Mehrabidavoodabadi, Abbas; Siekkinen, Matti; Gazi Karam Illahi, Gazi; Ylä-Jääski, Antti

D2D-Enabled Collaborative Edge Caching and Processing with Adaptive Mobile Video Streaming

Published in:
IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM) 2019

DOI:
10.1109/WoWMoM.2019.8792981

Published: 12/08/2019

Document Version
Peer reviewed version

Please cite the original version:
Mehrabidavoodabadi, A., Siekkinen, M., Gazi Karam Illahi, G., & Ylä-Jääski, A. (2019). D2D-Enabled Collaborative Edge Caching and Processing with Adaptive Mobile Video Streaming. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM) 2019* (pp. 1-10). IEEE. https://doi.org/10.1109/WoWMoM.2019.8792981

D2D-enabled Collaborative Edge Caching and Processing with Adaptive Mobile Video Streaming

Abbas Mehrabi, Matti Siekkinen, Gazi Illahi, and Antti Ylä-Jääski
Department of Computer Science, Aalto University, Espoo, Finland
Emails: {abbas.mehrabidavoodabadi,matti.siekkinen,gazi.illahi,antti.yla-jaaski}@aalto.fi

***Abstract*—Multi-access edge computing (MEC) enables placing video content at the edge of a mobile network with the aim of reducing data traffic in the backhaul network. Direct device-to-device (D2D) communication can further alleviate load from the backhaul network. Both MEC and D2D have already been examined by prior work, but their combination applied to adaptive video streaming have not yet been explored in detail. In this paper, we analyze how enabling D2D jointly with edge computing affects the quality of experience (QoE) of video streaming clients and contributes to reducing the backhaul traffic. To this end, we formulate the problem of jointly maximizing the QoE of the clients and minimizing the backhaul traffic and edge processing as an integer non-linear programming (INLP) optimization model and propose a low-complexity algorithm using self-parameterization technique to solve the problem. The main takeaway from simulation results is that enabling D2D with edge computing reduces the backhaul traffic by approximately 18% and edge processing by 30% on average while maintaining roughly the same average video bitrate per client compared to edge computing without D2D. Our results provide a guideline for system designers to judge the effectiveness of enabling D2D into MEC in the next generation of 5G mobile networks.**

***Index Terms*—Multi-access edge computing (MEC), Device-to-Device communication, DASH, Quality of experience (QoE), Fairness, Greedy-based algorithm, 5G networks.**

I. Introduction

According to the Cisco VNI, 82% of all IP traffic by 2021 would be video and mobile and wireless traffic would exceed 63% of the IP traffic [1]. Wireless traffic suffers from variable network conditions which affect QoE of video streaming. To ameliorate disruptions caused by variable network conditions, modern video streaming solutions follow the client-based dynamic adaptive video streaming (DASH) standard which is mainly based on video buffer occupancy level [2] or predicted network throughput [3]. Due to lack of coordination among the clients in purely client-based adaptation heuristics, network-assisted solutions [30] are emerging in addition to technologies such as multi-access edge computing (MEC), massive MIMO, and device to device communication (D2D) to support the massive video content delivery [6], [7]. MEC is considered to be one of the key capabilities brought by 5G networks [28].

Edge caching brings caches to the edge of the wireless network and has the potential of reducing video delivery latency and backhaul traffic, and improving energy and spectral efficiency [12]. Collaborative caching, wherein various caches in the network cooperate to deliver content, improves upon baseline edge caching [8],[10],[11],[15]. Beyond collaborative edge caching, synergistic use of MEC and edge caching further improves video delivery performance by employing video transcoding at the edge to serve lower bitrate versions of a cached video on demand [10], [13]. One of the enabling technologies in 5G is envisioned to be D2D communications [7]. Collaborative edge caching with D2D communication is also predicted to improve video delivery [11][15]. In all of the above caching schemes, there is a trade-off between improvement in content delivery and resources like cache size and processing capacity at the edge. Joint optimization solutions have been proposed to handle these trade-offs [10],[15].

Despite several studies on the advantages of both MEC and D2D, the combination of video edge caching, processing with D2D compensates some performance degradation which arise when these solutions are used independently. To the best of our knowledge, the combined behavior of D2D functionality and network-assisted rate adaptation in MEC environments have not yet been addressed in detail. Specifically, the impact of D2D on joint QoE-traffic optimization for DASH using edge computing (edge-assisted DASH) is still underexplored. Motivated by these facts, we aim to quantify how enabling D2D functionality in edge-assisted adaptive mobile video streaming helps to reduce the traffic in the backhaul of mobile network while achieving maximum QoE for the clients. To this end, we propose joint optimization of QoE, backhaul data traffic, and edge processing for MEC environments integrated with D2D functionality. Our simulation-based study suggests that enabling D2D can reduce the backhaul traffic by 18% and edge processing by 30% on average and that the cache replacement algorithm used plays an important role in achieving these benefits. We summarize our contributions as follows:

- We present a system for joint optimization of QoE, backhaul traffic and edge processing that combines direct D2D communication with joint edge caching and processing in adaptive mobile video streaming.
- To tackle the intractability of the problem, we design a low complexity greedy-based algorithm which requires the minimum parameter tuning hence making it attractive for easy deployment by mobile network operator (MNO).
- We run simulations with radio access link level traces obtained from a full-fledged LTE simulator and evaluate the performance of the D2D and MEC combination in terms of backhaul data traffic and QoE.

The remainder of the paper is organized as follows: Related work is discussed in Section II. The proposed system

architecture and its components are detailed in Section III and the optimization problem is formulated in Section IV. The proposed algorithm is presented in Section V and the simulation results are discussed in Section VI. Finally, Section VII concludes the paper.

II. Related Work

Research on DASH-based video streaming solutions have focused on the client side rate adaptation logic as the DASH standard leaves rate adaptation logic up to the client. Client side rate adaptation methods, based on the buffer occupancy level of the streaming client [2], predicted network throughput [3] or a combination of both [18], suffer from a limited visibility of the network. They compete greedily with each other over bottleneck resources, resulting in deteriorated performance for all the clients and congestion in the network [19]. Detailed surveys on QoE and rate adaptation in DASH solutions are presented in [5] and [20], respectively.

To ameliorate this situation, server and network assisted bitrate adaptation has been proposed both in academia [21] and by industry forums [30]. Server and network assisted DASH (SAND-DASH) stipulates means for various elements of the video streaming pipeline, collectively called DASH-aware network elements (DANES), to collaborate with each other and improve QoE and fair use of network resources. Cofano *et al.* [21] evaluate two strategies of network assisted streaming: bandwidth reservation and bitrate guidance and consider optimization of fairness and video quality. Bentalib *et al.* [22] propose leveraging SDN to improve video QoE with fair resource allocation. Mehrabi *et. al* [9] propose edge computing assisted adaptive mobile video streaming and study the problem of client-to-edge server mapping and joint optimization of QoE and fairness.

Recent research on edge caching has presented promising results [14]. In contrast to distant content delivery networks (CDNs), video delivery at the edges of the network close to end users provides low latency with efficient utilization of radio access network (RAN) information [13], [16]. Unlike the CDNs, edge computing also facilitates the joint caching and processing by utilizing the transcoding capability of the edge servers [10], [11]. Wang *et al.* [13] propose caching in wireless networks in the core and the RAN. Liu *et al.* [12] consider various caching scenarios in wireless networks, showing significant improvements in spectral and energy efficiency [12]. In [15], a collaborative hierarchical caching is proposed for wireless networks, comprising of a central cache and multiple base station caches, collaborating to jointly optimize access delay. Advantages of cooperative caching over traditional schemes are illustrated in [8]. Tran *et al.* [10], [11] propose collaborative multi-bitrate video caching and processing in a MEC environment. Pedersen *et al.* [16] propose a RAN cache scheme augmented with limited transrating capacity and perform joint optimization of user QoE and network capacity.

Enabling D2D communications within a collaborative MEC caching environment is expected to improve aspects of content delivery within wireless networks [11][12][15]. Generic framework for edge caching with D2D communication, where the base station assists the D2D communication is presented in [24] and [23]. Wang *et al.* [29] consider cooperative cache placement in a D2D cache-enabled network and jointly optimize the cache resources at the base station and mobile users to satisfy QoS requirements. Ji *et al.* [17] discuss base station assisted cache-enabled D2D wireless networks and demonstrate improved system throughput. Using the techniques from the stochastic geometry, Malak *et. al* [25] design efficient caching strategies in the mobile devices in order to maximize the amount of data which is successfully exchanged among the devices under the noisy channel condition. However, they do not consider the integration of D2D into the edge computing and the minimization of the backhaul traffic was not considered in this work. Deng *et. al* [26] investigated the problem of minimizing the delay of receiving the data when two neighborhood devices contact each other through D2D communication. Similarly, this work does not take into account the edge functionality as well as the objective of the caching strategies should be rather mainly improving the cache hit rate. Authors in [27] considered the problem of jointly minimizing the delay and wireless channel allocation in content delivery between user devices through D2D and the cellular base stations. However, the joint optimization in this work does not take into account the QoE of the end users and also the processing capability of edge servers is not utilized.

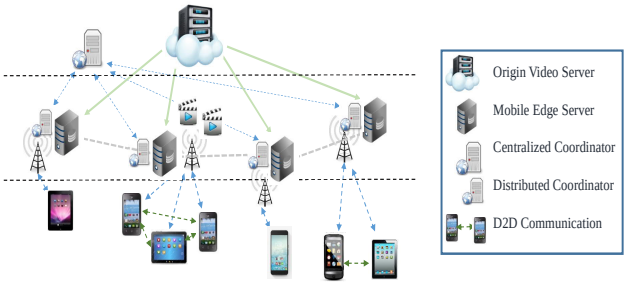


Fig. 1: Collaborative edge caching and processing with direct device-to-device communication.

While some of the previous works discussed above have considered optimization in collaborative edge caching in MEC environments, we feel that a broader view that comprises edge caches, edge transcoding servers, and D2D communication is still underexplored. For this reason, we propose in this paper a D2D-enabled edge-assisted collaborative caching and processing framework that jointly maximizes the QoE of the clients and minimizes the backhaul traffic and edge processing.

III. Collaborative Edge Caching and Processing with D2D

In this section, we first describe the architecture of the proposed system and then describe the mathematical notations.

A. System Overview

Fig. 1 illustrates the system design for collaborative edge caching and processing with direct device-to-device communication. Video segments with multiple qualities are initially

located at the origin server on the cloud and are cached (or processed) at the edge servers once they are requested by the mobile clients. In addition to the caching and processing at the edge servers, the clients can also cache the downloaded chunks/bitrates in their own devices and retrieve them either locally or from the cache of neighborhood clients through D2D communication. Associated with each edge server, the eNodeB (base station) allocates the downlink radio resource blocks (RBs) to the clients in a proportionally fair (PF) manner. Utilizing the infrastructures which support the scalability requirement of 5G mobile networks, our system also provides a network-assisted bitrate adaptation strategy in which the distributed coordinators nearby the edge servers collaborate with the mobile clients and the edge devices (servers, base stations) to solve a formulated optimization problem. Optimization results guide the clients toward a fair and optimal bitrate allocation. The combination of each edge server and associated base station works as an independent operating unit without the need for sharing the data with other edge servers. This facilitates the decentralized implementation of the system which is one of the main objectives of 5G networks.

In bottom-top hierarchical caching structure, the selected chunk/bitrate for a client is first looked up in the cache of its device and the local edge server. If the chunk is not locally available, it is then searched in the cache of neighborhood devices (which are assigned to the same server) through one-hop D2D communication. If miss again, the retrieval from the cache of neighborhood edge servers within a cluster is investigated. If the selected chunk does not exist in the caches, the potential of transrating from the chunks with higher bitrates available in either the local or the neighborhood edge servers is then investigated. The transrating task increases the volume of processed data at the edge. Finally, if the desired chunk does not exist in the caches and there is no way of transrating, the possibility of downloading it directly from the origin server with the increase in backhaul traffic is investigated.

B. System Notations

We consider the scheduling of total S mobile DASH clients during $|T|$ time slots with Δt as the duration of each time slot (*in seconds*). Multiple videos with different popularities are divided into several chunks with equal size of C seconds and each chunk is encoded into multiple discrete bitrates represented by set R initially stored on the origin video server. K edge servers are deployed in the system and with each server, eNodeB (base station) is associated such that $W_k^{(t)}$ available downlink resource blocks at base station k are allocated to its connected clients in time slot t in a PF manner according to their wireless link quality. The arrival (the time slot that client starts its streaming session) and the departure (the time slot that client either departs or abandons from its session) of client i are represented by A_i and D_i. The requested video by client i is also denoted by v_i. Media player of client i holds a video buffer with the capacity of B_i^{max} such that $B_i^{(t)}$ represents the buffer size at time slot t. The fixed size of the cache at every edge server and mobile device (client) are represented by respectively Q_M and Q_D and the set of cached chunks in edge server k and mobile device s at time slot t are denoted by respectively $M_k^{(t)}$ and $D_s^{(t)}$. Furthermore, due to the mobility of mobile devices, we denote by $N_s^{(t)}$ as the set of neighborhood clients (devices) of client s at time slot t i.e. the clients which are assigned to the same edge server as the client s is allocated in time slot t.

The binary variable $a_{ik}^{(t)}$ denotes the allocation of client i to edge server k at time slot t. In our system model, we assume that the server allocation is decided at the beginning of each new chunk and the allocated server remains unchanged if the client is downloading at the middle of the chunk. At the beginning of new chunk, the client is mapped to the server from where it receives the highest downlink signal-to-noise ratio (SNR) from its associated base station. Represented by $SNR_{ik}^{(t)}$ as the received downlink SNR of client i form base station k at time slot t, the corresponding theoretical throughput denoted by $Thr_{ik}^{(t)}$ is computed using the Shannon theorem [32]. Client downloads one second of video chunk at each time slot such that the integer variable $r_{ik}^{(t)}$ is defined to represent the allocated bitrate to client i assigned to edge server k at time slot t. Also, the integer variable $tr_{ij}^{(t)} > r_{ik}^{(t)}$ denotes the higher bitrate available at edge server j which is transreated to the requested bitrate of client i. The binary decision variable $x_{ikc}^{(t)}$ is also defined such that $x_{ikc}^{(t)} = 1$ states that client i allocated to server k at time slot t downloads its chunk/bitrate from the origin server, otherwise, it downloads from the cache of edge server or the mobile device if $x_{ikc}^{(t)} = 0$. The binary decision variable $y_{ij}^{(t)} = 1$ indicates that the allocated chunk/bitrate to client i at time slot t has been processed at edge server j. Since the video segments have the same DASH encoding format with variable bitrates, we use the simple linear transcoding model following the work in [10]. More precisely, the associated data for transrating from the higher bitrate $tr_{ij}^{(t)}$ to the desired bitrate of client i is equal to $(tr_{ij}^{(t)} - r_{ik}^{(t)}) \cdot \phi$ where ϕ represents the constant weighting of processing capacity at each edge server. Notations BT_i and PD_i are also defined which represent respectively the overall backhaul data traffic and the processed data at the edges caused by the video streaming of client i. The complete list of system notations have been summarized in Table. I.

C. Quality of Experience and Fairness

As pointed out in several research studies, playback stalling caused by buffer underrun is the most critical influencing factor of QoE in video streaming. Although avoiding stalling completely is difficult in practice, we design the bitrate selection in such a way to avoid the stalling whenever possible i.e., it is a constraint of the optimization problem. Other major factors in adaptive streaming are the perceived *average video bitrate* and the frequency and magnitude of *bitrate switching*. The initial buffering delay is also factor of QoE but, according to the study in [4], its impact is negligible in on-demand video streaming compared to the previously mentioned ones.

TABLE I: Description of system notations.

Notation	Description
K, S, R	Number of edge servers, number of DASH clients and the discrete set of available bitrates on each server
$\|T\|$, Δt, C	Total number of scheduling time slots, the duration of each slot and the constant size of each video chunk in seconds
$W_k^{(t)}$	Available resource blocks at base station k in time slot t
Q_D, Q_M	Cache size at respectively mobile device and edge server
$M_k^{(t)}$, $D_s^{(t)}$	Set of available chunks/bitrates in edge server k and mobile device s, respectively, at time slot t
P_j, ϕ	The processing capacity of edge server j and the constant weighting of edge processing capacity
A_i, D_i	Arrival and departure times of client i
v_i	Requested video by client i
B_i^{max}, $B_i^{(t)}$	Buffer capacity of client i and its buffer level at slot t
$SNR_{ik}^{(t)}$, $Thr_{ik}^{(t)}$	Achievable signal-to-noise ratio and the theoretical data throughput by client i from base station k at slot t
AQ_i, E_i, F_i	Average video bitrate, the accumulated bitrate switching and the associated fairness value for client i
BT_i	Overall backhaul traffic from video streaming of client i
PD_i	The overall processed data from video streaming of client i
$N_s^{(t)}$	Set of neighborhood devices of client s at time slot t
ρ, ω, γ	Adjustable weighting parameters for average bitrate, bitrate switching and fairness, respectively
α, β, θ	Adjustable weighting parameters for QoE, data traffic and the processed data, respectively
δ_S, δ_F	Switching and fairness thresholds
$a_{ik}^{(t)}$, $y_{ij}^{(t)}$	Binary indicator of allocating client i to server k at time slot t and, the binary decision variable indicating the processing of bitrate client i at edge sever j in time slot t
$x_{ikc}^{(t)}$	Binary decision variable indicating the chunk download from origin server by client i at server k in time slot t
$r_{ik}^{(t)}, tr_{ij}^{(t)} \in R$	Discrete video bitrate (integer decision variable) of client i at time slot t and the transrated bitrate for client i at edge server j in time slot t

We should note that the bitrate adaptation solution based on QoE objective model which combines the video quality metrics with fairness index has been verified to outperform the state-of-the-art client-based adaptation heuristics [9].

1) Average Bitrate: represents the quality at which the client watches the video. Although the relation between video bitrate and the perceived quality by the client may not be linear in practice, we consider the linear relation for the sake of simplicity while our model is easily adaptable to no-linear quality models. The average video bitrate perceived by client i during its streaming session denoted by AQ_i is given by:

$$AQ_i = \frac{1}{|D_i - A_i|} \sum_{t=A_i}^{D_i} \sum_{k=1}^{K} a_{ik}^{(t)} \cdot r_{ik}^{(t)} \tag{1}$$

2) Bitrate Switching: affects the QoE negatively [5]. The magnitude of switching refers to amount of change in the bitrate of consecutive chunks. Since client i streams the total number of $\lceil (D_i - A_i)/C \rceil$ chunks, the accumulated switching magnitude during its video streaming session is given by:

$$E_i = \sum_{p=1}^{\lceil (D_i - A_i)/C \rceil} \sum_{k=1}^{K} (a_{ik}^{((p-1)C+1)} \cdot r_{ik}^{(p-1)C+1} - a_{ik}^{(p-2)C+1} \cdot r_{ik}^{(p-2)C+1}) \tag{2}$$

3) Fairness: Allocating the bitrates merely based on the client-based heuristics may not be fair due to the lack of coordinations among the competing clients. Our system design takes explicit measures to ensure fairness when allocating the bitrates to the set of competing clients at each time slot. More precisely, the bitrates are allocated so that for each active client at each time slot, the difference between its allocated bitrate with the average bitrate of other simultaneous clients is minimized. We define F_i as the fairness value associated with the whole video streaming session of client i.

$$F_i = \sum_{t=A_i}^{D_i} \sum_{k=1}^{K} a_{ik}^{(t)} \cdot |r_{ik}^{(t)} - \bar{r}^{(t)}| \tag{3}$$

where $\bar{r}^{(t)}$ is the average bitrate of other simultaneous clients at time slot t.

D. Backhaul Traffic and Data Processing

With the decision variables defined in the system model, the overall backhaul data traffic during the whole video streaming duration of client i is given by the following relation:

$$BT_i = \sum_{t=A_i}^{D_i} \sum_{k=1}^{K} a_{ik}^{(t)} \cdot x_{ikc}^{(t)} \cdot \Delta t \cdot r_{ik}^{(t)} \tag{4}$$

The overall volume of processed data at local or neighborhood edges due to video streaming of client i is given by:

$$PD_i = \sum_{t=A_i, 1 \le j \le K}^{D_i} y_{ij}^{(t)} \cdot \mathbb{I}((v_i, \lceil (t - A_i)/C, tr_{ij}^{(t)}) \in M_j^{(t)}) \cdot \Delta t \cdot (tr_{ij}^{(t)} - r_{ik}^{(t)})\phi \tag{5}$$

In equation (5), the inner identity function $\mathbb{I}(.)$ indicates the availability of video chunk of client i with higher quality $tr_{ij}^{(t)}$ at the local (the server to which client is assigned) or neighborhood edge server j. In the case of availability, the volume of processed data is equal to the amount of transrated data from higher quality $tr_{ij}^{(t)}$ to the allocated video quality to client i (by coordinator) at server k in time slot t i.e., $r_{ik}^{(t)}$.

IV. Joint Optimization Problem

The problem of jointly maximizing the QoE of individual client i and minimizing the backhaul data traffic and edge data processing is formulated as the following integer non-linear programming (INLP) optimization model:

$$\underset{x,y,r,tr}{Maximize}\ \alpha(\rho AQ_i - \omega E_i - \gamma F_i)\Delta t - \beta BT_i - \theta PD_i \tag{6}$$

Subject to:

$$\sum_{j \in S} a_{jk}^{(t)} \cdot \lceil \frac{r_{jk}^{(t)}}{Thr_{jk}^{(t)}} \rceil \le W_k^{(t)},\ \forall 1 \le k \le K,\ \ 1 \le t \le |T| \tag{7}$$

$$0 < B_i^{(t)} \le B_i^{max}, \qquad \forall A_i \le t \le D_i \tag{8}$$

$$a_{ik}^{(t)} = \begin{cases} a_{ik}^{(t-1)}, & t \ mod \ C \neq 1 \\ 1, & t \ mod \ C = 1 \wedge k = arg \ \mathbf{max}\{SNR_{ik}^{(t)}\} \\ 0, & \text{Otherwise} \end{cases} \tag{9}$$

$$\sum_{s \in S} \sum_{t=A_s}^{D_s} \sum_{k=1}^{K} a_{sk}^{(t)} \cdot y_{sj}^{(t)} \cdot \Delta t \cdot (tr_{sj}^{(t)} - r_{sk}^{(t)})\phi \leq P_j, \quad \forall 1 \leq j \leq k \tag{10}$$

$$x_{ikc}^{(t)} \leq 1, \quad \forall 1 \leq k \leq K, A_i \leq t \leq D_i \tag{11}$$

$$\sum_{j=1}^{K} y_{ij}^{(t)} \leq 1, \quad \forall A_i \leq t \leq D_i \tag{12}$$

$$x_{ikc}^{(t)}, y_{ij}^{(t)} \in \{0,1\}, \quad \forall 1 \leq j,k \leq K, A_i \leq t \leq D_i \tag{13}$$

$$r_{ik}^{(t)}, tr_{ij}^{(t)} \in R, \quad \forall 1 \leq j,k \leq K, A_i \leq t \leq D_i \tag{14}$$

In INLP problem (6)-(14), the only decision variables are binary variables $x_{ikc}^{(t)}$, $y_{ij}^{(t)}$ and integer variables $r_{ik}^{(t)}$, $tr_{ij}^{(t)}$ while, the variables $a_{ik}^{(t)}$ are not counted as decision variables since clients to server mappings are determined beforehand.

The objective function (6) aims to jointly maximize the QoE of the client and minimize the overall data traffic and data processing caused by the client. Constraint (7) ensures that at each base station, the overall resource blocks allocated to the associated clients at each time slot does not exceed the total available resource blocks at that time slot. Constraint (8) ensures that no stalling happens in the client's buffer during its whole video streaming session. Constraint (9) determines the clients to server mapping at each time slot and (10) ensures that the overall volume of processed data at each edge server does not exceed the processing capacity of the server. Constraint (11) guarantees that at each time slot, the client retrieves its chunk/bitrate from only one location and (12) states that the transrating operation can be performed at only one edge server. Finally, constraints (13) and (14) determine the range of decision variables.

V. Online Algorithm

The optimization problem (6)-(14) belongs to the class of NP-hard problems due to integer decision variables in the model. To cope with the intractability, we design and implement a suboptimal greedy bitrate adaptation solution for the problem. Using greedy algorithm leads to high quality solutions with low computational complexity. Our solution involves the in-network collaboration among the edge servers, coordinator and the mobile clients, and also accommodates a self-tuning mechanism for adjusting the weight parameters of QoE term in (6). The minimum need for parameter tuning makes the algorithm easy for practical deployment. Pseudo-code of the proposed algorithm named Self-tuned Greedy Bitrate Allocation (SGBA) is illustrated in Algorithm 1.

At each time slot and for every active client in a greedy manner, the algorithm first maps the client to the most suitable edge server according to relation (9) and initializes the buffer status, backhaul traffic and the processed data at the edges if the client has just started its video streaming at the current time slot (lines 6-8). The algorithm then allocates the same video bitrate as the previous time slot and also updates the buffer status, backhaul traffic and edge processed data if the client is downloading at the middle of its chunk (lines 9-13). Otherwise, if the client is about to download the new chunk, the subroutine Self-tuned Bitrate Selection is called to decide on the optimal and fair bitrate to the new chunk (lines 14,15).

As illustrated in Subroutine 1, the algorithm allocates the highest available bitrate if the client is downloading the first chunk of video (lines 1-3). For the subsequent chunks in either the startup or steady phase, the algorithm allocates the maximum sustainable bitrate (available in set R in decreasing order) to the client according to its link quality (obtained from the edge server) which results in less amount of switching with the bitrate of the previous chunk and also yields a high degree of fairness with respect to other simultaneous clients. More precisely, the allocated bitrate should result in a switching which is less than the threshold δ_S and a fairness value greater than the threshold δ_F (lines 6-7). Among those set of bitrates which satisfy both thresholds, the one which maximizes the objective value (6) is chosen as the allocated bitrate to the current chunk of the client (lines 8-14). Note that the evaluation of objective value (6) for each candidate bitrate is based on the availability (or processing) of the bitrate in bottom-top hierarchical caches as well as the weighting values in the objective function (indicated in conditions C1-C2).

Algorithm 1 Self-tuned Greedy Bitrate Allocation (SGBA) (Run by the coordinators)

1: **Input:** $|T|, K, R$: Number of time slots, number of edge servers, set of available discrete bitrates at origin server.
2: **Output:** Binary allocation $x_{ikc}^{(t)}$, $y_{ij}^{(t)}$ and integer bitrate allocation $r_{ik}^{(t)}$, $tr_{ij}^{(t)}$ for each client i, edge server $1 \leq j,k \leq K$ and time slot $1 \leq t \leq |T|$, $Utility$, $BackhaulTraffic$, $ProcessedData$
$Utility = BackhaulTraffic = ProcessedData = 0$;
3: **for** each time slot $1 \leq t \leq |T|$ **do**
4: **for** each client i such that $A_i \leq t \leq D_i$ **do**
5: $maxUtility = -\infty$;
6: **Allocate** client i to server $1 \leq k \leq K$ according to (9)
7: **if** $t = A_i$ **then**
8: **Initialize** $BufferStatus$, BT_i, PD_i and L_i;
9: **if** $(t - A_i)$ **mod** $C \neq 1$ **then**
10: **Allocate** client i to the same server and with the same bitrate as with time slot $t-1$;
11: **Update** $B_i^{(t)}$, BT_i, PD_i;
12: **if** $BufferStatus = False$ **And** $B_i^{(t)} = B_i^{max}$ **then**
13: $BuffetrStatus = True; L_i = t - A_i$;
14: **if** $(t - A_i)$ mod $C = 1$ **then**
15: **Call** Subroutine Self-tuned Bitrate Selection;
16: **if** $t = D_i$ **then**
17: $Utility = Utility + maxUtility$;
18: $BackhaulTraffic = BackhaulTraffic + BT_i$;
19: $ProcessedData = ProcessedData + PD_i$;
20: **Return** $Utility$, $BackhaulTraffic$, $ProcessedData$;

If there is no available bitrate satisfying both thresholds, the algorithm then checks the availability of bitrates that satisfy the switching threshold δ_S compromising the fairness (line 15-19). If there is no available bitrate in set R satisfying the switching threshold, the largest sustainable bitrate that maximizes the objective value is finally chosen (lines 20-23). In other words, the algorithm tries to allocate the bitrates in a way that best satisfies the thresholds while maximizing its objective value. The results in [9] show that using buffer-based adaptation heuristic for bitrate selection results in highest amount of switching. Motivated by this, our algorithm sets the value of switching threshold δ_S equal to the switching magnitude which happens when the bitrates of the chunks are allocated merely based on the buffer occupancy level. In contrast, the fairness threshold δ_F is given as input to the algorithm at the deployment phase.

$Data = 0$;
C1: if $(v_i, \lceil \frac{t-A_i}{C} \rceil, r) \notin D_s^{(t)}, \ \forall s \in N_i^{(t)} \cup \{i\}$
AND $(v_i, \lceil \frac{t-A_i}{C} \rceil, r) \notin M_{k'}^{(t)}, \ \forall 1 \leq k' \leq K$
then $Data = r\Delta t$

C2: if $\exists\, (j \wedge tr = min\{r' \in R | r' > r\})$
$\ni (v_i, \lceil \frac{t-A_i}{C} \rceil, tr) \in M_j^{(t)}$ **AND**
$\theta(tr - r)\phi\Delta t < \beta \cdot Data$ **then**
$Processing = (tr - r)\phi\Delta t$; $Data = 0$;

After the allocation of the bitrate to the current chunk, the weighting parameters ρ, ω and γ in the QoE term are dynamically updated at the current time slot (line 24) according to the following adjustments:

$$\rho' = 1 - (r_{max} - r_{ik}^{(t)})/r_{max} \tag{15}$$

$$\omega' = 1 - |r_{ik}^{(t)} - r_{ik}^{((\lceil t/C \rceil - 1)\cdot C)}| / max(r_{ik}^{(t)}, r_{ik}^{((\lceil t/C \rceil - 1)\cdot C)}) \tag{16}$$

$$\gamma' = 1 - |r_{ik}^{(t)} - \bar{r}^{(t)}| / max(r_{ik}^{(t)}, \bar{r}^{(t)}) \tag{17}$$

$$\rho = \rho'/(\rho' + \omega' + \gamma') \tag{18}$$

$$\omega = \omega'/(\rho' + \omega' + \gamma') \tag{19}$$

$$\gamma = \gamma'/(\rho' + \omega' + \gamma') \tag{20}$$

As stated by equations (15) and (18), the weighting parameter ρ is adjusted based on how far is the selected bitrate of the current chunk from the maximum available bitrate. According to (16) and (19), the weighting parameter ω is adjusted based on how far is the selected bitrate from the one which yields no switching. Obviously, the bitrate which yields no switching is the one which was allocated to the previous chunk. Similarly, the equations (17) and (20) state that the fairness weighting γ is computed based on the distance between the allocated bitrate and the average bitrate of other simultaneous clients.

The algorithm then computes each of the optimization terms (average quality, switching, fairness and the objective value in line 25) and subsequently updates the binary decision variables, the amount of processed data and the incurred backhaul traffic at the current time slot (line 28).

A. Retention-based Collaborative Caching (RBCC) Heuristic

After the bitrate allocation at each time slot, the cache replacement heuristics are called to update the contents of the cache at the edge servers and mobile devices if some chunks are downloaded from the neighborhood servers or devices.

Subroutine 1: Self-tuned Bitrate Selection

1: **if** $t - A_i \leq C$ **then**
2: **Allocate** the highest available bitrate;
3: **Update** *BufferStatus*, $B_i^{(t)}$, BT_i, PD_i
4: **Compute** $estThr$ and threshold δ_S;
5: **for** each bitrate $r \in R$ in decreasing order **do**
6: **if** allocation of r satisfy (7) **AND** $r \leq$ **max**$(estThr, \hat{Thr}_{ik}^{(t)}, B_i^{(t)})$ **then**
7: **if** $|r - r_{ik}^{(t-1)}| \leq \delta_S$ **AND** $1 - |r - \bar{r}|/(R_{max} - R_{min}) >= \delta_F$ **then**
8: $Data = 0$; $Processing = 0$;
9: **Compute** weightings ρ, ω and γ;
10: $QE = (\rho r - \omega|r - r_{ik}^{(t-1)}| - \gamma|r - \bar{r}|) \cdot \Delta t$;
11: **Determine** $Data$ and $Processing$ according to conditions C1 and C2;
12: **if** $\alpha QE - \beta Data - \theta Processing > maxUtility$ **then**
13: $maxUtility = \alpha QE - \beta Data - \theta Processing$; $r_{ik}^{(t)} = r$;
14: **if** $Processing \neq 0$ **then** $tr_{ij}^{(t)} = tr$;
15: **if** $r_{ik}^{(t)} = 0$ **then**
16: **for** each bitrate $r \in R$ in decreasing order **do**
17: **if** allocation of r satisfy (7) **AND** $r \leq$ **max**$(estThr, \hat{Thr}_{ik}^{(t)}, B_i^{(t)})$ **then**
18: **if** $|r - r_{ik}^{(t-1)}| \leq \delta_S$ **then**
19: **Perform** the same operations as in the lines 8-14;
20: **if** $r_{ik}^{(t)} = 0$ **then**
21: **for** each bitrate $r \in R$ in decreasing order **do**
22: **if** allocation of r satisfy (7) **AND** $r \leq$ **max**$(estThr, \hat{Thr}_{ik}^{(t)}, B_i^{(t)})$
23: **Perform** the same operations as in the lines 8-14;
24: **Update** weighting parameters ρ, ω γ at time slot t;
25: **Compute** AQ_i, E_i, F_i and U_i up to time slot t according to respectively (1), (2), (3) and (6); **Update** $B_i^{(t)}$
26: **if** $B_i^{(t)} = B_i^{max}$ **AND** $BufferStatus = False$ **then**
27: $BufferStatus = True$; $L_i = t - A_i$;
28: Update the volume of processed data, PD_i, the binary decision variables $x_{ikc}^{(t)}$, $y_{ij}^{(t)}$ and the backhaul data traffic BT_i according to the conditions C1 and C2;
29: **Return** $maxUtility$, PD_i, BT_i, $x_{ikc}^{(t)}$, $y_{ij}^{(t)}$;

Intuitively, prior statistical knowledge about the video streaming behaviors of the mobile clients can help improve cache replacement heuristics. We use a proactive cache replacement heuristic called retention-based collaborative caching (RBCC) which relies on two sources of statistical information when prioritizing among multiple set of chunks/bitrates for caching: First, how likely will each combination of chunk/bitrate be requested by the clients in the future time slots. Second, how frequently has a specific bitrate been requested by the client in the previous time slots. The first set of information can be approximated using the knowledge about the retention of the clients toward different

video contents which are mostly available by the origin video server in advance. The second set of information that aims to estimate the requested bitrate by the client based on its network condition can be obtained from the video streaming history. In addition, the heuristic also takes into account the impact of collaboration among the edge servers.

Consider updating the cache contents at edge server $1 \leq k \leq K$ in time slot t. The RBCC heuristic first computes the caching values for every local client i (the client which is allocated to server k at time slot t) as follows:

$$P_{Lcache}(i,k,t) = \cup P_L(j,k,t) \cup P_L(new,k,t), \quad \forall j \in S \ni a_{jk}^{(t)} = 1 \qquad (21)$$

where $P_L(j,k,t)$ is the probability that the local client j (allocated to the same server as client i at time slot t) will request in the future time slots the same chunk of video which client i has requested at the current time slot. This probability is derived as follows:

$$P_L(j,k,t) = P_{Lr}(j,k,t) \times P_{Lacc}(j,k,t) \qquad (22)$$

where the first term in the right hand side of (22) is the probability that client j will be still active in its streaming session when it reaches the current chunk of client i and the second term is the probability that the client has requested the same bitrate as the current bitrate of client i in the previous time slots. P_{Lr} is approximated knowing P_{ac}, the function which describes the retention pattern of clients toward different video requests and is normally available at the origin server while P_{Lacc} is computed by looking at the client streaming history. In order to consider the impact of a new arrival on the caching decisions, our heuristic also takes into account the computation of $P_L(new,k,t)$ in which the probability $P_{Lr}(new,k,t)$ is computed in the similar way as explained for relation (22) and $P_{Lacc}(new,k,t) \approx 1/|R|$ since the new arrival has not yet streamed the video and its history is empty.

In addition to the local clients, the caching values $P_{Ncache}(i',k',t)$ are also computed for every client i' allocated to the neighborhood edge server k' in the similar as for the local clients (equations (21) and (22)). After computing the caching values for all the chunks at the current time slot (the existing chunks in the cache and the requested ones from the local/neighborhood clients), the heuristic then sorts them in decreasing order of caching values and inserts them into the cache until there is available space.

In contrast to the set of clients assigned to an edge server, the set of neighborhood devices of a given client change more frequently. Due to this fact, our system applies the commonly adopted least recently used (LRU) heuristic for local cache updating at the mobile devices.

B. Computational Complexity

It is seen from Algorithm 1 that the overall computations happen during $\frac{|T|}{C}$ chunks with total number of $|T|$ time slots. For every active client at the beginning of each chunk, the server mapping (relation (9)) is performed with the worst case complexity of $O(K)$. The main computing task is then executing the self-tuned bitrate selection procedure which our analysis shows the worst case complexity of $O(S \cdot K \cdot (|R|^2 + (|R| + C) \cdot S + |R|)$ for this procedure. With S clients and K edge servers, the following worst case time complexity is derived for the algorithm SGBA:

$$T_{SGBA} \in O(\frac{|T|}{C} \cdot S \cdot K \cdot T_{Subroutine1}) = O(\frac{|T|}{C} \cdot S \cdot K \cdot (|R|^2 + (|R| + C) \cdot S + |R|)) \qquad (23)$$

Our analytical derivations (computing the caching values, sorting and inserting the chunks into cache) shows the following time complexity for edge caching during $|T|$ time slots:

$$T_{RBCC} \in O(|T|((\frac{Q_M}{C \cdot R_{min}} + S)(S(|T|+1) + log(\frac{Q_M}{C \cdot R_{min}} + S) + 1))) \qquad (24)$$

where Q_M is fixed cache size at edge server. Also, the caching at mobile devices using the LRU policy results in time complexity of $O(|T| \cdot S \cdot (\frac{Q_D}{C \cdot R_{min}} + 1) \cdot (log(\frac{Q_D}{C \cdot R_{min}} + 1) + 1))$ where, Q_D is the fixed cache size at mobile device.

It is also noteworthy to mention that at each time slot, the proposed algorithm checks for the lowest possible bitrate for transrating at either local or neighborhood edge server. Furthermore, the algorithm compares the cost of edge processing with the cost of downloading from the backhaul network and makes the optimal decision. These algorithmic insights verify that our proposed algorithm ensures the convergence to high quality suboptimal solutions when the number of mobile clients or edge servers dramatically increases.

VI. Simulation Results

In this section, we conduct the simulations to evaluate the performance of the proposed strategy compared to the exiting alternatives approaches. Our simulation objectives are to compare four following edge-assisted DASH strategies:

- **Collaborative edge caching with LRU cache replacement (CC_LRU)** [8]: Following this strategy, the edge servers collaborate with each other to serve the clients request without the edge processing and D2D capabilities. LRU reactive replacement strategy is employed to update the cache contents at each server.
- **Collaborative edge caching and processing with LRU cache replacement (CCP_LRU)** [10]: Under this strategy, the potential of collaborative edge processing is also utilized in addition to the collaborative edge caching and also, the LRU policy is used to update the cache contents.
- **Collaborative edge caching and processing with D2D using LRU (CCP-D2D_LRU)**: The proposed collaborative edge caching and processing with D2D solution in this work which uses the LRU policy for updating the cache contents at both edge servers and mobile devices.
- **Collaborative edge caching and processing with D2D using RBCC heuristic (CCP-D2D_RBCC)**: The proposed strategy which employs the RBCC and LRU

TABLE II: Simulation parameters and their values.

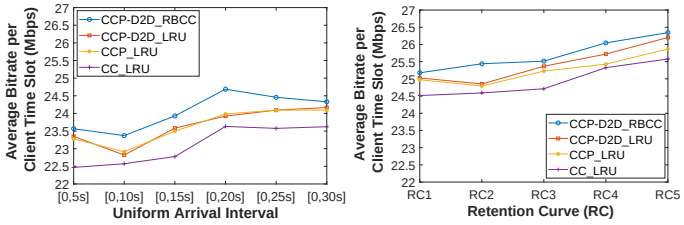
Simulation Parameter	Corresponding Value
Number of UEs	*30*
Number of eNodeBs	*5*
UE antenna gain	*0 dBi*
eNodeB antenna gain	*18 dBi*
UE speed	*4 mps*
Maximum Tx power per UE	*26 dBm*
Channel bandwidth	*5 MHz*
Number of downlink RBs	*28*
Scheduler	*Proportional Fairness*
Channel model	*Urban Macrocell*
Shadowing	*Disabled*
Simulation time	*300s*
Time slot duration	*1s*
Video chunk size	*5s*
Fairness threshold (δ_F)	*0.5*
Edge server cache size	*3Gb*
Mobile device cache size	*0.3Gb*
Edge server processing capacity	*3Gb*
Processing weighting (ϕ)	*1*

heuristics for updating the cache contents at the edge servers and the mobile devices, respectively.

A. Simulation Setup

We consider a mobile edge computing scenario with 5 deployed edge servers (5 associated eNodeBs) and 30 video streaming mobile clients (UEs) during $|T| = 300$ time slots scheduling with the time duration of $\Delta t = 1s$ for each time slot. The set of edge servers form one single cluster which means every edge server is the neighborhood of the other servers. Under the urban macrocell wireless channel model specification [3GPP TR 36.814 V.9.0.0 2010] [32], the mobile clients (UEs) keep the constant speed of $4mps$ and their downlink and D2D signal-to-noise ratio (SNR) traces during 300 time slots are obtained from the third party simulator SimuLTE [31]. The instantaneous fluctuating effective throughputs of the clients are then obtained using the Shannon upper bound approximation and are used as input to the algorithm (Algorithm 1). We note that a moderate number of UEs and eNodeBs are used for the simulations due to long time duration for SNR data collection from SimuLTE when the number of clients or base stations gets larger. However, our solution works with large scale deployments too since the proposed algorithm uses the obtainable downlink and D2D SNRs for throughput computation in an online manner.

Four videos with different popularities are divided into the consecutive chunks each with fixed size $C = 5s$ and are initially available at the origin server in ten different bitrates $R = \{15, 17, 22, 26, 30, 35, 38, 43, 45, 50Mbps\}$. Although the number of videos is randomly chosen, the proposed system model and algorithm easily support large number of videos. Furthermore, the set of 4K $\sim$ 5K video bitrates will be prevalent in future mobile networks thanks to the availability of high bandwidth brought by 5G/B5G. The arrival and departure times of the clients are randomly chosen from the uniform time intervals and, the polynomial $p(t) = at^2 + bt + c$ is used for modeling the retention behaviors (curves) of the clients with respect to different video contents. Here $p(t)$ states the probability that the client remains active in its streaming session at time slot t. Different retention curves are generated by varying the curvature of polynomial $p(t)$ and determining the values of coefficients a, b and c. The equal weightings $\alpha = \beta = \theta = 1/3$ are also used for the optimization terms in (6). The list of parameters settings for simulations have been illustrated in Table. II. We also note that at each part, the average of the results taken over 20 runs of the simulation with confidence interval of 95% are presented.



(a) Arrival Time (b) Retention Curve

Fig. 2: Comparison between four bitrate adaptation and caching strategies in terms of average video bitrate for different (a) arrival intervals and (b) retention curves.

B. Video Bitrate

We first analyze the average video bitrate per client time slot with the four above-mentioned bitrate adaptation and caching strategies. Fig. 2a shows results with six different uniform intervals for the arrival of the clients. Fig. 2b shows similar results with five different retention curves. We can see that the integration of D2D into the joint edge caching and processing only marginally improves the video bitrate. Using the RBCC cache replacement heuristic at the edge servers instead of LRU increases the average bitrate a tiny bit more.

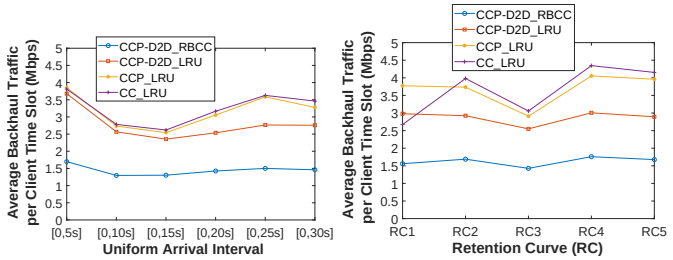
C. Video Data Traffic

Next, we compared the bitrate adaptation and caching strategies in term of average backhaul data traffic per client time slot. Again we consider different uniform intervals for the arrival time of the clients as well as different retention curves. The results are plotted in Figures 3a and 3b, respectively.

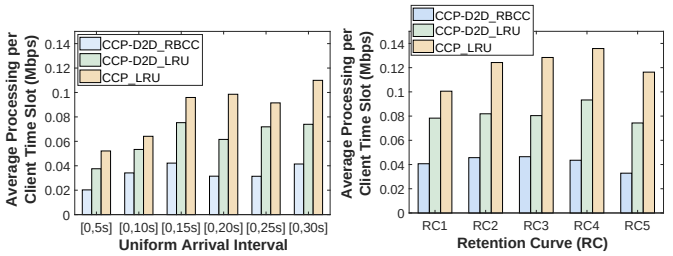
Now we observe that the integration of D2D into the collaborative edge caching and processing really helps to alleviate load from the backhaul network. The traffic is reduced by 18% on average. The reason is obviously that clients have the opportunity of downloading some of the requested chunks from the neighborhood devices through D2D communication therefore helping to reduce the volume of the downloaded video contents from the origin server. A more striking result is that using RBCC instead of LRU at the edge servers delivers an additional 45% reduction in the backhaul traffic.

D. Processed Data

We have further evaluated the performance of the proposed schema in term of average processed data per client time slot.



(a) Arrival Interval (b) Retention Curve



(c) Arrival Interval (d) Retention Curve

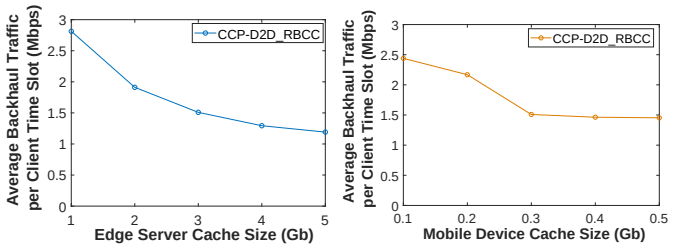
Fig. 3: Comparison between four strategies in terms of average backhaul traffic for different (a) arrival intervals and (b) retention curves. Comparison results in term of average processed data for different (c) arrival intervals and (d) retention curves.

The results with different arrival intervals of the clients and retention curves are shown in Figures 3c and 3d, respectively. They suggest that D2D communication combined with collaborative edge caching and processing reduces the average processed data per client time slot by approximately 30%. This is because D2D provides the possibility of fetching some of the locally missing video chunks from the neighborhood clients therefore contributing to the less amount of data processing at the network edges. Similar to the case of backhaul traffic, the RBCC heuristic yields further benefits of about 47.5%.

Although the impact of using RBCC heuristic tends to be more than the impact of D2D functionality with the current simulation setup, our expectation is that the integration of D2D into our system introduces bigger improvements when larger number of clients with slightly lower mobility speed are considered. The reason is that large number of clients with slightly lower speed increases the instantaneous neighborhood density for each client which in turn increases the opportunities for fetching the locally missing content from the cache of neighborhood devices. With the same size of cache at edge servers, this implies that under such circumstances, the impact of RBCC heuristic for cache replacement at the edges will be smaller compared to the impact of D2D functionality.

E. Impact of Cache Size

We also show some results on the impact of cache size at both edge servers and mobile devices on the average backhaul data traffic per client time slot in Figures 4a and 4b. Interestingly, the increase of cache size at the edge server is more beneficial in reducing the backhaul traffic than increasing the cache size at the mobile device. The reason is that the set



(a) Average Backhaul Traffic (b) Average Backhaul Traffic

Fig. 4: Impact of increasing the cache size at (a) the edge server (b) at the mobile device on average backhaul data traffic.

of clients assigned to an edge server change less frequently compared to the neighborhoods of a client.

F. Impact of Edge Processing Capacity

Finally, we examine the impact of processing capacity of the edge servers on the average backhaul data traffic. Fig. 5 visualizes the results for different edge processing capacities. Increasing the processing capacity at the edges provides more possibilities for transrating the clients' requested bitrates at the edge servers and therefore leads to less data traffic on the backhaul network. The figure shows that no further backhaul traffic reduction is obtained by increasing the processing capacity beyond $1.5Gb$. The optimal edge processing capacity is however upper bounded by the system parameters and the setup. This optimal point can indeed help the system designers to decide on the optimal edge processing capacity in edge computing assisted DASH.

Another interesting observation from the results in Fig. 4a and 5 is that the effect of increasing the edge cache size is bigger than that of increasing the edge processing capacity in term of backhaul traffic reduction. The reason is that in most of the cases, downloading from the origin server is less costly than the edge processing and therefore, increasing the edge cache size is more beneficial in reducing the traffic. Although the edge storage and processing capacity are practically in different units, this observation provides a guideline for edge system designers to choose the economically optimal solution.

VII. Conclusion and Future Work

In this paper, we propose joint video caching and processing at mobile network edge combined with D2D communication for adaptive video streaming. We formulate a joint optimization problem that considers the QoE of video clients, the backhaul data traffic, and the processed data at the network edges. An efficient low-complexity algorithm with minimum need for parameter tuning is presented to solve the optimization problem in a suboptimal way. Our solution also utilized the advantage of a retention-based collaborative caching heuristic for periodically updating the cache contents at the edge servers. Simulation results using SNR traces of mobile clients show that the proposed combination of mechanisms with the joint optimization outperforms the other edge-assisted DASH strategies in terms of the average backhaul data traffic, the

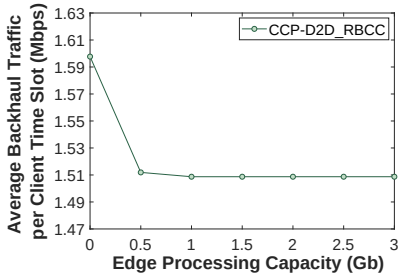


Fig. 5: Impact of increasing the edge processing capacity on the average backhaul traffic per client time slot.

average processed video data at the edges, while maintaining practically the same average video bitrate.

As future work, we wish to derive analytically the optimal edge processing capacity where the minimum backhaul traffic is obtained according to the system parameters. Furthermore, in the present system, mobile devices employ the LRU heuristic for periodically updating their cache contents without collaboration between the neighborhoods. We plan to study whether switching to use the RBCC heuristic also at the mobile devices' local cache leads to improved system performance.

ACKNOWLEDGMENT

This research has been financially supported by Lacrimosa project grant number 297892 and the Nokia Center for Advanced Research.

REFERENCES

[1] I. Cisco, Cisco visual networking index: Forecast and methodology, 20162021, CISCO White paper, Sep. 2017.

[2] 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 Conference on SIGCOMM (SIGCOMM'14)*, pp. 187-198, Aug. 2014.

[3] 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 International Workshop on Mobile Video*, pp. 1-6, May 2016.

[4] T. Hossfeld, S. Egger, R. Schatz, M. Fiedler, K. Masuch, and C. Lorentzen, "Initial delays vs. interruptions: Between the devil and deep blue", *in Proc. IEEE International Workshop on Quality of Multimedia Experience (QoMEX)*, pp. 1-6, Aug. 2012.

[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. Surveys & Tuts.*, vol. 17, no. 1, pp. 469-492, First Quarter 2015.

[6] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal, "Mobile-edge computing", ETSI Introductory Technical White Paper, Sep. 2014.

[7] M. Agiwal, A. Roy and N. Saxena, "Next generation 5G wireless networks: A comprehensive survey", *IEEE Commun. Surveys & Tuts*, vol. 18, no. 3, pp. 1617-1655, third quarter 2016.

[8] T. X. Tran, and D. Pompili, "Octopus: A cooperative hierarchical caching strategy for cloud radio access networks", *in Proc. 13th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 154-162, Oct. 2016.

[9] A. Mehrabi, M. Siekkinen, and A. Ylä-Jääski, "Edge computing assisted adaptive mobile video streaming", *IEEE Trans. Mob. Comput.*, vol. 18, no. 4, pp. 787-800, Apr. 2019.

[10] T. X. Tran, P. Pandey, A. Hajisami, and D. Pompili, "Collaborative multi-bitrate video caching and processing in mobile-edge computing networks", *in Proc. 13th Annual IEEE Conference on Wireless On-demand network systems and services (WONS)*, pp. 165-172, Feb. 2017.

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

[12] D. Liu, B. Chen, C. Yang, and A. F. Molisch, "Caching at the wireless edge: Design aspects, challenges and future directions", *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 22-28, Sep. 2016.

[13] X. Wang, M. Chen, T. Taleb, A. Ksentini, V. C. M. Leung, "Cache in the air: Exploiting content caching and delivery techniques for 5G systems", *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 131-139, Feb. 2014.

[14] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen and W. Zhu, "Understanding performance of edge content caching for mobile video streaming", *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1076-1089, May 2017.

[15] X. Li, P. Wu, X. Wang, K. Li, Z. Han and V. C. M. Leung, "Collaborative hierarchical caching in cloud radio access networks", *in Proc. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 462-467, May 2017.

[16] H. A. Pedersen and S. Dey, "Enhancing mobile video capacity and quality using rate adaptation, RAN caching and processing", *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 996-1010, Apr. 2016.

[17] M. Ji, G. Caire and A. F. Molisch, "Wireless device-to-device caching networks: Basic principles and system performance", *IEEE J. Sel. Areas Commun.*, vol. 34, no. 1, pp. 176-189, Jan. 2016.

[18] Tian, Guibin, and Y Liu. "Towards agile and smooth video adaptation in dynamic HTTP streaming", *in Proc. 8th ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT'12)*, pp. 109-120, Dec. 2012.

[19] S. Akhshabi, A C. Begen, and C. Dovrolis. "An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP", *in Proc. of the Second Annual ACM conference on Multimedia Systems (MMSys'11)*, pp. 157-168, Feb. 2011.

[20] J. Kua, G. Armitage and P. Branch, "A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP", *IEEE Commun. Surveys & Tuts.*, vol. 19, no. 3, pp. 1842-1866, third quarter 2017.

[21] 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 International Conference on Multimedia Systems (MMSys16)*, pp. 1-12, May 2016.

[22] A. Bentaleb, A. C. Begen, and R. Zimmermann, SDNDASH :Improving QoE of HTTP adaptive streaming using software defined networking, *in Proc. 2016 ACM Conference on Multimedia (MM16)*, pp. 1296-1305, Oct. 2016.

[23] N. Golrezaei, A. F. Molisch, A. G. Dimakis and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution", *IEEE Commun. Mag.*, vol. 51, no. 4, pp. 142-149, Apr. 2013.

[24] N. Golrezaei, P. Mansourifard, A. F. Molisch and A. G. Dimakis, "Base-station assisted device-to-device communications for high-throughput wireless video networks", *IEEE Trans. Wireless Commun.*, vol. 13, no. 7, pp. 3665-3676, Jul. 2014.

[25] D. Malak, M. Al-Shalash, and J. G. Andrews, "Optimizing content caching to maximize the density of successful reception in device-to-device networking", *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4365-4380, Oct. 2016.

[26] T. Deng, L. You, P. Fan, and D. Yuan, "Device caching for network offloading: Delay minimization with presence of user mobility", *IEEE Wireless Commun. Lett.*, vol. 7, no. 4, pp. 558-561, Aug. 2018.

[27] J. Liu, B. Bai, J. Zhang, K. B. Letaief, and Y. Li, "Joint device caching and channel allocation for D2D-assisted wireless content delivery", *in Proc. IEEE Int. Conf. Commun.*, pp. 1-6, Jul. 2018.

[28] Sami Kekki, *et. al*, "MEC in 5G networks", *ETSI White Paper, First Edition, No. 28*, Jun. 2018.

[29] Y. Wang, X. Tao, X. Zhang and Y. Gu, "Cooperative caching placement in cache-enabled D2D underlaid cellular network", *IEEE Commun. Lett.*, vol. 21, no. 5, pp. 1151-1154, May 2017.

[30] ISO/IEC 23009-5. 2017. Dynamic adaptive streaming over HTTP (DASH) Part 5: Server and network assisted DASH (SAND). (2017)

[31] http://www.simulte.com

[32] http://www.etsi.org/deliver/etsi_tr/136900_136999/136942/08.02.00_60/tr_136942v080200p.pdf