Wireless Video Caching and Dynamic Streaming Under Differentiated Quality Requirements

Minseok Choi, *Member, IEEE*, Joongheon Kim, *Senior Member, IEEE*, and Jaekyun Moon, *Fellow, IEEE*

***Abstract*— This paper considers one-hop device-to-device-assisted wireless caching networks that cache video files of varying quality levels, with the assumption that the base station can control the video quality but cache-enabled devices cannot. Two problems arise in such a caching network: *file placement problem* and *node association problem*. This paper suggests a method to cache videos of different qualities, and thus of varying file sizes, by maximizing the sum of video quality measures that users can enjoy. There exists an interesting tradeoff between video quality and video diversity, i.e., the ability to provision diverse video files. By caching high-quality files, the cache-enabled devices can provide high-quality video, but cannot cache a variety of files. Conversely, when the device caches various files, it cannot provide a good quality for file-requesting users. In addition, when multiple devices cache the same file but their qualities are different, advanced node association is required for file delivery. This paper proposes a node association algorithm that maximizes time-averaged video quality for multiple users under a playback delay constraint. In this algorithm, we also consider *request collision*, the situation where several users request files from the same device at the same time, and we propose two ways to cope with the collision: scheduling of one user and non-orthogonal multiple access. Simulation results verify that the proposed caching method and the node association algorithm work reliably.**

***Index Terms*— Wireless caching network, D2D communication, video streaming, caching policy, node association.**

I. Introduction

EXCEEDINGLY large amounts of data traffic generated by rapidly growing wireless mobile devices in recent years have created formidable challenges for wireless communication. Within just a few years, it is expected that tens of exabytes of global data traffic be handled on daily basis with on-demand video streaming services accounting for about 70% of them [1].

On-demand video streaming is characterized by a relatively small number of popular contents being requested at ultra high rates; as such playback delay is often the more important measure of goodness to the user than other typical performance metrics like video quality [2]. In this regard, the wireless caching technology as discussed in [3] and [4], wherein the base station (BS) pushes popular contents for off-load time to cache-enabled nodes with limited storage spaces so that these nodes provide popular contents directly to nearby mobile users, is advantageous for video streaming services. By caching popular files on cache-enabled nodes, there is no need to repeatedly receive files from the BS every time users request.

Caching popular contents on the finite storage of the helper node near mobile users, which acts like a small BS, has been proposed to reduce latency in file transmission [6]. Further, a device-to-device (D2D)-assisted caching network has been studied [7]–[10], where mobile devices can store popular contents and directly respond to the file requests of neighboring users. In the wireless caching network, there are two main issues: 1) *file placement problem* - how to cache the popular contents at the caching nodes, e.g., caching helpers or cache-enabled devices, and 2) *node association problem* - which caching node is optimal to deliver the requested file to the user for providing smooth video streaming services.

Video files can be encoded to multiple versions which differ in the quality level, e.g., peak-signal-to-noise-ratio (PSNR) or spatial resolution [11], [12]. Since the file size of video varies by quality, it is also important in caching network to determine which file of what quality is stored in the caching node (*file placement problem*) and what quality of video is requested from which caching node by the streaming user (*node association problem*) [13].

The goal of the file placement problem is to find the optimal caching policy according to popularity distribution of contents and network topology. There have been some research efforts to find the optimal caching policy in stochastic wireless caching networks [14]–[16], but contents with different quality levels were not considered. Traditionally, caching strategies for videos with various qualities have been researched with radio access network (RAN) caches which enable transcoding or transrating of video files [17]–[19]. However, deployments of the transcoder in mobile devices are inefficient, thus it is reasonable that only the video file of certain quality pushed by the BS for off-load time can be delivered by cache-enabled devices.

Due to the finite storage size of caching devices, there exists a trade-off between video quality and video diversity, i.e., if the device wants to cache the high-quality files, it cannot store many types of videos. Jarray and Giovanidis [20] consider caching files of different sizes, but they assume that the different-sized files account for the same unit of cache storage,

Manuscript received December 11, 2017; revised April 17, 2018; accepted April 18, 2018. Date of publication June 7, 2018; date of current version September 12, 2018. This work was supported in part by the National Research Foundation of Korea under Grant 2016R1A2B4011298 and Grant 2017R1A4A1015675. *(Corresponding author: Joongheon Kim.)*

M. Choi and J. Moon are with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon 34141, South Korea (e-mail: ejaqmf@kaist.ac.kr; jmoon@kaist.edu).

J. Kim is with the School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, South Korea (e-mail: joongheon@cau.ac.kr).

Digital Object Identifier 10.1109/JSAC.2018.2844980

thus it does not reflect the above trade-off. Many researchers have proposed the static file placement policies under the consideration of differentiated quality requests for the same file, given probabilistic quality requests [12], [22], [23] or minimum quality requirements [21]. In [24], joint optimization of the static file placement and routing is proposed. Further, the probabilistic caching policy for video files of various quality levels is presented in [25] by using stochastic geometry, given the user preference for quality level.

The node association problem for video delivery in wireless caching networks has been also extensively researched. In most of the research works that do not consider different quality levels for the same file, the file-requesting user is allowed to receive the content from the caching node under the strongest channel condition [15], [26]. Node associations for video delivery in heterogeneous caching networks have been studied in [27]–[29]. Especially, dynamic video streaming allows each chunk, which consists of the whole video file and occupies a part of playback time, to have a different quality depending on time-varying network conditions [36]. There are some research results addressing the transmission scheme which provides the video by dynamically selecting the quality level [33], [34] or the scheduling policy that maximizes a network utility function of time-averaged video quality in a network with caching helpers [35]. While the video delivery polices of [33]–[35] are operated at the BS side, however, decisions of video delivery requests at user sides have been largely neglected. This scenario is consistent with the practical real-world software implementation of dynamic adaptive streaming over HTTP (DASH) [36], in which users dynamically choose the most appropriate video quality.

In this paper, we consider the stochastic D2D-assisted caching network for dynamic video streaming services. For *file placement problem*, each BS has all video files and is equipped with a quality controller, which controls the video quality. However, deploying the video quality controller in small mobile devices is not desirable; we assume in the present paper that the BS pushes the video files with certain quality levels and the cache-enabled devices can provide given video quality measures to mobile users who request the cached files.

For *node association problem*, users dynamically request different quality levels for the same video and associate with one of the neighboring nodes caching the file of desired quality. Assuming that delivered video chunks are waiting for playback in the user queue, the quality level of the next chunk should be chosen at the user side depending on user's channel condition and queue state to avoid playback delay, which is different from the assumption made in [35]. Depending on the desired quality, node association is updated for video delivery on chunk-by-chunk basis over the playtime.

In this paper, file placement and node association operate on different time scales, unlike [37], which jointly optimizes caching and transmission policies at the BS side. In general, the BS pushes popular contents to caching nodes for off-load time, and users request video files after file placement is completed. In addition, file popularity does not change as rapidly as dynamic changes of quality requests during video streaming, so file placement and node association are independently considered in this paper.

The main contributions of this paper can be summarized as follows:

- This paper proposes the probabilistic caching policy for video files of varying quality levels by maximizing the successfully enjoyable video quality sum. Since the streaming user dynamically requests the quality level of video, the expected quality of video which can be reliably delivered to the user, i.e., successfully enjoyable video quality summation, is a reasonable metric. We derive the closed-form caching probabilities for every video file of every quality level. The trade-off between video quality and video diversity is reflected in the proposed caching placement policy.
- This paper models the node association cases when video files of different quality levels are stored in cache-enabled devices. We specify the cases which require an advanced node association scheme to carefully choose the cache-enabled device for video delivery with desired quality.
- This paper proposes a node association algorithm for file-requesting users to choose the appropriate quality and to associate with the device which caches the requested file of desired video quality. The proposed algorithm maximizes the sum of the time-averaged quality measures of all users while avoiding playback delay in streaming communications. In this paper, playback delay is interpreted based on the user queue model, and the algorithm aims at avoiding playback delay by preventing queue emptiness. Simply, when there is no video chunk in the user queue, the user has to wait for the next chunk and video playback is inevitably delayed. Compared to pursuing only quality and only preventing playback latency, numerical results show that the proposed algorithm allows video chunks to be stacked in queue enough to maintain smooth video playback, while pursuing high video quality.
- We provide two ways to handle *request collision*, which occurs when multiple users request video files simultaneously from the same cache-enabled device. One is to schedule one of the file-requesting users for video delivery. Another method is utilization of NOMA to serve all file-requesting users at the expense of data rate degradation. In the proposed algorithm, a scheduling scheme maximizes the time-average video quality for the given user while preventing playback delay.

The rest of the paper is organized as follows. The D2D-assisted caching network model with different-quality video files is given in Section II. Caching policy for video files of various quality levels is proposed in Section III. Node association cases with different-quality video files and the node association algorithm are presented in Section IV. Simulation results are shown in Section V and Section VI concludes the paper.

II. D2D Caching Network Model With Different-Quality Video Files

This paper considers a cellular model where some cache-enabled devices exist and N users enjoy video streaming

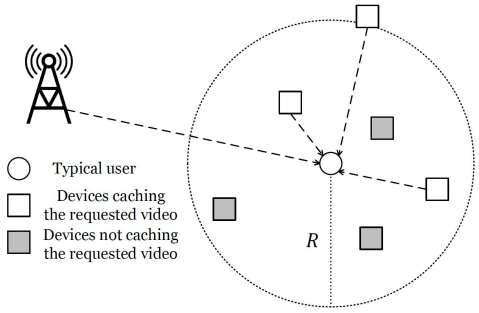

Fig. 1. D2D caching network model.

services. When certain user n requests a particular video file, she searches through the device candidates that cache the requested file within a radius of R, as shown in Fig. 1. User n selects one of the candidates for file delivery. If there is no device caching the requested file within the radius R from user n, the BS can transmit the desired file via a cellular link. Since the caching devices are usually much closer to the file-requesting users than the BS, the users are assumed to prefer downloading the file from the caching devices rather than directly from the BS, due to transmission delay. Therefore, direct transmission from the BS is not considered in this paper.

There is a file library $\mathcal{F}$ and each file $i \in \mathcal{F}$ has a popularity probability f_i, which follows the Zipf distribution [10]: $f_i = i^{-\gamma} / \sum_{j=1}^{F} j^{-\gamma}$ where γ denotes the popularity distribution skewness. Let i_n be the index of the file requested by user n. Assume that all files have Q quality levels. Suppose that there is no quality controller in cache-enabled devices, so devices can only transmit video files of the fixed quality which the BS pushes. In this case, user n can choose the quality level of the receiving video file; let q_n denote the desired quality level of file i_n. The file size varies with video quality, and let M_q be the normalized file size of quality level q for every video. Each cache-enabled device has a limited storage size of M.

The cache-enabled devices are modeled using the independent Poisson point processes (PPPs) with intensity λ. This paper utilizes the probabilistic caching placement method [14] for cache-enabled devices to cache file i of quality q with probability $p_{i,q}$. Let $\lambda p_{i,q}$ be the intensity of the independent PPPs for the devices caching file i of quality level q. Suppose that the system does not allow any additional D2D link within the radius R of the user who is already downloading the file from certain cache-enabled device. By taking R sufficiently large and/or exploiting orthogonal resources for each D2D coverage, the system can guarantee the negligible interference among multiple D2D links. When an additional user requests a video file within the coverage, the user should download the file from the BS via the cellular link.

The Rayleigh fading channel is assumed for the communication links from the users to the cache-enabled devices. Denote the channel with $h = \sqrt{L}g$, where $L = 1/l^2$ controls slow fading with l being the user-device distance and g represents the fast fading component having a complex Gaussian distribution, $g \sim CN(0,1)$.

The main research issues in the entire wireless caching network can be largely classified as follows:

- *File placement problem*: When the BS pushes video files to cache-enabled devices for off-load time, the BS determines which file of which quality level is cached in each cache-enabled device. This paper chooses the probabilistic caching placement method [14] for cache-enabled devices to cache file i of quality q with probability of $p_{i,q}$.
- *Node association problem*: Each file-requesting user should find the candidate set of devices caching the requested video first. Next, each user chooses one of the candidate devices for file delivery. A careful choice of the device to be associated with is important to ensure good video quality and smooth playback without delay.
- *Request collision*: When multiple users request video files from the same device, we say *request collision* occurs. In this instance, the device should determine how to serve those users. One way is to deliver the requested file to only one user, expecting that each of the rest of the users finds another cache-enabled device to request video files. The other method is NOMA, which serves multiple users in the same time/frequency/code simultaneously, but a transmission rate reduction is inevitable.

III. Caching Policy for Different-Quality Video Files

A. Probabilistic Caching With Different Quality Video Files

As mentioned earlier, the file placement problem in this paper is based on the probabilistic caching method [14], where the file is independently placed in devices according to the same distribution. Since we consider video files of different sizes, however, a certain modification of the probabilistic placement policy of [14] is necessary. As in [14], we also start with M continuous memory intervals of unit length, and then place all files of all quality levels one by one to fill the M unit-length intervals with every $p_{i,q}$. The main difference from the approach of [14] is that the file of quality level q occupies a vertical size of M_q. Accordingly, we need to impose the following constraints:

$$\sum_{i=1}^{F} \sum_{q=1}^{Q} M_q p_{i,q} \leq M \tag{1}$$

$$0 \leq p_{i,q} \leq 1, \quad \forall i \in \mathcal{F}, \ \forall q \in \mathcal{Q}. \tag{2}$$

The constraint (2) is obvious, and the constraint (1) is necessary and sufficient for the existence of a random file placement policy requiring no more storage than M. The sufficiency of (1) is proven by obtaining the caching policy requiring no more storage than M in the following sections (see Table I). The necessity of (1) can be also proven by establishing that the left-hand side of (1) is equal to the expected required memory size of the caching device, similar to Fact 1 in [14]. In addition, if the device caches file i of quality q_1, then the same file of another quality level, say q_2, is better not to be cached in the device [29]. However, it is not necessary to prevent caching copies of the same file with different qualities on a device for obtaining the caching policy.

Fig. 2 gives an example of the probabilistic caching method with files of different sizes where $F = 5$, $M = 6$, $Q = 3$,

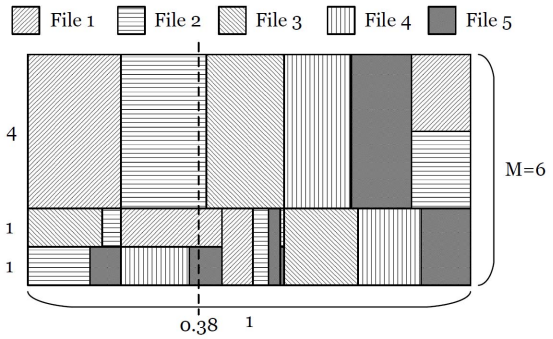

Fig. 2. An example of probabilistic caching method with the files of different quality levels.

$M_1 = 1$, $M_2 = 2$, and $M_3 = 4$. This example satisfies the equality of (1). As mentioned before, there are three kinds of blocks with vertical sizes $M_1 = 1$, $M_2 = 2$ and $M_3 = 4$ for each file type. After obtaining $p_{i,q}$, we have to build a $M \times 1$ rectangle consisting of $F \times Q$ rectangles with heights of M_q and widths of $p_{i,q}$ for all $i \in \mathcal{F}$ and $q \in \mathcal{Q}$, as shown in Fig. 2. Each element rectangle corresponds to the file of certain quality. The cache-enabled device generates uniformly a random number within $[0, 1]$, and draws a vertical line. Finally, the device stores files which the vertical line goes through. In the example of Fig. 2, assuming we draw a vertical line at 0.38, the device stores File 2 of quality level 3, File 1 of quality level 1, and File 5 of quality level 1.

Remark: Caching different-quality/different-size files would make storage inefficient in some system environments. For example, consider the cases of $M_1 = 1$, $M_2 = 3$, $M_3 = 4$, and $M = 6$. In this case, a device cannot store two files of quality levels 2 and 3. The only possible combinations here are: caching one of quality 2 and three files of quality 1, and caching one of quality 3 and two files of quality 1. It is highly likely that the placements of $F \times Q$ rectangles with heights of M_q and widths of $p_{i,q}$ do not fit perfectly in a $M \times 1$ rectangle in this scenario. This situation indicates that storage is not being used efficiently. Therefore, the file sizes of different qualities and the maximum storage size of the device should be carefully considered for efficient caching. However, this does not mean that the proposed constraints (1) and (2) would not lead to random file placement policy of different qualities.

B. Optimal File Placement Rule

There still remains the important question: how to find the optimal $p_{i,q}$? Since we assume that multiple D2D links do not interfere with one another, we can refer to the file placement rule in the noise-limited network [15]. The differences here are the constraint of the probabilistic caching method (1) and the optimization metric used. The method of [15] maximizes the average file delivery success probability, but since we are concerned with the video quality, the successfully enjoyable video quality sum is chosen as the performance metric. The successfully enjoyable video quality sum is defined as

$$\sum_{i=1}^{F} f_i \sum_{q=1}^{Q} \mathcal{P}(q) \cdot P\{R_{i,q} \geq \rho_{i,q}\}, \tag{3}$$

where $\mathcal{P}(q)$ is the measure of the quality q, $R_{i,q}$ is the data rate of the user to download file i of quality q, and $\rho_{i,q}$ is the threshold of the data rate for reliable transmission of file i of quality q. Denoting the Rayleigh fading channel from the user to the associated device for downloading file i of quality q by $h_{i,q}$, the data rate of the user for downloading file i of quality q in the noise-limited environment is given by

$$R_{i,q} = \mathcal{B} \log_2 \left(1 + \frac{|h_{i,q}|^2}{\sigma^2}\right), \tag{4}$$

where $\mathcal{B}$ is the bandwidth, assuming a unit transmit power and a normalized noise variance of σ^2. If the user desires the file i of quality q and there are multiple device candidates caching file i of quality q, it is reasonable for the user to download the file from the device whose channel condition is the strongest among the candidates.

Since the channel power $|h_{i,q}|^2$ follows the chi-squared distribution, i.e., Nakagami-1 fading channel, according to [15], the reliable transmission probability can be obtained by

$$P\{R_{i,q} \geq \rho_{i,q}\} = 1 - \exp\Big\{ - \frac{\kappa p_{i,q}}{\sigma^2 (2^{\rho_{i,q}/\mathcal{B}} - 1)}\Big\}, \tag{5}$$

where $\kappa = \pi \lambda \Gamma(2)$.

Thus, we can formulate the optimization problem to find the optimal caching probabilities:

$$\{p^*_{i,q}\} = \arg\max_{\{p_{i,q}\}} \sum_{i=1}^{F} f_i \sum_{q=1}^{Q} \mathcal{P}(q) \cdot P\{R_{i,q} \geq \rho_{i,q}\} \tag{6}$$

$$= \arg\min_{\{p_{i,q}\}} \sum_{i=1}^{F} f_i \sum_{q=1}^{Q} \mathcal{P}(q) e^{-C_{i,q} p_{i,q}} \tag{7}$$

$$\text{s.t.} \sum_{i=1}^{F} \sum_{q=1}^{Q} M_q p_{i,q} \leq M \tag{8}$$

$$0 \leq p_{i,q} \leq 1, \quad \forall i \in \mathcal{F}, \ \forall q \in \mathcal{Q} \tag{9}$$

where $C_{i,q} = \frac{\kappa}{\sigma^2 (2^{\rho_{i,q}/\mathcal{B}} - 1)}$. Since $\frac{d^2}{d^2\, p_{i,q}} \{e^{-C_{i,q} p_{i,q}}\} \geq 0$ and the objective function in (7) is the weighted function of $e^{-C_{i,q} p_{i,q}}$, the optimization problem (7) is convex.

The Lagrangian function of the objective (7) is given by

$$\begin{aligned} \mathcal{L}(\{p_{i,q}, \mu_{i,q}\}, \nu) = & \sum_{i=1}^{F} f_i \sum_{q=1}^{Q} \mathcal{P}(q) \cdot e^{-C_{i,q} p_{i,q}} \\ & + \nu \Big(\sum_{i=1}^{F} \sum_{q=1}^{Q} M_q p_{i,q} - M \Big) \\ & + \sum_{i=1}^{F} \sum_{q=1}^{Q} \mu_{i,q} (p_{i,q} - 1), \end{aligned} \tag{10}$$

and the derivative of (10) with respect to $p_{i,q}$, is

$$\frac{\partial \mathcal{L}(\{p_{i,q}, \mu_{i,q}\}, \nu)}{\partial p_{i,q}} = -f_i \mathcal{P}(q) C_{i,q} e^{-C_{i,q} p_{i,q}} + \nu M_q + \mu_{i,q}, \tag{11}$$

where ν and $\mu_{i,q}$ are the nonnegative Lagrangian multipliers. Then, the Karush-Kuhn-Tucker (KKT) conditions for the optimization problem (7) are given by

$$\frac{\partial \mathcal{L}(\{p_{i,q}, \mu_{i,q}\}, \nu)}{\partial p_{i,q}} = 0 \tag{12}$$

$$\nu(\sum_{i=1}^{F}\sum_{q=1}^{Q} M_q p_{i,q} - M) = 0 \tag{13}$$

$$\mu_{i,q}(p_{i,q} - 1) = 0, \tag{14}$$

(8)-(9), and $\mu_{i,q}, \nu \geq 0$ for all $i \in \mathcal{F}$ and $q \in \mathcal{Q}$.

From (12), we can obtain the optimal caching probabilities:

$$p^*_{i,q} = \frac{1}{C_{i,q}}\{\ln(f_i \mathcal{P}(q) C_{i,q}) - \ln(\nu M_q + \mu_{i,q})\}, \quad \forall i, \ \forall q. \tag{15}$$

We can easily note from (15) that the better the quality of the video file, the higher the probability of being stored in the device. On the other hand, larger file size of the higher-quality video makes caching probability smaller and decreases video diversity. Thus, the trade-off between video quality and video diversity is observed in (15). This trade-off depends on the constant value, $C_{i,q}$, and Lagrangian multipliers, ν and $\mu_{i,q}$.

The next step is to find the Lagrangian multipliers. We can determine the intervals of the Lagrangian multipliers by categorizing the caching probability value into three cases. First, when $p_{i,q} = 0$, $\mu_{i,q} = 0$ because of (14). To satisfy (12), $v = \frac{f_i \mathcal{P}(q) C_{i,q}}{M_q}$, but it is impossible because f_i, $C_{i,q}$, and M_q are different for i and q. We can set $\nu = \max_{i,q} \frac{f_i \mathcal{P}(q) C_{i,q}}{M_q}$ to guarantee $p^*_{i,q} \geq 0$. Therefore,

$$\nu \geq \frac{f_i \mathcal{P}(q) C_{i,q}}{M_q}, \text{ if } p_{i,q} = 0. \tag{16}$$

When $0 < p_{i,q} < 1$, $\mu_{i,q} = 0$ also, and $\nu = \frac{f_i \mathcal{P}(q) C_{i,q}}{M_q} e^{-\kappa p_{i,q} C_{i,q}}$ is obtained for (12). Therefore,

$$\frac{f_i \mathcal{P}(q) C_{i,q}}{M_q} e^{-C_{i,q}} < \nu < \frac{f_i \mathcal{P}(q) C_{i,q}}{M_q}, \text{ if } 0 < p_{i,q} < 1. \tag{17}$$

Finally, when $p_{i,q} = 1$, if $\nu = 0$, $\mu_{i,q} = f_i \mathcal{P}(q) C_{i,q} e^{-C_{i,q}}$, otherwise, $\mu_{i,q} = f_i \mathcal{P}(q) C_{i,q} e^{-C_{i,q}} - \nu M_q$, according to (15). To satisfy $\mu_{i,q} \geq 0$,

$$\nu \leq \frac{f_i \mathcal{P}(q) C_{i,q}}{M_q} e^{-C_{i,q}}, \text{ if } p_{i,q} = 1. \tag{18}$$

From (16)-(18), we can realize that $p_{i,q}$ and $\mu_{i,q}$ are functions of ν, so we only need to find the optimal value of ν to obtain the optimal caching probabilities. If $\nu \leq \min\{\frac{f_i \mathcal{P}(q) C_{i,q}}{M_q} e^{-C_{i,q}}, \ \forall i \in \mathcal{F}, \ \forall q \in \mathcal{Q}\}$, $p_{i,q} = 1, \ \forall i \in \mathcal{F}, \ \forall q \in \mathcal{Q}$. Therefore,

$$\sum_{i=1}^{F}\sum_{q=1}^{Q} M_q p_{i,q} = F \cdot \sum_{q=1}^{Q} M_q, \quad \text{if } \nu \leq \min\left\{\frac{f_i \mathcal{P}(q) C_{i,q}}{M_q} e^{-C_{i,q}}, \quad \forall i, \ \forall q\right\}. \tag{19}$$

However, if $\nu \geq \max\{\frac{f_i \mathcal{P}(q) C_{i,q}}{M_q}, \ \forall i \in \mathcal{F}, \ \forall q \in \mathcal{Q}\}$, $p_{i,q} = 0, \ \forall i \in \mathcal{F}, \ \forall q \in \mathcal{Q}$, and

$$\sum_{i=1}^{F}\sum_{q=1}^{Q} M_q p_{i,q} = 0, \text{ if } \nu \leq \max\left\{\frac{f_i \mathcal{P}(q) C_{i,q}}{M_q}, \ \forall i, \ \forall q\right\}. \tag{20}$$

Thus, if $\min\{\frac{f_i \mathcal{P}(q) C_{i,q}}{M_q} e^{-C_{i,q}}, \ \forall i, \ \forall q\} \leq \nu \leq \max\{\frac{f_i \mathcal{P}(q) C_{i,q}}{M_q}, \ \forall i, \ \forall q\}$,

$$0 \leq \sum_{i=1}^{F}\sum_{q=1}^{Q} M_q p_{i,q} \leq F \cdot \sum_{q=1}^{Q} M_q. \tag{21}$$

Assuming that $M < F \cdot \sum_{q=1}^{Q} M_q$, since $\sum_{i=1}^{F}\sum_{q=1}^{Q} M_q p_{i,q}$ is decreasing with ν, we can find the optimal ν^* and $p^*_{i,q}$ by the bi-section method. The details of the bi-section method for optimal file placement are shown in Algorithm 2.

Algorithm 1 Bisection Method for Optimal File Placement Rule

Initialize $\epsilon, \nu_- = \min\{l_{i,q}, \forall i \in \mathcal{F}, \forall q \in \mathcal{Q}\}$, $\nu_+ = \max\{u_{i,q}, \forall i \in \mathcal{F}, \forall q \in \mathcal{Q}\}$,
  and $p^*_{i,q} = -1, \forall i \in \mathcal{F}, \forall q \in \mathcal{Q}$ ▷ ϵ: error tolerance threshold
while $|\sum_{i=1}^{F}\sum_{q=1}^{Q} M_q p^*_{i,q} - M| \geq \epsilon$ **do**
  $\nu^* = (\nu_- + \nu_+)/2$
  $\mu^*_{i,q} = [f_i \mathcal{P}(q) C_{i,q} e^{-C_{i,q}} - \nu^* M_q]^+, \ \forall i \in \mathcal{F}, \ \forall q \in \mathcal{Q}$
  $p^*_{i,q} = \frac{1}{C_{i,q}}\left[\log_2(f_i \mathcal{P}(q) C_{i,q}) - \log_2(\nu^* M_q + \mu^*_{i,q})\right]^+, \ \forall i \in \mathcal{F}, \ \forall q \in \mathcal{Q}$
  if $\sum_{i=1}^{F}\sum_{q=1}^{Q} M_q p^*_{i,q} > M$ **then** $\nu_- \leftarrow \nu^*$
  else if $\sum_{i=1}^{F}\sum_{q=1}^{Q} M_q p^*_{i,q} < M$ **then** $\nu_+ \leftarrow \nu^*$
  end if
end while

IV. Node Association Maximizing Video Quality With Playback Delay Constraint

The node association problem in this paper amounts to choosing the cache-enabled devices for N users to request video files. After making the candidate set of devices which caches the requested file, the user has to choose the specific device as well as the level of quality. This paper proposes a dynamic algorithm for users to associate with cache-enabled devices to maximize time-average video quality measures with a playback delay constraint. Improvement in video playback latency can be explained based on the user queue model.

A. User Queue Model

A video file consists of many sequential chunks. User terminals receive video files from cache-enabled devices and process data for video streaming services in units of chunks. Each chunk of a file is responsible for some playback time of the entire stream. As long as all chunks are in correct sequence, each chunk can have different quality in dynamic streaming. Therefore, users can dynamically choose video

quality levels in every chunk processing time. By using the queue model, it can be said that the playback delay occurs when the chunk to be played does not yet arrive at the queue. In this sense, receiver queue dynamics collectively reflects the various factors which cause the playback delay.

In general, user queue models have their own arrival and departure processes. For each user $n \in \{1, \cdots, N\}$, the queue dynamics in each time slot $t \in \{0, 1, \cdots, \}$ can be represented as follows:

$$Q_n[t+1] = \max\{Q_n[t] - b_n[t], 0\} + a_n[t] \tag{22}$$

$$Q_n[0] = 0 \tag{23}$$

where $Q_n[t]$, $a_n[t]$, and $b_n[t]$ stand for the queue backlog, the arrival and departure processes of user n at time t, respectively. The queue states are updated and every user performs node association in each unit time slot t. In this paper, the interval of each slot is determined to be the channel coherence time, τ_c. Suppose a block fading channel, whose channel gain is static during the processing of multiple chunks, $t_c = m\tau$, where τ is a chunk processing time and m is the positive integer.

In this paper, queue backlog $Q_n[t]$ counts the number of video chunks in the queue. $a_n[t]$ and $b_n[t]$ semantically mean the numbers of received and processed chunks. Simply, m chunks are processed in each time slot, so $b_n[t] = m$. On the other hand, $a_n[t]$ obviously depends on the data rate of the communication link between user n and its associated device and the chunk size. The departure and arrival processes are given as follows:

$$a_n[t] = \left\lfloor \frac{R_n(\alpha_n(t), t) \cdot \tau_c}{L(q_n(\alpha_n(t), i_n), t)} \right\rfloor \tag{24}$$

$$b_n[t] = m \tag{25}$$

where $\alpha_n(t)$ denotes the cache-enabled device associated with user n at time t, and $q_n(\alpha_n(t), i_n)$ is the quality level of file i_n which user n requests from the device $\alpha_n(t)$. Also, $R_n(\alpha_n(t), t)$ and $L(q_n(\alpha_n(t), i_n), t)$ indicate the data rate of a D2D link between user n and the device $\alpha_n(t)$, and a chunk size of file i_n of the desired quality $q_n(\alpha_n(t), i_n)$ at time t, respectively. Some video chunks can be only partially delivered as the channel condition varies and node association is updated at every time slot t. Since partial chunk transmission is meaningless in our algorithm, the flooring is used in (24).

Let the Rayleigh fading channel between user n and device $\alpha_n(t)$ denoted by $h_{n,t}$. Then, the link rate between user n and device $\alpha_n(t)$ is simply given by

$$R_n(\alpha_n(t), t) = \mathcal{B} \log_2 \left(1 + \frac{|h_{n,t}|^2}{\sigma^2}\right). \tag{26}$$

For video streaming service, it is important to avoid playback delay. The user needs a chunk in the next sequence during video playback. If the next chunk has not yet arrived in the queue, there will be a delay in playback. Therefore, stacking enough queue backlogs, i.e., video chunks in sequence, is necessary for averting playback delay. Suppose that the queue is almost empty. In this case, the cache-enabled device whose channel is strong and which stores the requested file of low quality (i.e., small chunk size) is preferable for the user. On the other hand, when the queue is filled with a lot of video chunks, the user can request the high-quality video file without worrying about playback delay.

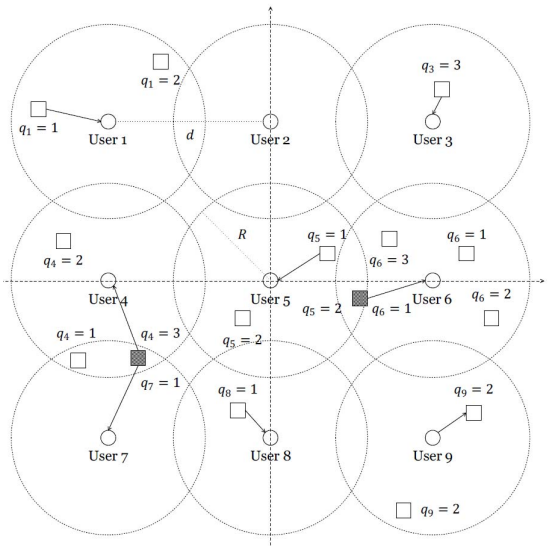

Fig. 3. Node association cases.

Remark: If τ_c is too long, it is better to update node associations more frequently than channel variations. For example, consider a user whose queue is filled with many chunks and thus is associated with the caching device delivering the chunks smaller than $b_n[t] = m$. If this situation persists for a long time, chunks in the queue will be emptied out soon and playback delay will occur, therefore several updates of node association are required over the time interval of τ_c. On the other hand, if τ_c is too short, the requested video cannot be successfully delivered even when the data rate of the link is good, because of the flooring in (24). Therefore, block fading is assumed with τ_c large enough for the users to receive the video chunks.

B. Node Association Cases

Depending on geological locations of cache-enabled devices, node association of certain user with an appropriate cache-enabled device can be classified into a number of cases. These example cases are illustrated in Fig. 3. Fig. 3 assumes that each user requests the video file from one of cache-enabled devices within radius R, and there are quality levels of 1, 2, and 3. Only the devices which cache the requested file are depicted in Fig. 3. The quality levels of requested videos in cache-enabled devices are written as $q_n = c,\ c \in \{1, 2, 3\}$, to indicate that the device caches the video of quality level c requested by user n. In particular, the devices which receive multiple file delivery requests are shown as the shaded squares. The proposed dynamic algorithm for node association can be applied to cases 4, 5, 6, and 7.

- Case 1: When there is no caching device which caches the requested file within radius R of the user, the user should download the video file from the BS via a cellular link. (user 2)

- Case 2: When there is only one device caching the requested file within radius R of the user, the user just downloads the video from this device, but only the fixed quality can be provided. If the user wants the high-quality file, it can download file from the BS but this option is not considered in this paper. (user 3, 8)
- Case 3: When there are multiple devices caching the requested files of the same quality within radius R of the user, the user requests the video from one of the devices whose channel is the strongest. (user 9) Similar to Case 2, only the fixed quality can be provided.
- Case 4: When there are multiple devices caching the requested files of different quality levels within radius R of the user, the proposed dynamic algorithm can be applied for node association. The proposed algorithm maximizes the expected video quality constrained on sufficiently large queue backlog to avoid playback delay. (user 1)
- Case 5: When the cache-enabled device receives two or more file delivery requests including the target user's, i.e., *request collision* occurs, the device serves multiple users by NOMA. (user 4, 7) The proposed algorithm determines to whether to exploit NOMA.
- Case 6: When the cache-enabled device receives two or more file delivery requests including the target user's, the proposed algorithm makes the device to schedule the target user and to ignore other requests. (user 6)
- Case 7: When the cache-enabled device receives two or more file delivery requests including the target user's, the proposed algorithm determines the device to schedule another user and to ignore the request of the target user. Then, the target user should find another cache-enabled device, and if there is no other device which stores the requested file within radius R , it has to download the file from the BS. (user 5)

C. Dynamic Node Association for Video File Delivery Under Queue Stability

We specifically go after the following optimization problem:

$$\max. \sum_{n\in\mathcal{N}} \mathbb{E}[\mathcal{P}(q_n\ (\alpha_n(t), i_n))] \tag{27}$$

$$\text{s.t. } \lim_{t\to\infty}\frac{1}{t}\sum_{t'=0}^{t-1}\mathbb{E}[Z_n[t']] < \infty, \quad \forall n \in \mathcal{N} \tag{28}$$

where $\mathcal{N}$ is the set of N file-requesting users via D2D links, and $Z_n[t] = \tilde{Q} - Q_n[t]$. The optimization metric (27) is the sum of the time averaged video quality measures of the file-requesting users as given by

$$\sum_{n\in\mathcal{N}}\Big[\lim_{t\to\infty}\frac{1}{t}\sum_{t'=0}^{t-1}\mathcal{P}(q_n(\alpha_n(t'), i_n))\Big]. \tag{29}$$

Here, $Z_n[t]$ is introduced to make $Q_n[t]$ large enough to avoid playback delay, and $\tilde{Q}$ is a sufficiently large parameter which affects the maximal queue backlog. From (22) and (23), the queue dynamics of $Z_n[t]$ can be represented as follows:

$$Z_n[t+1] = \min\{Z_n[t] + b_n[t], \tilde{Q}\} - a_n[t] \tag{30}$$

$$Z_n[0] = \tilde{Q}. \tag{31}$$

Even though the update rules of $Q_n[t]$ and $Z_n[t]$ are different, both queue dynamics mean the same video chunk processing. Therefore, playback delay due to emptiness of $Q_n[t]$ can be explained by queuing delay of $Z_n[t]$. By Little's theorem [38], the expected value of $Z_n[t]$ is proportional to the time-averaged queuing delay. Therefore, we hope to limit the queuing delay by addressing the constraint (28), and it is well known that Lyapunov optimization with the constraint (28) can make $Z_n[t]$ bounded [39].

Let $\mathbf{Z}[t]$ denote the column vector of $Z_n[t]$ of all users at time t, and define the quadratic Lyapunov function $L(\mathbf{Z}[t])$ as follows:

$$L(\mathbf{Z}[t]) = \frac{1}{N}\sum_{n\in\mathcal{N}}(Z_n[t])^2 \tag{32}$$

Then, let $\Delta\ (.)$ be a conditional quadratic Lyapunov function that can be formulated as $\mathbb{E}[L(\mathbf{Z}[t+1]) - L(\mathbf{Z}[t])|\mathbf{Z}[t]]$, i.e., the drift on t. The dynamic policy is designed to solve the given optimization problem (27) by observing the current queue state, $Z_n[t]$, and determining the node association to minimize a upper bound on *drift-plus-penalty* [35]:

$$\Delta\ (\mathbf{Z}[t]) - \tilde{V}\mathbb{E}\Big[\sum_{n\in\mathcal{N}}\mathcal{P}(q_n(\alpha_n(t), t))|\mathbf{Z}[t]\Big]. \tag{33}$$

At first, find the upper bound on the change in the Lyapunov function.

$$L(\mathbf{Z}[t+1]) - L(\mathbf{Z}[t])$$

$$= \frac{1}{N}\sum_{n\in\mathcal{N}}\Big[Z_n[t+1]^2 - Z_n[t]^2\Big] \tag{34}$$

$$= \frac{1}{N}\sum_{n\in\mathcal{N}}\Big[Q_n[t+1]^2 - Q_n[t]^2 - 2\tilde{Q}(Q_n[t+1] - Q_n[t])\Big] \tag{35}$$

$$= \frac{1}{N}\sum_{n\in\mathcal{N}}\Big[(\max[Q_n[t] - b[t], 0] + a_n[t])^2 - Q_n[t]^2 - 2\tilde{Q}(\max[Q_n[t] - b_n[t], 0] + a_n[t] - Q_n[t])\Big] \tag{36}$$

$$\leq \frac{1}{N}\sum_{n\in\mathcal{N}}\Big[b_n[t]^2 + a_n[t]^2 - 2Q_n[t]b_n[t] - 2(\tilde{Q} - Q_n[t])a_n[t] + 2\tilde{Q}Q_n[t]\Big] \tag{37}$$

Then, the upper bound on the conditional Lyapunov drift is obtained as

$$\Delta\ (\mathbf{Z}[t]) = \mathbb{E}[L(\mathbf{Z}[t+1]) - L(\mathbf{Z}[t])|\mathbf{Z}[t]] \tag{38}$$

$$\leq \frac{1}{N}\sum_{n\in\mathcal{N}}\Big[1 - 2Q_n[t] + 2\tilde{Q}Q_n[t]\Big] + \mathbb{E}\Big[\frac{1}{N}\sum_{n\in\mathcal{N}}a_n[t]^2\Big|\mathbf{Z}[t]\Big] - \mathbb{E}\Big[\frac{1}{N}\sum_{n\in\mathcal{N}}2(\tilde{Q} - Q_n[t])\cdot a_n[t]\Big|\mathbf{Z}[t]\Big]. \tag{39}$$

According to (33), minimizing a bound on *drift-plus-penalty* is consistent with minimizing

$$\mathbb{E}\Big[\frac{1}{N}\sum_{n\in\mathcal{N}} a_n[t]^2\Big|\mathbf{Z}[t]\Big] - \tilde{V}\mathbb{E}\Big[\sum_{n\in\mathcal{N}}\mathcal{P}(q_n(\alpha_n(t),t))\Big|\mathbf{Z}[t]\Big] - \mathbb{E}\Big[\frac{1}{N}\sum_{n\in\mathcal{N}} 2(\tilde{Q}-Q_n[t])\cdot a_n[t]\Big|\mathbf{Z}[t]\Big]. \quad (40)$$

We now use the concept of opportunistically minimizing the expectations, so (40) is minimized by the algorithm which observes the current queue state, $\mathbf{Z}[t]$ (i.e., $\mathbf{Q}[t]$ given $\tilde{Q}$) and chooses $\alpha_n(t)$ for all $n\in\mathcal{N}$ to minimize

$$\sum_{n\in\mathcal{N}} a_n[\alpha_n(t),t]^2 - V\sum_{n\in\mathcal{N}}\mathcal{P}(q_n(\alpha_n(t),t)) - \sum_{n\in\mathcal{N}} 2(\tilde{Q}-Q_n[t])\cdot a_n[\alpha_n(t),t], \quad (41)$$

where $V=\tilde{V}\cdot N$ and $a_n[t]$ is replaced by $a_n[\alpha_n(t),t]$ to emphasize the decision parameter of $\alpha_n(t)$.

From (41), we can anticipate how the algorithm works. When the queue of user n is almost empty, the large arrivals are necessary for user n not to wait the next video chunk. In this case, user n prefers the device which gives many arrivals. On the other hand, when the queue backlogs are stacked enough to avoid playback delay, $Q_n(t)\simeq\tilde{Q}$, user n requests the video of high quality without worrying about playback latency.

System parameter V in (41) is a weight factor for the term representing video quality measure. The relative value of V to $\tilde{Q}-Q_n(t)$ is important to control the queue backlogs and quality measures at every time. The appropriate initial value of V needs to be obtained by experiment because it depends on the distribution of the cache-enabled devices, the channel environments, and the threshold of queue backlog, $\tilde{Q}$. Also, $V\geq 0$ should be satisfied. If $V<0$, users prefer low-quality videos even when a lot of video chunks have already arrived at the user queue. Moreover, in the case of $V=0$, the user only aims at stacking queue backlogs without consideration of video quality. On the other hand, when $V\to\infty$, users do not consider the queue state, and thus they just request the highest-quality files. V can be regarded as the parameter to control the trade-off between video quality and playback delay.

Since streaming users cannot know other users' channel gains, each user independently finds the cache-enabled device which stores the video file of desired quality. Therefore, (41) is treated separately, and each user minimizes its own objective function:

$$g_n(\alpha_n(t),t) = a_n[\alpha_n(t),t]^2 - V\mathcal{P}(q_n(\alpha_n(t),t)) - 2(\tilde{Q}-Q_n[t])\cdot a_n[\alpha_n(t),t]. \quad (42)$$

Since there is a finite number of cache-enabled devices within the radius R of the user, each user can easily find the device for video delivery, i.e., determination of $\alpha_n(t)$, by greedy search.

However, if two or more users simultaneously request files from the same cache-enabled device, the objective functions of those users are not independent. The reason is that the data rates of the users are obtained for one-to-one communication, (26), but the device which receives multiple file requests cannot provide the data rate of (26) to all file-requesting users. We shall call this situation the *request collision*. Since the cache-enabled device which experiences request collision can receive channel information of all file-requesting users from them, the device should resolve request collision by jointly minimizing the sum of objective functions of those users.

Assume that there are J user sets, $\mathcal{N}_{rc}(j),\ j=1,\cdots,J$, whose element users request files from the same device. Note that $\alpha_n(t)$ is the same for all $n\in\mathcal{N}_{rc}(j)$. Let $\mathcal{N}_{rc}=\mathcal{N}_{rc}(1)\cup\mathcal{N}_{rc}(2)\cup\cdots\cup\mathcal{N}_{rc}(J)$. Then, (41) can be re-written as

$$\sum_{n\in\mathcal{N}-\mathcal{N}_{rc}} g_n(\alpha_n(t),t) + \sum_{j=1}^{J}\sum_{n\in\mathcal{N}_{rc}(j)} g_n(\alpha_n(t),t). \quad (43)$$

The first term of (43) is separable, so each user $n\in\mathcal{N}-\mathcal{N}_{rc}$ just minimizes its own objective function of (42). Likewise, the summations over users $n\in\mathcal{N}_{rc}(j)$ for different j are also separable, so we can independently minimize

$$\sum_{n\in\mathcal{N}_{rc}(j)} g_n(\alpha_n(t),t) \quad (44)$$

for every $j=1,\cdots,J$. However, the element terms of summation over certain user set $\mathcal{N}_{rc}(j)$ are not independent, so additional steps are necessary to handle the occurrence of request collisions. There are two solutions: 1) scheduling of one user minimizing objective function (44) and 2) NOMA to response to the multiple requests simultaneously.

D. Approaches Against Request Collision

1) Scheduling of One User Minimizing Objective Function: In this approach, the cache-enabled device at which request collision occurs simply schedules one of the file-requesting users for video delivery, by minimizing the value of (44). After scheduling of only one user, say user n_0, others find another cache-enabled devices, $\alpha'_n(t),\forall n\in\mathcal{N}_{rc}(j), n\neq n_0$, within radius R of each user, separately. For the choices of $\alpha'_n(t)$, users follow the steps of Section IV-C, without the consideration of the cache-enabled device chosen at first, $\alpha_n(t)$. If there is no device for video delivery except for $\alpha_n(t)$, then this user should request the file from the BS.

Then, the caching device at which request collision occurs, $\alpha_n(t)$, computes $|\mathcal{N}_{rc}(j)|$ metrics of (44) for every case of scheduling of user $n\in\mathcal{N}_{rc}(j)$ and finds the one giving the minimum value. Thus, a choice of user n_0 can be obtained by

$$n_0 = \underset{n\in\mathcal{N}_{\mathrm{rc}}(j)}{\arg\min}\ g_n(\alpha_n(t),t) + \sum_{\substack{m\in\mathcal{N}_{rc}(j)\\ m\neq n}} g_m(\alpha'_m(t),t), \quad (45)$$

and let the minimum value denoted by $\mathcal{M}_O(j)$:

$$\mathcal{M}_O(j) = g_{n_0}(\alpha_{n_0}(t),t) + \sum_{\substack{m\in\mathcal{N}_{rc}(j)\\ m\neq n_0}} g_m(\alpha'_m(t),t). \quad (46)$$

Unfortunately, scheduling of one user could have a serious problem that conflicts with the noise-limited constraint, which does not allow the additional D2D link within the radius R of the streaming user whose D2D link is already constructed. If there are large overlaps among users' coverages of radius R, the cache-enabled device which is newly

found by the unscheduled user n, $n \neq n_0$, would be in the coverage of the user n_0. If so, this newly found link cannot be activated, and the unscheduled user should find another device again or directly receive the file from the BS. Furthermore, when λ is small, i.e., cache-enabled devices are sparsely located, it is likely that the unscheduled users cannot find another neighboring device. To combat these problems, NOMA is proposed to handle the multiple requests simultaneously. Since receiving the file from the neighboring device is much more advantageous in terms of transmission latency than downloading from the BS via a cellular link, NOMA would be preferred in above cases.

2) NOMA: The cache-enabled device can respond to multiple file requests simultaneously by employing NOMA. Although the NOMA signals transmitted to users interfere with each other, an advanced receiver, e.g., successive interference cancellation (SIC), can successfully remove interference [40]. However, since multiple users are served within the same resource in NOMA, degradations of data rates are inevitable. Therefore, NOMA would be useful if the system prefers to guarantee reduced transmission latency at the expense of data rate degradation.

When the cache-enabled device utilizes power-multiplexing NOMA, different power ratios, $\beta = [\beta_{m_{j,1}}, \cdots, \beta_{m_{j,|\mathcal{N}_{rc}(j)|}}]$, are weighted on the signals of all users, $m_{j,l} \in \mathcal{N}_{rc}(j),\ l \in \{1, \cdots, |\mathcal{N}_{rc}(j)|\}$. Larger power is usually allocated to the user which experiences the weaker channel condition, so power allocation ratios for file-requesting users satisfy $\beta_{m_{j,1}} < \beta_{m_{j,2}} < \cdots < \beta_{m_{j,|\mathcal{N}_{rc}(j)|}}$, with the assumption that $|h_{m_{j,1},t}|^2 > |h_{m_{j,2},t}|^2 > \cdots > |h_{m_{j,|\mathcal{N}_{rc}|},t}|^2$. The data rate of user $m_{j,l} \in \mathcal{N}_{rc}(j),\ l \in \{1, \cdots, |\mathcal{N}_{rc}(j)|\}$ in NOMA system is given by [40]

$$R_n^N(\alpha_n(t), t) = \mathcal{B}\log_2\left(1 + \frac{|h_{m_{j,l},t}|^2 \beta_{m_{j,l}}}{|h_{m_{j,l},t}|^2 \sum_{l'=1}^{l-1} \beta_{m_{j,l'}} + \sigma^2}\right). \tag{47}$$

The data rate of (47) can be obtained by performing SIC for the signals of the users with weaker channels than user $m_{j,l}$. In this case, as N increases, data rates of all file-requesting users are significantly degraded. The objective function for users $n \in \mathcal{N}_{rc}(j)$ is changed as follows:

$$\mathcal{M}_N(j) = \sum_{n \in \mathcal{N}_{rc}(j)} g_n^N(\alpha_n(t), t), \tag{48}$$

where $g_n^N(\alpha_n(t), t)$ is obtained by substituting $R_n^N(\alpha_n(t), t)$ for $R_n(\alpha_n(t), t)$ in (42).

Finally, we decide which approach is better to handle the request collision for each user set, $\mathcal{N}_{rc}(j)$ for all $j = 1, \cdots, J$, by comparing $\mathcal{M}_N(j)$ with $\mathcal{M}_O(j)$. If $\mathcal{M}_N(j) > \mathcal{M}_O(j)$, scheduling of one user is better than NOMA but, otherwise, NOMA is preferred.

V. Performance Evaluation

In this section, we show that the proposed algorithms for file placement and node association work well with video files of different quality levels. We set the parameters, $F = 5$, $Q = 3$, and $M = 6$. Also, we assume that $\gamma = 1$ and $\rho_{i,q} = \mathcal{B},\ \forall i,\ \forall q$. PSNR is considered as a video quality measure, and according to [41], quality measures and file sizes depending on quality levels are $\mathcal{P}(q) = [34,\ 36.64,\ 39.11]$ dB and $L(q) = [2621,\ 5073,\ 10658]$ kbits, respectively. Especially for finding the optimal caching probabilities, the approximately normalized file size $M_q = [1,\ 2,\ 4]$ is used.

Algorithm 2 Dynamic Node Association for Maximization of Time-Average Video Streaming Quality Sum

Precondition:
- V: parameter for streaming quality-delay trade-offs
- $\tilde{Q}$: threshold for queue backlog size

$t = 0$ // T: number of discrete-time operations
while $t \leq T$ **do**
  Observe $Q_n[t]$
  For users $n \in \mathcal{N}$, associate with the cache-enabled device, $\alpha_n^*(t) = \arg\min_{\alpha_n(t)}$ (42).
  Find $\mathcal{N}_{rc}(j),\ j = 1, \cdots, J$.
  for $j = 1 : J$ **do**
    Compute $\mathcal{M}_N(j)$ and $\mathcal{M}_O(j)$, and find n_0 and $\alpha_n'(t),\ \forall n \in \mathcal{N}_{rc}(j),\ n \neq n_0$.
    if $\mathcal{M}_N(j) > \mathcal{M}_O(j)$ **then**
      $\alpha_{n_0}^*(t) = \alpha_n(t)$
      $\alpha_n^*(t) = \alpha_n'(t),\ \forall n \in \mathcal{N}_{rc}(j),\ n \neq n_0$
    end if
  end for
end while

A. Optimal Caching Probabilities and Effects of Storage Size, Device Intensity, and SNR

According to (15), the optimal caching probabilities depend on λ and SNR. As an example, the optimal caching probabilities with $\lambda = 0.1$ and SNR = 20dB are shown in Table I. In Table I, the caching probability of the popular and low-quality file is larger than that of the unpopular and high-quality file. However, caching probabilities for different quality levels are not much different in this system, and this means that the trade-off between video quality and video diversity is unbiased. Actually, this trade-off depends on the relative values of the quality measures to the file sizes. If we arbitrarily change the file size of the quality level 3 with the fixed quality measure value, different caching probabilities are obtained. In Table I, all the first values in parentheses are for $M_q(3) = 3$ and the second values are for $M_q(3) = 6$, rather than $M_q(3) = 4$. When the file size of quality level 3 reduces to $M_q(3) = 3$, the relative file size to the quality measure decreases also, so all the caching probabilities of files of quality level 3 increase. On the other hand, when the file size of quality level 3 increases to $M_q(3) = 6$, the differences of caching probabilities between quality 1 and quality 3 increase, compared to when $M_q(3) = 4$.

Fig. 4 gives the plots of caching probabilities versus file indices with different storage sizes, M, assuming $\lambda = 0.1$ and SNR = 20dB. As M grows, all caching probabilities increase

TABLE I
OPTIMAL CACHING PROBABILITIES WITH $\lambda = 0.1$ AND SNR $= 20$DB, WHEN $M_q(3) = 4$ ($M_q(3) = 3,\ M_q(3) = 6$)

File type	Quality level		
	1	2	3
1	0.2222 (0.2438, 0.1972)	0.2183 (0.2399, 0.1932)	0.2126 (0.2474, 0.1689)
2	0.1904 (0.2120, 0.1653)	0.1865 (0.2080, 0.1614)	0.1807 (0.2155, 0.1371)
3	0.1717 (0.1933, 0.1467)	0.1678 (0.1894, 0.1428)	0.1621 (0.1969, 0.1185)
4	0.1585 (0.1801, 0.1335)	0.1546 (0.1762, 0.1296)	0.1489 (0.1837, 0.1052)
5	0.1483 (0.1699, 0.1233)	0.1444 (0.1660, 0.1193)	0.1387 (0.1735, 0.0950)

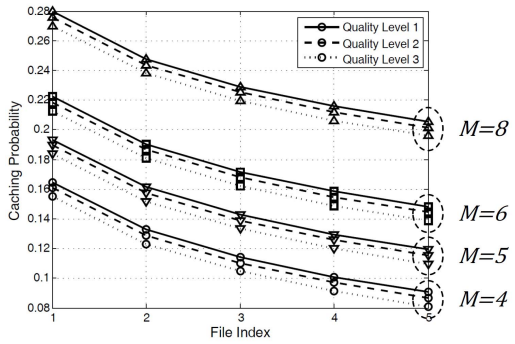

Fig. 4. Caching probabilities with different values of M.

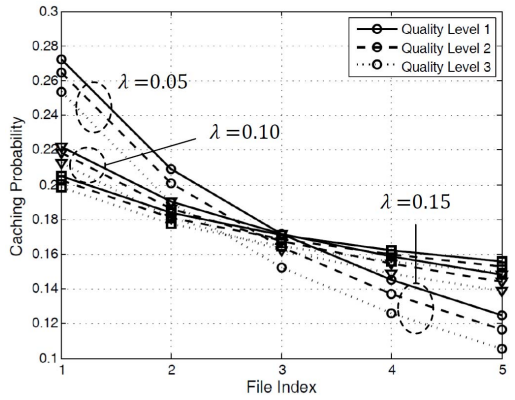

Fig. 5. Caching probabilities with different values of λ.

almost linearly. However, the differences among the caching probabilities with different quality levels are not changed, because they are influenced by the relative values of the quality measures to file sizes, as explained above.

Fig. 5 and 6 show the effects of λ and SNR on caching probabilities. Here, smaller λ and smaller SNR give the similar effects, i.e., both make file delivery via a D2D link difficult. Smaller λ means that there are a smaller number of cache-enabled devices within radius R of the user, and a smaller SNR makes the successful file delivery more difficult. Therefore, when λ and/or SNR are small, caching probabilities become biased to the popular file. When there are not many devices which can deliver the video files successfully to users, it is better to focus on storing highly demanding videos. Especially for SNR $= 10$dB in Fig. 6, files 4 and 5 will not be cached at any device. Also, the caching probability gap between high-quality and low-quality files grows as λ and SNR decrease,

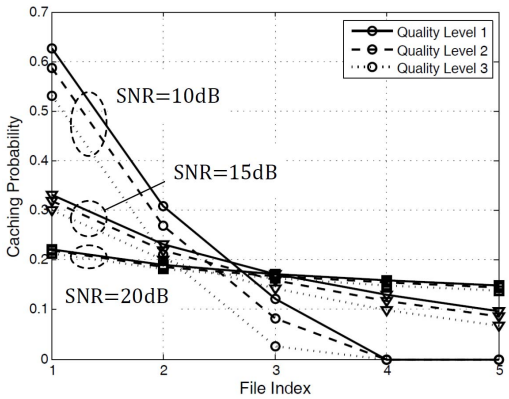

Fig. 6. Caching probabilities with different SNRs.

but the increments are not large, as long as the relative values of the quality measure to the file size is maintained.

B. Queue Backlogs and Time-Average Quality Level With Optimal Node Association

In this subsection, we examine the numerical results to verify the proposed node association algorithm. All parameters of video files and storage size are the same as the prior subsection, but SNR $= 20$ dB and $\lambda = 0.2$ are basically used here. Additionally, we set $\mathcal{B} = 1MHz$, $\tau_c = 5 \times 10^{-3}$, $V = 0.01$, $\tilde{Q} = 10^2$, and $m = 1$ so $\tau = \tau_c$. Numerical results in this subsection are based on the system model of Fig. 3. $N = 9$ users are assumed to be located in a grid structure as shown in Fig. 3, and the nearest users are separated by a distance of $d = 20$. If $R \leq d/2 = 10$, there will be no request collision, because coverage regions of users do not overlap. However, if $R \geq d/2 = 10$, request collision can occur. For NOMA, the fixed power allocation ratios $\beta = [0.8, 0.2]$ and $\beta = [9/13, 3/13, 1/13]$ are assumed for the 2-user and 3-user cases, respectively. Grouping more than three users for NOMA transmission is very rare, so NOMA for more than three users is not considered here.

To verify the advantages of the proposed node association algorithm, this paper compares the proposed one with two other comparison schemes:

- 'Maximum Arrival': The file-requesting user associates with the cache-enabled device which provides maximum arrivals within radius R.
- 'Highest-Quality': The file-requesting user associates with the cache-enabled device which caches the requested file of the highest-quality, within radius R.

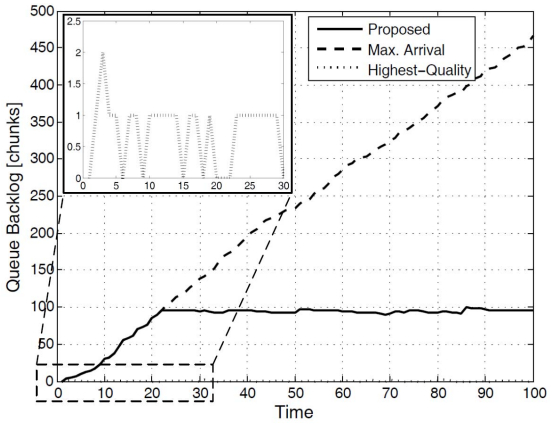

Fig. 7. Queue backlog comparisons among node association schemes.

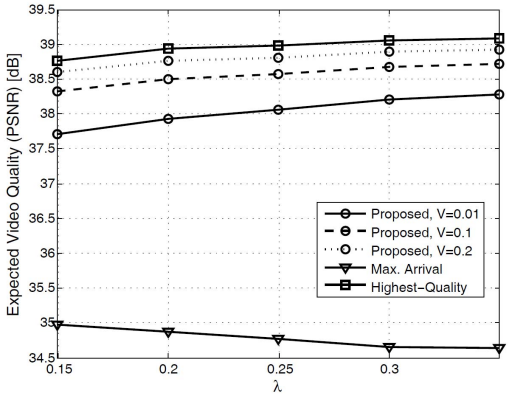

Fig. 8. Time-averaged video quality measures with different values of λ.

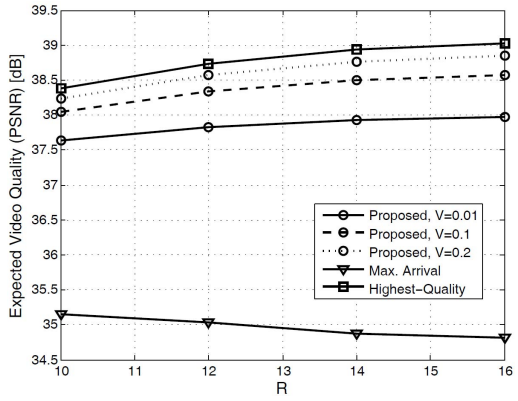

Fig. 9. Time-averaged video quality measures with different values of R.

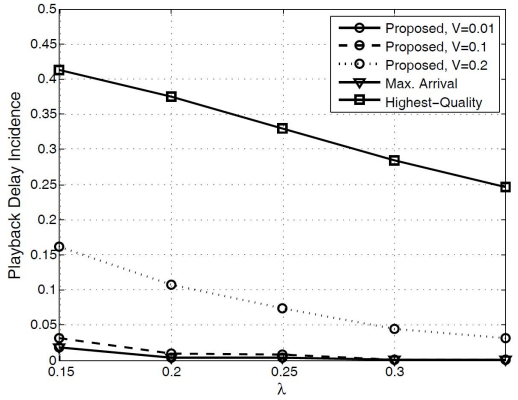

Fig. 10. Playback delay incidence with different values of λ.

Fig. 7 gives the plots of queue backlogs, i.e. the number of video chunks stacked in queue, versus time slot. The largest backlogs are stacked with the 'Max-Arrival' scheme, the proposed algorithm is the next, and backlogs are hardly accumulated with the 'Highest-Quality' scheme, as shown in Fig. 7. The 'Max-Arrival' scheme communicates with the link providing the largest number of chunks, so it does not have to worry about playback delay, compared to other schemes. For the proposed algorithm, the smaller video chunks are stacked in queue than 'Max-Arrival', but its backlogs are large enough to avoid playback latency. Specifically, Fig. 7 shows the effect of $\tilde{Q}$ which limits the maximal backlogs. Since $\tilde{Q} = 100$ chunks are enough to avoid playback delay, there is no need to stack too many chunks like 'Max. Arrival'. Therefore, when $Q_n[t] \approx \tilde{Q}$, the proposed algorithm strongly pursues the high-quality file even though the D2D link of the device with the high-quality file is not good. In addition, the queue size is finite in practical, the use of $\tilde{Q}$ can prevent queue overflow. The 'Highest-Quality' scheme has very little margin of the backlog for smooth video playback, and its enlarged graph is also shown in Fig. 7. The user queue of 'Highest-Quality' scheme is frequently empty, thus several occurrences of playback delay are expected.

The time-averaged video quality measures with different values of λ and R are shown in Figs. 8 and 9, respectively. Obviously, the 'Highest-Quality' scheme gives the best video quality. Since the 'Max. Arrival' scheme does not pursue video quality enhancement, it is obvious that its performance is the worst among the compared techniques in Figs. 8 and 9. The performance of the proposed algorithm is better than 'Max-Arrival' and worse than 'Highest-Quality'; however the proposed algorithm provides the similar quality measures to 'Highest-Quality' as V increases. In addition, the performances of the proposed scheme and 'Highest-Quality' improve with λ or R because these schemes pursue video quality enhancement. On the other hand, 'Max. Arrival' associates with the device which provides the maximal number of arrivals, preferring the strong channel and small file size (i.e., low-quality file). Thus, the quality measure of 'Max. Arrival' degrades with λ and R.

Figs. 10 and 11 are plots of playback delay incidence versus λ and R, respectively. As we explained earlier, when there is no chunk in the queue while enjoying streaming service, playback delay occurs. Therefore, playback delay incidence means how much queue emptiness occurs over the total playback time. Among comparison schemes, the proposed algorithm with $V = 0.01$ and 'Max. Arrival' show the lowest playback delay, whereas there are much buffering times expected for the 'Highest-Quality' scheme. As λ increases, more device candidates which can provide the desired file with good channel conditions are expected; thus delay incidences of

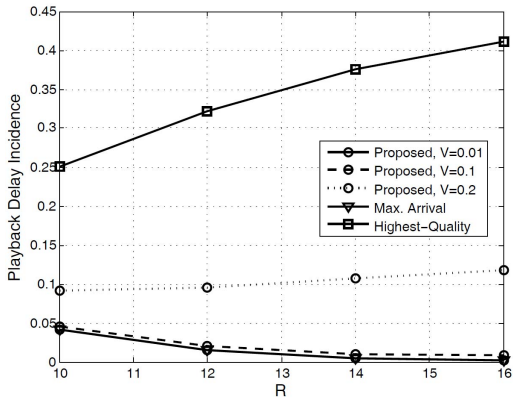

Fig. 11. Playback delay incidence with different values of R.

all schemes decrease, whereas the trends of delay incidences in accordance with R are different. In the 'Highest-Quality' scheme, when R is large and λ is fixed, the distance between the streaming user and the device storing the best-quality file would be large, i.e., the associated device would experience a bad channel. On the contrary, for the 'Max. Arrival' scheme it becomes easier to find the device candidate which can deliver more chunks, assuming R is large. Thus, delay incidence of 'Highest-Quality' increases with R whereas that of 'Max. Arrival' decreases. In the case of the proposed algorithm, as V becomes larger, the streaming user pursues the video quality rather than reduced playback delay. Thus, the delay incidence with $V = 0.2$ increases with R, whereas those with $V = 0.1$ and $V = 0.01$ do not.

Considering the results of both quality measure and playback delay incidence, we can say that the proposed algorithm smooths out the trade-off between video quality and playback delay. The 'Max-Arrival' scheme is good to avoid playback latency, but the file-requesting users would be suffered from the degraded video quality. On the other hand, the 'Highest-Quality' scheme provides the best video quality, but its user experiences too much buffering times to enjoy the smooth streaming service. Thus, the proposed node association algorithm can be useful for achieving both acceptable playback delay and high enough video quality. In addition, the trade-off between video quality and reduced playback delay in our proposed algorithm can be controlled by adjusting the system parameter V. In Figs. 8 and 9, the expected video quality increases as V grows up, whereas the playback delays occur more frequently as shown in Figs. 10 and 11.

VI. Concluding Remarks

This paper considered video files of various quality levels in the D2D-assisted wireless caching network. This paper suggests the optimal caching policy for video files of different quality levels and thus of different sizes which maximizes the successfully enjoyable quality sum. In addition, a node association algorithm has been proposed that maximizes the sum of the time-average video quality that file-requesting users enjoy while preventing playback delay, the most important user QoS in video streaming service. In this paper, *request collision*, the situation where a device receives file requests from multiple users at the same time, has been considered, and the solutions based on NOMA as well as scheduling of one user have been presented. The proposed file placement rule and the node association algorithm have been verified by simulation results.

References

[1] "Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021," Cisco Syst., San Jose, CA, USA, White Paper 1454457600805266. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/serviceprovider/visual-networking-index-vni/mobile-white-paper-c11-520862.html

[2] X. Cheng, J. Liu, and C. Dale, "Understanding the characteristics of Internet short video sharing: A YouTube-based measurement study," *IEEE Trans. Multimedia*, vol. 15, no. 5, pp. 1184–1194, Aug. 2013.

[3] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5G wireless networks," *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 82–89, Aug. 2014.

[4] 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.*, vol. 52, no. 2, pp. 131–139, Feb. 2014.

[5] K. Poularakis, G. Iosifidis, V. Sourlas, and L. Tassiulas, "Exploiting caching and multicast for 5G wireless networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 4, pp. 2995–3007, Apr. 2016.

[6] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "FemtoCaching: Wireless content delivery through distributed caching helpers," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 8402–8413, Dec. 2013.

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

[8] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless D2D networks," *IEEE Trans. Inf. Theory*, vol. 62, no. 2, pp. 849–869, Feb. 2016.

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

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

[11] F. Hartanto, J. Kangasharju, M. Reisslein, and K. W. Ross, "Caching video objects: Layers vs versions?" in *Proc. IEEE ICME*, Lausanne, Switzerland, Aug. 2002, pp. 45–48.

[12] K. Poularakis, G. Iosifidis, A. Argyriou, and L. Tassiulas, "Video delivery over heterogeneous cellular networks: Optimizing cost and performance," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Toronto, ON, Canada, Apr./May 2014, pp. 1078–1086.

[13] A. Argyriou, K. Poularakis, G. Iosifidis, and L. Tassiulas, "Video delivery in dense 5G cellular networks," *IEEE Netw.*, vol. 31, no. 4, pp. 28–34, Jul./Aug. 2017.

[14] B. Blaszczyszyn and A. Giovanidis, "Optimal geographic caching in cellular networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, U.K., Jun. 2015, pp. 3358–3363.

[15] S. H. Chae and W. Choi, "Caching placement in stochastic wireless caching helper networks: Channel selection diversity via caching," *IEEE Trans. Wireless Commun.*, vol. 15, no. 10, pp. 6626–6637, Oct. 2016.

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

[17] B. Shen, S.-J. Lee, and S. Basu, "Caching strategies in transcoding-enabled proxy systems for streaming media distribution networks," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 375–386, Apr. 2004.

[18] 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, pp. 1431–1445, Oct. 2013.

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

[20] C. Jarray and A. Giovanidis, "The effects of mobility on the hit performance of cached D2D networks," in *Proc. 14th Int. Symp. Modeling Optim. Mobile, Ad Hoc, Wireless Netw. (WiOpt)*, Tempe, AZ, USA, May 2016, pp. 1–8.
[21] K. Poularakis, G. Iosifidis, A. Argyriou, I. Koutsopoulos, and L. Tassiulas, "Caching and operator cooperation policies for layered video content delivery," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, San Francisco, CA, USA, Apr. 2016, pp. 1–9.
[22] Z. Ye, F. De Pellegrini, R. El-Azouzi, L. Maggi, and T. Jimenez, "Quality-aware DASH video caching schemes at mobile edge," in *Proc. 29th Int. Teletraffic Congr. (ITC)*, Genoa, Italy, Sep. 2017, pp. 205–213.
[23] C. Zhan and Z. Wen, "Content cache placement for scalable video in heterogeneous wireless network," *IEEE Commun. Lett.*, vol. 21, no. 12, pp. 2714–2717, Dec. 2017.
[24] A. Araldo, F. Martignon, and D. Rossi, "Representation selection problem: Optimizing video delivery through caching," in *Proc. IFIP Netw. Conf. (IFIP Netw.) Workshops*, Vienna, Austria, May 2016, pp. 323–331.
[25] L. Wu and W. Zhang, "Caching-based scalable video transmission over cellular networks," *IEEE Commun. Lett.*, vol. 20, no. 6, pp. 1156–1159, Jun. 2016.
[26] C. Yang, Y. Yao, Z. Chen, and B. Xia, "Analysis on cache-enabled wireless heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 1, pp. 131–145, Jan. 2016.
[27] K. Poularakis, G. Iosifidis, and L. Tassiulas, "Approximation algorithms for mobile data caching in small cell networks," *IEEE Trans. Commun.*, vol. 62, no. 10, pp. 3665–3677, Oct. 2014.
[28] L. Zhang, M. Xiao, G. Wu, and S. Li, "Efficient scheduling and power allocation for d2d-assisted wireless caching networks," *IEEE J. Sel. Areas Commun.*, vol. 64, no. 6, pp. 2438–2452, Jun. 2016.
[29] W. Jiang, G. Feng, and S. Qin, "Optimal cooperative content caching and delivery policy for heterogeneous cellular networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 5, pp. 1382–1393, May 2017.
[30] H. Ahlehagh and S. Dey, "Adaptive bit rate capable video caching and scheduling," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Shanghai, China, Apr. 2013, pp. 1357–1362.
[31] T. X. Tran, P. Pandey, A. Hajisami, and D. Pompili, "Collaborative multi-bitrate video caching and processing in mobile-edge computing networks," in *Proc. 13th Annu. Conf. Wireless On-Demand Netw. Syst. Services (WONS)*, Jackson, WY, USA, Feb. 2017, pp. 165–172.
[32] C. Liang and S. Hu. (Jun. 2017). "Dynamic video streaming in caching-enabled wireless mobile networks." [Online]. Available: https://arxiv.org/abs/1706.09536
[33] X. Wang, M. Chen, T. T. Kwon, L. T. Yang, and V. C. M. Leung, "AMES-cloud: A framework of adaptive mobile video streaming and efficient social video sharing in the clouds," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 811–820, Jun. 2013.
[34] J. Kim, G. Caire, and A. F. Molisch, "Quality-aware streaming and scheduling for device-to-device video delivery," *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, pp. 2319–2331, Apr. 2016.
[35] D. Bethanabhotla, G. Caire, and M. J. Neely, "Adaptive video streaming for wireless networks with multiple users and helpers," *IEEE Trans. Commun.*, vol. 63, no. 1, pp. 268–285, Jan. 2015.
[36] T. Stockhammer, "Dynamic adaptive streaming over HTTP: Standards and design principles," in *Proc. ACM MMSys*, San Jose, CA, USA, Feb. 2011, pp. 133–144.
[37] M. Gregori, J. Gómez-Vilardebó, J. Matamoros, and D. Gündüz, "Wireless content caching for small cell and D2D networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1222–1234, May 2016.
[38] D. Bertsekas and R. G. Gallager, *Data Networks*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1992.
[39] M. J. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan & Claypool, 2010.
[40] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge, U.K.: Cambridge Univ. Press, 2005.
[41] J. Kim and E.-S. Ryu, "Feasibility study of stochastic streaming with 4K UHD video traces," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Jeju, South Korea, Oct. 2015, pp. 1350–1355.

Minseok Choi received the B.S. and M.S. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2016, where he is currently pursuing the Ph.D. degree. His research interests include NOMA, wireless caching networks, 5G communications, and mmWave.

Joongheon Kim (M'06–SM'18) received the B.S. and M.S. degrees in computer science and engineering from Korea University, Seoul, South Korea, in 2004 and 2006, respectively, and the Ph.D. degree in computer science from the University of Southern California, Los Angeles, CA, USA. In industry, he was with LG Electronics Seocho Research and Development Campus, Seoul, from 2006 to 2009, InterDigital, San Diego, CA, USA, in 2012, and Intel Corporation, Santa Clara, CA, USA, from 2013 to 2016. He has been an Assistant Professor with Chung-Ang University, Seoul, since 2016.

He is a member of the IEEE Communications Society. He was a recipient of the Annenberg Graduate Fellowship with his Ph.D. admission from USC in 2009.

Jaekyun Moon (F'05) received the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA. He is currently a Professor of electrical engineering with KAIST. From 1990 to 2009, he was with the faculty of the Department of Electrical and Computer Engineering, University of Minnesota, Twin Cities. He was consulted as a Chief Scientist for DSPG, Inc., from 2004 to 2007. He was also a Chief Technology Officer with Link-A-Media Devices Corporation. His research interests include channel characterization, signal processing, and coding for data storage and digital communication. He received the McKnight Land-Grant Professorship from the University of Minnesota, the IBM Faculty Development Awards, the IBM Partnership Awards, the National Storage Industry Consortium Technical Achievement Award for the invention of the maximum transition run code, a widely used error-control/modulation code in commercial storage systems. He served as the Program Chair for the 1997 IEEE Magnetic Recording Conference. He is also Past Chair of the Signal Processing for Storage Technical Committee of the IEEE Communications Society. He served as a Guest Editor for the 2001 IEEE JSAC Issue on Signal Processing for High Density Recording. He also served as an Editor for the IEEE TRANSACTIONS ON MAGNETICS in the area of signal processing and coding for 2001–2006.