Coded Caching for a Large Number Of Users

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

***Abstract*—We consider the coded caching problem with a central server containing N files, each of length F bits, and K users, each equipped with a cache of capacity MF bits. We assume that coded contents can be proactively placed into users' caches at no cost during the *placement phase*. During the *delivery phase*, each user requests exactly one file from the database, and all the requests are served simultaneously by the server over an error-free common link. The goal is to utilize the local cache memories at the users to reduce the delivery rate from the server during the peak period. Here, we focus on a system which has more users than files, i.e., $K > N$. We first consider the *centralized* caching problem, in which the number and identity of active users are known in advance, and propose a group-based coded caching scheme for $M = N/K$, which improves upon the best achievable scheme in the literature. The proposed centralized caching scheme is then exploited in a *decentralized* setting, in which neither the number nor the identity of the active users are known during the placement phase. It is shown that the proposed coded caching scheme improves upon the best known decentralized delivery rate as well.**

***Index Terms*—Network coding, centralized coded caching, decentralized coded caching.**

I. Introduction

Content caching at the network edge has been recognized as a promising approach to tackle the continuous growth of network data traffic by bringing popular content closer to end users. Going one step further, contents can be cached directly into user terminals even before they are requested, called *proactive caching*. Here, we consider the coded caching model introduced in [1], which considers proactively caching contents at user terminals at no cost over the off-peak traffic periods. This initial *placement phase* is carried out without the knowledge of the user demands, which are assumed to be revealed during the peak traffic period. The server satisfies all the demands simultaneously over an error-free shared link during the *delivery phase*. For given number of users in the system, K, number of popular files in the database, N, and normalized cache capacities at the users, M, the goal is to achieve the minimum delivery rate for any demand combination.

Conventional caching schemes focus on the placement phase, the gain of which mainly derives from predicting popular files and making them available locally. In contrast, in "*coded caching*", proposed in [1] and [2], further gains can be obtained by jointly designing the placement and delivery phases to create multicasting opportunities even among users with different requests.

In *centralized* coded caching [1], there is a central server that knows the number of users and their identities during the placement phase; and therefore, cache contents can be designed jointly in advance to maximize the multicasting opportunities. However, in wireless networks, users are mobile, and the placement phase is typically carried out over different networks and time frames. Therefore, it is not possible to know in advance which users will participate in the delivery phase, and to coordinate the placement phases for these users. In *decentralized* coded caching [2], it is assumed that the server has no prior knowledge on the number of users or their identities during the placement phase.

Following the seminal papers of Maddah-Ali and Niesen [1] and [2], significant research efforts have been invested into improving the gains from coded caching in both centralized and decentralized settings. Assuming a symmetric cache capacity of MF bits at each user, where F is the length of each file, the authors in [3] introduced an optimal centralized coded caching scheme for $M \leq 1/K$ when $N \leq K$, which achieves the theoretical lower bound on the delivery rate. Another centralized caching scheme is introduced in [4] for $N < K$, which improves upon the scheme achieved through memory-sharing between [1] and [3] in certain scenarios. The centralized caching scheme proposed in [5] provides a reduced delivery rate for $M = (N-1)/K$, when $4 \leq N < K \leq 3N/2$, and K and N have a common divisor greater than 1, and it is shown to improve the delivery rate compared to other schemes. While the cut-set bound provides a lower bound on the delivery rate, a tighter lower bound is obtained in [6]. Further variations of this caching model have also been studied in the literature, such as caching with non-uniform demand distributions [7], [8], and online coded caching [9].

In this paper, we introduce a novel coded caching scheme when the number of users is more than the number of files, i.e., $K > N$, and the cache capacity of the users is exactly sufficient to cache all the files in the system, i.e., $M = N/K$. The proposed caching scheme achieves a delivery rate below the state-of-the-art (combination of [1] and [3] by memory-sharing) for $K > N \geq 3$. This improvement can be extended to other cache capacities through memory-sharing arguments. We then apply the proposed caching scheme in the decentralized scenario, and show that it achieves the best known decentralized delivery rate in the literature as well.

We organize the rest of the paper as follows. Section II presents the system model and the relevant literature. We introduce our centralized coded caching scheme in Section III, and its extension to the decentralized scenario in Section IV.

Simulation results are presented in Section V. Finally, Section VI concludes the paper.

II. System Model and Previous Results

We consider a wireless caching system, in which a server has N popular contents $W_1, W_2, ..., W_N$, each of size F bits. The contents are delivered by the server to K users $U_1, U_2, ..., U_K$, through an error-free shared link. Each user is equipped with a cache of capacity MF bits.

The operation of the system is divided into two phases. In the *placement phase*, caches are filled without the knowledge of the particular user requests. We denote the contents of user U_k's cache at the end of the placement phase by Z_k, for $k = 1, \ldots, K$. User requests are revealed after the placement phase. Each user U_k requests a single file from the database, denoted by d_k. We have $d_k \in \{1, ..., N\}$. In the *delivery phase*, a common message X is transmitted from the server to all the users over an error-free shared link to satisfy all the demands $(d_1, d_2, ..., d_K)$, together with the cache contents available locally to the users.

The *centralized* and *decentralized* models differ in their placement phases. In centralized caching, a certain set of users participate in both phases; and therefore, the cache contents $Z_1, \ldots, Z_K$ can be designed jointly, and can depend on K. In decentralized caching, neither the number, nor the identities of the users, which will participate in the delivery phase, is known during the placement phase; and thus, cache contents cannot be designed jointly.

For a given cache capacity M, we denote by $R_{(d_1,d_2,...,d_K)}(M)$ the normalized size (by F) of the common message sent over the shared link during the delivery phase, i.e., a total of $R_{(d_1,d_2,...,d_K)}(M)F$ bits are transmitted. We say that the rate $R_{(d_1,d_2,...,d_K)}(M)$ is *achievable*, if for F large enough, each user U_k can decode the requested content with probability arbitrarily close to 1 using the signal transmitted in the delivery phase, X, together with the local cache content, Z_k, for $k = 1, \ldots, K$. For a given cache capacity M, the *achievable* delivery rate is defined for the worst case user demands as follows:

$$R(M) \triangleq \max_{d_1,...,d_K} R_{(d_1,d_2,...,d_K)}(M). \tag{1}$$

We aim to characterize the best $R(M)$, i.e., the minimal delivery rate for $0 \leq M \leq N$.

For $N > K$ and $0 \leq M \leq N$, the minimum delivery rate in the literature is achieved by the scheme proposed in [1], which will be called as the MAN scheme. When $N \leq K$, the scheme presented in [3], which will be called as the CFL scheme, achieves the best delivery rate for $M \leq 1/K$. For $N \leq K$ and $M > 1/K$, memory sharing between the MAN and CFL schemes achieves the best delivery rate. To characterize the best achievable rate in this regime, we define, for $t = 1, \ldots, K$,

$$f(N, K, t) \triangleq \frac{(N-1)(K-t)}{(t+1)(tN-1)} + N^2 \left(1 - \frac{1}{K}\right) \left(\frac{t-1}{tN-1}\right), \tag{2}$$

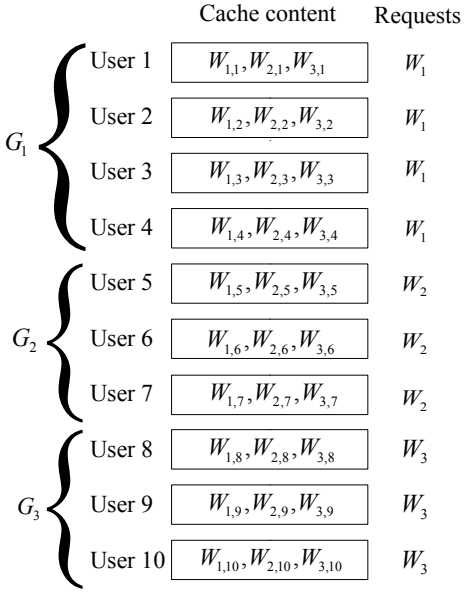

Fig. 1. Illustration of the cache placement phase for the proposed GBC scheme for $K = 10$ users and $N = 3$ files, and a worst-case user request combination. Users are grouped based on their requests in the delivery phase.

and

$$t^* \triangleq \arg\min_{t \in \{1,...,K\}} f(N, K, t). \tag{3}$$

Let $R_b(M)$ denote the best achievable delivery rate-cache capacity trade-off in the literature. For $M \geq \hat{t}N/K$, where $\hat{t} \triangleq \max\{t^*, 2\}$, $R_b(M)$ is achieved by the MAN scheme. For $1/K \leq M \leq \hat{t}N/K$, $R_b(M)$ is achieved by memory-sharing between the MAN and CFL schemes. For $M = N/K$, we have

$$R_b(N/K) = \min_{t \in \{1,...,K\}} \{f(N, K, t)\}. \tag{4}$$

III. Centralized Coded Caching

In this section, we introduce our novel centralized coded caching scheme, analyze its delivery rate, and compare it with $R_b(M)$. For clarity, we first illustrate this scheme on an example with $N = 3$ files and $K = 10$ users.

Example 1. Consider $N = 3$ files W_1, W_2, and W_3 in the database, and $K = 10$ users, each equipped with a cache of capacity $M = 3/10$. Similarly to [1], each file W_i is partitioned into 10 non-overlapping subfiles $W_{i,j}$ with equal size of $F/10$ bits, and $Z_j = (W_{1,j}, W_{2,j}, W_{3,j})$ is cached at user j during the placement phase, for $i = 1, 2, 3$ and $j = 1, \ldots, 10$ as depicted in Fig. 1.

Worst case user demands happens when the files requested by the users are as distinct as possible. Without loss of generality, by re-ordering the users, we can focus on the following worst-case user demands:

$$d_j = \begin{cases} 1, & 1 \leq j \leq 4, \\ 2, & 5 \leq j \leq 7, \\ 3, & 8 \leq j \leq 10. \end{cases} \tag{5}$$

The delivery phase consists of two distinct parts; that is, the common message can be written as $X = (X_1, X_2)$, where X_i corresponds to the message delivered in part i, for $i \in \{1, 2\}$. We have:

Part 1: $X_1 = (W_{1,1} \oplus W_{1,2}\,,\, W_{1,2} \oplus W_{1,3},\, W_{1,3} \oplus W_{1,4},\, W_{2,5} \oplus W_{2,6},\, W_{2,6} \oplus W_{2,7},\, W_{3,8} \oplus W_{3,9},\, W_{3,9} \oplus W_{3,10})$,

Part 2: $X_2 = (W_{1,5} \oplus W_{1,6}\,,\, W_{1,6} \oplus W_{1,7},\, W_{2,1} \oplus W_{2,2},\, W_{2,2} \oplus W_{2,3},\, W_{2,3} \oplus W_{2,4},\, W_{1,7} \oplus W_{2,4},\, W_{1,8} \oplus W_{1,9},\, W_{1,9} \oplus W_{1,10},\, W_{3,1} \oplus W_{3,2},\, W_{3,2} \oplus W_{3,3},\, W_{3,3} \oplus W_{3,4},\, W_{1,10} \oplus W_{3,4},\, W_{2,8} \oplus W_{2,9},\, W_{2,9} \oplus W_{2,10},\, W_{3,5} \oplus W_{3,6},\, W_{3,6} \oplus W_{3,7},\, W_{3,7} \oplus W_{2,10})$,

where $\oplus$ denotes the bitwise XOR operation. Our proposed scheme groups users according to their demands in the delivery phase, and part 1 is designed so that each user in each group decodes all the subfiles of the requested content of that group that are available in the caches of the other users in the same group. For example, users $U_1, \ldots, U_4$ all have $W_{1,1}, W_{1,2}, W_{1,3}, W_{1,4}$ by receiving part 1. Part 2 is designed so that users decode the parts of their requests that are located in the caches of users from other groups. It is easy to verify for the above example that, together with the coded contents of the local cache memory, Z_k, each user U_k can decode its desired file W_{d_k} after receiving the coded bits, X_1 and X_2. Therefore, a total of $12F/5$ bits are served over the shared link in the delivery phase, corresponding to an achievable delivery rate of $R(3/10) = 2.4$, which is less than $R_b(3/10) = 2.43$, the best achievable delivery rate in the literature for this setting, given by (4). □

A. Group-Based Coded Caching

Next, we present the placement and delivery phases of our group-based centralized coded caching scheme, denoted as the GBC scheme, for the general case with $N < K$, and a cache capacity of $M = N/K$. Since the server does not know the requests of the users in advance, we design the placement phase in a symmetric manner in order to be able to serve all user demands in the most efficient manner in the delivery phase. As shown in Example 1, for $M = N/K$, each file W_i is split into K non-overlapping subfiles $W_{i,k}$, each of size F/K bits, and $Z_k \triangleq (W_{1,k}, W_{2,k}, ..., W_{N,k})$ is placed at user U_k's cache, for $i \in \{1, ..., N\}$ and $k \in \{1, ..., K\}$ (refer to Fig. 1 as an example of the placement phase). Therefore, each subfile $W_{i,k}$ of file W_i is cached only at user U_k's cache.

Since $N < K$, there is at least one file requested by more than one user. Without loss of generality, by re-labeling the files and re-ordering the users, we can assume that the first K_1 users, referred to as group G_1, have the same request W_1, the next K_2 users form the group G_2 and demand file W_2, and so on so forth. Hence, we have

$$d_j = i, \quad i \in \{1, ..., N\}\,,\; j \in \left\{S_1^{i-1} + 1, ..., S_1^i\right\}, \quad (6)$$

where $S_i^j \triangleq \sum_{l=i}^{j} K_l$. An example of grouping users can be seen in Fig. 1, in which $N = 3$, $K_1 = 4$, and $K_2 = K_3 = 3$. We can argue that the worst-case of user demands happens when each file in the database is requested by at least one user, i.e., $K_l > 0$, $\forall l \in \{1, ..., N\}$. We will see that the delivery rate of our caching scheme is independent of the values of K_i, for $i = 1, \ldots, N$.

The delivery phase of the GBC scheme, given in Algorithm 1, consists of two parts, where X_i represents the bits delivered in part i, for $i = 1, 2$. As in Example 1, with the first part, X_1, each user in group G_i can recover the subfiles of its requested file which are in the cache of other users in the same group, i.e., the users that have the same request. This is achieved by delivering $(K_i - 1)$ XOR-ed contents, for $i = 1, ..., N$. With the second part of coded delivery, X_2, each user in group G_i can obtain the subfiles of its desired file placed in the cache of users in group G_j, for $i, j \in \{1, ..., N\}$ and $i \neq j$. Therefore, at the end of the delivery phase, each user can decode the subfiles of its requested file placed in the caches of all the other users. Having access to one subfile of its demand locally, each user can thus recover its desired file.

Algorithm 1 Coded Delivery Algorithm

Part 1: Exchanging contents between users in the same group
for $i = 1, \ldots, N$ **do**
  $X_1 \leftarrow \left(X_1, \bigcup_{k=S_1^{i-1}+1}^{S_1^i - 1} (W_{i,k} \oplus W_{i,k+1})\right)$.
end for

Part 2: Exchanging contents between users in different groups
for $i = 1, \ldots, N-1$ **do**
  for $j = i+1, \ldots, N$ **do**
    $X_2 \leftarrow \left(X_2, \bigcup_{k=S_1^{j-1}+1}^{S_1^j - 1} (W_{i,k} \oplus W_{i,k+1}),\; \bigcup_{k=S_1^{i-1}+1}^{S_1^i - 1} (W_{j,k} \oplus W_{j,k+1}),\; W_{i,S_1^j} \oplus W_{j,S_1^i}\right)$.
  end for
end for

The delivery rate of the proposed centralized coded caching scheme with the worst-case user demands is provided in the following theorem, which is stated here without a proof due to space limitations.

Theorem 1. *When $N < K$, and each user is equipped with a cache of capacity $M = N/K$, the following delivery rate-cache capacity trade-off is achievable with the GBC scheme:*

$$R_{\mathrm{GBC}}\left(\frac{N}{K}\right) = N - \frac{N(N+1)}{2K}. \quad (7)$$

Remark 1. *The achievable delivery rate in* (7) *depends only on N and K, and it does not depend on the values of $K_1, \ldots, K_N$, which implies the popularity of the files. However, the more distinct the requests of the users, the higher the delivery rate. Accordingly, the above delivery rate is obtained under the worst-case user demand assumption.*

$$R_{\mathrm{GBD}}(M)=\left(1-\frac{M}{N}\right)\min\left\{\frac{N}{M}-\left[(K-N-2)\left(1+\frac{1}{2}(K-N-1)\frac{M}{N}\right)+\frac{N}{M}\right]\left(1-\frac{M}{N}\right)^{K-2},N\right\}. \quad (8)$$

Remark 2. *To prove that $R_{\mathrm{GBC}}(N/K)$ is less than $R_b(N/K)$, it suffices to illustrate that $R_{\mathrm{GBC}}(N/K) \leq f(N,K,t)$, $\forall t \in \{1,...,K\}$. The proof of this is given in [10] for $K > N \geq 3$. Thus, for $K > N \geq 3$, the proposed group-based centralized coded caching scheme achieves a delivery rate smaller than the best achievable scheme in the literature. This improvement can be expanded to a larger range of cache capacities through memory-sharing between GBC and the existing caching schemes of MAN and CFL.*

The delivery rate $R_b(N/K)$ given by (4) is achieved by memory-sharing between CFL scheme for $M = 1/K$, and the MAN scheme for $M = t^*N/K$, where t^* is given by (3). Therefore, the delivery rate can be reduced compared to $R_b(M)$ for all cache capacities $1/K < M < \hat{t}N/K$[1].

Corollary 1. *In a centralized caching system, consisting of a server with N popular contents and K users, each equipped with a normalized cache capacity of M, the following delivery rate-cache capacity trade-off is achievable for any cache capacity $1/K \leq M \leq \hat{t}N/K$ by memory-sharing between the proposed GBC scheme and the MAN and CFL schemes:*

$$R_{\mathrm{GBC}}(M) = \begin{cases} N\left(1-\frac{M}{2}-\frac{1}{2K}\right), & \text{if } \frac{1}{K} \leq M \leq \frac{N}{K}, \\ \frac{K-\hat{t}}{\hat{t}^2-1}\left(\frac{KM}{N}-1\right)+\frac{K-N}{\hat{t}-1}\left(\frac{\hat{t}N}{K}-M\right), & \text{if } \frac{N}{K} \leq M \leq \frac{\hat{t}N}{K}, \end{cases} \quad (9)$$

where $\hat{t} \triangleq \max\{2,t^*\}$, and t^* is as defined in (3).

In the next section, we will apply the proposed centralized coded caching scheme to a decentralized setting.

IV. Decentralized Coded Caching

In practice, users may fill their cache memories from different servers at different times, and the number of users to be served during the delivery phase may not be known in advance. Therefore, the coordination across users during the placement phase may not be possible since the identity of active users is unknown. However, coded delivery can still be used to exploit multicasting opportunities.

To overcome the uncertainty of the number and identity of active users during the placement phase, contents are placed randomly and independently in each user's cache. However, to keep the symmetry among the cached contents, same number of randomly chosen bits from each file are cached by each user. Similarly to the placement phase proposed in [2], for a cache capacity of MF bits at each user, MF/N random bits from each file are independently chosen and proactively cached by user k, for $k = 1,\ldots,K$. Since there are a total of N files in the database, each user caches a total of MF bits, exactly filling the local cache memory.

At the beginning of the delivery phase, the number of users and their demands are revealed. The server then sends a common message X to serve all the users. If $N \geq K$, the delivery algorithm consisting of the two procedures proposed in [2, Algorithm 1] is performed. On the other hand, if $N < K$, i.e., when there is at least one file requested by more than one user, we can utilize the coded delivery algorithm presented in Algorithm 1. This can be achieved by revising the [2, Algorithm 1] as follows: for $s = 2$ in line 7 of the first procedure of [2, Algorithm 1], Algorithm 1 in Section III can enable each user U_k to retrieve the pieces of its desired file which are cached exclusively in the cache of user U_l, for $k,l \in \{1,...,K\}$ and $l \neq k$. The delivery phase proposed in [2, Algorithm 1] can be used for other cases. The details of the proposed decentralized coded delivery algorithm, referred to as GBD, can be found in [10, Algorithm 2]. The following corollary, stated here without proof, provides the achievable delivery rate of the GBD scheme.

Corollary 2. *In a decentralized caching system with N files, each of size F bits, and K active users, each with a normalized cache capacity of M, if the number of users is more than the number of files, i.e., $K > N$, then the delivery rate-cache capacity trade-off $R_{\mathrm{GBD}}(M)$, given by* (8) *at the top of this page, is achievable for sufficiently large F, by employing the GBC scheme introduced in Section III-A.*

It is possible to show that, for $K > N$, the GBD scheme achieves a lower decentralized delivery rate than the one proposed in [2], originated from the superiority of our scheme in the centralized caching setting.

V. Simulation Results

In this section, the proposed coded caching schemes are compared with the state-of-the-art in both centralized and decentralized settings. We first consider a centralized caching system with $N = 40$ files and $K = 130$ users, for which we have $t^* = 4$ from (3), and hence, $\hat{t} = 4$. Therefore, the improved delivery rate for $M = N/K$ can be extended to any cache capacities satisfying $1/K < M < 4N/K$. In Fig. 2, the delivery rate for this setting is plotted as a function of the cache capacity. The delivery rate achieved through memory-sharing between MAN and CFL schemes for cache capacities $M = 1/K$ and $M = 4N/K$, respectively, is referred to as MNC in this figure. It can be seen that the GBC scheme requires a smaller delivery rate than MNC for the whole range of cache capacity values, $1/K < M < 4N/K$. Another centralized caching scheme is proposed in [4], denoted here

[1]Note that, when $t^* \geq 2$, the range of cache capacities with improved delivery rate is $1/K < M < t^*N/K$, but when $t^* = 1$, the superiority of our scheme can be extended to the interval $1/K < M < 2N/K$.

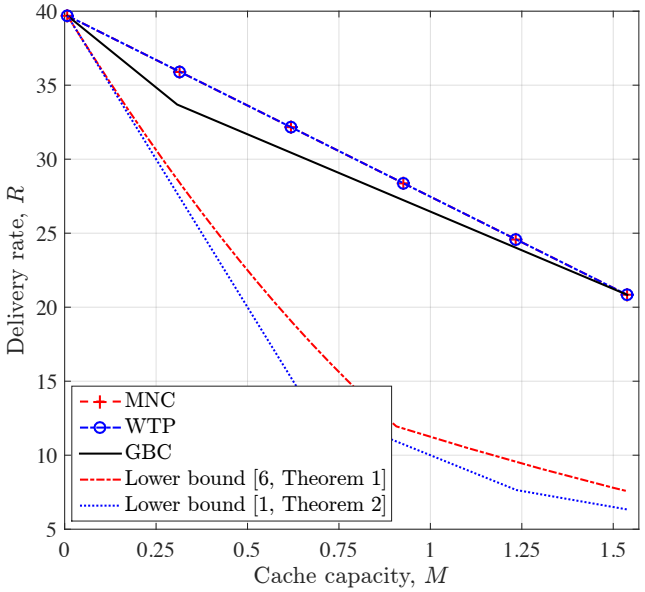

Fig. 2. The delivery rate-cache capacity trade-off for centralized caching with $N = 40$ and $K = 130$, i.e., $\hat{t} = 4$, for $1/K \leq M \leq 4N/K$.

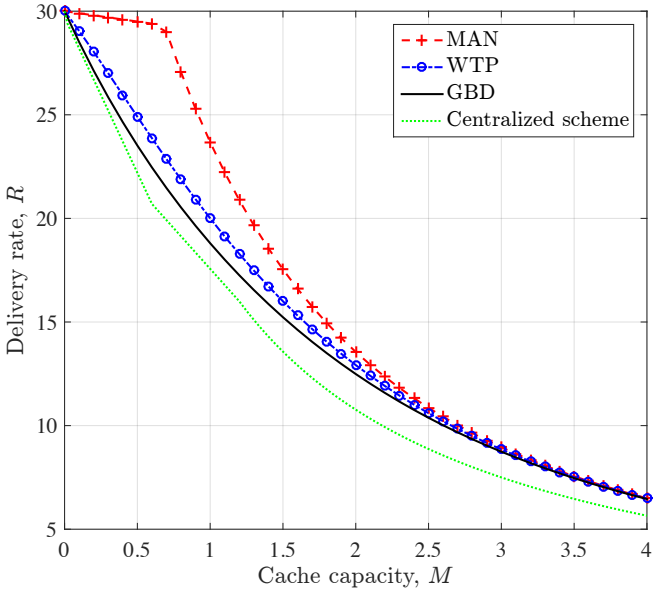

Fig. 3. The delivery rate-cache capacity trade-off for decentralized caching with $N = 30$ and $K = 50$.

as WTP, achieves the same delivery rate as the MNC scheme in this scenario. We have also included two lower bounds on the delivery rate, the cut-set bound and the bound derived in [6]. We note that, despite the improvement of the proposed group-based caching scheme, the gap between the lower and upper bounds remain large. We believe that, this is mainly due to the looseness of the lower bound.

In Fig. 3, we compare the achievable delivery rate of the proposed GBD scheme with the schemes proposed in [2] and [4], referred to as MAN and WTP, respectively, for $N = 30$ files, $K = 50$ users, and relatively small cache capacities. We observe that the GBD scheme outperforms the existing schemes in the literature. In Fig. 3, we also include the best delivery rate that could be achieved by a centralized caching scheme in this setting. This is achieved by the CFL scheme for $M \leq 1/K$, and by the GBC for $1/K \leq M \leq \hat{t}N/K$ (in this case, $\hat{t} = t^* = 2$), and the MAN scheme for $M \geq \hat{t}N/K$. Note that, this is not a lower bound on the optimal decentralized delivery rate in general since it is not the optimal centralized delivery rate. However, the difference between the decentralized curves and the centralized curve indicates the loss due to decentralization for these specific coded caching schemes under consideration. The improvement of the GBD scheme over the state-of-the-art is more pronounced for the relatively smaller cache capacities, for which the performance of GBD approaches the best achievable centralized caching performance in a decentralized manner.

VI. Conclusions

We have introduced a novel group-based coded caching scheme, called the GBC scheme, suitable for caching systems that have more users than the number of popular files in the database. We have introduced this scheme in a centralized setting when the total cache capacity across the system is exactly sufficient to cache all the files in the database, i.e., $M = N/K$. Having shown that the GBC scheme achieves the lowest delivery rate known in the literature in this case, we then extend the improvement to a larger set of cache capacities through memory-sharing with the known schemes. Finally, we have also developed a novel decentralized coded caching scheme, i.e., GBD scheme, based on the GBC scheme, and have shown that it reduces the required delivery rate in this setting as well.

References

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

[2] ——, "Decentralized caching attains order optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 1029–1040, Apr. 2014.

[3] Z. Chen, P. Fan, and K. B. Letaief, "Fundamental limits of caching: Improved bounds for small buffer users," *arXiv: 1407.1935v2 [cs.IT]*, Jul. 2014.

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

[5] M. Mohammadi Amiri and D. Gündüz, "Improved delivery rate-cache capacity trade-off for centralized coded caching," *Int'l Symp. on Inform. Theory and Its Applications (ISITA)*, Monterey, CA, Oct. 2016.

[6] 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*, Hong Kong, Jun. 2015, pp. 1691–1695.

[7] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Toronto, ON, Apr. 2014, pp. 221–226.

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

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

[10] M. Mohammadi Amiri, Q. Yang, and D. Gündüz, "Coded caching for a large number of users," *arXiv:1605.01993v1 [cs.IT]*, May 2016.