Allies: Tile-Based Joint Transcoding, Delivery and Caching of 360° Videos in Edge Cloud Networks

Jianxin Shi[1,2,3], Lingjun Pu[1,2,3], Jingdong Xu[1,3]
[1]Institute of Systems and Networks, College of Computer Science, Nankai University, Tianjin, China
[2]State Key Laboratory of Integrated Services Networks (Xidian University), Xi'an, China
[3]Tianjin Key Laboratory of Network and Data Security Technology, Tianjin, China
The corresponding author is Lingjun Pu (pulingjun@nankai.edu.cn)

***Abstract*—360° or panoramic video applications have seen booming development and absorbed great attention in recent years. However, as they are of significant size and usually watched from a close distance, they require an extremely higher bandwidth and frame rate for a good immersible experience, which poses a great challenge on mobile networks. Realizing the great potentials of tile-based transcoding, viewport adaptive streaming, and edge caching, we propose *Allies*, a tile-based joint transcoding, delivery, and caching framework for 360° video services in edge cloud networks. Meanwhile, an innovative idea about 360° video caching way is proposed and applied to improve the cache utilization of edge clouds. In this framework, we formulate the joint optimization problem as an integer nonlinear program and propose a greedy suboptimal algorithm with polynomial running time to minimize video service costs. Finally, extensive simulations with real user's head movement traces and corresponding 360° video datasets corroborate the efficiency, flexibility, and lightweight of our proposed algorithm; for instance, it achieves over 25% performance improvement compared to state-of-the-art works in various system settings.**

***Keywords*-transcoding; viewport adaptive streaming; caching; request scheduling; edge cloud.**

I. Introduction

Recent advances in smart mobile devices have laid the foundation to support a broad range of 360° or panoramic video services such as Virtual Reality (VR) and Augmented Reality (AR). Based on the prediction of Cisco Visual Networking Index [1], virtual and augmented reality mobile data traffic will grow nearly 12-fold from 22 petabytes per month in 2017, to 254 petabytes per month in 2022. Besides, panoramic video services need to delivery the ultra-high resolution (up to 12K) and high frame rate (up to 100 fps) full-degree 360° video to ensure a good immersion experience [2]. However, the growing mobile data traffic and the significant size of 360° video bring a great challenge for future mobile networks. There is an urgent need for innovations in the operations (e.g. content delivery technologies and novel network architectures) for 360° video services.

Viewport Adaptive Streaming (VAS) is regarded as a promising deliver way for the panoramic video streaming. In 360° video services, the two-dimensional panoramic video is mapped to the internal surface of a 3D-Sphere [3]. The user in the center of sphere only views a portion projection scene, defined as viewport or Filed of View (FoV). In VAS, 360° video is spatially cropped into tiles that can be delivered and encoded independently. Then, it is performed that high-quality (e.g. resolution and bitrate) tiles reserve within the FoV, while the margin area around FoV is delivered in low-quality or even discarded [4]. VAS greatly decreases data transmission and saves bandwidth consumption. However, ever-increasing video services cause severe transmission burden and network congestion for backhaul links due to content retrieval from remote content servers or Content Delivery Networks (CDN). This implies that reforming transmission mode solely is still insufficient for 360° video services.

Cloud Radio Access Network (C-RAN) [5] and edge caching [6] is proposed as a novel architecture for future mobile networks. Specifically, edge facilities (e.g. edge clouds and cloudlets) can cache popular videos to alleviate the traffic load of backhaul links and perform transcoding of a video to different versions to satisfy user requests. Moreover, the deployment about adaptive video streaming technologies for standard videos can satisfy heterogeneous demands of users in these novel network architectures [7]. However, deploying VAS in edge facilities solely is ill-suited for 360° video services, because caching a video with diverse quality versions will run out of storage space rapidly. Therefore, seeking a novel and efficient 360° video caching strategy is significantly urgent and meaningful in edge cloud networks.

In this paper, realizing the novel technologies and open issues above, we propose *Allies*, a tile-based joint transcoding, delivery, and caching framework deployed in edge cloud networks, to minimize service costs. Here, edge clouds adopt an innovative caching strategy to efficiently store the popular video contents and provide transcoding computing service. VAS technology is applied to reduce video data transmission and the tile-based diverse request scheduling is provided to ensure a good immersible experience. In order to reap profound benefits, we need to solve three critical issues: **1) video cache placement, 2) tile quality cache placement, and 3) request scheduling.** In *Allies*, we formulate a tile-based collaborative optimization problem and propose a suboptimal algorithm that approaches the solution within polynomial running time and significantly reduces service deployment costs. In summary, we make the following contributions in this article:

- We creatively propose an innovative idea that each 360°

video is stored at most once in each edge cloud, but its tiles can be cached with heterogeneous quality to efficiently utilize storage resources (Section III).

- We propose *Allies* and formulate the tile-based joint transcoding, delivery and caching problem as an Integer Nonlinear Program (INLP) that minimizes total service deployment costs of edge clouds (Section III).
- We decompose INLP into a serious of subproblems via the divide and conquer technique and present low complexity solutions via the greedy method (Section IV).
- Extensive experiments show our solution achieves over 25% performance improvement compared with state-of-the-art works in various system settings (Section V).

II. Related Work

The existing studies related to our work can be grouped into three categories: (1) viewport adaptive streaming; (2) edge caching; (3) collaborative optimization strategies.

Viewport adaptive streaming: The panoramic video generally has a high bitrate and frame rate due to the abundant content and the unique viewing pattern. The viewpoint adaptive steaming is proposed to reduce video traffic data. In [8], the authors fetched only portions of a panoramic scene that cover what a viewer is about to perceive and determined their corresponding qualities. [9] proposed that only tiles belonging to the viewport are streamed at the highest quality and other tiles are instead streamed at a lower quality. [10] proposed a novel adaptive video streaming method for 360° videos using scalable video coding. Those theories reduce the occurrence of rebuffering on links with varying bandwidth without compromising playback quality or bandwidth efficiency. However, they only consider the single video streaming and ignore that the ever-increasing mobile users cause serious transmission burden for limited backhaul links. Moreover, extensive content retrieval from remote content servers or CDN occasion network congestion and result in poor user experience.

Edge caching: Edge caching is an emergent architecture on the edge of mobile networks. It provides computing and storage services [6]. It is an important component in 5G architecture, which supports a variety of innovative applications with the ultra-low latency requirement such as virtual and augmented reality. [11] combining Mobile Edge Computing (MEC) and 360° video steaming indicated that caching popular content close to the end-users can decrease network latency and alleviate network bandwidth demands. The reason is that it reduces the number of future requests that have to be sent to remote content servers or CDN. In [12], the authors presented the benefits of using edge computing to save cost, bandwidth, and energy for the interactive media and video streaming. However, edge cloud generally has limited cache capacity. Deploying VAS at edge cloud solely is unreasonable and extremely restricts the utility of edge caching, because that a video cached with diverse quality versions will use up storage resources quickly.

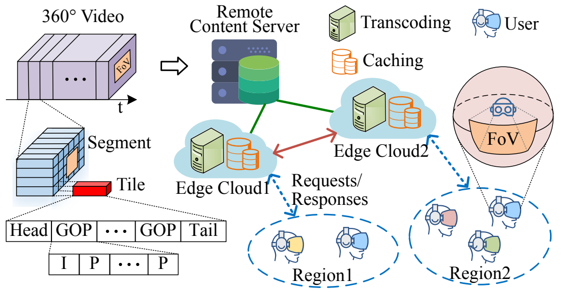


Fig. 1. Illustration of a typical edge cloud network.

Collaborative optimization strategies: In [13], the authors explored fundamental trade-offs between caching, computing, and communication for emerging VR/AR applications of broad societal impact. [14] presented a novel MEC-based mobile VR delivery framework to minimize the average transmission rate. However, these works mainly focus on offloading rendering computing to the edge cloud but do not consider the viewpoint adaptive streaming. In this work, we consider the property of smart mobile devices is enough to complete the 360° video playback computing. The authors in [15] studied joint radio communication, caching and computing decision problem, aiming to maximize the average tolerant delay. [16] proposed joint Computing, Caching, Communication, and Control (4C) collaboration at the edge with MEC server to optimize a linear combination of bandwidth consumption and network latency. The authors in [17] proposed a collaborative joint caching and transcoding scheme in mobile edge networks to minimize the user's perceived delay. However, these solutions focus on common videos. The unique viewing pattern, ultra-high bitrate and high frame rate of 360° videos make it complex.

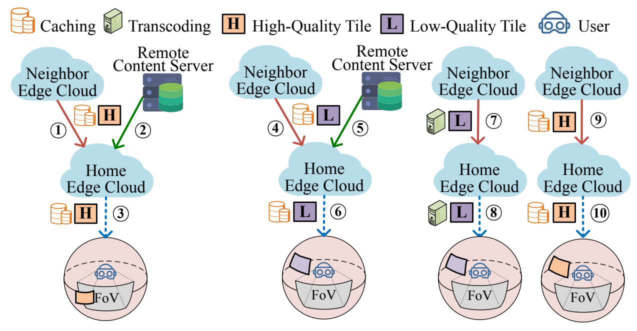
III. System Model and Problem Formulation

In this section, we present the distributed caching and request scheduling framework *Allies* deployed in the edge cloud network and followed by the settings of the considered model. And then, the tile-based joint transcoding, delivery and caching optimization problem is formulated.

A. System Architecture and 360° Videos

As depicted in Fig. 1, a typical edge cloud network consists of a set $\mathcal{N}=\{1,2,....,N\}$ of edge clouds (ECs) and a remote content server (CS). The ECs are connected to each other via high-bandwidth and low-latency links (e.g. packet-based fiber optic networks). Each EC $n \in \mathcal{N}$ is connected to the CS via backbone networks [5] and provides transcoding and caching services. The collaborative optimization framework *Allies* mainly contains three parts, which are the transcoding unit, the delivery unit, and the caching unit. *The transcoding unit* can transcode the cached video or tile to the desired quality and push it out to the output buffer. *The delivery unit* provides diverse request scheduling ways to achieve viewport adaptive streaming. *The caching unit* makes a strategic decision to store some popular videos and share cached content with other peer caching units (from other ECs).

The collection of 360° videos available for request is indexed as $\mathcal{V}=\{1,2,...,V\}$. All videos are pre-recorded. Each



(a) Tile within FoV (b) Tile within margin area

Fig. 2. Illustration of possible scheduling events that happen when a tile request arrives at home edge cloud.

video $v \in \mathcal{V}$ is temporally divided into a series of consecutive segments denoted by the set $\mathcal{S}_v = \{v_1, v_2, ..., v_S\}$, and each segment $v_s \in \mathcal{S}_v$ is spatially divided into $\mathcal{K}\times\mathcal{M}$ tiles denoted by the set $\mathcal{T}_{vs} = \{v_{s1}, v_{s2}, ..., v_{s(\mathcal{K}\times\mathcal{M})}\}$. Each tile $v_{st} \in \mathcal{T}_{vs}$ with the granularity of segment includes a series of I (intra) frames and P (predicted) frames (as illustrated in Fig. 1). Each tile of video can be stored with two encoding versions: high-quality level and low-quality level, which are denoted as $Q^h_{v_{st}}$ and $Q^l_{v_{st}}$. The size of high-quality and low-quality tile is separately denoted as $D^h_{v_{st}}$ (MB) and $D^l_{v_{st}}$ (MB). We consider that the high-quality tile can be transcoded into the low-quality, and high-quality level could be adaptive across different segment transmissions (e.g. when taking DASH into account).

B. Cache Placement and Request Scheduling

In the delivery unit of *Allies*, each user only associates and transports data with the EC in the same region, which is later referred to as the user's *home* EC. Besides, each EC not only acts as a server to respond to user requests but also as a client to retrieve video contents from the CS or other ECs (them are later referred to as the user's *neighbor* ECs). To model the video cache placement decision, we define the variable $x_{nv} \in \{0, 1\}$ where $x_{nv} = 1$ indicates that the video v is cached in the EC n; and $x_{nv} = 0$ otherwise. We creatively propose an innovative idea that tiles of the video v can be cached with heterogeneous quality. We define the variable $y_{nv_{st}} \in \{0, 1\}$ to model the tile quality cache placement decision where $y_{nv_{st}} = 1$ indicates that the tile v_{st} is cached in the EC n with high-quality version $Q^h_{v_{st}}$; and $y_{nv_{st}} = 0$ otherwise. We consider each video is integrally stored. Thus, the cache state of low-quality version tile $Q^l_{v_{st}}$ can be expressed as the formula $x_{nv} - y_{nv_{st}}$. Besides, the tile quality cache placement decision could not exceed the corresponding decision of video placement, which is expressed as the following constraint:

$$y_{nv_{st}} \le x_{nv}. \tag{1}$$

According to VAS, tiles within user's FoV must be served with high-quality version and tiles within margin area can be allocated with low-quality version to handle the case that the user may occasionally show an orientation against the prediction. When the requested tile v_{st} is in FoV, we introduce variables $\{a_{nv_{st}}, b_{nv_{st}}, c_{nv_{st}}\} \in \{0, 1\}$ to model possible request scheduling/response events (as illustrated in Fig. 2(a)):

1) $a_{nv_{st}} = 1$ (③) indicates that the requested tile is directly obtained high-quality version $Q^h_{v_{st}}$ from cache of home EC; and $a_{nv_{st}} = 0$ otherwise.
2) $b_{nv_{st}} = 1$ (①③) indicates that the requested tile is gained high-quality version $Q^h_{v_{st}}$ by retrieving from cache of neighbor ECs; and $b_{nv_{st}} = 0$ otherwise.
3) $c_{nv_{st}} = 1$ (②③) indicates that the requested tile is achieved high-quality version $Q^h_{v_{st}}$ by retrieving from cache of the CS; and $c_{nv_{st}} = 0$ otherwise.

Similarly, we introduce binary variables $\{d_{nv_{st}}, e_{nv_{st}}, f_{nv_{st}}, g_{nv_{st}}, h_{nv_{st}}, i_{nv_{st}}, j_{nv_{st}}\} \in \{0, 1\}$ to indicate possible request scheduling events that happen when the requested tile v_{st} is in the margin area (as illustrated in Fig. 2(b)):

1) $d_{nv_{st}} = 1$ (⑥) indicates that the requested tile is directly obtained low-quality version $Q^l_{v_{st}}$ from cache of home EC; and $d_{nv_{st}} = 0$ otherwise.
2) $e_{nv_{st}} = 1$ (⑧) indicates that the requested tile is gained low-quality version $Q^l_{v_{st}}$ by transcoding high-quality version at home EC; and $e_{nv_{st}} = 0$ otherwise.
3) $f_{nv_{st}} = 1$ (⑩) indicates that the requested tile is directly achieved high-quality version $Q^h_{v_{st}}$ from cache of home EC; and $f_{nv_{st}} = 0$ otherwise.
4) $g_{nv_{st}} = 1$ (④⑥) indicates that the requested tile is obtained low-quality version $Q^l_{v_{st}}$ by retrieving from cache of neighbor ECs; and $g_{nv_{st}} = 0$ otherwise.
5) $h_{nv_{st}} = 1$ (⑦⑧) indicates that the requested tile is gained low-quality version $Q^l_{v_{st}}$ by transcoding high-quality version at some neighbor EC; and $h_{nv_{st}} = 0$ otherwise.
6) $i_{nv_{st}} = 1$ indicates (⑨⑩) that the requested tile is achieved high-quality version $Q^h_{v_{st}}$ by retrieving from cache of neighbor ECs; and $i_{nv_{st}} = 0$ otherwise.
7) $j_{nv_{st}} = 1$ indicates (⑤⑥) that the requested tile is obtained low-quality version $Q^l_{v_{st}}$ by retrieving from cache of the CS; and $j_{nv_{st}} = 0$ otherwise.

In order to ensure that only one request scheduling event will be responded for the requested tile belonging in FoV or margin area, we impose the following constraints:

$$a_{nv_{st}} + b_{nv_{st}} + c_{nv_{st}} = 1, \tag{2}$$

$$d_{nv_{st}} + e_{nv_{st}} + f_{nv_{st}} + g_{nv_{st}} + h_{nv_{st}} + i_{nv_{st}} + j_{nv_{st}} = 1. \tag{3}$$

The complexity of request redirect and cache replacement have a significant impact on system utility. To decrease too many request redirects and reduce the complication of cache replacement, we consider the principle of video cache consistency, which means that the same tile v_{st} cached in different ECs is stored with the same quality. This can be expressed as the following constraint ($n' \in \mathcal{N}$):

$$x_{nv} y_{n'v_{st}} = x_{n'v} y_{nv_{st}}. \tag{4}$$

To describe the limited cache capacity of EC, we assume that each EC allocates a cache size of S_n (MB). In this case, the total size of cached videos at the EC n could not exceed its caching capacity, which is denoted as the following constraint:

$$\sum_v \sum_s \sum_t y_{nv_{st}} D^h_{v_{st}} + (x_{nv} - y_{nv_{st}}) D^l_{v_{st}} \le S_n. \tag{5}$$

C. Cost Model

In this section, we focus on the 360° video service costs of ECs, which is related to video popularity and viewing probability of tile. The symbol $P_{v_{st}} \in [0,1]$ denotes the probability for the tile v_{st} being in FoV, which can be gained from users' historical viewing traces. It describes different preferences users for the same video content [11]. And then, the viewing probability for the tile within margin area is $1-P_{v_{st}}$. For a cache efficiency perspective, our objective of optimization policy is to minimize the total service costs. Here, we study the following four kinds of cost consumption when the home EC n response to the requested tile v_{st} .

Caching Cost, $C_{nv_{st}}$: We adopt a cache size proportional model. The caching cost of tile v_{st} at EC n is given the follows, where α (Units/MB) indicates unit caching cost:

$$C_{nv_{st}} = \alpha[y_{nv_{st}} D^h_{v_{st}} + (x_{nv} - y_{nv_{st}}) D^l_{v_{st}}]. \tag{6}$$

Delivery Cost, $F_{nv_{st}}$ and $M_{nv_{st}}$: Notice the fact that it is an inevitable process to deliver the basic requested tile v_{st} from home EC to user under aforementioned scheduling events. It means this part contributes to the same delivery cost for total costs. For simplicity, we omit this basic transmission part and focus on the delivery cost caused by retrieving content from the CS or neighbor ECs and transporting redundant content data. We consider the delivery cost represents the delay cost or bandwidth cost, which is proportional to the data size transported by network. These symbols μ_{ce} (Units/MB), μ_{ee} (Units/MB) and μ_{eu} (Units/MB) respectively denote the unit delivery cost of links from CS to EC, from EC to EC, and from EC to user. The delivery cost of tile within FoV is denoted as $F_{nv_{st}}$ and the delivery cost of tile within margin area is indicated as $M_{nv_{st}}$:

$$F_{nv_{st}} = P_{v_{st}}(b_{nv_{st}}\mu_{ee} + c_{nv_{st}}\mu_{ce}) D^h_{v_{st}}, \tag{7}$$

$$\begin{aligned} M_{nv_{st}} = (1-P_{v_{st}})\{&[(g_{nv_{st}} + h_{nv_{st}})\mu_{ee} + j_{nv_{st}}\mu_{ce}] D^l_{v_{st}} \\ &+ i_{nv_{st}}\mu_{ee} D^h_{v_{st}} + (f_{nv_{st}} + i_{nv_{st}})\mu_{eu}(D^h_{v_{st}} - D^l_{v_{st}})\} \end{aligned} \tag{8}$$

Transcoding Cost, $T_{nv_{st}}$: We consider the transcoding cost mainly focus on the computing resource consumption. The symbol β (Units/GHz) denotes the unit computing resource cost and $\gamma_{v_{st}}$ (GHz) indicates consumed computing resources for transcoding operation of the tile v_{st}. The transcoding cost of requested tile can be given as the following formula:

$$T_{nv_{st}} = \beta(1-P_{v_{st}})(e_{nv_{st}} + h_{nv_{st}})\gamma_{v_{st}}. \tag{9}$$

D. Problem Formulation

In *Allies*, we formulate the joint transcoding, delivery and caching optimization problem to optimize video cache placement variable x_{nv}, tile quality cache placement variable $y_{nv_{st}}$, and scheduling decision variables $\{a_{nv_{st}}, b_{nv_{st}}, c_{nv_{st}}, d_{nv_{st}}, e_{nv_{st}}, f_{nv_{st}}, g_{nv_{st}}, h_{nv_{st}}, i_{nv_{st}}, j_{nv_{st}}\}$, aiming at minimizing the total service costs of ECs, satisfying user requests and respecting system constraints. The simulation is mathematically formulated as the problem $\boldsymbol{P_1}$. The symbol $\lambda_{nv} \in [0,1]$ indicates the popularity of video v at EC n, which describes the impact of social network and geographical locations for the user preference [18]. The constraints (10a)-(10f) ensure the availability of request scheduling variables and formula (10g) indicates the binary constraint of optimization variables.

$$\boldsymbol{P_1}: \min \sum_n \sum_v \sum_s \sum_t \lambda_{nv}(C_{nv_{st}} + F_{nv_{st}} + M_{nv_{st}} + T_{nv_{st}})$$

$$s.t. \quad (1)-(5),$$

$$a_{nv_{st}} \le y_{nv_{st}}, \quad \forall n, \forall v, \forall s, \forall t, \tag{10a}$$

$$b_{nv_{st}} \le \sum_{n' \ne n} y_{n'v_{st}}, \quad \forall n, \forall n', \forall v, \forall s, \forall t, \tag{10b}$$

$$d_{nv_{st}} \le x_{nv} - y_{nv_{st}}, \quad \forall n, \forall v, \forall s, \forall t, \tag{10c}$$

$$e_{nv_{st}}, f_{nv_{st}} \le y_{nv_{st}}, \quad \forall n, \forall v, \forall s, \forall t, \tag{10d}$$

$$g_{nv_{st}} \le \sum_{n' \ne n} (x_{n'v} - y_{n'v_{st}}), \quad \forall n, \forall n', \forall v, \forall s, \forall t, \tag{10e}$$

$$h_{nv_{st}}, i_{nv_{st}} \le \sum_{n' \ne n} y_{n'v_{st}}, \quad \forall n, \forall n', \forall v, \forall s, \forall t, \tag{10f}$$

$$\begin{aligned} &x_{nv}, y_{nv_{st}}, a_{nv_{st}}, b_{nv_{st}}, c_{nv_{st}}, d_{nv_{st}}, e_{nv_{st}}, f_{nv_{st}}, g_{nv_{st}} \\ &h_{nv_{st}}, i_{nv_{st}}, j_{nv_{st}} \in \{0,1\}, \quad \forall n, \forall v, \forall s, \forall t. \end{aligned} \tag{10g}$$

The problem $\boldsymbol{P_1}$ is an Integer Nonlinear Program (INLP) and is NP-hard, which can be shown by reduction from a knapsack problem. Meanwhile, there are numerous mutual coupling control variables and constraints. Thus, solving this problem to optimality within polynomial time is extremely challenging. Our goal in this article is to design a low-complexity efficient algorithm to approach the solution. Specific details of our proposed algorithm are presented in the following section.

IV. Proposed Efficient Algorithm

In this section, we develop an efficient algorithm, resorting to *greedy, divide and conquer* techniques, to solve the problem $\boldsymbol{P_1}$ and reduce service costs. Our basic idea is shown in Fig. 3, which consists of following steps.

- **Step1:** We divide original problem $\boldsymbol{P_1}$ into two stages via the divide and conquer technique [19]. The first stage solves the **tile response and quality cache placement problem** for a specific video and the second stage solves the **video cache placement problem** in EC networks.
- **Step2:** In the first stage, the tile response and quality cache placement problem for a specific video can be decomposed into three situations: (1) $\boldsymbol{S_1}$, tile response and quality cache placement at home EC; (2) $\boldsymbol{S_2}$, tile response from neighbor ECs; (3) $\boldsymbol{S_3}$, tile response from the CS. Each of them is formulated with current system information and solved within polynomial running time.
- **Step3:** In the second stage, based on the results of first stage, we propose a *Greedy Video Cache Placement (GVCP)* algorithm to solve the video cache placement problem $\boldsymbol{P_4}$ within polynomial running time. Finally, the results of all subproblems constitute the entire solution of original problem.

A. Tile Response and Quality Cache Placement

In the first stage, the three situations of tile response and quality cache placement for a specific video are formulated, aiming to minimize service costs of the given video v.

1) $\boldsymbol{S_1}$, Tile response and quality cache placement at home EC: In this scenario, we envision that the requested video v

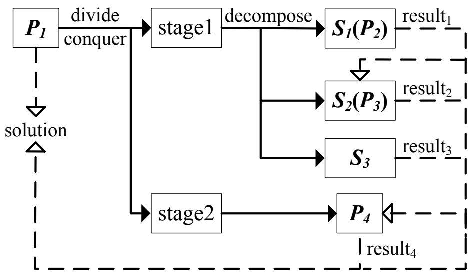


Fig. 3. Basic idea of our proposed algorithm (Black arrows) and main route of calculation results (Dotted arrows)

has been cached at home EC. Due to the constraint (4) about principle of video cache consistency, these tiles of given video can only be obtained from the home EC or the CS. For clarity, we simplify aforementioned variables without reference to the EC and redefine a new variable $j'_{v_{st}} \in \{0,1\}$ to replace $j_{v_{st}}$ in this scenario. Here, $j'_{v_{st}} = 1$ and $j_{v_{st}} = 1$ indicates the same scheduling event in different scenario. This situation contains the caching cost $C^h_{v_{st}}$, the additional delivery cost $F^h_{v_{st}}$ (tile within FoV) and $M^h_{v_{st}}$ (tile within margin area), and the transcoding cost $T^h_{v_{st}}$.

$$C^h_{v_{st}} = \alpha[y_{v_{st}} D^h_{v_{st}} + (1-y_{v_{st}})D^l_{v_{st}}], \tag{11}$$
$$F^h_{v_{st}} = P_{v_{st}}(1-y_{v_{st}})\mu_{ce} D^h_{v_{st}}, \tag{12}$$
$$M^h_{v_{st}} = (1-P_{v_{st}})[f_{v_{st}}\mu_{eu}(D^h_{v_{st}} - D^l_{v_{st}}) + j'_{v_{st}}\mu_{ce} D^l_{v_{st}}], \tag{13}$$
$$T^h_{v_{st}} = \beta(1-P_{v_{st}})e_{v_{st}}\gamma_{v_{st}}. \tag{14}$$

In this scenario, the problem formulation about minimizing service costs of given video v is expressed as follows:

$$\boldsymbol{P_2}: H_v = \min \sum_s \sum_t \omega_{v_{st}} (C^h_{v_{st}} + F^h_{v_{st}} + M^h_{v_{st}} + T^h_{v_{st}})$$
$$s.t. \quad d_{v_{st}} \le 1 - y_{v_{st}}, \quad \forall s, \forall t, \tag{15a}$$
$$e_{v_{st}}, f_{v_{st}} \le y_{v_{st}}, \quad \forall s, \forall t, \tag{15b}$$
$$d_{v_{st}} + e_{v_{st}} + f_{v_{st}} + j'_{v_{st}} = 1, \quad \forall s, \forall t, \tag{15c}$$
$$y_{v_{st}}, d_{v_{st}}, e_{v_{st}}, f_{v_{st}}, j'_{v_{st}} \in \{0,1\}, \quad \forall s, \forall t. \tag{15d}$$

In this formulation, the symbol $\omega_{nv_{st}}$ is scale factor about costs and the size of tile, which can be calculated by the formula $\omega_{v_{st}} = [y_{v_{st}} D^h_{v_{st}} + (1-y_{v_{st}})D^l_{v_{st}}]/D^h_{v_{st}}$. Here, constraints (15a) and (15b) ensure availability of tile response events; constraint (15c) ensures there is only one scheduling event for requested tile within margin area; formula (15d) indicates the binary constraint of variables. The problem is a simple Integer Linear Program (ILP) and can obtain the optimal solution with Alg. 1 in polynomial running time ($\boldsymbol{v_S} \times \boldsymbol{\mathcal{K}} \times \boldsymbol{\mathcal{M}}$). The occupied cache size of video v is calculated as follows:

$$W_v = \sum_s \sum_t y_{v_{st}} D^h_{v_{st}} + (1-y_{v_{st}})D^l_{v_{st}}. \tag{16}$$

2) $\boldsymbol{S_2}$*, Tile response from neighbor ECs:* We envision that the requested video v is cached at a certain neighbor EC in advance rather than the home EC. Based on the constraint (4) and results of problem $\boldsymbol{P_2}$, we already know the tile quality cache placement decision $y_{v_{st}}$. Similarly, we simplify aforementioned variables without reference to EC and redefine new variable $j''_{v_{st}} \in \{0,1\}$ to replace $j_{v_{st}}$ in this situation. $j''_{v_{st}} = 1$ and $j_{v_{st}} = 1$ indicate the same scheduling event in different scenario. Here, the video service costs contain the additional delivery cost $F^n_{v_{st}}$ (tile within FoV) and $M^n_{v_{st}}$ (tile within margin area), and the transcoding cost $T^n_{v_{st}}$.

$$F^n_{v_{st}} = P_{v_{st}}[y_{v_{st}}\mu_{ee} + (1-y_{v_{st}})\mu_{ce}]D^h_{v_{st}}, \tag{17}$$
$$M^n_{v_{st}} = (1-P_{v_{st}})\{[(g_{v_{st}} + h_{v_{st}})\mu_{ee} + j''_{v_{st}}\mu_{ce}]D^l_{v_{st}} + i_{v_{st}}[\mu_{ee} D^h_{v_{st}} + \mu_{eu}(D^h_{v_{st}} - D^l_{v_{st}})]\}, \tag{18}$$
$$T^n_{v_{st}} = \beta(1-P_{v_{st}})h_{v_{st}}\gamma_{v_{st}}. \tag{19}$$

In this situation, the problem of minimizing video service costs can be formulated as follows:

$$\boldsymbol{P_3}: N_v = \min \sum_s \sum_t (F^n_{v_{st}} + M^n_{v_{st}} + T^n_{v_{st}})$$
$$s.t. \quad g_{v_{st}} \le 1 - y_{v_{st}}, \quad \forall s, \forall t, \tag{20a}$$
$$h_{v_{st}}, i_{v_{st}} \le y_{v_{st}}, \quad \forall s, \forall t, \tag{20b}$$
$$g_{v_{st}} + h_{v_{st}} + i_{v_{st}} + j''_{v_{st}} = y_{v_{st}}, \quad \forall s, \forall t, \tag{20c}$$
$$g_{v_{st}}, h_{v_{st}}, i_{v_{st}}, j''_{v_{st}} \in \{0,1\}, \quad \forall s, \forall t. \tag{20d}$$

In the above formulation, the constraints can be explained as follows: constraints (20a) and (20b) ensure availability of tile response events; constraint (20c) ensures that there is only one response event for the requested tile within margin area; formula (20d) indicates the binary constraint of variables. Similarly, this problem also can be solved with Alg. 1.

Algorithm 1 Tile Response and Quality Cache Placement

```
Input: $D^h_{v_{st}}$, $D^l_{v_{st}}$, $\mu_{ce}$, $\mu_{ee}$, $\mu_{eu}$, $\gamma_{v_{st}}$, $P_{v_{st}}$, $\alpha$, $\beta$
Output: $y_{v_{st}}$, $d_{v_{st}}$, $e_{v_{st}}$, $f_{v_{st}}$, $g_{v_{st}}$, $h_{v_{st}}$, $i_{v_{st}}$, $j'_{v_{st}}$, $j''_{v_{st}}$
1: Dividing the problem into a serious of subproblems about tile
   via divide and conquer technique;
2: For each subproblem, solving it with the implicit enumeration
   method;
```

3) $\boldsymbol{S_3}$*, Tile response from the CS:* In this situation, the requested video v is not cached at any ECs in advance. The video service costs can be formulated as formula (21), where R_v indicates total service costs caused by retrieving redundant content data from the CS to home EC, the first item indicates the delivery cost for the tile within FoV, and the second item indicates the delivery cost for the tile within margin area:

$$R_v = \sum_s \sum_t P_{v_{st}}\mu_{ce} D^h_{v_{st}} + (1-P_{v_{st}})\mu_{ce} D^l_{v_{st}}. \tag{21}$$

B. Video Cache Placement

In the second stage, we formulate the video cache placement problem at ECs based on the aforementioned calculation results, aiming to minimize the total service costs of all ECs. We define new variables $\{u_{nv}, z_{nv}\} \in \{0,1\}$ to model possible video request redirection when a request for video v arrives at home EC n. $u_{nv} = 1$ denotes the requested video is retrieved from neighbor ECs; and $u_{nv} = 0$ otherwise; $z_{nv} = 1$ denotes the requested video is retrieved from the CS; and $z_{nv} = 0$ otherwise. The problem can be formulated as follows:

$$\boldsymbol{P_4}: \min \sum_n \sum_v \lambda_{nv}(x_{nv} H_v + u_{nv} N_v + z_{nv} R_v)$$
$$s.t. \quad u_{nv} \le \sum_{n' \ne n} x_{n'v}, \quad \forall n, \forall n', \forall v, \tag{22a}$$
$$x_{nv} + u_{nv} + z_{nv} = 1, \quad \forall n, \forall v, \tag{22b}$$
$$\sum_v x_{nv} W_v \le S_n, \quad \forall n, \forall v, \tag{22c}$$
$$x_{nv}, u_{nv}, z_{nv} \in \{0,1\}, \quad \forall n, \forall v. \tag{22d}$$

In the above formulation, constraint (22a) ensures availability of the video request redirection variable; constraint (22b) ensures there is only one response event for the requested video; constraint (22c) means the cache capacity limitation of EC; formula (22d) indicates the binary constraint of variables. To solve the above problem, we design the GVCP algorithm as shown in Alg. 2 and its time complexity is $\boldsymbol{N^2 \times V^2}$. In practice, the number of edge clouds need to restrict according to geographical location (e.g. the top 10 edge clouds with the smallest distance). The binary variables $\{E_n^s, V_v^s\} \in \{0,1\}$ are used to record caching state, where $E_n^s = 1$ denotes the cache space of EC n run out; and $E_n^s = 0$ otherwise; $V_v^s = 1$ denotes that the video v has been cached in some EC; and $V_v^s = 0$ otherwise. Besides, the symbol rem_n denotes remaining storage space of EC n.

Algorithm 2 GVCP

```
Input: $H_v, W_v, N_v, R_v, \lambda_{nv}, S_n$
Output: $x_{nv}$
 1: for Each EC $n \in \mathcal{N}$ do
 2:     for Each video $v \in \mathcal{V}$ do
 3:         $x_{nv} \leftarrow 0,\ E_n^s \leftarrow 0,\ V_v^s \leftarrow 0,$
 4:         $rem_n \leftarrow S_n,\ h_v \leftarrow H_v/W_v,\ o_v \leftarrow N_v/W_v$
 5:         $r_v \leftarrow R_v/W_v,\ \Delta_{nv} \leftarrow \lambda_{nv}(r_v - h_v)$
 6: while (True) do
 7:     $\Delta_{ij} \leftarrow \max\{\Delta_{nv} | E_n^s = 0,\ x_{nv} = 0,\ rem_n \geq W_v,\ v \in \mathcal{V},\ n \in \mathcal{N}\}$
 8:     if $\Delta_{ij} \leq 0$ then
 9:         Break
10:     $x_{ij} \leftarrow 1,\ rem_i \leftarrow rem_i - W_j,\ \Delta_{ij} \leftarrow 0$
11:     $W_k \leftarrow \min\{W_v | x_{iv} = 0, v \in \mathcal{V}\}$
12:     if $W_k > rem_i$ then
13:         $E_i^s \leftarrow 1$
14:     if $V_j^s = 0$ then
15:         $V_j^s \leftarrow 1$
16:         for Each EC $n' \in \mathcal{N}$ do
17:             if $n' \neq i$ then
18:                 $\Delta_{n'j} \leftarrow \lambda_{n'j}(\min\{o_j, r_j\} - h_j)$
19:     if $\min\{E_n^s | n \in \mathcal{N}\} = 1$ then
20:         Break
```

V. Performance Evaluation

In this section, we evaluate the performance of proposed solution for the collaborated optimization problem. First, we briefly describe all compared schemes and give the simulation setup. Next, because the transcoding cost is unknown in advance at 360° video services, we adopt a machine learning method to predict it. Finally, we show and analyze the impact of various system parameters for the performance of all compared schemes.

A. Experiment Setup

All compared schemes are divided into two groups. In the first group, we consider four different video *caching schemes* in related works: (1) **FoVEC**, the FoV-Aware edge caching scheme proposed by [11], which is deployed in a single edge facility based on video popularity and viewing probability of tile; (2) **FoVEC+**, the scheme expanded by FoVEC, where the edge facilities can respond to tile requests for each other; (3) **CJCT**, the collaborative joint caching and transcoding scheme proposed by [17]; (4) **B-LH**, the universal scheme, which caches both high and low quality versions of each video by proposing GVCP algorithm. In the second group, we compare the impact of each part of our proposed scheme. There are three *incomplete schemes*: (1) **No-TH**, the scheme omits request scheduling events that are transcoding response and high-quality version directly respond to the tile within margin area; (2) **No-VAS**, the scheme omits viewport adaptive streaming, where all requested tiles are severed with high-quality version; (3) **No-caching**, the scheme omits edge caching architecture.

Unless otherwise stated, we consider an edge cloud network with $N_n = 3$ ECs. Based on the pricing of Alibaba's edge service, we set the unit delivery cost $\mu_{ce} = 2.5$ (Unit/MB), $\mu_{ee} = 0.35$ (Unit/MB), $\mu_{eu} = 0.7$ (Unit/MB), the unit caching cost $\alpha = 0.00001167$ (Unit/MB), and the unit transcoding cost $\beta = 2.11$ (Unit/GHz) for each EC. We choose 70 panoramic videos (4k resolution) from the dataset provided by [20] as the video library. These videos exhibit a high diversity in terms of contents (e.g. indoor scene, outdoor activities, sports games and so on). The duration of each video ranges from 20 to 60 seconds and the size ranges from 10 to 90 MB. Besides, the dataset provides user's head movement traces, and each video is viewed by 32 users at least. It is easy to obtain the real user's FoV. In our experiments, each video is recoded and divided into segments with 2 seconds, and each segment is divided into 24 tiles, 6 columns and 4 rows, as described in [11]. The original resolution of each tile is 640 x 540 pixels, which is considered as the high-quality version. We transcoded each tile to the resolution, 270 x 180 pixels, with FFmpeg H.264 encoder, which is considered as low-quality version. Then, we achieve the viewing probability of tile based on those historical FoV traces of users. The video popularity is generated following a Zipf distribution with the skewness parameter $\kappa = 0.8$ (i.e. the popularity of video v is given as $\lambda_v = \frac{1/v^\kappa}{\sum_{i=1}^{V} 1/i^\kappa}$) [17]. For satisfying the scenario that the same video has different popularity in different locations, we randomly shuffle the video popularity distributions at different regions as described in [19].

B. Transcoding Prediction

Transcoding is a computationally sensitive operation, and its cost is mainly due to the computing resource consumption. The product of the transcoding time, the mean CPU utilization and the clock frequency is used as consumed computing resources. We choose Catboost decision trees to predict transcoding cost. We transcoded each high-quality tile into low-quality at Alibaba's Elastic Compute Service (1 vCPU and 2 GiB memory) and obtained 30984 instances about computing resource consumption. The 80% instances are selected for training and the other 20% for testing. The transcoding operation is related to the bitrate, duration, contents and encoding [21]. Thus, the input feature vector of our prediction model consists of bitrate, FPS, the number of I frames, the total size of I frames, the average size of I frames, the number of P frames, the total size of P frames and the average size

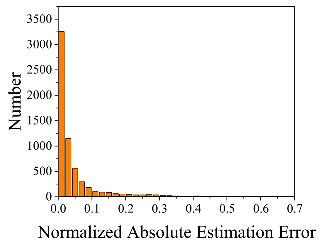


Fig. 4. Transcoding prediction.

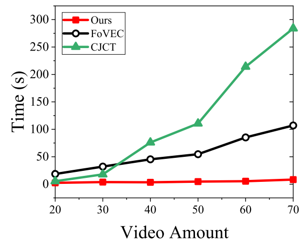
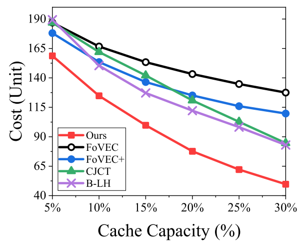


Fig. 5. Executed time of algorithms.


(a) Results of caching schemes.

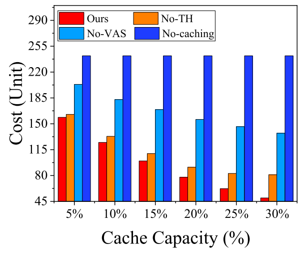
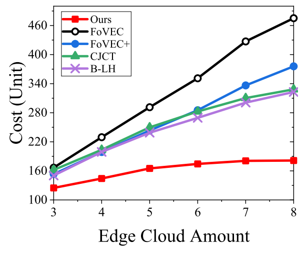

(b) Results of incomplete schemes.

Fig. 6. Performance comparison of caching and incomplete schemes when increasing relative cache capacity of each EC; $\kappa = 0.8$.


(a) Results of caching schemes.

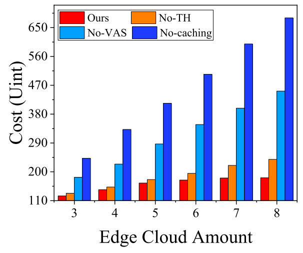

(b) Results of incomplete schemes.

Fig. 7. Performance comparison of caching and incomplete schemes when increasing EC amount; $S_n = 10\%$[Lib.Size], $\kappa = 0.8$.

of P frames. We used mean absolute error as loss function, and five-fold cross-validation method to train. The normalized absolute estimation error φ is used as evaluation metric:

$$\varphi = \frac{|EstimatedValue - RealValue|}{RealValue}$$

The average values of normalized absolute evaluation error of training and testing instances are 4.4% and 4.7%. The normalized absolute error histogram of test instances is illustrated in Fig. 4, where the error mainly ranges from 0 to 0.1. It manifests that our prediction model can accurately estimate computing resource consumption of transcoding operation.

C. Performance Analysis

In this section, we evaluate the performance of proposed joint optimization scheme under variable cache capacities, the EC amount, the video amount and the skewness parameters of video popularity. In our experiments, we consider the total service costs of ECs as the performance evaluation metric.

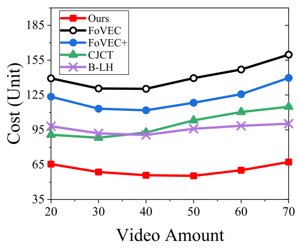
1) Executed time of algorithms: We examine the executed time of Ours, FoVEC and CJCT schemes at different number of videos as shown in Fig. 5. The cache capacity of each EC is the 25% total size of current video library. When increasing the number of videos, the executed time of Ours scheme slowly increase due to the proposed low-complexity efficient algorithm. However, the executed time of FoVEC and CJCT schemes rapidly increase along with the growth of video amount, which is because that these schemes make too many comparison operations and do not consider the integrity and consistency of video caching.

2) Impact of cache capacity: To compare the impact of cache capacity, we evaluate the performance of considered schemes at different relative cache size, given by the ratio between the cache capacity of EC and the total size of video library. The results are shown in Fig. 6(a) for the case of caching schemes and in Fig. 6(b) for the case of incomplete schemes. From these figures, we can see that the growth of cache size results in performance improvement for all compared schemes except No-caching scheme. This is because that each EC can store lots of video content at the high cache capacity. Meanwhile, the rise of cache size results in a growing gap between Ours and other considered schemes. This means Ours scheme maintains more competitive performance than other schemes with increasing cache capacity. Notice that the results of FoVEC and FoVEC+ schemes indicate that the collaboration optimization of multiple edge facilities outperforms the single one.

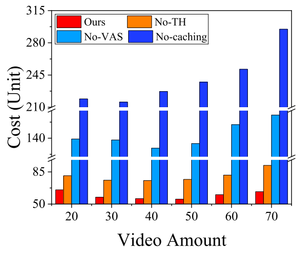
3) Impact of edge cloud amount: We evaluate the performance of all considered schemes at varying number of ECs in Fig. 7(a) for caching schemes and in Fig. 7(b) for incomplete schemes. The service costs of all schemes grow along with the extension of EC amount, which is because of growing service regions. Notice that Ours scheme always outperforms other compared schemes and its growth trend is gradually flattening. In Fig. 7(a), it is observed that there is a growing gap between Ours scheme and other caching schemes. Interestingly, FoVEC+, CJCT and B-LH schemes cause approximate results. In Fig. 7(b), the gap between No-TH and Ours schemes slowly increases at the small and moderate amount of ECs and rapidly increases at the high number. This is because of the growing influence of collaborated optimization of multiple ECs. On the other hand, the No-VAS and No-caching schemes always keep rapid growth and a large space with Ours scheme.

4) Impact of video amount: The impact of video amount is presented in Fig. 8(a) for caching schemes and in Fig. 8(b) for incomplete schemes. The cache capacity of each EC is the 25% total size of current video library. For clarity, we omit tedious results about No-VAS and No-caching schemes, but highlight the changing trend in Fig. 8(b). These figures illustrate the video service costs of all schemes first decrease and then increase when increasing the number of videos. The reason is that there is a trade-off between the increasingly fragmented video popularity and growing video requests. We can see that Ours scheme always maintains a significant advantage and a big gap compared with other considered schemes. Besides, there is always a relatively big difference between different caching schemes in Fig. 8(a).

5) Impact of the skewness parameter: We examine the impact of variable skewness parameters of Zipf distribution for all considered schemes. The larger skewness parameter

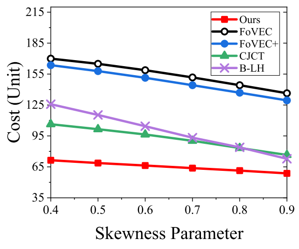


(a) Results of caching schemes.

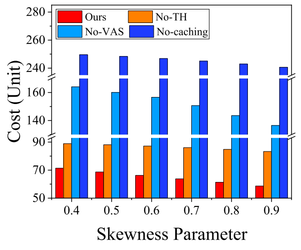


(b) Results of incomplete schemes.

Fig. 8. Performance comparison of caching and incomplete schemes when increasing video amount; $S_n = 25\%$[CurrentLib.Size], $\kappa = 0.8$.



(a) Results of caching schemes.



(b) Results of incomplete schemes.

Fig. 9. Performance comparison of caching and incomplete schemes when increasing the skewness parameter; $S_n = 25\%$[Lib.Size].

means more similar user preference. For each change of the skewness parameter, we generate video popularity in the same randomized way. The results are illustrated in Fig. 9(a) for the case of caching schemes and in Fig. 9(b) for the case of incomplete schemes. Similarly, we omit tedious results of incomplete schemes. Notice that Ours scheme always keeps lower service costs and a large gap with all compared schemes when increasing the skewness parameter. In Fig. 9(a), the growing skewness parameter always results in enhanced performance for all caching schemes and a reduced gap between Ours scheme and others. This is because that when increasing the skewness parameter, the video popularity gradually focus on a few amounts of videos and all caching schemes tend to cache these high popular videos. In Fig. 9(b), the big difference between Ours and other incomplete schemes denotes each component of Ours scheme plays a huge role.

VI. Conclusion

In this paper, we propose the tile-based joint transcoding, delivery and caching of 360° videos framework *Allies* in edge cloud networks. Specifically, we present an innovative video caching idea and diverse request scheduling ways to improve the utility of edge cloud. Then, the collaborative optimization problem is formulated to minimize the video service costs of multiple edge clouds. To deal with the collaborated problem, we propose a greedy suboptimal solution with polynomial running time. Empirical evaluations with real-world user's head movement traces and corresponding 360° videos demonstrate the superior performance of our proposed algorithm over state-of-the-art works.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (No.61702287), the Natural Science Foundation of Tianjin (No.18JCQNJC00200, No.18ZXZNGX00140), the Young Elite Scientist Sponsorship Program by the Tianjin Association for Science and Technology (No. TJSQNTJ-2018-19), the Technology Research and Development Program of Tianjin (No.18ZXZNGX00200) and the Open Project Fund of State Key Laboratory of Integrated Services Networks (No. ISN2013).

References

[1] G. Forecast, "Cisco visual networking index: Global mobile data traffic forecast update 2017–2022," *Update*, vol. 2017, p. 2022, 2019.

[2] X. Corbillon, G. Simon, A. Devlic, and J. Chakareski, "Viewport-adaptive navigable 360-degree video delivery," in *IEEE Int. Conf. Commun. (ICC)*, pp. 1–7, 2017.

[3] Y. He, X. Xiu, and Y. Ye, "360lib software manual," *Joint Video Exploration Team (JVET) of ITU-T SG*, vol. 16, 2017.

[4] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming," in *ACM Int. Conf. Multimedia (MM)*, pp. 315–323, 2017.

[5] L. Pu, L. Jiao, X. Chen, L. Wang, Q. Xie, and J. Xu, "Online resource allocation, content placement and request routing for cost-efficient edge caching in cloud radio access networks," *IEEE J. Sel. Areas Commun. (JSAC)*, vol. 36, no. 8, pp. 1751–1767, 2018.

[6] J. Yao, T. Han, and N. Ansari, "On mobile edge caching," *IEEE Commun. Surv. Tutor.*, vol. 36, no. 8, pp. 2525 – 2553, 2019.

[7] C. Li, L. Toni, J. Zou, H. Xiong, and P. Frossard, "Qoe-driven mobile edge caching placement for adaptive video streaming," *IEEE Trans. Multimedia (TMM)*, vol. 21, no. 3, pp. 965–984, 2017.

[8] F. Qian, B. Han, Q. Xiao, and V. Gopalakrishnan, "Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices," in *ACM Int. Conf. Mobile Comput. Netw. (MobiCom)*, pp. 99–114, 2018.

[9] S. Petrangeli, V. Swaminathan, M. Hosseini, and F. De Turck, "An http/2-based adaptive streaming framework for 360 virtual reality videos," in *ACM Int. Conf. Multimedia (MM)*, pp. 306–314, 2017.

[10] G. Papaioannou and I. Koutsopoulos, "Tile-based caching optimization for 360° videos," in *20th ACM Int. Symp. Mobile Ad Hoc Netw. Comput. (MobiHoc)*, pp. 171–180, 2019.

[11] A. Mahzari, A. Taghavi Nasrabadi, A. Samiei, and R. Prakash, "Fov-aware edge caching for adaptive 360 video streaming," in *ACM Int. Conf. Multimedia (MM)*, pp. 173–181, 2018.

[12] K. Bilal and A. Erbad, "Edge computing for interactive media and video streaming," in *IEEE Int. Conf. Fog Mobile Edge Comput. (FMEC)*, pp. 68–73, 2017.

[13] J. Chakareski, "VR/AR immersive communication: Caching, edge computing, and transmission trade-offs," in *ACM Workshop on Virtual Reality and Augmented Reality Network*, pp. 36–41, 2017.

[14] Y. Sun, Z. Chen, M. Tao, and H. Liu, "Communications, caching and computing for mobile virtual reality: modeling and tradeoff," *IEEE Trans. Commun. (TCOM)*, vol. 67, no. 11, pp. 7573–7586, 2019.

[15] T. Dang and M. Peng, "Joint radio communication, caching and computing design for mobile virtual reality delivery in fog radio access networks," *IEEE J. Sel. Areas Commun. (JSAC)*, vol. 37, no. 7, pp. 1594 – 1607, 2019.

[16] A. Ndikumana, N. H. Tran, T. M. Ho, Z. Han, W. Saad, D. Niyato, and C. S. Hong, "Joint communication, computation, caching, and control in big data multi-access edge computing," *IEEE Trans. Mobile Comput. (TMC)*, 2019.

[17] K. Bilal, E. Baccour, A. Erbad, A. Mohamed, and M. Guizani, "Collaborative joint caching and transcoding in mobile edge networks,"

[18] D. Liu and C. Yang, "Caching at base stations with heterogeneous user demands and spatial locality," *IEEE Trans. Commun. (TCOM)*, vol. 67, no. 2, pp. 1554–1569, 2018.

[19] T. Tran and D. Pompili, "Adaptive bitrate video caching and processing in mobile-edge computing networks," *IEEE Trans. Mobile Comput. (TMC)*, vol. 18, no. 9, pp. 1965–1978, 2018.

[20] Y. Xu, Y. Dong, J. Wu, Z. Sun, Z. Shi, J. Yu, and S. Gao, "Gaze prediction in dynamic 360 immersive videos," in *IEEE Int. Conf. Comput. Vision Pattern Recognition (CVPR)*, pp. 5333–5342, 2018.

[21] H. Zhao, Q. Zheng, W. Zhang, and J. Wang, "Prediction-based and locality-aware task scheduling for parallelizing video transcoding over heterogeneous mapreduce cluster," *IEEE Trans. Circuits Syst. Video Technol. (TCSVT)*, vol. 28, no. 4, pp. 1009–1020, 2016.