# A Mobile Edge Computing-Assisted Video Delivery Architecture for Wireless Heterogeneous Networks

Yue Li*‡, Pantelis A. Frangoudis§, Yassine Hadjadj-Aoul‡, Philippe Bertin*
*Orange Labs, Cesson Sévigné, France
§EURECOM, Sophia Antipolis, France
‡IRISA/University of Rennes 1, France
Email: yue2.li@orange.com, pantelis.frangoudis@eurecom.fr, yhadjadj@irisa.fr, philippe.bertin@orange.com

***Abstract*—We focus on QoE-optimized video delivery in a wireless heterogeneous network setting, where users equipped with multi-interface devices access a Dynamic Adaptive Streaming over HTTP (DASH) video service. We provide an Integer Linear Programming (ILP) formulation for the problem of optimal joint video quality and network interface selection and a heuristic algorithm to solve it, shown via simulation and testbed experiments to achieve near-optimal performance in terms of Quality of Experience (QoE), with reduced execution time. We further design a video delivery architecture based on the emerging Mobile Edge Computing (MEC) standard, where our video quality and network selection functionality is executed as a MEC application. Notably, our architecture transparently operates with standard DASH clients.**

***Keywords*—Heterogeneous Network, DASH, QoE, Fairness, Mobile Edge Computing**

## I. Introduction

With the proliferation of advanced mobile devices such as tablets and smartphones, video streaming is gaining popularity and is becoming one of the major Internet services for mobile consumers. Mobile video traffic, which accounted for 55% of the total mobile traffic in 2015, will represent more than 70% in 2020 [1]. This significant growth is accompanied by a widespread adoption of the DASH standard [2], [3], one of the implementations of HTTP-based adaptive streaming (HAS). DASH offers clients the possibility to switch between different video qualities (and, thus, bitrates) in case of variations in network conditions. This helps avoiding play-out interruptions, thus improving video QoE. However, when the clients share the same bottleneck link, as is the case for mobile networks, the DASH protocol shows some instabilities, unfairness, and bandwidth under-utilization [4]. There is, therefore, a need to be aware of the end-user device context and network conditions to optimize video delivery.

Since contemporary mobile devices are usually equipped with multiple radio interfaces (e.g. 3G/LTE, Wi-Fi, etc.), exploiting such diverse wireless access networks to improve the video streaming services is promising. In case of congestion on one wireless network, a user should be able to switch to another network smoothly, without any QoE degradation.

Similarly, from an operator's perspective, it is essential to guarantee the QoE of mobile users and maximize network performance by assisting the users to make the network selection decision for the provided video streaming service. However, optimizing the video quality selection strategy for video streaming over multiple wireless networks, considering the video service's requirements, the wireless channel profiles and the costs of the different links remains an open issue. The emerging ETSI standard on Mobile Edge Computing (MEC) [5] may play an important role in this direction. MEC provides the possibility to leverage the cloud computing power by deploying IT application services at the edge of the mobile network (i.e. near the final users). Thus, to address the aforementioned issues, we introduce a MEC application which can take video quality and network selection decisions.

We make the following contributions: (i) For the problem of per user joint video quality and network selection in a multi-access heterogeneous network, we propose an ILP formulation (§ III) and heuristics (§ IV) to solve it efficiently and near-optimally from a QoE perspective, as our numerical results and testbed experiments indicate (§ VI). (ii) We design a MEC-assisted DASH video delivery architecture (§ V) where our algorithms are integrated; our design is compatible with the reference MEC framework and the DASH standard.

## II. Background and related work

In Dynamic Adaptive Streaming over HTTP (DASH) [2], [3] different representations (i.e., qualities/bitrates) of the same video content are available from an HTTP server. Video is stored in segments and information on the available representations, their timing, structure, and format/codec is included in a Media Presentation Description (MPD) file, which the client downloads to start viewing the video. A client can switch among representations in response to what it identifies as changes in network conditions (e.g., congestion), to reduce the risk of playout interruptions and improve video quality.

Video quality selection in DASH is client-driven. The MPEG-DASH specification only concerns the MPD and the segment formats, while the client behavior for fetching and playing video content is outside the scope of the standard.

Existing approaches to improve DASH performance can be classified as either client-based or server/proxy-based. Client-based approaches [6], [7] select the appropriate video rate dynamically based on parameters available at the terminal side, such as the client's player buffer state, and throughput and available bandwidth estimation. Some other studies propose proxy-based approaches [8], [9], where intermediary nodes are placed in the network to collect information regarding the available bandwidth in order to influence client behavior.

Our solution belongs to the latter class of approaches, as we consider using a mobile edge cloud for improving DASH performance. However, compared with most existing works which focus only on a single access network [10], the proposed solution assumes a multi-access heterogeneous network environment. Device multi-homing has the advantage of better resilience to changes in network conditions. It also allows going further in access network optimization with the objective to deliver a more stable video experience.

## III. Problem formulation

We assume a network setting where $n$ users, each having $m$ available networks, access a video service where $q$ distinct video representations are available. The binary variable $x_{i,j,k}$ represents whether user $i$ accesses video representation $k$ over network $j$, where $i \in [1, n]$, $j \in [1, m]$, $k \in [1, h]$, and $b_k$ and $E_k$ denote the video bitrate of representation $k$ and its translated QoE value, respectively. QoE is expressed in terms of the Mean Opinion Score (MOS), i.e., the expected rating that a panel of users would give to the quality of the transmitted video in the 1-5 (poor-excellent) scale.

Our objective is to maximize the sum of the MOS of all clients. To achieve this, the MEC application instantiated on the virtualisation infrastructure of the mobile edge host [11] should find the maximum permitted video quality per client and the network over which it can be downloaded. Our model assumptions are summarized below:

- At any time, each user downloads a single video representation.
- At any time, each user uses a single radio interface to access the video service[1].
- Each access network has a specific total capacity $C_j$, $j \in [1, m]$.
- Each individual radio link between the user's terminal $i$ and the network $j$ has a specific capacity $l_i^j$, which is determined by the modulation, codec, etc.

We model the problem of user-video representation-network assignment as a Binary Integer Linear Program:

$$\text{Maximize} \quad \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{h} x_{i,j,k} E_k \tag{1}$$

$$\text{subject to} \quad \sum_{i=1}^{n}\sum_{k=1}^{h} x_{i,j,k} b_k \leq C_j, \forall j \in [1, m] \tag{2}$$

$$x_{i,j,k} b_k \leq l_i^j, \forall i \in [1, n], \forall j \in [1, m], \forall k \in [1, h] \tag{3}$$

$$\sum_{k=1}^{h}\sum_{j=1}^{m} x_{i,j,k} \leq 1, \forall i \in [1, n] \tag{4}$$

$$x_{i,j,k} \in \{0, 1\}, \forall i \in [1, n], \forall j \in [1, m], \forall k \in [1, h]. \tag{5}$$

[1]Multi-path through multiple access networks is not considered in this paper.

The set of constraints in (2) guarantees that the capacity of network $j$ is not exceeded by the assignment, since, for each network, the sum of allocated bitrates to all users is constrained by the capacity of the network ($C_j$). Constraints (3) ensure that a video representation is not assigned to user $i$ over network $j$ if its bitrate cannot be accommodated by the user's link capacity. A guarantee that a user is assigned *at most* one video representation is given by (4). The latter set of constraints also guarantee that each terminal uses *at most* one network for the video service (since it receives at most one video representation).

Deriving exact optimal solutions for large problem instances is expensive computationally. We therefore propose a low-complexity heuristic algorithm in the following section.

## IV. Algorithm Design

We propose a heuristic algorithm to derive a solution that attempts to maximize the QoE across all the clients, with network-wide proportional fairness in mind (we want to avoid that some clients have the very high video quality, whereas other clients have very low video qualities). The proposed solution takes into account the number of active users' equipment (UE), the video quality, and the available bandwidth in each network. The main idea is to fill the network bandwidth by increasing the video quality gradually in a water-filling like manner.

The notations used in the proposed algorithm are:

- $\mathcal{U} = \{1, 2, \cdots, n\}$: The set of identifiers of the mobile users intending to view a video.
- $\mathcal{N} = \{1, 2, \cdots, m\}$: The set of identifiers of the access networks the mobile users can connect to.
- $q_k$, where $k \in [1, h]$: The video quality which can be rendered at the mobile users' level. We denote $q_1$ as the lowest quality and $q_h$ as the highest quality. The function $b(q_k)$ denotes the required bitrate for the video quality $k$.
- $\mathcal{S}_j$, where $j \in [1, m]$: This variable, which contains a set of tuples $(i, q_k)$, indicates the list of users associated with network $j$ and their corresponding quality. The function *TotalBR*$(\mathcal{S}_j)$ and *RemainBR*$(\mathcal{S}_j)$ denote the overall allocated bitrate to all the users in network $j$ and the remaining bitrate, respectively.
- $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_m\}$: This represents the global solution containing all the networks and their corresponding users and video quality, which will be the final output of the proposed heuristic algorithm.
- $B_j$: Current available bandwidth in network $j$.
- $p_j$: Current number of assigned users in network $j$.

The algorithm consists of the three following steps:

**Step 1**: The main idea behind this step (*Network Selection*), is to associate temporarily the users with an access network and the lowest video quality $q_1$ (see algorithm 1). First, the algorithm finds the set $\mathcal{N}_c$ of networks in $\mathcal{N}$ having enough bandwidth to transmit the lowest video quality. Then, it decides which network in $\mathcal{N}_c$ is the most suitable for each client. In order to place efficiently the users among the access networks while keeping the same level of enhancement, the users are placed successively in the network with the best fair share (see lines 12 and 14). After each placement, the network with not enough resources is automatically removed from the list $\mathcal{N}_c$, as shown in lines 17 and 18. Note that the algorithm ends when all users are assigned to an access network or when $\mathcal{N}_c$ is empty, which means that there are not enough resources to serve all users. In this last case, the *Block*$([i..n])$ function allows barring, for a certain period of time, all users from $i$ to $n$.

**Algorithm 1** Network Selection (Step 1)

**Init:** $\mathcal{S}_j = \{\phi\}$, $\mathcal{N}_c = \{\phi\}$, $p_j = 0$, **for all** $j \in \mathcal{N}$;
1: **for all** $j \in \mathcal{N}$ **do**
2:   **if** $B_j \geq b(q_1)$ **then**
3:     $\mathcal{N}_c = \mathcal{N}_c \cup \{j\}$;
4:   **end if**
5: **end for**
6: **for all** $i \in \mathcal{U}$ **do**
7:   **if** $\mathcal{N}_c = \{\phi\}$ **then**
8:     Block$([i..n])$;
9:     **return** $\mathcal{S}$;
10:   **else**
11:     **for all** $j \in \mathcal{N}_c$ **do**
12:       $Avg_j \leftarrow \frac{B_j}{p_j+1}$;
13:     **end for**
14:     $net = \arg\max_{j \in \mathcal{N}_c} \; Avg_j$;
15:     $\mathcal{S}_{net} = \mathcal{S}_{net} \cup \{(i, q_1)\}$;
16:     $p_{net} + +$;
17:     **if** $B_{net} < (p_{net} + 1) \times b(q_1)$ **then**
18:       $\mathcal{N}_c = \mathcal{N}_c - \{net\}$;
19:     **end if**
20:   **end if**
21: **end for**
22: **return** $\mathcal{S}$;

**Step 2**: The objective of this step (*Local optimization of resources*), is to increase fairly the video quality of the users in each network (see algorithm 2). It is based on the relaxation principle, in which the user with the worst quality is selected, using the function GetWorse(), and gradually assigned a better quality, when possible. Indeed, a better quality is assigned to a user only when it can handle it (see line 7).

The algorithm ends when there is no user to improve. This may happen when $\mathcal{S}'_j = \{\phi\}$, or when the worst user has the highest quality $q_h$, or when there are no available resources to allocate. In this last case, the remaining amount of resources, given by function RemainBR(), is too small for a quality improvement.

**Step 3**: After having assigned the different users to a predefined access network with the best possible quality, this step "*Global optimization of resources*)" optimizes further the quality and network assignment (see algorithm3). The main idea here is to first select the user with the lowest video quality using the function FindUserMin(), which returns a user Id equal to 0 when there is no user's quality to improve. In this

**Algorithm 2** Local optimization of resources (Step 2)

**Init:** $\mathcal{S}'_j = \mathcal{S}_j$;
1: **while** $\mathcal{S}'_j \neq \{\phi\}$ **do**
2:   $(i, q_k) = \text{GetWorse}(\mathcal{S}'_j)$;
3:   **if** $q_k \geq q_h$ **then**
4:     **return** $\mathcal{S}_j$;
5:   **end if**
6:   **if** $(\text{RemainBR}(\mathcal{S}_j) > b(q_{k+1}) - b(q_k))$ **then**
7:     **if** $l_i^j \geq b(q_{k+1})$ **then**
8:       Assign$(j, i, q_{k+1})$;
9:     **else**
10:       $\mathcal{S}'_j = \mathcal{S}'_j - \{(i, q_k)\}$;
11:     **end if**
12:   **else**
13:     **return** $\mathcal{S}_j$;
14:   **end if**
15: **end while**
16: **return** $\mathcal{S}_j$;

case the optimal solution $\mathcal{S}$ is returned. Otherwise, the network with the maximal remaining bandwidth is selected as a potential destination, using the function MaxNetRemainBR(). If there are enough resources to host the selected user, it is assigned to this network. Otherwise, the algorithm ends by returning the optimal solution $\mathcal{S}$. Note that this process repeats until no increment on video quality is possible, which guarantees the convergence of the proposed greedy algorithm.

**Algorithm 3** Global optimization of resources (Step 3)

1: **while** true **do**
2:   $(net_s, i, q_k) = \text{FindUserMin}(\mathcal{S})$;
3:   **if** $i = 0$ **then**
4:     **return** $\mathcal{S}$;
5:   **end if**
6:   $net_d = \text{MaxNetRemainBR}(\mathcal{S} - \{net_s\})$;
7:   **if** $(\text{RemainBR}(net_d) < b(q_{k+1}))$ **then**
8:     **return** $\mathcal{S}$;
9:   **else**
10:     Move$(i, q_{k+1}, net_s, net_d)$;
11:     Go back to **Step 2** for $net_s$;
12:   **end if**
13: **end while**

## V. Architecture description

In the proposed architecture illustrated in Fig. 1, the users can connect to the mobile core network through the different available access networks. The architecture is composed of the following entities:

**User Equipment (UE)** The UE is equipped with multiple radio interfaces (i.e. multi-homed terminal). When it receives an MPD file, the UE downloads video chunks from the URLs indicated in the file. Since the video streaming is over HTTP, session continuity is guaranteed when the UE switches across access networks.

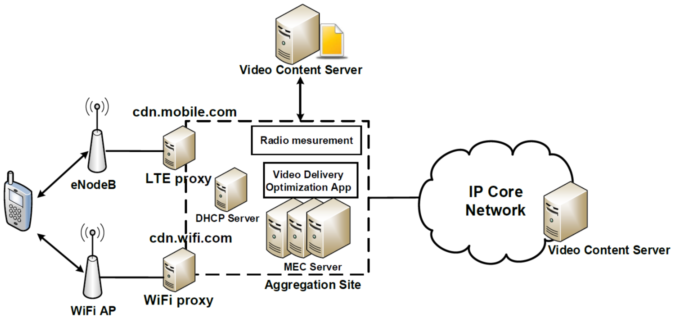


Fig. 1: Our MEC-assisted mobile adaptive HTTP streaming architecture.

**Access Point (AP)** The APs consist of different possible access technologies such as an eNodeB, a Wi-Fi AP, a Femtocell AP, etc. All these APs are managed by the network operator and they are connected to a common mobile backhaul network.

**MEC server** The MEC server, which is managed by the network operator, hosts various MEC applications.

**LTE and Wi-Fi proxies** The proxies are in charge of intercepting and relaying the UE requests and the MPD server responses. They are identified by different domain names (e.g. cdn.lte.com and cdn.wifi.com), which are included in the representation field of the MPD file to indicate the path for transmitting video chunks.

**Multi-homed DHCP server** The multi-homed DHCP server serves multiple radio access networks. It configures different gateways towards the multiple available access networks, to direct chunk requests via the appropriate network interface.

**MEC server-based video delivery optimization application** This application is located in the MEC server. It runs an optimization algorithm, which can run periodically or be triggered when network congestion or QoE degradation are detected. The application periodically measures the network status (link capacities, the network load, the video encoded quality), using, among others, the Radio Network Information Service (RNIS) [11] MEC API. These metrics are then used to optimize the path and the video quality selection schemes with the objective to guarantee an optimized video delivery. The calculated result is then applied by transforming the original MPD file, which is sent by the MPD server located at the service provider.

**MPD server** The MPD server, which is managed by the content or the service provider, stores the original MPD files with the video encoding levels and the default chunk addresses on the video content server(s). It responds to the request of the MEC application for sending the original MPD file.

**Video Content Server** Video content server is managed by the content provider. It stores the original video files.

To apply our optimization solution, we need a mechanism so that the clients can receive updated versions of the MPD file. We achieve this in an MPEG DASH compliant, receiver-driven and transparent way by exploiting the `minimumUpdatePeriod` attribute of the standard. When this attribute is present in the MPD file, the client is instructed to periodically request an up-to-date version of the file from the video server.

We assume that a number of distinct video representations are available. For each representation, the corresponding required transmission bitrate is indicated in the `bandwidth` field. This bitrate corresponds to the $b_k$ variable in our system model. The `media` field indicates the address where the video chunks are stored, and the `path` attribute indicates the base URL where the content is stored. In our approach, this is a domain name which is resolved by the DNS server as described in Fig. 1. Thus, by appropriately modifying this path, the mobile users are directed to select the appropriate network interface to download video chunks.

By modifying periodically `bandwidth`, `path`, and `media` fields in real-time according to the proposed algorithm, the MEC application can control the maximum permitted video quality, which can be downloaded by mobile users, and their corresponding network interface. Therefore, the mobile users can always get an updated MPD file and select the optimal video quality and network.

## VI. Performance evaluation

### A. Numerical results

We implement the model presented in Section III using the IBM ILOG CPLEX Optimizer [12]. Our proposed heuristic is implemented in Python. For evaluating the scalability of these two approaches, we measure the execution time of the CPLEX QoE-aware solution and our heuristic. We vary the number of users from 10 to 100, and the video is encoded into 4 bitrates: 128, 256, 512, 1024 Kbps. Each user has two available networks, $n_1$ and $n_2$, with bandwidth $b_1$ and $b_2$, respectively. We generate 1000 different network configurations randomly (i.e., the combination of $b_1$ and $b_2$). The individual link capacity of these networks is 1 Mbps. Each measurement is repeated 10 times. Then, we calculate the average execution time over the 1000 network configurations and the 10 repetitions to obtain the results presented in Fig. 2. It can be noticed that the execution time of CPLEX QoE-aware solution increases exponentially with the number of clients, which indicates that it is not scalable. Our proposed heuristic, though, scales linearly.

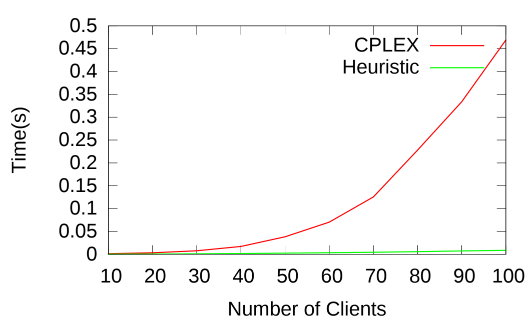


Fig. 2: Execution time with different number of users.

We further compare the two algorithms in terms of our model's QoE-aware objective function (sum of QoE). The bandwidth $b_1$ of network $n_1$ varies from $100$*Kbps* to $50$*Mbps* and the bandwidth $b_2$ of network $n_2$ varies from $50$*Kbps* to $25$*Mbps*. We define optimality as: $\frac{QoE_{\text{heuristic}}}{QoE_{\text{CPLEX}}} \times 100$.

Our results are shown in Fig. 3, for 10, 50, and 100 clients. We can see from Fig. 3 that from the beginning of the experiment, when the network is heavily loaded, optimality is equal to 100%, which indicates that the two algorithms derive equivalent solutions, reaching the same overall QoE. With the increase of network bandwidth, optimality starts to degrade. The worst case across our experiments for the heuristic solution is approximately at 80% of the optimal (with 10 clients), which is a positive result. As network bandwidth continues to increase, the two algorithms allow obtaining the same overall QoE (the optimality metric converges to 100%) when the network is lightly or not loaded. Based on our experiments, our heuristic approximates well the optimal solution while having an execution time which allows it to be applied to real MEC settings.

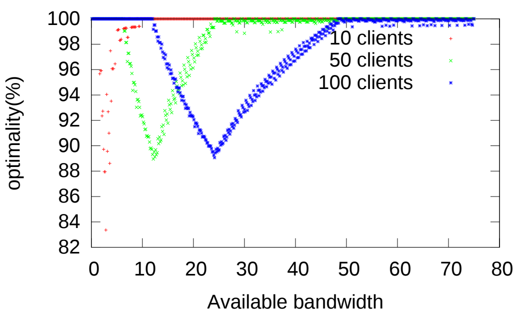


Fig. 3: Optimality with different number of users and bandwidths

### B. Testbed experiments

The evaluation is carried out with DASH clients accessing to two different networks. We have set up a local-area testbed of 9 Linux-based machines. One machine acts as the video server running the nginx [13] HTTP server, where video content is stored. For simplicity, the MPD server and the MEC application are hosted within this server. The other 8 machines (DASH clients) and equipped with two Ethernet interfaces $eth1$ and $eth2$ to emulate multi-homed UEs with LTE and WiFi interfaces. Clients run the MP4Client video player [14], which we have extended to record the Quantization Parameter (QP) and the video interruption statistics that are used for QoE assessment.

The MEC video delivery optimization application is aware of the current number of clients in the system, as well as the available bandwidth for each network. It intercepts client requests for MPD files and executes the adaptation algorithm according to its network awareness and the current client demand. It retrieves the requested MPD files from the source and appropriately transforms them to steer clients towards the best network and best video quality.

Each client accesses a 5 minute video sequence. In particular, using ffmpeg, we created different H.264/AVC representations of the Blender Foundation's "Big Buck Bunny" [15] open-source movie (with approximate bitrates of 128, 256, 512, 1024, 2048, 4096 Kbps) with a 1280×720 resolution, and prepared them for DASH delivery using GPAC's MP4Box utility [14]. Each video segment has a duration of 2 s. We use the `minimumUpdatePeriod` MPD attribute to instruct each client to request an updated version of the manifest file every 10 s.

### C. Performance metrics

We evaluate our system in terms of the average achieved QoE for all users at any time instant. We estimate QoE using objective, measurable video-service-level parameters (interruption statistics, encoding parameters), which we translate to MOS estimates. In DASH, two important parameters that affect QoE are video encoding quality (determined by the Quantization Parameter (QP) used for encoding) and playout interruptions (i.e., buffering delays due to poor bandwidth). Several QoE estimation model are proposed considering these parameters. In our previous work [10], we used the PSQA tool developed by Singh et al. [16] for H.264/AVC-encoded HTTP video quality estimation. The PSQA methodology involves training a Random Neural Network (RNN) using data from subjective tests, where a set of parameters affecting quality is monitored and the ratings of users are recorded. The trained RNN classifier can then be applied to calculate the expected MOS for specific values of the input parameters. This approach allows having an accurate estimation of the QoE when the video quality is high. However, this model is not accurate with low video quality (i.e., high QP value). On the other hand, the tool introduced in [17] (denoted as *Vipeer*) is a purely QP-based QoE estimation model without taking playout interruptions into account.

To compare the performance of the different strategies considered in our experiments, we combine these two QoE estimation approaches under the assumption that impairments due to playout interruptions and QP have an addictive effect on QoE.The additivity assumption is typical in the context of VoIP (see the E-model [18]), but it has also been proposed for video services [19]. We apply it in this work as it simplifies the comparison of the discussed strategies, but, as Hoßfeld et al. show [20], multi-factor QoE models and the assumptions therein require further study.

In this combined model, we calculate the two impairments independently. Since the maximum value of MOS is 5 (excellent perceived quality), the QP impairment is given by $I_{\text{QP}} = 5 - MOS_{\text{Vipeer}}(w)$ using the VIPEER model and assuming no interruptions. The playout interruption impairment is given by: $I_{\text{interruption}} = 5 - MOS_{\text{PSQA}}(w)$ using the PSQA model assuming the perfect QP. The variable $w$ denotes the measurement result from our experiment. The final MOS given in: $MOS_{\text{Final}} = 5 - I_{\text{QP}} - I_{\text{interruption}}$, is used as our QoE estimate.

### D. Candidate strategies

In our testbed experiments, since the number of clients is small, the running time for solving the optimization problem using CPLEX is only about several ms. Therefore, we can compare the optimal solution and the proposed heuristic to accurately evaluate the effectiveness of these solutions. We compare five adaptation strategies: (i) Wi-Fi first strategy. In this strategy, we do not control the MPD file adaptation. Clients decide on the video quality selection, and always select the Wi-Fi network when it is available. This is one of typical user behavior, since the monetary cost is always the clients' concern and Wi-Fi is often low cost. (ii) Random strategy. In this strategy, the original MPD file is not modified by MEC application and the clients make the video quality adaptation decision. During playing the video, all the clients connect to one of the two networks randomly without any preference. This can happen in real network since they often switch between Wi-Fi and mobile network to get a better service. (iii) Our proposed heuristic strategy. In this strategy, the algorithm is running in the MEC server and modifies the MPD file periodically according to the result calculated by the algorithm. (iv) CPLEX Bitrate-optimal strategy. In this strategy, we modify the objective function of the model in Section III to maximize the sum of the video bitrates received by all users. (v) CPLEX QoE-optimal strategy. In this strategy, the algorithm maximizes the overall QoE using IBM ILOG CPLEX Optimizer following the constraints presented in the formulated model.

### E. Results

The experiment starts with an initialization period of 15 s during which 8 users join the video service following a Poisson process. For simplicity, we assume that the capacity of each individual radio link (i.e. LTE and WiFi) of users is constant at 1 Mbps. We emulate the network load from $t = 90s$ to $t = 210s$, when the available bandwidth for LTE and Wi-Fi is limited to 6.4Mbps and 3.2Mbps, receptively. We repeat the experiment with the same load at the exact same time instances for the five candidate strategies (see Section VI-D). Each experiment is repeated 5 times.

The experiment result showed that the efficiency of the proposed heuristic strategy compared to the other four strategies since it performed approximately the optimal QoE for all users, which validates the numerical evaluation in section VI-A. The QoE-optimal and heuristic algorithms also improve on QoE fairness. In our experiments, we witnessed an up to $20\%$ increase in terms of Jain's Fairness Index [21] compared with the Wi-Fi First strategy in congested settings. Due to space limitations, we omit these figures of results.

## VII. Conclusion

We focused on enhancing DASH performance in wireless heterogeneous network environments. Since network and video quality selection are key factors that impact user experience, we proposed a MEC-based architecture and provided an ILP formulation for the problem of network and video quality selection. To address realistic use cases, we introduced an efficient heuristic to solve it. The proposed solution can be built into a MEC service without requiring any modifications at the client or the content provider ends. This service uses transparent and standards-compliant techniques for QoE-optimized video streaming, responding to network congestion and dynamic demand. Our testbed experiments demonstrated that our proposed algorithm can achieve close-to-optimal performance.

## References


[1] Cisco, "Cisco visual networking index: Forecast and methodology, 2014-2019," Mai 2015, White Paper.

[2] *Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats*, ISO/IEC Standard 23 009-1:2014, May 2014.

[3] 3GPP, "Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH)," 3GPP, International standard 3GPP TS 26.247, Dec 2015.

[4] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis, "What happens when http adaptive streaming players compete for bandwidth?" in *Proc. of the 22Nd Int. Workshop on Network and Operating System Support for Digital Audio and Video*, ser. NOSSDAV '12. New York, NY, USA: ACM, 2012, pp. 9–14.

[5] M. P. et al., "Mobile-edge computing introductory technical whitepaper," September 2015, White Paper.

[6] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," in *Proc. of the ACM Conf. on Special Interest Group on Data Com.*, ser. SIGCOMM '15. New York, NY, USA: ACM, 2015, pp. 325–338.

[7] C. Zhou, X. Zhang, L. Huo, and Z. Guo, "A control-theoretic approach to rate adaptation for dynamic http streaming," in *Visual Communications and Image Processing (VCIP), 2012 IEEE*, Nov 2012, pp. 1–6.

[8] S. Petrangeli, M. Claeys, S. Latr, J. Famaey, and F. D. Turck, "A multi-agent q-learning-based framework for achieving fairness in http adaptive streaming," in *2014 IEEE NOMS*, May 2014, pp. 1–9.

[9] N. Bouten, J. Famaey, S. Latr, R. Huysegems, B. D. Vleeschauwer, W. V. Leekwijck, and F. D. Turck, "Qoe optimization through in-network quality adaptation for http adaptive streaming." in *CNSM*. IEEE, 2012, pp. 336–342.

[10] Y. Li, P. A. Frangoudis, Y. Hadjadj-Aoul, and P. Bertin, "A mobile edge computing-based architecture for improved adaptive http video delivery," in *2016 IEEE CSCN*, Oct 2016, pp. 1–6.

[11] ETSI, *Mobile Edge Computing (MEC); Framework and Reference Architecture*, March 2016.

[12] Ibm ilog cplex optimizer. [Online]. Available: https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/

[13] nginx. [Online]. Available: http://nginx.org

[14] GPAC multimedia open source project. [Online]. Available: https://gpac.wp.mines-telecom.fr

[15] Big Buck Bunny. [Online]. Available: https://peach.blender.org/

[16] K. D. Singh, Y. Hadjadj-Aoul, and G. Rubino, "Quality of experience estimation for adaptive http/tcp video streaming using h.264/avc," in *2012 IEEE CCNC*, Jan 2012, pp. 127–131.

[17] Advances on monitoring and integration to the demonstrator. [Online]. Available: https://recherche.telecom-bretagne.eu/vipeer/publications/

[18] *The E-model: A computational model for use in transmission planning*, ITU-T Recommendation G.107 Standard, 2009.

[19] *Parametric non-intrusive assessment of audiovisual media streaming quality. Amendment 2: New Appendix III - Use of ITU-T P.1201 for non-adaptive, progressive download type media streaming*, ITU-T Recommendation P.1201 Standard, Dec. 2013.

[20] T. Hoßfeld, L. Skorin-Kapov, P. E. Heegaard, M. Varela, and K.-T. Chen, "On additive and multiplicative qos-qoe models for multiple qos parameters," in *Proc. 5th ISCA/DEGA Workshop on Perceptual Quality of Systems (PQS)*, 2016.

[21] R. Jain, D. Chiu, and W. Hawe, "A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems," DEC Research, Tech. Rep. TR-301, Sep.