2016

Coded Caching with Low Subpacketization Levels

Li Tang
Iowa State University, litang@iastate.edu

Aditya Ramamoorthy
Iowa State University, adityar@iastate.edu

Coded Caching with Low Subpacketization Levels

Li Tang and Aditya Ramamoorthy
Department of Electrical and Computer Engineering
Iowa State University
Ames, IA 50010
Emails:{litang, adityar}@iastate.edu

***Abstract*—Caching is popular technique in content delivery networks that allows for reductions in transmission rates from the content-hosting server to the end users. Coded caching is a generalization of conventional caching that considers the possibility of coding in the caches and transmitting coded signals from the server. Prior results in this area demonstrate that huge reductions in transmission rates are possible and this makes coded caching an attractive option for the next generation of content-delivery networks. However, these results require that each file hosted in the server be partitioned into a large number (i.e., the subpacketization level) of non-overlapping subfiles. From a practical perspective, this is problematic as it means that prior schemes are only applicable when the size of the files is extremely large. In this work, we propose a novel coded caching scheme that enjoys a significantly lower subpacketization level than prior schemes, while only suffering a marginal increase in the transmission rate. In particular, for a fixed cache size, the scaling with the number of users is such that the increase in transmission rate is negligible, but the decrease in subpacketization level is exponential.**

I. Introduction

Fast and efficient content delivery over the Internet is an important problem and is the core business of companies such as Akamai (which is estimated to serve 15-30% of all Web traffic). Crucial to Akamai's approach is a large network of servers that cache popular content closer to the end users. This serves to significantly reduce delivery time and improve the end user's experience. Traditional caching operates by storing popular content (or portions thereof) closer to or at the end user. Typically a cache serves a user request partially (or sometimes entirely) with the remainder of the content coming from the server.

Prior work in this area [1] demonstrates that allowing coding in the cache and coded transmission from the server (referred to as *coded caching*) to the end users can allow for huge reductions in the number of bits transmitted from the server to the end users. This is an exciting development given the central role of caching in supporting a significant fraction of Web traffic.

Reference [1] considered a scenario where a single server containing N files of size F bits connects to K users over a shared link and each user has a cache memory MF bits. Coded caching consists of two distinct phases: a *placement phase* and a *delivery phase*. In the placement phase, the user caches are populated. This phase does not depend on the user demands which are assumed to be arbitrary. In delivery phase, server sends a *coded* signal to each user such that each user's demand is satisfied.

There have been subsequent papers in this area. Several papers [2]–[4], have considered the problem of tighter lower bounds on the coded caching rate. Several variants of the problem have been examined. The case when files have different popularity levels has been examined in [5]–[7], device-to-device (D2D) wireless networks where there is no central server were considered in [8], [9] and systems with differing file sizes were investigated in [10]. Coded caching over a more general class of network topologies was examined in [11], [12].

In this work we investigate certain issues with the achievability scheme of [1]. In particular, in the placement phase of [1] each file is split into a large number of subfiles (henceforth, the subpacketization level); the number of subfiles grows exponentially with K for a fixed cache size. This can cause issues in actual implementations of coded caching. Specifically, even for moderate number of users (K), the size of the files stored in the server need to be very large. Moreover, in practice each subfile needs to have appropriate header information that allows for bookkeeping at the server and the users. The rate overhead associated with the header will also grow as the number of subfiles is large. We discuss this issue in more detail in Section II.

In this work, we propose new schemes for coded caching that have significantly smaller subpacketization level than the scheme of [1]. Our schemes are derived from constructions of combinatorial objects known as resolvable designs [13] in the literature. This issue was considered in the work of [14], but for the case of decentralized caching. In independent work, [15] arrived at a similar result to the one presented in our paper. However, the techniques used in our paper are quite different and our construction is significantly simpler than theirs.

A. Main contributions

- The subpacketization level of our scheme is exponentially lower than the scheme of [1]. This implies that our schemes are much more amenable to practical implementations even for smaller values of K
- The transmission rate of our scheme is not too much higher than the scheme of [1]. In particular, for large K, both schemes have almost the same rate.

This paper is organized as follows, Section II presents the problem formulation and preliminary definitions. In Section

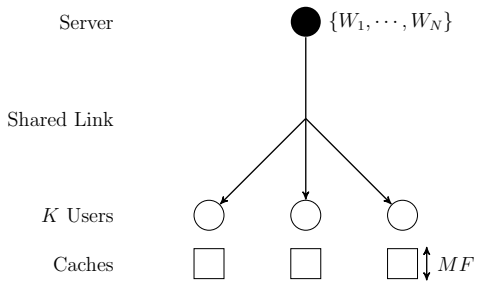

Fig. 1. A coded caching system with N files in the server, K users each equipped with a cache of size MF bits. The server communicates with the users over a shared link.

III, we describe our proposed coded caching scheme and analyze its performance. We compare the performance of our scheme with competing schemes in Section IV. Section V concludes the paper with a discussion of future work.

II. Problem Formulation and Preliminaries

In this work, we consider a caching system consisting of a single server and K users, $U_1, \cdots, U_K$, such that the server is connected to all the users through an error-free shared link (see Fig. 1). The server contains a library of N files where each file is of size F bits. These files are represented by random variables W_i, $i = 1, \cdots, N$, where W_i is distributed uniformly over the set $[2^F]$ (we use $[n]$ to denote the set $\{1, 2, \ldots, n\}$ throughout). Each user has a cache memory of MF bits, where $M \leq N$.

An (M, R) caching system can be defined as follows.

- K *caching functions:* User U_i caches $Z_i = \phi_i(W_1, \cdots, W_N)$ in the placement phase. Here, $\phi_i : [2^F] \to [2^{MF}]$.
- N^K *encoding functions:* The server transmits signals $X_{d_1,\cdots,d_K} = \psi_{d_1,\cdots,d_K}(W_1, \cdots, W_N)$ over the shared link to each user in the delivery phase. Here, $\psi_{d_1,\cdots,d_K} : [2^{NF}] \to [2^{RF}]$.
- KN^K *decoding functions:* User U_i uses the decoding function $\hat{W}_{d_1,\cdots,d_K;i} = \mu_{d_1,\cdots,d_K;i}(X_{d_1,\cdots,d_K}, Z_i)$. Here, $\mu_{d_1,\cdots,d_K;i}(X_{d_1,\cdots,d_K}, Z_i) : [2^{RF}] \times [2^{MF}] \to [2^F]$

The probability of error in a coded caching system is defined as $P_e = \max_{(d_1,\ldots,d_K)\in[N]^K} \max_{i\in[K]} P(\hat{W}_{d_1,\cdots,d_K;i} \neq W_{d_i})$. The pair (M, R) is said to be achievable if for every $\epsilon > 0$ and every large enough file size F, there exists a (M, R) caching system such that P_e is at most ϵ.

The work of Maddah-Ali and Niesen [1] proposed an achievable (M, R) caching scheme when $t = KM/N$ is an integer. Their scheme partitions each file into $F_s = \binom{K}{\frac{KM}{N}}$ non-overlapping subfiles of equal size and places judiciously chosen subsets of the subfiles into the user caches in the placement phase. This is referred to as uncoded placement in the literature. They achieve a rate of

$$R = K \times \left(1 - \frac{M}{N}\right) \times \frac{1}{1 + \frac{KM}{N}},$$

when $K \leq N$. It can be observed that for the scheme of [1], the subpacketization level F_s grows exponentially with K, when M/N is fixed. This can be problematic in practical implementations.

For instance, the atomic unit of storage on present day hard drives is a sector of size 512 bytes and the trend in the disk drive industry is to move this to 4096 bytes. Now, suppose that $K = 50$, with $\frac{M}{N} = \frac{1}{2}$ so that $F_s \approx 10^{14}$. In this case, it is evident that one needs the files to be of at least size $\approx 5 \times 10^8$ gigabytes for leveraging the gains promised by the scheme of [1]. Thus, their scheme is not practical in this setting. Even for smaller values of K, schemes with low subpacketization levels are desirable. This is because any practical scheme will require each of the subfiles to have some header information that allows for decoding at the end users. When there are a large number of subfiles, the header overhead may be non-negligible.

In this work, we propose novel placement and delivery schemes that operate with significantly lower subpacketization levels. Towards this end, we first demonstrate that uncoded placement where each user has the same amount of cache memory can be represented by a block design [13].

Definition 1. *A design is a pair $(X, \mathcal{A})$ such that*

1) *X is a set of elements called points, and*
2) *$\mathcal{A}$ is a collection (i.e., multiset) of nonempty subsets of X called blocks, where each block contains the same number of points.*

A design is in one-to-one correspondence with an incidence matrix $\mathcal{N}$ which is defined as follows.

Definition 2. *The incidence matrix $\mathcal{N}$ of a design $(X, \mathcal{A})$ is a binary matrix of dimension $|X| \times |\mathcal{A}|$, where the rows and columns correspond to the points and blocks respectively. Let $i \in X$ and $j \in \mathcal{A}$. Then,*

$$\mathcal{N}(i, j) = \begin{cases} 1 & \text{if } i \in j, \\ 0 & \text{otherwise.} \end{cases}$$

In general, we can define the placement schemes by using the incidence matrix. One can view the placement scheme of [1] when $KM/N = t$ is an integer as an instance of a block design as follows. We associate the users with the points, i.e., $X = [K]$ and the subfiles as the blocks, i.e., $\mathcal{A} = \{B : B \subset [K], |B| = t\}$. Each file W_n is divided into $\binom{K}{t}$ parts indexed as $W_{n,B}, B \in \mathcal{A}$. User i caches $W_{n,B}$ for $B \in \mathcal{A}$ if $i \in B$ or equivalently if the corresponding entry in the incidence matrix is a one. In general, we can reverse the roles of the points and blocks and choose to associate the users with the blocks and subfiles with the points instead. The transpose of the incidence matrix then allows us to specify the placement.

In this work, we will utilize resolvable designs which are a special class of block designs.

Definition 3. *A parallel class $\mathcal{P}$ in a design $(X, \mathcal{A})$ is a subset of disjoint blocks from $\mathcal{A}$ whose union is X. A partition of $\mathcal{A}$ into several parallel classes is called a resolution, and $(X, \mathcal{A})$ is said to be a resolvable design if $\mathcal{A}$ has at least one resolution.*

We now provide an example of how a design can be used in the placement scheme, when the users and subfiles correspond to the blocks and points, respectively.

Example 1. *Consider a block design specified as follows.*

$$X = \{1,2,3\}, \qquad \mathcal{A} = \{\{1,2\},\{1,3\},\{2,3\}\}, \text{ with}$$
$$\mathcal{N} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

As described below, it corresponds to a coded caching scheme with $K = 3$ and $M/N = 2/3$.

We let the blocks correspond to users which are denoted as U_{12}, U_{13}, U_{23}. Each file is subdivided into $|X| = 3$ subfiles denoted as $W_{n,1}, W_{n,2}, W_{n,3}$ for $n \in [N]$. The placement is specified as follows.

$$Z_{12} = \{W_{n,1}, W_{n,2}\}_{n=1}^{N}$$
$$Z_{13} = \{W_{n,1}, W_{n,3}\}_{n=1}^{N}$$
$$Z_{23} = \{W_{n,2}, W_{n,3}\}_{n=1}^{N}.$$

In the delivery phase, suppose that U_{12}, U_{13}, U_{23} request files $W_{d_{12}}, W_{d_{13}}, W_{d_{23}}$. Using the delivery signal

$$W_{d_{12},3} \oplus W_{d_{13},2} \oplus W_{d_{23},1}$$

all three users can recover their missing subfiles.

III. A Low Subpacketization Level Scheme

Consider a coded caching scenario where the number of users K can be factored as $K = q \times k$ (this requires K to be composite). In this section we use resolvable designs to arrive at a scheme where the subpacketization level is significantly smaller than prior schemes.

A. Resolvable Design Construction

Let $\mathbb{Z}_q$ denote the additive group of integers modulo q. Consider the generator matrix of a $(k, k-1)$ single parity check (SPC) code over $\mathbb{Z}_q$ defined below.

$$\mathbf{G}_{SPC} = \left[\begin{array}{c|c} & 1 \\ \mathbf{I_{k-1}} & \vdots \\ & 1 \end{array}\right]. \tag{1}$$

This code has q^{k-1} codewords which can be obtained by computing $\mathbf{c} = \mathbf{u} \cdot \mathbf{G}_{SPC}$ for all possible message vectors $\mathbf{u}$. We collect the q^{k-1} codewords $\mathbf{c}_i$ and construct a matrix $\mathbf{T}$ of size $k \times q^{k-1}$ specified as follows.

$$\mathbf{T} = [\mathbf{c}_1^T, \mathbf{c}_2^T, \cdots, \mathbf{c}_{q^{k-1}}^T]. \tag{2}$$

Let $X_{SPC} = [q^{k-1}]$ represent the point set of the design. We define the blocks as follows. For $0 \le l \le q-1$, let $B_{i,l}$ be a block defined as

$$B_{i,l} = \{j : \mathbf{T}_{i,j} = l\}.$$

The set of blocks $\mathcal{A}_{SPC}$ is given by the collection of all $B_{i,l}$ for $1 \le i \le k$ and $0 \le l \le q-1$ so that $|\mathcal{A}_{SPC}| = kq$.

Example 2. *Let $q = 2$, $k = 3$. Consider a $(3,2)$ SPC code over $\mathbb{Z}_2$ with generator matrix*

$$\mathbf{G}_{SPC} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

The four codewords in this code are $\mathbf{c}_1 = [0\ 0\ 0]$, $\mathbf{c}_2 = [0\ 1\ 1]$, $\mathbf{c}_3 = [1\ 0\ 1]$, $\mathbf{c}_4 = [1\ 1\ 0]$, and $\mathbf{T}$ is constructed as follows.

$$\mathbf{T} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

Using $\mathbf{T}$, we generate the resolvable block design $(X, \mathcal{A})$ as follows. The point set $X = \{1,2,3,4\}$. Block $B_{1,0}$ is obtained by determining the column indexes where the first row of $\mathbf{T}$ is zero. Thus $B_{1,0} = \{1,2\}$. Proceeding in this manner we obtain

$$\mathcal{A} = \{\{1,2\},\{3,4\},\{1,3\},\{2,4\},\{1,4\},\{2,3\}\}.$$

It can be observed that $\mathcal{A}$ has a resolution (cf. Definition 3) with the following parallel classes.

$$\mathcal{P}_1 = \{\{1,2\},\{3,4\}\},$$
$$\mathcal{P}_2 = \{\{1,3\},\{2,4\}\}, \text{ and}$$
$$\mathcal{P}_3 = \{\{1,4\},\{2,3\}\}.$$

The following lemma shows that the construction procedure above always results in a resolvable design.

Lemma 1. *The construction procedure above produces a design $(X_{SPC}, \mathcal{A}_{SPC})$ where $X_{SPC} = [q^{k-1}]$, $|B_{i,l}| = q^{k-2}$ for all $1 \le i \le k$ and $0 \le l \le q-1$. Furthermore, the design is resolvable with parallel classes given by $\mathcal{P}_i = \{B_{i,l} : 0 \le l \le q-1\}$, for $1 \le i \le k$.*

Proof: For a given i, we need to show that $|B_{i,l}| = q^{k-2}$ for all $0 \le l \le q-1$ and that $\cup_{l=0}^{q-1} B_{i,l} = [q^{k-1}]$. Towards this end we note that for $\Delta = [\Delta_1\ \Delta_2\ \dots\ \Delta_k] = \mathbf{u}\mathbf{G}_{SPC}$, we have

$$\Delta_i = \begin{cases} \mathbf{u}_i & i = 1, \dots, k-1, \\ \sum_{j=1}^{k-1} \mathbf{u}_j & i = k. \end{cases}$$

Thus, for $1 \le i \le k-1$, we have that $|B_{i,l}| = |\{\mathbf{u} : \mathbf{u}_i = l\}|$ which in turns equals q^{k-2} as it is the subset of all message vectors with the i-th coordinate equal to l. Moreover, as the i-th coordinate has to belong to $\{0, \dots, q-1\}$, we have that $\mathcal{P}_i = \{B_{i,l} : 0 \le l \le q-1\}$ forms a parallel class.

It remains to show the same result when $i = k$. For this consider the equation

$$\sum_{j=1}^{k-2} \mathbf{u}_j = l - \mathbf{u}_{k-1}$$

where l is fixed. For arbitrary $\mathbf{u}_j, 1 \le j \le k-2$, this equation has a unique solution for $\mathbf{u}_{k-1}$.This implies that for any l, $|B_{k,l}| = q^{k-2}$ and that $\mathcal{P}_k$ forms a parallel class. ∎

Remark 1. *If q is a prime power then constructions of affine resolvable balanced incomplete block designs (BIBDs) with*

significantly more parallel classes are known (see [13], Ch. 5). However, our proposed scheme above works for any value of q and is adapted for the application to coded caching that we consider.

B. Usage in a coded caching scenario

We first demonstrate our proposed placement scheme by using Example 3 below. We associate the users with the blocks and subfiles with the points of the design.

Example 3. *Consider the resolvable design from Example 2. The six blocks in $\mathcal{A}$ correspond to six users U_{12}, U_{34}, U_{13}, U_{24}, U_{14}, U_{23}. Each file is partitioned into $F_s = 4$ subfiles $W_{n,1}, W_{n,2}, W_{n,3}, W_{n,4}$ which correspond to the four points in X. The cache in user U_B, denoted Z_B is specified as*

$$\begin{aligned}
Z_{12} &= (W_{n,1}, W_{n,2})_{n=1}^{N} \\
Z_{34} &= (W_{n,3}, W_{n,4})_{n=1}^{N} \\
Z_{13} &= (W_{n,1}, W_{n,3})_{n=1}^{N} \\
Z_{24} &= (W_{n,2}, W_{n,4})_{n=1}^{N} \\
Z_{14} &= (W_{n,1}, W_{n,4})_{n=1}^{N} \\
Z_{23} &= (W_{n,2}, W_{n,3})_{n=1}^{N}
\end{aligned}$$

This corresponds to a coded caching system where each user caches half of each file so that $M/N = 1/2$.

Suppose that in the delivery phase user U_B requests file W_{d_B} where $d_B \in [N]$. These demands can be satisfied as follows.

Example 4. *Consider the placement scheme specified in Example 3. For a set of requests $W_{d_{12}}, W_{d_{34}}, W_{d_{13}}, W_{d_{24}}, W_{d_{14}}$ and $W_{d_{23}}$, we pick three blocks from three different parallel classes $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$ and generate the signals transmitted in the delivery phase as follows.*

$$\begin{aligned}
&W_{d_{12},3} \oplus W_{d_{13},2} \oplus W_{d_{23},1}, \\
&W_{d_{12},4} \oplus W_{d_{24},1} \oplus W_{d_{14},2}, \\
&W_{d_{34},1} \oplus W_{d_{13},4} \oplus W_{d_{14},3}, \text{ and} \\
&W_{d_{34},2} \oplus W_{d_{24},3} \oplus W_{d_{23},4}.
\end{aligned}$$

The three sums in the first signal correspond to blocks from different parallel classes $\{1,2\} \in \mathcal{P}_1, \{1,3\} \in \mathcal{P}_2, \{2,3\} \in \mathcal{P}_3$. It can be observed that this equation benefits each of the three users participating in it. Furthermore, it is also apparent that at the end of the delivery phase, each user obtains its missing subfiles. This scheme corresponds to a subpacketization level of 4 and a rate of 1. In contrast, the scheme of [1] would require a subpacketization level of $\binom{6}{3} = 20$ with a rate of 0.75.

Upon inspection, it can be observed that the proposed scheme works since it allows us to always generate an equation where one user from each parallel class can participate. Crucially, the equations can be chosen so that at the end of the transmission each user is satisfied.

The basic idea conveyed by the example above can be generalized as follows. For a coded caching scheme with $K = kq$ and $M/N = 1/q$, suppose that we generate the resolvable design $(X, \mathcal{A})$ by the procedure outlined in Section III-A. Let each block in $\mathcal{A}$ correspond to a user and each point in X correspond to a subfile. We split each file $W_n, n \in [N]$ into q^{k-1} subfiles, so that $W_n = \{W_{n,t} : t \in [q^{k-1}]\}$ and perform the cache placement by using the incidence matrix of the design. Thus, subfile $W_{n,t}$ is placed in the cache of user U_B if $t \in B$ and hence each user caches a total of Nq^{k-2} subfiles. Since each of subfiles has size $\frac{F}{q^{k-1}}$, this requires

$$Nq^{k-2}\frac{F}{q^{k-1}} = F\frac{N}{q} = FM$$

bits of cache memory at each user. Thus, the memory constraint is satisfied.

It remains to show that we can design a delivery phase scheme that satisfies any possible demand pattern. Towards this end we need the following claim, whose proof is deferred to the Appendix.

Claim 1. *Consider a resolvable design $(X, \mathcal{A})$ constructed by the procedure in Section III-A for given k and q. Let the parallel classes of the design be denoted $\mathcal{P}_1, \ldots, \mathcal{P}_k$. Consider blocks $B_{i_1,l_1}, \ldots, B_{i_{k-1},l_{k-1}}$ (where $i_j \in [k], l_j \in \{0, \ldots, q-1\}$) that are picked from $k-1$ distinct parallel classes $\mathcal{P}_{i_1}, \ldots, \mathcal{P}_{i_{k-1}}$. Then, $|\cap_{j=1}^{k-1} B_{i_j,l_j}| = 1$.*

Note that the blocks are in one to one correspondence with the users. Thus, Claim 1 shows that any $k-1$ users picked from different parallel classes have one subfile in common.

Roughly speaking, this implies that if we pick k users, one from each parallel class, then we can generate an equation that is simultaneously useful to each of them. A subsequent counting argument shows that each user can be satisfied.

We now formalize this argument. Let the request of user $U_B, B \in \mathcal{A}$ be denoted by W_{d_B}.

Delivery Phase algorithm

1) Pick users $U_{B_{1,l_1}}, \ldots, U_{B_{k,l_k}}$ where $l_i \in \{0, \ldots, q-1\}, i \in [k]$ and $B_{i,l_i} \in \mathcal{P}_i$, such that $\cap_{i=1}^{k} B_{i,l_i} = \phi$.
2) Let $\hat{l}_\alpha = \cap_{i \in [k] \setminus \{\alpha\}} B_{i,l_i}$ for $\alpha = 1, \ldots, k$.
3) Server transmits

$$\oplus_{\alpha \in [k]} W_{d_{B_{\alpha,l_\alpha}}, \hat{l}_\alpha},$$

4) Repeat Step 1, until all users are satisfied.

Claim 2. *The delivery phase algorithm proposed above terminates and allows each user's demand to be satisfied. Furthermore the rate of transmission of the server is $R = q-1$.*

The proof of the above claim appears in the Appendix.

IV. Comparison with Existing Schemes

In this section, we compare our proposed scheme with the scheme of [1] with $K = qk$ and $\frac{M}{N} = \frac{1}{q}$. Let R^* and F^* denote the rate and the subpacketization level of our proposed scheme, R^{MN} and F^{MN} denote the rate and the

K	2	4	8	10	12	14	16
R^{MN}	0.67	0.75	0.8	0.83	0.86	0.875	0.89
R^*	1	1	1	1	1	1	1
F^{MN}	6	20	70	252	924	3432	12870
F^*	2	4	8	16	32	64	128

TABLE I
NUMERICAL COMPARISON FOR $\frac{M}{N} = \frac{1}{2}$ AND DIFFERENT VALUES OF K

.

subpacketization level of [1], where

$$\begin{aligned} R^{MN} &= \frac{K(1-\frac{M}{N})}{1+\frac{KM}{N}} \\ &= \frac{qk-k}{1+k}, \text{ and} \\ F^{MN} &= \binom{K}{\frac{KM}{N}} \\ &= \binom{qk}{k}. \end{aligned}$$

In our proposed scheme,

$$\begin{aligned} R^* &= q-1, \text{ and} \\ F^* &= q^{k-1}. \end{aligned}$$

Thus, the following conclusions can be drawn.

$$\frac{R^{MN}}{R^*} = \frac{k}{1+k}.$$

This implies that the rate of our proposed scheme and the scheme of [1] is almost the same for large k.

For large k, we show (see Appendix, Lemma 2) that

$$\frac{F^{MN}}{F^*} \approx (\frac{q}{q-1})^{qk-k}.$$

This implies that our subpacketization is *exponentially* smaller compared to the scheme of [1]. Table IV shows a precise numerical comparison when $q = 2$, i.e., $M/N = 1/2$.

An alternate technique for achieving $M/N = 1/q$ with a lower subpacketization level is to perform memory-sharing between appropriate points using the scheme of [1]. Next, we compare our proposed scheme with memory-sharing for the case of $q = 2$, i.e., the number of users $K = 2k$.

Towards this end, we divide each file into two smaller files W_n^1, W_n^2 with equal size, and further split W_n^1, W_n^2 into $\binom{2k}{t}$ and $\binom{2k}{2k-t}$ subfiles, respectively, where $t < k$. In the placement phase, $\frac{t}{2k}$ fraction of the subfiles of W_n^1 and $\frac{2k-t}{2k}$ fraction of the subfiles of W_n^2 are placed in each user using the placement scheme of [1]. Thus, the overall cache at each user is $M = N \cdot \frac{1}{2}(\frac{t}{2k} + \frac{2k-t}{2k})$ so that $M/N = 1/2$. In the delivery phase, the transmission rate is given by

$$R^{MN,MS} = \frac{1}{2}\left(\frac{2k-t}{1+t} + \frac{t}{1+2k-t}\right).$$

The subpacketization level of this scheme $F_s^{M-D,MS} = 2\binom{2k}{t}$. We compare our proposed scheme with the memory sharing scheme considered above by choosing a value of t so that the rates of the schemes are approximately the same. Since $\frac{2k-t}{1+t} > 1$ and $\frac{t}{1+2k-t} < 1$, we approximate $R^{MN,MS} \approx \frac{2k-t}{2(1+t)}$. Then $R^{MN,MS} = R^*$ if $t = \frac{2k-2}{3}$. For this setting we show (see Appendix, Lemma 3) that

$$\frac{F^{MN,MS}}{F^*} \approx 2^{2.8k}.$$

Thus, our scheme has a significantly lower subpacketization level compared to a memory-sharing solution.

V. CONCLUSIONS AND FUTURE WORK

In this work, we proposed a novel scheme for coded caching whose subpacketization level is exponentially smaller than the scheme in [1]. Moreover, for large number of users, the rate of our scheme is almost the same as [1]. Our schemes are derived from resolvable block designs generated by a single parity-check code over $\mathbb{Z}_q$.

There are several opportunities for future work. Our proposed scheme currently only works when M/N is the reciprocal of a positive integer. Furthermore, even though our subpacketization level is significantly lower than [1], it still scales exponentially with the number of users, albeit much slowly. Investigating schemes with subpacketization levels that grow sub-exponentially with K and schemes that work with general values of M/N are interesting directions for future work.

REFERENCES

[1] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Info. Theory*, vol. 60, pp. 2856–2867, May 2014.

[2] H. Ghasemi and A. Ramamoorthy, "Improved lower bounds for coded caching," preprint, 2016, [Online] Available: http://arxiv.org/abs/1501.06003.

[3] ——, "Improved lower bounds for coded caching," in *Proc. of IEEE Int. Symp. Inform. Theory (ISIT)*, 2015.

[4] A. Sengupta, R. Tandon, and T. C. Clancy, "Improved approximation of storage-rate tradeoff for caching via new outer bounds," in *IEEE Intl. Symposium on Info. Th.* IEEE, 2015, pp. 1691–1695.

[5] U. Niesen and M. Maddah-Ali, "Coded caching with nonuniform demands," in *IEEE INFOCOM*, April 2014, pp. 221–226.

[6] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, "Order optimal coded caching-aided multicast under zipf demand distributions," in *The 11th Intl. Symp. on Wireless Comm. Sys.*, 2014.

[7] J. Hachem, N. Karamchandani, and S. Diggavi, "Multi-level coded caching," in *IEEE Intl. Symposium on Info. Th.*, 2014, pp. 56–60.

[8] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of distributed caching in d2d wireless networks," in *IEEE Info. Th. Workshop*, 2013, pp. 1–5.

[9] A. Sengupta and R. Tandon, "Beyond cut-set bounds-the approximate capacity of d2d networks," in *IEEE Info. Th. Workshop*, 2015, pp. 78–83.

[10] J. Zhang, X. Lin, C.-C. Wang, and X. Wang, "Coded caching for files with distinct file sizes," in *IEEE Intl. Symposium on Info. Th.*, 2015, pp. 1686–1690.

[11] M. Ji, M. F. Wong, A. M. Tulino, J. Llorca, G. Caire, M. Effros, and M. Langberg, "On the fundamental limits of caching in combination networks," in *IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, June 2015, pp. 695–699.

[12] L. Tang and A. Ramamoorthy, "Coded caching for networks with the resolvability property," in *Proc. of IEEE Int. Symp. Inform. Theory (ISIT)*, 2016.

[13] D. R. Stinson, *Combinatorial Designs: Construction and Analysis*. Springer, 2003.

[14] K. Shanmugam, M. Ji, A. M. Tulino, J. Llorca, and A. G. Dimakis, "Finite length analysis of caching-aided coded multicasting," in *52nd Annual Allerton Conference on Communication, Control, and Computing*, Sept 2014, pp. 914–920.
[15] Q. Yan, M. Cheng, X. Tang, and Q. Chen, "On the placement delivery array design in centralized coded caching scheme," preprint, 2016, [Online] Available: http://arxiv.org/abs/1510.05064.
[16] R. L. Graham, *Concrete mathematics: a foundation for computer science*. Pearson Education India, 1994.

APPENDIX

Proof of Claim 1

Following the construction in Section III-A, we note that a block $B_{i,l} \in \mathcal{P}_i$ is specified by

$$B_{i,l} = \{j : \mathbf{T}_{i,j} = l\}$$

Now, consider $B_{i_1,l_1}, \ldots, B_{i_{k-1},l_{k-1}}$ (where $i_j \in [k], l_j \in \{0, \ldots, q-1\}$) that are picked from $k-1$ distinct parallel classes $\mathcal{P}_{i_1}, \ldots, \mathcal{P}_{i_{k-1}}$. W.l.o.g. we assume that $i_1 < i_2 < \cdots < i_{k-1}$. Let $\mathcal{I} = \{i_1, \ldots, i_{k-1}\}$ and $\mathbf{T}_{\mathcal{I}}$ denote the submatrix of $\mathbf{T}$ obtained by retaining the rows in $\mathcal{I}$. We will show that the vector $[l_1\ l_2\ \ldots\ l_{k-1}]^T$ is a column in $\mathbf{T}_{\mathcal{I}}$.

To see this first we consider the case that $\mathcal{I} = \{1, \ldots, k-1\}$. In this case, the message vector $\mathbf{u} = [l_1\ l_2\ \ldots\ l_{k-1}]$ is such that $[\mathbf{u}\mathbf{G}_{SPC}]_{\mathcal{I}} = [l_1\ l_2\ \ldots\ l_{k-1}]^T$ so that $[l_1\ l_2\ \ldots\ l_{k-1}]^T$ is a column in $\mathbf{T}_{\mathcal{I}}$. On the other hand if $k \in \mathcal{I}$, then we have $i_{k-1} = k$. Now, consider the system of equations in variables $u_1, \ldots, u_{k-1}$.

$$\begin{aligned} u_{i_1} &= l_1, \\ u_{i_2} &= l_2, \\ &\vdots \\ u_{i_{k-2}} &= l_{k-2}, \\ u_1 + u_2 + \cdots + u_{k-1} &= l_{k-1}. \end{aligned}$$

It is evident that this system of $k-1$ equations in $k-1$ variables has a unique solution over $\mathbb{Z}_q$. The result follows. ∎

Proof of Claim 2

In the arguments below, for the sake of convenience we argue that user $U_{B_{1,0}}$ can recover all its missing subfiles. As the delivery phase algorithm is symmetric with respect to the users, this equivalently shows that all users can recover their missing subfiles.

Note that $|B_{1,0}| = q^{k-2}$. Thus, user $U_{B_{1,0}}$ needs to obtain $q^{k-1} - q^{k-2}$ missing subfiles. The delivery phase scheme repeatedly picks k users from different parallel classes $U_{B_{1,0}}, U_{B_{2,l_2}} \ldots, U_{B_{k,l_k}}$ such that $B_{1,0} \bigcap \cap_{i=2}^{k} B_{i,l_i} = \phi$. According to the equation transmitted in Step 3 of the algorithm, this allows $U_{B_{1,0}}$ to recover subfile $W_{d_{B_{1,0}},\hat{l}_1}$ where $\hat{l}_1 = \cap_{i=2}^{k} B_{i,l_i}$. Note that $U_{B_{1,0}}$ does not have $W_{d_{B_{1,0}},\hat{l}_1}$ in its caches since $B_{1,0} \bigcap \cap_{i=2}^{k} B_{i,l_i} = \phi$.

Next, we count the number of equations that $U_{B_{1,0}}$ participates in. We can pick $k-2$ users from parallel classes $\mathcal{P}_2, \ldots, \mathcal{P}_{k-1}$. Claim 1 ensures that blocks corresponding to these users intersect in a single point. Next we pick a block from the remaining parallel class $\mathcal{P}_k$ such that the intersection of all the blocks is empty; this can be done in $q-1$ ways. Thus, there are a total of $q^{k-2}(q-1) = q^{k-1} - q^{k-2}$ equations in which user $U_{B_{1,0}}$ participates.

We have previously argued that each such equation allows $U_{B_{1,0}}$ to decode a subfile that it does not have in its cache. If we can argue that each equation provides a distinct file part then our argument is complete. Towards this end suppose that there exist sets of blocks $\{B_{2,l_2}, \ldots, B_{k,l_k}\}$ and $\{B_{2,l'_2}, \ldots, B_{k,l'_k}\}$ such that $\{B_{2,l_2}, \ldots, B_{k,l_k}\} \neq \{B_{2,l'_2}, \ldots, B_{k,l'_k}\}$, but $\cap_{i=2}^{k} B_{i,l_i} = \cap_{i=2}^{k} B_{i,l'_i} = \{\beta\}$ for some $\beta \in [q^{k-1}]$. This is a contradiction since this in turn implies that $\cap_{i=2}^{k} B_{i,l_i} \bigcap \cap_{i=2}^{k} B_{i,l'_i} = \{\beta\}$, which is impossible since two blocks from the same parallel class have an empty intersection.

Finally, we calculate the rate of the delivery phase algorithm. We transmit a total of $q^{k-1}(q-1)$ equations, where each symbol is of size F/q^{k-1}. Thus, the rate is given by,

$$\begin{aligned} R &= q^{k-1}(q-1)\frac{F}{q^{k-1}} \\ &= (q-1)F. \end{aligned}$$

∎

Lemma 2. *Suppose $K = qk$ and let q be fixed. Then,*

$$\lim_{k\to\infty} \frac{1}{kq} \log_2 \frac{F_s^{MN}}{F_s^*} = \left(1 - \frac{1}{q}\right) \log_2 \left(\frac{q}{q-1}\right).$$

Proof: It is well known [16] that for $0 \leq p \leq 1$

$$\lim_{n\to\infty} \frac{1}{n} \log_2 \binom{n}{pn} = H(p), \tag{3}$$

where $H(\cdot)$ represents the binary entropy function. Using this

$$\begin{aligned} \lim_{k\to\infty} \frac{1}{qk} \log_2 \frac{F_s^{MN}}{F_s^*} &= \lim_{k\to\infty} \frac{1}{qk} \log_2 \frac{\binom{qk}{k}}{q^{k-1}} \\ &= H(\frac{1}{q}) - \frac{\log_2 q}{q} \\ &= \left(1 - \frac{1}{q}\right) \log_2 \left(\frac{q}{q-1}\right). \end{aligned}$$

∎

Lemma 3. *Suppose $K = 2k$, $\frac{M}{N} = \frac{1}{2}$ and $t = \frac{2k-2}{3}$. Then,*

$$\lim_{k\to\infty} \frac{1}{k} \log_2 \frac{F_s^{MN,MS}}{F_s^*} = 2.8.$$

Proof: This follows from eq. (3) in Lemma 2 and basic algebraic manipulations. ∎