Audience Retention Rate Aware Coded Video Caching

Qianqian Yang, Mohammad Mohammadi Amiri, and Deniz Gündüz
Electrical and Electronic Engineering Department, Imperial College London, London SW7 2AZ, U.K.
Email: {q.yang14, m.mohammadi-amiri15, d.gunduz}@imperial.ac.uk

Abstract—**Users often do not watch an online video content in its entirety, and abort the video before it is completed. This is captured by the notion of *audience retention rate*, which indicates the portion of a video users watch on average. A decentralized coded caching scheme, called *partial coded caching* (PCC), is proposed here to take into account both the popularity, and the audience retention rate of the video files in a database. The achievable average delivery rate of PCC is characterised over all possible demand combinations. Two different cache allocation schemes, called the *optimal cache allocation* (OCA) and the *popularity based cache allocation* (PCA), are proposed to allocate cache capacities among the different chunks of video files. Numerical results validate that the proposed coded caching scheme, either with the OCA or the PCA, outperforms conventional uncoded caching, as well as the state-of-the-art coded caching schemes that consider only file popularities.**

I. Introduction

The ever-increasing demand for video services has been the main driver for the recent explosive growth of wireless data traffic. A key feature of video services is that a small portion of highly popular contents dominate the traffic [1]. This led to the idea of prefetching popular contents over off-peak traffic periods, or at better channel conditions, and storing them at the network edge [2], or even directly at user devices [3], [4], referred to as *proactive caching*. Proactive caching can alleviate both the growing bandwidth requirement and the incremental delay due to the continuously increasing data traffic; and it becomes more viable thanks to the decreasing cost of memory (see [2], [3], [5], [6], and references therein).

Proactive caching shifts part of the data traffic from the peak traffic periods to off-peak periods. During the off-peak traffic periods, users' caches are filled as a function of the whole library of popular files, referred to as the *placement phase*. Users' demands are revealed during the peak traffic periods, and they are satisfied simultaneously over the *delivery phase*. Traditional uncoded caching schemes adopt orthogonal unicast transmissions, and the caching gain is simply given by the capacity of each user's local cache. On the other hand, *coded caching*, a novel caching paradigm introduced in [5], further exploits the cache resources through joint optimization of the two phases in order to create and exploit coded multicasting opportunities, even among distinct user requests.

The global caching gain, and the achievability of a delivery rate that is within a constant multiplicative factor of the information theoretic lower bound in [5], has ignited intense research activities on coded caching [4], [7]–[14]. While some studies aim to improve the gap between the upper and lower bounds [4], [8], [9], others adapt coded caching to different settings, such as decentralized coded caching [7], files with nonuniform popularities [11], [12], coded caching of files with lossy reconstruction [14].

The literature on coded caching shares a common assumption, that is, users request entire files, whereas in practice users rarely request and watch an entire video content. A recent report [1] suggests that, users on average watch 60% of the requested files from a trace of 7000 Youtube videos, and the number of views varies over different videos as well as different parts of each video. This phenomena is captured by the notion of *audience retention rate*, introduced by the mainstream online video platforms, such as Youtube and Netflix, to model the popularities of different parts of videos.

This nonuniform viewing behaviour calls for *partial caching*, where only a portion of each video file is cached. Partial caching is shown to improve the performance of uncoded caching in [15]. Here we investigate coded caching of video files taking into account the audience retention rate for each video. We consider that each video file consists of equal-length chunks, and the audience retention rate of each chunk is the fraction of users watching this chunk among total views of the corresponding video. Taking the audience retention rate into account, we propose a coded caching scheme with decentralized cache placement, referred to as *partial coded caching*, and derive a closed-form expression for the achievable average delivery rate over all possible demand combinations. Two different cache allocation schemes are proposed to allocate users' caches to cache different chunks, namely *optimal cache allocation* (OCA) and *popularity based cache allocation* (PCA). Note that the coded caching problem with different file popularities, studied in [11], is a special case of the problem considered in this paper obtained by setting the audience retention rates of all the chunks to 1. Numerical results indicate that the proposed audience retention rate aware partial coded caching scheme achieves a better delivery rate than both uncoded caching, and the scheme proposed in [11].

Notations: We denote the set of t-bit binary sequences by $[2^t]$, and the set of all binary sequences by $[2^*]$. For sets $\mathcal{A}$ and $\mathcal{B}$, we define $\mathcal{A}\backslash\mathcal{B} \triangleq \{x : x \in \mathcal{A}, x \notin \mathcal{B}\}$, and $|\mathcal{A}|$ denotes the cardinality of $\mathcal{A}$. Notation $\bar{\oplus}$ represents the bitwise XOR operation, where the arguments are zero-padded to have equal length. For event E, $\mathbb{1}\{E\} = 1$ if E is true; and $\mathbb{1}\{E\} = 0$, otherwise.

II. System Model

We consider a server holding a library of N video files, denoted by $\mathcal{F} = \{W_1, W_2, ..., W_N\}$. We assume, for simplicity, that all the files have the same size of F bits. Each file consists of B chunks of equal size, i.e., F/B bits, which is determined by various factors in practical applications, such as frame size, display settings of user devices, etc. [16]. We denote by W_{ij} the jth chunk of W_i. There are K users in the system, connected to the server through an error-free shared link. Each user is equipped with a cache of size MF bits.

The system operates in two phases: the *placement phase* and the *delivery phase*. In the placement phase, which takes place during an off-peak traffic period, each user pre-fetches data from the server to fill its cache. The cache content of user k at the end of this phase is denoted by Z_k, for $k = 1, \dots, K$. The delivery phase takes place during a peak traffic period, once the users reveal their demands. The user demand realization is denoted by $\mathbf{D} = (d_1, d_2, ..., d_K)$, the components of which are independent and identically distributed (i.i.d.) according to $\mathbf{p} \triangleq (p_1, ..., p_N)$, the *popularity distribution* of $\mathcal{F}$.

Unlike the previous literature on coded caching, we do not deliver the requested contents as a whole, as users often quit watching a video file before completion. Therefore, in our model, users are initially delivered only the first chunks of their desired files. Their demands of subsequent chunks are only revealed after receiving the previous ones, unless they abort watching the video. We denote by $\mathbf{d}^j \triangleq (d_1^j, d_2^j, ..., d_K^j)$ the demand vector of the jth chunks, for $j = 1, ..., B$, such that $d_k^j = 1$ indicates user k requests the jth chunk of its desired file, otherwise, $d_k^j = 0$, for $k = 1, ..., K$. After receiving the demand vector of the jth chunks, i.e., $\mathbf{d}^j$, the server sends a single message $X^j_{\mathbf{D},\mathbf{d}^j}$ over the shared link depending on users' requests as well as the contents of their caches.

We employ the notion of *audience retention rate*, defined as the fraction of users that request chunk W_{ij} among all the users that have requested W_i, denoted by p_{ij}, for $i = 1, ..., N$, $j = 1, ..., B$ [15]. Alternatively, we can regard p_{ij} as the probability that a user that has requested video W_i will watch chunk j, i.e.,

$$p_{ij} \triangleq \Pr\left\{ d_k^j = 1 \middle| d_k = i \right\}, \qquad \forall k. \tag{1}$$

We assume that p_{ij} is non-increasing in j, i.e., $1 = p_{i1} \geq p_{i2} \geq \cdots \geq p_{iB}$, which characterises a realistic viewing model that users start watching videos from the beginning and then stop watching after a random portion. We let $\mathbf{P} = \{p_{ij}, i = 1, ..., N, j = 1, ..., B\}$ denote the *retention rate matrix* for all the chunks in the library, which is identical for all the users. We refer to $p_i p_{ij}$ as the popularity of chunk W_{ij} in the sense that it denotes the probability that chunk W_{ij} will be requested by a user when it makes a request.

A caching scheme consists of K caching functions f_k,

$$f_k : \underbrace{[2^F] \times \cdots \times [2^F]}_{N \text{ files}} \times \mathbf{p} \times \mathbf{P} \rightarrow [2^{MF}], \tag{2}$$

for $k = 1, ..., K$, where $Z_k = f_k(\{W_i\}_{i=1}^N, \mathbf{p}, \mathbf{P})$, and B delivery functions,

$$g_j : \underbrace{[2^F] \times \cdots \times [2^F]}_{N \text{ files}} \times \mathbf{D} \times \mathbf{d}^j \rightarrow [2^*], \tag{3}$$

for $j = 1, ..., B$, and $X^j_{\mathbf{D},\mathbf{d}^j} = g_j(\{W_{ij}\}_{i=1}^N, \mathbf{D}, \mathbf{d}^j)$, and $B \times K$ decoding functions,

$$h_k^j : \mathbf{D} \times \mathbf{d}^j \times [2^{MF}] \times [2^*] \rightarrow [2^{F/B}], \tag{4}$$

for $j = 1, ..., B$, $k = 1, ..., K$, where $\hat{W}_{d_k j} = h_k^j(\mathbf{D}, \mathbf{d}^j, Z(k), X^j_{\mathbf{D},\mathbf{d}^j})$, is the reconstruction of jth chunk of W_{d_k} at user k. Note that the length of $X^j_{\mathbf{D},\mathbf{d}^j}$ (normalized by F/B), denoted by $R_j(\mathbf{D}, \mathbf{d}^j)$, depends on the demand configuration specified by $\mathbf{D}$ and $\mathbf{d}^j$. We emphasize that both the popularity distribution, $\mathbf{p}$, and the retention rate matrix, $\mathbf{P}$, are known a priori by the system. We define the average rate of a *caching* scheme, consisting of the caching, delivery, and reconstruction functions described above, as follows:

$$\bar{R} \triangleq \mathbb{E}\left(\sum_{j=1}^{B} R_j(\mathbf{D}, \mathbf{d}^j) \right), \tag{5}$$

where the expectation is over all possible demand combinations, represented by $\mathbf{D}$ and $\{\mathbf{d}^j\}_{j=1}^B$, distributed according to $\mathbf{p}$ and $\mathbf{P}$, respectively.

Definition 1. *An average-rate $\bar{R}$ is achievable for a system with N files, K users, each with a cache of capacity MF bits, and the popularity distribution $\mathbf{p}$ and the retention rate matrix $\mathbf{P}$, if there exists a caching scheme with average rate $\bar{R}$ such that for any demand realization $\mathbf{D}$ and $\{\mathbf{d}^j\}_{j=1}^B$,*

$$\lim_{F/B \rightarrow \infty} \Pr\left\{ \bigcup_{j=1}^{B} \bigcup_{k: d_k^j = 1} \left\{ \hat{W}_{d_k j} \neq W_{d_k j} \right\} \right\} = 0. \tag{6}$$

Our goal is to characterise the minimum average achievable rate $\bar{R}$ for a given system configuration.

Remark 1. *When $p_{ij} = 1$, $\forall i, j$, i.e., the users always watch the videos until the end once they start, the caching problem studied in this paper reduces to the one that considers only file popularities, which has been studied in [11], [12].*

III. Partial Coded Caching (PCC)

Here we first present the placement and delivery phases of our coded caching scheme, referred to as the *partial coded caching* (PCC) scheme, and then derive its achievable delivery rate. We remark that the number of bits and the delivery rate mentioned in the sequel are both normalized by F/B.

A. Placement Phase

During the cache placement phase, each user selects an independent random subset of $q_{ij}F/B$ bits from the jth chunk of file W_i, i.e., W_{ij}, to fill its cache, i.e., in a decentralized manner, where $0 \leq q_{ij} \leq 1$, such that $\sum\limits_{i=1}^{N} \sum\limits_{j=1}^{B} q_{ij} = MB$, which meets the limitation of the cache capacities. We refer to $\mathbf{Q} = \{q_{ij}\}$ as the *cache content distribution*. The optimization of $\mathbf{Q}$ is studied in Section III-D.

B. Delivery Phase

The delivery phase is performed once users reveal their demands, $\mathbf{D} = (d_1, d_2, ..., d_K)$. We emphasize that users' requests for the jth chunk, $\mathbf{d}^j = (d_1^j, d_2^j, ..., d_K^j)$, are not disclosed until the users have already received the $(j-1)$th chunk. We denote by $\mathcal{C}_j$ the set of users requesting the jth chunk, i.e., $\mathcal{C}_j = \{k : d_k^j = 1\}$. We represent by $\mathcal{D}^j$ the set of files whose jth chunks are requested by at least one user. We have

$$\mathcal{D}^j \triangleq \left\{ i : \sum_{k=1}^{K} \mathbb{1}\{d_k = i\} \cdot d_k^j \geq 1 \right\}. \tag{7}$$

$W_{ij,\mathcal{S}}$ denotes the bits of chunk W_{ij} that are cached exclusively by the users in $\mathcal{S}$. We present our coded delivery scheme to send the requested jth chunks to users in $\mathcal{C}_j$ in Algorithm 1. We note that, among the CODED DELIVERY and RANDOM DELIVERY procedures of Algorithm. 1, we perform the one that requires a smaller delivery rate, is performed.

Remark 2. *We remark that the coded delivery scheme in [17, Algorithm 1], which, compared to the proposed scheme, requires a smaller number of bits to be delivered to satisfy the same demand combination, can also be employed here. However, the average delivery rate of the scheme in [17, Algorithm 1] does not lend itself to a closed-form expression; therefore we consider the delivery scheme outlined in Algorithm 1. Despite its suboptimality, the proposed scheme outperforms the state-of-the-art results for coded caching with non-uniform file popularities, as it will be shown in the sequel.*

C. Achievable Rate

The following theorem provides a closed-form expression for the achievable average delivery rate over all possible demands using the proposed coded caching scheme. We first provide some definitions which simplify the presentation of our results. For any l-element subset $\mathcal{S}_l$ of $\{1, ..., K\}$, $g_{ij,\mathcal{S}_l}$ denotes the number of bits of chunk W_{ij} that are cached exclusively by $l-1$ users in $\mathcal{S}_l$ (which is identical for any $l-1$ users in $\mathcal{S}_l$ based on the law of large number). We have

$$g_{ij,\mathcal{S}_l} = (q_{ij})^{l-1}(1-q_{ij})^{K-l+1}. \tag{8}$$

For a given l-element subset $\mathcal{S}_l$ of $\{1, ..., K\}$, we define $\rho_{ij,\mathcal{S}_l}$ as the probability of the event that among the jth chunks requested by at least one user from $\mathcal{S}_l$, W_{ij} has the maximum number of bits cached exclusively by $l-1$ users from $\mathcal{S}_l$, i.e.,

$$\rho_{ij,\mathcal{S}_l} \triangleq \Pr\Big\{ g_{ij,\mathcal{S}_l} \geq g_{fj,\mathcal{S}_l} : i \in \mathcal{D}_l \cap \mathcal{D}^j, \forall f \in \mathcal{D}_l \cap \mathcal{D}^j \Big\}, \tag{9}$$

where D_l is the set of files requested by users in $\mathcal{S}_l$.

Due to symmetry across users, $\rho_{ij,\mathcal{S}_l}$ is identical for any $\mathcal{S}_l \subset \{1, ..., K\}$. Thus, for simplicity, we use $\rho_{ij,l}$ to denote $\rho_{ij,\mathcal{S}_l}$ for all subsets $\mathcal{S}_l$ as well as $g_{ij,l}$ instead of $g_{ij,\mathcal{S}_l}$.

Algorithm 1 Delivery scheme of the jth chunks

```
procedure CODED DELIVERY
    PART 1: Delivering the missing bits that are not in the cache of any user:
    for $i \in \mathcal{D}^j$ do
        Send $W_{ij,\emptyset}$
    end for

    PART 2: Delivering the missing bits that are in the cache of only one user; the one among PART 2.1 and PART 2.2 that requires a smaller delivery rate is executed:
    PART 2.1:
    for $\mathcal{P} \subset \underline{\{1,}..., K\}$: $|\mathcal{P}| = 2$ do
        Send $\bigoplus_{k \in \mathcal{P} \cap \mathcal{C}_j} W_{d_k j, \mathcal{P} \setminus \{k\}}$
    end for
    PART 2.2:
    for $i \in \mathcal{D}^j$ do
        Send $\left( \bigcup_{t=1}^{K-1} W_{ij,\{t\}} \bar{\oplus} W_{ij,\{t+1\}} \right)$
    end for

    PART 3: Delivering the missing bits that are in the cache of more than one user:
    for $\mathcal{P} \subset \underline{\{1,}..., K\}$: $|\mathcal{P}| > 2$ do
        Send $\bigoplus_{k \in \mathcal{P} \cap \mathcal{C}_j} W_{d_k j, \mathcal{P} \setminus \{k\}}$
    end for
end procedure

procedure RANDOM DELIVERY
    for $i = 1, 2, \ldots, N$ do
        Server sends enough random linear combinations of the bits of file $W_{ij}$ to enable the users demanding it to decode it.
    end for
end procedure
```

Theorem 1. *For the caching system described in Section II, given a cache content distribution* $\mathbf{Q}$*, the following average delivery rate is achievable*

$$\bar{R}(\mathbf{Q}) = \min\{\varphi(\mathbf{p}, \mathbf{P}, \mathbf{Q}), \bar{m}\}, \tag{10}$$

where

$$\bar{m} \triangleq \sum_{j=1}^{B} \sum_{i=1}^{N} \left(1 - (1 - p_i p_{ij})^K\right)(1 - q_{ij}), \tag{11}$$

and

$$\varphi(\mathbf{p}, \mathbf{P}, \mathbf{Q}) \triangleq \sum_{j=1}^{B} \sum_{i=1}^{N} \left(1 - (1 - p_i p_{ij})^K\right)(1 - q_{ij})^K$$
$$+ \sum_{j=1}^{B} \sum_{l=3}^{K} \binom{K}{l} \sum_{i=1}^{N} \rho_{ij,l}(q_{ij})^{l-1}(1 - q_{ij})^{K-l+1}$$

$$+\min\left\{\sum_{j=1}^{B}\sum_{i=1}^{N}\left(1-(1-p_i p_{ij})^K\right)q_{ij}(1-q_{ij})^{K-1},\right.$$
$$\left.\sum_{j=1}^{B}\binom{K}{2}\sum_{i=1}^{N}\rho_{ij,2}q_{ij}(1-q_{ij})^{K-1}\right\}. \tag{12}$$

Proof. The proof can be found in Appendix A. □

The value of $\rho_{ij,l}$ can be calculated as follows. We define

$$Y_{j,l} \triangleq \max_{f\in\mathcal{D}_l\cap\mathcal{D}^j} g_{fj,l}. \tag{13}$$

It follows that

$$\Pr\{Y_{j,l} \le g_{ij,l}\} = \left(\sum_{W_f\in\mathcal{F}:g_{fj,l}\le g_{ij,l}} p_f + \sum_{W_f\in\mathcal{F}:g_{fj,l}>g_{ij,l}} p_f(1-p_{fj})\right)^l, \tag{14}$$

that is, the probability of $Y_{j,l} \le g_{ij,l}$ is the probability that each element $f \in \mathcal{D}_l$ is either associated with $g_{fj,l}$ no larger than $g_{ij,l}$, or $f \notin \mathcal{D}^j$. Similarly,

$$\Pr\{Y_{j,l} < g_l(i,j)\} = \left(\sum_{W_f\in\mathcal{F}:g_{fj,l}<g_{ij,l}} p_f + \sum_{W_f\in\mathcal{F}:g_{fj,l}\ge g_{ij,l}} p_f(1-p_{fj})\right)^l, \tag{15}$$

i.e., the probability that each element $f \in \mathcal{D}_l$ is either associated with $g_{fj,l}$ less than $g_{ij,l}$, or $f \notin \mathcal{D}^j$. Then, we derive

$$\rho_{ij,l} = \frac{\Pr\{Y_{j,l} = g_{ij,l}\}}{\sum_{f=1}^{N}\mathbb{1}\left\{g_{fj,l} = g_{ij,l}\right\}}. \tag{16}$$

where $\Pr\{Y_{j,l} = g_{ij,l}\} = \Pr\{Y_{j,l} \le g_{ij,l}\} - \Pr\{Y_{j,l} < g_{ij,l}\}$. Hence, the denominator in (16) is the total number of files which have the same value of $g_{fj,l}$ and $g_{ij,l}$. Thus, $\rho_{ij,l}$ can be easily calculated by sorting $\{g_{fj,l}, W_f \in \mathcal{F}\}$.

D. Cache Allocation

We formulate the optimization of cache content distribution $\mathbf{Q}$ as follows:

$$\min \bar{R}(\mathbf{Q}) \tag{17a}$$
$$\text{s.t.} \sum_{i,j} q_{ij} = MB, \tag{17b}$$

where the objective is to minimize the average delivery rate over all possible demand combinations while the cached contents meet the limitation of the cache capacities. The optimization problem in (17) can be solved numerically, and the corresponding solution will be referred to as the *optimal cache allocation* (OCA).

However, in practice, there will be a large number of files in the library, and each video file can be partitioned into many chunks. In that case, optimizing $\mathbf{Q}$ over all the chunks in the library would require high computational complexity. Instead, we present a simple cache allocation scheme, referred to as *popularity based cache allocation* (PCA), in which only the most popular chunks are cached by the users; that is, we have

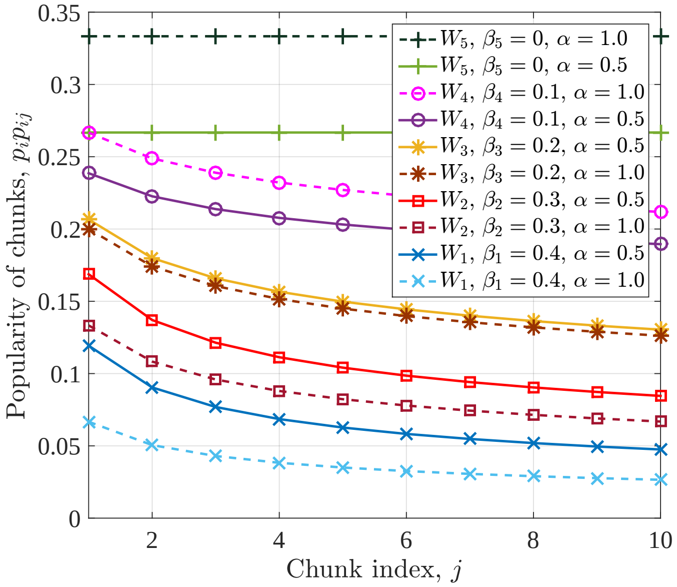

Fig. 1. The popularity of video chunks W_{ij}, i.e., $p_i p_{ij}$ given $\beta_i = 0.5-0.1i$, $i = 1, ..., 5$, $j = 1, ..., 10$.

$$q_{ij} = \begin{cases} q, & \text{if } p_i p_{ij} \ge \bar{n}, \\ 0, & \text{otherwise,} \end{cases} \tag{18}$$

where $q \in (0, 1]$ and $\bar{n}$ are the two parameters to be chosen to satisfy $\sum_{i=1}^{N}\sum_{j=1}^{B} q_{ij} = MB$. We denote the cache content distribution given by (18) as a function of q, i.e., $\mathbf{Q}(q)$. The optimization of q can be expressed as

$$q^* \triangleq \operatorname{argmin} \bar{R}(\mathbf{Q}(q)), \tag{19}$$

which could be computed through one-dimensional search.

IV. Numerical Results

In this section, we numerically evaluate the average delivery rate achieved by the proposed caching scheme with both cache allocation strategies, OCA and PCA, and compare it with the rate achieved by the RAP-GCC scheme proposed in [11] as well as uncoded caching. We consider a network with $K = 5$ users and $N = 5$ video files in the library. Each file consists of $B = 10$ chunks of equal size. We assume that the popularity of files follows a Zipf power law with parameter α [18], in which case we have

$$p_i = \frac{i^\alpha}{\sum_{f=1}^{N} f^\alpha}, \quad i = 1, ..., 5, \tag{20}$$

and the audience retention rates of the video files follow a Zipf-like distribution as well [19], given as:

$$p_{ij} = j^{-\beta_i}, \quad i = 1, ..., 5, \; j = 1, ..., 10, \tag{21}$$

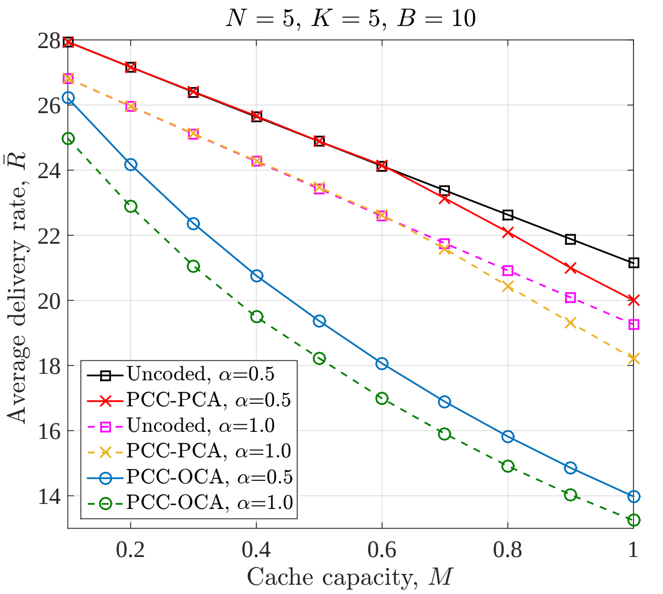

Fig. 2. Comparison between PCC and uncoded caching, $\beta_i = 0.6 - 0.1i$, $\alpha = 0.5$, and $\alpha = 1$.

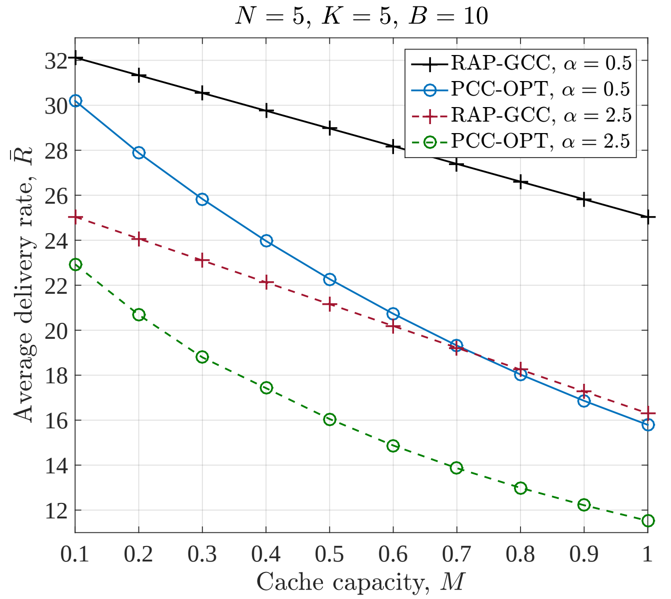

Fig. 3. Comparison between PCC with OCA and RAP-GCC, $\beta_i = 0$, $\alpha = 0.5$ and $\alpha = 2.5$.

with parameter $\beta_i \geq 0$. The larger β_i implies a shorter average watching time for file W_i. We set $\beta_i = 0.5 - 0.1i$, and the corresponding popularity of chunks, i.e., $p_i p_{ij}$, $i = 1, ..., 5$, $j = 1, ..., 10$, are presented in Fig. 1, which shows that the retention rates of the video files with larger β_i will decrease more quickly with the index of the chunks. Moreover, a larger α value results in a bigger difference in the file popularities.

We first compare PCC with uncoded caching, where each user fully caches as many of the most popular chunks as possible to fill its cache capacity. We observe from Fig. 2 that the PCC with OCA significantly reduces the average delivery rate compared to uncoded caching, and the improvement increases with the cache capacity. The PCC with PCA scheme has the same performance as uncoded caching when the cache capacity is small, which implies that the PCA caches the most popular chunks fully to better exploit the limited cache capacity. We can also observe that a larger α results in a smaller average delivery rate since the users tend to request the most popular files, and caching these files is more efficient in reducing the delivery rate.

In Fig. 3, we compare the performance of the PCC with RAP-GCC scheme in [11], which, to the best of our knowledge, is the only result in the literature on the average delivery rate considering heterogeneous file popularities. We set $\beta_i = 0$, $\forall i$, so that $p_{ij} = 1$, $\forall i, j$, and the partial caching problem studied in this paper reduces to the one in [11]. For fair comparison, we optimize the cache content distribution over the bits for the RAP-GCC scheme. It is notable in Fig. 3 that our scheme remarkably outperforms the RAP-GCC, and when α is large, i.e, the popularity distribution of the files is more skewed, and the users are more likely to request the same set of files, the improvement gained from the PCC is even larger. We also observe that the PCC with OCA results in a higher average delivery rate compared to the same setting in Fig. 2, since users always request entire files when $\beta_i = 0$.

V. Conclusions

We have studied coded caching taking into account the audience retention rates for video contents. We assume that each video file in the library consists of a number of chunks of equal size, and the audience retention rate is modeled as the heterogeneous popularity of the chunks of each file. We proposed a coded caching scheme that allocates different cache capacities for different chunks, depending on their popularities. We then evaluated the average delivery rate over all possible demand combinations as well as over the number of chunks requested by each user. We proposed two different methods for cache allocation, namely, the numerically optimized cache allocation scheme OCA, and the low complexity popularity-based cache allocation scheme PCA. The numerical results illustrated a significant improvement with the proposed scheme over uncoded caching in terms of the achievable rate. When the audience retention rates of all the chunks are equal, the problem reduces to the one where users watch entire files with different popularities, and the proposed coded caching scheme is shown to outperform the best known average delivery rate in the literature.

In this paper, we have assumed that the users' requests are synchronized, which is a common assumption in the coded caching literature. A future research direction is the extension of the current model and the proposed caching scheme to an asynchronous demand scenario.

Appendix A
Proof of Theorem 1

We first derive $\bar{m}$ in (11), which is provided by the RANDOM DELIVERY procedure in Algorithm 1. Since each user requesting chunk W_{ij} has $q_{ij}F/B$ bits from this chunk in its cache, according to [7, Appendix A], at most

$$(1-q_{ij})F/B + o(F/B) \tag{22}$$

bits are necessary to enable all the users requesting W_{ij} to decode it. The probability that chunk W_{ij} is requested by at least one user among K active users is given by:

$$\Pr\{i \in \mathcal{D}^j\} = 1-(1-p_i p_{ij})^K, \tag{23}$$

By summing over $i = 1, ..., N$ and $j = 1, ..., B$, we have

$$\bar{R} \leq \sum_{j=1}^{B}\sum_{i=1}^{N}\left(1-(1-p_i p_{ij})^K\right)(1-q_{ij}), \tag{24}$$

which implies (11).

We then prove $\varphi(\mathbf{p}, \mathbf{P}, \mathbf{Q})$ given by (12), which is provided by the CODED DELIVERY procedure in Algorithm 1. We first derive the expected number of bits sent in Part 1. Note that, in this part the server sends the missing bits which are not in the cache of any user. The expected number of bits of chunk W_{ij} that are not cached by any user is given by

$$F/B(1-q_{ij})^K + o(F/B). \tag{25}$$

We denote by φ_1 the expected total number of bits delivered in PART 1 summing over $j = 1, ..., B$. We have:

$$\varphi_1 = \sum_{j=1}^{B}\sum_{i=1}^{N}\left(1-(1-p_i p_{ij})^K\right)(1-q_{ij})^K. \tag{26}$$

Next, we derive the expected total number of bits sent in PART 3 summing over $j = 1, ..., B$, denoted by φ_3. Inspired by the proof in [11, Appendix A], the expected total number of bits sent in PART 3 for jth chunks is found to be:

$$\begin{aligned}\bar{R}^j =& \sum_{l=3}^{K}\binom{K}{l}\sum_{i=1}^{N} g_{ij,l} \\ &\cdot \Pr\Big\{g_{ij,l} \geq g_{fj,l} : i \in \mathcal{D}_l \cap \mathcal{D}^j, \forall f \in \mathcal{D}_l \cap \mathcal{D}^j\Big\}. \end{aligned}\tag{27}$$

The proof of (27) is skipped due to space limitation, and will be provided in a longer version of this paper. Together with (27) and (8), we have

$$\varphi_3 = \sum_{j=1}^{B}\sum_{l=3}^{K}\binom{K}{l}\sum_{i=1}^{N}\rho_{ij,l}(q_{ij})^{l-1}(1-q_{ij})^{K-l+1}, \tag{28}$$

where $\rho_{ij,l}$ is as derived in (16).

We then find the expected total number of bits delivered in PART 2 summing over $j = 1, ..., B$, denoted by φ_2. Following the similar procedure as the proof of (28), the number of bits sent by PART 2.1 is given as

$$\varphi_{2,1} = \sum_{j=1}^{B}\binom{K}{2}\sum_{i=1}^{N}\rho_{ij,2}q_{ij}(1-q_{ij})^{K-1}. \tag{29}$$

Since the expected number of bits of each chunk W_{ij} cached by only one user is

$$F/Bq_{ij}(1-q_{ij})^{K-1} + o(F/B), \tag{30}$$

with $\Pr\{i \in \mathcal{D}_j\}$ given in (23), we have the number of bits sent by PART 2.2 given by

$$\varphi_{2,2} = \sum_{j=1}^{B}\sum_{i=1}^{N}\left(1-(1-p_i p_{ij})^K\right)q_{ij}(1-q_{ij})^{K-1}. \tag{31}$$

We can conclude that $\varphi_2 = \min\{\varphi_{2,1}, \varphi_{2,1}\}$. With (26), (28), (29) and (31), we complete the proof of (12).

References

[1] M. Zeni, D. Miorandi, and F. De Pellegrini, "Youstatanalyzer: a tool for analysing the dynamics of youtube content popularity," in *Proc. of ICST VALUETOOLS*, Torino, Italy, Dec. 2013, pp. 286–289.

[2] 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.

[3] M. Gregori, J. Gomez-Vilardebo, 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, Mar 2016.

[4] M. Mohammadi Amiri and D. Gündüz, "Fundamental limits of coded caching: Improved delivery rate-cache capacity trade-off," *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 806–815, Feb. 2017.

[5] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inform. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

[6] S. O. Somuyiwa, A. György, and D. Gündüz, "Improved policy representation and policy search for proactive content caching in wireless networks," in *Proc. IEEE Int'l Symp. on Modeling and Opt. in Mobile, Ad Hoc, and Wireless Netw. (WiOpt)*, Paris, France, May 2017.

[7] M. A. Maddah-Ali and U. Niesen, "Decentralized caching attains order optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw*, vol. 23, no. 4, pp. 1029–1040, Apr. 2014.

[8] M. Mohammadi Amiri, Q. Yang, and D. Gündüz, "Coded caching for a large number of users," in *Proc. IEEE Inform. Theory Workshop (ITW)*, Cambridge, UK, Sep. 2016.

[9] K. Wan, D. Tuninetti, and P. Piantanida, "On caching with more users than files," *arXiv: 1601.063834v2 [cs.IT]*, Jan. 2016.

[10] J. Gomez-Vilardebo, "Fundamental limits of caching: Improved bounds with coded prefetching," *arXiv:1612.09071v2 [cs.IT]*, Jan. 2017.

[11] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, "Order-optimal rate of caching and coded multicasting with random demands," *arXiv: 1502.03124v1 [cs.IT]*, Feb. 2015.

[12] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," *IEEE Trans. Inform. Theory*, vol. 63, no. 2, pp. 1146–1158, Feb. 2017.

[13] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," in *Proc. IEEE Int'l Conf. Commun. (ICC)*, Sydney, Australia, Jun. 2014, pp. 1878–1883.

[14] Q. Yang and D. Gündüz, "Coded caching and content delivery with heterogeneous distortion requirements," *arXiv:1608.05660v1 [cs.IT]*, Aug. 2016.

[15] L. Maggi, L. Gkatzikis, G. Paschos, and J. Leguay, "Adapting caching to audience retention rate: Which video chunk to store?" *arXiv:1512.03274v1 [cs.NI]*, Dec. 2015.

[16] L. Wang, S. Bayhan, and J. Kangasharju, "Optimal chunking and partial caching in information-centric networks," *Comput. Commun.*, vol. 61, pp. 48–57, May 2015.

[17] M. Mohammadi Amiri, Q. Yang, and D. Gündüz, "Decentralized coded caching with distinct cache capacities," *arXiv:1611.01579v1 [cs.IT]* , Oct. 2016.

[18] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: Evidence and implications," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, NY, Mar. 1999, pp. 126–134.

[19] J. Yu, C. T. Chou, Z. Yang, X. Du, and T. Wang, "A dynamic caching algorithm based on internal popularity distribution of streaming media," *Multimedia Syst.*, vol. 12, no. 2, pp. 135–149, Jul. 2006.