Edge Computing Assisted Adaptive Mobile Video Streaming

Abbas Mehrabi, *Member, IEEE*, Matti Siekkinen, *Member, IEEE*, and Antti Ylä-Jääski, *Member, IEEE*

Abstract—Nearly all bitrate adaptive video content delivered today is streamed using protocols that run a purely client based adaptation logic. The resulting lack of coordination may lead to suboptimal user experience and resource utilization. As a response, approaches that include the network and servers in the adaptation process are emerging. In this article, we present an optimized solution for network assisted adaptation specifically targeted to mobile streaming in multi-access edge computing (MEC) environments. Due to NP-Hardness of the problem, we have designed a heuristic-based algorithm with minimum need for parameter tuning and having relatively low complexity. We then study the performance of this solution against two popular client-based solutions, namely Buffer-Based Adaptation (BBA) and Rate-Based Adaptation (RBA), as well as to another network assisted solution. Our objective is two fold: First, we want to demonstrate the efficiency of our solution and second to quantify the benefits of network-assisted adaptation over the client-based approaches in mobile edge computing scenarios. The results from our simulations reveal that the network assisted adaptation clearly outperforms the purely client-based DASH heuristics in some of the metrics, not all of them, particularly, in situations when the achievable throughput is moderately high or the link quality of the mobile clients does not differ from each other substantially.

Index Terms—Server and network assisted DASH, multi-access edge computing (MEC), quality of experience, fairness, load balancing, integer nonlinear programming (INLP), greedy scheduling algorithm

1 Introduction

According to statistics, the majority of Internet traffic is video, such as Netflix, YouTube, or other streaming applications [35], [36]. The network conditions, such as high fluctuation in the available bandwidth when multiple clients simultaneously compete for the shared bottleneck link, can significantly affect the users' quality of experience (QoE) in mobile video streaming applications [10], [16]. Mobile and wireless access further complicates the situation. In order to avoid playback interruption and rebuffering events due to changes in available bandwidth during a streaming session, most media players nowadays use adaptive streaming, such as the non-standard HTTP Live Streaming (HLS) or protocols based on the DASH standard [41].

Many research efforts have been carried out in recent years on designing efficient adaptation mechanisms for mobile video streaming [7], [19], [20]. Vast majority have focused on purely client-side adaptation strategies that do not include explicit coordination between clients, servers, and the network. Such approaches have been shown to lead to unfair bitrate allocation and underutilization of network resources in certain situations where multiple simultaneous clients stream video over the same access network [4]. To remedy this, both research (e.g., [14]) and standardization (SAND-DASH [42]) have started to look for solutions that allow including both the network and the servers in the adaptation process.

- *The authors are with the Department of Computer Science, Aalto University, Espoo FI-00076, Finland.*
  E-mail: {abbas.mehrabidavoodabadi, matti.siekkinen, antti.yla-jaaski}@aalto.fi.

Manuscript received 9 Feb. 2018; revised 12 June 2018; accepted 18 June 2018. Date of publication 25 June 2018; date of current version 4 Mar. 2019.
(Corresponding author: Abbas Mehrabi.)
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TMC.2018.2850026

Recently Multi-Access Edge Computing (MEC) [26] has emerged as a solution to bring computing and content storage to the edge of a mobile network, all the way to the radio access part of it. We have designed an optimized network-assisted video bitrate adaptation scheme for mobile streaming specifically for MEC scenarios. In addition to bitrate selection, the scheme also includes a client to edge server mapping logic. Since the underlying bitrate selection problem is NP-hard, our solution is based on a greedy heuristic. The optimization problem itself is parameterized but our solution is able to self-tune the parameter values, which alleviates the need for parameter tuning by operators in deployment phase.

Besides presenting an optimized network assisted bitrate adaptation solution, our goal in this work is to understand how much, in which way, and at which cost (esp. computational complexity) the QoE and fairness between mobile video streaming clients could be improved through network assisted adaptation compared to client-based approaches. The contributions of this work are the following:

- We propose an integer nonlinear programming (INLP) optimization model for video bitrate selection that jointly maximizes the quality of experience of clients and proportional fairness of the bitrate allocation in mobile video streaming with edge computing facilities.
- We design a near-optimal greedy-based scheduling algorithm to efficiently solve the clients to edge server mapping and the bitrate selection problem. The solution includes self-tuning of the values of the optimization problem parameters.
- We evaluate the solution through simulations where we also include traces of radio access metrics obtained with a standard LTE simulator. The results show that the proposed algorithm performs overall better in terms of QoE, fairness and resource utilization

compared to purely client-based DASH heuristics and one existing SAND-DASH solution. However, the results also suggest that the network assisted adaptation solution does not bring notable advantages over client-based adaptation alternatives in situations where the achievable throughput is relatively low or the wireless link characteristics of the mobile clients differ substantially from each other.

The rest of the paper is organized as follows: We discuss related work in Section 2 and describe the proposed network assisted DASH adaptation system and its components in Section 3. The optimization problem is laid out in Section 4 and the proposed scheduling algorithm is detailed in Section 5. We present simulation-based evaluation through Sections 6, 7, 8, and 9 and conclude the paper with pointing out avenues for future work in Section 10.

2 Related Work

Several quality adaptation approaches for improving the quality of experience of mobile users in dynamic adaptive video streaming over HTTP (DASH) have been proposed during the past years. Seufert et al. [5] provide a comprehensive study on DASH quality adaptation and the major factors that both client and network sides have to take into account from the perspective of QoE. Huang et al. [7] propose a DASH adaptation approach and some of its variants which decides on bitrate selection merely based on the buffer occupancy level. Spetiri et al. [13] design BOLA, an online bitrate adaptation heuristic for evolving DASH standard, which takes into account only the buffer occupancy level without any need for bandwidth prediction. They prove analytically the performance guarantee of the proposed algorithm compared to the offline case in which some knowledge of the future bandwidth variation between the server and the video player is available in advance. Since purely buffer-based adaptation mechanisms may fail specially under high throughput fluctuation, some techniques combine the buffer occupancy level with some extent of throughput prediction to determine the quality for the next video chunk to be downloaded [12], [19], [20]. Wang et al. [20] have recently proposed SQUAD, a DASH rate adaptation online algorithm with the objective of maximizing QoE while minimizing the quality switching by categorizing the quality levels according to the buffer level and estimated network throughput.

Recently, several pieces of work have investigated DASH quality adaptation considering multiple clients associated with either single or multiple video servers. Although Petrangeli et al. [1] investigate the fair bandwidth utilization when multiple clients compete on shared bottleneck link, their proposed objective function and the adaptation heuristic do not take into account the trade-off between the perceived QoE of individual clients and fairness. The objective of bitrate selection in [2] focuses on the maximization of video quality subject to the stability of servers' queues without considering in-network elements in the bitrate adaptation process. Bouten et al. [3] propose quality optimization considering in-network elements into their topology in order to adjust the client's bitrate according to the available bandwidth on multiple bottleneck links. However, in this work, only one bitrate is allocated to each client and also each client is associated to a specified server which is known in advance.

Concerning fairness in QoE optimization, the authors in [22] and [25] propose a DASH scheduling framework for the joint optimization of QoE and resource allocation in scalable video coding (SVC) streaming. In [21], the authors consider specifically energy efficient caching in addition to QoE optimization. The QoE part does not consider the constraint imposed by limited amount of radio resources at all and instead includes monetary cost by users associated to higher bitrates. We do not consider cost because its effect on user behavior in mobile streaming is not well understood and in some countries (Finland) the typical mobile data plans are limitless.

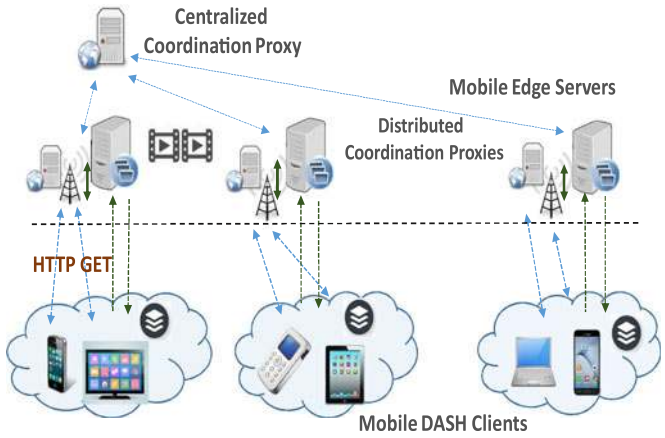


Fig. 1. Multi-access edge computing (MEC) assisted DASH system for the large scale of mobile clients.

Server and network assisted DASH known as SAND-DASH [42] is a recent addition to the DASH [41] standard. It specifies means for the video streaming clients, servers, and the network (DASH-Aware network elements, a.k.a. DANEs) to collaborate with each other. The idea is to enable mechanisms to improve QoE and fair resource utilization and this idea has already been studied by some research. Li et al. [11] propose the maximization of a simple utility function for improving the QoE of mobile clients over a shared bottleneck wireless link through the SAND-DASH collaboration mechanism. Cofano et al. [14] consider bandwidth reservation and bitrate guidance as two in-network components which can help the client-side adaptation approach to adjust the allocated bitrates toward the improvement of clients' QoE. Bentaleb et al. [15] leverage software defined networking to optimize video streaming QoE. The main difference of all the above mentioned works compared to ours is that we specifically consider mobile streaming with MEC and jointly optimize QoE and fairness.

3 Multi-Access Edge Computing Assisted DASH

3.1 Overview

Fig. 1 illustrates the proposed multi-access edge computing assisted DASH system for adaptive mobile video streaming. The edge servers, which are located within the radio access network (RAN) adjacent to base stations, host replicated video content in multiple qualities (representations) served with HTTP.

The core idea of our system is that the so called coordinators, which are also located at the edge servers, periodically do two things: 1) client-to-edge-server mapping and 2) per-client video bitrate selection. These operations are done with the help of radio access level information received from the base stations (one of the key things enabled by MEC) as well as application level information from the

mobile streaming clients. The combination of edge servers and the associated base stations work as independent operating units without need for sharing the data among each other or having a global view about the other edge servers.

The goal of the client-to-server mapping is to balance load, as we describe in detail in Section 3.5 The per-client video bitrate selection is done periodically and as optimally as possible. Optimal bitrate selection here means maximizing QoE but including a fairness factor too. We describe the QoE factors and rationale for fairness in Sections 3.3 and 3.4 and the QoE-fairness problem formulation is presented in Section 4.

Our design does not impose any modifications to the radio resource scheduling mechanisms of the base stations, which means that the base stations allocate available radio resource blocks (RBs) in a proportionally fair manner to the clients [4]. However, we require explicit support from the client applications so that they explicitly communicate with the coordinators and exchange required information (buffer status, selected bitrates, etc.). The communication could be implemented according to the SAND-DASH standard. We made this design decision in order to obtain network assisted adaptation with the best possible performance and evaluate it against client-based adaptation heuristics. It may be possible to support legacy clients that use client-based adaptation heuristics, i.e., off-the-shelf video players used today, by using deep packet inspection (identify video traffic flows and their properties) and per flow traffic shaping (guide rate-based heuristics to select the desired bitrate as in [14]) at the edge servers. Video traffic encryption complicates that approach but could possibly be tackled using machine learning [9]. However, such support for legacy clients is out of scope of this paper.

In our edge computing assisted system, the coordinators handle the clients to server mapping and the bitrate adaptation process using the data from the associated base stations and the connecting mobile clients. Clients to server mapping can be performed in two ways: The local coordinators nearby the edge servers can send their data to the central coordinator for making the server selection decisions while, the bitrate adaptation processes are still performed by the individual coordinators. Since the bitrate scheduling process is performed at the edge server level, it is highly unlikely that the centralized coordination of the connecting clients at the base station creates the scalability problem. Alternatively, the local coordinators can perform both the clients to server mapping and the bitrate adaptation processes which in turn facilitates the decentralized implementation of the network-assisted optimization solution in our system design. The key differences between these two different ways of client to server mapping lie in the computational complexity which is substantially reduced in the second way, and the optimality of the bitrate allocation which at least in theory should be better with the first one.

3.2 Notation and System Description

We follow the discrete time slotted DASH scheduling [3] with total number of $|T|$ time slots and the duration of each slot Δt seconds. At each time slot $1 \leq t \leq |T|$, the data transmission between the base station associated with video server k and different clients goes through a shared bottleneck link with capacity of $W_k^{(t)}$. Please note that $W_k^{(t)}$ refers to the available resource blocks i.e., the number of subcarriers in the frequency domain, at time slot t on base station k. It is also noted that in contrast to [30], we study the system performance for both stationary and mobile clients.

Let A_i and D_i denote the arrival and the departure times, respectively, of client i which correspond to the time that client sends its request for first chunk and the time that it either abandons the streaming session or finishes downloading the last chunk. In the ideal case when no stalling happens during the session and with negligible network delay, the quantity $|D_i - A_i|$ is obviously equal to the watching duration of the video requested by client i and consequently $|D_i - A_i|/C$ is the number of streaming chunks of the video. The media player of each client i maintains a playback buffer for which the client determines a fixed target filling level denoted by B_i^{max} (in kb). $B_i^{(t)} \leq B_i^{max}$ represents the level of data in the client's buffer at time slot t. At each time slot, the coordinators handle the client to server mapping (load balancing) based on the link quality of the clients and the utilization load of the servers. After the server allocation, the coordinators perform the joint optimization of QoE and fairness using the radio access level information from the base stations and the application level data from mobile clients. For the client to server mapping, we define a binary variable $x_{ik}^{(t)}$ such that $x_{ik}^{(t)} = 1$ if client i is allocated to server k for downloading the current chunk at time slot t and $x_{ik}^{(t)} = 0$, otherwise. Furthermore, the integer decision variable $r_{ik}^{(p)} \in R$ denotes the allocated bitrate for chunk index p which is downloaded by client i from server k. The list of parameters involved in the system and their descriptions have been summarized in Table 1. Before the formulation of optimization problem in Section 4, we discuss next the different optimization criteria related to QoE, fairness, and load balancing.

3.3 Quality of Experience

A recent comprehensive study on QoE in dynamic adaptive video streaming [5] shows that four major factors can significantly affect the quality of experience perceived by DASH clients: *video bitrate*, *startup delay*, *stalling ratio* and *bitrate switching*. Our system design assumes that providing optimal QoE requires accounting for all the four abovementioned criteria.

Video bitrate has the highest direct impact on the quality of experience of mobile clients. There is a trade-off between video bitrate and stalling: The higher the video bitrate, the higher the visual video quality but also the probability of experiencing a stall event because the download throughput has a higher chance to drop below the bitrate due to low bandwidth available on the bottleneck link. The average video bitrate over $|D_i - A_i|/C$ downloaded chunks by client i is therefore obtained using the following relation:

$$AQ_i = \frac{C}{|D_i - A_i|} \sum_{p=1}^{|D_i - A_i|/C} \sum_{k=1}^{K} x_{ik}^{(A_i + (p-1)\cdot C)} \cdot r_{ik}^{(p)}. \quad (1)$$

Startup delay refers to the time needed to reach the target buffer filling level of the client upon its arrival. It corresponds to the waiting time of client from click to start of the playback. According to [6], the startup delay has a clearly smaller impact on the dissatisfaction of a viewer than stall events. In order for the mobile client to experience a smoother streaming during the steady state, the video data is first stored in the client's buffer for a time duration equal to the *initial delay* without streaming the content. Denoted by L_i as the initial buffer delay, this means

TABLE 1
Description of Parameters Involved in MEC Assisted DASH System

Notation	Description
C	Constant size of each video chunk (in seconds)
K, S, R	Number of edge servers, number of DASH clients and the discrete set of available bitrates at each server
$R_{max}, R_{min} \in R$	Maximum and minimum available bitrates in set R
$\|T\|$, Δt	Total number of scheduling time slots and the duration of each slot in seconds
$W_k^{(t)}$	Available resource blocks at base station k in slot t
A_i, D_i	Arrival and departure times of client i
B_i^{max}	Maximum buffer filling level of client i
$B_i^{(t)}$	Buffer level of client i at time slot t
AQ_i	Average video bitrate for client i
L_i	Initial playback delay on the buffer of client i
E_i	Accumulated bitrate switching for client i
$d_{ik}^{(t)}$	Physical distance of client i from base station k at time slot t
$Thr_{ik}^{(t)}, \hat{Thr}_{ik}^{(t)}$	Theoretical and effective received data throughput by client i from base station k at time slot t
F_i	Fairness contribution by allocating bitrates to client i during its streaming session
P_{max}	Maximum transmission power of each base station
α	Path loss exponent parameter (normally between 2 and 5)
ρ, ω, θ	Adjustable weighting parameters for average bitrate, bitrate switching and fairness, respectively
δ_S, δ_F	Switching and fairness thresholds
Thr_T, Thr_B	Switching thresholds based on respectively the estimated throughput and the buffer level
$x_{ik}^{(t)}$	Binary variable for the allocation of client i to server k at slot t
$r_{ik}^{(p)} \in R$	Discrete video bitrate for chunk index p of client i allocated to server k

$$\sum_{t=A_i}^{A_i+L_i} \sum_{k=1}^{K} x_{ik}^{(t)} \cdot \hat{Thr}_{ik}^{(t)} \cdot \Delta t = B_i^{max}, \tag{2}$$

where B_i^{max} (in kb) is the target buffer filling level for client i and $\hat{Thr}_{ik}^{(t)}$ is the effective data throughput (in *kbps*) received by client i from server k at time slot t. Let $Thr_{ik}^{(t)}$ denote the theoretical throughput over the wireless link which depends on the modulation and coding schema (MCS) and here, we simply consider the path attenuation model $Thr_{ik}^{(t)} \propto P_{max}/d_{ik}^{(t)\alpha}$ [30] for its computation. P_{max} is the maximum transmission power of the base station, $d_{ik}^{(t)}$ denotes the physical distance between the client i and base station k at time slot t and α is the path loss exponent parameter which is normally between 2 and 5. The effective throughput of the client which is the actual receivable one is then obtained based on both the theoretical throughput and the available downlink RBs at the base station. More precisely, the effective throughput of client i is computed using the relation $\hat{Thr}_{ik}^{(t)} = (Thr_{ik}^{(t)2} / \sum_{\forall p} x_{pk}^{(t)} \cdot Thr_{pk}^{(t)}) \cdot W_k^{(t)}$ where the summation in denominator is taken over all clients p which have been assigned to base station k at time slot t. It is noteworthy to mention that since in our network model, the base station at each time slot allocates the whole RBs to the set of active clients according to their link quality (theoretical throughput), therefore, the effective throughput of the clients is affected with the presence of other simultaneous clients. Furthermore, the coordinators collect the instantaneous effective throughput of the clients meaning that our system model easily accommodates the clients mobility.

Stalling ratio is the the amount of time spent so that video playback is stalled divided by the total duration of the session. Stall events occur when playback buffer empties caused by too low download throughput compared to the video bitrate. Avoiding stall events is critically important because of their prominent role in determining QoE. *Hence, our system is designed to sacrifice video quality (bitrate) in order to avoid stall events whenever possible, i.e., whenever the sum of lowest possible quality bitrates over all active streaming clients does not exceed the network capacity*. To this end, we assume that the player starts to play the video after the startup phase. Given $\hat{Thr}_{ik}^{(t)}$, the buffer level (in *kb*) of client i at time slot t is given by

$$B_i^{(t)} = \begin{cases} B_i^{(t-1)} + \hat{Thr}_{ik}^{(t)} \cdot \Delta t, & A_i \le t \le A_i + L_i \\ B_i^{(t-1)} + (\hat{Thr}_{ik}^{(t)} - r_{ik}^{(p)}) \cdot \Delta t, & A_i + L_i < t \le D_i, \end{cases} \tag{3}$$

where $r_{ik}^{(p)}$ is the allocated bitrate for the currently played out chunk with index p. Accounting for the arrival time of client and initial playback delay, the index p of the chunk played out at time slot $t > A_i + L_i$ is equal to $p = \lceil (t - A_i - L_i)/C \rceil$. Later, we design the optimization problem with such constraints that stall events are avoided whenever possible, i.e., whenever the total amount of resources suffices to support lowest available video bitrates for all clients.

Frequent *bitrate switching* is also considered harmful for QoE [5]. We consider the difference between the bitrate levels of consecutive chunks of the video downloaded by the client as the QoE metric for switching. Hence, the accumulated bitrate switching of client i during the streaming session is given by

$$E_i = \sum_{p=2}^{|D_i - A_i|/C} \sum_{k=1}^{K} \{ x_{ik}^{(A_i + (p-1)\cdot C)} \cdot r_{ik}^{(p)} - x_{ik}^{(A_i + (p-2)\cdot C)} \cdot r_{ik}^{(p-1)} \}. \tag{4}$$

We should note that our proposed network assisted solution in this work is easily adoptable to different switching definitions without any change in the theoretical model.

3.4 Fair Bitrate Allocation

LTE base stations usually schedule radio resources to multiple competing client devices at each time slot according to a proportional fairness (PF) policy [4]. More precisely, the radio resource blocks are allocated to the clients according to their link quality. In deployment scenarios with the legacy mobile DASH clients, the coordinator will not be able to allocate the bitrates to the clients in a proper way when the video traffic is encrypted. In contrast, our proposed network-assisted solution is designed for the deployment scenarios with non-legacy clients in which the mobile clients explicitly communicate with the coordinators for the proper bitrate allocation in situations when the video traffic is encrypted.

Each mobile DASH client chooses the most sustainable bitrate according to its share of resources allocated by the PF scheduler at the base station. Due to the lack of coordination among multiple DASH clients sharing the radio access link, the client-based adaptation heuristics may allocate the bitrates in an unfair manner in some situations [4]. In order

to fairly allocate the bitrates among the competing clients, as a part of our network-assisted optimized solution, we strive to select for each client the best sustainable bitrate which has the least difference from the average of bitrates allocated to the other simultaneous clients at the same time slot. More precisely, the objective of fair bitrate allocation is to minimize the overall bitrate deviations of each client i during its whole video streaming session which is obtained using the following relation:

$$F_i = \sum_{t=A_i}^{D_i} \sum_{k=1}^{K} x_{ik}^{(t)} \cdot (r_{ik}^{(t)} - \bar{r}^{(t)}). \quad (5)$$

Where $\bar{r}^{(t)} = (1/N^{(t)}) \sum_{\forall j \neq i} \sum_{1 \leq p \leq K} x_{jp}^{(t)} \cdot r_{jp}^{(t)}$ is the average bitrates of other $N^{(t)} = \sum_{\forall j \neq i} \sum_{1 \leq p \leq K} x_{jp}^{(t)}$ active simultaneous clients at time slot t. It should be noted that for each individual client, the minimization of F_i should satisfy the available resource blocks at the base station in each time slot.

3.5 Load Balancing

Balancing the utilized resource blocks among the cellular base stations is an important criteria that should be taken into account by in-network resource allocation strategies in order to avoid radio access congestion. At the application level, load balancing can be achieved, for example, through a combination of DNS and HTTP redirect mechanisms, as is commonly done with video distribution using CDN [28]. These mechanisms are applicable in our system as well because the clients need to use DNS to resolve the URIs of video chunks and request the chunks using HTTP as in any HTTP-based streaming.

The coordinators in our system balance the load between edge servers (hence, also the base stations) by mapping each client to the most appropriate (least overloaded) server so that the clients effective throughput (the actual throughput received on the client side) is maximized. More precisely, at each time slot t, we map the clients that are about to request for a new video chunk $\{i | (t - A_i)$ **mod** $C = 1\}$ to servers by solving the following optimization problem:

$$Maximize \sum_{i=1}^{S} \sum_{k=1}^{K} x_{ik}^{(t)} \cdot \hat{Thr}_{ik}^{(t)}. \quad (6)$$

Subject to $\sum_{k=1}^{K} x_{ik}^{(t)} = 1, \ \forall 1 \leq i \leq S$.

Each chunk is always downloaded entirely from the same server meaning that client to server mapping is not changed while a client is in the process of downloading a chunk.

4 Joint Optimization of QoE and Fairness

The work done by the coordinators in our system is divided into two steps: First, client to edge server mapping is done by solving the optimization problem described in Section 3.5. In the second step, the coordinator computes bitrates for each client in order to maximize QoE but in a way that preserves a certain level of fairness. We call the problem solved by the coordinators in this second step *joint optimization of QoE and fairness* and formulate it in detail in this section. It should be noted that the optimization problem is constructed on the time-slotted scheduling basis with the instantaneous knowledge about the clients throughput. Although we first assume that the clients remain stationary during the whole scheduling process, we also verify the optimization performance under the clients mobility later in the simulations.

In the QoE-fairness problem formulation, we follow the discrete time slotted DASH scheduling operation [3] with fixed time duration for each time slot. As mentioned in Section 3.3, a justifiable relation for QoE must fairly accommodate the aforementioned criteria for each individual client. To achieve this goal, we define three adjustable weighting parameters $0 \leq \rho, \omega, \theta \leq 1$ $(\rho + \omega + \theta = 1)$ to control respectively the video bitrate, the accumulated switching and the fairness. A constraint is also included in the optimization problem to avoid the happening of stalling events on each client's buffer during the whole video streaming. It is noted that the objective function does not include the initial buffer delay due to the small (negligible) impact of delay on the satisfaction of the clients.

4.1 Problem Formulation

With the model parameters defined in Section 3 and relations (1), (2), (3), (4) and (5), the joint optimization for each client i is defined as a utility maximization problem with the following integer non-linear programming formulation

$$Maximize \ U_i = \rho \cdot AQ_i - \omega \cdot E_i - \theta \cdot F_i. \quad (7)$$

Subject to

$$\sum_{k=1}^{K} x_{ik}^{(t)} = 1, \ \forall A_i \leq t \leq D_i \quad (8)$$

$$\sum_{t'=\lfloor \frac{t}{C} \rfloor \cdot C + 1}^{\lceil \frac{t}{C} \rceil \cdot C} x_{ik}^{(t')} = \{0, C\}, \ \forall 1 \leq k \leq K, A_i \leq t \leq D_i \quad (9)$$

$$\sum_{j=1}^{S} x_{jk}^{(t)} \cdot \left\lceil \frac{r_{jk}^{(t)}}{Thr_{jk}^{(t)}} \right\rceil \leq W_k^{(t)}, \ \forall 1 \leq k \leq K, \ A_j \leq t \leq D_j \quad (10)$$

$$0 < B_i^{(t)} \leq B_i^{max}, \ \forall A_i \leq t \leq D_i \quad (11)$$

$$r_{ik}^{(t)} \in R, \ x_{ik}^{(t)} \in \{0, 1\}, \ \forall 1 \leq k \leq K, A_i \leq t \leq D_i. \quad (12)$$

Note that in the above optimization model, the equality (2) is also added to the set of constraints. The variables $x_{ik}^{(t)}$ and $r_{ik}^{(t)}$ are respectively the binary and integer decision variables, while, the values of other parameters are known in advance. The objective function (7) aims to maximize jointly the QoE of DASH client i and the fairness in bitrate allocation to the client. Constraint (8) states that at any time instant t, the DASH client is allocated to only one server for downloading its current video chunk and (9) enforces that the client receives one complete chunk of video upon its access to the allocated server. Constraint (10) ensures that at each time instant t, the total volume of allocated resources to the requested clients by each base station does not exceed the instantaneous available resources at the base station. Constraint (11) guarantees that no stalling happens during the whole streaming duration from the arrival to departure times. And finally, constraint (12) specifies that the discrete allocated bitrate for a requested chunk of video from each edge server belongs to the set of available bitrates and also the client to server mapping is a binary decision variable.

4.2 NP-Hardness

The joint optimization of load balancing, QoE and fairness formulated in problem (7)-(12) belongs to the class of NP-hard problems due to the existence of integer decision variables and hence the exhaustive possible enumerations for the

solution space. Before detailing the proposed scheduling algorithm, we prove first the NP-hardness of the problem in the following section.

The NP-hardness of the problem formulation (7)-(12) can be proved through first showing the hardness of the special case of the problem when each client is assigned with the maximum bitrate which is not more than its effective throughput. In other words, we fix the decision variable for bitrate selection in problem (7)-(12) and hence, the only decision variable is the allocation of the clients to the appropriate servers. The NP-hardness of the special case immediately implies the hardness of the general problem in which the clients can be assigned with a rate which belongs to a discrete set of available bitrates.

We prove the NP-hardness of the special case at a given time slot t via the reduction from the multidimensional knapsack problem (MDKP) as follows: Given m items $\{c_1, c_2, \ldots, c_m\}$ and K knapsacks, we consider an instance of MDKP in which m copies are available for each item. The copies are generated by combining the item with the others and considering it as a single item and each copy has different value. For example for the first item, m copies $\{c_1\}$, $\{c_1, c_2\}, \ldots, \{c_1, c_2, \ldots, c_m\}$ are available. The problem is the allocation of items to knapsacks to maximize the total profit subject to the constraints that each item should appear in exactly one of the allocated copies and each knapsack has the capacity of maximum one item.

Now, given any instance of MDKP can be mapped to an instance of our problem as follows: The set of m items and K knapsacks are mapped to the set of clients and servers and the profit maximization in MDKP is equivalent to utility maximization (objective value (7)) in our problem. Each copy of the item matches with the joint allocation of the corresponding client with other clients to a given server. It is obvious that joint allocation of client with different set of other clients results in different utility values since the share of throughput dynamically changes based on the number of other clients allocated to the same server. Furthermore, the mapping satisfies the constraint that each client should be allocated to exactly one server. Now, since the multidimensional knapsack problem is NP-complete [34], therefore, the special case of our problem is also NP-complete unless $P = NP$. This in turn implies the NP-hardness when all time slots are concerned as well as the hardness of the general problem formulation (7)-(12) in which the allocated bitrate to each client is chosen from a discrete set.

5 Online Optimization Algorithms

In this section, we present efficient algorithms for solving the optimization problems described in the previous sections. These algorithms are designed for online execution by the coordinators in our system. In offline situations, where all the information of clients (their arrival and departure times, the link quality) are known in advance, a straightforward brute-force search strategy can be used to investigate all possibilities of allocating clients to video servers and select the one with maximum achievable utility. However, the complexity of exhaustive approach grows dramatically with the increase in the number of servers or clients making it impractical for DASH scheduling at large scale. To reduce the complexity, we devise an efficient online and greedy-based algorithm which runs by the coordinators with the clients data obtained using the MEC facilities. The body of the proposed algorithm named as *GreedyMSMC* has been illustrated in Algorithm 1.

Algorithm 1. GreedyMSMC

Input: $|T|, S, K, R$: Number of scheduling time slots, number of DASH clients and edge servers and the set of available discrete bitrates.
Output: Binary allocations $x_{ik}^{(t)}$ and integer bitrate allocation $r_{ik}^{(t)}$ and U_i for each client $1 \leq i \leq S$, server $1 \leq k \leq K$ and time slot $1 \leq t \leq |T|$
for each time slot $1 \leq t \leq |T|$ **do**
  for each client $1 \leq i \leq S$ such that $A_i \leq t \leq D_i$ **do**
    Allocate client i to server k according to (13)
  for each client $1 \leq i \leq S$ such that $A_i \leq t \leq D_i$ **do**
    if $t == A_i$ **then**
      Initialize $BufferStatus$ and L_i
    if $(t - A_{ij}) \bmod C \neq 1$ **then**
      Allocate the same bitrate to client i as with the bitrate at time slot $t-1$;
      Update $B_i^{(t)}, BufferStatus, L_i$;
    if $(t - A_i) \bmod C == 1$ **then**
      if $BufferStatus == False$ **then**
        Run Subroutine Startup Phase;
      if $BufferStatus == True$ **then**
        Run Subroutine Steady State;
    if $t == D_i$ **then**
      $totalUtility = totalUtility + U_i$
Return $totalUtility$;

5.1 Client-to-Server Mapping

Based on the discrete time slot scheduling, the proposed algorithm takes the dynamic arrival of clients into account. Since the clients to servers mapping problem is NP-Hard, we rely on a heuristic which by taking into account the topology of the network decides on either allocating the mobile client to its nearest base station or allocating it in a greedy manner to the base station with highest achievable throughput. At each time slot, the heuristic uses the average distance of all the clients to their nearest BS as the threshold to decide on the server allocation. More precisely, assume $d_{min}^{i(t)} = min\{d_{is}^{(t)}, 1 \leq s \leq K\}$ and $d_{min}^{(\bar{t})}$ represent the distance of client i to the nearest BS and the average distance of all clients to their nearest BS at time slot t. The coordinator at each time slot resolves the load balancing based on whether the minimum distance of the client is above the average or it is below the average. In other words, each active client i is assigned to server k at time slot t where

$$k = \begin{cases} \{k' | x_{ik'}^{(t-1)} = 1\}, & \text{if}(t \bmod C) \neq 1 \\ arg_{1 \leq s \leq K} \mathbf{min}\{d_{is}^{(t)}\} & \text{if}(t \bmod C) = 1 \\ & \quad \textbf{AND } d_{min}^{i(t)} \geq d_{min}^{(\bar{t})} \\ arg_{1 \leq s \leq K} \mathbf{max}\{\hat{Thr}_{is}^{(t)}\} & \text{if}(t \bmod C) = 1 \\ & \quad \textbf{AND } d_{min}^{i(t)} < d_{min}^{(\bar{t})}. \end{cases} \tag{13}$$

Client to server mapping is in accordance with constraint (9) which states that the server allocation is decided at the beginning of each video chunk and it remains unchanged during the process of downloading the chunk.

5.2 Bitrate Selection with Parameter Auto-Tuning

One of the critical challenges in designing an efficient algorithm for the optimization problem is finding appropriate

values for the weighting parameters in the objective function. Since considering the fixed values for the weighting parameters may not be optimal when different scenarios or network topologies are concerned, we rely on the strategy of dynamically adjusting the weighting parameters at each time slot. This design strategy alleviates the need for manual parameter tuning and, consequently, makes the system deployment easier.

Subroutine 1. Startup Phase

if $(t - A_i) \leq C$ **then**
  Allocate the highest available bitrate;
  Update $B_i^{(t)}$, $BufferStatus$;
Compute the $estThrGreedy$ from the BS allocation according to (13)
Compute switching thresholds Thr_T, Thr_B and δ_S;
for each bitrate $r \in R$ in decreasing order **do**
  if allocation of r satisfy (10) **AND** $r \leq$ **max**$(estThrGreedy, \hat{Thr}_{ik}^{(t)}, B_i^{(t)})$
    if $|r - r_{ik}^{(t-1)}| \leq \delta_S$ **AND** $1 - |r - \bar{r}|/(R_{max} - R_{min}) >= \delta_F$
      $r_{ik}^{(t)} = r$; **Break;**
if $r_{ik}^{(t)} == 0$ **then**
  for each bitrate $r \in R$ in decreasing order **do**
    if allocation of r satisfy (10) **AND** $r \leq$ **max**$(estThrGreedy, \hat{Thr}_{ik}^{(t)}, B_i^{(t)})$
      If $|r - r_{ik}^{(t-1)}| \leq \delta_S$
        $r_{ik}^{(t)} = r$; **Break;**
if $r_{ik}^{(t)} == 0$ **then**
  for each bitrate $r \in R$ in decreasing order **do**
    if allocation of r satisfy (10) **AND** $r \leq$ **max**$(estThrGreedy, \hat{Thr}_{ik}^{(t)}, B_i^{(t)})$
      $r_{ik}^{(t)} = r$; **Break;**
Update weighting parameters ρ, ω θ;
Compute AQ_i, E_i, F_i and U_i according to respectively (1), (4), (5) and (7);
Update $B_i^{(t)}$;
if $B_i^{(t)} = B_i^{max}$ **then**
  $BufferStatus = True$; $L_i = t - A_i$;
Return U_i;

For each active client, GreedyMSMC algorithm selects the highest available bitrate for the first chunk of the video. For the subsequent video chunks in either startup or steady phase, the algorithm then chooses the most sustainable bitrate which results in less switching level and high fairness value. In other words, the proposed algorithm considers two known threshold values δ_S and δ_F for respectively the switching level and fairness index. The switching threshold δ_S is determined based on the switching level when relying only on the estimated throughput or the switching level when relying only on the buffer level. In other words, the switching levels based on the estimated throughput and buffer level are represented by thresholds Thr_T and Thr_B, respectively, and the switching threshold δ_S is stated in terms of Thr_T and Thr_B. As pointed out in the Startup phase of the algorithm, the quantity $1 - |r_{ik}^{(t)} - \bar{r}|/(R_{max} - R_{min})$ which takes a value between 0 and 1 is also computed as the fairness index associated with the selected bitrate $r_{ik}^{(t)}$. Here, the variable $\bar{r}$ denotes the average bitrate of other simultaneous clients and R_{max}, R_{min} are respectively the maximum and minimum bitrates in set R. It is noteworthy to mention that although the selection of the aforementioned thresholds still depends on the network topology, the proposed algorithm however has the minimum need for parameter tuning since the major weighting parameters are automatically adjusted by the algorithm.

Subroutine 2. Steady State

Run the same code lines (4)-(18) as in the subroutine Startup Phase;
Update weighting parameters ρ, ω, θ;
Compute AQ_i, E_i, F_i and U_i according to respectively (1), (4), (5) and (7);
Update $B_i^{(t)}$;
if $B_i^{(t)} == B_i^{(max)}$ **then**
  $BufferStatus = True$; $L_i = t - A_i$;
Return U_i;

In the decreasing order of the bitrates, GreedyMSMC algorithm first seeks for a bitrate which results in a switching which is not more than the switching threshold δ_S and yields a fairness value which is greater than the fairness threshold δ_F. If there is no such bitrate available, it then looks for the bitrate which yields a switching level less than the switching threshold δ_S. If still there is no such bitrate available, the algorithm then chooses the maximum sustainable bitrate. After the bitrate selection, the weighting parameters of the objective function at the current time slot are computed based on how far the selected bitrate is from the best possible bitrate. More precisely, for the weighting of the average bitrate, how far is the selected bitrate from the maximum possible bitrate and for the case of switching how far it is from the bitrate which gives no switching. Furthermore, the fairness weighting is determined based on how far is the selected bitrate from the average bitrates of other simultaneous clients at the same time slot. After updating the weighting parameters, the algorithm returns the local utility of the client which is obtained using the objective function (7).

5.3 Computational Complexity

Most of the computations of GreedyMSMC algorithm happen in overall $\frac{|T|}{C}$ times during the whole video streaming session of all clients where, $|T|$ and C are respectively the number of time slots and the chunk size. According to the relation (13), the server allocation for each client has the time complexity of order $O(max(K + S, K \cdot S + K)) = O(K \cdot S + K)$ in the worst case, where K is the number of edge servers. Furthermore, in the execution of Startup phase, the computation of estimated throughputs takes $O(C)$ and the evaluation of both switching and fairness thresholds takes $O(|R|)$ time, where $|R|$ is the number of available bitrates at each edge server. Similarly, the execution of Steady State phase results in the complexity of $O(C + |R|)$. Putting all the above together yields with S clients an overall worst case complexity $O(\frac{|T|}{C} \cdot S \cdot (K \cdot S + K + C + |R|))$, which is significantly less than the upper bound $O(\frac{|T|}{C} \cdot K^S \cdot R)$ of the exhaustive brute-force search strategy.

6 Methodology for Simulation-Based Comparative Evaluation

In this section, we describe our approach to evaluate the system performance. In particular, we describe the

simulation setup as well as the client-based adaptation approaches and another existing SAND-DASH solution that we compare our edge computing assisted bitrate adaptation solution against.

In our simulations, we implement a typical behavior of DASH video streaming clients in a mobile edge computing scenario and do not focus on any specific type of devices. At first, most of the simulations are performed under the scenario that the clients remain stationary during the whole scheduling process while, the in last part of the simulations, the performance of the algorithm is evaluated under the clients mobility using the LTE radio access link level traces. In the mobility case, we do not implement any wireless channel model or LTE protocol and only implement the optimization solution (which is acting on top of the LTE network) using the signal-to-noise ratio (SNR) traces obtained from the third-party simulator SimuLTE [38].

6.1 Simulation Setup

We consider the scheduling of DASH clients during one hour i.e. with $|T| = 3600$ number of time slots and the duration of $\Delta t = 1\ second$ for each slot.

For the network setup, we consider a rectangular area with size $400\ \mathrm{m} \times 1\ \mathrm{km}$ in which the base stations and the mobile clients are uniformly located depending on the considered topology. The clients arrival time is uniformly distributed within the first $20\ \mathrm{min}$ and they depart after an active streaming session which its length duration is chosen from the uniform interval $[20\ \mathrm{min}, 40\ \mathrm{min}]$. The video is divided into $C = 5s$ chunks and each chunk is available in ten different bitrates $[40\ \mathrm{kbps}, 60\ \mathrm{kbps}, 90\ \mathrm{kbps}, 110\ \mathrm{kbps}, 120\ \mathrm{kbps}, 150\ \mathrm{kbps}, 160\ \mathrm{kbps}, 190\ \mathrm{kbps}, 200\ \mathrm{kbps}, 220\ \mathrm{kbps}]$ with the same replication at each edge server. In the simulations, we consider 12 number of servers unless otherwise stated, and the number of clients vary from 100 to 500. As for the radio access, the maximum transmission power of each base station is fixed at $3.6\times\ 10^5\ \mathrm{mW}$ and the loss exponent value of $\alpha = 2$ is considered in the path attenuation model. With one second time slot duration and total available per slot bandwidth $U[45\ \mathrm{KHz}, 90\ \mathrm{KHz}]$, the total LTE downlink resource blocks per slot at each base station follows the uniform distribution $U[50, 100]$ [33].

We also note that at each part of simulation, the average of the results taken over 20 number of iterations with the confidence interval of 95 percent has been shown for each simulation instance. The list of parameters used in the simulations and their corresponding values have been summarized in Table 2.

TABLE 2
Simulation Parameters and Their Values

Simulation Parameter	Corresponding Value
Number of clients	100-500
Number of servers	12
Number of time slots	3600
Time slot duration	1 second
Bandwidth	U[45 KHz, 90 KHz]
Clients arrival	U[0s, 1200s]
Streaming duration	U[1200s, 2400s]
Chunk size	5s
Path loss exponent (α)	2
Max. Tx power of BS	3.6×10^5 mW
Number of downlink RBs	U[50,100]

6.2 Client-Based Adaptation Approaches

In this section, we compare our network-assisted method (the proposed algorithm) with two client-based adaptation strategies which are buffer based adaptation (BBA) and rate based adaptation (RBA) with the following descriptions. We choose the switching and fairness thresholds $\delta_S = (Thr_T + Thr_B)/2$ and $\delta_F = 0.5$ used by our algorithm in the following sections, unless, we explicitly mention about the new thresholds. It is also noted that the client-based adaptation approaches assign each client to the closest base station during the whole video streaming.

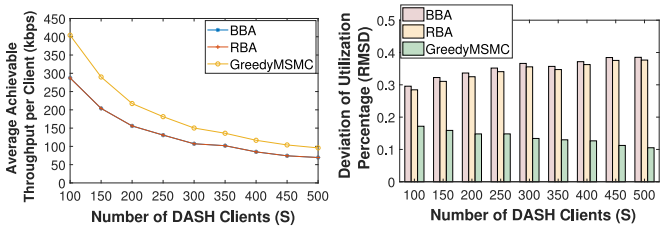
Buffer Based Adaptation [7] means that each client independently selects the bitrate for the next chunk to download based on instantaneous buffer occupancy level, i.e. the amount of video data in the playback buffer of the client at each time slot. The heuristic allocates the highest bitrate for the first chunk and then looks at the current client's buffer level to decide on the bitrate for the next video chunk to be downloaded. According to the set of available bitrates, the heuristic considers nine different thresholds $(\sum_{q=1}^{p} R_q / \sum_{q=1}^{|R|} R_q) \cdot (B_i^{(max)}),\ \forall 1 \leq p \leq 9$ for client i, where R_q and $|R|$ are respectively the qth bitrate in the sorted version of set R (increasing order) and the size of set R. Depending on the buffer level, the heuristic then chooses the most closest bitrate from the allocated server.

Rate Based Adaptation [8] works so that each client chooses the highest sustainable bitrate among the available ones based on the throughput obtained when downloading the previous m chunks. In particular, RBA computes a moving average of the download rate of the last consecutive m chunks (estimated throughput) to determine the bitrate for the next video chunk to be downloaded. The bitrate for chunk $i > m$ is obtained using the moving average $(1/m)\sum_{j=i-m}^{i-1} r_j$. Note that in the implementation of RBA, we set $m = 3$ as the number of previously observed chunks when estimating the achievable throughput for current chunk.

6.3 Another SAND-DASH Solution

We also compared the performance of our solution to an existing SAND-DASH solution presented in [11]. The proposed network assisted solution in this work considers the scheduling of multiple competing DASH clients for accessing the video contents from a single server and over a bottleneck radio access link. An optimization problem with simple utility function is solved to determine the allocation of the limited resources to the clients. To have a fair comparison under the same system setup, we adopt the SAND-DASH solution in [11] to the case of multiple edge servers such that as the default server assignment, the coordinator first allocates each client to the nearest base station. Then, the coordinator employs the greedy algorithm for solving the optimization problem (7)-(12) with equal weightings $\rho = \omega = \theta = 1/3$.

It is noteworthy to mention that applying the algorithmic approach used in [11] directly to our optimization problem leads to the solutions which are worse than the solutions when applying the greedy algorithm. This is due to the imperfectness of the objective function presented in [11]. Furthermore, increasing each of three weighting parameters in the objective function (7) may improve the corresponding metric compared to our algorithm, however, it causes the significant performance drop in two other metrics. Therefore, with the above-mentioned setup, we aim to have the



(a) Average throughput (b) Resource utilization

Fig. 2. Comparison between GreedyMSMC and the client-based DASH heuristics in terms of the achievable throughput and resource utilization.

best possible configuration for the SAND-DASH strategy of [11] and then compare its performance with our algorithm.

7 Evaluation of QoE and Fairness

7.1 Effect of Load Balancing

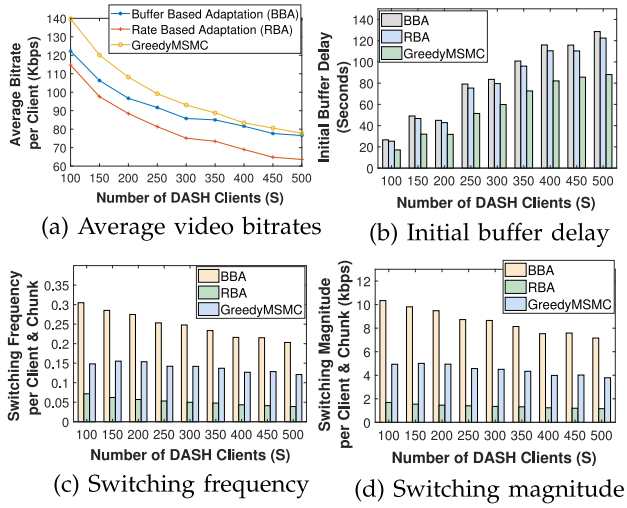
Fig. 2a shows the comparison between GreedyMSMC algorithm and two client-based adaptation heuristics in term of the average achievable throughput per client for 20 runs of the simulation. As we can see, both client-based heuristics achieve the same throughput (due to the same server allocation) which is less than the throughput obtained using GreedyMSMC algorithm. The reason for the difference is that allocating each client merely to the closest base station during the whole session of video streaming will result in lowering the average throughput especially under the high dynamic arrival and departure of the clients. In contrast, GreedyMSMC's client-to-server mapping is able to improve the situation.

Fig. 2b shows the comparison between the algorithms in term of resource utilization. For each algorithm, with the average taken over 20 runs, the percentage of utilized RBs (ratio between the utilized and the total available resources) has been first computed for each base station and then, the root mean square deviation (RMSD) value among the BSs have been shown in Fig. 2b. As we see, the proposed algorithm results in significantly reducing the utilization deviation (in average around 60 percent reduction) among the BSs, which demonstrates the utility of the explicit load balancing step of our solution.

7.2 QoE Comparison to Client-Based Approaches

We next compare the approaches using the QoE metrics. Fig. 3a shows that the network assisted adaptation yields 10 and 20 percent higher average video bitrate compared to BBA and RBA, respectively. The main reason is the higher effective throughput values (cf. Fig. 2a). However, the difference to BBA becomes marginal with large number of clients indicating that BBA works equally well under low effective throughput. On the other hand, GreedyMSMC chooses the bitrates in decreasing order of size so that the time duration to reach a full buffer reduces compared to the client-based heuristics, which leads to roughly 25 percent reduction in initial startup delay compared to client-based heuristics (Fig. 3b).

Figs. 3c and 3d show the frequency and the magnitude (kbps) of bitrate switching per client and chunk duration during the whole video streaming session of all clients. To understand how to read the y-axis values, consider RBA with 100 clients: On average, a quality switch happens between roughly 6 percent of its chunks for a given client and the average magnitude of the switch is around 1.5 kbps. BBA leads to most frequent and largest quality switches.



(a) Average video bitrates (b) Initial buffer delay

(c) Switching frequency (d) Switching magnitude

Fig. 3. The comparison between GreedyMSMC and the client-based heuristics in terms of QoE metrics.

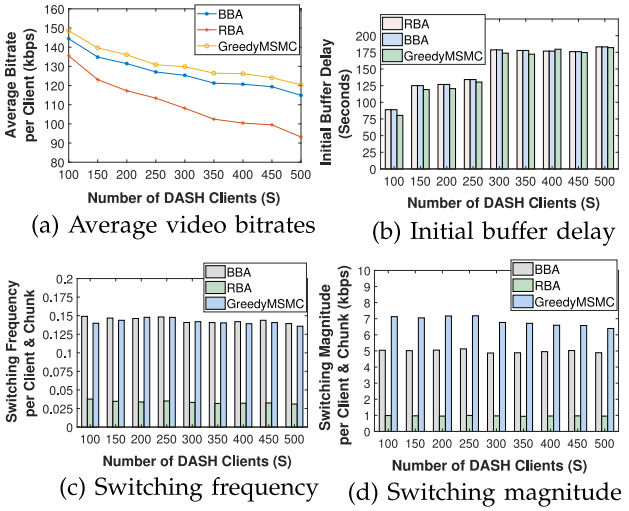
The reason is that the buffer occupancy level can fluctuate a lot under churn which results in larger number of bitrate switching per client. RBA turns out to be more effective in reducing the switching than GreedyMSMC, but it comes with the cost of significantly lower video bitrates (Fig. 3a) and higher buffer delay (Fig. 3b).

7.3 Effect of Network Topology

We also studied the different solutions using a different network topology. For this simulation, the $x-$axis of the location for both BSs and clients remain same while the $y-$axis of the location for BSs and clients are chosen from the uniform distributions $U[100, 150]$ and $U[0, 200]$, respectively. In other words, the locations of the base stations are shifted to the middle of clients so that the distances of the clients to BSs are more unevenly distributed compared to the previous topology.

The results of this comparison in terms of QoE metrics are shown in Figs. 4a, 4b, 4c, and 4d. Network assisted adaptation performs better than the client-based heuristics in terms of average bitrate and startup delay. Since clients are located less evenly compared to the previous topology, the switching threshold $\delta_S = Thr_B$ has been chosen this time in order to achieve higher average bitrate. This in turn increases the switching frequency and magnitude for our algorithm (Figs. 4c and 4d).

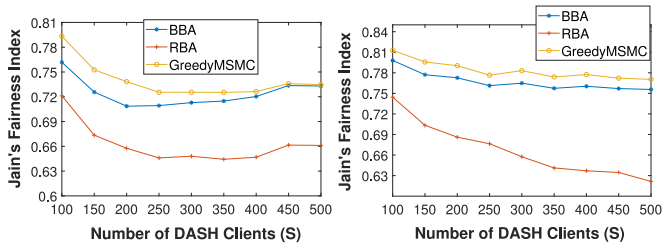
Fig. 5 shows the fairness of the different adaptation approaches under two different network topologies. We use the Jain's fairness index [3] which is defined as $JF = (\sum_i \bar{r}_i)^2/(S \cdot \sum_i \bar{r}_i^2)$, where S is the total number of clients and $\bar{r}_i$ denotes the average bitrate of client i during its streaming session. As expected, GreedyMSMC clearly improves fairness compared to client-based adaptation. We also note that in average the fairness drops as the number of clients increases, which is due to the fact that with more number of clients, the variations in the bitrates of simultaneous clients at each time slot increases. This in turn means that for each client, the likelihood of choosing a bitrate which is very close to the average will be reduced. However, this behavior is slightly different for the first topology as the fairness shows again small increase and then converges to a stable point. The reason is that the locations of the clients are evenly distributed under the first topology and therefore, with the small achievable throughput when



(a) Average video bitrates (b) Initial buffer delay

(c) Switching frequency (d) Switching magnitude

Fig. 4. The comparison between GreedyMSMC and client-based heuristics for different network topology.



(a) First topology (b) Second topology

Fig. 5. Comparison between GreedyMSMC and client-based DASH heuristics in term of Jain's fairness index.

the number of clients get congested, it is highly expected that some of them get very similar bitrates. This situation however does not hold under the second topology in which the clients' locations are unevenly distributed.

It is also noteworthy to mention that although the fairness values of GreedyMSMC and BBA converge for large number of clients, using our algorithm results in higher average bitrate (Figs. 3a and 4a).

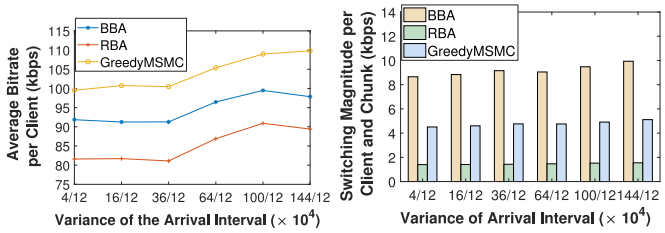
7.4 Effect of Inter-Arrival Time

In this section, we study the impact of client inter-arrival time on QoE. We consider 200 clients and six different uniform arrival patterns by varying the arrival time. The arrival of clients to the system happens during the period that has mean of 600 seconds and variance of $(4i^2 \times 10^4)/12, 1 \leq i \leq 6$.

Figs. 6a and 6b show the resulting bitrates and quality switching for different arrival patterns. The average bitrate increases when the variance in the arrival pattern increases. The reason is that increasing the variance increases the duration of the period during which the clients arrive to the system, which in turn somewhat decreases the competition for the shared radio resources. On the other side, the switching magnitude keeps slightly increasing as the duration of arrival interval increases (Fig. 6b) which is due to the fact that with larger duration of arrival, the chunks of different clients will have more interleaving such that the number of times that clients have to switch to different bitrates will increase.

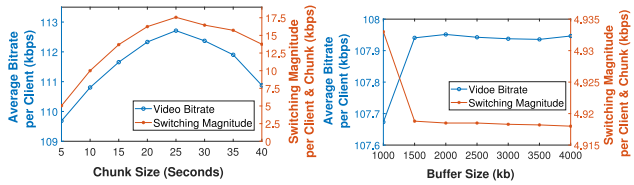
7.5 Effect of Chunk and Buffer Sizes

We next investigate the impact of the chunk and buffer size on the average video bitrate and bitrate switching. We



(a) Average bitrate (b) Switching magnitude

Fig. 6. Comparison between GreedyMSMC and client-based DASH heuristics for different arrival intervals.



(a) Impact of chunk size (b) Impact of buffer size

Fig. 7. Impact of chunk and buffer size on the average video bitrate and switching magnitude per client.

consider 200 clients and average the results over 20 simulation runs. Fig. 7a shows that increasing the chunk size initially causes the increase in video bitrate. The reason is that until some time durations, the bitrate selection logic of the algorithm is not sensible to the variations in the throughput and hence, the estimated throughput slightly increases. That means by increasing the chunk size up to some point, the average bitrate slightly improves. However, the bitrate starts decreasing as the chunk size gets larger. This is because the larger the chunk size, the less frequently the bitrate selection algorithm has the ability to react to fluctuation in available network resources. A similar pattern is also observed for the switching magnitude as the chunk size increases. This result reveals that by varying the chunk size, the improvement in video bitrate comes however with the cost of higher bitrate switching.

In Fig. 7b, the results for the impact of increasing the buffer size on the average bitrate and switching are shown. As we noticed from the results, the increases in average bitrate or decrease in bitrate switching is small when the buffer size increases. As the matter of fact, the reason for such phenomena is that our scheduling algorithm aims to avoid the stalling event under any circumstances which leads to negligible improvement when the variations in the set of available bitrates and the initial buffering delay are small.

7.6 Comparison to Another SAND-DASH Solution

We next compare the performance of the proposed network assisted adaptation with an existing SAND-DASH solution, which we described in Section 6.3. The results are shown in Figs. 8a, 8b, 8c, and 8d. GreedyMSMC algorithm improves the average bitrate, fairness and resource utilization compared to the alternative network assisted solution by in average (over different number of clients) about respectively 20, 3, and 60 percent. It is noteworthy to mention that these improvements are obtained by the help of clients to server mapping strategy employed by GreedyMSMC. From the switching point of view, it is observed that for larger number of clients, the alternative SAND-DASH solution shows however slightly better performance compared to our solution which is due to optimizing the switching under the allocation of clients to the nearest BS. Despite of this

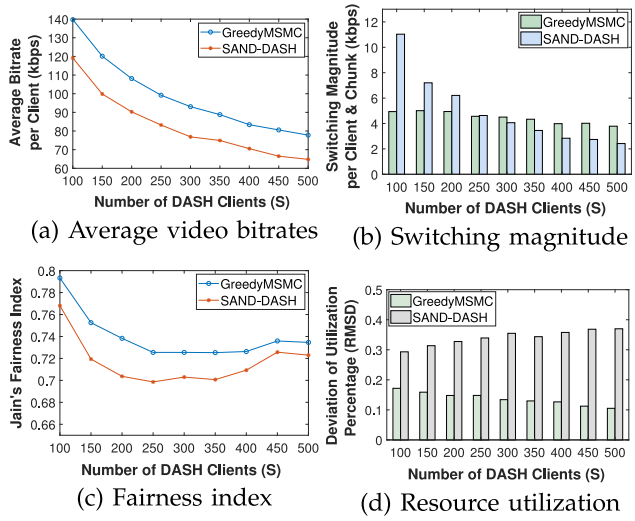


Fig. 8. The comparison between GreedyMSMC and the SAND-DASH approach in terms of QoE metrics.

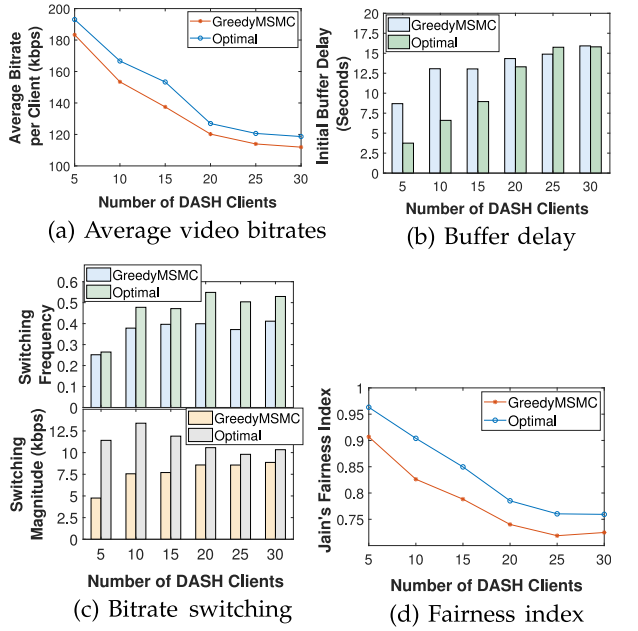


Fig. 9. The comparison between GreedyMSMC and the optimal scheduling for small number of clients.

improvement, the switching level of both solutions becomes stable when the number of clients gets larger.

It is noted that our network-assisted adaptation solution assumes the availability of the same replication of all video chunks at every edge server without the storage constraint. As an implication, our algorithm obviously reveals better performance in terms of QoE metrics compared to those network-assisted strategies which impose the limitation on the cache storage at the edge server. The reason is due to the fact that retrieving some of the locally missed chunks from the origin server increases the delay and hence degrading the QoE of the clients.

8 System Optimality and Scalability

In this section, we first examine the optimality of the heuristic-based algorithms of our system by comparing their performance to the optimal solutions in limited complexity scenarios. We then pay our attention to the scalability of the system by studying the performance of the decentralized implementation of the algorithm in terms of both utility gain and the computation time compared to the centralized one.

8.1 Comparison with Optimal Scheduling

As mentioned in Section 4.2, the optimization problem (7)-(12) belongs to the class of NP-hard problems and hence, finding the optimal solution for large instances is computationally intractable. However, we are interested in this part of simulation to evaluate the performance of the proposed heuristic compared to the optimal scheduling for small instances of the problem. Here, we assume that the number of clients ranges from 5 to 30 and three edge servers are deployed. In order to further reduce the computation time, we relax the assumption of integer bitrate allocation for both strategies, such that the allocated bitrates to each client can take any continues value between 40 kbps and 220 kbps. Furthermore, we also neglect here the constraint on avoiding the stalling on the buffer for the sake of simplicity in the implementation. For finding the optimal clients to servers mapping according to relation (13) as well as the optimal bitrate allocation, we use the Gurobi optimization solver [39] and implement the bitrate adaptation algorithm in Java with subroutins for calling Gurobi solver. To have the fair comparison, for the optimal scheduling, the weighting parameters in the objective function (7) are equally set to $\rho = \omega = \theta = 1/3$.

As the comparison result in Fig. 9a shows, for each client instance the average of optimal bitarte over 20 runs of simulation is no more than 1.2 times of the bitrate returned by the proposed algorithm GreedyMSMC. This in turn implies that considering bitrate as the comparison metric, GreedyMSMC is an $1.2-$approximation algorithm for the utility maximization problem (7)-(12). Considering the buffer delay metric, GreedyMSMC algorithm yields an initial buffer delay which is very close to the optimal one as the number of clients increases. This is due to the fact that buffer delay was considered as less important QoE factor and hence neglected in the optimization problem. From the bitrate switching point of view, the results in Fig. 9c suggest that using GreedyMSMC both switching frequency and magnitude get closer to the optimal and tend to exceed it as the number of clients becomes larger. Fig. 9d also confirms that GreedyMSMC is again $1.2-$approximation factor for the optimization problem considering the fairness index as the comparison metric.

It is also noteworthy to mention that each of the QoE metrics can be further optimized by increasing the corresponding weighting parameter in the optimization problem.

8.2 Decentralized Implementation

In this section, we want to investigate the impact of the decentralized implementation of GreedyMSMC algorithm on the performance gain from the average bitrate and computation time points of view. In the decentralized implementation, we restrict the search space for the clients to server mapping by considering multiple accessibility levels (denoted by N). For instance, $N = 2$ means that for each client, the search space for server selection is limited to the first $K/2$ closest edge servers where K is the total number of servers. In this way, the clients to server mapping and the scheduling process are both performed by the local coordinators at the edges. With the same number of clients and servers as the previous sections, four different accessibility

TABLE 3
Achievable Bitrate ($kbps$) for Different Accessibility Levels Using the Decentralized Scheduling

Number of Clients	$N = 2$	$N = 4$	$N = 6$	$N = 12$
100	137.5508	136.3277	131.4192	121.1129
150	121.4348	120.0969	115.3875	105.3555
200	109.1368	107.7879	103.6461	94.6087
250	100.3953	99.1173	95.7507	87.6662
300	93.6886	92.6487	89.5783	82.3203
350	88.2712	87.5216	84.6605	78.0915
400	84.0251	83.2511	80.5823	74.7188
450	80.4175	79.7770	77.3864	72.0802
500	77.6153	77.0341	74.7754	69.9263

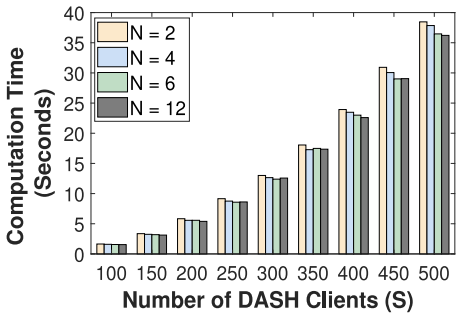


Fig. 10. Comparison between different accessibility levels in term of computation time.

levels $N = \{2, 4, 6, 12\}$ are considered. The average video bitrate generated by decentralized algorithm for different number of clients and N values have been shown in Table 3.

As it is seen, for different number of clients, the average bitrate decreases as the accessibility level increases. The reason is that for each client, the algorithm will have the local information (less choices for server selection) with larger accessibility levels which consequently leads to loosing some gain that could be obtained considering all edge servers. It is also noted that with the case of $N = 12$, the GreedyMSMC algorithm behaves in the similar way as the client-based adaptation heuristics in which each client is allocated to the nearest BS during the whole video streaming session. We have also shown in Fig. 10 the impact of accessibility level on the computation time taken by decentralized scheduling algorithm. As it is seen from the result, the running time slightly decreases as the accessibility level increases which is because of the computation tasks at the coordinators for server mapping decision reduces when the search space for server selection decreases.

It is noted that we do not solve the optimization problem in a distributed way but in fact, the similar optimization problems but with significantly reduced search space are handled with distributed implementation and are executed by the local coordinators.

9 Experiment with LTE Simulator

To obtain the simulation results presented in the previous sections, we used a relatively simple model to characterize the radio link data rate as a function of distance of the client from the base station. In order to make sure that the results are not biased because of using such a simple model, we next study the system behavior with radio access level traces from a full-fledged LTE simulator.

TABLE 4
SimuLTE Parameters and Their Values

SimuLTE Parameter	Corresponding Value
Number of UEs	50
Number of eNodeBs	5
UE antenna gain	0 dBi
eNodeB antenna gain	18 dBi
UE speed	8.3 mps
Maximum Tx power per UE	26 dBm
Channel bandwidth	5 MHz
Number of downlink RBs	28
Scheduler	Proportional Fairness
Channel model	Urban Macrocell
Shadowing	Disabled
Simulation time	300s

9.1 Setup Description

For the network setup, we use SimuLTE [38] program which is integrated into OMMNET++ simulator and enables the communication between mobile users known as UEs and the LTE cellular base stations known as eNodeBs. Under the urban macrocell channel model specification [3GPP TR 36.814 V.9.0.0 2010], we have obtained the downlink SNR values (in dB) of 50 mobile users connecting to 5 eNodeBs each associated with one video server. We follow the client mobility with constant speed of 8.33 *mps* as considered in [29] during 300 seconds (time slots) of video streaming session. Given the downlink SNR values, the theoretical throughput of mobile users are then obtained using the relation (14) which is an approximation of Shannon upper bound. Note that the values of parameters α, SNR_{min}, SNR_{max} and Thr_{max} are set to respectively 0.6, -10 dB, 23 dB and 4.4 bps/Hz according to the downlink specifications reported in [40].

$$Thr(SNR) = \begin{cases} 0, & SNR < SNR_{min} \\ \alpha \cdot log_2(1 + 10^{\frac{SNR}{10}}), & SNR_{min} \leq SNR < SNR_{max} \\ Thr_{max}, & SNR \geq SNR_{max}. \end{cases} \tag{14}$$

The set of video bitrates are available at ten different qualities 15 Mbps, 17 Mbps, 22 Mbps, 26 Mbps, 30 Mbps, 35 Mbps, 38 Mbps, 43 Mbps, 45 Mbps and 50 Mbps. The arrival time of each client is randomly chosen form the uniform interval $U[0\text{ s}, 10\text{ s}]$ and with a streaming session of $300s$, its departure time follows the uniform distribution $U[300\text{ s}, 310\text{ s}]$. Incorporating the emulated throughput values into the Matlab simulator, the proposed algorithm is compared against the client-based DASH heuristics in terms of QoE metrics, fairness and resource utilization. The parameters used in SimuLTE and their corresponding values have been summarized in Table 4.

9.2 QoE and Fairness Comparison

We have compared GreedyMSMC with two client-based DASH heuristics BBA and RBA in terms of QoE metrics using the throughput values which were obtained from the LTE simulator.

As the first result, Fig. 11a reveals that the clients receive on average higher effective throughput using the proposed algorithm compared to both BBA and RBA heuristics. This is because of the clients to server mapping strategy used by GreedyMSMC algorithm which aims to balance the utilized resource blocks among the base stations. Similarly, the

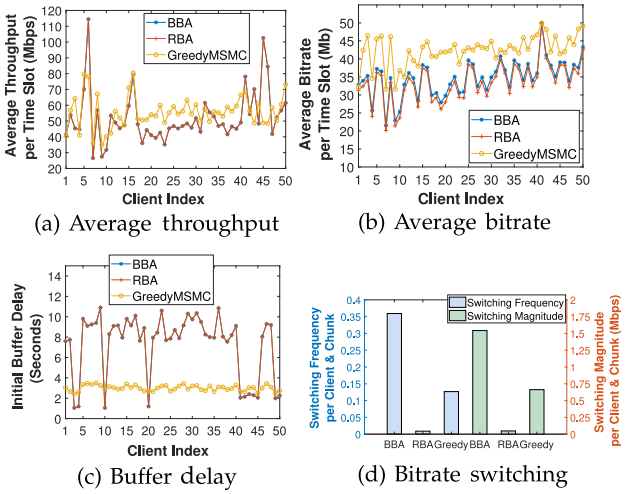


Fig. 11. Comparison between GreedyMSMC and client-based adaptation heuristics in terms of QoE metrics using SimuLTE data.

higher effective throughput obtained using GreedyMSMC algorithm helps the mobile clients to receive on average higher bitrate at each time slot as shown in Fig. 11b. We note that when representing the throughput/bitrate per client index, for few of the clients the achievable throughput using GreedyMSMC is lesser than both BBA and RBA, however, those clients obtain higher bitrates when using our algorithm. The reason is that in the bitrate selection phase, our algorithm also takes into account the instantaneous throughput while, the client-based heuristics rely on only either the estimated throughput or the buffer level. That means in some cases the allocated bitrates by the client-based heuristics may not reflect the instantaneous throughput. For the initial buffer delay, the result in Fig. 11c confirms that the mobile clients tolerate an initial delay on their playback buffer which is in average less than half of the delay caused by BBA and RBA. Again, the higher achievable effective throughput using GreedyMSMC algorithm helps the clients to reach faster the buffer filling level in the startup phase compared to client-based DASH heuristics. Controlling the bitrate switching using the defined threshold, the results in Fig. 11d show that GreedyMSMC algorithm is effective in reducing both switching frequency and magnitude compared to BBA. However, the switching of our algorithm is higher than RBA which however it gets benefit of higher average bitrate and less buffer delay compared to RBA.

The results of comparison in term of resource utilization deviation among the base stations show the RMSD values of 0.2353, 0.2213 and 0.0485 for respectively BBA, RBA and GreedyMSMC which again verifies the effectiveness of clients to server mapping strategy. Finally, the fairness index values of 0.9791, 0.9748 and 0.9877 are obtained when using BBA, RBA and GreedyMSMC algorithm, respectively, confirming the superiority of the proposed algorithm in term of fairness among the clients as well.

10 Conclusion

This article presents a multi-access edge computing assisted DASH system which facilitates the access of large scale of mobile clients to the set of replicated video contents over multiple edge servers. Our objective was to quantify the advantages of using the network-assisted adaptation strategy compared to the purely client-based DASH heuristics. Toward this objective, we designed an optimized solution for network-assisted adaptation which includes a clients to server mapping strategy and considers the joint weighted maximization of QoE and fairness. Due to the NP-hardness of the problem formulation, we then crafted a self-tuning greedy algorithm with low complexity which utilizes the MEC facilities.

Simulation results reveal that the proposed network-assisted scheduling algorithm outperforms the purely client-based DASH heuristics and one existing SAND-DASH solution in some metrics. Particularly, the noticeable improvements using the network-assisted strategy are obtained in situations when the achievable throughput is moderately high or the link quality of the mobile clients does not differ from each other substantially.

The proposed system assumes that the video chunks are replicated on all of the edge servers. However, these servers have a limited storage capacity and, therefore, some video chunks may need to be requested from a further away server, which may increase inter-ISP network traffic and increase the operational costs of the ISP. In our future work, we plan to investigate jointly maximizing QoE and minimizing the amount of inbound traffic using an optimized edge caching and bitrate adaptation solution for mobile video streaming.

Acknowledgments

This work has been financially supported by the Academy of Finland (grant numbers 278207 and 297892), Tekes - the Finnish Funding Agency for Innovation, and the Nokia Center for Advanced Research.

References

[1] S. Petrangeli, J. Famaey, M. Claeys, S. Latre, and F. D. Turk, "QoE driven rate adaptation heuristic for fair adaptive video streaming," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 12, no. 2, pp. 1–15, Oct. 2015.

[2] D. Bethanabhotla, G. Caire, and M. J. Neely, "Adoptive video streaming for wireless networks with multiple users and helpers," *IEEE Trans. Commun.*, vol. 63, no. 1, pp. 268–285, Jan. 2015.

[3] N. Bouten, S. Latre, and J. Famaey, W. V. Leekwijck, and F. D. Turk, "In-network quality optimization for adaptive video streaming services," *IEEE Trans. Multimedia*, vol. 16, no. 8, pp. 2281–2293, Dec. 2014.

[4] 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.

[5] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hossfeld, and P. Tran-Gia, "A survey on quality of experience of HTTP adaptive streaming," *IEEE Commun. Survey Tuts.*, vol. 17, no. 1, pp. 469–492, Jan.–Mar. 2015.

[6] T. Hossfeld, S. Egger, R. Schatz, M. Fiedler, K. Masuch, and C. Lorentzen, "Initial delays versus interruptions: Between the devil and deep blue," in *Proc. IEEE Int. Workshop Quality Multimedia Experience*, Aug. 2012, pp. 1–6.

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

[8] T. Mangla, N. Theera-Ampornpunt , M. Ammar, E. Zegura, and S. Bagchi, "Video through a crystal ball: Effect of bandwidth prediction quality on adaptive streaming in mobile environments," in *Proc. 8th ACM Int. Workshop Mobile Video*, May. 2016, pp. 1–6.

[9] G. Dimopoulos, I. Leontiadis, P. Barlet-Ros, and K. Papagiannaki, "Measuring video QoE from encrypted traffic," in *Proc. ACM Internet Meas. Conf.*, Nov. 2016, pp. 513–526.

[10] H. Riiser, T. Endestad, P. Vigmostad, C. Griwodz, and P. Halvorsen, "Video streaming using a location-based bandwidth-lookup service for bitrate planning," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 8, no. 3, pp. 1–19, Jul. 2012.
[11] Z. Li, S. Zhao, D. Medhi, and I. Bouazizi, "Wireless video traffic bottleneck coordination with a DASH SAND framework," in *Proc. IEEE Visual Commun. Image Process.*, Nov. 2016, pp. 1–4.
[12] Y. Sun, X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli, "CS2P: Improving video bitrate selection and adaptation with data-driven throughput prediction," in *Proc. ACM Conf. SIGCOMM*, Aug. 2016, pp. 272–285.
[13] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal adaptation for online videos," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.
[14] G. Cofano, L. D. Cicco, T. Zinner, A. Nguyen-Ngoc, P. Tran-Gia, and S. Mascolo, "Design and experimental evaluation of network-assisted strategies for HTTP adaptive streaming," in *Proc. 7th ACM Int. Conf. Multimedia Syst.*, May. 2016, pp. 1–12.
[15] A. Bentaleb, A. C. Begen, and R. Zimmermann, "SDNDASH : Improving QoE of HTTP adaptive streaming using software defined networking," in *Proc. ACM Conf. Multimedia*, Oct. 2016, pp. 1296–1305.
[16] J. Yao, S. Kanhere, I. Hossain, and M. Hassan, "Empirical evaluation of HTTP adaptive streaming under vehicular mobility," in *Proc. Int. Conf. Fed. Inf. Process.*, 2011, pp. 92–105.
[17] E. Thomas, M. O. V. Deventer, T. Stockhammer, A. C. Begen, M.-L. Champel, and O. Oyman, "Applications and deployments of server and network assisted DASH (SAND)," in *Proc. Int. Broadcast. Conv. Conf.*, 2016, pp. 1–8.
[18] R. Margolies, A. Sridharan, V. Aggarwal, R. Jana, N. K. Shankaranarayanan, V. A. Vaishampayan, and G. Zussman, "Exploiting mobility in proportional fair cellular scheduling : Measurements and algorithms," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 355–367, Feb. 2016.
[19] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency and stability in HTTP-based adaptive video streaming with FESTIVE," in *Proc. 8th ACM Int. Conf. Emerging Netw. Experiments Technol.*, Dec. 2012, pp. 97–108.
[20] C. Wang, A. Rizk, and M. Zink, "SQUAD: A spectrum-based quality adaptation for dynamic adaptive streaming over HTTP," in *Proc. 7th ACM Int. Conf. Multimedia Syst.*, May. 2016, pp. 1–12.
[21] J. Xie, R. Xie, T. Huang, J. Liu, and Y. Liu, "Energy-efficient cache resource allocation and QoE optimization for HHTP adaptive bit rate streaming over cellular networks," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–6.
[22] M. Zhao, X. Gong, J. Liang, W. Wang, X. Que, and S. Cheng, "Scheduling and resource allocation for wireless dynamic adaptive streaming of scalable video over HTTP," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2014, pp. 1681–1686.
[23] Y. Ran, Y. Shi, E. Yang, S. Chen, and J. Yang, "Dynamic resource allocation for video transcoding with QoS guaranteeing in cloud-based DASH system," in *Proc. IEEE Global Commun. Workshop*, Dec. 2014, pp. 144–149.
[24] S. Colonnese, F. Cuomo, T. Melodia, and I. Rubin, "A cross layer bandwidth allocation schema for HTTP-based video streaming in LTE cellular networks," *IEEE Commun. Lett.*, vol. 21, no. 2, pp. 386–389, Feb. 2017.
[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] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5G networks: New paradigms, challenges and scenarios," *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 54–61, Apr. 2017.
[27] S. V. Tran and A. M. Eltawil, "Optimized scheduling algorithm for LTE downlink system," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2012, pp. 1462–1466.
[28] V. Cardellini, M. Collajani, and P. S. Yu, "Dynamic load balancing on web server systems," *IEEE Internet Comput.*, vol. 3, no. 3, pp. 28–39, May/Jun. 1999.
[29] A. E. Essaili, Z. Wang, and E. Steinbach, "QoE-based cross-layer optimization for uplink video transmission," *ACM Trans. Multimedia Comput. Commun.*, vol. 12, no. 1, pp. 1–22, Aug. 2015.
[30] A. Mehrabi, M. Siekkinen, and A. Ylä-Jääski, "Joint optimization of QoE and fairness through network assisted adaptive mobile video streaming," in *Proc. 13th IEEE Conf. Wireless Mobile Comput. Netw. Commun.*, Oct. 2017, pp. 1–8.
[31] H. Ahlehagh and S. Dey, "Video-aware scheduling and caching in the radio access network," *IEEE/ACM Trans. Netw.*, vol. 22, no. 5, pp. 1444–1462, Oct. 2014.
[32] W. Zhang, Y. Weng, Z. Chen, and A. Khisti, "QoE-driven cache management for HTTP adaptive bitrate streaming over wireless networks," *IEEE Trans. Multimedia*, vol. 15, no. 6, pp. 1431–1445, Oct. 2013.
[33] S. Sesia, I. Toufik, and M. Baker, *LTE—The UMTS Long Term Evolution: From Theory to Practice*. New York, NY, USA: Wiley, 2009.
[34] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA, USA: Freeman, 1979.
[35] Sandvine: Global Internet Phenomena Report Q2, 2012. [Online]. Available: http://tinyurl.com/nyqyarq
[36] Sandvine: Global Internet Phenomena Report H2, 2013. [Online]. Available: http://tinyurl.com/nt5k5qw
[37] (2018). [Online]. Available: https://en.wikipedia.org/wiki/Fog_computing
[38] (2015). [Online]. Available: http://www.simulte.com
[39] (2018). [Online]. Available: http://www.gurobi.com
[40] (2009). [Online]. Available: http://www.etsi.org/deliver/etsi_tr/136900_136999/136942/08.02.00_60/tr_136942v080200p.pdf
[41] ISO/IEC 23009. Dynamic adaptive streaming over HTTP (DASH). (2014). [Online]. Available: https://www.iso.org/standard/65274.html
[42] ISO/IEC 23009–5. Dynamic adaptive streaming over HTTP (DASH) Part 5: Server and network assisted DASH (SAND). (2017). [Online]. Available: https://www.iso.org/standard/69079.html

Abbas Mehrabi received the BSc degree in computer engineering from the Shahid Bahonar University of Kerman, Iran, in 2008, the MSc degree in computer engineering from Azad University, South Tehran, in 2010, and the PhD degree from the School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology, Gwangju, South Korea, in 2017. He is currently a postdoctoral researcher with the Department of Computer Science, Aalto University, Espoo, Finland. His main research interests include quality of experience optimization and resource allocation for multimedia services in mobile edge computing environments, energy efficient mobile computing, and scheduling/planning problems in smart grids. He is a member of the IEEE.

Matti Siekkinen received the MSc degree in computer science from the Helsinki University of Technology, in 2003 and the PhD degree from the EURECOM / University of Nice Sophia-Antipolis, in 2006. He is currently a postdoctoral researcher with Aalto University. His research on multimedia systems combines techniques from multimedia signal processing, mobile networking, cloud computing, system analysis, machine learning, and HCI. He is a member of the IEEE.

Antti Ylä-Jääski received the PhD degree from ETH Zurich, in 1993. He has worked with Nokia between 1994-2009 in several research and research management positions with a focus on future Internet, mobile networks, applications, services, and service architectures. He has been a tenured professor with Aalto University, Department of Computer Science since 2004. His current research interests include mobile cloud computing, mobile multimedia systems, pervasive computing and communications, indoor positioning and navigation, energy efficient communications and computing, and Internet of Things. He is a member of the IEEE.