the existence of integer decision variables. Hence, the solution space includes exhaustive possible enumerations.

First, we prove the NP-hardness of the optimization problem by reduction from the unbounded variant of the 0–1 knapsack problem (KP) [38]. The 0–1 KP has been shown to be NP-complete. The unbounded variant of KP exhibits the same hardness as the 0–1 KP [38]. Given a set of k items $\{c_1, c_2, \ldots, c_k\}$ with associated weights and profit and a knapsack, we consider an instance of KP in which multiple copies of each item are available. The problem is the allocation of the items to the knapsack to maximize the total profit while the total weight of the items is kept less than the knapsack capacity. The unbounded variant of the KP does not put any limits on the number of the times that an item may be selected.

The unbounded variant of the 0–1 KP can be mapped to our problem. The set of k items and the knapsack are mapped to the set of available video rates at the video server and base station, respectively. The profit maximization is equivalent to the utility maximization (Eq. 12) in our problem. The objective problem given in Eq. (12) is an instance of the KP in which the number of video rates allocated by the base station is equal to the number of clients associated with the base station. The problem is the allocation of the video rates by the base station to the clients to maximize the total profit; the total resources distributed among the clients by the base station do not exceed the available resources at the base station. The joint allocation of different set of video rates by the base station to the clients results in different utility values. Furthermore, multiple copies of a discrete video rate can be assigned to multiple clients. Now, the unbounded KP is an NP-complete problem; therefore, our problem is also NP-complete.

VI. ONLINE OPTIMIZATION ALGORITHM

In this section, we explain the heuristic adaptation algorithm to solve the optimization problem presented in the previous section. The algorithm is designed for online execution at the MEC. A brute-force strategy can be employed to maximize utility by investigating all the possibilities for allocating video bit rates to the clients. However, when the number of available video rates and clients increases, the complexity will also grow exponentially. To reduce the complexity, we present a greedy-based algorithm, which is executed using the client data obtained for the MEC.

As explained in Section III, the clients first select the video rate R_c. The different video clients support different playback qualities. The proposed greedy algorithm runs in the MEC; therefore, this algorithm is unaware of the client's capabilities. Therefore, the client shares with the MEC the highest video rate that it can play back based on the display and the buffer level. Similarly, it is important to avoid the buffer underflow. The highest video rate supported by the client is denoted by R_{sup}. Given the observed throughput, T_k and current buffer level, B_k, we set R_c equal to the highest video rate that satisfies the condition: $max(B_k - r_{ik} / T_k \times \tau > 0.1 \times B_{max}, R_{sup})$.

Algorithm 1 Greedy Algorithm

r_{ik}: The i^{th} video bitrate of the k^{th} client
R_{avg}: Average of video rates of clients
R_{next}: The video rate selected for the next segment
R_{prev}: The video rate selected for the previous segment
R_c: Video rate suggested by the client
B_k: Playback buffer occupancy of the k^{th} client
δ_S: Switching threshold
δ_F: Fairness threshold
C: Number of Clients
S: Number of Segments
N: Available video rates
Initialization:
For each client $1 \leq i \leq$ C
  If *Buffer Level* == *Empty*
    If Segment Number == 1
      $R_{next} = R_{min}$
    Else
      For each bitrate $r \epsilon R$ in the decreasing order
        If $r_{ik} \leq T_k$
          $R_{next} = r$
          Break
  Else
    For each bitrate $r \epsilon R$ in the decreasing order
      If allocation of r satisfies (13) *and* $r < R_c$ *and* $r < \bar{T}_k$
        If bitrate $r > R_{prev}$
          If $1 - \frac{r - r_{avg}}{R_{\max} - R_{\min}} > \delta_F$
            Break
        Else If bitrate $r == R_{prev}$
          If $R_{prev} < r_{avg}$
            $R_{next} = R_{prev}$
            Break
        Else
          If $|r - R_{prev}| \leq \delta_S$ *and* $R_{prev} \geq r_{avg}$
            $R_{next} = r$
            Break
  If $R_{next} == 0$ **then**
    For each bitrate $r \epsilon R$ in the decreasing order
      If allocation of r satisfies (13) and $r < R_c$
        If $r_{ik} \leq \bar{T}_k$
          $R_{next} = r$
          Break
Compute Video Quality, Switching, Fairness and Utility Function according to (2), (3), (4) and (12)
Update Buffer Level

Algorithm 1 provides the details of the proposed rate adaptation algorithm. The algorithm selects the i^{th} video rate from the set R for the n^{th} segment. The video rate for the n^{th} segment is selected at the end of the download of the segment $n-1$. The video rate selected for the segment $n-1$

is denoted by R_{prev}. The proposed algorithm running on the client outputs the video rate for the n^{th} segment as R_{next}.

At the start of the streaming session, when the playback buffer is empty and no information was available about the link quality, we select the lowest available video rate for the first segment. For a buffer underflow, while downloading the subsequent segments, we select the highest available video rate less than the observed throughput.

When the buffer level fills up, the algorithm selects the video rate that reduces the switching frequency and improves the fairness value. To select the video, two conditions should be satisfied (1) the resources distributed among the clients should not exceed the total resources available at the base station, (2) the video rate should be less than R_c. These conditions are put in place to minimize the risk of playback interruption.

Here, we consider the scenario in which the video rate is increased. As explained in Section III-A, the aggressive increase in the video rate improves the user experience [35]. Although, the users found the frequent video rate switches annoying, they positively responded to the improvements in the video quality. Therefore, it is important to ensure that the increase in the video rate is not followed by a sudden decrease. The decrease in the video rate is forced by the risk of playback interruption. Therefore, the video rate is selected only if it satisfies the two conditions explained earlier. To increase the video rate, only the fairness condition should be satisfied. The proposed algorithm considers the threshold δ_F of fairness. The fairness index associated with the selected bit rate is computed as $1-(r-r_{avg})/(R_{max}-R_{min})$, which takes the value between 0 and 1. To increase the video rate, the selected video rate should satisfy the condition: Fairness index $> \delta_F$.

Next, we discuss the scenario when the current video rate is maintained. If the highest available video rate that satisfies the video selection conditions is R_{prev}, then we use the current video rate only if the current video rate is less than r_{avg}. This is because we want to improve the fairness and minimize the frequency of the video rate switches. By further decreasing the video rate, the fairness decreases, and the number of switches increases.

Finally, we discuss the scenario when the video rate is decreased. If the highest available rate that satisfies the video selection conditions is less than R_{prev}, the selected video rate should satisfy the following two conditions: (1) $R_{prev} > r_{avg}$, (2) Switching index $\leq \delta_S$. If the current video rate is greater than r_{avg}, we improve the fairness value by decreasing the selected video rate. The algorithm considers the threshold δ_S for the video rate switches. The switching threshold δ_S is computed as $|max\{R\} < T_{n-1} - \max\{R\} < T_n|$ where $max\{R\}$ is the highest video rate in the set R that is less than T_n. The switching index associated with the selected bit rate is computed as $|r - R_{prev}|$.

If there is no such bit rate that satisfies the conditions to increase or decrease the video rate, the algorithm selects the highest video rate less than the estimated throughput that satisfies both the video selection conditions. After the bit rate selection, the algorithm returns the local utility of the client, which was computed using (12).

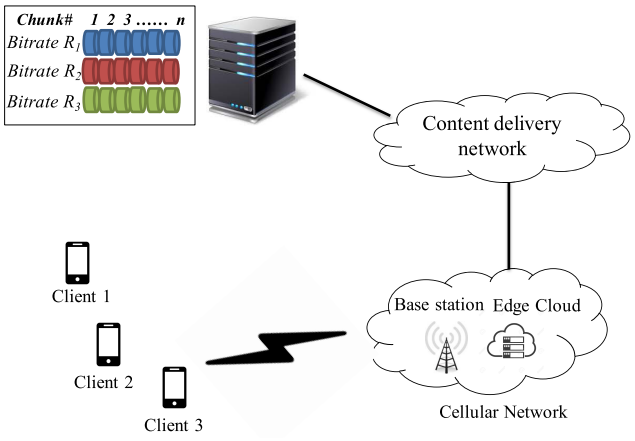

FIGURE 3. Network topology.

TABLE 2. Cellular network configuration.

Cell Layout	Single hexagonal cell
UE distribution	Uniform
Path loss model	Hata Model PCS Extension
BS transmission power	38 dBm
UE distance	250 ~ 500 m
Scheduler	Proportional fairness
UE speed	3 km/h and 60 km/h

VII. PERFORMANCE EVALUATION

We implement HTTP-based adaptive video streaming in the multi-access edge computing scenario shown in Fig. 3. We implement the LTE network as the underlying cellular network. A detailed configuration of the cellular network is shown in Table 2. To achieve adaptive streaming, the HTTP server offers the client 10 levels of representation to adapt the video rates. These video rates are 184, 380, 760, 1000, 1600, 2100, 2500, 3000, 3500 and 4000 kbps. We assume that all the clients can playback the highest available video rate. We set the fairness threshold δ_F equal to 0.7. The tuning parameters in the objective function in (12) are set to $\delta = 0.75$, $\beta = 0.25$ and $\varphi = 0.25$.

In the simulation, we evaluate the algorithms under varying network conditions, buffer sizes, and segment durations. We analyze the algorithms for the settings mentioned in Table 3. We demonstrate the impact of the buffer size and the segment duration on the performance of the algorithms. The HTTP clients offer distinct buffer sizes. The rate adaptation algorithms should be able to guarantee QoE under different client settings. We set the buffer sizes to 20, 40, and 60 s and evaluate the performance of the rate adaptation algorithms. Then, we demonstrate how the algorithms perform with variations in the segment duration. Microsoft Smooth Streaming and Adobe HTTP Dynamic Streaming

TABLE 3. Buffer size and segment duration settings.

No.	Buffer size (s)	Segment duration (s)
1	60	2
2	40	2
3	20	2
4	60	4

offer segment durations of 2 and 4 s [39], [40]. The users are randomly distributed within the cell. The users are moving at a speed of 60 kmph. The simulation runs 200 s. The experiment is repeated five times for each setting.

We adopt the solutions proposed in [24] and [30] as the benchmarks to demonstrate the efficiency of the proposal. In the results, we refer to the algorithms proposed in [24] and [30] as ECAA and SARA, respectively. The ECAA is an MEC-assisted DASH system for mobile video streaming that includes a client-to-edge-server mapping strategy and an optimization of QoE and fairness. We adopt the solution proposed in [30] for a single-cell scenario such that the MEC allocates the video rates to the clients. Unlike the MEC-assisted adaptation algorithm, SARA is a client-based adaptation algorithm. The clients select the video rate independently and are oblivious of the decisions taken by other clients.

Along with the factors that affect the user experience, fairness and inefficiency are also important factors that the algorithm must strive to achieve. The inefficiency at the time t is given as follows [17]:

$$Inefficiency = \frac{\left|\sum_{i=1}^{C} r_i - W\right|}{W} \tag{18}$$

where W is the bandwidth, and each client i selects the bit rate r_i. The unfairness is computed according to $\sqrt{1 - JainFair}$, where $JainFair$ is the Jain fairness index calculated using Eq. (19). We used the Jain fairness index to quantify the fairness [41]. The Jain fairness index of r_i over all the players i is given by

$$Fairness = \frac{\left(\sum_{i=1}^{C} r_i\right)^2}{C \sum_{i-1}^{C} r_i^2}, \quad i \geq 0 \tag{19}$$

Ideally, the values of inefficiency and unfairness should be zero. Low inefficiency values are desirable because it would mean that the client selects the highest feasible bit rates that are lower than the actual throughput. Low unfairness values are also desirable because the competing clients would achieve equitable video rates.

A. EVALUATION OF THROUGHPUT ESTIMATION METHOD

In this section, we show how the proposed throughput estimation method helps the rate adaptation algorithm to improve the user experience. To evaluate the performance of the proposed scheme, we compare our results with the two most commonly used throughput estimation methods: running average [3] and moving average of the last three segments [4]. The video bit rate was determined by selecting the highest video rate less than the estimated throughput.

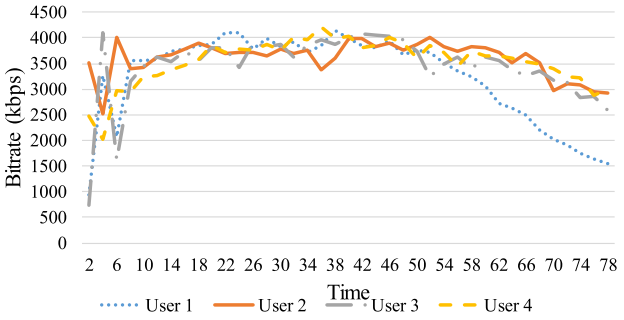

FIGURE 4. Throughput observed by the competing clients.

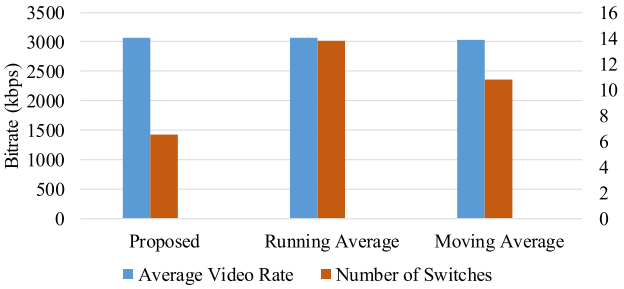

FIGURE 5. Average video rates selected and average video rate switches experienced by the clients.

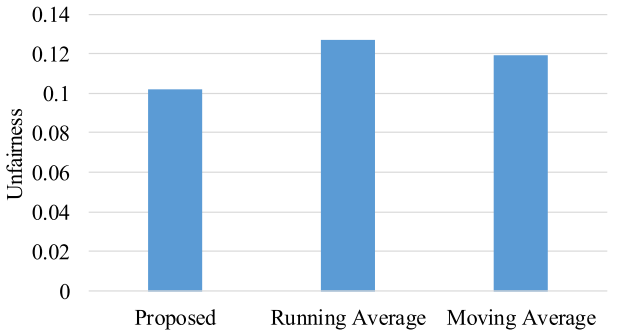

FIGURE 6. Comparison of the unfairness values when the clients employ different throughput estimation methods.

The throughput observed by the competing clients is depicted in Fig. 4. Figs. 5 and 6 show the performance parameters when the clients use the proposed method, the running average, and the moving average throughput estimation methods. Fig. 5 shows that all the compared throughput estimation methods achieve a similar video quality, but the proposed algorithm is able to minimize the number of video rate switches. This shows that the proposed method is robust to small variations in the throughput. To demonstrate this, we perform an experiment in which the client experiences small fluctuations followed by a large fluctuation in the throughput. Fig. 7 shows that the proposed method during

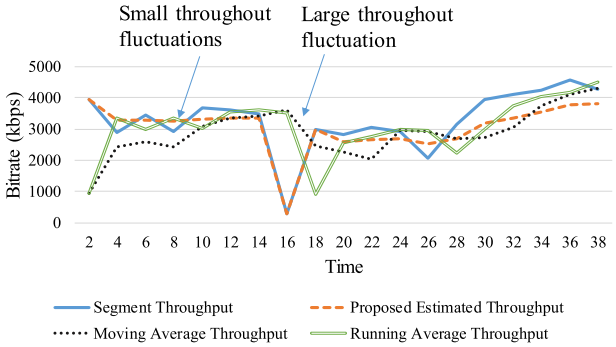

FIGURE 7. Throughput values estimated by the proposed, running and moving average methods.

this period provides a stable response to small fluctuations. This helps to minimize the video quality fluctuations by mitigating unnecessary fluctuations. Fig. 7 also shows that our proposed method is sensitive to large variations in the throughput. Our proposed method also responds quickly to large variations in the throughput. The rate adaptation algorithms select the video quality on the basis of the estimated throughput; therefore, our proposed method helps to reduce the risk of playback interruptions and improves the use of the available throughput. The running average method has the highest number of switches followed by the moving average method. Fig. 6 also shows that the proposed method has the lowest unfairness value, which means that the proposed method gives a fair estimate of the throughput for the competing users.

B. EVALUATION OF QoE AND FAIRNESS

In this section, we examine the performance of the algorithms in terms of QoE, fairness, and inefficiency. To evaluate the performance of the proposed algorithm, we compare the performances of the three algorithms under different client and server settings.

TABLE 4. Statistics of different adaptive methods when buffer size is set to 60 s.

10 Clients			
Buffer Size	60 s		
Schemes	Proposed method	SARA	ECAA
Average Video Rate (kbps)	1040.14	993.47	1067.05
Number of Switches	26.40	58.40	32.70
Fairness	0.86	0.84	0.82
Inefficiency	0.12	0.17	0.12

1) EFFECT OF BUFFER SIZE

First, we evaluate the performance of the algorithms under varying buffer sizes of the clients. Table 4 shows the statistics of the adaptation algorithms when the buffer size of the clients is set to 60 s. Table 4 shows that the proposed and ECAA algorithms outperform the SARA algorithm in terms of the average video rates. The proposed algorithm experiences the least number of video rate switches as compared with the competing algorithms. The proposed algorithm not only achieves high video rate but is also able to mitigate unnecessary video rate switches. The SARA algorithm achieves a high video rate at the expense of high number of video rate switches. Here, we can observe a tradeoff between selecting high video quality and minimizing the number of video rate switches. The SARA algorithm runs only on the client side. Therefore, a greedy client increases the video rate whenever it gets the opportunity but at the expense of higher video rate switches. Table 4 shows that the proposed algorithm has the highest fairness value followed by the SARA algorithm. The SARA algorithm has a high fairness value because all the clients start streaming simultaneously and are moving at a constant speed. Therefore, their throughput fluctuations are synchronized. This helps the SARA algorithm to achieve a high fairness value although the clients are oblivious of one another. The proposed and ECAA algorithms achieve a lower inefficiency value as compared with the SARA algorithm.

TABLE 5. Statistics of different adaptive methods when buffer size is set to 40 s.

10 Clients			
Buffer Size	40 s		
Schemes	Proposed method	SARA	ECAA
Average Video Rate (kbps)	1161.27	844.98	1009.61
Number of Switches	26.0	51.6	31.4
Fairness	0.86	0.86	0.82
Inefficiency	0.13	0.13	0.13

Next, we decrease the video rate to 40 s. Table 5 shows the statistics of the adaptation algorithms the buffer size of the clients is set to 40 s. A smaller buffer size increases the risk of playback interruption. The rate adaptation algorithm should react aggressively to avoid playback interruptions. Table 5 shows that the proposed algorithm outperforms other algorithms in terms of the average video rate achieved by the clients. The proposed algorithm achieves the highest video rate and experiences the lowest number of video rate switches. The ECAA algorithm is also able to reduce the video rate switches and maintain a high video rate. The SARA algorithm achieves the lowest video rate and experiences the highest number of video rate switches. Similar to the previous scenario, the proposed algorithm and the SARA algorithm achieve a high fairness value followed by the ECAA algorithm. Furthermore, all the competing algorithms achieve the same inefficiency value.

In the next experiment, we decrease the buffer size to 20 s. Table 6 shows the statistics of the adaptation algorithms the buffer size of the clients is set to 20 s. Table 6 shows that the proposed algorithm achieves the highest average video rate and is able to minimize the number of video

TABLE 6. Statistics of different adaptive methods when buffer size is set to 20 s.

10 Clients			
Buffer Size	20 s		
Schemes	Proposed method	SARA	ECAA
Average Video Rate (kbps)	1146.20	1028.44	1055.82
Number of Switches	24.6	58.4	40.4
Fairness	0.87	0.87	0.77
Inefficiency	0.12	0.15	0.12

rate switches. Our proposed algorithm also achieves high fairness value and efficiently uses the bandwidth. The ECAA algorithm achieves a higher video rate and a smaller number of switches than our proposed algorithm, but ECCA has the lowest fairness value among the competing clients. However, the SARA algorithm experiences the highest number of video rate switches. The video rate switches are high because the algorithm tries to bring about a balance between achieving a high video rate and minimizing the risk of playback interruption. The algorithm increases the video rate as soon as the bandwidth allows it to do so. However, the client moves at a high speed; therefore, a large number of fluctuations are experienced in the bandwidth. The client aggressively adjusts the video rate in response to the fluctuating bandwidth to reduce the risk of buffer underflow. Table 6 also shows that the proposed and the SARA algorithms have a marginally lower inefficiency value as compared with the SARA algorithm when the segment size was set to 20 s.

These results show that our proposed algorithm is able to select high video rates for the clients while reducing the number of video rate switches. Moreover, the video rates are selected fairly, and the clients efficiently use the resources. The ECAA algorithm can achieve a high video rate; however, as compared with the proposed algorithm, it has a higher number of video rate switches and a lower fairness value. The SARA algorithm achieves the lowest video rate among the competing clients and the highest number of video rate switches.

2) EFFECT OF SEGMENT DURATION

Here, we evaluate the performance of the algorithms under different segment durations. The different service providers provide videos that are divided into different segment durations. We set the value of the buffer size to 60 s. In the previous section, we explained the performance of the algorithms when the segment size is set to 2 s. Figs. 8 and 9 show a comparison of the performance of the algorithms for segment durations of 2 and 4 s, respectively. When the segment duration is increased to 4 s, the proposed algorithm and the SARA algorithm marginally improve their performances, whereas the performance of the ECAA algorithm is marginally degrades. The proposed algorithm achieves higher video rate and the number of video rate switches decrease.

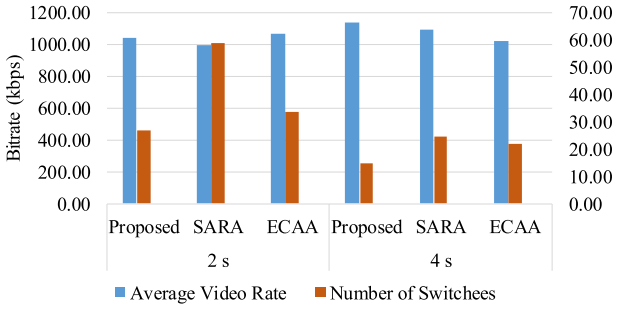

FIGURE 8. Average video rate and video rate switches experienced when the segment duration is set to 2 and 4 s.

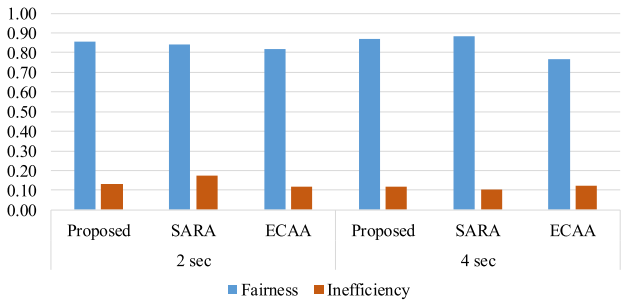

FIGURE 9. Fairness and inefficiency values achieved by the algorithms when the segment duration is set to 2 and 4 s.

Fig. 8 shows that the SARA algorithm experiences a video rate switch during the streaming of almost half the segments. The ECAA algorithm not only achieves a slightly lesser video rate, but the frequency of the video rate switches also increases. It is important to note here that when the segment duration is set to 4 s, the client downloads 50 segments as compared with the experiment when the segment duration is set to 2 s, and 100 segments are downloaded.

Fig. 9 shows that the proposed and SARA algorithms achieve similar fairness values, whereas the ECAA algorithm has a slightly lower fairness value when the segment duration is increased to 4 s. Fig. 9 shows that the SARA algorithm uses resources more efficiently when the segment duration is increased to 4 s. A closer look at Table 6 shows that when the buffer size is reduced to 20 s, the fairness value of the ECAA algorithm is low as compared with other experiments. Similarly, when the segment duration is increased, the ECAA algorithm shows a drop in the fairness value. One reason for this drop is that the fairness condition of the ECAA algorithm is not stringent. Secondly, if none of the available video rates satisfies the fairness condition, the ECAA algorithm selects the video rate based only on the basis of the switching condition. The fairness and efficiency values of the proposed algorithm remain similar regardless of the buffer size and segment duration, whereas for the SARA and ECAA algorithms, these values vary when the segment duration or buffer size is changed. The SARA algorithm has a slightly lower inefficiency value when the segment duration is increased. The segment duration is directly proportional to the risk of the

buffer underflow if the selected video rate is higher than the throughput. When the segment duration is small, the SARA algorithm increases the video rates aggressively and takes the risk of selecting the video rate higher than the available throughput. This leads to an increase in the inefficiency value. When the segment duration is large, the SARA algorithm increases the video rate conservatively because a slight mismatch between the video rate and the throughput could lead to a buffer underflow.

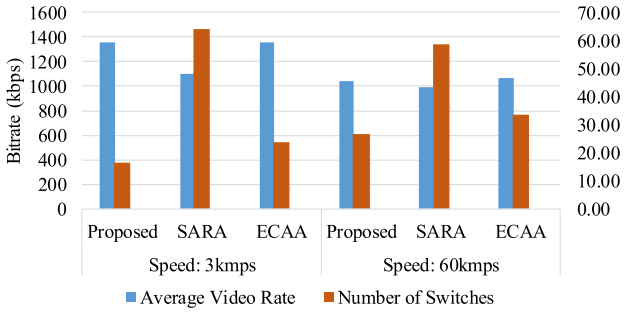

FIGURE 10. Average video rate and video rate switches experienced when the clients speed is set to 3 and 60 kmph.

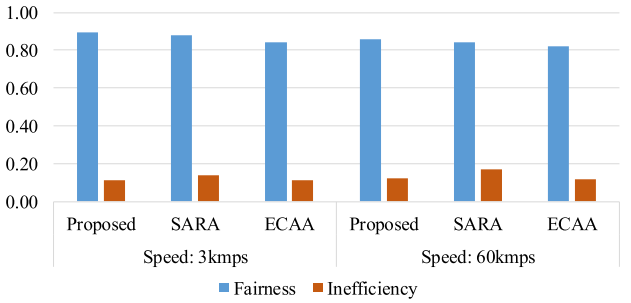

FIGURE 11. Average video rate and video rate switches experienced when the client speeds are set to 3 and 60 kmph.

3) EFFECT OF MOBILITY

In this section, we evaluate the performance of the algorithms when the clients move at the speed of a pedestrian and a vehicle. We first set the client speed to 3 kmph and then increase the speed to 60 kmph. Figs. 10 and 11 show the performances of the algorithms under different client speeds. We set the buffer size and segment duration to 60 s and 2 s, respectively. Fig. 10 shows that the proposed and ECAA algorithms show a significant increase in the video rate for low client speeds. There was a marginal increase in the clients' average video rate for the SARA algorithm. Fig. 10 shows that the proposed algorithm has the lowest number of video rate switches for both experiments; the next lowest number is given by the ECAA algorithm. The proposed and ECAA algorithms have fewer switches when the client moves at the pedestrian's speed than at the speed of the vehicle. Surprisingly, the SARA algorithm experiences more switches at the speed of a pedestrian. SARA is a greedy algorithm, and it runs on the client side; therefore, the tug-of-war between the clients to achieve a high video rate leads to higher video rate switches regardless of the client speeds. Fig. 11 shows that the proposed algorithm has the highest fairness value for the speeds of both a pedestrian and a vehicle. The ECAA algorithm has the lowest fairness value. The SARA algorithm achieves the highest inefficiency value in both the experiments. One reason for the high inefficiency value is that the SARA algorithm slowly increases the video rate at the start of the streaming session. If the available bandwidth is high, this leads to inefficient utilization of the algorithm. However, the ECAA algorithm selects the highest available video rate for the first segment. If the available bandwidth does not allow all the clients to stream at the highest available bandwidth, there is a sudden adjustment of the video rates to satisfy the available resources at the base station. This results in unfair bit rate selection at the start of the streaming session.

VIII. CONCLUSION

In this paper, we have presented an edge cloud–assisted rate adaptation solution to enhance the user experience in a cellular network. The edge cloud–assisted solution shifts the adaptation intelligence from the client to the edge cloud to jointly optimize the QoE of the streaming clients. We present an edge computing–assisted throughput estimation method that helps the rate adaptation algorithm to improve the video quality objective. Next, we propose an INLP optimization model that jointly optimizes the users viewing experience of the competing clients. Then, we propose a heuristic adaptation algorithm to jointly enhance multiple conflicting video quality objectives for improving the user experience of the competing clients in a cellular network. Using extensive simulations, we proved that the proposed throughput estimation method assists the rate adaptation algorithm to smoothen the frequency of the bit rate switches and improve the fairness. Next, we show that the proposed algorithm outperforms the existing algorithms in terms of the QoE and fairness under different client and server settings. Lastly, we show that the proposed algorithms maintain high user experience regardless of the clients' speeds.

REFERENCES

[1] Cisco. (Feb. 2019). *Cisco Visual Networking Index: Forecast and Trends, 2017–2022 White Paper*. Accessed: Apr. 27, 2019. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html

[2] *Information Technology—Dynamic Adaptive Streaming Over HTTP (DASH)—Part 1: Media Presentation Description and Segment Formats*, Standard ISO/IEC 23009-1:2014, 2014. Accessed: Apr. 27, 2019. [Online]. Available: https://www.iso.org/standard/65274.html

[3] F. Dobrian, A. Awan, D. Joseph, A. Ganjam, J. Zhan, V. Sekar, I. Stoica, and H. Zhang, "Understanding the impact of video quality on user engagement," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 91–99, Aug. 2013.

[4] P. Ni, R. Eg, A. Eichhorn, C. Griwodz, and P. Halvorsen, "Flicker effects in adaptive video streaming to handheld devices," in *Proc. ACM Int. Conf. Multimedia*, Scottsdale, AZ, USA, Nov. 2011, pp. 463–472.

[5] Y. Liu, S. Dey, D. Gillies, F. Ulupinar, and M. Luby, "User experience modeling for DASH video," in *Proc. IEEE Packet Video Workshop*, San Jose, CA, USA, Dec. 2013, pp. 1–8.

[6] Y. Shen, Y. Liu, H. Yang, and D. Yang, "Quality of experience study on dynamic adaptive streaming based on. http," *IEICE Tran. Commun.*, vol. E98-B, no. 1, pp. 62–70, Jan. 2015.

[7] S. Egger, B. Gardlo, M. Seufert, and R. Schatz, "The impact of adaptation strategies on perceived quality of HTTP adaptive streaming," in *Proc. ACM Workshop Design, Qual. Deployment Adapt. Video Streaming*, Sydney, NSW, Australia, Dec. 2014, pp. 31–36.

[8] 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 J. Sel. Areas Commun.*, vol. 32, no. 4, pp. 719–733, Apr. 2014.

[9] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis, "What happens when HTTP adaptive streaming players compete for bandwidth?" in *Proc. NOSSDAV*, New York, NY, USA, Jun. 2012, pp. 9–14.

[10] T.-Y. Huang, N. Handigol, B. Heller, N. McKeown, and R. Johari, "Confused, timid, and unstable: Picking a video streaming rate is hard," in *Proc. IMC*, Boston, MA, USA, Nov. 2012, pp. 225–238.

[11] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5G networks: New paradigms, scenarios, and challenges," *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 54–61, Apr. 2017.

[12] S. Yi, Z. Hao, Q. Zhang, Q. Zhang, W. Shi, and Q. Li, "LAVEA: Latency-aware video analytics on edge computing platform," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2017, pp. 2573–2574.

[13] D. Wang, Y. Peng, X. Ma, W. Ding, H. Jiang, F. Chen, and J. Liu, "Adaptive wireless video streaming based on edge computing: Opportunities and approaches," *IEEE Trans. Services Comput.*, to be published. doi: 10.1109/TSC.2018.2828426.

[14] W. Ur Rahman and K. Chung, "A multi-path-based adaptive scheme for multi-view streaming over HTTP," *IEEE Access*, vol. 6, pp. 77869–77879, 2019.

[15] R. Dubin, O. Hadar, and A. Dvir, "The effect of client buffer and MBR consideration on DASH Adaptation Logic," in *Proc. WCNC*, Apr. 2013, pp. 2178–2183.

[16] W. ur Rahman and K. Chung, "Buffer-based adaptive bitrate algorithm for streaming over HTTP," *KSII Trans. Internet Inf. Syst.*, vol. 9, no. 11, pp. 4585–4622, Nov. 2015.

[17] 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.

[18] S. Akhshabi, A. C. Begen, and C. Dovrolis, "An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP," in *Proc. MMSys*, San Jose, CA, USA, Feb. 2011, pp. 157–168.

[19] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate adaptation for adaptive HTTP streaming," in *Proc. MMSys*, San Jose, CA, USA, Feb. 2011, pp. 169–174.

[20] W. ur Rahman and K. Chung, "A novel adaptive logic for dynamic adaptive streaming over HTTP," *J. Vis. Commun. Image Represent.*, vol. 49, pp. 433–446, Nov. 2017.

[21] K. Miller, E. Quacchio, G. Gennari, and A. Wolisz, "Adaptation algorithm for adaptive streaming over HTTP," in *Proc. PV*, Munich, Germany, May 2012, pp. 173–178.

[22] 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," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 187–198, 2015.

[23] W. ur Rahman and K. Chung, "SABA: Segment and buffer aware rate adaptation algorithm for streaming over HTTP," *Multimedia Syst.*, vol. 24, no. 5, pp. 509–529, Oct. 2018.

[24] P. Juluri, V. Tamarapalli, and D. Medhi, "SARA: Segment aware rate adaptation algorithm for dynamic adaptive streaming over HTTP," in *Proc. IEEE Int. Conf. Commun. Workshop*, London, U.K., Jun. 2015, pp. 1765–1770. doi: 10.1109/ICCW.2015.7247436.

[25] M. Zhao, X. Gong, J. Liang, W. Wang, X. Que, and S. Cheng, "QoE-driven cross-layer optimization for wireless dynamic adaptive streaming of scalable videos over HTTP," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 451–465, Mar. 2015.

[26] J. Xie, R. Xie, T. Huang, J. Liu, and Y. Liu, "Energy-efficient cache resource allocation and QoE optimization for HTTP adaptive bit rate streaming over cellular networks," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–6.

[27] *Information Technology—Dynamic Adaptive Streaming Over HTTP (DASH)—Part 5: Server and Network Assisted DASH (SAND)*, Standard ISO/IEC 23009-5:2017, 2017. [Online]. Available: https://www.iso.org/standard/69079.html

[28] Z. Li, S. Zhao, D. Medhi, and I. Bouazizi, "Wireless video traffic bottleneck coordination with a DASH SAND framework," in *Proc. IEEE Vis. Commun. Image Process.*, Nov. 2016, pp. 1–4.

[29] Z. Yan, J. Xue, and C. W. Chen, "Prius: Hybrid edge cloud and client adaptation for HTTP adaptive streaming in cellular networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 1, pp. 209–222, Jan. 2017

[30] A. Mehrabi, M. Siekkinen, and A. Ylä-Jääski, "Edge computing assisted adaptive mobile video streaming," *IEEE Trans. Mobile Comput.*, vol. 18, no. 4, pp. 787–800, Jun. 2019. doi: 10.1109/TMC.2018.2850026.

[31] *Transparent End-to-End Packet-Switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming Over HTTP (3GP-DASH)*, document 3GPP TS 26.247 V12.1.0, 2013. [Online]. Available: http://goo.gl/4EJbvd

[32] Q. Huynh-Thu and M. Ghanbari, "Temporal aspect of perceived quality in mobile video broadcasting," *IEEE Trans. Broadcast.*, vol. 54, no. 3, pp. 641–651, Sep. 2008.

[33] Y. Qi and M. Dai, "The effect of frame freezing and frame skipping on video quality," in *Proc. IEEE Int. Conf. Intell. Inf. Hiding Multimedia Signal Process.*, Dec. 2006, pp. 423–426.

[34] N. Staelens, J. De Meulenaere, M. Claeys, G. Van Wallendael, W. Van den Broeck, J. De Cock, R. Van de Walle, P. Demeester, and F. De Turck, "Subjective quality assessment of longer duration video sequences delivered over HTTP adaptive streaming to tablet devices," *IEEE Trans. Broadcast.*, vol. 60, no. 4, pp. 707–714, Dec. 2014.

[35] A. K. Moorthy, L. K. Choi, A. C. Bovik, and G. de Veciana, "Video quality assessment on mobile devices: Subjective, behavioral and objective studies," *IEEE J. Sel. Topics Signal Process.*, vol. 6, no. 6, pp. 652–671, Oct. 2012.

[36] T. Hoßfeld, M. Seufert, C. Sieber, and T. Zinner, "Assessing effect sizes of influence factors towards a QoE model for HTTP adaptive streaming," in *Proc. IEEE Int. Conf. Multimedia Exper.*, Sep. 2014, pp. 111–116.

[37] J. Chen, R. Mahindra, M. A. Khojastepour, S. Rangarajan, and M. Chiang, "A scheduling framework for adaptive video delivery over cellular networks," in *Proc. ACM Annu. Int. Conf. Mobile Comput. Netw.*, Sep. 2013, pp. 389–400.

[38] M. J. Magazine and M. S. Chern, "A note on approximation schemes for multidimensional knapsack problems," *Math. Oper. Res.*, vol. 9, no. 2, pp. 244–247, 1984.

[39] A. Zambelli. *IIS Smooth Streaming Technical Overview*. Microsoft Corporation. Accessed: Jul. 20, 2019 [Online]. Available: https://www.bogotobogo.com/VideoStreaming/Files/iis8/IIS_Smooth_Streaming_Technical_Overview.pdf

[40] Adobe. *Configure HTTP Dynamic Streaming and HTTP Live Streaming*. Accessed: Jul. 20, 2019 [Online]. Available: https://helpx.adobe.com/adobeĚ/configure-dynamic-streaming-live-streaming.html

[41] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer system," Digit. Equip. Corp., Maynard, MA, USA, Tech. Rep. TR-301, Dec. 1984.

WAQAS UR RAHMAN (M'19) received the B.S. degree in electrical engineering from the COMSATS Institute of Information Technology, Islamabad, Pakistan, in 2009, the M.S. degree in computer engineering from the University of Engineering and Technology, Taxila, Pakistan, in 2012, and the Ph.D. degree in electronics and communications engineering from Kwangwoon University, South Korea, in 2018, where he was a Postdoctoral Researcher. He is currently a Postdoctoral Researcher with Kyung Hee University. His research interests include multi-access edge computing, the Internet of Things, QoS, QoE support, and rate control for video communications over wired/wireless networks.

CHOONG SEON HONG (S'95–M'97–SM'11) received the B.S. and M.S. degrees in electronic engineering from Kyung Hee University, Seoul, South Korea, in 1983 and 1985, respectively, and the Ph.D. degree from Keio University, Japan, in 1997. In 1988, he joined KT, where he was involved in broadband networks, as a member of Technical Staff. Since 1993, he has been with Keio University. He was with the Telecommunications Network Laboratory, KT, as a Senior Member of Technical Staff, and as the Director of the Networking Research Team, until 1999. Since 1999, he has been a Professor with the Department of Computer Science and Engineering, Kyung Hee University. His research interests include Future Internet, ad hoc networks, network management, and network security. He is a member of the ACM, IEICE, IPSJ, KIISE, KICS, KIPS, and OSIA. He has served as the General Chair, TPC Chair/Member, or an Organizing Committee Member for international conferences, such as NOMS, IM, APNOMS, E2EMON, CCNC, ADSN, ICPP, DIM, WISA, BcN, TINA, SAINT, and ICOIN. He was an Associate Editor of the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, and the IEEE JOURNAL OF COMMUNICATIONS AND NETWORKS. He currently serves as an Associate Editor of the *International Journal of Network Management*, and an Associate Technical Editor of the *IEEE Communications Magazine*.

EUI-NAM HUH (M'13) received the B.S. degree from Busan National University, South Korea, the master's degree in computer science from The University of Texas at Austin, USA, in 1995, and the Ph.D. degree from Ohio University, USA, in 2002. He is currently with Kyung Hee University, South Korea, as a Professor with the Department of Computer Science and Engineering. His research interests include cloud computing Internet of Things, the future Internet, distributed real time systems, mobile computing, and big data and security. He is a Review Board of the National Research Foundation of Korea. He has also served as various types of chairs in many community services for ICCSA, WPDRTS/IPDPS, the APAN Sensor Network Group, ICUIMC, ICONI, APIC-IST, ICUFN, and SoICT. He is a Vice Chairman of the Cloud/Bigdata Special Technical Group of TTA, and an Editor of ITU-T SG13 Q17.