Coded Caching and Content Delivery with Heterogeneous Distortion Requirements

Qianqian Yang, *Student Member, IEEE*, and Deniz Gündüz, *Senior Member, IEEE*

Abstract—Cache-aided coded content delivery is studied for devices with diverse quality-of-service (QoS) requirements, specified by a different average distortion target. The network consists of a server holding a database of independent contents, and users equipped with local caches of different capacities. User caches are filled by the server during a low traffic period without the knowledge of particular user demands. As opposed to the current literature, which assumes that the users request files in their entirety, it is assumed that the users in the system have distinct distortion requirements; and therefore, each user requests a single file from the database to be served at a different distortion level. Our goal in this work is to characterize the minimum *delivery rate* the server needs to transmit over an error-free shared link to satisfy all possible demand combinations at the requested distortion levels, considering both *centralized* and *decentralized* cache placement.

For centralized cache placement, the optimal delivery rate is characterized for the two-file two-user scenario for any pair of target distortion requirements, when the underlying source distribution is successively refinable. For the two-user scenario with more than two successively refinable files, the optimal scheme is characterized when the cache capacities of the users are the same and the number of files is a multiple of 3. For the general source distribution, not necessarily successively refinable, and with arbitrary number of users and files, a *layered caching and delivery scheme* is proposed, assuming that scalable source coding is employed at the server. This allows dividing the problem into two subproblems: the lossless caching of each layer with heterogeneous cache sizes, and cache allocation among layers. A delivery rate minimization problem is formulated and solved numerically for each layer; while two different schemes are proposed to allocate user caches among layers, namely, *proportional cache allocation (PCA)* and *ordered cache allocation (OCA)*. A decentralized lossy coded caching scheme is also proposed, and its delivery rate performance is studied. Simulation results validate the effectiveness of the proposed schemes in both settings.

I. Introduction

Consumer demand for mobile video services is growing at an unprecedented rate. This trend is expected to continue in the coming years driven by the proliferation of mobile devices with high quality display capabilities, and the explosion of high data rate multimedia contents available online. The majority of video traffic today is generated through "video surfing", i.e., streaming of video files stored in large databases, e.g., YouTube, Hulu, Dailymotion, BBC iPlayer, etc. Such video traffic is dominated by a relatively small number of viral contents that remain popular over a certain period of time, and downloaded repeatedly by many users, sometimes connected through the same access point, creating a huge amount of repetitive traffic both on the core and the radio access networks.

Caching has long been used in the Internet to reduce traffic, as well as latency (see [1], and references therein). More recently, research on content caching has regained popularity, targeting mainly wireless networks (see [2]–[4], and references therein). While content caching at the evolved packet core, or at the radio access network, can reduce traffic and latency on the backhaul links [3], [5], caching contents directly at user devices can bring further benefits [4], [6]. The latter strategy is called *proactive caching* since caching at a user device requires predicting the demand, and delivering the content even before it is requested by the user. Since mobile data traffic at wireless access points exhibits a high degree of variation across time, exploiting the radio resources during low traffic periods through proactive caching will also reduce the peak traffic rates; and therefore, improve the quality-of-experience (QoE) for users, and reduce infrastructure costs for network providers. Feasibility of proactive caching in future wireless networks is further supported by the low-cost and abundance of storage space in today's mobile devices.

In the proactive caching model considered here, users fill their caches during the off-peak traffic period, referred to as the *placement phase*. Each user's cache content at the end of the *placement phase* is a function of the whole database. User requests, one file per user, are revealed during the peak traffic period, and satisfied simultaneously during the *delivery phase*. Conventional *uncoded caching* schemes store contents, partially or fully, at each user's cache, and utilize orthogonal unicast transmissions during the delivery phase. The gain from uncoded caching for each user depends only on the local cache capacity. Recently, Maddah-Ali and Niesen introduced a novel model for cache networks [6], particularly appropriate for proactive caching in wireless networks, in which the delivery phase is carried out over a shared link, modeling the broadcast nature of wireless communications. They show that *centralized coded caching*, in which coded bits, rather than plain bits of the available contents in the database, are cached and delivered, achieves significant gains compared to uncoded caching. This gain is realized by jointly optimizing the placement and the delivery phases to create multicasting opportunities even among distinct requests [6]. In contrast to the *centralized setting* in [6], where the active users are

known in advance, [8] considers the so-called *decentralized setting*, in which, during the *placement phase*, the server has no prior knowledge on the number and identity of users that will participate in the *delivery phase*. It is shown that the multicasting opportunities still appear even if users randomly cache bits of files [8].

Numerous papers followed [6] and [8] in order to further improve the *coded caching* gain, and to apply it to various other network models. Chen et al. [9] achieve the optimal delivery rate for small buffer sizes by placing coded contents into users' caches during the *placement phase*. When the number of users is larger than the number of files, improved delivery rates are obtained in [10], [11]. Pedarsani et al. introduce a coded least-recently sent delivery and update rule that replaces the cache content during the delivery phase for online caching systems [12]. A multi-layer caching system, in which user terminals, proxies, base stations are all equipped with cache memories, is considered in [13]. A distributed caching system is investigated in [14] with single-hop device-to-device communication, which shows that coded caching has the same scaling law as the spatial reuse of user caches. Delivery over a noisy broadcast channel is considered in [15]–[18]. Similarly, delivery of contents over an interference channel is considered in [19]–[22], where both the transmitters and receivers have caches.

Common to the aforementioned works and most of the other follow-up papers in the literature, is the assumption that the files in the database have fixed sizes, and each user requests one of these files in whole. However, in practice, video contents are usually downloaded at different quality levels by users, which may be due to their viewing preferences, or the display and processing capabilities of their devices. For example, a laptop may require high quality descriptions of requested files, whereas a mobile phone is satisfied with much lower resolution. In current video coding standards, diverse reconstruction capacities and demands of users is handled through scalable video coding (SVC). The H.264/MPEG-4 standard [23], [24] allows temporal (frame rate), spatial (picture size) or SNR/quality scalability. This is achieved by encoding the videos into multiple bit streams, i.e., substreams, such that the more substreams users receive, the higher the corresponding resolution is. In this work, we consider users with heterogeneous quality-of-service (QoS) requirements, that is, each user, instead of requesting a file in the database in full, may request a lower resolution copy. Accordingly, we exploit scalable compression of the files available in the database. This provides flexibility to the server not only in supporting the multiple quality levels requested by the users, but also in exploiting the different cache capacities of the users.

In particular, we consider the lossy version of the coded caching problem, such that each user has a preset average distortion requirement. This distortion target is user-dependent, and is the same for any file the user may request. In centralized caching, it is assumed that the server knows the distortion requirements of all the users in the system during the *placement phase*, whereas no such knowledge is needed in decentralized caching. Given the cache capacities and the distortion requirements of the users, the objective of the server is to design the placement and the delivery phases jointly, in order to minimize the delivery rate while guaranteeing that all possible demand combinations can be satisfied at the required distortion levels.

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

- The model studied in this paper generalizes the original proactive caching model introduced in [6] in two directions: the users are equipped with cache memories of different capacities, and each user may have a different QoS requirement, which translates into a different average distortion target.
- We derive a theoretical lower bound on the delivery rate based on cut-set arguments.
- For the centralized lossy caching problem, we characterize the optimal delivery rate for the two-file two-user case when the underlying distribution of the files is successively refinable for the desired distortion measure.
- We propose a centralized coded caching scheme for the case with two users and an arbitrary number of files, which is proven to be optimal for a successively refinable distribution, when the cache capacities of the users are the same, and the number of files is a multiple of 3.
- For the general case, we propose a coded caching algorithm based on scalable coding of the files in the database into as many layers as the number of different distortion requirements. We then divide the problem into two subproblems: the lossless caching problem of each layer with heterogeneous cache sizes, and the cache allocation problem among different layers. The first subproblem is formulated as an optimization problem using the achievable delivery rates available in the literature [6], [9], [10], and solved numerically. Then, two cache allocation algorithms, i.e., *proportional cache allocation* and *ordered cache allocation*, are proposed, and their performances are compared with each other and the theoretical lower bound through numerical simulations.
- We propose a coded caching scheme for the *decentralized* lossy caching problem, and derive its delivery rate.

The most related work to this paper is [25], which exploits the decentralized coded caching scheme proposed in [26], and optimizes the allocation of cache capacities to minimize the average distortion across users, constrained by the delivery rate over the shared link and the cache capacities of the users. Scalable coding of the contents is also considered in [25]. Instead of scalable coding, Cacciapuoti et al. consider multiple description coding in [7], where the reconstruction quality depends solely on the number of received descriptions, irrespective of the order of these descriptions. The authors introduce a channel-aware caching scheme where the reconstruction qualities across users are decided according to the cache configuration and channel states. In [27], Timo et al. also consider lossy caching, taking into account the correlation among the available contents, based on which the tradeoff between the compression rate, reconstruction distortion and cache capacity is characterized for a single-user scenario, as well as some special cases of the two-user scenario.

The rest of the paper is organized as follows. We present the system model and the problem formulation in Section II. Centralized coded caching is studied in Section III, and a theoretical lower bound as well as achievable schemes are proposed. Decentralized lossy coded caching is considered in Section IV. Numerical results are presented in Section V. Finally, we conclude the paper in Section VI, followed by the appendices.

II. SYSTEM MODEL

We consider a server holding N independent source sequence, S_1^n, ..., S_N^n, which may correspond to N video files. Each S_i^n sequence consists of n independent and identically distributed (i.i.d) source samples $(S_{i,1}, ..., S_{i,n})$, for $i = 1, ..., N$. There are K users in the system that may request any of the files from the server. The operation of the caching system consists of two distinct phases: *placement phase* and *delivery phase*. In the *placement phase*, users pre-fetch bits from the server to fill their caches. In *centralized caching*, active users that will participate in the *delivery phase* are known in advance during the *placement phase*, enabling coordination of cache placement across users. On the contrary, in *decentralized caching*, users fill their caches from different servers through different access points, and equivalently, the server has no prior knowledge of the active users that will participate in a particular delivery phase. Therefore, the cache placement of each user is conducted independently.

We assume that each user is equipped with a cache of size $M_k n$ bits, $k = 1, ..., K$, and denote by Z_k the contents of the cache of user k at the end of the *placement phase*. The *delivery phase* starts after users reveal their demands, denoted by $\mathbf{d} \triangleq (d_1, ..., d_K)$, where $d_k \in \{1, ..., N\}$ denotes the demand of user k. During this phase, a single message $X_{(d_1,...,d_K)}^n$ of size nR bits is sent by the server over the shared link depending on all the users' requests and cache contents. User k reconstructs its requested file by combining Z_k and $X_{(d_1,...,d_K)}^n$.

An $(n, M_1, ..., M_K, R)$ "caching code" consists of K cache placement functions:

$$f_k^n : \underbrace{\mathbb{R}^n \times \cdots \times \mathbb{R}^n}_{N \ files} \to \{1, ..., 2^{nM_k}\} \ \text{ for } \ k = 1, ..., K, \quad (1)$$

one delivery function:

$$g^n : \underbrace{\mathbb{R}^n \times ... \times \mathbb{R}^n}_{N \ files} \times \underbrace{\{1, ..., N\}^K}_{K \ requests} \to \{1, ..., 2^{nR}\}, \quad (2)$$

where $Z_k^n = f_k^n(S_1^n, ..., S_N^n)$, $X_{(d_1,...,d_K)}^n = g^n(S_1^n, ..., S_N^n, d_1, ..., d_K)$, and K decoding functions:

$$h_k^n : \{1, ..., N\}^K \times \{1, ..., 2^{nM_k}\} \times \{1, ..., 2^{nR}\} \to \mathbb{R}^n, \quad (3)$$

where $\hat{S}_k^n = h_k^n(\mathbf{d}, Z_k^n, X^n)$. Note that, in this formulation the demand vector $\mathbf{d}$ is known by all the users.

The distortion between the original sequence S_i^n and its reconstruction $\hat{S}_i^n$ is measured by the same distortion function $d(\cdot, \cdot)$ at all receivers:

$$d(S_i^n, \hat{S}_i^n) = \frac{1}{n} \sum_{j=1}^{n} d(S_{ij}, \hat{S}_{ij}), \quad (4)$$

where $d : \mathbb{R} \times \mathbb{R} \to [0, \infty)$ is a per-letter distortion measure. We assume that each user has a preset distortion requirement D_k, $k = 1, ..., K$. Without loss of generality, we assume that $D_1 \geq D_2 \geq \cdots \geq D_K$. We emphasize that the distortion tuple $\mathbf{D} \triangleq (D_1, ..., D_K)$ is known during the *placement phase*, while the demand realization, $\mathbf{d}$, is unknown.

Definition 1. *A distortion tuple $\mathbf{D}$ is achievable if there exists a sequence of caching codes $(n, M_1, ..., M_K, R)$, such that*

$$\limsup_{n \to \infty} \mathbb{E}\Big[d(S_{d_k}^n, \hat{S}_{d_k}^n)\Big] \leq D_k, \quad k = 1, 2, ..., K, \quad (5)$$

holds for all possible request combinations $\mathbf{d}$, where the expectation is with respect to the distribution of S_{d_k}.

Definition 2. *For a given distortion tuple $\mathbf{D} \triangleq (D_1, ..., D_K)$, the cache capacity-delivery rate tradeoff is defined as follows:*

$$R^\star(M_1, ..., M_K) \triangleq \inf\{R : \mathbf{D} \text{ is achievable}\}. \quad (6)$$

For ease of exposure, we assume that all the files in the library follow the same distribution such that the compression rate for a fixed distortion target, that is, the rate-distortion function, is identical for all the files. We denote by r_k^* the minimum compression rate that achieves an average distortion of D_k, which is given as follows, in a single-letter form,

$$r_k^* \triangleq R(D_k) = \min_{\mathbb{E}[d(S_i, \hat{S}_i)] \leq D_k} I(S_i, \hat{S}_i), \qquad \forall i. \quad (7)$$

Our goal in this paper is to characterize the *cache capacity-delivery rate tradeoff* for a caching system with N files and K users with arbitrary distortion requirements for both the centralized and decentralized scenarios.

Remark 1. *We note here that, when all the users have the same distortion target, our problem reduces to the classical setting of [6] and [8], in which users request files in their entirety. We can consider that files are stored in the server in their compressed form. We also highlight that, in addition to allowing requests at different QoS levels, we extend [6] and [8] in another direction by allowing different cache capacities at the users.*

Remark 2. *To illustrate the richness of this model, consider the special case of a library with a single file, and two users, where only one user is equipped with a cache memory. This is equivalent to the well-known successive refinement problem, where the delivery phase corresponds to the base layer, while the placement phase, i.e., the cache content corresponds to the refinement layer.*

To further simplify the problem, we will assume that the server employs scalable coding, also known as successive refinement [29]; that is, each source sequence is encoded into multiple layers, each layer targeting a reduced distortion requirement. The base layer contains the least amount of details which enables a lowest-quality reconstruction of each video, while the following layers, referred to as the *enhancement layers*, can refine the quality of reconstruction successively. We remark that an enhancement layer is useful to a user only if this user has all the preceding enhancement layers and the base layer.

Equivalently, a scalable code maps the required QoS level of each user to a different code layer, where the first layer of each source sequence, consisting of r_1 bits per source sample (bpss) and referred to as r_1-*description*, provides an average distortion of D_1 when decoded successfully, while the kth layer, $k = 2, ..., K$, consists of $r_k - r_{k-1}$ bpss, and having received the first k layers, referred to as r_k-*description*, a user achieves an average distortion of D_k.

Remark 3. *Here we assume for simplicity that all the files in the database have the same distribution, which, in turn, leads to all the compressed files for a fixed distortion target to have the same rate, i.e., the same size. While this greatly simplifies the exposition, extension of the results to the more general scenario with files with different distributions is possible by zero-padding shorter codewords before multicasting, similarly to [31].*

Remark 4. *We remark here that with scalable coding we may not be able to meet the optimal rate-distortion function at each layer, that is, in general, $r_k^* \neq r_k$, since scalable coding may introduce coding overheads to allow various trade-offs among the rates of different layers. However, for some sources and under certain distortion measures scalable coding does not introduce rate loss, called successive refinability. For example, any finite alphabet source with Hamming distortion and Gaussian sources with square-error distortion are examples of successive refinable source and distortion measure pairs [25], [29].*

III. CENTRALIZED CODED CACHING WITH DISTORTION REQUIREMENTS

In this section, we investigate the lossy caching problem in the centralized setting. We start by presenting a theoretical lower bound on the delivery rate of the caching system described in Section II. We will then consider achievable lossy caching schemes, first for the two-user two-file ($N = K = 2$) scenario, then for the general two-user scenario, and finally for an arbitrary number of users and files.

A. Theoretical Lower Bound

We provide a lower bound on the delivery rate for the general setting (K users and N files) based on the cut-set arguments in Theorem 1.

Theorem 1. *(Cut-set Bound) For the lossy caching problem described in Section II, the optimal achievable delivery rate is lower bounded by*

$$R^\star(M_1, ..., M_K)$$
$$\geq \max_{s \in \{1, ..., \min\{N, K\}\}} \max_{\substack{\mathcal{U} \subset \{1, ..., K\} \\ |\mathcal{U}| = s}} \left(\sum_{k \in \mathcal{U}} r_k^* - \frac{\sum_{k \in \mathcal{U}} M_k}{\lfloor N/s \rfloor} \right). \quad (8)$$

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

It is known that the cut-set bound is not tight in general for the centralized coded caching problem [6]. We present another bound for the two-user case ($K = 2$), which, together with the

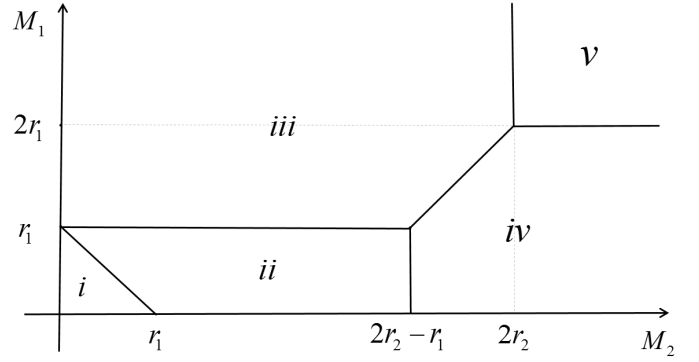Fig. 1. Illustration of the five distinct cases of the cache capacities, M_1 and M_2, depending on the distortion requirements of the users, r_1 and r_2 ($N = K = 2$).

cut-set bound, provides a tight lower bound on the delivery rate in certain scenarios.

Theorem 2. *For the lossy caching problem described in Section II with $K = 2$, we have*

$$R^\star(M_1, M_2) \geq \frac{r_1^*}{2} + r_2^* - \frac{(M_1 + M_2)}{2\lfloor N/2 \rfloor}. \quad (9)$$

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

B. Optimal Lossy Caching: Two Users and Two Files ($N = K = 2$)

In this section, we characterize the optimal cache capacity-delivery rate tradeoff for the lossy caching problem with two users ($K = 2$) and two files ($N = 2$), assuming that the underlying source distribution is *successively refinable*. We first present the lower bound on the delivery rate for given M_1 and M_2 in this particular scenario, followed by the coded caching scheme achieving this lower bound.

Corollary 1. *For the lossy caching problem with $N = K = 2$, a lower bound on the cache capacity-delivery rate tradeoff is given by*

$$R^\star(M_1, M_2) \geq R_c(M_1, M_2) \triangleq \max\{r_1^* - M_1/2,$$
$$r_2^* - M_2/2, r_1^* + r_2^* - (M_1 + M_2),$$
$$r_1^*/2 + r_2^* - (M_1 + M_2)/2, 0\}. \quad (10)$$

The first three terms in RHS of (10) are derived from the cut-set bound in Theorem 1, and the forth term follows from Theorem 2.

Now we consider the achievability, assuming that a scalable code is employed to compress each file into two layers, where the first layer of rate r_1 allows achieving a distortion of D_1, while the second layer of rate $r_2 - r_1$, together with the first layer results in distortion D_2. We denote by $A(B)$ the source codeword of length nr_2 bits that can achieve a distortion of D_2 for file $S_1^n(S_2^n)$. We refer to the first nr_1 bits as the *base layer*, and the remaining $n(r_2 - r_1)$ bits as the *refinement layer*. We consider the following five cases depending on the cache capacities of the users, illustrated in Fig. 1: *Case i*, $M_1 + M_2 \leq r_1$; *Case ii*, $M_1 + M_2 > r_1$, $M_1 \leq r_1$, $M_2 \leq 2r_2 - r_1$; *Case*

iii, $M_1 > r_1$, $M_2 \leq 2r_2$, $M_2 - M_1 \leq 2r_2 - 2r_1$; *Case iv*, $M_1 \leq 2r_1$, $M_2 > 2r_2 - r_1$, $M_2 - M_1 > 2r_2 - 2r_1$; *Case v*, $M_1 > 2r_1$, $M_2 > 2r_2$.

Before presenting the caching and delivery schemes for each of the above cases in detail, we briefly review two different techniques that have been considered in the literature for loseless coded caching systems. Assume that there are two files W_1 and W_2 in the library, and each of the two users requests one of these files fully. In the placement phase we divide each file into two equal-size non-intersecting sub-files, i.e., we have $W_1 = (W_{11}, W_{12})$ and $W_2 = (W_{21}, W_{22})$. When the cache sizes of the users are small, the users can cache XORed subfiles. For example, user 1 caches $W_{11} \oplus W_{21}$, while user 2 caches $W_{12} \oplus W_{22}$. Now, consider that user i requests file W_i, $i = 1, 2$. To satisfy the demands, the server transmits W_{21} and W_{12}, which are simultaneously useful to both users. Note that, for cache capacities $M_1 = M_2 = 1/2$, we achieved a delivery rate of $R = 1$. This method, referred to as *coded placement*, is first proposed in [6] for $N = K = 2$, and then generalized by [9]. Alternatively, if the users have larger caches, e.g., $M_1 = M_2 = 1$, they can cache disjoint subfiles. For example, user 1 caches W_{11}, W_{21}, while user 2 caches W_{12}, W_{22}. The same demand combination as above can be satisfied by simply transmitting $W_{21} \oplus W_{12}$ to the users which again benefits both users. This method, also proposed by Maddah-Ali and Niesen in [6], is referred to as *coded delivery*. The optimal caching scheme for the two-user two-file scenario for the lossless caching problem with the same cache sizes, i.e, $M_1 = M_2$, has been derived by memory sharing between these two methods and the two extreme cases corresponding to no user cashes ($M_1 = M_2 = 0$) and users caching both files ($M_1 = M_2 = 2$).

For the lossy caching problem studied in this paper, where the two users have different distortion requirements and different cache sizes, to fully exploit these two methods, we divide the first layers of codewords A and B into six disjoint parts denoted by A_1, ..., A_6 and B_1, ..., B_6, respectively, where the XORed combination of A_1 and B_1, i.e., $A_1 \oplus B_1$, is cached by user 1, the XORed combination of A_2 and B_2, i.e., $A_2 \oplus B_2$, is cached by user 2, A_3 and B_3 are cached exclusively by user 1, A_4 and B_4 are cached exclusively by user 2, A_5 and B_5 are cached by both user 1 and user 2, and A_6 and B_6 are cached by none of the users. Since only user 2 requires the refinement layers, we divided the refinement layers of A and B into two disjoint parts denoted by A_7, A_8 and B_7, B_8, respectively, where A_7 and B_7 are cached by user 2, and A_8 and B_8 are not cached by any user. Overall, user 1 caches $Z_1 = (A_1 \oplus B_1, A_3, B_3, A_5, B_5)$, while user 2 caches $Z_2 = (A_2 \oplus B_2, A_4, B_4, A_5, B_5, A_7, B_7)$. Let $|A_i| = |B_i|$ for $i = 1, ..., 8$, and $|A_3| = |A_4|$ where $|X|$ denotes the length of the binary sequence X (normalized by n).

Table I illustrates the placement of contents into users' caches for the five cases by specifying the size of each portion in each case. For example, the forth row implies that in *Case i*, $|A_1| = |B_1| = M_1$, $|A_2| = |B_2| = M_2$, $|A_6| = |B_6| = r_1 - M_1 - M_2$, $|A_8| = |B_8| = r_2 - r_1$, and the sizes of all other portions are equal to 0, which is equivalent to dividing $A(B)$ into four portions $A_1(B_1)$,

$A_2(B_2)$, $A_6(B_6)$ and $A_8(B_8)$. Thus, in the placement phase, user 1 caches $Z_1 = A_1 \oplus B_1$, and user 2 caches $Z_2 = A_2 \oplus B_2$ so that $|Z_1| = M_1$ and $|Z_2| = M_2$, which meets the cache capacity constraints. For *Case ii*, as presented in the fifth row, $|A_1| = |B_1| = M_1$, $|A_2| = |B_2| = r_1 - M_2$, $|A_7| = |B_7| = \frac{M_1 + M_2 - r_1}{2}$, $|A_8| = |B_8| = r_2 - r_1 - \frac{M_1 + M_2 - r_1}{2}$, and the sizes of all other portions are equal to 0, which is equivalent to dividing $A(B)$ into four portions $A_1(B_1)$, $A_2(B_2)$, $A_7(B_7)$ and $A_8(B_8)$. Thus, user 1 caches $Z_1 = A_1 \oplus B_1$, and user 2 caches $Z_2 = \{A_2 \oplus B_2, A_7, B_7\}$ so that $|Z_1| = M_1$ and $|Z_2| = M_2$, which meets the cache capacity constraints. The cache placements for the other three cases are presented in a similar manner in Table I.

Next, we focus on the delivery phase, and identify the minimum delivery rate in each case to satisfy the demands, $\mathbf{d} = (S_1^n, S_2^n)$. Other demand combinations can be satisfied similarly, without requiring higher delivery rates.

Case i ($M_1 + M_2 \leq r_1$): The server sends B_1, A_2, A_6, B_6 and B_8. Thus, the delivery rate is $R(M_1, M_2) = r_1 + r_2 - (M_1 + M_2)$.

Case ii ($M_1 + M_2 > r_1$, $M_1 \leq r_1$, $M_2 \leq 2r_2 - r_1$): The server sends B_1, A_2 and B_8. We have $R(M_1, M_2) = \frac{r_1}{2} + r_2 - \frac{M_1 + M_2}{2}$.

Case iii ($M_1 > r_1$, $M_2 \leq 2r_2$, $M_2 - M_1 \leq 2r_2 - 2r_1$): The values of l_1, l_2 and l_3 in Table I are given as:

$$l_1 = \max\{0, \min\{M_1 - r_1, M_2/2 - (r_2 - r_1)\}\}; \quad (11a)$$

$$l_2 = \max\{0, M_2/2 - (r_2 - r_1) - l_1\}; \quad (11b)$$

$$l_3 = \min\{r_2 - r_1, M_2/2\}. \quad (11c)$$

The server sends B_1, $B_3 \oplus A_4$ and B_8, which results in $R(M_1, M_2) = r_2 - \frac{M_2}{2}$.

Case iv ($M_1 \leq 2r_1$, $M_2 > 2r_2 - r_1$, $M_2 - M_1 > 2r_2 - 2r_1$): The server sends B_2, $B_3 \oplus A_4$ and we have $R(M_1, M_2) = r_1 - \frac{M_1}{2}$.

Case v ($M_1 > 2r_1$, $M_2 > 2r_2$): The cache capacities of both users are sufficient to cache the required descriptions for both files. Thus, any request can be satisfied from the local caches at the desired distortion levels, and we have $R(M_1, M_2) = 0$.

Theorem 3. *For $N = K = 2$, and a successively refinable source distribution, we have $R^*(M_1, M_2) = R_c(M_1, M_2)$, i.e., the proposed coded caching scheme meets the lower bound in Corollary 1; and hence, it is optimal.*

Proof. Theorem 3 follows by setting $r_1 = r_1^*$ and $r_2 = r_2^*$ in the above achievable scheme. We note that the achieved delivery rates match those of the lower bound in Corollary 1 for each of the five cases. □

Remark 5. *The optimality of the proposed scheme arises from the maximization of the multicasting opportunities in the discussed five cases. For example, in Case i, by placing disjoint XORed contents from the base layers into the caches of both users, $M_1 + M_2$ bits sent in the delivery phase is simultaneously useful to both users regardless of any demand combination. This corresponds to the maximal multicasting opportunities that can be created by any caching scheme as*

TABLE I
ILLUSTRATION OF CACHE PLACEMENT FOR $N = K = 2$

	Base Layer						Refinement Layer	
S_1	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
S_2	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
Case i	M_1	M_2	0	0	0	$r_1 - M_1 - M_2$	0	$r_2 - r_1$
Case ii	M_1	$r_1 - M_1$	0	0	0	0	$\frac{M_1 + M_2 - r_1}{2}$	$r_2 - r_1 - \frac{M_1 + M_2 - r_1}{2}$
Case iii	$r_1 - l_1 - 2l_2$	0	l_2	l_2	l_1	0	l_3	$r_2 - r_1 - l_3$
Case iv	0	$r_1 - M_1$	$M_1/2$	$M_1/2$	0	0	$r_2 - r_1$	0
Case v	0	0	0	0	r_1	0	$r_2 - r_1$	0

validated by the lower bound on the cache capacity-delivery rate tradeoff given in Corollary 1.

Remark 6. *In the special case of identical distortion requirements at the two users, i.e., when $D_1 = D_2$, Theorem 3 generalizes the optimal delivery rate result of [6] for $N = K = 2$ to different cache capacities.*

C. Lossy Caching: Two Users and N Files ($K = 2, N > 2$)

Next, we investigate the more general case with two users and an arbitrary number of files, i.e., $K = 2, N > 2$. We first present a lower bound on the delivery rate, in Lemma 1, and then present a coded caching scheme, followed by the analysis of the gap between the two.

Lemma 1. *For the lossy caching problem with $K = 2$ and $N > 2$, a lower bound on the cache capacity-delivery rate tradeoff is given by*

$$R^\star(M_1, M_2) \geq R_c(M_1, M_2) \triangleq \max \left\{ r_1^* - \frac{M_1}{N}, r_2^* - \frac{M_2}{N}, \right.$$
$$\left. \frac{r_1^*}{2} + r_2^* - \frac{M_1 + M_2}{2\lfloor N/2 \rfloor}, r_1^* + r_2^* - \frac{M_1 + M_2}{2\lfloor N/3 \rfloor}, 0 \right\}. \quad (12)$$

The first two terms in RHS of (12) follow from the cut-set bound in Theorem 1, while the third term from Theorem 2. The proof of the forth term can be found in Appendix C.

Next, we present a coded caching scheme for this scenario. Similarly to Section III-B, we employ scalable coding to compress the files. We denote by W_j the source codeword of length nr_2 bits that leads to a distortion of D_2 for file S_j^n, $j = 1, ..., N$. First nr_1 bits of W_j corresponds to the base layer that would provide a distortion level of D_1 if received. Note that, since $N > K$, *coded placement*, which places XORed contents into users' caches as described in Section III-B, no longer creates multicasting opportunities in this scenario. Hence, we exploit only *coded delivery*, where users cache distinct uncoded subfiles, and XORed subfiles are sent in the delivery phase as described in Section III-B. Accordingly, we divide the base layer of W_j, i.e., the first nr_1 bits, into four disjoint parts, denoted by W_{j1}, W_{j2}, W_{j3}, and W_{j4}, and let user 1 cache W_{j1} and W_{j2}, while user 2 caches W_{j2} and W_{j3}, $j = 1, ..., N$. Hence, in the delivery phase, for any demand pair, the server sends $W_{d_13} \bar\oplus W_{d_21}$, W_{d_14}, W_{d_24}, to enable both users to obtain the base layers of their required files. Here $\oplus$ represents the bitwise XOR operation, where the arguments are first zero-padded to the length of the longest one. The refinement layer, i.e., the remaining $n(r_2 - r_1)$ bits

of W_j, is divided into two disjoint parts, denoted by W_{j5}, W_{j6}, $j = 1, ..., N$. User 2 caches W_{j5}, while user 1 does not cache any part of the refinement layers. In the delivery phase, the server sends W_{d_26} to enable user 2 to obtain the second layer of its requested file. Let the size of each part be the same for all the files, e.g., $|W_{ik}| = |W_{jk}|, \forall i, j \in \{1, ..., N\}$, for $k = 1, ..., 6$. To conclude, we have $Z_1 = \bigcup_{j=1}^{N} \{W_{j1}, W_{j2}\}$ and $Z_2 = \bigcup_{j=1}^{N} \{W_{j2}, W_{j3}, W_{j5}\}$. By sending $W_{d_13} \bar\oplus W_{d_21}$, W_{d_14}, W_{d_24}, and W_{d_26}, for any possible demand pair (d_1, d_2), in the delivery phase, both users can achieve their target distortion values.

In the following, we specify the size of each portion depending on the cache capacities M_1 and M_2, and derive the corresponding delivery rate. We note that the multicasting opportunities only come from W_{j1} and W_{j3} (by sending $W_{d_13} \bar\oplus W_{d_21}$ in the delivery phase as described above). Hence, our goal is to maximize $W_{d_13} \bar\oplus W_{d_21}$ in order to minimize the delivery rate. We consider three cases:

Case i ($M_1 + M_2 \leq Nr_1$): We let: $|W_{j1}| = M_1/N$; $|W_{j2}| = 0$; $|W_{j3}| = M_2/N$; $|W_{j4}| = r_1 - M_1/N - M_2/N$; $|W_{j5}| = 0$; $|W_{j6}| = r_2 - r_1$. The delivery rate is $R(M_1, M_2) = r_1 + r_2 - \frac{2(M_1 + M_2)}{N} + \frac{\max\{M_1, M_2\}}{N}$.

Case ii ($M_1 + M_2 > Nr_1$, $M_1 \leq Nr_1$): We let: $|W_{j1}| = \max\{\min\{r_2 - M_2/N, M_1/N\}, 0\}$; $|W_{j2}| = \max\{M_1/N + M_2/N - r_2, 0\}$; $|W_{j3}| = r_1 - M_1/N$; $|W_{j4}| = 0$; $|W_{j5}| = \min\{M_1/N + M_2/N - r_1, r_2 - r_1\}$; $|W_{j6}| = \max\{r_2 - (M_1/N + M_2/N), 0\}$. We have $R(M_1, M_2) = \max\{r_2 - (M_1 + M_2)/N, 0\} + \max\{\min\{r_2 - M_2/N, M_1/N\}, r_1 - M_1/N\}$.

Case iii ($M_1 > Nr_1$): Let $|W_{j1}| = r_1 - \min\{M_2/N, r_1\}$; $|W_{j2}| = \min\{M_2/N, r_1\}$; $|W_{j3}| = 0$; $|W_{j4}| = 0$; $|W_{j5}| = \max\{0, \min\{r_2, M_2/N\} - r_1\}$; $|W_{j6}| = \min\{r_2 - r_1, \max\{0, r_2 - M_2/N\}\}$. It yields $R(M_1, M_2) = \max\{0, r_2 - M_2/N\}$.

We can summarize the achievable delivery rate as follows:

$$R(M_1, M_2)$$
$$= \begin{cases} r_1 + r_2 - 2M_1/N - M_2/N, & \text{if } (M_1, M_2) \in \mathcal{M}_1, \\ r_1 + r_2 - M_1/N - 2M_2/N, & \text{if } (M_1, M_2) \in \mathcal{M}_2, \\ r_1 - M_1/N, & \text{if } (M_1, M_2) \in \mathcal{M}_3, \\ r_2 - M_2/N, & \text{if } (M_1, M_2) \in \mathcal{M}_4, \\ 0, & \text{if } (M_1, M_2) \in \mathcal{M}_5, \end{cases}$$
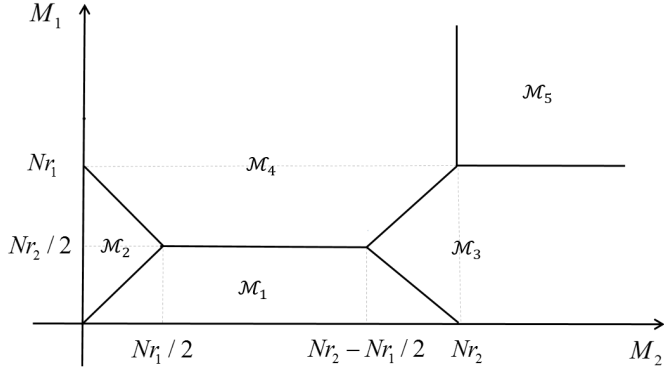$$(13)$$

Fig. 2. Illustration of the five distinct cases of the cache capacities, M_1 and M_2, depending on the distortion requirements of the users, r_1 and r_2 ($K = 2, N > 2$).

where $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, $\mathcal{M}_4$ and $\mathcal{M}_5$, illustrated in Fig. 2, are specified as follows:

$$\mathcal{M}_1 = \{(M_1, M_2) | 0 \leq M_1 \leq Nr_1/2, M_1 + M_2 \leq Nr_2,$$
$$M_1 \leq M_2\}; \quad (14a)$$
$$\mathcal{M}_2 = \{(M_1, M_2) | M_1 + M_2 \leq Nr_1, M_1 > M_2, M_2 \geq 0\}; \quad (14b)$$
$$\mathcal{M}_3 = \{(M_1, M_2) | 0 \leq M_1 \leq Nr_1, M_1 + M_2 > Nr_2,$$
$$M_2 - M_1 > N(r_2 - r_1)\}; \quad (14c)$$
$$\mathcal{M}_4 = \{(M_1, M_2) | M_1 > Nr_1/2, 0 \leq M_2 \leq Nr_2,$$
$$M_1 + M_2 > Nr_1, M_2 - M_1 \leq N(r_2 - r_1)\}; \quad (14d)$$
$$\mathcal{M}_5 = \{(M_1, M_2) | M_1 > Nr_1, M_2 > Nr_2\}. \quad (14e)$$

If the underlying source distribution is successively refinable, i.e., $r_1 = r_1^*$ and $r_2 = r_2^*$, comparing (12) with (13), we can conclude that for $(M_1, M_2) \in \mathcal{M}_3 \bigcup \mathcal{M}_4 \bigcup \mathcal{M}_5$, the achieved delivery rate meets the lower bound in Lemma 1. We analyse the optimality of the achieved delivery rate for $(M_1, M_2) \in \mathcal{M}_1 \bigcup \mathcal{M}_2$ in the following, assuming successive refinability, i.e., $r_1 = r_1^*$ and $r_2 = r_2^*$. Consider three cases: $N = 3c + i$, for $i = 0, 1, 2$ and $c \in \mathbb{Z}^+$.

Case 1 ($N = 3c$): The last term of (12) can be rewritten as $r_1^* + r_2^* - \frac{3(M_1 + M_2)}{2N}$. Thus, $\Delta R \triangleq R(M_1, M_2) - R_c(M_1, M_2) = \left| \frac{M_1 - M_2}{2N} \right|$ for any $(M_1, M_2) \in \mathcal{M}_1 \bigcup \mathcal{M}_2$. Note that when $M_1 = M_2$, $\Delta R = 0$, that is, the achieved delivery rate meets the lower bound.

Case 2 ($N = 3c + 1$): We can rewrite the last term of (12) as $r_1^* + r_2^* - \frac{3(M_1 + M_2)}{2(N-1)}$. We have $\Delta R = \left| \frac{M_1 - M_2}{2(N-1)} \right| + \frac{\max\{2M_1 + M_2, 2M_2 + M_1\}}{N(N-1)}$, for any $(M_1, M_2) \in \mathcal{M}_1 \bigcup \mathcal{M}_2$.

Case 3 ($N = 3c + 2$): The last term of (12) equals to $r_1 + r_2 - \frac{3(M_1 + M_2)}{2(N-2)}$. We have $\Delta R = \left| \frac{M_1 - M_2}{2(N-2)} \right| + \frac{2\max\{2M_1 + M_2, M_1 + 2M_2\}}{N(N-2)}$, for any $(M_1, M_2) \in \mathcal{M}_1 \bigcup \mathcal{M}_2$.

Corollary 2. *For $K = 2$, when N is divisible by 3 and the cache capacities of the users are identical, i.e., $M_1 = M_2$, if the underlying source is successively refinable, the proposed coded caching scheme meets the lower bound in Lemma 1; and hence, it is optimal, i.e., $R^*(M_1, M_2) = R_c(M_1, M_2)$.*

Remark 7. *When $D_1 = D_2$, Corollary 2 generalizes the optimal delivery rate result of [6] for $N = K = 2$ to all N values that are multiples of 3.*

D. General Case

In this section, we tackle the general scenario with N files and K users. As before, we denote by r_k the compression rate by the used scalable code that achieves an average distortion of D_k, $k = 1, ..., K$, where $D_1 \geq D_2 \geq \cdots \geq D_K$. We remind the reader that scalable code provides a layered structure of descriptions for each file, where the first layer consists of nr_1 bits, and achieves distortion D_1 when decoded successfully. The kth layer, $k = 2, ..., K$, consists of $n(r_k - r_{k-1})$ bits, and having received the first k layers, a user achieves a distortion of D_k.

The examples studied in Sections III-B and III-C illustrate the complexity of the lossy content caching problem; we had to consider five cases even with two users and two files. The problem becomes intractable quickly with the increasing number of files and users. However, note that, only users $k, k + 1, ..., K$, whose distortion requirements are lower than D_k, need to decode the kth layer for the file they request, for $k = 1, ..., K$. Therefore, once all the contents are compressed into K layers based on the distortion requirements of the users employing scalable code, we have, for each layer, a lossless caching problem. However, each user also has to decide how much of its cache capacity to allocate for each layer. Hence, the lossy caching problem under consideration can be divided into two subproblems: the lossless caching problem of each source coding layer, and the cache allocation problem among different layers.

In general, the optimal delivery rate achieved by this layered algorithm can be found by jointly optimizing the cache capacity allocated by each user to each layer, and the caching and delivery scheme to be used for each layer to minimize the corresponding delivery rate. However, we first note that the optimal delivery rate remains an open problem even in the case of equal cache capacities. Therefore, we are bound to use achievability results. Moreover, these achievability results are often characterized as piecewise linear functions, and memory-sharing among multiple schemes may be required, further complicating the identification of the minimal delivery rate. Therefore, we are not able to provide a low complexity algorithm that can optimize all the system parameters jointly. Indeed, we will propose heuristic cache allocation schemes at the users, and provide a suboptimal caching and delivery algorithm for given cache capacities.

1) Coded Lossless Caching of Each Layer: We first assume that the cache allocation at the users for each layer is already fixed, and focus on the first subproblem of centralized lossless caching with heterogeneous cache sizes. This problem has previously been studied in [30] and [32] in the decentralized setting, while, to the best of our knowledge, it has not been considered in the centralized setting. Consider, for example, the kth refinement layers of all the files. There are only $L_k \triangleq K - k + 1$ users (users $k, k + 1, ..., K$) who may request these layers. Let user j, $j \in \{k, ..., K\}$, allocate $M_{j,k}$

(normalized by n) of its cache capacity for this layer. Without loss of generality, we order users $k, ..., K$ according to the cache capacities they allocate, and re-index them, such that $M_{k,k} \leq M_{k+1,k} \leq \cdots \leq M_{K,k}$.

We would like to have symmetry among allocated cache capacities to enable multicasting to a group of users. Based on this intuition, we further divide layer k into L_k sub-layers, and let each user in $\{k, ..., K\}$ allocate $M_k^1 = M_{k,k}$ of its cache for the first sub-layer, and each user in $\{k+i-1, ..., K\}$ allocate $M_k^i = M_{k+i-1,k} - M_{k+i-2,k}$ of its cache for the ith sublayer, for $i = 2, \ldots, L_k$. Overall, we have L_k sub-layers, and users $k+i-1, k+i, ..., K$ allocate M_k^i of their caches for sub-layer i, whereas no cache is allocated by users $k, k+1, ..., k+i-2$.

We denote by r_k^i the size of the ith sub-layer of the kth refinement layer, and by $R(L_k, i, M_k^i, r_k^i, N)$ the minimum required delivery rate for this sub-layer. The rates, r_k^i, $i = 1, ..., L_k$, should be optimized jointly in order to minimize the total delivery rate for the kth layer. The optimization problem can be formulated as follows:

$$\min_{r_k^1, ..., r_k^{L_k}} \sum_{i=1}^{L_k} R(L_k, i, M_k^i, r_k^i, N) \tag{15a}$$

$$\text{s.t.} \sum_{i=1}^{L_k} r_k^i = r_k - r_{k-1}. \tag{15b}$$

We consider achievable $R(L_k, i, M_k^i, r_k^i, N)$ values based on *coded delivery* and *coded placement*, outlined in Section III-B, as well as the group-based centralized caching (GBC) scheme proposed in [10]. In *GBC* the user is grouped according to their demands in the delivery phase, i.e., users with the same demand are placed in the same group, and then contents are first exchanged within the same group and then across different groups. For the lossless caching problem, *GBC* is shown to achieve a lower delivery rate compared to memory-sharing between the *coded delivery* and *coded placement* schemes, when each user has a cache capacity of N/K (normalized by the file size). We consider two cases:

Case 1) $L_k < N$. In this case, in the worst case when users $\{k, ..., K\}$ request distinct files, *GBC* and *coded placement* provide no caching gain; thus, we employ *coded delivery*. Focus on the ith sub-layer: users $k+i-1, ..., K$ each allocate M_k^i of cache capacity, while users k to $k+i-2$ allocate no cache for this sublayer. If $r_k^i \in \mathcal{P}_{MAN} \triangleq \{0, M_k^i/N, M_k^i L_k^i/((L_k^i - 1)N), M_k^i L_k^i/((L_k^i - 2)N), ..., M_k^i L_k^i/N\}$, where $L_k^i = L_k + 1 - i$, we have

$$R_{MAN}(L_k, i, M_k^i, r_k^i, N) = (i-1) \cdot r_k^i$$
$$+ r_k^i L_k^i \cdot (1 - M_k^i/r_k^i N) \cdot \frac{1}{1 + M_k^i L_k^i/r_k^i N}. \tag{16}$$

The first term on the right hand side is due to unicasting to users k to $k + i - 2$, while the second term is the *coded delivery* rate to users $k + i - 1$ to K given in [6]. Based on the memory sharing argument, any point on the line connecting two points, $(r_1', R(L_k, i, M_k^i, r_1', N))$ and

$(r_2', R(L_k, i, M_k^i, r_2', N))$, is also achievable, i.e., if $r_k^i \in [0, M_k^i L_k^i/N]$ and $r_k^i \notin \mathcal{P}_{MAN}$, then we have

$$R(L_k, i, M_k^i, r_k^i, N)$$
$$= \min_{\substack{r_1 \in \mathcal{P}_{MAN}, \ r_1 < r_k^i \\ r_2 \in \mathcal{P}_{MAN}, \ r_1 > r_k^i}} \left\{ \frac{r_k^i - r_1'}{r_2' - r_1'} R(K_k, i, M_k^i, r_1', N) \right.$$
$$\left. + \frac{r_2' - r_k^i}{r_2' - r_1'} R(K_k, i, M_k^i, r_2', N) \right\}, \tag{17}$$

and if $r_k^i > M_k^i L_k^i/N$, we have

$$R(L_k, i, M_k^i, r_k^i, N) = (i-1) \cdot r_k^i + \frac{M_k^i L_k (L_k - 1)}{2N}$$
$$+ r_k^i M_k^i L_k L_k^i/N. \tag{18}$$

Case 2) $L_k \geq N$. In this case, *coded placement* achieves a lower delivery rate than *coded delivery* when $r_k^i \geq M_k^i L_k^i$ [9]. If $L_k^i > N \geq 3$, *GBC* outperforms *coded delivery* at point $r_k^i = M_k^i L_k^i/N$ [10]. Note that for the ith sub-layer, there are $i - 1$ users with no cache allocation. If $i - 1 \geq N$, there will be no gain with any of the schemes. When $i - 1 < N$ and $r_k^i \geq M_k^i L_k^i$, the delivery rate of *coded placement* is

$$R_{CFL}(L_k, i, M_k^i, r_k^i, N) = Nr_k^i - (N - i + 1)M_k^i. \tag{19}$$

If $i - 1 < N$, $L_k^i > N \geq 3$ and $r_k^i = M_k^i L_k^i/N$, the delivery rate provided by *GBC* is given by

$$R_{GBC}(L_k, i, M_k^i, r_k^i, N) = (i-1) \cdot r_k^i + Nr_k^i$$
$$- \frac{N(N+1)r_k^i}{2L_k^i}. \tag{20}$$

When $0 \leq r_k^i \leq M_k^i L_k^i$, the delivery rate is given by the lower convex envelope of points $(M_k^i L_k^i, R_{CFL}(L_k, i, M_k^i, M_k^i L_k^i, N))$ given by (19), $(M_k^i L_k^i/N, R_{GBC}(L_k, i, M_k^i, M_k^i L_k^i/N, N))$ given by (20), and $(r_k^i, R_{MAN}(L_k, i, M_k^i, r_k^i, N))$; and for $r_k^i \in \mathcal{P}_{MAN} \setminus \{M_k^i L_k^i/N\}$, given by (16).

Example 1. Consider $N = 2$ files and $K = 3$ users. We focus on the caching and delivery of the first layers by applying the above coded lossless caching scheme. Note that all three users require the base layer of their requested file, such that $L_1 = 3$. Assume that $r_1 = 3$ and the allocated cache capacities for this layer by the users are given by $M_{1,1} = 0.5$, $M_{2,1} = 1$, $M_{3,1} = 1.5$, respectively. The first layer of each file is thus divided into $L_1 = 3$ disjoint sub-layers. All the three users allocate a cache capacity of $M_1^1 = 0.5$ to cache the first sub-layers; user 2 and user 3 allocate a cache capacity of $M_1^2 = 0.5$ to cache the second sub-layers; and user 3 allocates $M_1^3 = 0.5$ of its cache capacity to cache the third sub-layers. The rates of these three sub-layers, r_1^1, r_1^2, r_1^3, are optimized in order to minimize the total delivery rate as given by (15). By solving (15) numerically, we obtain the optimal partition of the first layer given by $r_1^1 = 1.5$, $r_1^2 = 1.0$, and $r_1^3 = 0.5$. The delivery rates of these three sub-layers are 2, 1, 1, respectively, which results in a total delivery rate of 4.

2) Allocation of Cache Capacities: Next, we propose two algorithms for cache allocation among layers: *proportional cache allocation* (PCA) and *ordered cache allocation* (OCA), which are elaborated in Algorithms 1 and 2, respectively, where r_k is as defined earlier, and $r_0 = 0$.

Algorithm 1 Proportional Cache Allocation (PCA)

1: **Require: r** $= r_1, ..., r_K$
2: **for** $k = 1, ..., K$ **do**
3:     **for** $i = 1, ..., k$ **do**
4:         user k allocates $\frac{r_i - r_{i-1}}{r_k} M_k$ to layer i
5:     **end for**
6: **end for**

Algorithm 2 Ordered Cache Allocation (OCA)

1: **Require: r** $= r_1, ..., r_K$
2: **for** $k = 1, ..., K$ **do**
3:     user k allocates all of its cache to the first i layers, where $r_{i-1} < \frac{M_k}{N} \leq r_i$
4: **end for**

PCA allocates each user's cache among the layers it may request proportionally to the sizes of the layers, while OCA gives priority to lower layers. The server can choose the one resulting in a lower delivery rate. Numerical comparison of these two cache allocation schemes together with the delivery scheme proposed above will be presented in Section V.

Remark 8. *As stated earlier, when the QoS requirements of the users are identical, i.e., $D_1 = D_2 = \cdots = D_N$, the lossy caching problem considered here is equivalent to the loseless coded caching problem with distinct cache capacities. Therefore, the centralized coded caching scheme proposed in this section is also the first centralized caching scheme that generalizes the centralized caching problem in [6] to heterogeneous cache capacities.*

IV. DECENTRALIZED CODED CACHING WITH DISTORTION REQUIREMENTS

In this section, we consider the lossy coded caching problem in the *decentralized setting*; that is, the server is assumed to have only the set of possible distortion values that may be requested by the users, but has no prior knowledge of the number of users and their target distortion requirements. Accordingly, in decentralized caching, the *placement phase* is conducted locally and independently for each user since coordination among users is not possible. We note that the server needs to know the possible distortion values that can be requested by the users in advance in order to compress the files in the database using an appropriate successive refinement source code.

In the *placement phase* of the proposed coded caching scheme, *user k randomly caches $M_k n/N$ bits of the r_k-description of each file, for $k = 1, ..., K$.* Recall that the r_k-description corresponds to the codeword of length nr_k bits by scalable code, which achieves an average distortion of D_k when decoded. As mentioned in the previous section, scalable

code provides a layered structure of descriptions for each file; that is, r_k-distortion corresponds to the first nr_k bits of the r_{k+1}-description. Different layers of each file are cached by a different subset of users, and cached contents of each layer occupy different sizes of memory due to heterogeneous distortion requirements and heterogeneous cache sizes. To illustrate the achievable delivery rate, we first present an example with two users and two files ($K = N = 2$), and then extend our analysis to the general scenario.

Remark 9. *Here we assume that the server has the knowledge of all possible distortion values that may be requested by the users. This is needed for the server to employ scalable coding with the desired number of layers. Note that the server does not know either the number or the identity of the active users in advance; however, in practice, the number of layers, or equivalently, the number of QoS levels that can be requested will be limited either by the compression scheme employed, or due to the limited variety of devices available, and will be much smaller than the number of users in the system.*

A. Two Users and Two Files ($K = N = 2$)

Here we consider the same system model as in Section III-B with two users and two files ($N = K = 2$). User 1 has a cache of size M_1, while user 2 has a cache of size M_2.

In the *placement phase*, user 1 randomly caches $M_1 n/2$ bits from the r_1-description of each file; while user 2 randomly caches $M_2 n/2$ bits from the r_2-description of each file. We denote by $A_{\mathcal{U}}^i$ ($B_{\mathcal{U}}^i$) the bits of the ith layer of file S_1 (S_2) that are cached exclusively by the subset $\mathcal{U}$, where $\mathcal{U} \subset \{1, 2\}$ and $i = 1, 2$. For example, $A_{1,2}^1$ denotes the part of the base layer of file S_1^n cached by both users, while B_2^2 is the segment of bits from the second refinement layer of file S_2^n cached only by user 2. We list the expected size of each segment in Table II (normalized by n), where t_i denotes the probability of any bit from the r_i-description of each file cached by user i. We have $t_i = \min\{1, \frac{M_i}{2r_i}\}$, $i = 1, 2$.

In the *delivery phase*, for demand pair (S_1^n, S_2^n), the server sends $A_\emptyset^1$, $B_\emptyset^1$, $B_\emptyset^2$ and $A_2^1 \bar\oplus B_1^1$ to satisfy both requests. Note that A_2^1 and B_1^1 may be of different sizes. We employ the bitwise XOR operation $\bar\oplus$. Any other demand combination can be satisfied in a similar manner. Hence, the worst-case delivery rate is given by

$$R = 2r_1(1 - t_1)(1 - t_2) + (r_2 - r_1)(1 - t_2) + \max\{r_1 t_1(1 - t_2), r_1 t_2(1 - t_1)\}. \quad (21)$$

Similarly to centralized coded caching, multicasting is possible only for the delivery of the first layer that is requested by both users. In Fig. 3, we plot the achievable delivery rate for different cache capacity pairs (M_1, M_2), and different distortion requirement pairs (D_1, D_2), such that $r_1 = 1 - \alpha$ and $r_2 = 1 + \alpha$. The value of α varies from 0 to 1. The difference between the distortion requirements of the users becomes more significant as α increases, while the sum of the requested rates remains the same. When $\alpha = 0$, the users have the same distortion target, and the delivery rate increases with the difference between the cache capacities

TABLE II
ILLUSTRATION OF CACHE PLACEMENT

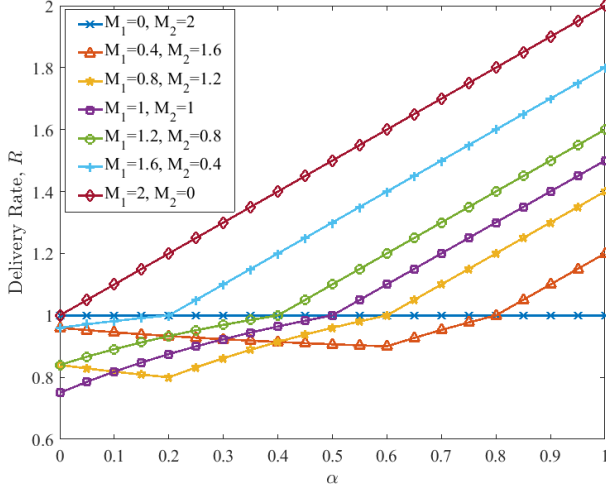|	First layer				Second layer	
Segment	$A_\emptyset^1(B_\emptyset^1)$	$A_1^1(B_1^1)$	$A_2^1(B_2^1)$	$A_{1,2}^2(B_{1,2}^2)$	$A_\emptyset^2(B_\emptyset^2)$	$A_2^2(B_2^2)$
Size	$r_1(1-t_1)(1-t_2)$	$r_1 t_1(1-t_2)$	$r_1 t_2(1-t_1)$	$r_1 t_1 t_2$	$(r_2-r_1)(1-t_2)$	$t_2(r_2-r_1)$

Fig. 3. Achievable delivery rate for $r_1 = 1 - \alpha$ and $r_2 = 1 + \alpha$, $\alpha \in [0,1]$, $N = 2$, $K = 2$.

even though the total cache capacity remains the same. We observe that the pair (M_1, M_2) that achieves the minimum delivery rate at a certain α has the closest ratio between M_1 and M_2 to the ratio between r_1 and r_2. For instance, when $\alpha = 0.2$, i.e., $r_1 = 0.8$ and $r_2 = 1.2$, $(M_1, M_2) = (0.8, 1.2)$ minimizes the delivery rate among the eight cache memory size pairs considered. Similarly, it can be observed that the delivery rate is smaller if the user with the lower distortion requirement has a larger cache capacity. We also observe that $(M_1, M_2) = (2, 0)$ always has the highest delivery rate since only one user has cache capacity. Finally, We observe from the $(M_1, M_2) = (1, 1)$ curve that, when the two users have the same cache capacity, a larger α, i.e., users with more divergent QoS requirements, results in a higher delivery rate.

B. General Case

We present a decentralized coded caching scheme for the general scenario in Algorithm 3, based on the decentralized caching scheme of [8], where the parameters r_k are as defined in Section III-D, and $W_{(d_s, D_s), \mathcal{S} \setminus \{s\}}$ denotes the part of the description of the file d_s requested by user s at distortion level D_s, that is cached exclusively by the subset of users $\mathcal{S} \setminus \{s\}$. Algorithm 3 contains two delivery procedures, DELIVERY1 and DELIVERY2, and according to the information on the active users received at the beginning of the delivery phase, the server can choose the delivery procedure with a lower delivery rate. The following lemma provides the expected size of each segment $W_{(d_s, D_s), \mathcal{S} \setminus \{s\}}$.

Algorithm 3 Decentralized Coded Caching Scheme with Lossy Distortion Requests

1: **procedure** PLACEMENT
2:     **for** $k = 1, ..., K$ **do**
3:         **for** $j = 1, ..., N$ **do**
4:             User k randomly caches $M_k n/N$ bits of the r_k-description of file j
5:         **end for**
6:     **end for**
7: **end procedure**
8: **procedure** DELIVERY1 $(d_1, ..., d_K)$
9:     **for** $k = 1, ..., K$ **do**
10:         **for** $\mathcal{S} \subset \{1, ..., K\}$: $|\mathcal{S}| = k$ **do**
11:             Send $X_\mathcal{S} = \bar{\oplus}_{s \in \mathcal{S}} W_{(d_s, D_s), \mathcal{S} \setminus \{s\}}$
12:         **end for**
13:     **end for**
14: **end procedure**
15: **procedure** DELIVERY2 $(d_1, ..., d_K)$
16:     **for** $i = 1, \ldots, N$ **do**
17:         server sends enough random linear combinations of the bits of the compressed version of file S_i^n to enable all the users demanding it to decode it at their desired distortion levels.
18:     **end for**
19: **end procedure**

Lemma 2. *According to the placement phase of Algorithm 3, as the blocklength n goes to infinity, by the law of large numbers, we have*

$$\frac{|W_{(d_s, D_s), \mathcal{S} \setminus \{s\}}|}{n} \to \sum_{i=1}^{\inf \mathcal{S}} (r_i - r_{i-1}) p_{(d_s, D_s), \mathcal{S} \setminus \{s\}}^i, \quad (22)$$

with probability 1, where $|W_{(d_s, D_s), \mathcal{S} \setminus \{s\}}|$ denotes the size of segment $W_{(d_s, D_s), \mathcal{S} \setminus \{s\}}$. We also have $r_0 = 0$, and

$$p_{(d_s, D_s), \mathcal{S} \setminus \{s\}}^i = \left(1 - \frac{M_s}{Nr_s}\right) \prod_{u \in \mathcal{S} \setminus \{s\}} \frac{M_u}{Nr_u} \cdot \prod_{u \in \{i, ..., K\} \setminus \mathcal{S}} \left(1 - \frac{M_u}{Nr_u}\right). \quad (23)$$

Proof. Based on Algorithm 3, user k caches $M_k n/N$ bits from the r_k-description of each file, which implies that each bit of the r_k-description is cached by user k with probability M_k/Nr_k, and no bit from the higher layers is cached by user k. We define $m(\mathcal{S})$ as the user with the smallest index in set $\mathcal{S}$, i.e., $m(\mathcal{S}) = \min\{s : s \in \mathcal{S}\}$. Then, each bit in $r_{m(\mathcal{S})}$-description is cached by user u, $u \in \mathcal{S}$, with probability M_u/Nr_u. Since $D_1 \geq D_2 \geq \cdots \geq D_K$, we have $D_{m(\mathcal{S})} \geq D_u$, i.e., $r_{m(\mathcal{S})} \leq r_u$, for any $u \in \mathcal{S}$. The file

segment $W_{(d_s,D_s),\mathcal{S}\setminus\{s\}}$ contains bits only from the $r_{m(\mathcal{S})}$-description of file $S^n_{d_s}$.

Note that user k caches bits only from the first k layers, i.e., the kth layer is exclusively cached by users $k, k+1, ..., K$. For the ith layer, $i = 1, ..., m(\mathcal{S})$, which is cached by users $i, ..., K$, every bit in the ith layer of file d_s is exclusively cached by users in subset $\mathcal{S}\setminus\{s\}$ with probability $p^i_{(d_s,D_s),\mathcal{S}\setminus\{s\}}$. Since the total rate of the ith layer is $r_i - r_{i-1}$, the expected number of bits from the ith layer in $W_{(d_s,D_s),\mathcal{S}\setminus\{s\}}$ is $n(r_i - r_{i-1})p^i_{(d_s,D_s),\mathcal{S}\setminus\{s\}}$. As $W_{(d_s,D_s),\mathcal{S}\setminus\{s\}}$ has bits from the first $m(\mathcal{S})$ layers, we sum up the expected rate of all these layers, which yields (22). $\square$

Since $\frac{M_k}{Nr_k}$ are not identical for $k = 1, .., K$, in the delivery phase, for the multicast subset $\mathcal{S}$, the sizes of the corresponding segments, $W_{(d_s,D_s),\mathcal{S}\setminus\{s\}}$, $s \in \mathcal{S}$, will be different. Therefore, we apply the $\oplus$ operation. Hence, the size of the multicasted segment for subset $\mathcal{S}$ is given by

$$|X_{\mathcal{S}}| = \max_{s \in \mathcal{S}} |W_{(d_s,D_s),\mathcal{S}\setminus\{s\}}| \tag{24a}$$

$$= |W_{(d_{r'(\mathcal{S})},D_{r'(\mathcal{S})}),\mathcal{S}\setminus\{r'(\mathcal{S})\}}| \tag{24b}$$

$$= n\sum_{i=1}^{m(\mathcal{S})}(r_i - r_{i-1})p^i_{(d_{r'(\mathcal{S})},D_{r'(\mathcal{S})}),\mathcal{S}\setminus\{r'(\mathcal{S})\}}. \tag{24c}$$

where $r'(\mathcal{S}) = \underset{s \in \mathcal{S}}{\arg\min} \frac{M_s}{Nr_s}$, which is the index of the user in subset $\mathcal{S}$ with the smallest $\frac{M_s}{Nr_s}$. Using (24), the delivery rate

---

Algorithm 4 Layered Content Delivery 1 (LCD1)

1: **procedure** DELIVER $(d_1, ..., d_K)$
2:    **for** $i = 1, ..., K$ **do**
3:       **procedure** DELIVERY iTH LAYER OF FILES $(d_i, ..., d_K)$ TO USERS $\{i, ..., K\}$
4:          **for** $k = 1, ..., K + 1 - i$ **do**
5:             **for** $\mathcal{S} \subset \{i, ..., K\}: |\mathcal{S}| = k$ **do**
6:                Send $X^i_{\mathcal{S}} = \oplus_{s \in \mathcal{S}} W^i_{d_s,\mathcal{S}\setminus\{s\}}$
7:             **end for**
8:          **end for**
9:       **end procedure**
10:    **end for**
11: **end procedure**

---

of Algorithm 3 can be derived.

Theorem 4. *For the decentralized coded caching system described above, Algorithm 3 achieves a delivery rate given by*

$$R(M_1,...,M_K) = \min\left\{\sum_{i=1}^{K}(r_i - r_{i-1})\sum_{l=1}^{K-i+1}\prod_{k=1}^{l}(1 - t^i_k),\right.$$
$$\left.\sum_{i=1}^{\min\{N,K\}} r_{K-i+1} - \min_{\substack{\mathcal{S}\subset\{1,...,K\}\\|\mathcal{S}|=\min\{N,K\}}}\sum_{k\in\mathcal{S}}\frac{M_k}{N}\right\}, \tag{25}$$

where $\{t^i_1, t^i_2, ..., t^i_{K-i+1}\}$ is an ordered permutation of $\left\{\frac{M_i}{Nr_i}, \frac{M_{i+1}}{Nr_{i+1}}, ..., \frac{M_K}{Nr_K}\right\}$ such that $t^i_1 \le t^i_2 \le \cdots \le t^i_{K-i+1}$, $i \in \{1, ..., K\}$.

Proof. We first prove the first term in (25), which is provided by DELIVERY 1. We sum up the rates corresponding to all possible multicasting subsets, $\mathcal{S} \subset \{1, ..., K\}$. From (24), we have

$$R(M_1, ..., M_K) = \sum_{k=1}^{K}\sum_{\substack{\mathcal{S}\subset\{1,...,K\}\\|\mathcal{S}|=k}}\sum_{i=1}^{m(\mathcal{S})}(r_i - r_{i-1})\cdot$$
$$p^i_{(d_{r'(\mathcal{S})},D_{r'(\mathcal{S})}),\mathcal{S}\setminus\{r'(\mathcal{S})\}} \tag{26a}$$

$$= \sum_{i=1}^{K}\sum_{k=1}^{K-i+1}\sum_{\substack{\mathcal{S}\subset\{i,...,K\}\\|\mathcal{S}|=k}}(r_i - r_{i-1})\cdot$$
$$p^i_{(d_{r'(\mathcal{S})},D_{r'(\mathcal{S})}),\mathcal{S}\setminus\{r'(\mathcal{S})\}}, \tag{26b}$$

where k denotes the cardinality of subsets, for $k = 1, ..., K$. Note that $n \cdot (r_i - r_{i-1})p^i_{(d_{r'(\mathcal{S})},D_{r'(\mathcal{S})}),\mathcal{S}\setminus\{r'(\mathcal{S})\}}$ denotes the number of bits from the ith layer in $|X_{\mathcal{S}}|$. The above equation implies that it is equivalent to delivering each layer separately. We present DELIVERY1 as the layered content delivery approach in Algorithm 4, where $W^i_{d_s,\mathcal{S}\setminus\{s\}}$ denotes the set of bits from the ith layer of file d_s that are cached exclusively by users in the subset $\mathcal{S}\setminus\{s\}$. Hence,

$$|X^i_{\mathcal{S}}| = n \cdot (r_i - r_{i-1})p^i_{(d_{r'(\mathcal{S})},D_{r'(\mathcal{S})}),\mathcal{S}\setminus\{r'(\mathcal{S})\}}. \tag{27}$$

We focus on the delivery of the ith layer to users $\{i, ..., K\}$ in the following. For this layer, we order and re-index users $\{i, ..., K\}$ with $\{1, .., K - i + 1\}$ such that $\frac{M_k}{Nr_k} = t^i_k$, for $k = 1, ..., K - i + 1$. Based on this re-indexing, we have

$$\sum_{k=1}^{K-i+1}\sum_{\substack{\mathcal{S}\subset\{i,...,K\}\\|\mathcal{S}|=k}}p^i_{(d_{r'(\mathcal{S})},D_{r'(\mathcal{S})}),\mathcal{S}\setminus\{r'(\mathcal{S})\}}$$
$$= \sum_{l=1}^{K-i+1}\prod_{k=1}^{l}(1 - t^i_k), \tag{28}$$

which yields the first term in (25). A detailed proof of (28) can be found in Appendix D.

Now, we continue with analysis of the DELIVERY2 procedure in Algorithm 3. Note that the message sent at step 17 of Algorithm 3 is targeted at a group of users requesting file S^n_i, for $i = 1, ..., N$. Let $\mathcal{S}_i$ denote the set of users requesting file S^n_i, $i = 1, ..., N$. Since each user $k \in \mathcal{S}_i$ has nM_k/N bits of file S^n_i stored in its caches, with the fact stated in [8, Appendix A], at most

$$n\left\{\max_{k\in\mathcal{S}_i}r_k - \min_{k\in\mathcal{S}_i}\frac{M_k}{N}\right\} \tag{29}$$

random linear combinations need to be sent for all those users in $\mathcal{S}_i$ to decode file S^n_i at their desired distortion values. If $N \ge K$, the worst case demand combination corresponds to each user requesting a distinct file. There are at most K files requested, which yields a delivery rate of

$$\sum_{i=1}^{K}\left\{r_i - \frac{M_i}{N}\right\} \tag{30}$$

for the DELIVERY2 procedure.

If $N < K$, the worst case demand combination occurs when the N users with the lowest distortion requirements, i.e., the N largest r_k values, $k = 1, ..., K$, and the N users with the smallest cache capacities are in different groups $\mathcal{S}_i$, $i = 1, ..., N$, that is, they request N distinct files. Since $D_1 \geq D_2 \geq \cdots \geq D_k$, i.e., $r_1 \leq r_2 \leq \cdots \leq r_K$, for this case, the DELIVERY2 procedure achieves a delivery rate of

$$\sum_{i=1}^{N} r_{K-i+1} - \min_{\substack{\mathcal{S} \subset \{1,..,K\} \\ |\mathcal{S}|=N}} \sum_{k \in \mathcal{S}} \frac{M_k}{N}. \tag{31}$$

Combing (30) and (31), we have the second term in (25). This completes the proof. $\qquad\square$

We remark that Eqn. (28) denotes the probability of any bit of the ith layer to be sent in the delivery phase. Note that (28) has a similar form to the expression of the achievable delivery rate in the lossless coded caching problem with heterogeneous cache sizes presented in [30, Theorem 3]. This implies that, with the *placement phase* carried out as in Algorithm 3, the delivery rate is equivalent to the lossless coded caching scheme proposed in [30] for each layer.

For the decentralized lossless coded caching problem with distinct cache capacities, [32] further exploits the multicasting gain among users requesting the same file, which achieves a lower delivery rate when the number of users is larger than the number of files, compared to [30]. Here, in Algorithm 5, we employ a caching strategy similar to [32, Algorithm 1]. Based on [32, Theorem 1], we have Theorem 5 specifying the delivery rate achieved by Algorithm 5.

Theorem 5. *For the decentralized coded caching system described above, Algorithm 5 achieves a delivery rate given by*

$$R(M_1, ..., M_K)$$
$$= \sum_{i=1}^{K} (r_i - r_{i-1}) \left(\sum_{l=1}^{K-i+1} \prod_{k=1}^{l} (1 - t_k^i) - \Delta R_i^1 - \Delta R_i^2 \right), \tag{32}$$

where t_k^i is defined as in Theorem 1, for $k \in \{1, ..., K-i+1\}$ and $i \in \{1, ..., K\}$, and

$$\Delta R_i^1 = \begin{cases} (L_i - N) \prod_{l=1}^{L_i} \left(1 - t_l^i \right), & \text{if} \quad L_i > N \\ 0, & \text{if} \quad L_i \leq N \end{cases} \tag{33a}$$

$$\Delta R_i^2 = \begin{cases} \left[\sum_{k=1}^{L_i - N} \left(\frac{(k-1)t_{k+N}^i}{1 - t_{k+N}^i} \right) \right] \prod_{l=1}^{L_i} \left(1 - t_l^i \right), & \text{if} \quad L_i > N \\ 0, & \text{if} \quad L_i \leq N \end{cases} \tag{33b}$$

and $L_k = K - i + 1$.

We can see that if $K \leq N$, $L_i \leq N$ holds for $i = 1, ..., K$, thus (32) is equivalent to (25), that is, LCD2 has the same performance with LCD1. For the case $K > N$, we have $L_i > N$ for $i = 1, ..., K - N$, and LCD2 results in a reduction in the

---

Algorithm 5 Layered Content Delivery 2 (LCD2)(Based on Algorithm 1 in [32])

1: **procedure** DELIVER $(d_1, ..., d_K)$
2:     **for** $i = 1, ..., K$ **do**
3:         **procedure** DELIVERY iTH LAYER OF FILES $(d_i, ..., d_K)$ TO USERS $\{i, ..., K\}$
4:           Define K_j as the number of users in $\{i, ..., K\}$ that requests file j, for $j = 1, ..., N$;
5:           Reorder and reindex users $\{i, ..., K\}$ such that $d_k = j$, for $j = 1, ..., N$, and $k = S_{j-1} + 1, ..., S_j$ where $S_0 = i - 1$ and $S_j = \sum_{l=1}^{j} K_l$;
6:         **Part 1**: Delivering bits that are not in the cache of any user
7:         **for** $j = 1, 2, \ldots, N$ **do**
8:           $X_1^i = \left(W_{d_{S_{j-1}+1}, \{\emptyset\}}^i \right)$
9:         **end for**
10:         **Part 2**: Delivering bits that are in the cache of only one user
11:

$$X_{21}^i = \left(\bigcup_{j=1}^{N} \bigcup_{k=S_{j-1}+1}^{S_j - 1} \left(W_{j, \{k\}}^i \bar{\oplus} W_{j, \{k+1\}}^i \right) \right)$$

12:

$$X_{22}^i = \bigcup_{j=1}^{N-1} \bigcup_{h=j+1}^{N} \left(\bigcup_{k=S_{h-1}+1}^{S_h - 1} \left(W_{j, \{k\}}^i \bar{\oplus} W_{j, \{k+1\}}^i \right), \right.$$
$$\left. \bigcup_{k=S_{j-1}+1}^{S_j - 1} \left(W_{h, \{k\}}^i \bar{\oplus} W_{h, \{k+1\}}^i \right) \right)$$

13:

$$X_{23}^i = \left(\bigcup_{j=1}^{N-1} \bigcup_{h=j+1}^{N} W_{j, \{S_{h-1}+1\}}^i \bar{\oplus} W_{h, \{S_{j-1}+1\}}^i \right)$$

14:         **Part 3**: Delivering bits that are in the cache of more than one user
15:         **for** $j = i, \ldots, K - 2$ **do**
16:           **for** $h = 2, 3, \ldots, K - i$ **do**
17:             **for** $V \subset [j + 1 : K] : |V| = h$ **do**
18:

$$X_3^i = \left(\left(\bigoplus_{v \in V} W_{d_v, \{V, j\} \setminus \{v\}}^i \right) \bar{\oplus} W_{d_j, V}^i \right)$$

19:             **end for**
20:           **end for**
21:         **end for**
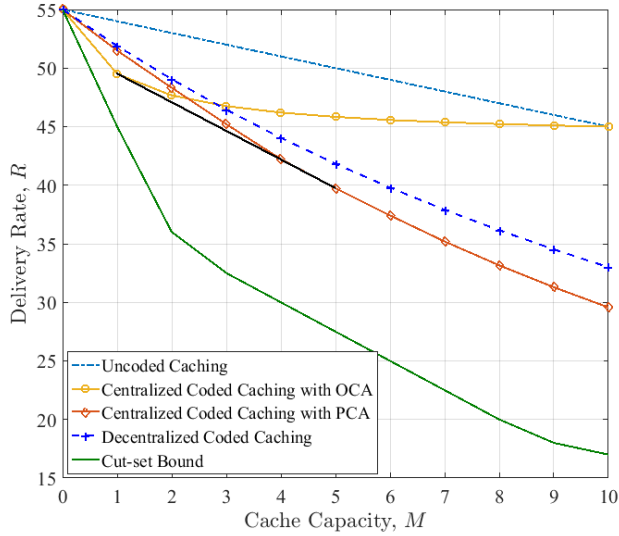22:   **end procedure**
23:     **end for**
24: **end procedure**

Fig. 4. Comparison of the achievable delivery rates with identical cache capacities, i.e., $M_1 = M_2 = \ldots = M_K$, for $N = 10$, $K = 10$.

Fig. 5. Comparison of the achievable delivery rates with heterogeneous cache capacities, i.e., $M_k = 0.2kM$, $k = 1, ..., K$, for $N = 10$, $K = 10$.

delivery rate for these layers, quantified by ΔR_i^1 and ΔR_i^2, in (33a) and (33b), respectively. Numerical comparison of the performance of these two content delivery algorithms, LCD1 and LCD2, is presented in Section V.

V. NUMERICAL RESULTS

In this section, we numerically compare the delivery rates of the proposed centralized and decentralized coded lossy caching schemes. We particularly consider two different cases depending on the relative numbers of users and files in the system, i.e., $N \geq K$ and $N < K$, as the proposed caching schemes exhibit different behaviors in these cases.

In the first scenario, we consider a caching system with a server containing $N = 10$ files serving $K = 10$ users. The target distortion levels of the users are given by $(D_1, D_2, ..., D_{10})$, which we assume to be achievable by a scalable code with rates $(r_1, r_2, ..., r_{10}) = (1, 2, ..., 10)$. To compare the achievable rates with the lower bound, we assume the underlying source distribution is successively refinable, that is, $r_k = r_k^*$, $k = 1, ..., K$. We consider two cases for the cache sizes: the first one with identical cache capacities, i.e., $M_1 = M_2 = \cdots = M_{10} = M$, and the second one with heterogeneous cache capacities, where $M_k = 0.2kM$, for $k = 1, ..., 10$. The results for these two scenarios for both the centralized and decentralized caching are plotted in Fig. 4 and Fig. 5, respectively.

In Fig. 4, we observe that the centralized coded caching scheme with OCA achieves the best delivery rate when the cache capacities are very small but its performance approaches that of the uncoded caching scheme as M increases. This implies that when the cache capacity is small, it should be allocated mainly to the first layer, which is requested by all the users. Since the cache capacities are identical, coded caching of the same layer across users creates more multicasting opportunities, which better exploits the limited cache capacity.
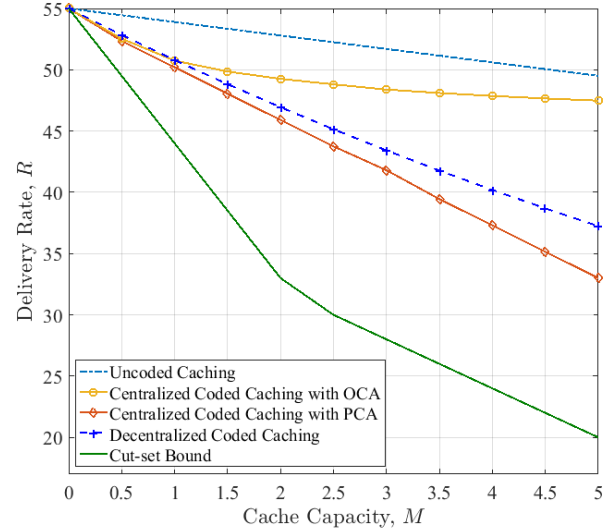
On the other hand, for medium to large cache capacities, PCA significantly outperforms OCA. This is because, when there is sufficient cache capacity, caching higher layers creates new multicasting opportunities, which further reduces the delivery rate. The black line in the figure is achievable by memory sharing between the two caching schemes, OCA and PCA, further reducing the delivery rate for moderate cache capacities. In Fig. 5, a network with heterogeneous cache capacities is considered, and it is observed that PCA outperforms OCA for all values of M, since PCA is capable of exploiting the additional cache capacity of users to meet the requirements of reduced distortion target, retaining the symmetry among the amount of cache allocated to each layer across different users.

We remark that, since $N = K$, decentralized caching with LCD1 and LCD2 have the same performance as characterised in Theorem 4. For very small cache capacities, the decentralized scheme achieves almost the same performance as the centralized scheme with PCA. This is because, when cache capacities are very small, users will cache distinct bits from their required layers with high probability despite random placement, which is similar to the cache placement phase used by the centralized scheme with PCA. The performance improvement of centralized caching becomes more pronounced as M increases since the collaboration between users during the placement phase will fully exploit the cache capacities to create the maximum number of multicasting opportunities. We observe that, despite lack of cache coordination across users, decentralized caching still achieves a performance not far from the best centralized caching scheme.

In both Fig. 4 and Fig. 5, it is observed that a significant improvement can be achieved by coded caching. We also remark that the total cache capacity across the whole network is $10M$ and $11M$ for the settings considered in Fig. 4 and Fig. 5, respectively. However, we can notice that the delivery rate achieved in Fig. 4 is significantly higher in both the centralized
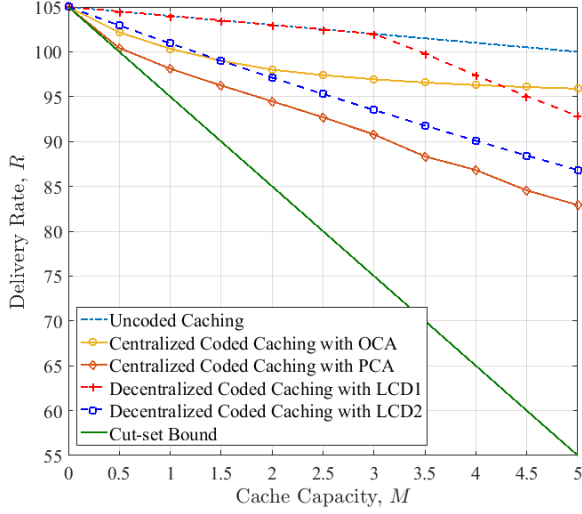
Fig. 6. Comparison of the achievable delivery rates with identical cache capacities, i.e., $M_1 = M_2 = \ldots = M_K$, for $N = 10$, $K = 15$.
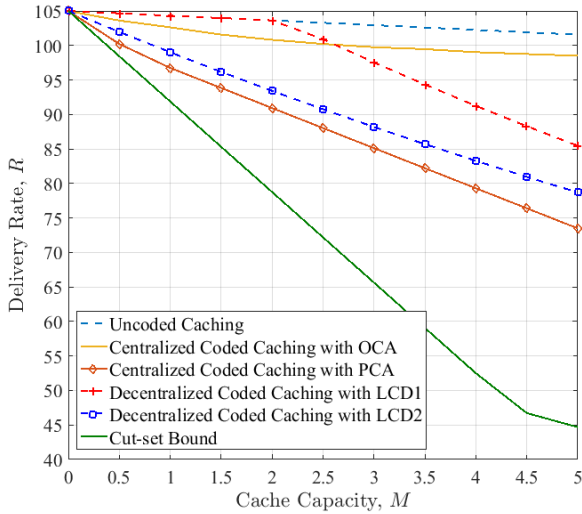

Fig. 8. Comparison of achievable delivery rates versus α, $\alpha \in [0, 1]$, $M_1 = M_2 = \cdots = M_{10} = 5$, for $N = 10$, $K = 10$.

Fig. 7. Comparison of the achievable delivery rates with heterogeneous cache capacities, i.e., $M_k = 0.125kM$, $k = 1, ..., K$, for $N = 10$, $K = 15$.

and decentralized scenarios. This is due to the distribution of the cache capacity across the users. In the latter scenario, the users with lower distortion requirements have larger cache capacities, allowing them to achieve their desired QoS targets without increasing the delivery rate.

In the second scenario, we consider a server with $N = 10$ files serving $K = 15$ users. The target distortion levels are such that $(r_1, r_2, ..., r_{15}) = (1, 2, ..., 15)$. As in the first scenario, we consider two cases: identical cache sizes, i.e., $M_1 = M_2 = \cdots = M_{15} = M$, plotted in Fig. 6; and heterogeneous cache sizes, i.e., $M_k = 0.125kM$, for $k = 1, ..., 15$, plotted in Fig. 7, such that the total cache capacity across the network for both scenarios is $15M$.

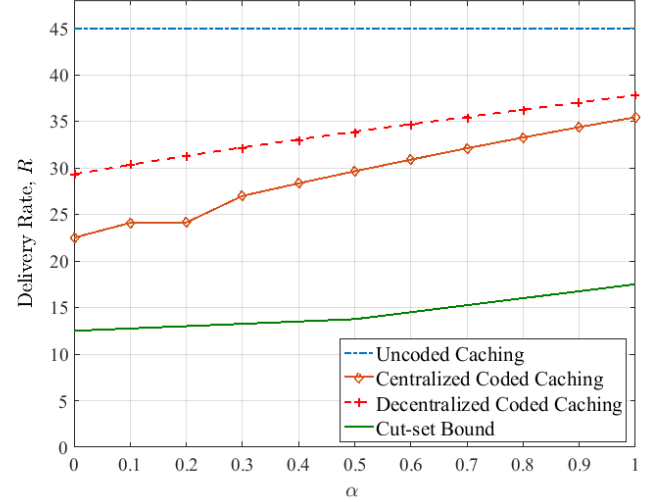We observe that, for centralized coded caching, PCA outperforms OCA in both Fig. 6 and Fig. 7. This implies that it is

always better for each user to distribute its cache capacity to all the layers it may require, rather than to allocate it only to the first layer. Since $N < K$, the number of users requesting the first layer is larger than the number of files. There are at least two users requesting the same file, which creates multicasting opportunities even with uncoded caching, reducing the gain of coded caching over uncoded caching. Allocating the cache capacities to other layers will better exploit coded caching. In decentralized caching, we see that, LCD1 has the same performance as uncoded caching for small cache capacities. However, Fig. 6 shows a larger range of cache capacities where LCD1 and uncoded caching have the same performance, since when the cache capacities are identical, for each layer, the expected number of bits cached by each user is distinct, which reduces the gain from coded delivery. It is observed in both figures, LCD2 greatly outperforms LCD1, but the gap between two schemes reduces with the cache capacity. This is because, the improvement of LCD2 over LCD1 derives from more effectively delivering the bits that are cached by at most one user. When the cache capacity increases, the number of such bits reduces, hence, the performance of LCD2 approaches that of LCD1. Although the total cache capacity across the network is the same for the scenarios considered in Fig. 6 and Fig. 7, it is observed that in both the centralized and decentralized settings, the rates achieved in Fig. 7 are significantly lower, similar to the $N = K$ scenario, since the larger cache capacities can be exploited to improve the QoS of users by the proposed layered caching approach.

We also observe that in all the figures, the gain of coded caching, both in centralized and decentralized settings, becomes more significant as the cache capacity, M, increases. We also note that there is a relatively large gap between the best achievable delivery rates and the cut-set lower bound, but part of this gap is potentially due to the looseness of the cut-set bound, as also suggested in [6].
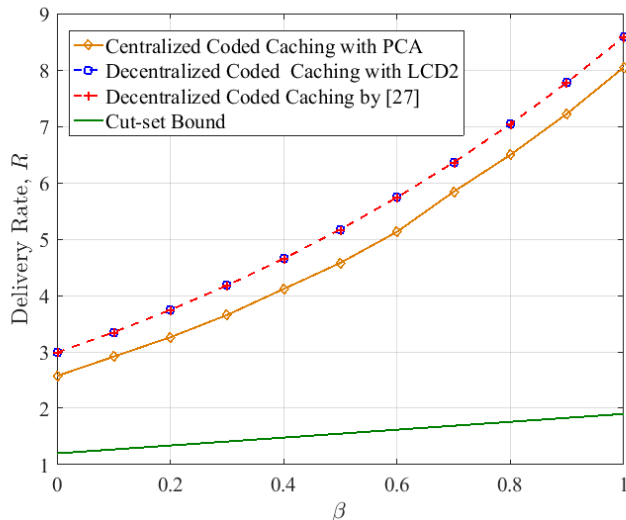
Next, we consider a server with $N = 10$ files serving

Fig. 9. The achievable delivery rates versus β, $\beta \in [0,1]$, $r_1 = r_2 = \cdots = r_{15}$, for $N = 10$, $K = 15$.

$K = 10$ users, with identical cache capacities, i.e., $M_1 = M_2 = \cdots = M_{10} = 5$. The target distortion levels of the users, $(D_1, D_2, \ldots, D_{10})$, are given by $r_k = 5 + (k-5.5)\alpha$, for $k = 1, ..., 10$ and $\alpha \in [0,1]$. In this scenario, the average value of r_k is 5, i.e., $\sum_{k=1}^{10} r_k/10 = 5$, independent of the value of α. As α increases, the distortion requirements of users become more diverse. The delivery rates of the proposed caching schemes, in centralized and decentralized setting, are shown in Fig. 8, compared with *uncoded caching* and cut-set bound. We emphasize that the delivery rate of the centralized caching scheme in Fig. 8 is the lower one of the delivery rates achieved by PCA and OCA. We observe that, while the delivery rate of *uncoded caching* remain the same, the delivery rates of both centralized and decentralized coded caching schemes increase with α, which indicates the loss of coded caching gain due to the diversity of distortion requirements.

Finally, we consider a server with $N = 10$ files, and $K = 15$ users with identical QoS requirements, i.e., $D_1 = D_2 = \cdots = D_{15}$, such that $r_1 = r_2 = \cdots = r_{15} = 2$. The cache size of each user is given by $M_k = 8 + (k - 8)\beta$, for $k = 1, ..., 15$ and $\beta \in [0,1]$. The larger β, the more skewed the cache size distribution is. This setting is equivalent to a lossless coded caching problem with distinct cache capacities. The state-of-art decentralized coded caching scheme for this setting is presented in [32]. In Fig. 9, we compare the best achievable rates by the proposed centralized and decentralized coded caching schemes with the one proposed in [32]. Our decentralized coded caching scheme is shown to achieve the same performance as the scheme proposed in [32]. This is because we adopt the scheme proposed in [32] for each layer. Since there is only one layer in this scenario, it is equivalent to applying the scheme proposed in [32] to this layer. For all the three schemes, the gain from coded caching becomes more pronounced as β becomes small, i.e., as the cache capacities of users become more similar.

VI. CONCLUSIONS

We have considered coded caching and delivery of contents in wireless networks, taking into account the heterogeneous distortion requirements of users, in both centralized and decentralized settings. The caching and delivery schemes considered here exploit the specific properties of lossy reconstruction of contents by users. In particular, we have exploited scalable coding, which allowed us to cache contents incrementally across users depending on their cache capacities and distortion requirements. In the *centralized* setting, for a simple setting of two users and two files, we have derived the optimal coded caching scheme that achieves the information-theoretic lower bound when the underlying source distribution is successively refinable. We have further explored the case with two users and an arbitrary number of files. The proposed scheme is proven to be optimal for successively refinable sources when the cache capacities of the two users are the same and the number of files in the database is divisible by 3. Then, assuming that the server employs scalable coding to compress all the files in the database at the required distortion levels, we tackled the general case with K users and N files in two steps: delivery rate minimization, which finds the minimum delivery rate for each layer separately, and cache capacity allocation among layers. We proposed two algorithms for the latter, namely, PCA and OCA.

In the decentralized setting, since the number and identity of the active users are not known during the *placement phase*, we have employed random cache placement. We have proved that a layered delivery scheme, which delivers each layer separately, is without loss of optimality, compared to joint delivery across layers. We have applied the existing coded caching scheme for the lossless caching problem with heterogeneous cache capacities to the delivery of each layer, together with the analysis on the achievable delivery rate. We have validated the improvement of the proposed coded caching schemes compared to uncoded caching through numerical results. However, there is still a remarkable gap between the best achieved delivery rate and the cut-set lower bound in all the scenarios considered in Section V. We have derived a tighter bound for a special scenario with two users, which was then used to prove the optimality of the proposed coded caching scheme for this scenario. Extending the tighter lower bounds for the lossless caching problem proposed in [33], [34] to the lossy caching problem studied here is currently under consideration to better understand the optimal performance.

APPENDIX A
PROOF OF THEOREM 1

Let $s \in \{1, ..., \min\{N, K\}\}$, and consider a set of users $\mathcal{U}$ with $|\mathcal{U}| = s$. There exists a demand combination and a corresponding message over the shared link, say X_1^n, such that X_1^n and $\{Z_k^n \mid k \in \mathcal{U}\}$ allow the reconstruction of files $S_1^n, ..., S_s^n$, each at the required distortion level from $\{D_k \mid k \in \mathcal{U}\}$. Similarly, there exists an input to the shared link, say X_2^n, such that X_2^n and $\{Z_k^n \mid k \in \mathcal{U}\}$ allow the reconstruction of files $S_{s+1}^n, ..., S_{2s}^n$, each at the required distortion level from $\{D_k \mid k \in \mathcal{U}\}$, and so on so forth. We can continue in the

same manner considering messages $X_3^n, ..., X_{\lfloor N/s \rfloor}^n$ for further demand combinations. Hence, with $X_1^n, ..., X_{\lfloor N/s \rfloor}^n$ and $\{Z_k^n \mid k \in \mathcal{U}\}$, each user $k \in \mathcal{U}$ should be able to reconstruct a distinct set of $\lfloor N/s \rfloor$ files at the corresponding distortion level D_k. By considering a cut separating $X_1^n, ..., X_{\lfloor N/s \rfloor}^n$ and $\{Z_k^n \mid k \in \mathcal{U}\}$ from the corresponding users, we have

$$\sum_{i=1}^{\lfloor N/s \rfloor} |X_i^n| + \sum_{k \in \mathcal{U}} |Z_k| \geq \lfloor N/s \rfloor \sum_{k \in \mathcal{U}} R(D_k), \qquad (34)$$

according to the cut-set bound [28, Theorem 14.10.1]. From the definition of $R^*(M_1, ..., M_K)$, we have $R^*(M_1, ..., M_K) \geq |X_i^n|$, for $i = 1, ..., \lfloor N/s \rfloor$. We also have the cache capacity constraints $M_k \geq |Z_k|$, for $k \in \mathcal{U}$. Plugging these into (34), we obtain

$$\lfloor N/s \rfloor R^*(M_1, ..., M_K) + \sum_{k \in \mathcal{U}} M_k \geq \lfloor N/s \rfloor \sum_{k \in \mathcal{U}} R(D_k). \qquad (35)$$

Since this inequality must hold for all possible choices of s, and corresponding subset of users $\mathcal{U}$ with $|\mathcal{U}| = s$, we can obtain the folowing cut-set bound by substituting $r_k^* = R(D_k)$:

$$R^*(M_1, ..., M_K) \geq R_C(M_1, ..., M_K)$$
$$\triangleq \max_{s \in \{1, ..., \min\{N, K\}\}} \max_{\mathcal{U} \subset \{1, ..., K\}, |\mathcal{U}| = s} \left(\sum_{k \in \mathcal{U}} r_k^* - \frac{\sum_{k \in \mathcal{U}} M_k}{\lfloor N/s \rfloor} \right). \qquad (36)$$

APPENDIX B
PROOF OF THEOREM 2

We consider two groups of demands, i.e., $\{(2i+1, 2i+2) \mid i = 0, 1, ..., \lfloor N/2 \rfloor - 1\}$ and $\{(2i+2, 2i+1) \mid i = 0, 1, ..., \lfloor N/2 \rfloor - 1\}$. We define the vector of channel inputs corresponding to these demand combinations as follows $\bar{X}_{12}^n = \bigcup_{i=0}^{\lfloor N/2 \rfloor - 1} X_{(2i+1, 2i+2)}^n$, and $\bar{X}_{21}^n = \bigcup_{i=0}^{\lfloor N/2 \rfloor - 1} X_{(2i+2, 2i+1)}^n$. Then, for any (n, M_1, M_2, R) caching code that achieves the distortion tuple (D_1, D_2), we have

$$2\lfloor N/2 \rfloor nR + nM_1 + nM_2$$
$$\geq H(\bar{X}_{12}^n, Z_1^n) + H(\bar{X}_{21}^n, Z_2^n) \qquad (37a)$$
$$= H(\bar{X}_{12}^n, Z_1^n | \bar{S}_1^n) + H(\bar{X}_{21}^n, Z_2^n | \bar{S}_1^n)$$
$$\quad + I(\bar{S}_1^n; \bar{X}_{12}^n, Z_1^n) + I(\bar{S}_1^n; \bar{X}_{21}^n, Z_2^n) \qquad (37b)$$
$$\geq H(\bar{X}_{12}^n, Z_1^n, \bar{X}_{21}^n, Z_2^n | \bar{S}_1^n)$$
$$\quad + I(\bar{S}_1^n; \bar{X}_{12}^n, Z_1^n) + I(\bar{S}_1^n; \bar{X}_{21}^n, Z_2^n) \qquad (37c)$$
$$\geq I(\bar{S}_2^n; \bar{X}_{12}^n, Z_1^n, \bar{X}_{21}^n, Z_2^n | \bar{S}_1^n)$$
$$\quad + I(\bar{S}_1^n; \bar{X}_{12}^n, Z_1^n) + I(\bar{S}_1^n; \bar{X}_{21}^n, Z_2^n) \qquad (37d)$$

where $\bar{S}_1^n = \bigcup_{i=0}^{\lfloor N/2 \rfloor - 1} S_{2i+1}^n$ and $\bar{S}_2^n = \bigcup_{i=0}^{\lfloor N/2 \rfloor - 1} S_{2i+2}^n$. We denote by $\hat{S}_{i,k}^n$ the reconstruction of S_i^n at user k, for

$i = 1, ..., N$ and $k = 1, 2$. Let $\hat{\bar{S}}_{11}^n = \bigcup_{i=0}^{\lfloor N/2 \rfloor - 1} \hat{S}_{2i+1,1}^n$, $\hat{\bar{S}}_{12}^n = \bigcup_{i=0}^{\lfloor N/2 \rfloor - 1} \hat{S}_{2i+1,2}^n$ and $\hat{\bar{S}}_{22}^n = \bigcup_{i=0}^{\lfloor N/2 \rfloor - 1} \hat{S}_{2i+2,2}^n$. We have

$$I(\bar{S}_1^n; \bar{X}_{12}^n, Z_1^n) \geq I(\bar{S}_1^n; \hat{\bar{S}}_{11}^n) \qquad (38a)$$
$$\geq \sum_{i=0}^{\lfloor N/2 \rfloor - 1} I(S_{2i+1}^n; \hat{\bar{S}}_{11}^n) \qquad (38b)$$
$$\geq \sum_{i=0}^{\lfloor N/2 \rfloor - 1} I(S_{2i+1}^n; \hat{S}_{2i+1,1}^n) \qquad (38c)$$
$$\geq \lfloor N/2 \rfloor nR(D_1), \qquad (38d)$$

where the inequality (38a) follows from the data processing inequality since $\bar{S}_1^n - (\bar{X}_{12}^n, Z_1^n) - \hat{\bar{S}}_{11}^n$ forms a Markov chain; (38b) holds due to the independence of files. Since the reconstruction of S_{2i+1}^n at user 1, i.e., $\hat{S}_{2i+1,1}^n$, is required to be within distortion D_1, the inequality (38d) follows from the definition of the rate distortion function [29].

Similarly, we have

$$I(\bar{S}_1^n; \bar{X}_{21}^n, Z_2^n) \geq I(\bar{S}_1^n; \hat{\bar{S}}_{12}^n) \qquad (39a)$$
$$\geq \sum_{i=0}^{\lfloor N/2 \rfloor - 1} I(S_{2i+1}^n; \hat{\bar{S}}_{12}^n) \qquad (39b)$$
$$\geq \sum_{i=0}^{\lfloor N/2 \rfloor - 1} I(S_{2i+1}^n; \hat{S}_{2i+1,2}^n) \qquad (39c)$$
$$\geq \lfloor N/2 \rfloor nR(D_2). \qquad (39d)$$

Finally, also using the data processing inequality due to the Markov chain, $(\bar{S}_1^n, \bar{S}_2^n) - (\bar{X}_{12}^n, Z_2^n) - \hat{\bar{S}}_{22}^n$, we can write

$$I(\bar{S}_2^n; \bar{X}_{12}^n, Z_1^n, \bar{X}_{21}^n, Z_2^n | \bar{S}_1^n) \geq I(\bar{S}_2^n; \bar{X}_{12}^n, Z_2^n | \bar{S}_1^n) \qquad (40a)$$
$$\geq I(\bar{S}_2^n; \hat{\bar{S}}_{22}^n | \bar{S}_1^n) \qquad (40b)$$
$$= I(\bar{S}_2^n; \hat{\bar{S}}_{22}^n, \bar{S}_1^n) \qquad (40c)$$
$$= I(\bar{S}_2^n; \hat{\bar{S}}_{22}^n) \geq \lfloor N/2 \rfloor nR(D_2), \qquad (40d)$$

where (40c) holds since $\bar{S}_2^n$ is independent $\bar{S}_1^n$; and (40d) follows from the definition of the rate-distortion function.

Substituting inequalities (38), (39), (40d) into (37d) and replacing $R(D_k)$ with r_k^*, $k = 1, 2$, we obtain

$$R \geq r_1^*/2 + r_2^* - \frac{(M_1 + M_2)}{2\lfloor N/2 \rfloor}. \qquad (41)$$

APPENDIX C
LOWER BOUND OF LEMMA 2

We consider two groups of $\lfloor N/3 \rfloor$ demands, i.e., $\{(3i+1, 3i+2) \mid i = 0, 1, ..., \lfloor N/3 \rfloor - 1\}$ and $\{(3i+2, 3i+3) \mid i = 0, 1, ..., \lfloor N/3 \rfloor - 1\}$. Similarly to the proof of Theorem 2, we define the vector of channel inputs corresponding to these demand combination as $\bar{X}_{12}^n = \bigcup_{i=0}^{\lfloor N/3 \rfloor - 1} X_{(3i+1, 3i+2)}^n$, and $\bar{X}_{23}^n = \bigcup_{i=0}^{\lfloor N/3 \rfloor - 1} X_{(3i+2, 3i+3)}^n$. Then for any (n, M_1, M_2, R)

caching code that achieves the distortion tuple (D_1, D_2), we have

$$2\lfloor N/3 \rfloor nR + nM_1 + nM_2$$

$$\geq H(\bar{X}_{12}^n, Z_2^n) + H(\bar{X}_{23}^n, Z_1^n) \tag{42a}$$

$$= H(\bar{X}_{12}^n, Z_2^n | \bar{S}_2^n) + H(\bar{X}_{23}^n, Z_1^n | \bar{S}_2^n)$$
$$+ I(\bar{S}_2^n; \bar{X}_{12}^n, Z_2^n) + I(\bar{S}_2^n; \bar{X}_{23}^n, Z_1^n) \tag{42b}$$

$$\geq H(\bar{X}_{12}^n, Z_2^n, \bar{X}_{23}^n, Z_1^n | \bar{S}_2^n) + I(\bar{S}_2^n; \bar{X}_{12}^n, Z_2^n)$$
$$+ I(\bar{S}_2^n; \bar{X}_{23}^n, Z_1^n) \tag{42c}$$

$$\geq I(\bar{S}_1^n, \bar{S}_3^n; \bar{X}_{12}^n, Z_2^n, \bar{X}_{23}^n, Z_1^n | \bar{S}_2^n)$$
$$+ I(\bar{S}_2^n; \bar{X}_{12}^n, Z_2^n) + I(\bar{S}_2^n; \bar{X}_{23}^n, Z_1^n) \tag{42d}$$

where $\bar{S}_1^n = \bigcup_{i=0}^{\lfloor N/3 \rfloor - 1} S_{3i+1}^n$, $\bar{S}_2^n = \bigcup_{i=0}^{\lfloor N/3 \rfloor - 1} S_{3i+2}^n$, and $\bar{S}_3^n = \bigcup_{i=0}^{\lfloor N/3 \rfloor - 1} S_{3i+3}^n$. We remind that $\hat{S}_{i,k}^n$ denotes the reconstruction of S_i^n at user k, for $i = 1, ..., N$ and $k = 1, 2$. We define $\hat{\bar{S}}_{11}^n = \bigcup_{i=0}^{\lfloor N/3 \rfloor - 1} \hat{S}_{3i+1,1}^n$, $\hat{\bar{S}}_{21}^n = \bigcup_{i=0}^{\lfloor N/3 \rfloor - 1} \hat{S}_{3i+2,1}^n$, $\hat{\bar{S}}_{22}^n = \bigcup_{i=0}^{\lfloor N/3 \rfloor - 1} \hat{S}_{3i+2,2}^n$ and $\hat{\bar{S}}_{32}^n = \bigcup_{i=0}^{\lfloor N/3 \rfloor - 1} \hat{S}_{3i+3,2}^n$. Following the similar arguments as in (38) and (39), we have

$$I(\bar{S}_2^n; \bar{X}_{12}^n, Z_2^n)$$
$$\geq I(\bar{S}_2^n; \hat{\bar{S}}_{22}^n) \geq n\lfloor N/3 \rfloor R(D_2) = n\lfloor N/3 \rfloor r_2^*, \tag{43a}$$
$$I(\bar{S}_2^n; \bar{X}_{23}^n, Z_1^n)$$
$$\geq I(\bar{S}_2^n; \hat{\bar{S}}_{21}^n) \geq n\lfloor N/3 \rfloor R(D_1) = n\lfloor N/3 \rfloor r_1^*. \tag{43b}$$

We also have

$$I(\bar{S}_1^n, \bar{S}_3^n; \bar{X}_{12}^n, Z_2^n, \bar{X}_{23}^n, Z_1^n | \bar{S}_2^n)$$
$$\geq I(\bar{S}_1^n, \bar{S}_3^n; \bar{X}_{12}^n, Z_2^n, \bar{X}_{23}^n, Z_1^n) - I(\bar{S}_1^n, \bar{S}_3^n; \bar{S}_2^n) \tag{44a}$$
$$\geq I(\bar{S}_1^n; \bar{X}_{12}^n, Z_2^n, \bar{X}_{23}^n, Z_1^n) + I(\bar{S}_3^n; \bar{X}_{12}^n, Z_2^n, \bar{X}_{23}^n, Z_1^n | \bar{S}_1^n) \tag{44b}$$
$$\geq I(\bar{S}_1^n; \bar{X}_{12}^n, Z_1^n) + I(\bar{S}_3^n; \bar{X}_{23}^n, Z_2^n) \tag{44c}$$
$$\geq I(\bar{S}_1^n; \hat{\bar{S}}_{11}^n) + I(\bar{S}_3^n; \hat{\bar{S}}_{32}^n) \tag{44d}$$
$$\geq n\lfloor N/3 \rfloor R(D_1) + n\lfloor N/3 \rfloor R(D_2) \tag{44e}$$
$$= n\lfloor N/3 \rfloor r_1^* + n\lfloor N/3 \rfloor r_2^*, \tag{44f}$$

where (44b) follows since $\bar{S}_1^n$, $\bar{S}_2^n$ and $\bar{S}_3^n$ are independent; (44c) follows due to the nonnegativity of mutual information; and inequality (44d) is due to the data processing inequality and the fact that $\bar{S}_1^n - (\bar{X}_{12}^n, Z_1^n) - \hat{\bar{S}}_{11}^n$ and $\bar{S}_3^n - (\bar{X}_{23}^n, Z_2^n) - \hat{\bar{S}}_{32}^n$ are both Markov chains. Substituting inequalities (43a), (43b) and (44f) into (42d), we obtain

$$R \geq r_1^* + r_2^* - \frac{(M_1 + M_2)}{2\lfloor N/3 \rfloor} \tag{45}$$

Here, we prove Eqn. (28). We have

$$\sum_{k=1}^{K-i+1} \sum_{\substack{\mathcal{S} \subset \{i,...,K\} \\ |\mathcal{S}|=k}} p^i_{(d_{r'(\mathcal{S})}, D_{r'(\mathcal{S})}), \mathcal{S} \setminus \{r'(\mathcal{S})\}}$$

$$= \sum_{k=1}^{K-i+1} \sum_{\substack{\mathcal{S} \subset \{1,...,K-i+1\} \\ |\mathcal{S}|=k}} p^i_{(d_{m(\mathcal{S})}, D_{m(\mathcal{S})}), \mathcal{S} \setminus \{m(\mathcal{S})\}} \tag{46a}$$

$$= \sum_{k=1}^{K-i+1} \sum_{l=1}^{K-i-k+2} \sum_{\substack{\mathcal{S} \subset \{l+1,...,K+1-i\} \\ |\mathcal{S}|=k-1}} p^i_{(d_l, D_l), \mathcal{S}} \tag{46b}$$

$$= \sum_{l=1}^{K-i+1} \sum_{k=1}^{K-i-l+2} \sum_{\substack{\mathcal{S} \subset \{l+1,...,K+1-i\} \\ |\mathcal{S}|=k-1}} p^i_{(d_l, D_l), \mathcal{S}} \tag{46c}$$

$$= \sum_{l=1}^{K-i+1} \sum_{k=1}^{K-i-l+2} \sum_{\substack{\mathcal{S} \subset \{l+1,...,K+1-i\} \\ |\mathcal{S}|=k-1}} \prod_{s=1}^{l} (1 - t_s^i) \cdot$$
$$\prod_{s \in \{l+1,...,K+1-i\} \setminus \mathcal{S}} (1 - t_s^i) \prod_{s \in \mathcal{S}} t_s^i \tag{46d}$$

$$= \sum_{l=1}^{K+1-i} \prod_{s=1}^{l} (1 - t_s^i) \cdot \left[\sum_{k=1}^{K-i-l+2} \sum_{\substack{\mathcal{S} \subset \{l+1,...,K+1-i\} \\ |\mathcal{S}|=k-1}} \right.$$
$$\left. \prod_{s \in \{l+1,...,K+1-i\} \setminus \mathcal{S}} (1 - t_s^i) \prod_{s \in \mathcal{S}} t_s^i \right], \tag{46e}$$

where equality (46a) holds by ordering and re-indexing users $\{i, ..., K\}$ to $\{1, ..., K+1-i\}$ such that $\frac{M_k}{N r_k} = t_k^i$, for $k = 1, ..., K + 1 - i$; equality (46b) comes from iterating the smallest index in the multicasting set from 1 to $K - i - k + 2$, which is the maximum that the value of the smallest index could be in a multicasting set with cardinality equals to k; equality (46c) is derived by switching the order of summations regarding to l and k; equality (46d) is obtained by expressing $p^i_{(d_l, D_l), \mathcal{S}}$ as a function of t_s^i; since $\prod_{s=1}^{l} (1 - t_s^i)$ is independent of the value of k and subset $\mathcal{S}$, equality (46e) holds. Since $\sum_{j=1}^{K-i-l+2} \sum_{\substack{\mathcal{S} \subset \{l+1,...,K+1-i\} \\ |\mathcal{S}|=j-1}} \prod_{s \in \{l+1,...,K+1-i\} \setminus \mathcal{S}} (1 - t_s^i) \prod_{s \in \mathcal{S}} t_s^i = 1$, we have

$$\sum_{j=1}^{K+1-i} \sum_{\substack{\mathcal{S} \subset \{i,...,K\} \\ |\mathcal{S}|=j}} p^i_{(d_{r'(\mathcal{S})}, D_{r'(\mathcal{S})}), \mathcal{S} \setminus \{r'(\mathcal{S})\}}$$

$$= \sum_{l=1}^{K+1-i} \prod_{s=1}^{l} (1 - t_s^i) = \sum_{l=1}^{K+1-i} \prod_{k=1}^{l} (1 - t_k^i), \tag{47}$$

which yields (28).

REFERENCES

[1] L. Fan, P. Cai, J. Almeida and A. Z. Broder, "Summary Cache: a scalable wide-area web cache sharing protocol," *IEEE/ACM Trans. Networking*, vol. 8, no. 3, pp. 281–293, Jun. 2000.

[2] K. C. Almeroth and M. H. Ammar, "The use of multicast delivery to provide a scalable and interactive video-on-demand service," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 6, pp. 1110–1122, Aug. 1996.

[3] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch and G. Caire, "Femtocaching: wireless video content delivery through distributed caching helpers," *IEEE Trans. Inform. Theory*, vol. 59, no. 12, pp. 1110–1122, Dec. 2013.

[4] 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, May. 2016.

[5] 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.

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

[7] A. S. Cacciapuoti, M. Caleffi, M. Ji, J. Llorca and A. M. Tulino, "Speeding up future video distribution via channel-aware caching-aided coded multicast," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 8, pp. 2207-2218, Jun. 2016.

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

[9] Z. Chen, P. Fan and K. B. Letaief, "Fundamental limits of caching: Improved bounds for small buffer users," *IET Communications*, vol. 10, no. 17, pp. 2315–2318, Nov. 2016.

[10] 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.

[11] K. Wan, D. Tuninetti and P. Piantanida, "On caching with more users than files," in *Proc. IEEE Int'l Symp. on Inform. Theory (ISIT)*, Barcelona, Spain, Jul. 2016.

[12] R. Pedarsani, M. Maddah-Ali and U. Niesen, "Online coded caching," *IEEE/ACM Trans. Networking*, vol. 24, no. 2, pp. 836–845, April. 2016.

[13] N. Karamchandani, U. Niesen, M. Maddah-Ali, and S. Diggavi "Hierarchical coded caching," *IEEE Trans. Inform. Theory*, vol. 62, no. 6, pp. 3212–3229, Apr. 2016.

[14] M. Ji, G. Caire and A. F. Molisch, "Fundamental limits of distributed caching in D2D wireless networks," in *Proc. IEEE Inform. Theory Workshop (ITW)*, Jeju Island, Korea, Oct. 2015.

[15] R. Timo and M. Wigger, "Joint cache-channel coding over erasure broadcast channels," in *Proc. IEEE Int'l Symp. on Wireless Commun. Syst. (ISWCS)*, Brussels, Belgium, Aug. 2015.

[16] M. Mohammadi Amiri and D. Gündüz, "Cache-aided content delivery over erasure broadcast channels," *IEEE Trans. Commun.*, vol. 66, no. 1, pp. 370-381, Jan. 2018.

[17] M. Mohammadi Amiri and D. Gündüz, "Caching and coded delivery over Gaussian broadcast channels for energy efficiency," *arXiv:1712.03433v1 [cs.IT]*, Dec. 2017.

[18] J. Zhang and P. Elia, "Fundamental limits of cache-aided wireless BC: interplay of coded-caching and CSIT feedback," *IEEE Trans. Inform. Theory*, vol. 63, no. 5, pp. 3142–3160, Feb. 2017.

[19] M. Maddah-Ali and U. Niesen, "Cache-aided interference channels," in *Proc. IEEE Int'l Symp. on Inform. Theory (ISIT)*, Hongkong, China, Jun. 2015.

[20] J. S. P. Roig, D. Gündüz, and F. Tosato "Interference networks with caches at both ends," in *Proc. IEEE Int'l. Conf. Commun. (ICC)*, Paris, France, May. 2017, pp. 1938–1883.

[21] A. Sengupta, R. Tandon and O. Simeone, "Cache aided wireless networks: tradeoffs between storage and latency," *arXiv:1512.07856v1 [cs.IT]*, Dec. 2015.

[22] F. Xu, M. Tao and K. Liu, "Fundamental tradeoff between storage and latency in cache-aided wireless interference networks," *arXiv:1605.00203v2 [cs.IT]*, May. 2016.

[23] T. Stockhammer, "Dynamic adaptive streaming over HTTP: Standards and design principles," in *Proc. ACM MMsys*, San Jose, CA, Feb. 2015, pp. 133–144.

[24] H. Schwarz, D. Marpe and T. Wiegand, "Overview of the scalable video coding extension of the H. 264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, Sept. 2014.

[25] P. Hassanzadeh, E. Erkip, J. Llorca and A. Tulino, "Distortion-memory tradeoffs in cache-aided wireless video delivery," *arXiv:1511.03932v1 [cs.IT]*, Nov. 2015.

[26] M. Ji, A. M. Tulino, J. Llorca and G. Caire, "Order-optimal rate of caching and coded multicasting with random demands," *IEEE Trans. Inform. Theory*, vol. 63, no. 6, pp. 3923–3949, Apr. 2017.

[27] R. Timo, S. S. Bidokthi, M. Wigger and B. Geiger, "A rate-distortion approach to caching," in *Proc. Int'l. Zurich Seminar (IZS)*, Zurich, Switzerland, Mar. 2016.

[28] T. M. Cover and J. A. Thomas, *Elements of information theory*, John Wiley & Sons, 2012.

[29] T. M. Cover and W. H. Equitz, "Successive refinement of information," *IEEE Trans. Inform. Theory*, vol. 37, no. 2, pp. 269–275, Mar. 1991.

[30] S. Wang, W. Li, X. Tian and H. Liu, "Coded caching with heterogeneous cache sizes," *arXiv:1504.01123v3 [cs.IT]*, Aug. 2015.

[31] J. Zhang, X. Lin, C-C. Wang and X. Wang, "Coded caching for files with distinct file sizes," in *Proc. IEEE Int'l Symp. on Inform. Theory (ISIT)*, Hongkong, Jun. 2015, pp. 1686–1690.

[32] M. Mohammadi Amiri, Q. Yang and D. Gündüz, "Decentralized coded caching with distinct cache capacities," *IEEE Trans. Commun.*, vol. 65, no. 11, pp. 4657–4669, Nov. 2017.

[33] A. Sengupta, R. Tandon and T. C. Clancy, "Improved approximation of storage-rate tradeoff for caching via new outer bounds," in *Proc. IEEE Int'l Symp. on Inform. Theory (ISIT)*, Hongkong, Jun. 2015, pp. 1691–1695.

[34] H. Ghasemi and A. Ramamoorthy, "Improved lower bounds for coded caching," in *Proc. IEEE Int'l Symp. on Inform. Theory (ISIT)*, Hongkong, Jun. 2015, pp. 1696–1700.

[35] Q. Yang and D. Gündüz, "Centralized Coded Caching for Heterogeneous Lossy Requests," in *Proc. IEEE Int'l Symp. on Inform. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 405–409.

Qianqian Yang (S'12) received the B.S. degree in automation from Chongqing University in 2011, and the M.S. degree in control engineering from Zhejiang University in 2014. She is currently a Ph.D. candidate with Imperial College London. Her current research interests include information and coding theory, resource optimization in wireless sensor networks, neural network and machine learning.

Deniz Gündüz (S'03-M'08-SM'13) received the B.S. degree in electrical and electronics engineering from METU, Turkey in 2002, and the M.S. and Ph.D. degrees in electrical engineering from NYU Polytechnic School of Engineering (formerly Polytechnic University) in 2004 and 2007, respectively. After his PhD, he served as a postdoctoral research associate at Princeton University, and as a consulting assistant professor at Stanford University. He was a research associate at CTTC in Barcelona, Spain until September 2012, when he joined the Electrical and Electronic Engineering Department of Imperial College London, UK, where he is currently a Reader in information theory and communications, and leads the Information Processing and Communications Lab.

His research interests lie in the areas of communications and information theory, machine learning, and security and privacy in cyber-physical systems. Dr. Gündüz is an Editor of the IEEE TRANSACTIONS ON COMMUNICATIONS, and the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING. He is the recipient of the IEEE Communications Society - Communication Theory Technical Committee (CTTC) Early Achievement Award in 2017, a Starting Grant of the European Research Council (ERC) in 2016, IEEE Communications Society Best Young Researcher Award for the Europe, Middle East, and Africa Region in 2014, Best Paper Award at the 2016 IEEE Wireless Communications and Networking Conference (WCNC), and the Best Student Paper Award at the 2007 IEEE International Symposium on Information Theory (ISIT). He is the General Co-chair of the 2018 Workshop on Smart Antennas, and previously served as the General Co-chair of the 2016 IEEE Information Theory Workshop, and a Co-chair of the PHY and Fundamentals Track of the 2017 IEEE Wireless Communications and Networking Conference.