Caching and Delivery via Interference Elimination

Chao Tian, *Senior Member, IEEE*, and Jun Chen, *Senior Member, IEEE*

***Abstract*— We propose a new coded caching scheme where linear combinations of the file segments are cached at the users, for the cases where the number of files is no greater than the number of users. When a user requests a certain file in the delivery phase, the other file segments in the cached linear combinations can be viewed as interference. The proposed scheme combines rank-metric codes and maximum distance-separable codes to facilitate the decoding and elimination of the interference and also to simultaneously deliver useful contents to the intended users. The performance of the proposed scheme can be explicitly evaluated, and we show that it can achieve improvement over known memory-rate tradeoff achievable results in the literature in some regime; for certain special cases, the new memory-rate tradeoff points can be shown to be optimal.**

***Index Terms*— Cache memory, information theory.**

I. Introduction

CACHING is a natural data management strategy when communication has a bursty characteristic. During off-peak time, local cache can be filled with data that is anticipated to be useful later to reduce the delay when the communication resources become scarce during peak time.

In a recent work [1], Maddah-Ali and Niesen provided a formal information theoretic formulation for the caching problem. In this formulation, there are N files, each of F bits, and K users. Each user has a local cache memory of capacity MF (thus a normalized capacity of M). In the placement phase, the users can fill their caches with contents from the central server without the knowledge of the precise requests. In the delivery phase, each user will request one file from the central server, and the central server must multicast certain common (and possibly coded) information to all the users in order to accommodate these requests. Since in the placement phase, the requests at the later phase are unknown, the cached contents must be strategically prepared at all the users. The goal is to minimize the amount of multicast information which has rate RF (or equivalently the normalized rate of R), under the constraint on cache memory M; see Fig. 1.

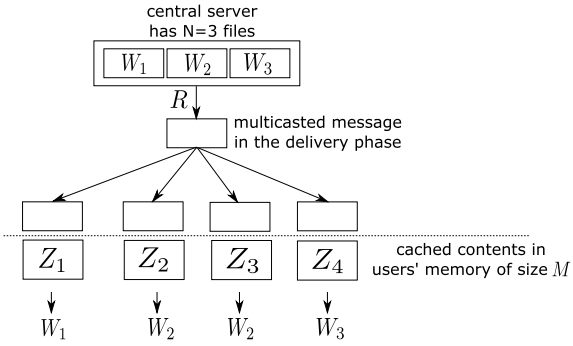

Fig. 1. An example caching system instance, where there are $N = 3$ files, denoted as (W_1, W_2, W_3), and $K = 4$ users, whose cached contents are (Z_1, Z_2, Z_3, Z_4), respectively. In this instance the users request files $(1, 2, 2, 3)$, respectively, the multicast information together with the cached contents at each user, can be used to recover the requested files.

It was shown in [1] that coding can be rather beneficial in this setting, while uncoded solutions suffer a significant loss. Subsequent works extended it also to decentralized caching placements [2], caching with nonuniform demands [3], online caching placements [4], and hierarchical coded caching [5]; the caching methods have also found their applications in other communication systems [6].

The scheme given in [1] utilizes uncoded placement and coded delivery transmission. A close inspection of the performance of the scheme reveals that when $N \leq K$, many individual memory-rate tradeoff points achieved by the scheme in [1] are not on the lower convex envelope, and thus an effective scheme is lacking for this case, particularly when the cache capacity is small. Though the scheme in [1] was shown to be within a constant factor of the optimum, the loss of efficiency can be relatively significant when either N or K is small. Particularly, for more sophisticated caching scenarios, usually either files or users need to be classified into smaller groups (see *e.g.* [3]), and such loss of efficiency may be magnified. Recently a special scheme given in [1] for the case of $N = K = 2$ was extended to the case $N \leq K$ in [7], and it was showed that the memory-rate tradeoff pair $\left(\frac{1}{K}, \frac{N(K-1)}{K}\right)$ is achievable and in fact optimal.

In this work, we propose a new coded caching scheme when $N \leq K$ that caches linear combinations of the file segments. When files are not being requested by a user, their segments in the cached linear combinations can be considered as interference by this user. Our scheme strategically eliminates the interference by utilizing a combination of rank metric codes and maximum distance separable codes; the delivery transmission also simultaneously serves the role of content

Manuscript received April 19, 2016; revised November 17, 2017; accepted December 28, 2017. Date of publication January 17, 2018; date of current version February 15, 2018. C. Tian was supported by the National Science Foundation under Grant CCF-15-26095. This paper was presented in part at the 2016 International Symposium on Information Theory.

C. Tian was with the Department of Electrical Engineering and Computer Science, The University of Tennessee, Knoxville, TN 37996 USA. He is now with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: chao.tian@tamu.edu).

J. Chen is with the Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON L8S 4L8, Canada (e-mail: junchen@ece.mcmaster.ca).

Communicated by K. Narayanan, Associate Editor for Coding Techniques.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2018.2794543

delivery to other users. We show that the proposed scheme can provide new memory-rate tradeoff points that are strictly better than the known achievable memory-rate tradeoff results in the literature. In fact, in certain cases, it can achieve points on the optimal tradeoff function. In contrast to previous schemes in the literature, the proposed codes are not binary, but in larger finite fields. Although we utilize rank metric codes as a convenient tool to construct general explicit codes, it is by no means necessary. In fact one disadvantage of using rank metric codes is the large field size that the codes require, and we show that by directly considering the underlying rank constraints and utilizing generic linear codes, a smaller field size is sufficient for such codes to exist.

In the rest of the paper, we shall first give the main theorem in Section II, then introduce some preliminaries in Section III. Before presenting the new codes, we provide three examples to illustrate the design principles in Section IV. The coding scheme, the corresponding proofs of correctness and analysis are given in Section V and Section VI, respectively. We conclude the paper in Section VII, and relegate some more technical proofs to the appendix.

II. Main Theorem

The main result of this paper is summarized below, where $\mathbb{N}$ is used to denote the set of natural numbers.

Theorem 1: For $N \in \mathbb{N}$ files and $K \in \mathbb{N}$ users each with a cache of size M, where $N \leq K$, the following memory-rate (M, R) pairs are achievable

$$\left(\frac{t[(N-1)t+K-N]}{K(K-1)}, \frac{N(K-t)}{K}\right), \quad t = 0, 1, \ldots, K. \tag{1}$$

With $t = 0$ the memory-rate tradeoff point degenerates to the trivial one $(M, R) = (0, N)$, *i.e.,* no cache; when $t = 1$, it gives the same memory-rate pair as given in [7]; when $t = K$, we obtain another trivial point of $(M, R) = (N, 0)$, *i.e.,* no delivery transmission.

The new memory-rate tradeoff points are illustrated for the case $(N, K) = (2, 4)$ in Fig. 2. For reference, the tradeoff points achieved by the scheme in [1] and the cut-set based lower bound [1] are also shown, together with a computation-based lower bound established in a separate work (see [8]) using a method developed in [9]. The two leftmost points achieved by the proposed scheme (the solid black line labeled with diamonds) are previously known, which are the trivial case with no cache, and the point given in [7], respectively. The third point is previously unknown to be achievable, and it is explained in detail in Section IV. Here all three points given by the new code are in fact on the optimal tradeoff function. The new memory-rate point $(M, R) = (2/3, 1)$ is strictly better than the known upper bound given in [1], which is shown in dashed magenta in Fig. 2.

In the proposed scheme, for demands where not all files are requested, the scheme can be viewed as degenerate cases of the scheme for certain enhanced demands, where all files are being requested. Although the scheme for such demands can be viewed as degenerate, this does not imply the memory-rate tradeoff points achieved by the proposed scheme are only

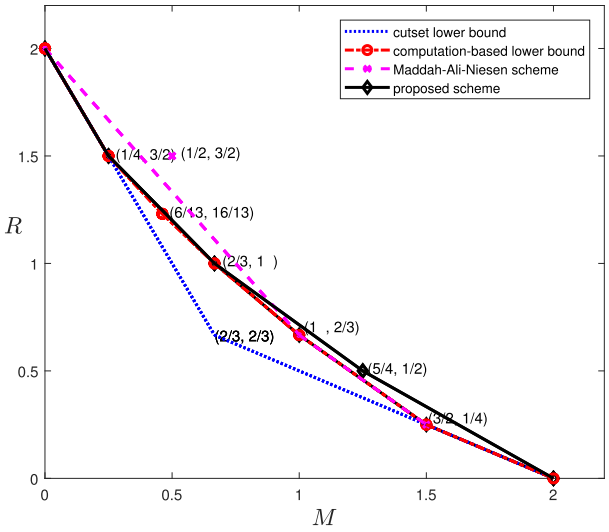

Fig. 2. Illustration of the memory-rate tradeoff upper bounds and the lower bounds for $(N, K) = (2, 4)$. The pair $(1/2, 3/2)$ can be achieved by the scheme given in [1], but it is not on the convex envelope of the previously known upper bound.

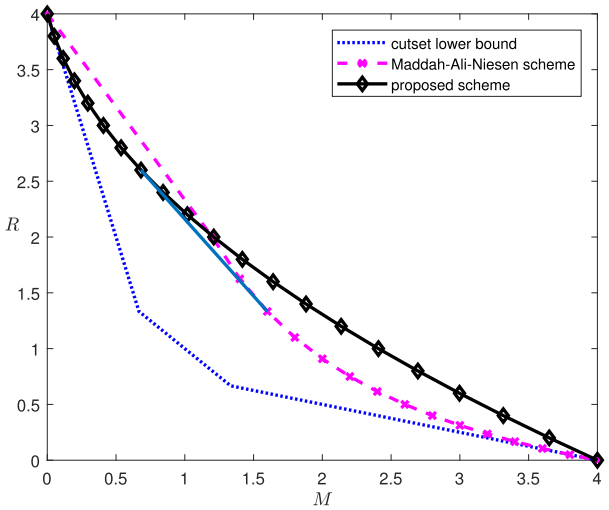

Fig. 3. Illustration of the achievable memory-rate tradeoffs for $(N, K) = (4, 20)$. The solid teal line completes the lower convex envelope of the memory-rate pairs by the proposed scheme and those by the scheme in [1].

effective when $R \geq N - 1$, for which non-trivial codes are required only for the demands that all files are requested; see Sections V and VI. The different memory-rate tradeoff points achieved by the proposed scheme and those achieved by the scheme given in [1] are illustrated for the case of $(N, K) = (4, 20)$ in Fig. 3. The point $(M, R) = (259/380, 13/5)$ is on the lower convex envelope, and the transmission rate is less than $N - 1 = 3$ here.

As indicated by Fig. 2 and Fig. 3, for the cases $N \leq K$ where the proposed codes are valid, there are in general three regimes: the proposed scheme performs better in the low cache memory regime, the scheme in [1] performs better in the high cache memory regime, and there is a transition regime in between. Since the proposed scheme in fact naturally

generalizes the code given in [7], which was shown to be optimal when $M \leq \frac{1}{K}$, the improvement by the proposed scheme in the low memory regime is indeed expected. Moreover, it was shown in [7] that the scheme in [1] is strictly sub-optimal in this range, and as a consequence the scheme proposed in our work is strictly better than that in [1] when $M \leq \frac{1}{K}$, however as M increases, the advantage appears to decrease gradually and then vanishes. The precise amount of improvement and the exact transition point between the proposed scheme and that in [1] turn out to be quite difficult to characterize, mainly due to the fact that both are represented as the convex envelopes of some discrete memory-rate pairs. Nevertheless we can form a new upper bound to the optimal memory-rate tradeoff by taking the lower convex envelope of the union of the memory-rate tradeoff points in Theorem 1 and those in [1], as illustrated more explicitly in Fig. 3.

III. Preliminaries

In this section we review some existing results on the caching problem, and then provide some necessary background information on maximum distance separable (MDS) codes and rank metric codes.

A. Known Caching Schemes and Memory-Rate Tradeoffs

Theorem 2 (Maddah-Ali and Niesen [1]): For $N \in \mathbb{N}$ files and $K \in \mathbb{N}$ users each with a cache of size $M \in \{0, N/K, 2N/K, \ldots, N\}$, the rate

$$R = K(1 - M/N) \cdot \min\left\{\frac{1}{1 + KM/N}, \frac{N}{K}\right\} \tag{2}$$

is achievable. For general $0 \leq M \leq N$, the lower convex envelope of these memory-rate (M, R) points is achievable.

The first value in the minimization is achieved by the scheme of uncoded placement together with coded transmission [1], while the latter value is by simple uncoded placement and uncoded transmission. Though this theorem is indeed correct, it can be slightly misleading since it may give the impression that the simple uncoded placement and uncoded transmission scheme can be effective in certain regime when $N < K$. A close examination reveals that this trivial scheme only provides one operating point $(N, 0)$ in the convex envelope when $N \leq K$, as illustrated in Fig. 2. Thus a good caching strategy for the low memory case is still lacking.

As mentioned early, in a recent work [7], Chen *et al.* extended a special scheme for the case $N = K = 2$ discussed in [1] to the general case $N \leq K$, and showed that the memory-rate tradeoff pair $\left(\frac{1}{K}, \frac{N(K-1)}{K}\right)$ is achievable. It should be noted that the scheme given in [1] uses uncoded placement with coded transmission, while the scheme in [7] uses coded placement and coded transmission. Both schemes use only binary coding, in contrast to the codes we propose in this work. As noted in [1], the caching problem is related to the well known index coding problem [10], however, this connection has not led to any concrete results on the caching problem.

B. Maximum Distance Separable Codes

A linear code of length n and dimension k is called an (n, k) code. The Singleton bound (see *e.g.*, [11]) is a well known upper bound on the minimum distance for any (n, k) code, given as

$$d_{\min} \leq n - k + 1. \tag{3}$$

An (n, k) code that satisfies the Singleton bound with equality is called a maximum distance separable (MDS) code. A key property of an MDS code is that it can correct any $(n - k)$ or fewer erasures [11]. For any (n, k) pairs where $n \geq k$, MDS codes exist in any finite field $\mathbb{F}_q$ when $q \geq n$.

C. Linearized Polynomial and Rank Metric Codes

In order to handle the competing coding requirements in the caching problem, we use rank metric codes based on linearized polynomials (see [12]), for which the following lemma is particularly relevant; see, *e.g.,* [13].

Lemma 1: A linearized polynomial in the finite field $\mathbb{F}_{q^m}$

$$f(x) = \sum_{i=1}^{P} v_i x^{q^{i-1}}, \quad v_i \in \mathbb{F}_{q^m} \tag{4}$$

can be uniquely identified from evaluations at any P points $x = \theta_i \in \mathbb{F}_{q^m}$, $i = 1, 2, \ldots, P$, that are linearly independent over $\mathbb{F}_q$.

Another relevant property of linearized polynomials is that they satisfy the following condition

$$f(ax + by) = af(x) + bf(y), \quad a, b \in \mathbb{F}_q, \ x, y \in \mathbb{F}_{q^m}, \tag{5}$$

which is the reason that they are called "linearized". This property implies the following lemma.

Lemma 2: Let $f(x)$ be a linearized polynomial in $\mathbb{F}_{q^m}$ as given in (4), and let $\theta_i \in \mathbb{F}_{q^m}$, $i = 1, 2, \ldots, P_o$, be linearly independent over $\mathbb{F}_q$. Let G be a $P_o \times P$ full rank (rank P) matrix with entries in $\mathbb{F}_q$, then $f(x)$ can be uniquely identified from

$$[f(\theta_1), f(\theta_2), \ldots, f(\theta_{P_o})] \cdot G. \tag{6}$$

Proof: We slightly abuse the notation by allowing the function $f(x)$ to take vector input in $\mathbb{F}_{q^m}^{P_0}$, and define the output as the vector obtained by concatenating the output of $f(x)$ on each input component. Then by the linearized property,

$$\begin{aligned}[f(\theta_1), f(\theta_2), \ldots, f(\theta_{P_o})] \cdot G &= [f(\theta_1, \theta_2, \ldots, \theta_{P_o})] \cdot G \\ &= f[(\theta_1, \theta_2, \ldots, \theta_{P_o}) \cdot G].\end{aligned}$$

Recall when each θ_i is viewed as a vector in $\mathbb{F}_q^m$, the $(\theta_1, \theta_2, \ldots, \theta_{P_o})$ vectors are linearly independent. Since G has rank P, $(\theta_1, \theta_2, \ldots, \theta_{P_o}) \cdot G$ has rank P in $\mathbb{F}_q$, *i.e.*, we have P evaluations of $f(x)$ at P linearly independent values, and thus by Lemma 1, $f(x)$ can be uniquely identified. ∎

With a fixed set of $\theta_i \in \mathbb{F}_{q^m}$, $i = 1, 2, \ldots, P_o$, which are linearly independent, we can view $(v_1, \ldots, v_P)$ as information symbols to be encoded, and the evaluations $[f(\theta_1), f(\theta_2), \ldots, f(\theta_{P_o})]$ as the coded symbols. This is a (P_o, P) MDS code in terms of rank metric [12], where $P_o \geq P$. More importantly, the above lemma says any full rank

TABLE I
CACHING CONTENT FOR $(N, K) = (2, 4)$

User 1	$A_1 + B_1$	$A_2 + B_2$	$A_3 + B_3$	$A_1 + A_2 + A_3 + 2(B_1 + B_2 + B_3)$
User 2	$A_1 + B_1$	$A_4 + B_4$	$A_5 + B_5$	$A_1 + A_4 + A_5 + 2(B_1 + B_4 + B_5)$
User 3	$A_2 + B_2$	$A_4 + B_4$	$A_6 + B_6$	$A_2 + A_4 + A_6 + 2(B_2 + B_4 + B_6)$
User 4	$A_3 + B_3$	$A_5 + B_5$	$A_6 + B_6$	$A_3 + A_5 + A_6 + 2(B_3 + B_5 + B_6)$

(rank P) $\mathbb{F}_q$ linear combinations of the coded symbols are sufficient to decode all the information symbols. This linear-transform-invariant property had been utilized previously in other coding problems such as network coding with errors and erasures [14], locally repairable codes with regeneration [15], and layered regenerating codes [16].

The codes thus obtained are not systematic, but they can be converted to systematic codes by viewing the information symbols $(w_1, w_2, \ldots, w_P)$ as the first P evaluations $[f(\theta_1), f(\theta_2), \ldots, f(\theta_P)]$, which can be used to find the coefficients of the linearized polynomial $(v_1, v_2, \ldots, v_P)$, and then the additional parity symbols can be generated by evaluating this linearized polynomial at the remaining points $(\theta_{P+1}, \ldots, \theta_{P_o})$. Systematic rank-metric codes are instrumental in our construction.

IV. THREE EXAMPLES

In this section, we provide three examples to illustrate the placement and transmission mechanism and discuss several critical observations. These observations provide important intuitions, which are used to design the placement and the transmission strategy for the general case.

A. A $(2, 4)$ *Code: The Rank Counting Perspective*

This example is the main prototype code that leads to the discovery of the general code construction, and we present it next in an explicit and symmetric form in a specific small alphabet. This form bears certain similarity to the code in both [1] and [7], and readers familiar with those works may find this connection of interest. The intuitions behind this code will then be discussed from a more general viewpoint.

In this example, the two files are denoted as A and B, each of which is partitioned into 6 segments of equal size, denoted as A_i and B_i, respectively, $i = 1, 2, \ldots, 6$. This example corresponds to $t = 2$, and the number of segments being 6 is deduced from $\binom{K}{t}$, for reasons soon to be made clear in the sequel. The contents in the cache of each user are given in Table I. By the symmetry of the cached contents, we only need to consider the demand (A, A, A, B), *i.e.*, the first three users requesting A and user 4 requesting B, and the demand (A, A, B, B), *i.e.,* the first two users requesting A and the other two requesting B. Assume the file segments are in $\mathbb{F}_5$, which is the field we operate in. This code we present next can achieve $(M, R) = (\frac{2}{3}, 1)$ which is strictly outside the known achievable memory-rate tradeoff, as illustrated in Fig. 2.

- For the demands (A, A, A, B), the transmissions are

  Step 1: B_1, B_2, B_4;
  Step 2: $A_3 + 2A_5 + 3A_6, A_3 + 3A_5 + 4A_6$;
  Step 3: $A_1 + A_2 + A_4$.

  After Step 1, user 1 can recover (A_1, A_2); furthermore, he has $(A_3 + B_3, A_3 + 2B_3)$ by eliminating known symbols (A_1, A_2, B_1, B_2), from which A_3 can be recovered. After Step 2, he can obtain $(2A_5 + 3A_6, 3A_5 + 4A_6)$ to recover (A_5, A_6). Using the transmission in Step 3, he can obtain A_4 since he has (A_1, A_2). User 2 and user 3 can use a similar strategy to reconstruct all file segments in A. User 4 only needs B_3, B_5, B_6 after Step 1, which he already has in his cache, however they are contaminated by file segments from A. Nevertheless, he knows $A_3 + A_5 + A_6$ by recognizing

  $$(A_3 + A_5 + A_6) = 2 \sum_{i=3,5,6} (A_i + B_i) - [A_3 + A_5 + A_6 + 2(B_3 + B_5 + B_6)]. \quad (7)$$

  Together with the transmission in Step 2, user 4 has three linearly independent combinations of (A_3, A_5, A_6). After recovering them, he can remove the interference from the cached content for (B_3, B_5, B_6).
- For the demand (A, A, B, B), we can send

  Step 1: B_1, A_6;
  Step 2: $A_2 + 2A_4, A_3 + 2A_5, B_2 + 2B_3, B_4 + 2B_5$.

  User 1 has A_1, B_1, A_6 after Step 1, and he can also form

  $$B_2 + B_3 = [A_2 + A_3 + 2(B_2 + B_3)] - (A_2 + B_2) - (A_3 + B_3),$$

  and together with $B_2 + 2B_3$ in the transmission of Step 2, he can recover (B_2, B_3), and thus A_2, A_3. He still needs (A_4, A_5), which can be recovered straightforwardly from the transmission $(A_2 + 2A_4, A_3 + 2A_5)$ since he already has (A_2, A_3). Other users can use a similar strategy to decode their requested files.

This example may seem rather complicated and arbitrary at first sight, however, we can make a few observations which should clarify the purpose of each transmission.

The placement of the file segments has certain similarity to the scheme in [1]. Each file is partitioned into segments, and each segments are given to multiple users, however, they are stored only as linear combinations with segments from other files. The first several (3 in this example) cached symbols can be viewed as *semi-systematic*, as they are simple summations

of the corresponding file segments, while the last symbol is a local parity symbol. However, it is not necessary to classify the cached contents at a user into these two categories, but we choose this form to facilitate presentation. In the next two examples and the general construction, we present the code in a more general manner.

Transmissions in Step 1 are uncoded which provide certain segments to users that request it, but at the same time help to eliminates some interference at other users. A segment from a file is transmitted uncoded only when it is not present at any users[1] that are requesting this file. Step 2 is coded transmission, and it also serves the dual role of interference elimination and content delivery. In this step, we only transmit linear combinations of segments, each of which is formed by linearly combining segments from a single file; in fact, each such combination is formed with symbols present at a single user that is not requesting this file. For example, for the case (A, A, A, B), the transmission $A_3 + 2A_5 + 3A_6$ has symbols in the cache of user 4, but user 4 is not requesting file A. The coefficients of the linear combinations in the placement and the transmission need to be chosen carefully to guarantee certain full rank property; *cf.* again, the transmissions by user 4 for the case (A, A, A, B) in the example.

The most important observation is the following alternative view of the transmission and decoding process. Take for instance the case with demand (A, A, A, B): user 4 receives symbols $(A_3 + 2A_5 + 3A_6, A_3 + 3A_5 + 4A_6)$, together with 4 cached symbols, all of which are linear combinations of basis $(A_3, A_5, A_6, B_3, B_5, B_6)$. If these linear combinations are linearly independent, then all these symbols can be solved. A close inspection reveals they are indeed linearly independent, and in fact the decoding process at any given user can be understood this way. The precise linear combination coefficients are not important, however, the linear independence (or the coding matrix being full rank) directly leads to the resolution of all interference. For this reason, in the next example we do not explicitly specify the linear combination coefficients, but only the basis of the subspace and the dimension. For this purpose, we introduce the linear subspace notation of

$$\mathcal{L}[\text{subset of files; index subset; dimension}], \tag{8}$$

which means a subspace of the given dimension with the basis being the segments from the given files with the given subscript indices. For example, the subspace spanned by $(A_3 + 2A_5 + 3A_6, A_3 + 3A_5 + 4A_6)$ shall be written as $\mathcal{L}[A; \{3, 5, 6\}; 2]$, which means a dimension 2 linear subspace in the subspace with basis (A_3, A_5, A_6). Further notice that if the dimension is chosen to be the same as the dimension of the subspace, it is equivalent to the uncoded transmissions of this basis in its entirety. We shall assume in the next example all necessary full rank properties can be satisfied by properly choosing the coefficients, and in the general scheme, we show that one particular choice of such coefficients based on linearized polynomials indeed exists.

[1]In the proposed scheme, a file segment is present in a user's cache only as a component in some linear combinations, however we shall simply refer to it as "present" at the user.

TABLE II
INTERFERENCE PATTERN FROM FILE A FOR $(N, K) = (3, 6)$

User 4	$\underline{A_{1,4,5}}$	$\underline{A_{2,4,5}}$	$\underline{A_{3,4,5}}$	$A_{1,4,6}$	$A_{2,4,6}$	$A_{3,4,6}$
User 5	$\underline{A_{1,4,5}}$	$\underline{A_{2,4,5}}$	$\underline{A_{3,4,5}}$	$\overline{A_{1,5,6}}$	$\overline{A_{2,5,6}}$	$\overline{A_{3,5,6}}$
User 6	$A_{1,4,6}$	$A_{2,4,6}$	$A_{3,4,6}$	$\overline{A_{1,5,6}}$	$\overline{A_{2,5,6}}$	$\overline{A_{3,5,6}}$

B. A $(3, 6)$ *Code: Efficient Interference Elimination*

Given the observations above, we shall from here on adopt the indexing method in [1], and enumerate the file segments by the subset of users' caches that they are present in. For example when $(N, K) = (3, 6)$, file A has segments $A_{1,2,3}, A_{1,2,4}$, etc., and $A_{1,2,3}$ is present at users 1, 2, and 3 in some linear combinations; *i.e.,* we choose to place any file segment at $t = 3$ users, possibly as a component of some linear combinations. In this example, we reserve the letter $\mathcal{S}$ to enumerate some subset $\mathcal{S} \subseteq \{1, 2, \ldots, 6\}$ and $|\mathcal{S}| = t = 3$, where $|\cdot|$ is used to denote the cardinality of a set. For the case of $K = 6$, the k-th user caches the following linear combinations of files (A, B, C):

$$\mathcal{L}[A, B, C; \{\mathcal{S} : k \in \mathcal{S}\}; 18],$$

where the dimension 18 is chosen because the memory usage at this point is $9/10$ as in Theorem 1, and each file is partitioned into $\binom{6}{3} = 20$ segments, which implies that each user should cache $20 \times 9/10 = 18$ symbols.

We shall not discuss all the cases of file demands for this example because it is rather lengthy, but will consider one case, since it brings out a very important ingredient in our transmission strategy.

Let us consider the case when the users request (A, A, A, B, B, C). The transmissions in Step 1 are uncoded transmissions similar as in the previous case, however let us focus our attention on users 4, 5, 6 which are not requesting A, in the subsequent steps. After the transmissions in Step 1, these users still have the file segments in Table II as *interference*, among many others (such as $A_{1,2,4}, A_{2,3,4}, \ldots$ for user 4). Though we can transmit linear combinations of the basis

$$A_{1,4,5}, A_{2,4,5}, A_{3,4,5}, A_{1,4,6}, A_{2,4,6}, A_{3,4,6}, \tag{9}$$

directly to eliminate this interference at user 4, this strategy is not very efficient. Observe the following: the basis $(A_{1,4,5}, A_{2,4,5}, A_{3,4,5})$, which are underlined in the table, are present in both user 4 and user 5; the basis $(A_{1,4,6}, A_{2,4,6}, A_{3,4,6})$ are at both user 4 and user 6; $(A_{1,5,6}, A_{2,5,6}, A_{3,5,6})$, which are overlined in the table, are at user 5 and user 6. We can thus alternatively transmit

$$\mathcal{L}[A; \{\{1, 4, 5\}, \{2, 4, 5\}, \{3, 4, 5\}\}; 2],$$
$$\mathcal{L}[A; \{\{1, 4, 6\}, \{2, 4, 6\}, \{3, 4, 6\}\}; 2],$$
$$\mathcal{L}[A; \{\{1, 5, 6\}, \{2, 5, 6\}, \{3, 5, 6\}\}; 2],$$

TABLE III
CACHING CONTENT FOR THE EXAMPLE $(N, K) = (3, 4)$

User 1	$\mathcal{L}[A, B, C; \{\{1,2\},\{1,3\},\{1,4\}\}; 5]$
User 2	$\mathcal{L}[A, B, C; \{\{1,2\},\{2,3\},\{2,4\}\}; 5]$
User 3	$\mathcal{L}[A, B, C; \{\{1,3\},\{2,3\},\{3,4\}\}; 5]$
User 4	$\mathcal{L}[A, B, C; \{\{1,4\},\{2,4\},\{3,4\}\}; 5]$

where because of the change of indexing method, we need to specify the segments for a file by a collection of subsets; for example the first row means 2 linear combinations of $(A_{1,4,5}, A_{2,4,5}, A_{3,4,5})$. Each of these subspaces provides 2 dimensional reduction of the interference at 2 users simultaneously. This results in a total of dimension 4 interference reduction at each user with transmission of 6 symbols, which is difficult to accomplish without taking advantage of these subspace intersections.

C. A $(3, 4)$ *Code: Degenerate File Requests*

In this example, there are three files (A, B, C), and we choose the parameter $t = 2$, *i.e.,* each file is partitioned into 6 segments and each segment is placed at two users. We wish to show that the memory-rate pair $(M, R) = (\frac{5}{6}, \frac{3}{2})$ is achievable by extending the code given in the previous examples, though this memory-rate point is actually worse than known results in the literature. Note that since $R \leq 2$, the types of demands where only two files are requested cannot be satisfied by simply transmitting these files directly. As it turns out, these cases can be considered as degenerate from the cases when all files are being requested by the users.

The three users cache the contents as shown in Table. III. Only the following three types of requests need to be considered due to symmetry:

- For the case (A, A, B, C), the transmissions are:

  Step 1: $A_{3,4}, B_{1,2}, B_{1,4}, B_{2,4}, C_{1,2}, C_{1,3}, C_{2,3}$;
  Step 2: $\mathcal{L}[A; \{\{1,3\},\{2,3\}\}; 1], \mathcal{L}[A; \{\{1,4\},\{2,4\}\}; 1]$.

- For the case (A, A, B, B), the transmissions are:

  Step 1: $A_{3,4}, B_{1,2}, B_{1,4}, B_{2,4}$;
  Step 2: $\mathcal{L}[A; \{\{1,3\},\{2,3\}\}; 1], \mathcal{L}[A; \{\{1,4\},\{2,4\}\}; 1]$;
  Step 4: $B_{1,3}, B_{2,3}, C_{1,2}$.

- For the case (A, A, A, C), the transmissions are:

  Step 1: $A_{3,4}, C_{1,2}, C_{1,3}, C_{2,3}$;
  Step 2: $\mathcal{L}[A; \{\{1,3\},\{2,3\}\}; 1], \mathcal{L}[A; \{\{1,4\},\{2,4\}\}; 1]$;
  Step 4: $A_{1,2}, A_{1,4}, B_{2,4}$.

It can be verified that these transmissions indeed fulfill all the demands by counting the rank reduction for the purpose of interference elimination, as discussed in the first example. Next let us make a few more observations in this solution.

The transmission for the first case follows the strategy we have identified in the first example, but the other two cases require additional attention. For those two cases, the first two steps are still in line with our previous example for $(N, K) = (2, 4)$, but there is an additional Step 4 (Step 3 is void in this specific example), where uncoded transmissions are used. In fact, the transmissions in the first two steps for the latter two cases are precisely those in the first two steps for the first case, except that the transmissions involving files not being requested are omitted. In the transmissions of Step 4, instead of transmitting the segments of the file not being requested, the corresponding file segments from another file are transmitted, with a few exceptions when those substituted segments have already been transmitted; if this occurs, the corresponding segments from the file not being requested are in fact transmitted.

We can view the transmissions in the latter two cases as a variation from that in the first case. Let us focus on the case (A, A, B, B): the only difference from the case (A, A, B, C) is that user 4 is requesting file B instead of C. A closer examination of the case (A, A, B, C) reveals that all transmissions involving file C are uncoded. Now to build the transmissions for the case (A, A, B, B) from the transmissions for the case (A, A, B, C), we replace these uncoded transmissions with the matching transmissions of segments of file B, however, only when there is no redundancy in such transmissions. For example, the last symbol to be transmitted should have been $B_{1,2}$ with such a straightforward substitution, but since we have already transmitted $B_{1,2}$, retransmitting it is unnecessary and wasteful; instead the file segment $C_{1,2}$ is transmitted. In this case although no user is requesting file C, the last transmission does not cause any essential loss. In summary, a case when only a subset of files are requested can be viewed as degenerate, for which the transmission strategy can be deduced from some other case when all files are requested.

V. THE GENERAL CACHING SCHEME

Before presenting the general coding scheme, let us reexamine the intuitions obtained from the three example cases given above. Firstly, we should for now focus on the cases where all files are requested, because the last example suggests that other cases may be viewed as degenerate. Secondly, during the delivery phase, we need to choose carefully for each transmission an appropriate basis (i.e., which symbols to form the linear combinations), and in particular, attempt to take advantage of the common subspaces which can be extracted from the linear combinations cached at different users, as suggested by the second example. The last but also the most technical issue is that we need to choose two sets of coding coefficients, one for forming the linear combinations of the cached contents, and the other for forming the delivery transmissions which are also linear combinations of certain symbols of the aforementioned basis.

The first example in the previous section suggests that the coding coefficients need to be assigned such that a set of full rank conditions are satisfied which can guarantee that all involved file symbols can be decoded from their coded form. However, specifying them explicitly turns out to be difficult for generic parameters. In the construction we present next, this issue is resolved by a combination of MDS codes and rank metric codes. The apparent main advantage of this

approach is that the construction is indeed explicit, instead of only an existence proof if we were to argue directly that a valid coefficient assignment exists (see, e.g., [17]). In fact, even if we were to adopt the latter approach directly, one needs to show that certain coding matrices are not always rank deficient (i.e., the determinant is not identically zero). Though this can be done, it is not only notationally tedious but also conceptually less revealing. Using rank metric codes in the construction also resolves this issue as a byproduct, since its validity alone guarantees the existence of non-rank-deficient coefficient assignments. In Section 6.4, we indeed invoke this fact to show that the alphabet size bound sufficient to guarantee the construction based on rank metric codes to be valid is not necessary, if we allow general linear codes in place of the rank metric codes.

We next clarify the notation that will be used in the sequel. The set of integers $\{1, 2, \ldots, n\}$ is written as I_n, and the cardinality of a set $\mathcal{A}$ is written as $|\mathcal{A}|$. Denote the N files as $W_1, W_2, \ldots, W_N$. Fix an integer parameter $t \in \{1, 2, \ldots, K\}$ in the proposed scheme, then each file in our scheme is partitioned into $\binom{K}{t}$ segments of equal size, and each file segment will be placed at t users, possibly as a component of some linear combinations. Each segment $W_{n,\mathcal{S}}$, where $n \in I_N$ and $\mathcal{S} \subseteq I_K$ with $|\mathcal{S}| = t$, is in $\mathbb{F}_{q^m}$ for some sufficiently large q and m, however it will become clear that any $q \geq 2^{\binom{K-N+1}{\max(\lfloor (K-N+1)/2 \rfloor, t)}}$, and $m \geq P_o$ as defined in (11) are sufficient. We reserve the calligraphic letter $\mathcal{S}$ for the purpose of enumerating some of the subsets of I_K of carnality t, without explicitly writing these conditions for notational simplicity.

To present the general scheme, a few additional coding components are required. We first need a set of generic systematic linear MDS codes whose generator matrices have entries in $\mathbb{F}_q$ with parameters (n_c, k_c), for all $n_c \geq k_c \geq 1$ and $n_c \leq q$; such codes can be found for any sufficiently large q, for example, using Cauchy matrix [18]. We also allow the information symbols and coded symbols to be in $\mathbb{F}_{q^m}$, by taking the natural $\mathbb{F}_{q^m}$ finite field operation, for which $\mathbb{F}_q$ is a subfield of; this essentially boils down to writing the symbols as vectors of length-m in $\mathbb{F}_q$. Furthermore, fix the parameter

$$P = \binom{K-1}{t-1} N \tag{10}$$

in the linearized polynomial and also fix

$$P_o = 2\binom{K-1}{t-1} N - \binom{K-2}{t-1}(N-1) \tag{11}$$

values $\theta_i \in \mathbb{F}_{q^m}$, $i = 1, 2, \ldots, P_o$, which are linearly independent in $\mathbb{F}_q$. This polynomial can be used to construct a (P_o, P) systematic rank metric code as discussed in Section III-C; we shall refer to this code as $\mathcal{C}(P_o, P)$. We are now ready to present the general caching strategy.

A. Placement Strategy

The placement strategy of the proposed scheme can be described as follows. For user k, collect the file segment symbols:

$$\{W_{n,\mathcal{S}}, \quad \text{for all } n \in I_N, \text{ and all } \mathcal{S} \text{ such that } k \in \mathcal{S}\}$$

and encode them using the systematic rank metric code $\mathcal{C}(P_o, P)$; the parity symbols are then placed in the cache of user k. It follows that each user needs a cache to hold $P_o - P$ symbols.

B. Transmission Strategy When All Files are Requested

Fix a parameter $t \in \{1, 2, \ldots, K-1\}$. Let us first consider the case when all the files are being requested; the cases $t = 0$ or $t = K$ are omitted for which the scheme is trivial. For a given set of file requests from all the users, we define

$$I^{[n]} \triangleq \{k \in I_K : \text{user } k \text{ requests file } W_n\}, \quad n = 1, 2, \ldots, N, \tag{12}$$

and $m_n = |I^{[n]}| \geq 1$, $n = 1, 2, \ldots, N$. Furthermore, define the complementary set $\bar{I}^{[n]} \triangleq I_K \setminus I^{[n]}$.

For each file W_n, we classify its segments $W_{n,\mathcal{S}}$ by its intersection with $\bar{I}^{[n]}$, and address them differently. More precisely, there are three steps of transmissions:

- Step 1: All the file segments in the set $\{W_{n,\mathcal{S}} : \mathcal{S} \subseteq \bar{I}^{[n]}\}$ are transmitted uncoded directly;
- Step 2: For each subset $\mathcal{A} \subseteq \bar{I}^{[n]}$ with $\max(1, t - m_n) \leq |\mathcal{A}| \leq \min(t-1, K - m_n)$, we encode the set of file segments

$$\mathcal{W}_{n,\mathcal{A}} \triangleq \{W_{n,\mathcal{S}} : \mathcal{S} \cap \bar{I}^{[n]} = \mathcal{A}\} \tag{13}$$

using a

$$\left(2\binom{m_n}{t-|\mathcal{A}|} - \binom{m_n-1}{t-|\mathcal{A}|-1}, \binom{m_n}{t-|\mathcal{A}|}\right) = \left(\binom{m_n}{t-|\mathcal{A}|} + \binom{m_n-1}{t-|\mathcal{A}|}, \binom{m_n}{t-|\mathcal{A}|}\right) \tag{14}$$

systematic MDS code (whose coding coefficients are in $\mathbb{F}_q$), and then transmit all the parity symbols; here we take the convention of $\binom{n}{k} = 1$ when $k = 0$.

- Step 3: Encode all the file segments in the set $\mathcal{W}_{n,\emptyset} \triangleq \{W_{n,\mathcal{S}} : \mathcal{S} \subseteq I^{[n]}\}$ using a

$$\left(2\binom{m_n}{t} - \binom{m_n-1}{t-1}, \binom{m_n}{t}\right) = \left(\binom{m_n}{t} + \binom{m_n-1}{t}, \binom{m_n}{t}\right) \tag{15}$$

systematic MDS code (whose coding coefficients are in $\mathbb{F}_q$), and then transmit all the parity symbols.

For the required MDS codes to exist, a trivially sufficient finite field size is $q \geq 2^{\binom{K-N+1}{\max(\lfloor (K-N+1)/2 \rfloor, t)}}$. For the required rank metric codes to exist, we can choose any $m \geq P_o$.

It is clear that each file segment $W_{n,\mathcal{S}}$ either belongs to a singleton set $\{W_{n,\mathcal{S}}\}$ when $\mathcal{S} \subseteq \bar{I}^{[n]}$, or one of the sets $\mathcal{W}_{n,\mathcal{A}}$ for some subset $\mathcal{A} \subseteq \bar{I}^{[n]}$, which includes the case $\mathcal{A} = \emptyset$; in other words, for each n, the transmission strategy provides a classification of all the subsets $\mathcal{S}$ for $S \subseteq I_K$ and $|\mathcal{S}| = t$ (and also induces a classification of all the file segments $W_{n,\mathcal{S}}$). For each n, we denote the mapping from a subset $\mathcal{S}$ to the corresponding subset that specifics the partition it belongs to as $\mathcal{A}_{I^{[n]}}(\mathcal{S})$, *i.e.,* $W_{n,\mathcal{S}} \in \mathcal{W}_{n,\mathcal{A}_{I^{[n]}}(\mathcal{S})}$.

C. Transmission Strategy When Some Files are Requested

Again fix a parameter $t \in \{1, 2, \ldots, K-1\}$, and consider the case when $N^* < N$ files are requested. Without loss of generality, let us assume that the first N^* files are being requested, and $I^{[n]}$, m_n and $\bar{I}^{[n]}$ are defined similarly as in the last subsection, but only for $n = 1, 2, \ldots, N^*$. To describe the transmission strategy, we first find another set of "enhanced demands", parametrized by $\dot{I}^{[1]}, \dot{I}^{[2]}, \ldots, \dot{I}^{[N]}$, where all files are being requested; *i.e.,* $|\dot{I}^{[n]}| \geq 1$ for $n = 1, 2, \ldots, N$. Additionally, these enhanced demands must satisfy the following properties:

- $|\dot{I}^{[n]}| = 1$ for $n = N^* + 1, \ldots, N$;
- For any $k \in \{1, 2, \ldots, K\}$, if $k \in I^{[n]}$, then either $k \in \dot{I}^{[n]}$, or $k \in \dot{I}^{[n']}$, for some $n' \in \{N^* + 1, \ldots, N\}$; for the latter case, denote the mapping from n' to n as $f(n') = n$, and denote the mapping from n' to k as $u(n')$.

We also write $|\dot{I}^{[n]}| = \dot{m}_n$ for simplicity. The enhancement replaces some users' requests with requests for files that originally are not being requested, and each of these files is now being requested by only one user in the enhanced version. Note that this enhancement can always be found under the condition $N \leq K$.

A set of counters need to be initialized before presenting the transmission strategy, which is given as

$$\tau_{n,\mathcal{A}} \triangleq \binom{\dot{m}_n - 1}{t - |\mathcal{A}| - 1}, \quad n = 1, 2, \ldots, N^* \text{ and } \mathcal{A} \subseteq \bar{I}^{[n]}. \tag{16}$$

Note that the set $\mathcal{A}$ can be $\emptyset$, and in fact in the proposed scheme we only need to consider the sets $\mathcal{A}$ where $|\mathcal{A}| \leq t-1$, though the definition is still valid for other cases, by taking the convention $\binom{n}{k} = 0$ if $k < 0$.

The transmission strategy is as follows:

- For each file W_n, $n = 1, 2, \ldots, N^*$, transmit according to Steps 1-3 for the **enhanced demands**;
- Step 4: for each n, $n = N^* + 1, \ldots, N$, perform the following operations. For each $\mathcal{S}$, where $u(n) \notin \mathcal{S}$, reduce the counter $\tau_{f(n),\mathcal{A}_{j[f(n)]}(\mathcal{S})}$ by 1, and then transmit

$$\begin{cases} W_{f(n),\mathcal{S}}, & \text{if } \tau_{f(n),\mathcal{A}_{j[f(n)]}(\mathcal{S})} \geq 0 \\ W_{n,\mathcal{S}}, & \text{otherwise.} \end{cases} \tag{17}$$

D. Revisiting the $(N, K) = (2, 4)$ Example

Let us revisit the example code for the $(2, 4)$ case within the context of the general caching scheme. The two indexing methods now have the following mapping

$$A_1 \to W_{1,\{1,2\}}, \quad A_2 \to W_{1,\{1,3\}}, \quad A_3 \to W_{1,\{1,4\}},$$
$$A_4 \to W_{1,\{2,3\}}, \quad A_5 \to W_{1,\{2,4\}}, \quad A_6 \to W_{1,\{3,4\}},$$

and similarly for file segments of file B.

The scheme presented earlier is for $t = 2$. Let us consider replacing the coding coefficients with that of a rank metric code. The parameters can be determined as $P_0 = 10$ and $P = 6$, and thus $P_0 - P = 4$ symbols are generated and cached at each user. It is also sufficient to choose $q = 7$ and $m = 10$; as such, each information and coded symbol is in $\mathbb{F}_{7^{10}}$, which is considerably larger than $\mathbb{F}_5$ used in the example.

Now consider requests (A, A, A, B), for which $m_1 = 3$ and $m_2 = 1$. It is clear that the uncoded transmissions in the general scheme match exactly what we have presented. Next consider the transmission in Step 2 for $W_1 = A$, $\mathcal{A} = \{4\}$ for which we have

$$\mathcal{W}_{1,\{4\}} = \{W_{1,\{1,4\}}, W_{1,\{2,4\}}, W_{1,\{3,4\}}\} = \{A_3, A_5, A_6\}, \tag{18}$$

and the parities of a $(2\binom{3}{1} - \binom{2}{0}, \binom{3}{1})) = (5, 3)$ MDS code, whose coefficients are in $\mathbb{F}_7$, are transmitted; one choice is exactly as that given previously, *i.e.*, the symbols $(A_3 + 2A_5 + 3A_6, A_3 + 3A_5 + 4A_6)$. In Step 3, we have the following segments

$$\mathcal{W}_{1,\emptyset} = \{W_{1,\{1,2\}}, W_{1,\{1,3\}}, W_{1,\{2,3\}}\} = \{A_1, A_2, A_4\}, \tag{19}$$

and the parity symbol of a $(2\binom{3}{2} - \binom{2}{1}, \binom{3}{2}) = (4, 3)$ MDS code, whose coefficients are in $\mathbb{F}_7$, is transmitted; one choice is exactly as that given previously, *i.e.*, the symbol $A_1 + A_2 + A_4$. For file $W_2 = B$, we can only take $|\mathcal{A}| = 1$ in Step 2 since $\max(1, t - m_2) = \min(t - 1, 4 - 1) = 1$, however in this case, a $(2\binom{1}{1} - \binom{0}{0}, \binom{1}{1}) = (1, 1)$ MDS code does not have any parity symbols, and thus no transmission of file B is required in Step 2; there is also no transmission of file B in Step 3.

We can similarly walk through the example for $(N, K) = (3, 4)$ using the general transmission strategy; this simple exercise is left to interested readers.

VI. Proof of the Main Theorem

We establish the correctness and the performance of the caching scheme in three propositions, and Theorem 1 follows directly from them. Two related issues are then discussed, regarding the format of the cached linear combinations and the required field size of the code.

A. Correctness

Proposition 1: For any $t \in \{1, 2, \ldots, K-1\}$, the afore-given placement strategy can be used to satisfy any demands that request all files with the afore-given transmission strategy.

Proof: To show that any demands that request all N files can be satisfied, we need to consider any single user. Without loss of generality, we can consider the first user and assume it requests file W_1. Let us count the number of linear combinations he receives which consist of interference symbols in his cache in the first two transmission steps.

In Step 1, user 1 can collect all uncoded symbols for file W_n, $n = 2, 3, \ldots, N$, in the form

$$\{W_{n,\mathcal{S}} : 1 \in \mathcal{S} \subseteq \bar{I}^{[n]}\}, \tag{20}$$

and there are a total of

$$\tilde{T}^{(1)} = \sum_{n=2}^{N} \binom{K - m_n - 1}{t - 1} \tag{21}$$

such symbols, where we have taken the convention $\binom{n}{k} = 0$ when $n < k$.

In Step 2, user 1 collects linear combinations of W_n, $n = 2, 3, \ldots, N$, however only those in the following form. For each such n, and each subset $\mathcal{A} \subseteq \bar{I}^{[n]}$ such that $\max(1, t - m_n) \leq |\mathcal{A}| \leq \min(t-1, K - m_n)$ and moreover $1 \in \mathcal{A}$, user 1 collects the parity symbols of encoding $\mathcal{W}_{n,\mathcal{A}}$ using the systematic MDS code. Thus user 1 collects a total of

$$\tilde{T}^{(2)} = \sum_{n=2}^{N} \sum_{j=\max(1,t-m_n)}^{\min(t-1,K-m_n)} \binom{K-m_n-1}{j-1}\binom{m_n-1}{t-j} \tag{22}$$

such symbols.

User 1 now has collected $\tilde{T}^{(1)} + \tilde{T}^{(2)}$ useful symbols, and has in his cache $P_o - P$ symbols of the same basis. Observe for the summands in $\tilde{T}^{(1)}$ and $\tilde{T}^{(2)}$, we have

$$\sum_{j=\max(1,t-m_n)}^{\min(t-1,K-m_n)} \binom{K-m_n-1}{j-1}\binom{m_n-1}{t-j} + \binom{K-m_n-1}{t-1} = \binom{K-2}{t-1}. \tag{23}$$

To see this equality holds, first consider the case $1 > t - m_n$, i.e., $m_n \geq t$, for which the left hand side is simply all the possible ways of choosing $t-1$ balls in a total of $K-2$ balls, however counted when these balls are partitioned into two groups of size $K-2-(m_n-1)$ and $m_n - 1$, respectively; for the other case $1 \leq t - m_n$, i.e., $t \geq m_n + 1$, the left hand side reduces to

$$\sum_{j=t-m_n+1}^{\min(t-1,K-m_n)} \binom{K-m_n-1}{j-1}\binom{m_n-1}{t-j} + \binom{K-m_n-1}{t-1} \tag{24}$$

which again clearly gives the same result using the same ball counting argument. It follows that

$$\tilde{T}^{(1)} + \tilde{T}^{(2)} + P_o - P = P. \tag{25}$$

These P linear combinations can be represented as the product of the length-P_o codeword (both systematic and parity symbols) of the rank metric code $\mathcal{C}(P_o, P)$ and a matrix G of size $P_o \times P$. Recall the systematic rank metric code we used to encode the P file segments in user 1's cache, and by Lemma 2, as long as the matrix G is full rank, all the P segments can be recovered. This fact is proved in the appendix, but an outline of the proof is given here. We recognize that if the columns and rows of the matrix G are rearranged to

- Group the file segments $W_{1,\mathcal{S}}$ in user 1's cache together;
- For each $n = 2, 3, \ldots, N$, group the segments of $\{W_{n,\mathcal{S}} : 1 \in \mathcal{S} \subseteq \bar{I}^{[n]}\}$ together;
- For each $n = 2, 3, \ldots, N$, and for each subset $\mathcal{A} \subseteq \bar{I}^{[n]}$ such that $\max(1, t-m_n) \leq |\mathcal{A}| \leq \min(t-1, K-m_n)$ and moreover $1 \in \mathcal{A}$, group the segments of $\mathcal{W}_{n,\mathcal{A}}$ together,

then the resulting matrix is block diagonal, and each block is either of size 1×1 with entry 1 or full rank because they are columns of generator matrices of MDS codes. Thus the matrix G is indeed full rank.

Thus user 1 can eliminate the interference in its cached contents, and recover all the file segments of $W_{1,\mathcal{S}}$ that are already present in its cache. It remains to show that all the file segments $W_{1,\mathcal{S}}$ that are not present in his cache can also be recovered.

First, observe that in Step 1, user 1 can collect all uncoded W_1 file segments that are not in the cache of any users $k \in I^{[1]}$, *i.e.,* $\{W_{1,\mathcal{S}} : 1 \notin \mathcal{S} \subseteq \bar{I}^{[1]}\}$. As mentioned earlier, in Step 2 after eliminating the interference, user 1 can recover all $W_{1,\mathcal{S}}$ for $\mathcal{S}$ such that $1 \in \mathcal{S}$. Furthermore, for each subset $\mathcal{A} \subseteq \bar{I}^{[1]}$ such that $\max(1, t - m_1) \leq |\mathcal{A}| \leq \min(t-1, K - m_1)$, user 1 can collect the parity symbols of encoding $\mathcal{W}_{1,\mathcal{A}}$ using the $\left(2\binom{m_1}{t-|\mathcal{A}|} - \binom{m_1-1}{t-|\mathcal{A}|-1}, \binom{m_1}{t-|\mathcal{A}|}\right)$ systematic MDS code. Since user 1 has in its cache $\binom{m_1-1}{t-|\mathcal{A}|-1}$ of the total $\binom{m_1}{t-|\mathcal{A}|}$ symbols of $\mathcal{W}_{1,\mathcal{A}}$, together with the collected parity symbols, he can recover all $\binom{m_1}{t-|\mathcal{A}|}$ symbols in this set. Thus after Step 2, user 1 can also recover all file segments $W_{1,\mathcal{S}}$ where $\mathcal{S}$ has elements in both $I^{[1]}$ and $\bar{I}^{[1]}$. The only missing segments are a subset of $\{W_{1,\mathcal{S}} : 1 \notin \mathcal{S} \subseteq I^{[1]}\}$. However, Step 3 transmits the parities of a $\left(2\binom{m_1}{t} - \binom{m_1-1}{t-1}, \binom{m_1}{t}\right)$ MDS code that encodes all $\{W_{1,\mathcal{S}} : \mathcal{S} \subseteq I^{[1]}\}$, and since user 1 already has $\binom{m_1-1}{t-1}$ elements, he can thus also recover the rest of the symbols in this set. At this point, we can conclude that user 1 can recover all file segments of W_1, which completes the proof. ∎

Proposition 2: For any $t \in \{1, 2, \ldots, K-1\}$, the afore-given placement strategy can be used to satisfy any demands that request a strict subset of all the files with the afore-given transmission strategy.

The proof of this proposition can be intuitively explained as follows. When we replace a file demand W_i in the enhanced demands with a demand W_j, the effect of not transmitting the file segments involving W_i in the first three steps needs to compensated. In order to do so, let us examine the roles that these W_i transmissions play: firstly, they are used to eliminate the interference by W_i at certain other users, and secondly, they are used to provide the missing segments to the single user that was requesting W_i in the enhanced demands. Our strategy is to transmit the corresponding segments from W_j instead of sending the segments from W_i. With such substituted transmissions, the first role can be fulfilled as long as it is not a redundant transmission, and we rely on the counter $\tau_{n,\mathcal{A}}$ to avoid any such redundancy. The second role can clearly also be fulfilled by any such non redundant transmissions. When a transmission of the file segment from W_j is indeed redundant, we can safely conclude that the second role has already been fulfilled by previous transmissions, and thus transmitting this segment of W_i is now sufficient to serve the first role alone. The proof below makes this intuition more rigorous.

Proof: Without loss of generality, we only need to consider the first user and assume his request is for file W_1. Two cases need to be examined: the first case is when in the enhanced demands, the first user was also requesting file W_1; the second case is when in the enhanced demands, the first user was requesting n^*, *i.e.,* $f(n^*) = 1$ and $u(n^*) = 1$, for some $n^* \in \{N^* + 1, \ldots, N\}$.

Let us consider the proof for the first case, which is similar to the proof for the Proposition 1. In Step 1, user 1 collects all

uncoded symbols for file W_n, $n = 2, 3, \ldots, N^*$, in the form

$$\{W_{n,\mathcal{S}} : 1 \in \mathcal{S} \subseteq \bar{I}^{[n]}\}, \tag{26}$$

and there are a total of

$$\tilde{\bar{T}}^{(1)} = \sum_{n=2}^{N^*} \binom{K - \dot{m}_n - 1}{t - 1} \tag{27}$$

such symbols.

In Step 2, user 1 collects linear combinations of W_n, $n = 2, 3, \ldots, N^*$, however only those in the following form. For each such n, and each subset $\mathcal{A} \subseteq \bar{I}^{[n]}$ such that $\max(1, t - \dot{m}_n) \leq |\mathcal{A}| \leq \min(t - 1, K - \dot{m}_n)$ and moreover $1 \in \mathcal{A}$, user 1 collects the parity symbols of encoding $\mathcal{W}_{n,\mathcal{A}}$ using the systematic MDS code. Thus user 1 collects a total of

$$\tilde{\bar{T}}^{(2)} = \sum_{n=2}^{N^*} \sum_{j=\max(1,t-\dot{m}_n)}^{\min(t-1,K-\dot{m}_n)} \binom{K - \dot{m}_n - 1}{j - 1} \binom{\dot{m}_n - 1}{t - j} \tag{28}$$

such symbols.

In Step 4, user 1 collects for each $n = N^* + 1, \ldots, N$, for any $\mathcal{A} \subseteq \bar{I}^{[n]}$ where $|\mathcal{A}| = t - 1$ and $1 \in \mathcal{A}$, either $W_{n,\mathcal{A}\cup\{u(n)\}}$ or $W_{f(n),\mathcal{A}\cup\{u(n)\}}$, whichever was transmitted in Step 4. Note that in this case $u(n) \neq 1$ for any $n = N^* + 1, \ldots, N$, which implies that $|\mathcal{A} \cup \{u(n)\}| = t$. Thus user 1 collects another total of

$$\tilde{\bar{T}}^{(4)} = (N - N^*) \binom{K - 2}{t - 1} \tag{29}$$

uncoded symbols.

User 1 now has collected $\tilde{\bar{T}}^{(1)} + \tilde{\bar{T}}^{(2)} + \tilde{\bar{T}}^{(4)}$ useful symbols, and has in his cache $P_o - P$ symbols of the same basis. It is seen that

$$\tilde{T}^{(1)} + \tilde{T}^{(2)} + \tilde{T}^{(4)} + P_o - P = P. \tag{30}$$

These P linear combinations, which can again be represented as the product of the length-P_o output (both systematic and parity symbols) of the rank metric code $\mathcal{C}(P_o, P)$ and a matrix G^* of size $P_o \times P$. As long as the matrix G^* is full rank, user 1 can recover all the file segments $W_{1,\mathcal{S}}$ where $1 \in \mathcal{S}$, and the rest of file segments from W_1 can be recovered as in the case of Proposition 1. The fact of the matrix G^* being full rank is obvious for the similar reason that the G matrix is full rank under the enhanced demands; in fact, since the transmissions in Step 4 are all uncoded, the full-rank property directly follows from the full-rank property of the corresponding generator matrix of the MDS codes when encoding the file segments of files $W_{f(n)}$, $n = N^*+1, \ldots, N$. User 1 can indeed recover any missing file segments, since the transmissions for the enhanced demands for $n = 1, \ldots, N^*$ in the first three steps guarantee the completion of this task.

Now let us now consider the second case, where user 1 is demanding W_1, but in the enhanced demands, he was requesting file n^* for some $n^* \in \{N^* + 1, \ldots, N\}$. By a similar argument as above, user 1 can recover all segments $W_{1,\mathcal{S}}$ present in his cache, *i.e.,* for $W_{1,\mathcal{S}}$ where $1 \in \mathcal{S}$, by eliminating the interference. More precisely, in Step 1, user 1 collects all uncoded symbols for file W_n, $1, 2, \ldots, N^*$, in the form

$$\{W_{n,\mathcal{S}} : 1 \in \mathcal{S} \subseteq \bar{I}^{[n]}\}, \tag{31}$$

and there are a total of

$$\tilde{\bar{T}}^{(1')} = \sum_{n=1}^{N^*} \binom{K - \dot{m}_n - 1}{t - 1} \tag{32}$$

such symbols. In Step 2, user 1 collects linear combinations of W_n, $n = 1, 2, \ldots, N^*$, however only those in the following form. For each such n, and each subset $\mathcal{A} \subseteq \bar{I}^{[n]}$ such that $\max(1, t - \dot{m}_n) \leq |\mathcal{A}| \leq \min(t - 1, K - \dot{m}_n)$ and moreover $1 \in \mathcal{A}$, user 1 collects the parity symbols of encoding $\mathcal{W}_{n,\mathcal{A}}$ using the systematic MDS code. Thus user 1 collects a total of

$$\tilde{\bar{T}}^{(2')} = \sum_{n=1}^{N^*} \sum_{j=\max(1,t-\dot{m}_n)}^{\min(t-1,K-\dot{m}_n)} \binom{K - \dot{m}_n - 1}{j - 1} \binom{\dot{m}_n - 1}{t - j} \tag{33}$$

such symbols. In Step 4, user 1 collects for each $n = N^* + 1, \ldots, n^*-1, n^*+1, \ldots, N$, for any $\mathcal{A} \subseteq \bar{I}^{[n]}$ where $|\mathcal{A}| = t - 1$ and $1 \in \mathcal{A}$, either $W_{n,\mathcal{A}\cup\{u(n)\}}$ or $W_{f(n),\mathcal{A}\cup\{u(n)\}}$, whichever was transmitted in Step 4. Thus user 1 collects another total of

$$\tilde{\bar{T}}^{(4')} = (N - N^* - 1) \binom{K - 2}{t - 1} \tag{34}$$

uncoded symbols. User 1 now has collected $\tilde{\bar{T}}^{(1)} + \tilde{\bar{T}}^{(2)} + \tilde{\bar{T}}^{(4)}$ useful symbols, and has in his cache $P_o - P$ symbols of the same basis. It is seen that

$$\tilde{T}^{(1')} + \tilde{T}^{(2')} + \tilde{T}^{(4')} + P_o - P = P. \tag{35}$$

It only remains to show that for the second case, the transmissions in Step 4 suffice to provide any missing segments of W_1 in user 1's cache, possibly jointly with transmissions from W_1 in the first three steps. This is rather straightforward, since all file segments $W_{1,\mathcal{S}}$'s with $1 \notin \mathcal{S}$ are transmitted uncoded in Step 4, unless $\tau_{1,\mathcal{A}_{j[1]}(\mathcal{S})} < 0$; when the latter scenario occurs, a total of $\binom{\dot{m}_1 - 1}{t - |\mathcal{A}_{j[1]}(\mathcal{S})| - 1}$ uncoded symbols have already been transmitted in the set $\mathcal{W}_{1,\mathcal{A}_{j[1]}(\mathcal{S})}$, and together with the $\binom{\dot{m}_n - 1}{t - |\mathcal{A}_{j[1]}(\mathcal{S})|}$ parity symbols encoding the set $\mathcal{W}_{1,\mathcal{A}_{j[1]}(\mathcal{S})}$ which were transmitted in the first three steps, user 1 can indeed recover all $\binom{\dot{m}_n}{t - |\mathcal{A}_{j[1]}(\mathcal{S})|}$ symbols in $\mathcal{W}_{1,\mathcal{A}_{j[1]}(\mathcal{S})}$. Thus user 1 is able to recover all segments of W_1, and the proof is complete. ∎

B. Performance

Proposition 3: For any $t \in \{1, 2, \ldots, K - 1\}$, the aforegiven placement strategy and transmission strategy achieve the memory-transmission pair

$$(M, R) = \left(\frac{t[(N - 1)t + K - N]}{K(K - 1)}, \frac{N(K - t)}{K} \right). \tag{36}$$

Proof: Recall each file of unit size is partition into $\binom{K}{t}$ segment symbols, and each user caches $P_o - P$ symbols,

and thus the memory usage is straightforwardly to calculate. It remains to calculate the total number of transmitted symbols.

We only need to consider the first three steps of transmission when all files are being requested, since for the other cases where only a subset of files are requested, each transmission in Step 4 corresponds to exactly one transmission in Step 3 for the enhanced demands, and thus the rate remains the same as for the case of the enhanced demands.

Clearly, in Step 1, the total number of transmitted uncoded symbols of file W_n is

$$T_n^{(1)} = \binom{K - m_n}{t}. \tag{37}$$

In Step 2, the total number of transmitted linear combinations of file W_n is given as

$$T_n^{(2)} = \sum_{j=\max(1,t-m_n)}^{\min(t-1,K-m_n)} \binom{K - m_n}{j}\binom{m_n - 1}{t - j}.$$

In Step 3, the total number of transmitted linear combinations of file W_n is given as

$$T_n^{(3)} = \binom{m_n - 1}{t}.$$

Note that

$$T_n^{(1)} + T_n^{(2)} + T_n^{(3)} = \binom{K - 1}{t},$$

because it is all the ways to choose t balls in a total of $K - 1$ balls. Thus the total transmissions amount to $N\binom{K-1}{t}$ symbols. The proof can now be completed with a simple normalization by the number of segments in each file. ∎

C. The Semi-Systematic Variant of the Placement Strategy

The general placement strategy we provide does not enforce any special structure on the linear combinations, unlike the code given in the $(2, 4)$ example. However, even for general parameters (N, K) and the same range of parameter t, we can indeed choose to use the semi-systematic format. More precisely, the first $\binom{K-1}{t-1}$ semi-systematic symbols in the cache of user k are

$$\sum_{n=1}^{N} W_{n,\mathcal{S}}, \quad k \in \mathcal{S}, \tag{38}$$

where the addition is in finite field $\mathbb{F}_{q^m}$. Moreover, we use the same parameter P, but choose

$$P'_o = \binom{K-1}{t-1}(2N-1) - \binom{K-2}{t-1}(N-1), \tag{39}$$

and construct a (P'_o, P) systematic rank metric code, which is denoted as $\mathcal{C}(P'_o, P)$. The local parity symbols stored in user k's cache are the parity symbols when encoding the set of file segment symbols $\{W_{n,\mathcal{S}} : n = 1, 2, \ldots, N, k \in \mathcal{S}\}$ using $\mathcal{C}(P'_o, P)$. The transmission strategy remains the same.

In order to prove the correctness of this placement variant, we only need to show that the corresponding matrix G', similarly as in the proof of Proposition 1, is also full rank. This is again rather immediate. Since the only difference is the columns corresponding to the semi-systematic symbols in the cache. However, it is easily seen that although the matrix G' is no longer block diagonal after the rearrangement of columns and rows, the new columns has non-zero entries on rows corresponding to $W_{1,\mathcal{S}}$ (in fact it has an identity matrix if we restrict it to these columns and rows with proper row and column indexing), while no other columns in G' have non-zero entries on these rows. Thus indeed this variant of placement strategy is also valid; a more precise proof is given in the appendix.

We choose to present the general construction in the last section instead of this variant directly in order to emphasize the fact that the semi-systematic format is not fundamentally important in our construction. Note that in the semi-systematic variant, the bound on the parameter m can be made smaller, since the parameters of the rank metric code are reduced: choosing $m \geq P'_o$ suffices here.

D. Reducing the Field Size With Generic Linear Codes

In the proposed code construction, we rely on rank metric codes to guarantee certain full rank properties, and the overall code design problem essentially reduces to a rank counting problem on the proper basis. However, one obvious disadvantage of using rank metric codes in the construction is that the size of the field $\mathbb{F}_{q^m}$ needs to be quite large. We can in fact replace the rank metric code with a generic systematic linear code, and directly require the full rank properties to hold. In this section, we provide such a simple argument and show that a reduced field size is sufficient.

Let us consider the cache encoding for the k-th user. A total of P symbols are present at this user, and a total of $P_o - P$ parity symbols are generated during the encoding. In this subsection, we shall assume that the entries of this $P \times (P_o - P)$ encoding matrix are from $\mathbb{F}_q$, *i.e.,* the same finite field as the set of MDS codes. Denote this matrix as G_k, and its entry on the i-th row and j-th column as $g_{k,i,j}$, which is to be determined; note that this code is not necessarily a rank-metric code any longer.

Consider a specific set of demands $(d_1, d_2, \ldots, d_K)$, (*i.e.*, the k-th user demands file d_k), where all files are requested. In the delivery phase, the symbols user-k collects during Step 1 and Step 2 are linear combinations of all the symbols present at this user. This can be represented also by a $P \times (\tilde{T}^{(1)} + \tilde{T}^{(2)})$ encoding matrix $G'_{k,(d_1,d_2,\ldots,d_K)}$. The full rank condition in the proof of Proposition 1 essentially requires that the $P \times P$ matrix $[G_k, G'_{k,(d_1,d_2,\ldots,d_K)}]$ being full rank. The determinant of the matrix $[G_k, G'_{k,(d_1,d_2,\ldots,d_K)}]$ can be expressed as a function of the coefficients $g_{k,i,j}$'s, *i.e.*

$$\det(G_n, G'_{n,(d_1,d_2,\ldots,d_K)}) = f_{n,(d_1,d_2,\ldots,d_K)}(\{g_{n,i,j}\}).$$

By the proof of Proposition 2, the full rank condition for demands where only a subset of the files are requested is implied by the full rank condition for the enhanced demands. Thus as long as the following polynomial has a non-zero solution, then the choice of coefficients $\{g_{k,i,j}\}$ is

valid

$$\prod_{k=1}^{K} \prod_{\substack{(d_1,d_2,\ldots,d_K):\\ \text{all files requested}}} f_{k,(d_1,d_2,\ldots,d_K)}(\{g_{k,i,j}\}). \tag{40}$$

We can now invoke the following lemma.

Lemma 3 [19] (Combinatorial Nullstellansatz): Let $\mathbb{F}$ be a field, and let $f = f(x_1, \cdots, x_n)$ be a polynomial in $\mathbb{F}[x_1, \cdots, x_n]$. Suppose the degree $deg(f)$ of f is expressible in the form $\sum_{i=1}^{n} t_i$, where each t_i is a non-negative integer and suppose that the coefficient of the monomial term $\prod_{i=1}^{n} x_i^{t_i}$ is nonzero. Then if $S_1, \ldots, S_n$ are subsets of $\mathbb{F}$ with sizes $|S_i|$ satisfying $|S_i| > t_i$, then there exist elements $s_1 \in S_1, s_2 \in S_2 \ldots, s_n \in S_n$ such that $f(s_1, s_2, \cdots, s_n) \neq 0$.

In this lemma above, the condition that the coefficient of the monomial term $\prod_{i=1}^{n} x_i^{t_i}$ is nonzero is equivalent to requiring $f = f(x_1, \cdots, x_n)$ to be not identically zero. We note that $f_{n,(d_1,d_2,\ldots,d_K)}(\{g_{n,i,j}\})$ is indeed not identically zero, because the code construction based on rank metric codes directly provides a non-zero assignment.

Since the degree of any indeterminate in each of $f_{k,(d_1,d_2,\ldots,d_K)}(\{g_{n,i,j}\})$ is 1, the maximum among the degrees of a single indeterminate of the polynomial (40) is upper bounded by the total number of demands where all files are requested, which is given by $S(K, N)N!$. Here

$$S(K, N) = \frac{1}{N!} \sum_{j=1}^{N} (-1)^{N-j} \binom{N}{j} j^K, \tag{41}$$

is the Sterling number of the second kind [20], which counts the number of ways to partition a set of K objects into N non-empty subsets. Hence by Lemma 3, it is possible to find a suitable assignment for $\{g_{n,i,j}\}$, if the entries are picked from a finite field $\mathbb{F}_q$ with $q > S(K, N)N!$. Alternatively, we can simply count the total number of demands, instead of those where all files are requested, and this leads to a looser bound of N^K on the field size. In contrast, recall the sufficient condition on the field size given in Section V when rank metric codes are used, which in the best case of $t = 2$ would result in a field size of order $O([(K - N + 1)(K - N)]^{2(K-1)N})$. This large difference suggests that there is significant potential to find alternative code constructions in a much smaller finite field, which is part of our ongoing work.

VII. Conclusion

We proposed a new coding scheme for the caching problem when $N \leq K$, based on a combination of rank metric codes and MDS codes. The performance of the scheme has a particularly simple form, and it provides new memory-rate tradeoff points beyond what were known in the literature. Compared to known coded caching schemes, the proposed scheme uses coding for both placement and delivery, as well as larger finite field instead of the binary field.

The file segments in a user's cache from files that are not being requested by the user can be viewed as interference in the delivery phase, and in the proposed scheme, they are resolved completely. We believe other advanced interference alignment techniques are also applicable in the caching problem, and this is part of our ongoing research. Another immediate variation of the proposed scheme is its decentralized counterpart, motivated by the investigation of the decentralized caching scheme [2], which is a variation of the centralized caching scheme in [1].

After the initial publication of the codes proposed in this work [21], more codes have been discovered for the caching problem, the most notable ones among which are perhaps [22] and [23]. Nevertheless, the codes given in this work still provide the best performance at the low memory regime as discussed in [23].

Appendix
Full Rank of Matrix G and G'

The key to the proof is to express the matrix G of size $P_o \times P$ in a more structured manner. For this purpose, let us again consider the cache and decoding process at user 1. First rearrange the systematic and parity symbols of the code $\mathcal{C}(P_o, P)$, such that the $P_o - P$ cached symbol are indexed in the set I_{P_o-P}; similarly we arrange the columns of G such that its first $P_o - P$ columns correspond to these cached symbols. The next rows and columns correspond to the symbols that user 1 collected during the Step 1 transmission

$$\{W_{n,\mathcal{S}} : 1 \in \mathcal{S} \subseteq \bar{I}^{[n]}\}, \quad n = 2, 3, \ldots, N,$$

and there are a total of $\sum_{n=2}^{N} \tilde{T}_n^{(1)}$ such symbols.

The next rows and columns correspond to a fixed $n \in \{2, 3, \ldots, N\}$ and a fixed subset $\mathcal{A} \subseteq \bar{I}^{[n]}$ where $\max(1, t - m_n) \leq |\mathcal{A}| \leq \min(t - 1, K - m_n)$ and moreover $1 \in \mathcal{A}$. Denote the parity check portion of generator matrix of the $\left(2\binom{m_n}{t-|\mathcal{A}|} - \binom{m_n-1}{t-|\mathcal{A}|-1}, \binom{m_n}{t-|\mathcal{A}|}\right)$ systematic MDS as $Q_{n,\mathcal{A}}$, which has dimension $\binom{m_n}{t-|\mathcal{A}|} \times \left[\binom{m_n}{t-|\mathcal{A}|} - \binom{m_n-1}{t-|\mathcal{A}|-1}\right]$, and it is full rank since it is part of a generator matrix of an MDS code and it has less columns than rows.

Now the matrix G can be written in the following form

$$G = \begin{bmatrix} I & & & & \\ & Q_{2,\mathcal{A}_{2,1}} & & & \\ & & Q_{2,\mathcal{A}_{2,2}} & & \\ & & & \cdots & \\ & & & & Q_{N,\mathcal{A}_{N,L_N}} \end{bmatrix} \tag{42}$$

where the identity matrix at the top-left has dimension $(P_o - P + \sum_{n=2}^{N} \tilde{T}_n^{(1)}) \times (P_o - P + \sum_{n=2}^{N} \tilde{T}_n^{(1)})$, and we have enumerated the aforementioned matrix $\mathcal{A}$'s for each n by using the subscript as $\mathcal{A}_{n,\ell}$, and L_N is the total number of such subsets $\mathcal{A}$ when $n = N$. It is now clear that the matrix G is block diagonal and each block is full rank, and thus G indeed has full rank.

For the semi-systematic variant of the caching scheme, the matrix G' is slightly different. First index the symbols

$$\{W_{1,\mathcal{S}} : 1 \in \mathcal{S}\} \tag{43}$$

using the set $I_{\binom{K-1}{t-1}}$, and rearrange the columns and rows of G' such that they correspond to the top $\binom{K-1}{t-1} \times \binom{K-1}{t-1}$ submatrix using the same order. Next rearrange the systematic and parity

symbols of the code $\mathcal{C}(P'_o, P)$, such that the $P'_o - P$ cached symbol correspond to the next $P'_o - P$ columns and rows. The rest of the G' matrix is arranged exactly as for the case G. It is now clear that the matrix G' has the following form

$$G' = \begin{bmatrix} I_a & & & & & \\ & I_b & & & & \\ F_{2,\mathcal{A}_{2,1}} & & Q_{2,\mathcal{A}_{2,1}} & & & \\ F_{2,\mathcal{A}_{2,2}} & & & Q_{2,\mathcal{A}_{2,2}} & & \\ & & & & \cdots & \\ F_{N,\mathcal{A}_{N,L_N}} & & & & & Q_{N,\mathcal{A}_{N,L_N}} \end{bmatrix} \tag{44}$$

where the identity matrix I_a is of dimension $\binom{K-1}{t-1} \times \binom{K-1}{t-1}$, and the identity matrix I_b is of dimension $(P'_o - P + \sum_{n=2}^{N} \tilde{T}_n^{(1)}) \times (P'_o - P + \sum_{n=2}^{N} \tilde{T}_n^{(1)})$, and the $F_{n,\mathcal{A}_{n,\ell}}$ matrices have some nonzero entries but their exact forms are not important here; the other off block-diagonal entries are all zeros. It is now clear that the matrix G' also has full rank. □

REFERENCES

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

[2] M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 1029–1040, Aug. 2015.

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

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

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

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

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

[8] C. Tian, "Symmetry, demand types and outer bounds in caching systems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2016, pp. 825–829.

[9] C. Tian, "Characterizing the rate region of the (4, 3, 3) exact-repair regenerating codes," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 967–975, May 2014.

[10] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol, "Index coding with side information," *IEEE Trans. Inf. Theory*, vol. 57, no. 3, pp. 1479–1494, Mar. 2011.

[11] S. Wicker, *Error Control Systems for Digital Communication and Storage*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1995.

[12] È. M. Gabidulin, "Theory of codes with maximum rank distance," *Problemy Peredachi Inform.*, vol. 21, no. 1, pp. 3–16, 1985.

[13] R. Lidl and H. Niederreiter, *Finite Fields* (Encyclopedia of Mathematics and Its Applications). Cambridge, U.K.: Cambridge Univ. Press, 1997.

[14] R. Koetter and F. R. Kschischang, "Coding for errors and erasures in random network coding," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3579–3591, Aug. 2008.

[15] N. Silberstein, A. S. Rawat, and S. Vishwanath, "Error-correcting regenerating and locally repairable codes via rank-metric codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 11, pp. 5765–5778, Nov. 2015.

[16] C. Tian, B. Sasidharan, V. Aggarwal, P. V. Kumar, and V. A. Vaishampayan, "Layered exact-repair regenerating codes via embedded error correction and block designs," *IEEE Trans. Inf. Theory*, vol. 61, no. 4, pp. 1933–1947, Apr. 2015.

[17] T. Ho *et al.*, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[18] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam, The Netherlands: North Holland, 1977.

[19] N. Alon, "Combinatorial nullstellensatz," *Combinat., Probab. Comput.*, vol. 8, nos. 1–2, pp. 7–29, 1999.

[20] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics*. Reading, MA, USA: Addison-Wesley, 1988.

[21] C. Tian and J. Chen, "Caching and delivery via interference elimination," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 830–834.

[22] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "The exact rate-memory tradeoff for caching with uncoded prefetching," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Dec. 2017, pp. 1613–1617.

[23] J. Gómez-Vilardebó. (Dec. 2016). "Fundamental limits of caching: Improved bounds with coded prefetching." [Online]. Available: https://arxiv.org/abs/1612.09071

Chao Tian (S'00–M'05–SM'12) received the B.E. degree in Electronic Engineering from Tsinghua University, Beijing, China, in 2000 and the M.S. and Ph. D. degrees in Electrical and Computer Engineering from Cornell University, Ithaca, NY in 2003 and 2005, respectively. Dr. Tian was a postdoctoral researcher at Ecole Polytechnique Federale de Lausanne (EPFL) from 2005 to 2007, a member of technical staff–research at AT&T Labs–Research in New Jersey from 2007 to 2014, and an Associate Professor in the Department of Electrical Engineering and Computer Science at the University of Tennessee Knoxville from 2014 to 2017. He joined the Department of Electrical and Computer Engineering at Texas A&M University as an Associate Professor in 2017. His research interests include data storage systems, multi-user information theory, joint source-channel coding, signal processing, and compute algorithms.

Dr. Tian received the Liu Memorial Award at Cornell University in 2004, AT&T Key Contributor Award in 2010, 2011 and 2013, and 2014 IEEE ComSoc DSTC Data Storage Best Paper Award. He was an Associate Editor for the IEEE SIGNAL PROCESSING LETTERS from 2012 to 2014, and is currently an Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS.

Jun Chen (S'03–M'06–SM'16) received the B.E. degree with honors in communication engineering from Shanghai Jiao Tong University, Shanghai, China, in 2001 and the M.S. and Ph.D. degrees in electrical and computer engineering from Cornell University, Ithaca, NY, in 2004 and 2006, respectively.

He was a Postdoctoral Research Associate in the Coordinated Science Laboratory at the University of Illinois at Urbana-Champaign, Urbana, IL, from September 2005 to July 2006, and a Postdoctoral Fellow at the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, from July 2006 to August 2007. Since September 2007 he has been with the Department of Electrical and Computer Engineering at McMaster University, Hamilton, ON, Canada, where he is currently an Associate Professor and a Joseph Ip Distinguished Engineering Fellow. His research interests include information theory, wireless communications, and signal processing.

He received several awards for his research, including the Josef Raviv Memorial Postdoctoral Fellowship in 2006, the Early Researcher Award from the Province of Ontario in 2010, and the IBM Faculty Award in 2010. He served as an Associate Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY from 2014 to 2016.