

# Fundamental limits of caching: Improved rate-memory tradeoff with coded prefetching

Jesus Gomez-Vilardebo

Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA)

Castelldefels, Spain

jesus.gomez@cttc.es

**Abstract**—We consider a cache network in which a single server is connected to multiple users via a shared error free link. The server has access to a database with  $N$  files of equal length  $F$ , and serves  $K$  users each with a cache memory of  $MF$  bits. A novel centralized coded caching scheme is proposed for scenarios with more users than files  $N \leq K$  and cache capacities satisfying  $\frac{1}{K} \leq M \leq \frac{N}{K}$ . The proposed scheme outperforms the best rate-memory region known in the literature if  $N \leq K \leq \frac{N^2+1}{2}$ .

**Index Terms**—Centralized coded caching, network coding, index coding.

## I. INTRODUCTION

Content caching techniques are recently increasing attention to combat peak hour traffic in content delivery services. The basic idea is simple. If contents are made available at user terminals during low traffic periods, then the peak rate can be reduced. However, content requests are unknown to the server and thus content caching at user memories must be carefully chosen in order to be useful regardless of the contents requested during peak hours. The simplest caching scheme consists of storing each file partially at each user memory. Then, the server transmits the remaining requested data uncoded [1], [2]. For single user caching systems, this strategy is optimal. However, for multi-user systems, the seminal work in [3] by Maddah-Ali and Niesen shows that important gains can be obtained by a new coded caching strategy. Specifically, there authors show that, besides the local caching gain that is obtained by placing contents at user caches before they are requested, it is possible to obtain a global caching gain by creating broadcast opportunities. This is, by carefully choosing the content caches at different users, and using network coding techniques it is possible to transform the initial multi-cast network, where every user is requesting a different file, into a broadcast network, where every user requests exactly the same “coded” file, obtaining the new global caching gain.

The fundamental caching scheme developed in [3] was latter extended to more realistic situations. The decentralized setting was considered in [4], non-uniform demands in [5]–[7], and online coded caching in [8], hierarchical cache network were considered in [9], [10], among others. In addition, new schemes pushing further the fundamental limits of caching systems have appeared in [11]–[19]. There have been also

efforts to obtain theoretical lower bounds on the delivery rate. The cut-set bound was studied in [3]. A tighter lower bound was obtained in [20]. Through a computational approach a lower-bound for the special case  $N = K = 3$  is derived in [21]. Other lower bounds have appeared in [13], [22], [23].

The work here proposed investigates the fundamental achievable rate for the particular situation where there are more users than files, and the caching memories at users are small compared to the number of files in the system. Besides its theoretical relevance, this situation can be readily found in the real world. For instance, global content delivery services such a Netflix serve a few multimedia contents to millions of users across the world. In addition, it was shown in [5] that a near optimal caching strategy consists in dividing the files into groups with similar popularity, and then applying the coded caching strategy to each group separately. Since the amount of users in each groups remains the same, when there are many groups, the cache size dedicated to each group is small as well as the number of files per user in each group.

The rest of this paper is organized as follows. In Section II, we present the system model together with the more relevant previous results. In Section III, we summarize the main results. Section IV describes the caching scheme proposed. Finally, conclusions are drawn in Section V.

## II. SYSTEM MODEL AND PREVIOUS RESULTS

We consider a communication system with one server connected to  $K$  users, denoted as  $U_1, \dots, U_K$ , through a shared, error-free link. There is a database at the server with  $N$  files, each of length  $F$  bits, denoted as  $W_1, \dots, W_N$ . Each user is equipped with a local cache of capacity  $MF$  bits and is assumed to request only one full file. Here, we consider the special case where  $M \in [0, \frac{N}{K}]$  and there are more users than files  $N \leq K$ . For convenience, we define parameter  $q = \frac{N}{MK}$ .

We consider the communication model introduced in [3]. The caching system operates in two phases: the *placement phase* and the *delivery phase*. In the placement phase, users have access to the server database, and each user fills his cache. As in [15], we allow coding in the prefetching phase. Then, each user  $U_k$  requests a single full file  $W_{\mathbf{d}(k)}$  where  $\mathbf{d} = (\mathbf{d}(1), \dots, \mathbf{d}(K))$  denotes the demand vector. We denote the number of distinct requests in  $\mathbf{d}$  as  $N_e(\mathbf{d})$ . In the delivery phase, only the server has access to the database. After being informed of the user demands, the server transmits a signal

This work was partially supported by the Catalan Government under grant SGR2017-1479 and the Spanish Government under grant TEC2013-44591-P (INTENSIV).

$Y$  of size  $RF$  bits over the shared link to satisfy all user requests simultaneously. The signal  $Y$  is a function of the demand vector  $\mathbf{d}$ , all the files in the data base  $W_1, \dots, W_N$ , and the content in the user caches  $\mathcal{M} = \{\mathcal{M}_1, \dots, \mathcal{M}_k\}$ . Using the local cache content and the received signal  $Y$ , each user  $U_k$  reconstructs its requested file  $W_{\mathbf{d}(k)}$ .

Let  $\mathcal{D} = \{1, \dots, N\}^K$ , for a caching system  $(M, N, K)$ , given a particular prefetching  $\mathcal{M} = \{\mathcal{M}_1, \dots, \mathcal{M}_k\}$  and a particular demand  $\mathbf{d}$ , we say that communication rate  $R$  is achievable if and only if there exists a message  $Y$  of length  $RF$  bits such that every user  $U_k$  is able to reconstruct its desired file  $W_{\mathbf{d}(k)}$ . For a particular prefetching  $\mathcal{M}$  and demand  $\mathbf{d}$ , we denote the achievable rate as  $R(\mathbf{d}, \mathcal{M})$ . Then, the rate needed for the worst demand is given by  $R^*(\mathcal{M}) = \max_{\mathbf{d} \in \mathcal{D}} R(\mathbf{d}, \mathcal{M})$ . Finally, we define the rate-memory pair  $(R^*, M)$  and the rate-memory function  $R^*(M)$ .

#### A. Previous Results

For the special case considered here  $M \in [0, \frac{N}{K}]$  and  $N \leq K$ , the best known rate-memory function can be obtained by memory sharing between two achievable rate-memory pairs

$$(R_{\text{GBC}}^*, M_{\text{GBC}}) = \left( N - \frac{N(N+1)}{2K}, \frac{N}{K} \right), \quad (1)$$

and the rate-memory pairs obtained in [17]

$$(R_{\text{MDS}}^*, M_{\text{MDS}}) = \left( \frac{N(K-t)}{K}, \frac{t[(N-1)t + K - N]}{K(K-1)} \right), \quad (2)$$

with  $t = 0, 1, \dots, K$ . The lower convex envelope of all these rate-memory pairs, provides the best rate-memory function in the literature. The rate-memory pairs in (2) include as special case the rate-memory pair

$$(R_{\text{CFL}}^*, M_{\text{CFL}}) = \left( N - \frac{N}{K}, \frac{1}{K} \right) \quad (3)$$

obtained in [15]. The design of the scheme here proposed was initially motivated to connect the schemes achieving the rate memory-pairs in [15] and (1), beyond the simple memory sharing between both rate-memory points. The scheme developed in [17] for situations with more users than files  $K \geq N$  makes use of binary codes, in particular maximum distant separable (MDS) codes and rank metric codes to obtain the rate-memory pairs in (2), which are shown to be optimal at certain points. A method to obtain new rate-memory points is described in [18]. However no explicit characterization of these rate-memory pairs is given. There have been other coded prefetching schemes proposed in the literature, see [14] and [12] but either they do not improve the current best known rate-memory trade-off or they apply to other situations. The optimal rate-memory trade-off for a caching systems remains an open problem. Besides the achievable rate-memory trade-off described above, there have been efforts to obtain theoretical lower bounds on the delivery rate. The cut-set bound was studied in [3]. A tighter lower bound was obtained in [20]. Through a computational approach a lower-bound for the special case  $N = K = 3$  is derived in [21]. Other lower bounds have appeared in [13], [22], [23].

### III. MAIN RESULT

The following theorem presents the delivery rate obtained by the proposed caching scheme for a particular demand  $\mathbf{d}$ .

*Theorem 1.* For a caching problem with  $K$  users and  $N$  files, local cache size of  $M$  files at each user, and parameter  $q = \frac{N}{MK}$ . Given a particular demand  $\mathbf{d}$ , let  $N_e(\mathbf{d})$  be the number of distinct file requests, then the delivery rate

$$R = \frac{KN_e(\mathbf{d}) \binom{N-1}{q-1} - q \binom{N_e(\mathbf{d})+1}{q+1}}{K \binom{N-1}{q-1}} \quad (4)$$

is achievable for  $q \in \{1, \dots, N\}$ . Furthermore, for  $q \in [1, N]$  the rate-memory pairs in the lower convex envelope of its values at  $q \in \{1, \dots, N\}$  are achievable. .

We prove this result in the following section by describing the new caching scheme. The delivery rate presented in Theorem 1 is valid for any  $K$  and  $N$ , however, it is particularly useful for  $K \geq N$ , as we detail in the next corollaries.

The next corollary, establishes the achievable rate memory pair for the worst case demand. The proof is provided in the Appendix.

*Corollary 1.1.* For a caching problem with  $K$  users and  $N$  files, local cache size of  $M$  files at each user, and parameter  $q = \frac{N}{MK}$ , the delivery rate-memory pairs  $(R^*, M)$

$$(R^*, M) = \left( \bar{N} - \frac{N\bar{N}+1}{K} \frac{\binom{\bar{N}}{q}}{q+1}, \frac{N}{Kq} \right) \quad (5)$$

with  $\bar{N} = \min(N, K)$  are achievable for  $q \in \{1, \dots, N\}$ . Furthermore, for  $q \in [1, N]$  the rate-memory pairs in the lower convex envelope of its values at  $q \in \{1, \dots, N\}$  are achievable.

*Remark 1.* The rate-memory function in (5) coincides with (3) for  $q = N$  and with (1) for  $q = 1$ . For  $q = 1$ ,  $M = \frac{N}{K}$  our scheme is essentially the same as the one described in [16] and [11]. However, the scheme proposed in [15] to achieve (3) differs slightly from the one considered here for  $K > N$ . Indeed, while [15] divides each file into  $NK$  subfiles, our scheme requires only  $K$  subfiles per file.

The next corollaries compare the proposed scheme with the scheme presented [16], the scheme in [17], the cut set bound derived in [3], and the outer bound obtained in [20]. Due to space limitations, theses results are proved in [24].

*Corollary 1.2.* For a caching problem with  $K$  users and  $N$  files,  $K \geq N$ , and local cache size of  $\frac{1}{K} \leq M \leq \frac{N}{K}$ , a sufficient condition for the proposed scheme to outperform the rate-memory region obtained by the lower convex envelope of the rate-memory pairs in (2), is  $K \leq \frac{N^2+1}{2}$ .

*Corollary 1.3.* Let  $R_{\text{CB}}(M)$  denote the rate-memory function obtained for the cut set bound. The rate difference between the cut set rate and the rate-memory pairs in Corollary 1.1 is

$$R_{\text{CB}}(M) - R^*(M) = \frac{N}{K} \left( \frac{N}{q} - \frac{N+1}{q+1} \right).$$

| Prefetching User $U_i$ | $M$           | Coded cached subfiles   |  |  |
|------------------------|---------------|---|--|--|
| $q = 1$                | $\frac{1}{2}$ | $Z_1^{(i)} = W_1^{(i)}$   | $Z_2^{(i)} = W_2^{(i)}$  | $Z_3^{(i)} = W_3^{(i)}$  |
| $q = 2$                | $\frac{1}{4}$ | $Z_{\{1,2\}}^{(i)} = W_{1,\{1,2\}}^{(i)} \oplus W_{2,\{1,2\}}^{(i)}$                                    | $Z_{\{1,3\}}^{(i)} = W_{1,\{1,3\}}^{(i)} \oplus W_{3,\{1,3\}}^{(i)}$ | $Z_{\{2,3\}}^{(i)} = W_{2,\{2,3\}}^{(i)} \oplus W_{3,\{2,3\}}^{(i)}$ |
| $q = 3$                | $\frac{1}{6}$ | $Z_{\{1,2,3\}}^{(i)} = W_{1,\{1,2,3\}}^{(i)} \oplus W_{2,\{1,2,3\}}^{(i)} \oplus W_{3,\{1,2,3\}}^{(i)}$ |  |  |

TABLE I: Prefetching schemes at user  $U_i$  for the proposed coded caching scheme when  $K = 6$ ,  $N = 3$ , and  $q \in \{1, 2, 3\}$ .

|                             |   |
|-----------------------------|---|
| Subfiles                    | $\mathcal{T}(q) = \{(i, f, \mathcal{A}) : \text{for all } i \in \mathcal{K}, \mathcal{A} \subseteq \mathcal{F},  \mathcal{A}  = q \text{ and } f \in \mathcal{A}\}$             |
| Requested subfiles          | $\mathcal{R}(q, \mathbf{d}) = \{(i, f, \mathcal{A}) \in \mathcal{T}(q) : \text{for all } f \in \mathcal{N}_e(\mathbf{d})\}$   |
| Requested subfiles Type I   | $\mathcal{R}_I(q, \mathbf{d}) = \{(i, f, \mathcal{A}) \in \mathcal{R}(q, \mathbf{d}) : \mathcal{A} \not\subseteq \mathcal{N}_e(\mathbf{d})\}$                                   |
| Requested subfiles Type II  | $\mathcal{R}_{II}(q, \mathbf{d}) = \{(i, f, \mathcal{A}) \in \mathcal{R}(q, \mathbf{d}) : \mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d}), \mathbf{d}(i) \in \mathcal{A}\}$     |
| Requested Subfiles Type III | $\mathcal{R}_{III}(q, \mathbf{d}) = \{(i, f, \mathcal{A}) \in \mathcal{R}(q, \mathbf{d}) : \mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d}), \mathbf{d}(i) \notin \mathcal{A}\}$ |

TABLE II: 3-tuple sets identifying subfiles  $W_{f,\mathcal{A}}^{(i)}$ .

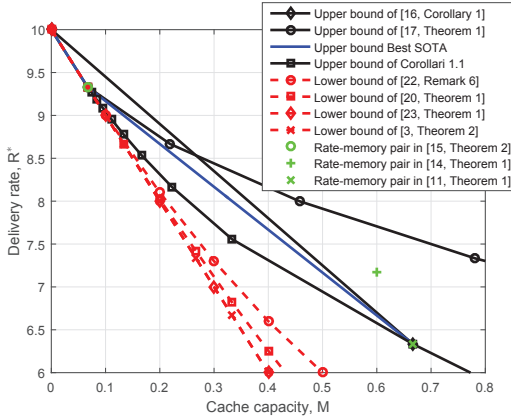


Fig. 1: Rate-memory functions of the proposed scheme in Corollary 1.1 compared with existing schemes and lower bounds in the literature for  $N = 10$  and  $K = 15$ .

Observe that, as first reported in [15, Theorems 3 and 4], for  $q = N$ ,  $M = \frac{1}{K}$  the cut set lower bound is achieved by the proposed strategy.

*Corollary 1.4.* Let  $R_{\text{STC}}(M)$  denote the outer bound on the rate-memory function presented in [20]. For  $K = N$  and  $M = \frac{N}{(N-1)K} = \frac{1}{N-1}$ , this bound is achievable by the rate-memory function in Corollary 1.1. This is  $R^* \left( \frac{1}{N-1} \right) = R_{\text{STC}} \left( \frac{1}{N-1} \right)$ .

We conclude this section by illustrating in Fig. 1 the worst demand rate-memory function for the proposed scheme Corollary 1.1 and for the state of the art (SOTA). We consider the case  $N = 10$  files and  $K = 15$  users. We provide the rate-memory regions in [16, Corollary 1], [17, Theorem 1], the rate memory pairs in [15, Theorem 2], [14, Theorem 1], and [11, Theorem 1]. We also include in this figure for comparison, the cut set lower bound [3, Theorem 2], the information-theoretical lower bound obtained in [20, Theorem 1], and the lower bounds recently appeared in [23, Theorem 1] and [22, Remark 6]. We observe that, for the special situation considered here, the new proposed scheme obtains a significant improvement with respect to the previous best SOTA.

#### IV. PROPOSED CACHING SCHEME

In this section, we describe the caching scheme proposed. We provide an example in the next section. Let us define the set of user indexes as  $\mathcal{K} = \{1, \dots, K\}$ , and the set of file indexes as  $\mathcal{F} = \{1, \dots, N\}$ . Consider a cache capacity at users of  $M = \frac{N}{qK}$ . To achieve the rate  $R$  stated in Theorem 1, we present a prefetching and delivery scheme for  $q \in \{1, \dots, N\}$ , since for general  $\frac{1}{K} \leq M \leq \frac{N}{K}$ , the minimum rate can be achieved by memory sharing.

*Prefetching scheme:* Given  $q \in \{1, \dots, N\}$ , consider the  $\binom{N}{q}$  possible subsets  $\mathcal{A} \subseteq \mathcal{F}$  of  $q$  different files,  $|\mathcal{A}| = q$ .

First, each file  $W_f$  is partitioned into  $K$  parts,  $W_f^{(i)}$ , one for each user  $i \in \mathcal{K}$ . Then, each of these parts is further partitioned into  $\binom{N-1}{q-1}$  subfiles  $W_{f,\mathcal{A}}^{(i)}$ , one for each subset  $\mathcal{A}$  of  $q$  files that satisfies  $f \in \mathcal{A}$ . Thus, we broke each file into a total of  $K \binom{N-1}{q-1}$  subfiles. Finally, user  $U_i$  computes and stores the *coded cached subfiles*

$$Z_{\mathcal{A}}^{(i)} = \bigoplus_{f \in \mathcal{A}} W_{f,\mathcal{A}}^{(i)}$$

for all subsets  $\mathcal{A} \in \{\mathcal{A} \subseteq \mathcal{F} : |\mathcal{A}| = q\}$ . Because there are  $\binom{N}{q}$  subsets  $\mathcal{A}$  and each subfile has  $F/K \binom{N-1}{q-1}$  bits, the required cache load at each user equals  $MF = \binom{N}{q} F/K \binom{N-1}{q-1} = \frac{N}{qK} F$  bits. The coded cached subfiles at user  $U_i$  for a particular scenario with  $N = 3$  and  $q \in \{1, \dots, 3\}$  are detailed in Table I.

*Delivery scheme:* Consider the sets defined in Table II. Each subfile  $W_{f,\mathcal{A}}^{(i)}$  satisfies  $f \in \mathcal{A}$  and is only XORed in one coded cached subfile  $Z_{\mathcal{A}}^{(i)}$ . Thus, each subfile  $W_{f,\mathcal{A}}^{(i)}$  can be identified by a 3-tuple  $(i, f, \mathcal{A}) \in \mathcal{T}(q)$ . Let  $\mathcal{N}_e(\mathbf{d})$  be the set of requested files. Then, the subfiles requested by some user satisfy  $f \in \mathcal{N}_e(\mathbf{d})$ , and are thus, identified by the 3-tuples  $(i, f, \mathcal{A}) \in \mathcal{R}(q, \mathbf{d})$ . The delivery scheme proposed here divides the requested subfiles into three types, and obtains the requested subfiles in each type, separately. The requested subfiles Type I,  $(i, f, \mathcal{A}) \in \mathcal{R}_I(q, \mathbf{d})$ , are

those requested subfiles, which are coded in the coded cached subfiles  $Z_{\mathcal{A}}^{(i)}$  together with at least one subfile not requested by any user  $\mathcal{A} \not\subseteq \mathcal{N}_e(\mathbf{d})$ . The requested subfiles Type II,  $(i, f, \mathcal{A}) \in \mathcal{R}_{II}(q, \mathbf{d})$ , include all the requested subfiles coded in the coded cached subfiles  $Z_{\mathcal{A}}^{(i)}$  together with other requested files  $\mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d})$  and placed in the cache of a user  $U_i$  requesting one of them, i.e.  $\mathbf{d}(i) \in \mathcal{A}$ . Finally, the requested subfiles Type III,  $(i, f, \mathcal{A}) \in \mathcal{R}_{III}(q, \mathbf{d})$ , include all the requested subfiles coded in the coded cached subfiles  $Z_{\mathcal{A}}^{(i)}$  together with other also requested files  $\mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d})$ , different from the file requested by the user caching them, i.e.  $\mathbf{d}(i) \notin \mathcal{A}$ .

Delivery of requested subfiles Type I: For these subfiles, the server simply broadcasts them one by one, i.e.  $Y_{f, \mathcal{A}}^{(i)} = W_{f, \mathcal{A}}^{(i)}$  for all  $(i, f, \mathcal{A}) \in \mathcal{R}_I(q, \mathbf{d})$ . Given that

$$\mathcal{R}_I(q, \mathbf{d}) = \mathcal{T}(q, \mathbf{d}) \setminus \{(i, f, \mathcal{A}) \in \mathcal{R}(q, \mathbf{d}) : \mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d})\}.$$

The total number of requested subfiles Type I, and thus of broadcasted subfiles Type I is

$$\begin{aligned} T_I &= \sum_{(i, f, \mathcal{A}) \in \mathcal{T}(q, \mathbf{d})} 1 - \sum_{(i, f, \mathcal{A}) \in \mathcal{R}(q, \mathbf{d}) : \mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d})} 1 \\ &= \sum_{i \in \mathcal{K}} \sum_{f \in \mathcal{N}_e(\mathbf{d})} \left( \sum_{\mathcal{A}: f \in \mathcal{A}} 1 - \sum_{\mathcal{A}: \{f \in \mathcal{A}, \mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d})\}} 1 \right) \\ &= K N_e(\mathbf{d}) \left( \binom{N-1}{q-1} - \binom{N_e(\mathbf{d})-1}{q-1} \right). \end{aligned} \quad (6)$$

Although not explicit written for brevity, we require  $\mathcal{A} \subseteq \mathcal{F}$  and  $|\mathcal{A}| = q$ . Equality (6) follows since, from right to left, first, the number of sets  $\mathcal{A} \subseteq \mathcal{F}$ , with  $|\mathcal{A}| = q$  that include a particular file  $f \in \mathcal{A}$  is  $\binom{N-1}{q-1}$ , whereas the number of sets satisfying  $\mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d})$  that include a particular file  $f \in \mathcal{A}$  is  $\binom{N_e(\mathbf{d})-1}{q-1}$ , second, there are  $N_e(\mathbf{d})$  files in  $\mathcal{N}_e(\mathbf{d})$  and, third, there are  $K$  users in  $\mathcal{K}$ .

Delivery of requested subfiles Type II: For these subfiles, first, the server arbitrarily selects one user *leader*  $u_f \in \mathcal{K}(f)$  for each file  $f \in \mathcal{F}$ . Let  $\bar{\mathcal{K}}(f)$  denote the set of users requesting a file different from  $W_f$ . Then, the server broadcasts

$$Y_{f, \mathcal{A}}^{(i)} = \begin{cases} W_{f, \mathcal{A}}^{(i)} & \text{if } i \in \bar{\mathcal{K}}(f) \\ W_{f, \mathcal{A}}^{(i)} \oplus W_{f, \mathcal{A}}^{(u_f)} & \text{if } i \in \mathcal{K}(f) \setminus u_f \end{cases} \quad (7)$$

for all  $(i, f, \mathcal{A}) \in \mathcal{R}_{II}(q, \mathbf{d})$  satisfying  $i \in \mathcal{K} \setminus u_f$ .

Next, we detail the decoding of subfiles Type II at user  $U_k$ . This user requests file  $W_{\mathbf{d}(k)}$  and, thus, is only interested in the subfiles  $W_{\mathbf{d}(k), \mathcal{A}}^{(i)}$  for all 2-tuples  $(i, \mathcal{A})$  such that  $(i, \mathbf{d}(k), \mathcal{A}) \in \mathcal{R}_{II}(q, \mathbf{d})$ . The decoding process at user  $k$  begins by computing

$$W_{\mathbf{d}(k), \mathcal{A}}^{(i)} = Z_{\mathcal{A}}^{(i)} \oplus \bigoplus_{f \in \mathcal{A} \setminus \mathbf{d}(k)} Y_{f, \mathcal{A}}^{(i)} \quad (12)$$

for all  $(i, \mathcal{A})$  such that  $i = k$  and  $(i, \mathbf{d}(k), \mathcal{A}) \in \mathcal{R}_{II}(q, \mathbf{d})$ , and then  $W_{\mathbf{d}(k), \mathcal{A}}^{(i)} =$

$$= \begin{cases} Y_{\mathbf{d}(k), \mathcal{A}}^{(i)} & \text{if } i \in \bar{\mathcal{K}}(\mathbf{d}(k)) \\ Y_{\mathbf{d}(k), \mathcal{A}}^{(i)} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(k)} & \text{if } i \in \mathcal{K}(\mathbf{d}(k)), k = u_{\mathbf{d}(k)} \\ Y_{\mathbf{d}(k), \mathcal{A}}^{(i)} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(k)} & \text{if } i = u_{\mathbf{d}(k)}, k \neq u_{\mathbf{d}(k)} \\ Y_{\mathbf{d}(k), \mathcal{A}}^{(i)} \oplus Y_{\mathbf{d}(k), \mathcal{A}}^{(k)} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(k)} & \text{if } i \in \mathcal{K}(\mathbf{d}(k)) \setminus u_{\mathbf{d}(k)}, k \neq u_{\mathbf{d}(k)} \end{cases} \quad (13)$$

for all 2-tuples  $(i, \mathcal{A})$  such that  $i \neq k$  and  $(i, \mathbf{d}(k), \mathcal{A}) \in \mathcal{R}_{II}(q, \mathbf{d})$ .

To show this result, observe that for any 2-tuple  $(f, \mathcal{A})$  such that  $(k, f, \mathcal{A}) \in \mathcal{R}_{II}(q, \mathbf{d})$  and  $f \in \mathcal{A} \setminus \mathbf{d}(k)$ , we have  $\mathcal{A} \setminus \mathbf{d}(k) \subseteq \bar{\mathcal{K}}(\mathbf{d}(k))$  and thus,  $Y_{f, \mathcal{A}}^{(k)} = W_{f, \mathcal{A}}^{(k)}$  from (7). Using the coded cached subfiles,  $Z_{\mathcal{A}}^{(k)}$ , user  $U_k$  can compute

$$\begin{aligned} W_{\mathbf{d}(k), \mathcal{A}}^{(i)} &= Z_{\mathcal{A}}^{(k)} \oplus \bigoplus_{f \in \mathcal{A} \setminus \mathbf{d}(k)} Y_{f, \mathcal{A}}^{(k)} \\ &= \bigoplus_{f \in \mathcal{A}} W_{f, \mathcal{A}}^{(k)} \oplus \bigoplus_{f \in \mathcal{A} \setminus \mathbf{d}(k)} W_{f, \mathcal{A}}^{(k)} \end{aligned} \quad (14)$$

for all  $(i, \mathcal{A})$  such that  $i = k$  and  $(i, \mathbf{d}(k), \mathcal{A}) \in \mathcal{R}_{II}(q, \mathbf{d})$ .

Next, consider the decoding of subfiles  $W_{\mathbf{d}(k), \mathcal{A}}^{(i)}$  in (13) for all  $(i, \mathcal{A})$  such that  $i \neq k$  and  $(i, \mathbf{d}(k), \mathcal{A}) \in \mathcal{R}_{II}(q, \mathbf{d})$ . User  $U_k$  can obtain, directly, from the broadcasted subfiles Type II in (7), the subfiles

$$W_{\mathbf{d}(k), \mathcal{A}}^{(i)} = Y_{\mathbf{d}(k), \mathcal{A}}^{(i)}$$

for all  $i \in \bar{\mathcal{K}}(\mathbf{d}(k))$ . Next, if  $k = u_{\mathbf{d}(k)}$ , using  $W_{\mathbf{d}(k), \mathcal{A}}^{(k)}$  from (14), he obtains

$$\begin{aligned} W_{\mathbf{d}(k), \mathcal{A}}^{(i)} &= Y_{\mathbf{d}(k), \mathcal{A}}^{(i)} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(k)} \\ &= W_{\mathbf{d}(k), \mathcal{A}}^{(i)} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(u_{\mathbf{d}(k)})} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(k)} \\ &= W_{\mathbf{d}(k), \mathcal{A}}^{(i)} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(k)} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(k)} \end{aligned}$$

for all  $i \in \mathcal{K}(\mathbf{d}(k))$ . Instead, if  $k \neq u_{\mathbf{d}(k)}$ , then he first computes

$$\begin{aligned} W_{\mathbf{d}(k), \mathcal{A}}^{(i)} &= W_{\mathbf{d}(k), \mathcal{A}}^{(k)} \oplus Y_{\mathbf{d}(k), \mathcal{A}}^{(k)} \\ &= W_{\mathbf{d}(k), \mathcal{A}}^{(k)} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(k)} \oplus W_{\mathbf{d}(k), \mathcal{A}}^{(u_{\mathbf{d}(k)})} \end{aligned}$$

for  $i = u_{\mathbf{d}(k)}$ , and for all 2-tuples  $(i, \mathcal{A})$  such that  $i \neq k$  and  $(i, \mathbf{d}(k), \mathcal{A}) \in \mathcal{R}_{II}(q, \mathbf{d})$ .

$$W_{\mathbf{d}(k), \mathcal{A}}^{(i)} = W_{\mathbf{d}(k), \mathcal{A}}^{(u_{\mathbf{d}(k)})} \oplus Y_{\mathbf{d}(k), \mathcal{A}}^{(i)}$$

for all  $i \in \mathcal{K}(\mathbf{d}(k)) \setminus u_{\mathbf{d}(k)}$ .

Next, we count the number of broadcasted subfile Type II required. Observe that, there is a broadcasted subfile Type II,  $Y_{f, \mathcal{A}}^{(i)}$ , for each subfile Type II except for  $i = u_f$ . The total number of subfiles Type II is

$$\sum_{i \in \mathcal{K}} \sum_{\mathcal{A}: \mathbf{d}(i) \in \mathcal{A}} \sum_{f \in \mathcal{A}} 1 = K \binom{N_e(\mathbf{d})-1}{q-1} q \quad (15)$$

here, although not explicitly written for brevity, we require  $|\mathcal{A}| = q$  and  $\mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d})$ . The result in (15) follow since, from right to left, first, for any set  $\mathcal{A}$  with  $|\mathcal{A}| = q$ , we

$$\begin{aligned}
\mathcal{R}_{III}(q, \mathbf{d}) &= \{(i, f, \mathcal{A}) \in \mathcal{R}(q, \mathbf{d}) : \mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d}), \mathbf{d}(i) \notin \mathcal{A}\} \\
&= \{(i, f, \mathcal{A}) : |\mathcal{A}| = q, \mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d}), \mathbf{d}(i) \notin \mathcal{A}, i \in \mathcal{K}, f \in \mathcal{A}\} \\
&= \{(i, f, \mathcal{A}) : |\mathcal{A} \cup \mathbf{d}(i)| = q + 1, \mathcal{A} \cup \mathbf{d}(i) \subseteq \mathcal{N}_e(\mathbf{d}), \mathbf{d}(i) \notin \mathcal{A}, i \in \mathcal{K}, f \in \mathcal{A}\} \\
&= \{(i, f, \mathcal{A}) : \mathcal{B} = \mathcal{A} \cup \mathbf{d}(i), |\mathcal{B}| = q + 1, \mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d}), i \in \mathcal{K}, \mathbf{d}(i) \notin \mathcal{A}, f \in \mathcal{B} \setminus \mathbf{d}(i)\} \\
&= \{(i, f, \mathcal{A}) : \mathcal{A} = \mathcal{B} \setminus \mathbf{d}(i), \mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d}), |\mathcal{B}| = q + 1, i \in \mathcal{K}, \mathbf{d}(i) \in \mathcal{B}, f \in \mathcal{B} \setminus \mathbf{d}(i)\} \\
&= \{(i, f, \mathcal{A}) : \mathcal{A} = \mathcal{B} \setminus \mathbf{d}(i), \mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d}), |\mathcal{B}| = q + 1, i \in \mathcal{K}, \mathbf{d}(i) \in \mathcal{B} \setminus f, f \in \mathcal{B}\} \\
&= \{(i, f, \mathcal{A}) : \mathcal{A} = \mathcal{B} \setminus \mathbf{d}(i), (i, f, \mathcal{B}) \in \mathcal{R}'_{III}(q, \mathbf{d})\}.
\end{aligned} \tag{8}$$

$$\mathcal{R}'_{III}(q, \mathbf{d}) = \{(i, f, \mathcal{B}) : \mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d}), |\mathcal{B}| = q + 1, i \in \mathcal{K}, \mathbf{d}(i) \in \mathcal{B} \setminus f, f \in \mathcal{B}\} \tag{11}$$

have  $\sum_{f \in \mathcal{A}} 1 = q$ , second, the number of sets  $\mathcal{A}$  satisfying  $\mathcal{A} \subseteq \mathcal{N}_e(\mathbf{d})$  and  $|\mathcal{A}| = q$  that include a particular file  $\mathbf{d}(i)$  is  $\sum_{\mathcal{A}: \mathbf{d}(i) \in \mathcal{A}} 1 = \binom{N_e(\mathbf{d}) - 1}{q - 1}$ , and, third, there are  $K$  users. Similarly, we can compute the total number of subfiles Type II with no broadcasted subfile associated to, as

$$\sum_{u \in \mathcal{U}} \sum_{\mathcal{A}: \mathbf{d}(u) \in \mathcal{A}} \sum_{f = \mathbf{d}(u)} 1 = N_e(\mathbf{d}) \binom{N_e(\mathbf{d}) - 1}{q - 1} \tag{16}$$

where the set  $\mathcal{U}$  contains all user leaders and thus  $|\mathcal{U}| = N_e(\mathbf{d})$ . Finally, subtracting (16) to (15), we have that the total number of broadcasted subfiles Type II is

$$T_{II} = (Kq - N_e(\mathbf{d})) \binom{N_e(\mathbf{d}) - 1}{q - 1}.$$

Coded cached subfiles Type III: Finally, we consider the delivery of the requested subfiles Type III,  $W_{f, \mathcal{A}}^{(i)}$  for all  $(i, f, \mathcal{A}) \in \mathcal{R}_{III}(q, \mathbf{d})$ . Given that for these subfiles  $\mathbf{d}(i) \notin \mathcal{A}$ , we can rewrite subfiles Type III, equivalently, as  $W_{f, \mathcal{B} \setminus \mathbf{d}(i)}^{(i)}$  for all  $(i, f, \mathcal{B}) \in \mathcal{R}'_{III}(q, \mathbf{d})$  with  $\mathcal{R}'_{III}(q, \mathbf{d})$  given in (11). We show this equivalence in (8)-(10). Equality (9) follows since  $\mathcal{A} = \mathcal{B} \setminus \mathbf{d}(i)$ , and  $\mathbf{d}(i) \in \mathcal{B}$  imply  $\mathbf{d}(i) \notin \mathcal{A}$  and  $\mathcal{B} = \mathcal{A} \cup \mathbf{d}(i)$ , and vice-versa.

First, for each file  $f \in \mathcal{B}$ , we select a *user leader*  $u_f \in \mathcal{K}(f)$ . Next, for each set  $\mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d})$  with  $|\mathcal{B}| = q + 1$ , we define an arbitrary one to one mapping function  $g_{\mathcal{B}}(f)$  which for each file index  $f \in \mathcal{B}$  returns a file index  $g_{\mathcal{B}}(f) \in \mathcal{B} \setminus f$  satisfying  $g_{\mathcal{B}}(f_1) \neq g_{\mathcal{B}}(f_2)$  if  $f_1 \neq f_2$ . e.g. if  $\mathcal{B} = \{0, 1, \dots, q\}$ , then we can use  $g_{\mathcal{B}}(f) = (f - 1) \bmod (q + 1)$ . Then, the server first broadcasts

$$Y_{f, \mathcal{B}}^{(i)} = W_{f, \mathcal{B} \setminus g_{\mathcal{B}}(f)}^{(u_{g_{\mathcal{B}}(f)})} \oplus W_{f, \mathcal{B} \setminus \mathbf{d}(i)}^{(i)} \tag{17}$$

for each 3-tuple  $(i, f, \mathcal{B}) \in \mathcal{R}_{III}(q, \mathbf{d})$  with  $i \neq u_{g_{\mathcal{B}}(f)}$ , and then

$$Y_{\mathcal{B}} = \bigoplus_{f \in \mathcal{B}} W_{f, \mathcal{B} \setminus g_{\mathcal{B}}(f)}^{(u_{g_{\mathcal{B}}(f)})} \tag{18}$$

for each set  $\mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d})$  with  $|\mathcal{B}| = q + 1$ . Although not broadcasted, let us set  $Y_{f, \mathcal{B}}^{(i)} = \mathbf{0}$  for  $i = u_{g_{\mathcal{B}}(f)}$ .

Next we detail the decoding operations at user  $U_k$  to obtain the requested subfiles Type III,  $W_{\mathbf{d}(k), \mathcal{B} \setminus \mathbf{d}(i)}^{(i)}$  for all  $(i, \mathcal{B})$  such

that  $(i, \mathbf{d}(k), \mathcal{B}) \in \mathcal{R}'_{III}(q, \mathbf{d})$ . The decoding process begins by computing

$$C_{\mathbf{d}(k), \mathcal{B}}^{(k)} = Z_{\mathcal{B} \setminus \mathbf{d}(k)}^{(k)} \oplus \bigoplus_{f \in \mathcal{B} \setminus \mathbf{d}(k)} Y_{f, \mathcal{B}}^{(k)}$$

for all  $\mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d})$  with  $|\mathcal{B}| = q + 1$ , and  $\mathbf{d}(k) \in \mathcal{B}$ , and then

$$W_{\mathbf{d}(k), \mathcal{B} \setminus \mathbf{d}(i)}^{(i)} = \begin{cases} C_{\mathbf{d}(k), \mathcal{B}}^{(k)} \oplus Y_{\mathcal{B}} & \text{if } i = u_{g_{\mathcal{B}}(\mathbf{d}(k))} \\ C_{\mathbf{d}(k), \mathcal{B}}^{(k)} \oplus Y_{\mathcal{B}} \oplus Y_{\mathbf{d}(k), \mathcal{B}}^{(i)} & \text{if } i \neq u_{g_{\mathcal{B}}(\mathbf{d}(k))}, \end{cases} \tag{19}$$

for all  $(i, \mathcal{B})$  such that  $(i, \mathbf{d}(k), \mathcal{B}) \in \mathcal{R}'_{III}(q, \mathbf{d})$ .

To show this result, observe that

$$\begin{aligned}
C_{\mathbf{d}(k), \mathcal{B}} &= Z_{\mathcal{B} \setminus \mathbf{d}(k)}^{(k)} \oplus \bigoplus_{f \in \mathcal{B} \setminus \mathbf{d}(k)} Y_{f, \mathcal{B}}^{(k)} \\
&= \bigoplus_{f \in \mathcal{B} \setminus \mathbf{d}(k)} W_{f, \mathcal{B} \setminus \mathbf{d}(k)}^{(k)} \oplus \bigoplus_{f \in \mathcal{B} \setminus \mathbf{d}(k)} W_{f, \mathcal{B} \setminus g_{\mathcal{B}}(f)}^{(u_{g_{\mathcal{B}}(f)})} \oplus W_{f, \mathcal{B} \setminus \mathbf{d}(k)}^{(k)} \\
&= \bigoplus_{f \in \mathcal{B} \setminus \mathbf{d}(k)} W_{f, \mathcal{B} \setminus g_{\mathcal{B}}(f)}^{(u_{g_{\mathcal{B}}(f)})}
\end{aligned}$$

and, that XORing  $C_{\mathbf{d}(k), \mathcal{B}}$  and  $Y_{\mathcal{B}}$ , we can obtain

$$\begin{aligned}
W_{\mathbf{d}(k), \mathcal{B} \setminus \mathbf{d}(i)}^{(i)} &= C_{\mathbf{d}(k), \mathcal{B}} \oplus Y_{\mathcal{B}} \\
&= W_{\mathbf{d}(k), \mathcal{B} \setminus g_{\mathcal{B}}(\mathbf{d}(k))}^{(u_{g_{\mathcal{B}}(\mathbf{d}(k))})}
\end{aligned}$$

for all  $(i, \mathcal{B})$  such that  $i = u_{g_{\mathcal{B}}(\mathbf{d}(k))}$  and  $(i, \mathbf{d}(k), \mathcal{B}) \in \mathcal{R}'_{III}(q, \mathbf{d})$ . Then, he can obtain the remaining subfiles, as

$$W_{\mathbf{d}(k), \mathcal{B} \setminus \mathbf{d}(i)}^{(i)} = W_{\mathbf{d}(k), \mathcal{B} \setminus g_{\mathcal{B}}(\mathbf{d}(k))}^{(u_{g_{\mathcal{B}}(\mathbf{d}(k))})} \oplus Y_{\mathbf{d}(k), \mathcal{B}}^{(i)}$$

for all  $(i, \mathcal{B})$  such that  $i \neq u_{g_{\mathcal{B}}(\mathbf{d}(k))}$  and  $(i, \mathbf{d}(k), \mathcal{B}) \in \mathcal{R}'_{III}(q, \mathbf{d})$ .

Next, we compute the total number of broadcasted subfiles Type III. For brevity, although not explicitly written, the summations over  $\mathcal{B}$  require  $\mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d})$  and  $|\mathcal{B}| = q + 1$ . Observe that there is one broadcasted subfile  $Y_{\mathcal{B}}$  for each set  $\mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d})$  with  $|\mathcal{B}| = q + 1$ , and one broadcasted subfile  $Y_{f, \mathcal{B}}^{(i)}$  for all  $(i, f, \mathcal{B}) \in \mathcal{R}_{III}(q, \mathbf{d})$  with  $i \neq u_{g_{\mathcal{B}}(f)}$ , or more explicitly for all  $(i, f, \mathcal{B}) \in \{\mathcal{B} \subseteq \mathcal{N}_e(\mathbf{d}), |\mathcal{B}| = q + 1, i \in \mathcal{K} \setminus u_{g_{\mathcal{B}}(f)}, \mathbf{d}(i) \in \mathcal{B} \setminus f, f \in \mathcal{B}\}$ .

Thus, the total number of broadcasted subfiles Type III is  $T_{III}$

$$\begin{aligned}
&= \sum_{\mathcal{B}} \left( \sum_{f \in \mathcal{B}} \left( \sum_{d \in \mathcal{B} \setminus f} \sum_{i \in \mathcal{K}(d) \setminus u_{g_{\mathcal{B}}}(f)} 1 \right) + 1 \right) \\
&= \sum_{\mathcal{B}} \left( \sum_{f \in \mathcal{B}} \left( \sum_{d \in \mathcal{B} \setminus f} \mathcal{K}(d) - 1 \right) + 1 \right) \quad (20) \\
&= \sum_{\mathcal{B}} \sum_{f \in \mathcal{B}} \sum_{d \in \mathcal{B} \setminus f} \mathcal{K}(d) - \sum_{\mathcal{B}} \sum_{f \in \mathcal{B}} 1 + \sum_{\mathcal{B}} 1 \\
&= \sum_{\mathcal{B}} \sum_{f \in \mathcal{B}} \sum_{d \in \mathcal{B}} \mathcal{K}(d) - \sum_{\mathcal{B}} \sum_{f \in \mathcal{B}} (\mathcal{K}(f) + 1) + \sum_{\mathcal{B}} 1 \\
&= (qK - N_e(\mathbf{d})) \binom{N_e(\mathbf{d}) - 1}{q} + \binom{N_e(\mathbf{d})}{q+1} \quad (21) \\
&= \left( qK - \left(1 - \frac{1}{q}\right) N_e(\mathbf{d}) \right) \binom{N_e(\mathbf{d}) - 1}{q} \quad (22)
\end{aligned}$$

where (20) follow since for each  $f$ ,  $u_{g_{\mathcal{B}}}(f)$  is only found in  $\mathcal{K}(d)$  for  $d = g_{\mathcal{B}}(f)$ , and (21) follows since

$$\sum_{\mathcal{B}} 1 = \binom{N_e(\mathbf{d})}{q+1},$$

$$\begin{aligned}
\sum_{\mathcal{B}} \sum_{f \in \mathcal{B}} (\mathcal{K}(f) + 1) &= \sum_{f \in \mathcal{N}_e(\mathbf{d})} (\mathcal{K}(f) + 1) \sum_{\mathcal{B}: f \in \mathcal{B}} 1 \\
&= (K + N_e(\mathbf{d})) \binom{N_e(\mathbf{d}) - 1}{q},
\end{aligned}$$

$$\begin{aligned}
\sum_{\mathcal{B}} \sum_{f \in \mathcal{B}} \sum_{d \in \mathcal{B}} \mathcal{K}(d) &= (q+1) \sum_{\mathcal{B}} \sum_{d \in \mathcal{B}} \mathcal{K}(d) \\
&= (q+1) \sum_{d \in \mathcal{N}_e(\mathbf{d})} \mathcal{K}(d) \sum_{\mathcal{B}: d \in \mathcal{B}} 1 \\
&= (q+1)K \binom{N_e(\mathbf{d}) - 1}{q}.
\end{aligned}$$

Next, (22) follow since  $\binom{n}{k} = \frac{n}{k} \binom{n-1}{k-1}$ . Finally, adding together  $T_I$ ,  $T_{II}$ , and  $T_{III}$ , we obtain

$$\begin{aligned}
T &= KN_e(\mathbf{d}) \binom{N-1}{q-1} \\
&\quad - N_e(\mathbf{d}) \binom{N_e(\mathbf{d})+1}{q+1} \binom{N_e(\mathbf{d})-1}{q-1}
\end{aligned}$$

which leads to the rate (4) stated in Theorem 1.

## V. CONCLUSIONS

In this work, we proposed a novel centralized coded caching scheme for the case where there are more users than files  $K \geq N$  and users are equipped with small memories  $M \leq \frac{N}{K}$ . The scheme uses coded prefetching and outperforms previously proposed schemes for moderate-high number of users,  $N \leq K \leq \frac{N^2+1}{2}$ . Due to the limited region of applicability, the practical interest of this scheme might be small. However, the ideas and strategy here presented may motivate further

developments in the coded caching problem. Our current and future work is in this direction, and includes the extension of the proposed coded prefetching technique to larger memories and scenarios with more users than files

## REFERENCES

- [1] M. R. Korupolu, C. Plaxton, and R. Rajaraman, "Placement algorithms for hierarchical cooperative caching," *Journal of Algorithms*, vol. 38, no. 1, pp. 260 – 302, 2001.
- [2] A. Dan, D. Sitaram, and P. Shahabuddin, "Dynamic batching policies for an on-demand video server," *Multimedia Systems*, vol. 4, no. 3, pp. 112–121, 1996.
- [3] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.
- [4] —, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Transactions on Networking*, vol. 23, no. 4, pp. 1029–1040, Aug 2015.
- [5] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," *IEEE Transactions on Information Theory*, vol. PP, no. 99, pp. 1–1, 2016.
- [6] J. Zhang, X. Lin, and X. Wang, "Coded caching under arbitrary popularity distributions," in *ITA workshop*, Feb 2015, pp. 98–107.
- [7] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, "Order-optimal rate of caching and coded multicasting with random demands," *IEEE Transactions on Information Theory*, vol. 63, no. 6, pp. 3923–3949, June 2017.
- [8] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 836–845, April 2016.
- [9] N. Karamchandani, U. Niesen, M. A. Maddah-Ali, and S. N. Diggavi, "Hierarchical coded caching," *IEEE Transactions on Information Theory*, vol. 62, no. 6, pp. 3212–3229, June 2016.
- [10] L. Zhang, Z. Wang, M. Xiao, G. Wu, and S. Li, "Centralized caching in two-layer networks: Algorithms and limits," in *IEEE WiMob*, Oct 2016, pp. 1–5.
- [11] M. M. Amiri, Q. Yang, and D. Gündüz, "Coded caching for a large number of users," in *2016 IEEE Information Theory Workshop (ITW)*, Sept 2016, pp. 171–175.
- [12] S. Sahraei and M. Gastpar, "K users caching two files: An improved achievable rate," in *2016 Annual Conference on Information Science and Systems (CISS)*, March 2016, pp. 620–624.
- [13] H. Ghasemi and A. Ramamoorthy, "Improved lower bounds for coded caching," in *IEEE ISIT*, June 2015, pp. 1696–1700.
- [14] M. M. Amiri and D. Gunduz, "Fundamental limits of coded caching: Improved delivery rate-cache capacity trade-off," *IEEE Transactions on Communications*, vol. PP, no. 99, pp. 1–1, 2016.
- [15] Z. Chen, P. Fan, and K. B. Letaief, "Fundamental limits of caching: improved bounds for users with small buffers," *IET Communications*, vol. 10, no. 17, pp. 2315–2318, 2016.
- [16] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "The exact rate-memory tradeoff for caching with uncoded prefetching," in *IEEE ISIT*, June 2017, pp. 1613–1617.
- [17] C. Tian and J. Chen, "Caching and Delivery via Interference Elimination," *ArXiv e-prints*, Apr. 2016.
- [18] C. Tian and K. Zhang, "From uncoded prefetching to coded prefetching in coded caching," *CoRR*, vol. abs/1704.07901, 2017. [Online]. Available: <http://arxiv.org/abs/1704.07901>
- [19] K. Wan, D. Tuninetti, and P. Piantanida, "On caching with more users than files," in *IEEE ISIT*, July 2016, pp. 135–139.
- [20] A. Sengupta, R. Tandon, and T. C. Clancy, "Improved approximation of storage-rate tradeoff for caching via new outer bounds," in *IEEE ISIT*, June 2015, pp. 1691–1695.
- [21] C. Tian, "A note on the fundamental limits of coded caching," *CoRR*, vol. abs/1503.00010, 2015.
- [22] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Characterizing the rate-memory tradeoff in cache networks within a factor of 2," in *IEEE ISIT*, June 2017, pp. 386–390.
- [23] C. Wang, S. S. Bidokhti, and M. A. Wigger, "Improved converses and gap-results for coded caching," *CoRR*, vol. abs/1702.04834, 2017. [Online]. Available: <http://arxiv.org/abs/1702.04834>
- [24] J. Gómez-Vilardebó, "Fundamental Limits of Caching: Improved Bounds with Coded Prefetching," *ArXiv e-prints*, Dec. 2016.