# Collaborative Social-Aware and QoE-Driven Video Caching and Adaptation in Edge Network

Yao Chiang, *Student Member, IEEE,* Chih-Ho Hsu and Hung-Yu Wei, *Senior Member, IEEE*

***Abstract*—With the emerging demand for high-definition videos in recent years, Multi-access Edge Computing (MEC) has become a promising solution to leverage Quality of Experience (QoE) of users in the 5G mobile network, which provides computing and cache resource at network edges to serve end users with less latency. Also, since mobile users tend to be influenced by the trends in social media, the performance of video caching will become more effective if we can extract the hidden information from interaction among them. In this paper, we propose a novel Collaborative Social-aware QoE-driven video Caching and Adaptation (CSQCA) framework. Specifically, we first design a 2-tier MEC collaborative video caching architecture, which partially caches popular videos among multiple edge servers. Second, we propose a social-aware proactive cache strategy, which embeds interactions of users and video dissemination process in social networks into the caching mechanism. Third, a QoE-driven video adaptation algorithm is presented to dynamically transcode the cached videos into appropriate resolution on edge server for each request. Finally, we conduct our simulation based on real-world datasets. The simulation results show that the proposed CSQCA framework outperforms traditional cache algorithms, in terms of the average hit ratio and QoE.**

***Index Terms*—multi-access edge computing, collaborative edge caching, quality of experience, social network, content diffusion, video adaptation.**

## I. Introduction

With the proliferation of mobile devices, network traffic has grown exponentially in recent years, most of which stem from video streaming services [1]. However, since most of the videos are stored in remote cloud servers, the generated traffic will exert a heavy load on the backhaul network, ending up degrading users' QoE. Facing this, Multi-access Edge Computing (MEC), which provide computing and cache resource at network edge closed to users [2], has become a promising paradigm to alleviate data traffic of backhaul network and leverage user's QoE. In this way, the delivery delay of video and the incidence of backhaul network congestion can be significantly reduced.

Nonetheless, due to the large size of high-definition video and the limited storage capacity of edge nodes compared with cloud content servers, resource utilization is a critical problem for edge caching schemes. Thus, the partial caching schemes, either only caching some segment of a video [3]-[5] or few resolution versions of a video [5]-[7], is implemented in many of MEC designs since they can cache more video on edge nodes while ensuring user's QoE. Another potential solution is the collaborative caching scheme [8]-[10], which aims to promote the overall cache hit ratio by utilizing scarce cache capacity among distributed edge nodes. On the other hand, cache lifetime, the time duration of caching a video, is another imperative issue to tackle for the edge caching scheme [11]. Since the demand for the same video may vary with time and caching a video occupies the limited storage space, we can better utilize the storage resource of edge nodes by removing the cached videos from cache space when their popularity decreases. However, how to obtain the optimal cache lifetime for videos is still an open but critical issue in MEC architecture.

From another perspective, since the popularity of contents highly depends on the trends in the social community [12]-[13], the performance of edge caching would become more effective if we can utilize social influence in cache mechanisms. The metrics for measuring social influence can be preliminary divided into two aspects: the interest of users [14]-[15] and interaction between users [16]-[17]. The pros and cons of these two metrics are that though the interest-based methods can achieve high accuracy in predicting the popularity of the contents, these approaches require adequate data of users such as viewing history and rating toward different types of content, which may not be viable for Internet Service Providers (ISP) who only have limited information about online users in contrast to the application vendors (e.g. Youtube). On the other hand, although interaction-based methods don't need a tremendous amount of training data, they may not be as accurate as interest-based methods under some specific conditions because they are only aware of rudimentary information of users without intricate features. In order to give more insights into the relevance between the interaction of users and the popularity of contents, the content dissemination process among edge users becomes an emerging metric [18]-[19], which extrapolates the potential demand of content in the edge area through the interaction of users, ending up improving the performance of the edge cache system.

However, since the social behavior of users in real-world can be affected by numerous factors such as personality, mood or even weather, it can hardly be simulated with existing mathematical models. Thus, adopting real world data to verify these social-aware edge caching method is especially important, which can reduce the gap between theory and reality. On the other hand, the existing social-aware caching method usually adopts undirected graph to model their social network while the interaction of users in real-world is bidirectional and varies with time. Thus, how to accurately model the social relationship of users in real-world is also a key but has not been widely discussed issue.

Though the above studies have made efforts to the social-aware content cache method, how to utilize social influence in the video cache method is still not clear, which is even more

imperative since the majority of Internet traffic arises from video service. Compared to file downloading service, video service has more complex features. For instance, when caching a video file, we need to determine how many chunks to cache and their resolution version. To address this, the transcoding technique has been widely discussed in the video cache related issues [3],[6]-[7],[20]-[21]. This enables transforming a higher resolution version video cached in edge nodes into a lower resolution version to meet the downlink capacity of users. With the aids of transcoding, we no longer need to cache various resolution versions of the same video to satisfy different video resolution requests, enhancing the resource utilization of edge nodes.

Motivated by the above work as well as the challenges, we proposed a Collaborative Social-aware QoE-driven video Caching and Adaption (CSQCA) framework to maximize QoE of streaming service of edge communities. Specifically, our main contributions are summarized as follows:

*1*) To analyze the relevance between social behavior of users and the popularity of videos, we model the social network with a dynamic bi-direct graph, predicting the time-varying social relationships of users with an Auto Regressive Moving Average (ARMA) model to make it more realistic and statistically accurate.

*2*) Following the ETSI standards, we design a 2-tier collaborative MEC cache architecture, jointly optimizing video cache and video adaptation problem. Specifically, through orchestrating proactive caching decision of edge nodes and performing real-time video transcoding on each edge node, our 2-tier system can maximize users' QoE in whole region.

*3*) We propose a novel social-aware video caching algorithm, which embeds social information of users and video dissemination in community into its mechanism. The proposed algorithm caches the videos with an appropriate cache ratio and lifetime to further enhance the resources utilization of the edge nodes.

*4*) A QoE-driven video adaptation algorithm is proposed in our edge system, which optimizes video resolution decisions given users' real-time channel conditions and capacity of backhaul network. Hence, the edge servers can transcode the cached video into suitable resolutions that both maximize user's instant QoS and QoE in long term.

*5*) We conduct a series of experiments with real-world traces to further validate the performance of the proposed framework. By comparing with the several existing cache method, the result illustrates the proposed method outperforms other compared solutions in terms of hit ratio and QoE.

*6*) To our best knowledge, this is the first attempt to propose a 2-tier MEC cache solution that integrates dynamic social influence, video dissemination process among users and video transcoding technique in its collaborative cache decisions, including cache ratio and lifetime, while further utilizing real data to simulate the time-varying behavior and video dissemination process of users.

The rest of this work is presented as the following: Section II and Section III discuss the related works and our system model respectively. Then, our problem formulation is organized in Section IV. Section V elaborates on the proposed CSQCA framework, involving the proposed social-aware cache mechanism and QoE-driven video adaptation method. Finally, Section VI evaluates CSQCA with extensive simulations and Section VII closes this work with conclusions.

## II. Related Work

Different approaches for various objectives have been investigated for implementing video caching and adaptation in MEC framework. In this section, we outline some of the most related works based on MEC and 5G system, social-aware cache and cache-enabled video adaptation mechanisms.

### A. Multi-access Edge Computing and 5G system

MEC system, proposed by European Telecommunications Standards Institute (ETSI) [2], is a brand-new paradigm to offers personalized services for mobile users at network edges by overcoming the drawbacks of cloud computing that incur large latency due to the long distance from end devices to remote servers. With an anticipated large number of innovative applications in the 5G network, various techniques have been investigated in the MEC framework. To begin with, the potential techniques for content caching and delivery in 5G mobile edge networks are discussed in [22]. Also, as radio resources and computation resources in the MEC network can be further exploited for executing mobile users' computational tasks, the authors in [23]-[24] discussed task offloading decision and the corresponding resource allocation problem aiming to reduce latency and energy consumption of users.

However, one of the challenges to reach the potential of MEC is the migration and placement of the dynamic services under the mobility of users. Thus, in [25], they jointly optimize the access network selection and service placement problem to promote the QoS in terms of system delay. Besides, to satisfy the heterogeneous service requirements in the 5G network, the concept of network slicing, where each network slice can serve as a dedicated network for a customized service with an aid of network virtualization techniques, has emerged recently. In [26], they propose a general 5G network slice framework to tackle the Virtual Network Functions (VNF) placement problem, with a goal to optimize the total throughput of the accepted VNFs. Complementarily, the paper [27] proposes a novel algorithm to embed each Virtual Network Request (VNR) onto corresponding infrastructures in the network slice.

### B. Social-Aware Cache

The social influence could heavily affect the request pattern [12]-[13] and thus may ultimately affect the performance of edge caching. To illustrate, the authors in [12] analyze video traffic in the campus network. Their results showed that video popularity in local areas can vary significantly from the global popularity of the video, implying that social community has impacts on regional video request patterns. Further, the influence of social interaction on video popularity is studied in [13]. By analyzing a large set of Facebook users and their public video-related interaction (e.g. share, posts and comments), they show that, in general, the popularity of video increases with interaction among users.

TABLE I
A COMPARISON OF RELATED WORK IN THE LITERATURE

| # | Social relationship | Social network | Architecture | Cache scheme | Content type | Cache lifetime | Partial cache | Dynamic transcoding | Real data |
|---|---|---|---|---|---|---|---|---|---|
| 14 | IE | Undirect | Heterog. | D2D-centralized | Unspecified | X | X | X | X |
| 15 | IE | Undirect | Cloud | Centralized | Unspecified | X | X | X | O |
| 16 | IC | Undirect | Heterog. | D2D-centralized | Unspecified | X | X | X | X |
| 17 | IE+IC | Undirect | Heterog. | D2D-distributed | Unspecified | X | X | X | X |
| 18 | AK | Undirect | Heterog. | Distributed | Unspecified | X | X | X | X |
| 19 | AK | Undirect | Edge | Centralized | Unspecified | X | X | X | X |
| 11 | AK | Undirect | Heterog. | D2D-distributed | Unspecified | O | X | X | X |
| 3 | X | X | Cloud | Centralized | Video | X | O | O | X |
| 4 | X | X | 2-tier edge | Collaborative | Video | X | O | X | X |
| 5 | X | X | Edge | Distributed | Video | X | O | X | X |
| 6 | X | X | Edge | Distributed | Video | X | O | X | X |
| 7 | X | X | Cloud | Centralized | Video | X | X | O | X |
| 20 | X | X | Edge | Distributed | Video | X | X | O | X |
| 21 | X | X | Edge | Distributed | Video | X | X | O | X |
| 28 | X | X | Edge | Distributed | Video | X | X | X | O |
| 29 | X | X | Edge | Distributed | Video | X | X | X | X |
| We | T-V IC | T-V bi-directed | 2-tier edge | Collaborative | Video | O | O | O | O |

**Hints**: **AK** = Assumed Known **IE** = Interest-based **IC** = Interaction-based **T-V** = Time-Varying

Therefore, there are several works considering social influence in their cache schemes to provide a better service. The interest of users is utilized to determine their social relationships in [14]-[15]. Specifically, they adopt cooperative filtering to predict the probability that one user would be interested in certain content according to the history preference of similar users. Then, they select top $k$ popular content among the similar users as their cache decision. On the other hand, the contact history of users is adopted to determine the social relationship in [16]-[17]. The authors consider a Device-to-Device (D2D) cache scenario where users can download contents from their partner. Then, by assuming that the better relationship between the users, the higher probability of content sharing between them, social relationship are utilize in determining their cache placement in [16]-[17].

Besides, since content sharing among users is influential to the popularity of contents in the community [13], some previous works have analyzed the effect of content dissemination in the social network [11], [18]-[19]. The Markov chain based approach is proposed in [18]-[19] to predict the viewing probability of certain contents under the content dissemination model. Finally, the optimal cache lifetime for content dissemination in social community is investigated in [10]. However, the dissemination probability are all assumed as prior information to the system in the above works, neglecting the time-varying relationship of users.

### C. Cache-enabled Video Adaptation

There has been a series of research work focusing on cache-enabled adaptive video streaming [3]-[7], [20]-[21], [28]-[29], where video adaptation is made based on the caching result of edge nodes. Specifically, QoE-driven cache schemes for Dynamic Adaptive Video Streaming over HTTP (DASH) is studied in [5],[7],[28]. In [7], they aimed to maximize its QoE by finding the optimized number of video segments to be cached. On the other hand, The authors introduce a QoE-driven cache algorithm based on both segment popularity and network conditions in [28]. Though a comprehensive QoE model and novel collaborative edge caching are designed in [5], the varying channel condition of users is assumed to be known based on their bitrate allocation history. Moreover, the cache scheme for transcoding enabled Adaptive Bit Rate (ABR) streaming is considered in [3],[20]-[21]. To begin with, the RAN-aware proactive cache policy that utilizes the preference of active users in a cell is proposed in [3]. The authors of [20] propose the MEC-ABR video delivery scheme to improve streaming service by jointly optimizing cache and radio resource allocation. Then, in [21], they propose a MEC framework that supports ABR with the goal to minimize the expected delay by determining cache placement of video and the corresponding resolution.

Besides, the aim of [4] is to maximize the average QoE of Video on Demand (VoD) service by optimizing the proactive edge caching policies including the cached fraction and encoding bit rate of every video. Nonetheless, they assume video quality versions to be continuous, which is a relaxation of the practical finite quality versions. In [6], they study a QoE-driven mobile edge cache placement optimization for dynamic adaptive video streaming that properly takes into account the different rate-distortion characteristics of videos and the coordination among distributed edge servers. Finally, a novel mechanism is designed in [29] to jointly optimize bandwidth and caching strategies in software-defined wireless network. However, all of the mentioned works assume the probability of users requesting certain videos is known to their system, which is less practical in the real world.

A comparison of related work in the literature is listed in Table I. To compare the features with each other, we classify them according to consideration of social influence, system architecture, cache method, type of cached content, partial cache and transcoding, and using real-world data or not. We observe that so far there is no paper simultaneously considers: 1) 2-tier MEC cache architecture, 2) time-varying social relationship of users, 3) partial cache scheme 4) video transcoding technique 5) cache lifetime and 6) real-world data. Therefore, we start to draw a design and implement this novel architecture based on these 6 features.

## III. System Model

In this paper, we consider a 2-tier MEC architecture to partially cache popular videos on edge nodes for efficiently serving the requests generated by different social communities in the region. In order to meet the service scenario in real world, two different time scales are considered in our system. Specifically, the scale of the system time slot (i.e. $k$) refers to hours while the scale of the system instant (i.e. $t$) is seconds. As shown in Fig. 1, the system is composed of four major components: social communities, edge servers, regional server, and video content server. The functionalities of each component are described as follow:

*1) Social Community:* There are multiple communities within the whole region. We denote the set of users within the coverage of the edge server in community $m$ at time slot $k$ by $U_m^{(k)} = \{ID(1), ID(2), ..., ID(u_m^{(k)})\}$ where $u_m^{(k)}$ is the number of associated users in community $m$ and $ID(i)$ is the identity number of the user with index $i$ in the community. In our scenario, the users within the same community have similar hobbies and may have connections with each other due to the geopolitical relationship (e.g. classmates in school or colleagues in office). In this way, users may share videos with their friends when interacting on the Internet (e.g. Facebook, TikTok, or even Email). Moreover, due to users' mobility, the set of users in each community varies with time. Thus, each user could keep in touch with more than one community and the connection of users may also be dynamic.

*2) Edge Server:* The edge servers in our 2-tier MEC architecture are responsible for: 1) serving the collected video requests from social communities by providing video cache and video transcoding service and 2) recording the viewing history and social information of each user. Thus, each edge server $m$ is equipped with cache space with size $D_m^{eg}$, computing capacity $p$ for video transcoding and transmission power $PW$ for video delivery. There is a set of edge servers in the region, denoted by $M = \{1, 2.....,h\}$, and each of them is located at facilities corresponding to different communities such as a classroom in campus and a department building so that it can deliver videos to users with less transmission latency. Accordingly, we use the terms edge server $m$ and community $m$ interchangeably in this paper.

*3) Regional Server:* The regional server is the orchestrator in our architecture assumed to has all information about edge users. It is in charge of managing the cache list of each edge server to realize collaborative cache in the region. Also, the regional server reports the available throughput of the backhaul network to the edge server for real-time video scheduling.

*4) Video Contents Server:* We consider that each video can be divided into consecutive chunks with a fixed time duration $\Delta t$ and there is a library of videos stored in content server, indicated by $F = \{1, 2, ..., L\}$ with $\{n_1, n_2 \ldots, n_L\}$, where $n_f$ indicates the number of chunks of entire video $f$. Furthermore, each video chunk has multiple resolution levels, denoted by $\chi = \{\chi_1, \chi_2, ..., \chi^*\}$ where $\chi_i$ is the encoding bitrate of $i^{th}$ resolution level and $\chi^*$ is the highest resolution. Then, we can express the size of a video chunk with resolution $\chi_i$ as $v(\chi_i)$.

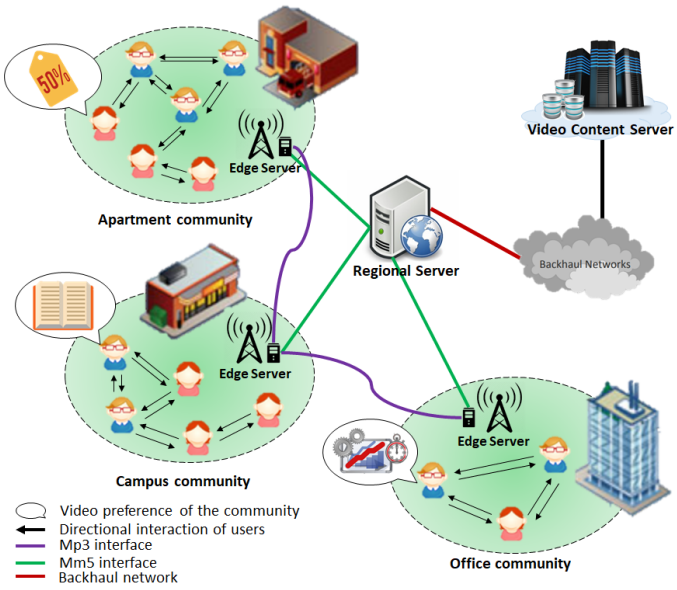



Fig. 1. The proposed 2-tier MEC video cache architecture

The details of each component will be discussed comprehensively in the following subsections. Also, the main notations involved in our system are summarized in Table II.

### A. Interaction Model

Since social interaction between users is influential to video request patterns in a community [13], in this subsection, we focus on modeling the interaction of users to leverage performances of our proposed system. It is worth noting that the interaction discussed in this work refers to all kinds of interaction, including e-mail, chat message, or reaction to posts on social media. This consideration is reasonable because, in general, the correlation between all types of interactions and video-related interactions is statistically significant [13].

To begin with, we introduce an indicator $s_{i,j}(t) \in \{0, 1\}$ to denote interaction events between two users. That is, $s_{i,j}(t) = 1$ if user $i$ interacts with user $j$ at instant $t$ and $s_{i,j}(t) = 0$ otherwise. Note that in order to distinguish the direction of interaction, $s_{i,j}(t)$ and $s_{j,i}(t)$ are regarded as different. Then, we define the interaction intensity variable to describe the relationship of users, which is the number of interaction event during a time slot. For two users $i, j$ whose $ID(i), ID(j) \in U_m^{(k)}$, the interaction intensity from $i$ to $j$ during time slot $k$ can be calculated by:

$$S_{i,j}^{(k)} = \frac{1}{T} \int_{(k-1)T}^{kT} s_{i,j}(t) dt \tag{1}$$

where $T$ is the duration of a time slot.

Furthermore, we consider a more realistic scenario where the relationship between users may vary with time. For instance, classmates may reduce their contact frequency after graduation. In this work, we adopt the ARMA model, a statistics-based method widely used in time series prediction [30], to fit the time-varying interaction pattern of users. In the ARMA model, the future value of a variable is a linear combination of past values and past errors [31]. Thus, given

the historical value of interaction intensity $S_{i,j}^{(k)}$, the predicted interaction intensity at present $\hat{S}_{i,j}$ can be obtained by:

$$\hat{S}_{i,j} = \sum_{n=1}^{q} \varphi_n S_{i,j}^{(k-n)} + \sum_{n=1}^{q} \psi_n \zeta_{i,j}^{(k-n)} \tag{2}$$

In this equation, $\varphi_n$ and $\psi_n$ are the weight coefficients of ARMA model, $k$ signifies the present time slot and $q$ is the order of the model. Also, $\zeta_{i,j}^{(k-n)}$ is the prediction error of $\hat{S}_{i,j}$ in the previous $n$ time slot.

Nevertheless, the value defined above would be affected by different usage habits of users. For example, the value of (1) would be high for a user who keeps sending messages to others even if they are actually not familiar with each other. On the other hand, we can't tell the relationship between two users who aren't tight if they don't frequently interact on social media. Instead, to eliminate the effect of the different usage patterns of users, we turn to normalize the interaction intensity of each user. Assuming that the more intense a user interacts with a person on the Internet, the more important that person is to the user. Thus, we define the importance of user $j$ toward the user $i$ in community $m$ at present as:

$$Im_{i,j} = \frac{\hat{S}_{i,j}}{\max\left\{\hat{S}_{i,u} | u \neq i, ID(u) \in U_m^{(k)}\right\}} \tag{3}$$

which is the value of interaction intensity from $i$ to $j$ divided by the maximum value of interaction intensity of user $i$.

### B. Video Requests Model

In the social community, we consider that video requests can be made in two ways: 1) actively watch and 2) passively watch. We describe the two circumstances in the following.

*1) actively watch:* this refers to the situation that the users actively watch videos they are interested in on streaming platforms such as Youtube. Mathematically, we assume the active video request from each user obeys the Poisson process with parameter $\omega$. Then, we consider that each user in the same community shares a common preference for videos while video preference in each community may be different. Also, each video has different popularity in the community following the Zipf distribution [32]. To present such concept more succinctly, we set $\tau_{m,f} \in \{1, ..., L\}$ as the rank of the popularity of video $f$ in community $m$. Note that the rank of videos will change every period of time. Finally, the probability for the users in community $m$ requesting for video $f$ are given as:

$$P_{m,f}^{a} = \frac{(\tau_{m,f})^{-\nu}}{\sum_{l=0}^{L} l^{-\nu}}, \forall f \in F \tag{4}$$

where $\nu$ characterizes the steepness of the Zipf distribution. Namely, when $\nu$ goes high, few popular videos account for more of the majority of requests than $\nu$ goes low.

*2) passively watch:* this happens when the users watch videos shared by his friends on social media such as Facebook. Generally, we say that a video is disseminated from user $i$ to user $j$ if user $j$ requests for the video shared by the user $i$. To describe this phenomenon, we first consider the usage habit of users by denoting $0 \leqq \Omega_j \leqq 1$ as the tendency of user $j$ to react to a post on social media, which can be obtained by his usage history on social media. Then, assume the willingness to watch the shared video depends on the degree of importance of the user who shares the video, we can calculate the likelihood of video dissemination from user $i$ to user $j$ by:

$$P_{i,j}^{s} = \Omega_j \cdot Im_{j,i} \tag{5}$$

Also, we consider dissemination delay between users, where a video can only be disseminated 1-hop away from the user who shared it during a time slot. We further assume that in each time slot: i) each user can request for multiple videos through either actively watch or passively watch but can only watch one video at the same time ii) each user will share the video after watching it, and iii) each user will share the same video only once. To signify the viewing status of a user to a video, we introduce an auxiliary binary variable in (6):

$$\Theta_{i,f} = \begin{cases} 1, & \text{if user } i \text{ has not seen the video } f \\ 0, & \text{Otherwise.} \end{cases} \tag{6}$$

Based on the above deliberation, we can abstract the dissemination state of a video in community $m$ as a bi-directed weighted graph $G_m^{(k)} = \{U_m^{(k)}, P^s, \Theta\}$, where $U_m^{(k)}$ is the set of nodes denoting users, $P^s = \{P_{1,2}^s, P_{2,1}^s, P_{2,3}^s \cdots\cdots P_{i,j}^s\}$

TABLE II
LIST OF NOTATIONS

| Notation | Definition |
|---|---|
| $U_m^{(k)}$ | The set of users in community $m$ in time slot $k$ |
| $F, M, \chi$ | The set of videos, edge servers and resolution level, respectively |
| $s_{i,j}(t)$ | Indicator of interaction event from user $i$ to user $j$ at instant $t$ |
| $S_{i,j}^{(k)}$ | Interaction intensity from user $i$ to user $j$ in time slot $k$ |
| $\hat{S}_{i,j}$ | Predicted interaction intensity from user $i$ to user $j$ at present |
| $Im_{i,j}$ | Importance of user $j$ toward user $i$ at present |
| $\tau_{m,f}$ | The rank of the popularity of video $f$ in community $m$. |
| $P_{m,f}^{a}$ | Probability of a user in community $m$ requesting for video $f$ |
| $P_{i,j}^{s}$ | Liklihood of user $i$ request for video shared by user $j$ |
| $\bar{P}_{i,f}^{(q)}$ | Infected probability of user $i$ to video $f$ within $q$ time slot |
| $P_{i,f}^{(k)}$ | Probability of user $i$ request for video $f$ at time slot $k$ |
| $R_{i,f}^{(k)}$ | Indicator denoting if user $i$ request for video $f$ in time slot $k$ |
| $\Omega_i$ | The tendency of user $i$ to react to a post on social media |
| $\Theta_{i,f}$ | Indicator denoting if user $i$ has seen the video $f$ |
| $r_{i,f}^{(t)}$ | Resolution decision of user $i$ to video $f$ at instant $t$ |
| $b_i^{(t)}$ | Downlink capacity of user $i$ at instant $t$ |
| $Bh^{(t)}$ | Avaliable download rate through backhaul network |
| $Bs_m^{(t)}$(f) | Level of video $f$ in edge server $m$'s processing buffer at instant $t$ |
| $Bu_i^{(t)}$(f) | Level of video $f$ in user $i$'s playback buffer at instant $t$ |
| $EN_{m,f}^{(k)}$ | Expected # of requests for $f$ in community $m$ in time slot $k$ |
| $Q_{i,f}$ | Quality of experience of user $i$ toward video $f$ |
| $\bar{Q}_{i,f}$ | The expected QoE value of user $i$ toward video $f$ |
| $QoS_{i,f}^{(t)}$ | Quality of service of user $i$ toward video $f$ at instant $t$, |
| $EI_{i,f}^{(t)}$ | Expect intactness of user $i$ toward video $f$ at instant $t$ |
| $D_m^{eg}, p$ | Cache size, processing capacity of edge server $m$ |
| $C_{m,f}$ | Edge server $m$'s cache decision on video $f$ |
| $T_f^L, n_f$ | The cache lifetime/ The number of chunks of entire video $f$ |
| $v(\chi^*)$ | The size of a video chunk with highest resolution $\chi^*$ |
| $\hat{T}_{i,f}^{(t)}$ | Remaining time of video $f$ at instant $t$ during $i$'s playback |
| $k, t$ | Present system time slot (in hours)/ Present moment (in seconds) |
| $T, \Delta t$ | Time duration of a system time slot and a video chunk |

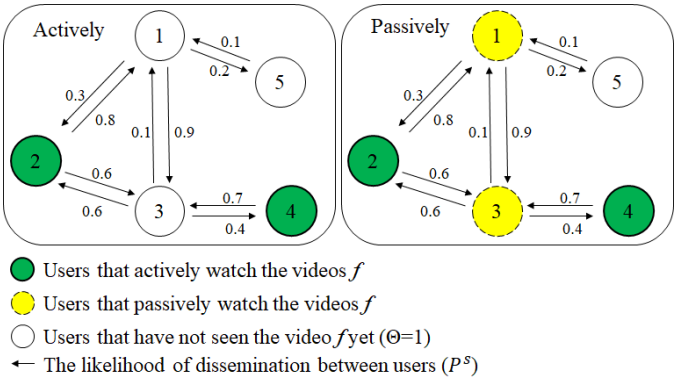



Fig. 2. Example of dissemination process of video $f$ in a community

denotes the set of edge representing the likelihood of dissemination between users and $\Theta$ is the set of value of each node, signifying the viewing status of users to videos. While the video disseminating in the community, the graph $G_m^{(k)}$ would be updated in each time slot accordingly. An example of the video dissemination process is illustrated in Fig. 2.

### C. Network Model

In this subsection, we first introduce how downlink transmission of users is considered and then elaborate on the envisioned 2-tier MEC cache architecture.

*1) Wireless Communication:* We consider that the spectrum used by different users in the same community is orthogonal and that the spectrum reuse factor for all communities is 1. Thus, only inter-cell interference is considered. Then, the achievable data rate of user $i$ at instant $t$ can be written as:

$$b_i^{(t)} = W \log_2 \left( 1 + \frac{g_{i,l} PW}{\sigma^2 + \sum\limits_{m \in M, m \neq l} g_{i,m} PW} \right) \tag{7}$$

where $W$ is the dedicated frequency band, $g_{i,l}$ is the channel gain between $i$ and local edge server $l$ including large-scale pathloss and shadowing. Additionally, $\sigma^2$ is the noise variance of the additive white Gaussian noise (AWGN).

*2) Collaborative Cache:* To begin with, assuming that each user does not forward-seek the video during the playback, we consider a partial cache scheme where each video is cached with different number of chunks, continuously counted from the beginning of the video. Then, we denote the vector $C_m = [C_{m,1}, C_{m,2}, ...C_{m,L}]$ as the cache decision of edge server $m$, where $C_{m,f} \in \mathbb{Z}$ denotes the number of chunks the video $f$ is cached. Eventually, we denote the collaborative cache decision of all edge servers as $C = [C_1, C_2, ....., C_h]$ where $C_m \in C$ signify the cache strategy of the edge server $m$.

For each video chunk, instead of caching every resolution version, the edge servers will only cache the highest resolution version (i.e. $\chi^*$) to save cache space. Then, during the playback, the edge server would dynamically transcode the cached video chunks from the highest resolution into the requested resolution. In this way, though it will cost additional computing resources, we can further enhance the cache hit ratio while alleviating traffic of the backhaul network [33] in that video requests for all possible resolutions can be meet through performing transcoding on cached chunks.

From another viewpoint, the variation of the popularity of each video in terms of time may be quite different in reality. For instance, in the flash crowd phenomenon [34] where a large number of video requests arrive at the system within a short time to participate in certain online events, the popularity of the requested videos may fade quickly after the end of events. Therefore, to improve the utilization of storage resources, we set the cached videos with different lifetimes based on their properties. If a video is out of its lifetime, it is removed from the cache list of the edge server so that we can cache other videos that may go viral. To derive proper cache lifetime, we first denote the expected number of requests for video $f$ in community $m$ in time slot $k$ by $EN_{m,f}^{(k)}$, which can be obtained by:

$$EN_{m,f}^{(k)} = \sum_{i=0}^{u_m} P_{i,f}^{(k)}, \forall f \in F \tag{8}$$

where $P_{i,f}^{(k)}$ is the probability of user $i$ request for video $f$ at time slot $k$. Then, we design the lifetime of a video as the time that minimum increment of $EN$ may occur based on our community model, which can occur when the dissemination process slows down or is terminated. Mathematically, the cache lifetime of the video $f$ can be obtained by:

$$T_f^L = T \cdot \arg\min_k \left( EN_{m,f}^{(k)} - EN_{m,f}^{(k-1)} \right) \tag{9}$$

Finally, we consider that each edge server is connected with each other with Mp3 interface and that the regional server manages all the edge servers through Mm5 interface, as defined by ETSI in its MEC related documents [2]. This can enable edge servers to realize collaborative cache by sharing their cached videos with each other through Mp3 interface.

According to the system mentioned above, there are 3 cases of how a video request will be handled: $Case1$ : the requested video hasn't been cached. $Case2$ : the requested video is cached at the local edge server. $Case3$ : the requested video is available in the region, but is cached at another edge server. For case 1, the video request will be forwarded to the content server and the user can only fetch the video through the backhaul network with relatively long latency. For case 2, the edge server will transcode the cached video chunks into the requested resolution version before delivery, which takes processing time of $\frac{v(\chi^*)}{p}$. For case 3, the regional server redirects the request from local edge server $l$ to target edge server $s$ that has cached the requested video with propagation time $t_{l,s}^p$. Afterward, the edge server also transcode the requested video with processing time $\frac{v(\chi^*)}{p}$. Note that since Mp3 interfaces are dedicated channels with little occupation, the propagation time is relatively short compared with backhaul network.

### D. QoE Model

In this paper, different from state-of-the-art adaptive streaming solutions [35]-[36] that are designed for client-side adaptation, we consider the server-side bitrate selection scheme [37], where system-level information such as network throughput is

available. In this way, the MEC cache strategy can cooperate with the video adaptation procedure to provide better service. Hence, we express the streaming process as a two-buffer system, consisting of user's playback buffer and edge server's processing buffer. We assume that both buffers are linear so that data will continuously be stored in it while the video is playing. Accordingly, both in terms of bits, we denote $Bu_i^{(t)}(f)$ as the level of video $f$ in user $i$'s buffer and $Bs_m^{(t)}(f)$ as the level of video $f$ in edge server $m$'s buffer at instant $t$, as shown in (10,11):

$$Bu_i^{(t)}(f) = \max\left(Bu_i^{(t-1)}(f) + \left(b_i^{(t)} - r_{i,f}^{(t)}\right), 0\right) \tag{10}$$

$$Bs_m^{(t)}(f) = \max\left(Bs_m^{(t-1)}(f) + \left(Bh^{(t)} - \frac{\chi^*}{r_{i,f}^{(t)}} b_i^{(t)}\right), 0\right) \tag{11}$$

where $(t-1)$ indicate previous moment and $r_{i,f}^{(t)}$ is the bitrate of resolution decision of the user $i$ to video $f$ at instant $t$.

Afterward, as pointed out by several works in DASH [5],[7],[29], the main factors that affect the QoE in video streaming are 1) video resolution, 2) quality variation rate, 3) initial waiting time and 4) video stalling time. We will mathematically describe these factors in the following.

First, we consider video resolution in our QoE model as the average bitrate during streaming. Thus, the resolution of video $f$ that user $i$ receives is specified as:

$$VQ_{i,f} = \frac{1}{n_f \Delta t} \int_{t=0}^{n_f \Delta t} r_{i,f}^{(t)} dt \tag{12}$$

Another metric toward users' QoE is the quality variation rate, which can be obtained by the difference of current resolution and resolution in the previous moment. Here, we take the square of the resulting value to keep it positive. Then, the average quality variation rate of video $f$ that the user $i$ experiences is defined as:

$$SW_{i,f} = \frac{1}{n_f \Delta t} \int_{t=1}^{n_f \Delta t} \left(r_{i,f}^{(t)} - r_{i,f}^{(t-1)}\right)^2 dt \tag{13}$$

As for initial waiting time, we define it as the time duration from the arrival time of the user until the time that the data in users' playback buffer reaches the maximum capacity of $D_{bf}$. Based on discussions in the previous subsection, the initial buffer time for delivering video $f$ to user $i$ can be written as:

$$T_{i,f}^w = \begin{cases} t_B^R + \frac{D_{bf}}{b_i^{(t)}}, & \text{Case 3} \\ t_F^R + \frac{v(\chi^*)}{p} + \frac{D_{bf}}{b_i^{(t)}}, & \text{Case 1} \\ t_F^R + t_{l,s}^p + \frac{v(\chi^*)}{p} + \frac{D_{bf}}{b_i^{(t)}}, & \text{Case 2} \end{cases} \tag{14}$$

where $t_F^R$ and $t_B^R$ denote round trip time (RTT) of fronthaul network and backhaul network, respectively. The last QoE metric is video stalling, which happens when the playback buffer gets empty (e.g. $Bu_i^{(t)}(f) = 0$). Without loss of generality, we formulate the stalling time of video $f$ perceived by user $i$ as:

$$T_{i,f}^{st} = \frac{1}{n_f \Delta t} \int_{t=0}^{n_f \Delta t} ST_{i,f}^{(t)} dt \tag{15}$$

where $ST_{i,f}^{(t)} \in \{0,1\}$ is an indicator denoting whether user $i$'s playback buffer gets empty. That is, $ST_{i,f}^{(t)} = 1$ if $Bu_i^{(t)}(f) = 0$ and $ST_{i,f}^{(t)} = 0$ otherwise. Finally, we define the QoE function of user $i$ toward video $f$ as a weighted sum of 4 factors mentioned above, expressed as follow:

$$Q_{i,f} = \alpha VQ_{i,f} - \beta SW_{i,f} - \gamma T_{i,f}^w - \delta T_{i,f}^{st}, \begin{cases} \alpha+\beta+\gamma+\delta = 1 \\ 0 \le \alpha,\beta,\gamma,\delta \le 1 \end{cases} \tag{16}$$

where the $\alpha, \beta, \gamma, \delta$ are the weight factors representing the importance of these four different metrics respectively and the constraint ensures that the result of weight sum can be normalized into [0, 1]. Also, since the considered 4 QoE factors have different units and scales, we will normalize these factors respectively before aggregating.

## IV. Problem Formulation

In this section, we will mathematically formulate the objective of this work, which is to optimize QoE of video requests from multiple social communities by performing video caching and adaptation. Specifically, the video adaptation will be conducted in each system instant in order to maximize users' QoE under dynamic wireless conditions. In contrast, the video cache decision is only made in each system time slot to avoid frequently updating the cache list, costing considerable bandwidth of the backhaul network. Afterward, due to the heterogeneous time scale of these two operations, we further decompose the original problem into two corresponding subproblems to optimize our ultimate goal.

To begin with, we first introduce an auxiliary binary variable to present the state of video requests received from users during each time slot $k$:

$$R_{i,f}^{(k)} = \begin{cases} 1 & \text{if user } i \text{ request for video } f \text{ in time slot } k \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

Then, to formulate the adaptation process of each video request in (17), we denote $r_{i,f} = \{r_{i,f}^{(1)}, r_{i,f}^{(2)}, ...\}$ as the vector of resolution decision of user $i$ to video $f$ at any instant $t$ during the time slot $k$ (i.e. $\forall t \in [(k-1)T, kT]$). Note that for the instant $t$ outside of user playback, the value of $r_{i,f}^{(t)}$ will be 0. Based on the above elaboration, we can formulate our optimization problem in the following:

$$\begin{aligned} &\max_C \max_r \sum_{n=1}^{q} \sum_{m=0}^{h} \sum_{i=0}^{u_m} \sum_{f=0}^{L} R_{i,f}^{(k+n)} \cdot Q_{i,f}(C_{m,f}, r_{i,f}) \\ \text{s.t.}\quad & C1: \sum_{f \in F} C_{m,f} \cdot v(\chi^*) \le D_m^{eg}, \forall m \in M \\ & C2: C_{m,f} \le n_f, \forall m \in M, f \in F \\ & C3: \sum_{f \in F} C_{m_1,f} \cdot C_{m_2,f} = 0, \forall m_1 \ne m_2, m_1, m_2 \in M \\ & C4: \sum_{i=0}^{u_m} \sum_{f=0}^{L} r_{i,f}^{(t)} \le p, \forall t, m \in M \\ & C5: r_{i,f}^{(t)} \le b_i^{(t)}, \forall t, m \in M, ID(i) \in U_m^{(k)}, f \in F \\ & C6: r_{i,f}^{(t)} \in \chi, \forall t, m \in M, ID(i) \in U_m^{(k)}, f \in F \end{aligned} \tag{18}$$

which aims to maximize overall QoE of requests from all communities within the next $q$ time slot by determining collaborative cache decision of edge servers (i.e. $C$) and resolution decision of each request (i.e. $r_{i,f}$). In (18), constraint C1 describes that for each edge server, the total size of all video chunks to cache should not exceed available cache space. Constraint C2 states that the number of cached chunks of a video can't exceed the total number of chunks of the video. Then, C3 is set to avoid duplicate cache decisions in different edge servers. Also, constraint C4 sets that the sum of required bitrates for videos can't exceed processing capacity for each edge server. Constraint C5 signifies that each user's resolution decision can't exceed his downlink capacity. Finally, constraint C6 represents the encoding bitrate of the transcoded videos must fall in the set of the resolution level $\chi$.

Since cache decision are coupled with resolution decision, and their time scale is unequal, we decompose problem (18) into two different phase and corresponding subproblems:

*1) Caching phase **P1**:* optimizing collaborative caching decision of all edge servers $C$ given their available cache space and users' information at the beginning of each time slot.

*2) Adaptation phase **P2**:* optimizing resolution decision $r$ of all users in each instant based on network condition and the results of the collaborative caching.

We first elaborate on the subproblem **P1**, which is designed to be optimized by the regional server in the caching phase. To begin with, on account that we have no way of knowing what requests for which videos will come in the future, we should replace the indicator of the actual request from the users (i.e. $R_{i,f}^{(k)}$) by the probability of request (i.e. $P_{i,f}^{(k)}$). Similarly, we should replace the actual QoE of users (i.e. $Q$) with the expected QoE value (i.e. $\bar{Q}$). However, since the channel condition and resolution decision of users is unknown in advance, here we calculate the value of $\bar{Q}$ by substituting users' average resolution in the previous time slot into the original QoE model (16). Afterward, the only variable that affects the value of $\bar{Q}$ is the collaborative cache decision $C$. Finally, after adapting from (18), we define the subproblem **P1** as the following:

$$\mathbf{P1}: \quad \max_{C} \sum_{n=1}^{q} \sum_{m=0}^{h} \sum_{i=0}^{u_m} \sum_{f=0}^{L} P_{i,f}^{(k+n)} \cdot \bar{Q}_{i,f}(C_{m,f}) \quad \text{s.t.C1, C2, C3} \tag{19}$$

which aims to maximize the expected QoE (i.e. $\bar{Q}$) of each possible video request within next $q$ time slot by determining the collaborative cache decision of all edge servers (i.e. $C$). It can be observed that since $C$ is an integer vector, **P1** is an integer non-linear programming problem, which is NP-hard.

Then, we turn to define the subproblem **P2**, which is designed to be optimized by each edge server in the adaptation phase. However, since the video adaptation is executed in real time, a quality of service (QoS) function should be further designed for the optimization problem, instead of using the original QoE model, which can only be measured at the end of playing.

To begin with, we don't need to consider the initial waiting time in our QoS function since **P2** is operated in real-time. In addition, since video stalling time can only be measured at the end of playing, we intend to provide a more reasonable method to forecast the degradation of users' QoE caused by interruptions during playback. To achieve that goal, we introduce the concept of expected intactness of video, which is defined as the percentage of the video that can be played without interruption if current resolution decision $r_{i,f}^{(t)}$ and available download rate through backhaul network $Bh^{(t)}$ remain unchanged during the rest of the playback. Unlike the uncertainty of the stalling event, we can directly derive the expected intactness at any instant $t$ during playback. Based on our two buffer systems in Sec. III, the expected intactness $EI_{i,f}^{(t)} \in [0,1]$ for user $i$ toward video $f$ in community $m$ at instant $t$ can be calculated by:

$$EI_{i,f}^{(t)} = \begin{cases} \min\left(1, \frac{b_i^{(t)} \cdot Bs_m^{(t)}(f)}{\hat{T}_{i,f}^{(t)}\left(\chi^* b_i^{(t)} - r_{i,f}^{(t)} Bh^{(t)}\right)}\right), & \text{if } Bh^{(t)} \leq r_{i,f}^{(t)} \\ 1, & \text{otherwise.} \end{cases} \tag{20}$$

In this equation we can observe that if $Bh^{(t)} > r_{i,f}^{(t)}$, the video $f$ can be played without any stalling (i.e. $EI_{i,j}^{(t)} = 1$). Otherwise, the value of $EI_{i,f}^{(t)}$ would depend on the current buffer level of edge server $Bs_m^{(t)}(f)$, the remaining time of the video $\hat{T}_{i,f}^{(t)}$, selected resolution $r_{i,f}^{(t)}$ and download rate $Bh^{(t)}$. Note that we assume current network conditions remain unchanged only to estimate the expected intactness. Meanwhile, the value of both backhaul throught $Bh^{(t)}$ and downlink capacity $b_i^{(t)}$ will vary with time and affect users' QoE.

Afterward, adapting from the QoE model (16) and the above elaboration, the metrics that may affect the QoS in real time will be the instant resolution, instant quality variation, and the expected intactness of the remaining video. Thus, the QoS of user $i$ toward video $f$ at instant $t$ is written as:

$$QoS_{i,f}^{(t)} = \alpha r_{i,f}^{(t)} - \beta\left(r_{i,f}^{(t)} - r_{i,f}^{(t-1)}\right)^2 + \delta EI_{i,f}^{(t)} \tag{21}$$

Here, the value of the weighted parameter $\alpha, \beta$ and $\delta$ is set to be the same as in the QoE function (16). Finally, we define the objective of **P2** as maximizing QoS of the set of video requests collected in community $m$ at instant $t$ of time slot $k$ by determining video resolution $r$:

$$\mathbf{P2}: \quad \max_{r} \sum_{i=0}^{u_m} \sum_{f=0}^{L} R_{i,f}^{(k)} \cdot QoS_{i,f}^{(t)}(r_{i,f}^{(t)}) \quad \text{s.t.C4, C5, C6} \tag{22}$$

which is also an integer non-linear programming problem according to our definition.

It should be noted that these two subproblems didn't equivalent to the original problem (18). Instead, we intend to achieve objective of (18) by optimizing cache decision and resolution decisions on the regional server and edge servers respectively.

## V. The Proposed Video Cache and Adaptation Framework

In this section, we will first address the way we optimize the subproblems **P1** and, **P2** respectively. Afterward, we will propose our optimization framework and elaborate on its workflow.

### A. Collaborative Social- aware Video Caching

In this subsection, we propose a social-aware proactive video cache algorithm which embeds video dissemination process into the cache mechanism to address problem **P1**.

We should first obtain the probabilities $P_{i,f}^{(k)}$ in the objective of **P1** so that **P1** can be solved. Therefore, we elaborate on how to mathematically derive $P_{i,f}^{(k)}$ by utilizing the video dissemination among users. We start from the special case that there is a source user $s$ being the first to share the video $f$ on social media in community $m$. If the number of the considered time slot is 1 (i.e. $q = 1$), it is obvious that only the users 1-hop away from $s$ can be influenced by the video dissemination process. Therefore, we can directly obtain the probability that a video is disseminated from source user $s$ to target user $t$ by $P_{t,s}^{s}$ in (5). Nonetheless, for the case that $q$>1, all of target users $t$ within $q$-hop from $s$ can be influenced. For that case, we should consider all of the directed acyclic paths from $s$ to $t$ in the graph $G_m^{(k)}$ whose length is less than $q$, which can be easily obtained by path search algorithm such as breadth-first search (BST). To that end, we denote the set of all possible path from $s$ to $t$ by:

$$d_{s,t} = \{\sigma = [s, ....., t] | \forall P_{\sigma[i-1],\sigma[i]}^{s} > 0\} \tag{23}$$

where each element $\sigma$ in $d_{s,t}$ is a sequence starting from $s$ and ended in $t$. We then compute the probability of all of the hop users in single path $\sigma$ sharing the video $f$ to their friends, called the sharing probability of the path $\sigma$. Since users do not share the same video again if they have seen it (i.e. $\Theta = 0$), we can derive the sharing probability of $\sigma$ as:

$$\prod_{n=1}^{len(\sigma)} \Theta_{\sigma[n],f} \cdot P_{\sigma[n-1],\sigma[n]}^{s} \tag{24}$$

where $\sigma[n]$ is the $n^{th}$ hop user of the possible path $\sigma$ from $s$ to $t$ will go through and $len(\sigma)$ denote the length of the sequence $\sigma$.

Next, to compute the probability that a target user $t$ request for the video shared by a source user $s$, called the infected probability of $t$, we aggregate the sharing probability of each possible path in the result of (23) since the sharing probability of each path can be regarded as independent. Therefore, the infected probability of target user $t$ can be derived by:

$$\sum_{\sigma \in d_{s,t}} \prod_{n=1}^{len(\sigma)} \Theta_{\sigma[n],f} \cdot P_{\sigma[n-1],\sigma[n]}^{s} \tag{25}$$

Then, we extend the result of (25) to the general scenario that there are multiple source users who have shared the video $f$ on social media. Given the set of source users $SU$ in the community, the infected probability of any user $i$ within $q$-hop from any source user in $SU$ can be derived as:

$$\widetilde{P}_{i,f}^{(q)} = 1 - \prod_{s \in SU} \left(1 - \sum_{\sigma \in d_{s,i}} \prod_{n=1}^{len(\sigma)} \Theta_{\sigma[n],f} \cdot P_{\sigma[n-1],\sigma[n]}^{s}\right) \tag{26}$$

Besides, recall that in our system model, a video can be requested either actively or passively (i.e. being infected). That is, $P_{i,f}^{(k)} = \widetilde{P}_{i,f}^{(k)} \cup P_{m,f}^{a}$. Since these two situations are pairwise independent events, the following equation will hold: $\widetilde{P}_{i,f}^{(k)} \cap P_{m,f}^{a} = \widetilde{P}_{i,f}^{(k)} \cdot P_{m,f}^{a}$. Accordingly, the probability of a user $i$ in community $m$ requesting for the video $f$ at time slot $k$ (i.e. $P_{i,f}^{(k)}$ in **P1**), can be calculated as follow:

$$P_{i,f}^{(k)} = \widetilde{P}_{i,f}^{(k)} + P_{m,f}^{a} - \widetilde{P}_{i,f}^{(k)} \cdot P_{m,f}^{a} \tag{27}$$

Finally, given the user's information in social community (i.e. $P^s, \Theta$), we can predict the probabilities $P_{i,f}^{(k)}$ in the objective of **P1** by (27). Nonetheless, since **P1** is an NP-hard problem, it is challenging to find the optimal solution. Inspired by the simulated annealing (SA) method [38], we proposed an efficient heuristic algorithm to search for the sub-optimal solution to **P1**.

SA is based on the idea of exploring the neighborhood of a potential solution randomly, accepting occasional changes that may worsen the solution with a probability that decreases over time. To apply SA in solving our problem, we first describe a few necessarily notations. Similar to $C$ in Sec. III. C,the transitive solution of a collaborative caching strategy is encoded into a sequence, denoted by $C^* = [C_1, C_2, ....., C_h]$. Then, we say a sequence $C^{'}$ is the neighbor solution to $C^*$ if it not only is a valid solution to **P1** but also satisfies the following constraint:

$$\left| C_{m,j}^{'} - C_{m,j}^{*} \right| \leq 1, \forall m \in M, f \in F \tag{28}$$

which implies that the difference of each cache decision in $C^{'}$ and $C^*$ must be either 1, -1 or 0. For example, $C^{'} = [1, 0, 2]$ is a neighbor solution to $C^* = [1, 1, 2]$. Also, we denote the objective function of **P1** by $\varpi$. Afterward, we can compute the value of $\varpi(C^*)$ to evaluate how well a potential solution $C^*$ is. Generally, the procedure of SA contains three steps: Initialization, Selection and Cooling. We then go through these steps in the following.

*1) Initialization:* First, the algorithm will generate an initial solution $C^*$ with an initial temperature $\Lambda$. However, since the performance of solutions derived by SA to some degree depends on the initial solution, we design the algorithm to select the initial solution in a greedy manner to speed up convergence. Specifically, we obtain the initial neighbor solution set $\Gamma_0$, containing all of the neighbor solutions to [0, 0, ....., 0]. Then, we set the initial solution $C^*$ to be the neighbor solution $C^{'}$ in $\Gamma_0$ with the highest value of $\varpi(C^{'})$. Note that this procedure only performs once at the beginning of the algorithm.

*2) Selection:* During each iteration $k$, we will obtain the potential solution set $\Gamma_k$, containing all of the neighbor solution to the current solution $C^*$. Then, the SA will select a neighbor solution $C^{'}$ from $\Gamma_k$ at random. After that, the algorithm will decide whether to move to the new solution $C^{'}$ or to stay at the current solution $C^*$. To that end, we denote $\Delta = \varpi(C^{'}) - \varpi(C^*)$. If the new solution is better (i.e. $\Delta > 0$), the algorithm will accept $C^{'}$ as the current solution. On the other hand, if the new solution is worse than the current one, the algorithm will accept $C^{'}$ with probability $e^{-\frac{\Delta}{\Lambda}}$. Eventually, if the selected solution $C^{'}$ is not accepted, the algorithm will select a new neighbor solution $C^{'}$ from $\Gamma_k$ until a new solution

**Algorithm 1:** Collaborative Social-aware Video Caching

**Input:** Information of users: $P^s, \Theta$, Video requests at last time slot $R^{(k-1)}$, Available cache space $D_m^{eg}$
**Output:** The cache decision $C^*$ and cache lifetime $T_f^L$

Let $C^*$ to be transition solution, $\Lambda$ to be inital temperature
Calculate $P_{i,f}^{(k+n)}, \forall i, f, n$ by (27), and initialze $C^*$
**while** $\Lambda > \Lambda_{min}$ **do**
  $\Gamma_k \leftarrow$ the set of neighbor solution to $C^*$
  **while** $S \neq \phi$ **do**
    Randomly pop a neighbor solution $C^{'}$ from $\Gamma_k$
    Set $\Delta = \varpi(C^{'}) - \varpi(C^*)$
    **if** $\Delta > 0$ **then**
      $C^* \leftarrow C^{'}$
    **else**
      $C^* \leftarrow C^{'}$ with probability $e^{-\frac{\Delta}{\Lambda}}$
    **if** *New solution is accepted* **then**
      break
  Cooling: $\Lambda = \rho \cdot \Lambda$
**foreach** $C_{m,f}$ *in* $C^*$ **do**
  **if** $C_{m,f} > 0$ **then**
    Edge server $m$ cache $C_{m,f}$ chunks of video $f$ with lifetime $T_f^L$, calculated by (9)

**Algorithm 2:** QoE-Driven Video Adaptation

**Input:** Resolutions in last instant $r^{(t-1)}$, Backhaul download rate $Bh^{(t)}$, Buffer level $Bs_m^{(t)}$, Remaining time $\hat{T}^{(t)}$
**Output:** The set of resolution decision $r^{(t)}$

Set dual variables $\lambda, \mu$ to be non-negative, the set of transition solution $r^{(t)} \leftarrow r^{(t-1)}$, $\tilde{L}^* \leftarrow -\infty$
**while** $|\tilde{L}^* - \tilde{L}^k| < \varepsilon$ **do**
  Update all of $r_{i,f}$ in $r^{(t)}$ with current value of $\lambda_{i,f}, \mu$ by (35-36) and then update dual variable $\lambda_{i,f}, \mu$ by (37)
  **if** $\tilde{L}^k > \tilde{L}^*$ **then**
    $\tilde{L}^* \leftarrow \tilde{L}^k$
**foreach** $r_{i,f}^{(t)}$ *in* $r^{(t)}$ **do**
  **if** $QoS_{i,f}\big(h(r_{i,f}^{(t)})\big) > QoS_{i,f}\big(l(r_{i,f}^{(t)})\big)$ **then**
    $r_{i,f}^{(t)} = h(r_{i,f}^{(t)})$
  **else**
    $r_{i,f}^{(t)} = l(r_{i,f}^{(t)})$
**while** $\sum_{i \in U_m} \sum_{f \in F} r_{i,f}^{(t)} > p$ **do**
  Set $\Delta QoS = \{QoS_{i,f}(l(r_{i,f}^{(t)})) - QoS_{i,f}\big(r_{i,f}^{(t)}\big) | \forall i, f\}$
  Sort $\Delta QoS$ in ascending order
  **foreach** *(i,f) in* $\Delta QoS$ **do**
    Set $r_{i,f}^{(t)} = l(r_{i,f}^{(t)})$
    **if** $\sum_{i \in U_m} \sum_{f \in F} r_{i,f} \leq p$ **then**
      Return the set of resolution decision $r^{(t)}$

is accepted or the potential solution set become empty.

*3) Cooling:* In the final step, the algorithm will update current temperature by $\Lambda = \rho \cdot \Lambda$ where $0 < \rho < 1$ is cooling coefficient. In this way, the temperature will progressively decrease in each iteration, resulting in the probability of accepting a worse new solution gradually decreasing to 0. Thus, the algorithm will ultimately converge.

The proposed algorithm would repeat steps (2-3) until the temperature $\Lambda$ is lower than the minimum temperature $\Lambda_{min}$. After the cooling process is terminated, the algorithm sets each cache decision in finally solution $C^*$ with corresponding lifetime, obtained by (9). Note that we can obtain the value of (9) by substituting the result of (27). The overall procedure is illustrated in Alg. 1.

### *B. QoE-Driven Video Adaptation*

In this subsection, we resolve to optimize the subproblem **P2** based on the caching result of **P1**. However, since the bitrate of transcoded video for every user must fall in the set of the resolution level $\chi$, as the constraint C6 noted, we can't directly address the original problem. Instead, we relax each resolution variable $r_{i,f}^{(t)}$ to real number. After obtaining the near optimal value of the relaxed resolution, we recover it from real number to satisfy the constraint C6. To begin with, we relax each resolution variable (in bitrate) $r_{i,f}^{(t)}$ into $[0, b_i^{(t)}]$ since the transcoded bitrate must be less than the downlink capacity of users. After reassigning the domain of the resolution variable to a continuous set of real numbers, we adopt the Lagrange multiplier method to transform **P2** into an unconstrained optimization problem by introducing the dual variables $\lambda$ and $\mu$ for constraints C4 and C5. For any community $m$ during time slot $k$ at instant $t$, the Lagrange function of **P2** can be shown as:

$$\tilde{L}(r, \lambda, \mu) = \sum_{i}^{u_m} \sum_{f=0}^{L} R_{i,f}^{(k)} \Big( \alpha r_{i,f}^{(t)} - \beta \big( r_{i,f}^{(t)} - r_{i,f}^{(t-1)} \big)^2 + \delta EI_{i,f}^{(t)} \Big) + \sum_{i=0}^{u_m} \sum_{f=0}^{L} \lambda_{i,f} \big( r_{i,f}^{(t)} - b_i^{(t)} \big) + \mu \Big( \sum_{i=0}^{u_m} \sum_{f=0}^{L} r_{i,f}^{(t)} - p \Big) \tag{29}$$

It is noted that $\tilde{L}$ is a continuous and differentiable function of $r$, $\lambda$ and $\mu$. Secondly, we derive the Karush-Kuhn-Tucker (KKT) conditions of our Lagrange function, which is necessary and sufficient for a solution in non-linear programming with inequality constraints to be optimal, as follow:

*1) Primal Feasibility:*

$$\begin{aligned} & r_{i,f}^{(t)} - b_i^{(t)} \leq 0, \forall i, \forall f \in F \\ & \sum_{i=0}^{u_m} \sum_{f=0}^{L} r_{i,f}^{(t)} - p \leq 0 \end{aligned} \tag{30}$$

*2) Dual Feasibility:*

$$\begin{aligned} & \lambda_{i,f} \geq 0, \forall i, \forall f \in F \\ & \mu \geq 0 \end{aligned} \tag{31}$$

*3) Complementary Slackness:*

$$\begin{aligned} & \lambda_{i,f} \big( r_{i,f}^{(t)} - b_i^{(t)} \big) = 0, \forall i, \forall f \in F \\ & \mu \Big( \sum_{i=0}^{u_m} \sum_{f=0}^{L} r_{i,f}^{(t)} - p \Big) = 0 \end{aligned} \tag{32}$$

*4) Stationarity:*

$$\frac{\partial \tilde{L}}{\partial r_{i,f}^{(t)}} = \alpha - 2\beta\left(r_{i,f}^{(t)} - r_{i,f}^{(t-1)}\right) - \delta\frac{\partial EI_{i,f}^{(t)}}{\partial r_{i,f}^{(t)}} + \lambda_{i,f} + \mu = 0, \qquad \forall i, \forall f \in F \tag{33}$$

According to the definition of the expected intactness (20), for the general case where $EI_{i,f}^{(t)} < 1$, we can directly differentiate $EI_{i,f}$ with respect to each $r_{i,f}$ by its definition while for the case that $EI_{i,f}^{(t)} = 1$, its derivative will be 0. In summarize, the partial differentiation of $EI_{i,f}$ with respective to $r_{i,f}^{(t)}$ can be expressed as:

$$\frac{\partial EI_{i,f}^{(t)}}{\partial r_{i,f}^{(t)}} = \begin{cases} 0, & \text{if } EI_{i,f}^{(t)} = 1 \\ \frac{b_i^{(t)} \cdot Bh^{(t)} \cdot Bs_m^{(t)}(f)}{\hat{T}_{i,f}^{(t)}\left(Bh^{(t)} r_{i,f}^{(t)} - \chi^* b_i^{(t)}\right)^2}, & \text{else.} \end{cases}, \forall i, \forall f \in F \tag{34}$$

For the case that $EI_{i,f}^{(t)} = 1$, we can compute the solution of $r_{i,f}^{(t)}$ in (33) for the user $i$ given dual variable$\lambda_{i,f}$ and $\mu$ by directly substituting the equation (34) into (33). The result is shown as follow:

$$r_{i,f}^{(t)} = \frac{\alpha + \lambda_{i,f} + \mu}{2\beta} + r_{i,f}^{(t-1)} \tag{35}$$

As for the case that $EI_{i,f}^{(t)} < 1$, we first arrange the result of the equation obtained by substituting (34) into (33). After that, we get a cubic polynomial equation of the transcoding resolution $r_{i,f}^{(t)}$, as shown in the following:

$$2\beta r_{i,f}^{(t)}\left(Bh^{(t)} r_{i,f}^{(t)} - \chi^* b_i^{(t)}\right)^2 - \left(2\beta r_{i,f}^{(t-1)} + \alpha + \lambda_{i,f} + \mu\right)\left(Bh^{(t)} r_{i,f}^{(t)} - \chi^* b_i^{(t)}\right)^2 - \delta\frac{b_i^{(t)} Bh^{(t)} \cdot Bs_m^{(t)}(f)}{\hat{T}_{i,f}^{(t)}(Bh^{(t)} r_{i,f}^{(t)} - \chi^* b_i^{(t)})^2} = 0 \tag{36}$$

From the properties that the cubic polynomial must have at least one real root, the equation (36) is said to be solvable since all of its coefficients are real number. Thus, we can obtain the optimal solutions of (33) given dual variables $\lambda_{i,f}$ and $\mu$.

We then update the dual variables $\lambda_{i,f}$ and $\mu$ by the subgradient descent method, as follow:

$$\lambda_{i,f} = \left[\lambda_{i,f} - \eta\left(r_{i,f}^{(t)} - b_i^{(t)}\right)\right]^+, \forall i, \forall f \in F$$
$$\mu = \left[\mu - \eta\left(\sum_{i=0}^{u_m} r_{i,f}^{(t)} - p\right)\right]^+ \tag{37}$$

where $[\cdot]^+ = \max\{\cdot, 0\}$ and $\eta$ denotes the step length.

In the designed solving process, we will first calculate the set of optimal resolution decisions under the fixed dual variable $\lambda_{i,f}$, $\mu$ and then update the dual variables by (37). We repeat the process until convergence conditions is satisfied, defined as $|\tilde{L}^* - \tilde{L}^k| < \varepsilon$, where $\tilde{L}^k$ is the value of $\tilde{L}$ at $k^{th}$ iteration, $\tilde{L}^*$ is historical maximum of $\tilde{L}$ and $\varepsilon$ is a minor constant. Afterward, we should recover the set of $r$ to meet the constraint C6 without violating the constraint C4.

To begin with, we initialize the transition resolution decision of each user by comparing the resulting QoS of the higher and lower supported resolution in $\chi$ closest to the relaxed resolution $r_{i,f}^{(t)}$, denoted by $h(r_{i,f}^{(t)})$ and $l(r_{i,f}^{(t)})$. If the higher resolution decision can bring more QoS to the user, the algorithm sets the transition resolution decision to be the higher one and vice versa. Then, if the summation of the resulting processing bitrate exceeds computing capacity of the edge server, which may happen when there is a massive number of video requests arriving at the same time, the resolution decision set will first be sorted in ascending order based on the difference in QoS if degrading its resolution, denoted by $\Delta QoS$. After that, the algorithm iteratively degrade the resolution in the sorted set until the resolution decision set doesn't violate the constrain C4. If all of the element in the transition set is the lowest resolution in $\chi$, the algorithm would be terminated. Finally, we can obtain the resolution decision of each user under the constraints. The detailed procedure is presented in Alg. 2.

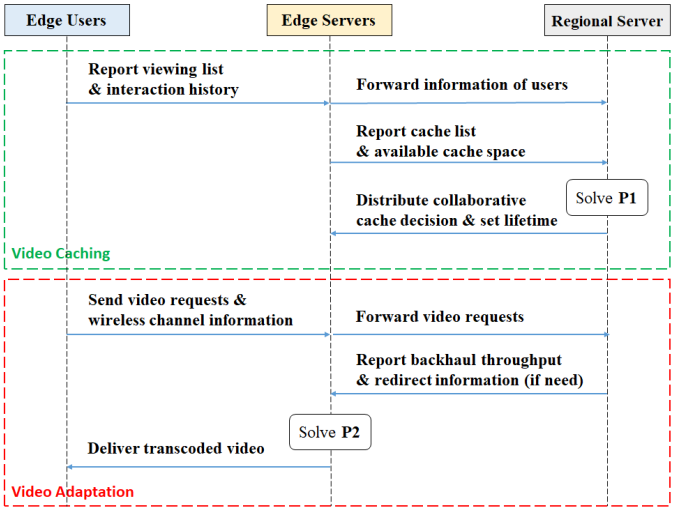



Fig. 3. Workflow of CSQCA framework

## C. The Proposed Framework

We propose the collaborative social-aware and QoE-driven video caching and adaptation (CSQCA) framework to address our objective, involving optimizations of these two subproblems mentioned above. The overall workflow of CSQCA framework is depicted in Fig. 3. In the video caching phase, the regional server will periodically update the cache decision by solving **P1** based on users' information and available cache space forwarded by edge servers. Afterward, the edge server will proactively cache video chunks with explicit lifetime under the instructions of the region server. In the video adaptation phase, the regional server will first assign each request to edge servers based on their cache decision. Afterward, each edge server will dynamically transcode the requested video chunk into the appropriate resolution by solving **P2** in real time based on wireless channel information of users and throughput of backhaul network reported by the regional server.

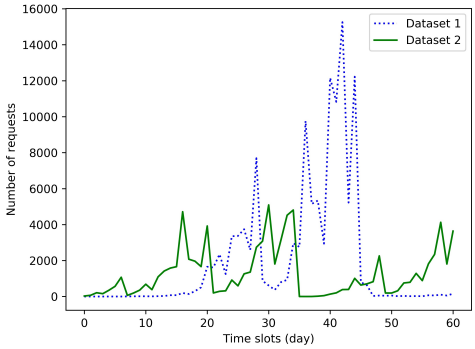



Fig. 4. Traffic pattern in different social communities

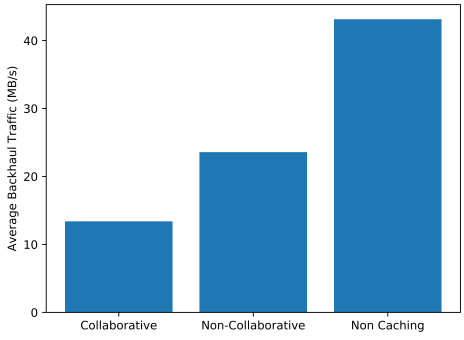



Fig. 5. Average backhaul traffic in different cache scheme

## VI. Simulation Results

In this section, we will first describe the simulation environment. Afterward, we conduct our simulations with real-world dataset to evaluate the performance of the proposed CSQCA framework. Furthermore, we compare the performance of the proposed method with a few widely used cache algorithms:

*1) Random Access (RA):* the edge servers cache each requested video and discard it at random if the cache space is full.

*2) Least Recently Used (LRU):* the edge servers cache each requested video and discard the least recently used video in cache list if the cache space is full.

*3) Most Popular Content (MP):* the edge servers cache the most popular video in terms of viewing count.

*4) Adaptive Replacement cache (ARC):* the edge server cache the videos in both LRU and LFU manner. Also, it manage cache list by adaptively tuning the proportion of LRU and LFU [39].

*5) CSQCA-F:* The proposed video cache and adaptation framework but without partial caching design.

*6) CSQCA:* The proposed video cache and adaptation framework with complete design.

Note that CSQCA is the only one considering the partial cache scheme among the comparison.

### A. Simulation Setup

In the simulation, there are 2 edge servers corresponding to 2 different social communities, and each of them is equipped with cache space of 100 GB, video transcoding rate of 100 Mbps and connected to a regional server with backhaul capacity of 100 Mbps. Besides, there are 100,000 available videos in the content server with popularity following a Zipf distribution steepness of 0.56 [5]-[6],[29]. Also, the video has resolution option $\chi = \{12, 10, 5, 2, 1.5\}$ Mbps. Each user in community requests video obeying Poisson process with $\omega = 0.1$. As for wireless video delivery, the bandwidth $W$ is 10MHz and that we adopt the Rayleigh fading to describe the small-scale time-varying wireless channel condition. Furthermore, we set the interval of a time slot $T$ to be one hour and the order of ARMA model $q$ to be 5. Finally, the parameters for QoE model (16) are set with $(\alpha, \beta, \gamma, \delta) = (0.25, 0.25, 0.1, 0.4)$.

### B. Datasets Description

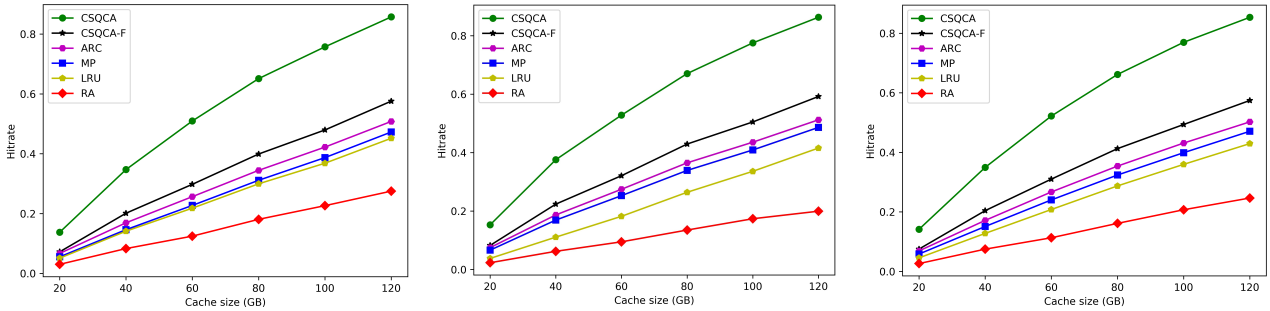
With an eye to further validating the performance of the proposed approach, we utilized 2 different real world datasets [40] to represent behaviors of 2 social communities in our simulation. Containing temporal interaction traces of users, these 2 datasets can match our scenario since all of the recorded users have geopolitical relationship with each other. To begin with, the dataset 1 contains chat messge of 1900 users collected from a Facebook-like platform available in UCI campus. On the other hand, the dataset 2 records email communication of 1005 users in a European research institution.

Specifically, the interaction pattern of users in these 2 datasets are quite different. To begin with, the interaction interval between users in dataset 2 are usually longer than that in dataset 1, resulting in slower video dissemination speed in dataset 2 than in dataset 1. Speaking of time-varying relationship of users, users in dataset 1 tend to have more connections but most of them vary with time rapidly. In average, the contact frequency of users in dataset 1 are lower than that in dataset 2. Thus, dataset 1 has higher potential coverage of video dissemination but has a lower sharing probability of each user. On the contrary, users in dataset 2 tend to communicate with few certain people but their connection are rather consistent. That is, their time-varying feature is insignificant. Thus, dataset 2 has low potential coverage of video dissemination and has high sharing probability of each user.

These 2 features reflect on their different video traffic patterns. We can observe from Fig. 4 that the rising rate of video requests in dataset 1 is higer than that in dataset 2 owing to different video dissemination speed. Then, we can observe that, due to different interaction pattern, video requests in dataset 1 rise dramatically in a short period of time while only some sporadic videos are requested at other times. On the other hand, the video requests from dataset 2 are rather uniform in time.

### C. Analysis of Caching Performance

*1) The impact of collaborative cache:* We first examine the effect of collaborative cache technique in our framework by comparing the network performance of CSQCA with that of non-collaborative CSQCA. In the non-collaborative version of the CSQCA, each edge server determines its own cache decision individually without orchestrating, which implies that




(a) Community 1 (b) Community 2 (c) Overall region

Fig. 6. Average hitrate versus cache size in different social communities.

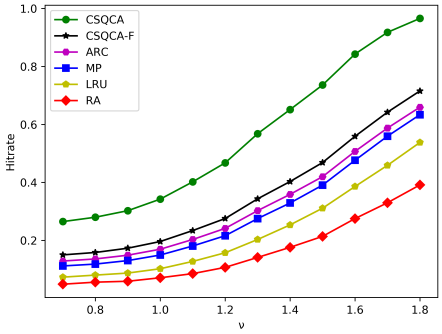



Fig. 7. Average cache hitrate with different Zipf steepness

the duplicate video cache may happen among different edge servers. Furthermore, we set the fully non-cache scheme as a baseline to illustrate the meaning of the value.

Fig. 5 presents the average backhaul traffic of different cache scheme. First, we can observe that the cache can reduce backhaul traffic by directly delivering the repeated video request to users. Then, it can be shown that the collaborative cache between multiple edge servers will significantly reduce the backhaul traffic compared with the non-collaborative scheme since the edge servers can share the cached contents with each other through mp3 interface. Nevertheless, the tradeoff is that the transmission process among edge servers slightly increases the initial latency of the end users whereas in non-collaborative scheme the contents can be delivery without further transmission.

*2) The impact of cache size:* Fig. 6. presents the hitrate of different algorithm with respect to the cache size in different communities. First, it is obvious that CSQCA's hitrate is much high than CSQCA-F since CSQCA-F stores entire video while CSQCA only cache part of a video so that it has more space to cache other videos. The trade-off is that though partial cache can hit more video requests, the QoE of each partially cached video may degrade compared with entirely cached video.

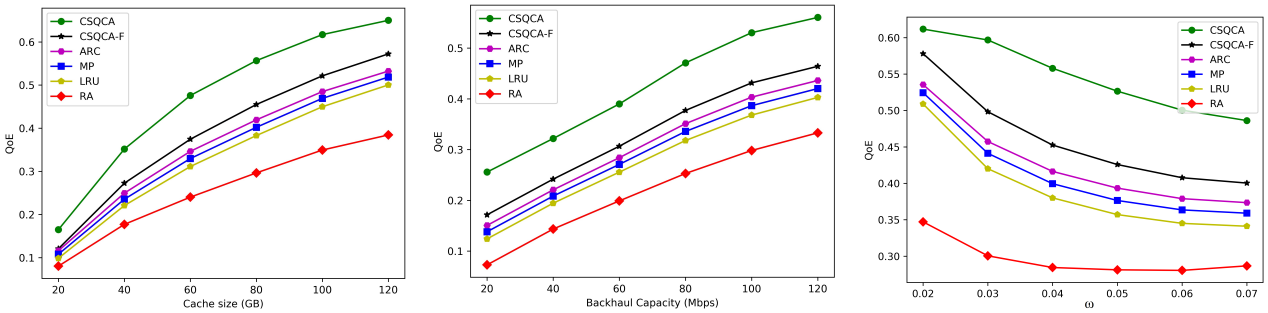
On the other hand, we can see that the MP method slight outperforms LRU in each community. This is mainly because the popularity of each video remains constant with Zipf 0.56 in our simulation. Thus, MP can capture what the truly popular video is in long term while LRU's performance will be affected by different traffic patterns since its cache decision depend on access time of videos. Also, we can observe that the performance of LRU in Fig. 6(a) is lower than that in Fig. 6(b), which can also be results from different interaction pattern of 2 different datasets. Since the video dissemination speed is relatively low in dataset 2, some videos may be removed from the cache list of LRU before it gains popularity through sharing among users, resulting lower hitrate. Similarly, the performance of ARC slightly degrades in dataset 2 because it also consider the time factor in its cache decision.

Finally, Fig. 6(c) depicts the overall hitrate among the whole region. We can see that the curve is approximated to the mixed of the curves of 2 different community in the region. Generally, the performance of CSQCA is better than the other method in both communityies, since CSQCA determine its cache decision by considering video dissemination process in community, instead of access time or viewing count that may vary with different video requests patterns.

*3) The impact of Zipf steepness:* Fig. 7 depicts the hitrate of different algorithm with the change of the Zipf steepness. Since a larger $\nu$ implies a steeper request probability distribution where the majority of requests concentrate on a small number of videos, we can observe that the hitrate grows rapidly with steepness initially. Nevertheless, the rising rate of hitrate slows down when steepness is greater than 1 and it will become saturated ultimately. This can be stem from the definition for Zipf distribution as (4), which is a normalized exponential function. From the feature of the exponential function, it is obvious that the value of hitrate grows differently before and after $\nu = 1$.

### D. Analysis of QoE Performance

*1) The impact of cache size:* Fig. 8(a) shows the QoE performance of different algorithms under different cache sizes. It can be seen that generally, QoE increases with the cache size since a lager cache size results in higher hitrate, ending up increasing the average QoE. However, it also can be observed that the difference in QoE of CSQCA and the other methods is not as high as the difference in hitrate in Fig. 6. This is mainly because CSQCA only caches parts of video, which may increase the incidence of video stalling of each cached video, degrading average QoE of cached videos while the other methods cache entire video. Eventually, it can be shown that after considering tradeoff between hitrate and




(a) Cache storage size (b) Backhaul capacity (c) Poisson distribution parameter

Fig. 8. Average QoE with different network setups

QoE per cached video, the overall QoE of CSQCA is higher than CSQCA-F and the other methods.

*2) The impact of backhual capacity:* Here, we resolve to examine the effect of backhaul capacity toward the QoE. We can first observe from Fig. 8(b) that the value of QoE gradually increases with backhaul capacity and ends up being saturated. This can be extrapolated from the process of streaming: the higher download rate from the content server through backhaul network, the less possible the processing buffer of edge servers is to get empty and the QoE is not be degraded by experiencing stalling events. Also, as the download rate goes high, the average video quality can increase accordingly.

In addition, it can be noted that the difference in QoE of all compared method remain almost constant throughout the figure. The reason is that their cache size are fixed. Thus, their QoE performance directly depends on their different hitrate, which also remain constant.

*3) The impact of poisson distribution parameter:* Finally, we examine on the effect of different poisson distribution parameter $\omega$. In Fig. 8(c), we can observe that the QoE decreases when the value of $\omega$ goes high. This can result from the fact that the higher $\omega$ causes more video requests. Since the backhaul capacity is limited, the average QoE degrades accordingly if the number of requests increases. However, it can be noted that the decreasing rate of CSQCA is much lower than the other methods since it has a higher hitrate, which can reduce more identical traffic. Thus, our proposed CSQCQ is less affected by the higher value of $\omega$ compared with the other methods.

## VII. Conclusions

In this paper, a novel collaborative social-aware QoE-driven video caching and adaption (CSQCA) framework is proposed. We investigate the interactions among users and video dissemination process in social network to determine collaborative cache strategy in multiple edge servers. Furthermore, in order to provide better QoE of streaming service in edge network, the edge servers are designed to dynamically transcode the cached video chunk to an appropriate resolution. Finally, we adopt real world datasets to conduct our simulation, which reveals that CSQCA offers much better performance in average hit ratio and reduction in backhaul traffic than traditional cache algorithms. The experimental results also illustrate that CSQCA can provide better QoE in different datasets under different systematic factors.

## References

[1] Cisco Virtual Networking Index: "Global Mobile Data Traffic Forcast Updatae, 2016-2021," San Jose, CA, USA.

[2] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal et al., "Mobile-edge computing introductory technical white paper," White Paper, Mobile-edge Computing (MEC) industry initiative, 2014.

[3] 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, April 2016.

[4] S. Han, H. Su, C. Yang and A. F. Molisch, "Proactive Edge Caching for Video on Demand With Quality Adaptation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 218-234, Jan. 2020.

[5] A. Mehrabi, M. Siekkinen and A. Ylä-Jääski, "QoE-Traffic Optimization Through Collaborative dge Caching in Adaptive Mobile Video Streaming," *IEEE Access*, vol. 6, pp. 52261-52276, 2018.

[6] C. Li, L. Toni, J. Zou, H. Xiong and P. Frossard, "QoE-Driven Mobile Edge Caching Placement for Adaptive Video Streaming," *IEEE Trans. Multimedia*, vol. 20, no. 4, pp. 965-984, April 2018.

[7] W. Zhang, Y. Wen, Z. Chen, and A. Khisti, "QoE-Driven Cache Management for HTTP Adaptive Bit Rate Streaming Over Wireless Networks," *IEEE Trans. Multimedia*, vol. 15, no. 6, Oct. 2013.

[8] J. Dai, F. Liu, B. Li, B. Li and J. Liu, "Collaborative Caching in Wireless Video Streaming Through Resource Auctions," *J. Sel. Areas Commun*, vol. 30, no. 2, pp. 458-466, Feb. 2012.

[9] F. Liu, B. Li, B. Li and H. Jin, "Peer-Assisted On-Demand Streaming: Characterizing Demands and Optimizing Supplies," *IEEE Trans. Comput.*, vol. 62, no. 2, pp. 351-361, Feb. 2013.

[10] M. F. Tuysuz and M. E. Aydin, "QoE-based Mobility-aware Collaborative Video Streaming on the Edge of 5G,"*IEEE Trans. Ind. Informat.*, to be published.

[11] H. Hsu and K. Chen, "Optimal caching time for epidemic content dissemination in mobile social networks," *IEEE International Conference on Communications*, Kuala Lumpur, pp. 1-6, 2016.

[12] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Watch global, cache local: YouTube network traffic at a campus network - Measurements and implications" *Proc. ACM Multimedia Comput. Netw*, 2008.

[13] B. Nie, H. Zhang and Y. Liu, "Social interaction based video recommendation: Recommending YouTube videos to facebook users," *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 97-102, 2014.

[14] Y. Wang, M. Ding, Z. Chen and L. Luo, "Caching Placement with Recommendation Systems for Cache-Enabled Mobile Social Networks," *IEEE Communications Letters*, vol. 21, no. 10, pp. 2266-2269, Oct. 2017.

[15] N. Zhang, J. Guan, C. Xu and H. Zhang, "A dynamic social content caching under user mobility pattern," *International Wireless Communications and Mobile Computing Conference*, Nicosia, pp. 1136-1141, 2014.

[16] H. Zhou, L. Xu and J. Zhang, "A Physical-social-based Group Utility Maximization framework for Cooperative Caching in Mobile Networks," *International Conference on Wireless Communications and Signal Processing*, Hangzhou, pp. 1-7, 2018.

[17] B. Wang, Y. Sun, S. Li, Q. Cao, Y. Chen and J. Xu, "Hierarchical Matching with Peer Effect for Latency-Aware Caching in Social IoT," *IEEE International Conference on Smart Internet of Things*, pp. 255-262, Xi'an, 2018.

[18] X. Wang, S. Leng and K. Yang, "Social-Aware Edge Caching in Fog Radio Access Networks," *IEEE Access*, vol. 5, pp. 8492-8501, 2017.

[19] S. He, H. Tian and X. Lyu, "Edge Popularity Prediction Based on Social-Driven Propagation Dynamics," *IEEE Communications Letters*, vol. 21, no. 5, pp. 1027-1030, May 2017.

[20] X. Xu, J. Liu, and X. Tao, "Mobile Edge Computing Enhanced Adaptive Bitrate Video Delivery With Joint Cache and Radio Resource Allocation," *IEEE Access*, vol. 5, Aug. 2017.

[21] T. X. Tran and D. Pompili, "Adaptive Bitrate Video Caching and Processing in Mobile-Edge Computing Networks," *IEEE Trans. Mobile Comput.*, vol. 18, no. 9, pp. 1965-1978, 1 Sept. 2019.

[22] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. M. Leung, "Cache in the air: Exploiting content caching and delivery techniques for 5G systems," *IEEE Commun. Mag.*, Feb. 2014.

[23] T.-Y. Kan, Y. Chiang, and H.-Y. Wei, "Task Offloading and Resource Allocation in Mobile-Edge Computing System," *The 27th Wireless and Optical Communication Conference*, Hualien, Taiwan, Apr. 2018.

[24] T.-Y. Kan, Y. Chiang, and H.-Y. Wei, "QoS-aware Mobile Edge Computing System: Multi-server Multi-user Scenario," *IEEE GLOBECOM Workshops:Emerging Technologies for 5G and Beyond Wireless and Mobile Networks*, Abu Dhabi, Dec. 2018.

[25] B. Gao, Z. Zhou, F. Liu and F. Xu, "Winning at the Starting Line: Joint Network Selection and Service Placement for Mobile Edge Computing," *IEEE Conference on Computer Communications (INFOCOM)*, Paris, pp. 1459-1467, 2019.

[26] Q. Zhang, F. Liu and C. Zeng, "Adaptive Interference-Aware VNF Placement for Service-Customized 5G Network Slices," *IEEE Conference on Computer Communications (INFOCOM)*, Paris, pp. 2449-2457, 2019.

[27] Y. Chiang, Y. Chao, C. Hsu, C. Chou and H. Wei, "Virtual Network Embedding With Dynamic Speed Switching Orchestration in Fog/Edge Network," *IEEE Access*, vol. 8, pp. 84753-84768, 2020.

[28] C. Ge, N. Wang, S. Skillman, G. Foster, and Y. Cao, "QoE-driven DASH video caching and adaptation at 5G mobile edge," *ACM Conference on Information-Centric Networking*, pp. 237–242. Sep, 2016.

[29] C. Liang, Y. He, F. R. Yu and N. Zhao, "Enhancing QoE-Aware Wireless Edge Caching With Software-Defined Wireless Networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 10, pp. 6912-6925, Oct. 2017.

[30] G. E. P. Box, G. M. Jenkins and G. C. Reinsel, "Time series analysis: Forecasting and control," Prentice-Hall, Englewood Cliffs, NJ, 1994.

[31] N. İlhan and Ş. G. Öğüdücü, "Predicting community evolution based on time series modeling," *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Paris, pp. 1509-1516, 2015.

[32] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn and S. Moon, "Analyzing the Video Popularity Characteristics of Large-Scale User Generated Content Systems," *IEEE/ACM Trans. Netw.*, vol. 17, no. 5, pp. 1357-1370, Oct. 2009.

[33] Y. Jin and Y. Wen, "PAINT: Partial in-network transcoding for adaptive streaming in information centric network," *IEEE 22nd International Symposium of Quality of Service (IWQoS)*, 2014.

[34] F. Liu, B. Li, L. Zhong, B. Li, H. Jin and X. Liao, "Flash Crowd in P2P Live Streaming Systems: Fundamental Characteristics and Design Implications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 7, pp. 1227-1239, July 2012.

[35] H. Mao, R. Netravali, and M. Alizadeh, "Neural Adaptive Video Streaming with Pensieve," *ACM Conference on Special Interest Group on Data Communication (SIGCOMM '17)*, 2017.

[36] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP," *ACM Conference on Special Interest Group on Data Communication (SIGCOMM '15),* 2015.

[37] J. Kua, G. Armitage and P. Branch, "A Survey of Rate Adaptation Techniques for Dynamic Adaptive Streaming Over HTTP," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1842-1866, thirdquarter 2017.

[38] G. Neglia, D. Carra and P. Michiardi, "Cache Policies for Linear Utility Maximization," *IEEE/ACM Trans. Netw.*, vol. 26, no. 1, pp. 302-313, Feb. 2018.

[39] N. Megiddo and D.S, Modha, "ARC: A Self-Tuning, Low Overhead Replacement Cache," FAST. Vol. 3. No. 2003.

[40] Stanford Large Network Dataset Collection [Online].Available: http://snap.stanford.edu/data/