

Multi-Access Coded Caching Schemes From Cross Resolvable Designs

Digvijay Katyal, Pooja Nayak Muralidhar, and B. Sundar Rajan^{ID}, *Fellow, IEEE*

Abstract—We present a novel caching and coded delivery scheme for a multi-access network where multiple users can have access to the same cache (shared cache) and multiple caches can be accessed by the same user. This scheme is obtained from resolvable designs satisfying certain conditions which we call *cross resolvable designs*. To be able to compare different multi-access coded schemes with different number of users we normalize the rate of the schemes by the number of users served. Based on this per-user-rate we show that our scheme performs better than the well known Maddah-Ali - Niesen (MaN) scheme and the recently proposed (“Multi-access coded caching: gains beyond cache-redundancy” by Serbetci, Parrinello and Elia) SPE scheme. It is shown that the resolvable designs from affine planes are cross resolvable designs and our scheme based on these performs better than the MaN scheme for large memory size cases. The exact size beyond which our performance is better is also presented. The SPE scheme considers only the cases where the product of the number of users and the normalized cache size is 2, whereas the proposed scheme allows different choices depending on the choice of the cross resolvable design.

Index Terms—Coded caching, multi-access, resolvable designs.

I. INTRODUCTION

CACHING techniques help to reduce data transmissions during the times of high network congestion by prefetching parts of popularly demanded contents into the memories of end users. The seminal work of [1] provided a coded delivery scheme which performed within a constant factor of the information-theoretic optimum for all values of the problem parameters.

The idea of a placement delivery array (PDA) to represent the placement and delivery phase of a coded caching problem first appeared in [2]. These PDAs could represent any coded caching problem with symmetric prefetching and the popular Ali-Niesen scheme was also found to be a special case. Recently, placement delivery arrays have found applications

in different variants of coded caching scenarios like in Device to device (D2D) networks as D2D placement delivery array (DPDA) and in Combination networks as combinational PDA (C-PDA) [4], [5].

Most of the works on coded caching consider scenarios where each user has its own dedicated cache. However in a variety of settings, such as different cellular networks, multiple users share a single cache or users can conceivably connect to multiple caches whose coverage areas may overlap. The possibility of users to access more than one cache was first addressed in [6]. This was motivated by the upcoming heterogeneous cellular architecture which will contain a dense deployment of wireless access points with small coverage and relatively large data rates, in addition to sparse cellular base stations with large coverage area and small data rates. Placing cache at local access points could significantly reduce the base station transmission rate, with each user being able to access content at multiple access points along with the base station broadcast. The work in [6] considered a K -user shared-link broadcast channel where each user is assisted by exactly $L > 1$ caches (with a cyclic wrap around), and where each cache can serve exactly L users. The authors called this as multi-access problem and derived an achievable rate and information theoretic lower bound which differs by a multiplicative gap scaling linearly with L . Later in [7], [9], new bounds for the optimal rate-memory trade-off were derived for the same problem. Also a new achievable rate for general multi-access setup which is order-wise better than the rate in [6] is derived. The authors focus on the special case with $L \geq K/2$ and provide a general lower bound on the optimal rate and establish its order optimal memory rate trade off under the restrictions of uncoded placement. For few special cases like $L = K - 1$; $L = K - 2$; $L = K - 3$ with K even, exact optimal uncoded memory-rate trade off is derived.

The work of Serbetci *et al.* [8] gives yet another class of coded caching schemes for the multi-access setup, where each user in a K -user shared link broadcast channel is connected to $z > 1$ caches (with a cyclic wrap around), and where each cache can serve exactly z users. This was the first instance where authors have analyzed this scenario in the context of worst case delivery time and when the number of files in server database is greater than equal to the number of users, the proposed scheme experiences a larger gain than in [1].

In [11] authors consider the shared cache scenario where multiple users share the same cache memory, and each user is connected to only one cache. The work in [11] considers the

Manuscript received July 26, 2020; revised November 27, 2020; accepted January 8, 2021. Date of publication January 20, 2021; date of current version May 18, 2021. This work was supported partly by the Science and Engineering Research Board (SERB) of Department of Science and Technology (DST), Government of India, through J.C. Bose National Fellowship to B. Sundar Rajan. The associate editor coordinating the review of this article and approving it for publication was G. Durisi. (*Corresponding author: B. Sundar Rajan.*)

Digvijay Katyal is with Samsung Semiconductor India Research (SSIR), Division of Samsung R&D Institute India, Bangalore Pvt. Ltd., Bengaluru 560048, Karnataka (e-mail: digvijayk@iisc.ac.in).

Pooja Nayak Muralidhar and B. Sundar Rajan are with the Department of Electrical Communication Engineering, Indian Institute of Science, Bengaluru, Bengaluru 560012, India (e-mail: poojam@iisc.ac.in; bsrajan@iisc.ac.in).

Digital Object Identifier 10.1109/TCOMM.2021.3053048

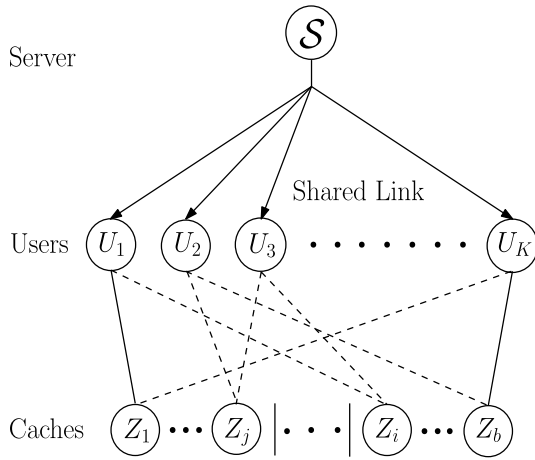


Fig. 1. Problem setup for multi-access coded caching with K users, b caches and each user, connected to z caches.

shared link network, with K users and Λ , $\Lambda \leq K$ helper caches, where each cache is assisting arbitrary number of distinct users. Also each user is assigned to single cache. For this set up, the authors identify the fundamental limits under the assumption of uncoded placement and any possible user to cache association profile. The authors derive the exact optimal worst-case delivery time.

In [12], the authors propose a coded placement scheme for the setup where the users share the end-caches and showed that the proposed scheme outperforms the scheme in [11]. In this scheme the authors use both coded and uncoded data at the caches, taking into consideration the users connectivity pattern. Firstly, for a two-cache system, authors provided an explicit characterization of the gain from coded placement, then the scheme is extended to L -cache systems, where authors obtain optimal parameters for the caching scheme by solving a linear program.

The schemes so far mentioned above in context of multi-access scenarios consider the special framework with cyclic wrap around, while for shared cache problems cyclic wraparound was not asked for. The cache placement followed in the schemes proposed in these works, ensured that the intersection of contents stored in the caches that a user is connected to, is empty. In [13], the authors addressed a system model involving a cache sharing strategy where the set of users served by any two caches is no longer empty. In [13], the authors consider the caching problem with shared caches, consisting of a server connected to users through a shared link, where a pair of users share two caches.

Various combinatorial designs have been used in different setups of coded caching [14], [16], [17].

A. Multi-Access Coded Caching - System Model

Fig. 1 shows a multi-access coded caching system with a unique server S storing N files $W_1, W_2, W_3, \dots, W_N$ each of unit size. There are K users in the network connected via an error free shared link to the server S . The set of users is denoted by \mathcal{K} . There are b number of helper caches each of size M files. Each user has access to z out of the b helper

caches. Let Z_k denotes the content in the k -th cache. It is assumed that each user has an unlimited capacity link to the caches it is connected to.

There are two phases: the placement phase and the delivery phase. During the placement phase certain parts of each file are stored in each cache which is carried out during the off-peak hours. During the peak hours each user demands a file and the server broadcasts coded transmissions such that each user can recover its demand by combining the received transmissions with what has been stored in the caches it has access to. This is the delivery phase. The coded caching problem is to jointly design the placement and the delivery with minimal number of transmissions to satisfy the demands of all the users. The amount of transmissions used in the unit of files is called the *rate* or the *delivery time*. Subpacketization level is the number of packets that a file is divided into. Coding gain is defined as the number of users benefited in a transmission.

B. The Maddah-Ali Niesen (MaN) Coded Caching Scheme

The framework of the seminal paper [1] considers a network with K users, each equipped with memory of size M and N files of very large size among which each user is likely to demand any one file. The rate R achieved is $K(1 - \frac{M}{N}) \frac{1}{(1 + K \frac{M}{N})}$. The factor $(1 + K \frac{M}{N})$ which is originally called global caching gain is also known as the *coding gain* or the *Degrees of Freedom (DoF)*. We refer to this scheme as the MaN scheme henceforth. This original setup can be viewed as a special case of the scheme corresponding to Fig. 1 with $b = K$ and $z = 1$ which may be viewed as each user having a dedicated cache of its own.

C. Serbetci-Parrinello-Elia (SPE) Multi-Access Coded Caching Scheme

In [8], a network consisting of K users connected via an error free shared link to a server S storing N files $W_1, W_2, W_3, \dots, W_N$ is considered. Each user in the network can access z caches out of K helper caches, each of size $M = N\gamma$ (units of file), where $\gamma \in \{\frac{1}{K}, \frac{2}{K}, \dots, 1\}$. The setup of this scheme, which we refer henceforth as the SPE scheme, can be considered as a special case of the setup shown in Fig. 1 where each user $k \in \mathcal{K}$ is associated with the caches,

$$C_k := (k, k + 1, \dots, k + z - 1)_z \subseteq \mathcal{K}.$$

In [8] the authors focus on the special case, $K\gamma = 2$ and provide a scheme which exceed the Ali-Niesen coding gain $K\gamma + 1$. Also, for the special case with access to an integer number $z = \frac{K-1}{K\gamma}$ of caches of normalized size γ , the optimal rate taking the form $K \frac{1-\gamma z}{K\gamma z + 1} = \frac{1}{K}$ corresponding to a degrees of freedom (DoF) of $K\gamma z + 1$ users served at a time is reported.

D. Comparing Different Multi-Access Coded Caching Schemes

In any Multi-access Coded caching problem the design parameters are the number of files N , number of users K , number of caches b , the memory size of the cache, M files and number of caches a user has access to, z . For two Multi-access

schemes the number of users may be different depending upon the cache-user topology, keeping other parameters the same. So comparing such two schemes with respect to rate R the existing rate R may be misleading. Therefore to compare our multi-access scheme with other existing schemes we introduce a new term that normalizes the existing rate R with the number of users K supported i.e., rate per user or per user rate $\frac{R}{K}$. The lower the rate per user for a given b, M, N, z the better the scheme.

E. Contributions

The contributions of this paper may be summarized as follows:

- A subclass of resolvable designs (called cross resolvable designs) is identified using which new classes of multi-access coding schemes are presented.
- To be able to compare different multi-access coded schemes with different number of users we normalize the rate of the schemes by the number of users served. Based on this per-user-rate we show that our scheme performs better than the MaN scheme and the SPE scheme for several cross resolvable designs.
- It is shown that the resolvable designs from affine planes [15] are cross resolvable designs and our scheme performs better than the MaN scheme for large memory size cases. The exact size beyond which our performance is better is also presented.
- The resolvable designs from affine planes are a special case of the affine resolvable designs from balanced incomplete block designs (AR-BIBDs) parametrized by an integer $m \geq 2$, which are CRDs. The case affine resolvable designs from affine planes corresponds to $m = 2$. It is shown that the advantage in terms of subpacketization holds for all values of m the advantage in terms of per user rate holds only for $m = 2$ and $m = 3$ for large memory sizes.
- The SPE scheme [8] considers only the cases where $\frac{KM}{N} = 2$, while the proposed scheme allows different choices of $\frac{KM}{N}$ depending on the choice of the cross resolvable design.

The paper is organized as follows. Section II describes all the details related to resolvable designs and defines a subclass of resolvable designs termed in this paper as cross resolvable designs (CRDs). Our proposed scheme associated with CRDs is described in Section III. Comparisons of performance of our scheme with the MaN and the SPE schemes constitute Section IV. In Section V we discuss schemes obtained from resolvable designs for which the number of caches is equal to the number of users. This includes coded caching schemes from resolvable designs that are not cross resolvable. Concluding remarks constitute Section VI and the proof of correctness of our delivery algorithm is given in the Appendix. For a positive integer n , $[n]$ denotes the set $\{1, 2, \dots, n\}$.

II. CROSS RESOLVABLE DESIGNS

We use a class of combinatorial designs called resolvable designs [15] to specify placement in the caches.

Definition 1 [3]: A design is a pair (X, \mathcal{A}) such that

- X is a finite set of elements called points, and
- \mathcal{A} is a collection of nonempty subsets of X called blocks, where each block contains the same number of points.

Definition 2 [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.

Example 1: Consider a design specified as follows.

$$X = \{1, 2, 3, 4\}, \text{ and}$$

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

It can be observed that this design is resolvable with the following parallel classes.

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

$$\mathcal{P}_3 = \{\{1, 4\}, \{2, 3\}\}.$$

Note that in above example, $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ forms a partition of \mathcal{A} . If $\mathcal{A} = \{\{1, 2\}, \{1, 3\}, \{3, 4\}, \{2, 4\}\}$, we get another resolvable design with two parallel classes \mathcal{P}_1 and \mathcal{P}_2 .

Example 2: Consider a design specified as follows.

$$X = \{1, 2, 3, 4, 5, 6\}, \text{ and}$$

$$\mathcal{A} = \{\{1, 2, 3\}, \{4, 5, 6\}, \{1, 4, 5\}, \{2, 3, 6\}\}.$$

It can be observed that this design is resolvable with the following parallel classes.

$$\mathcal{P}_1 = \{\{1, 2, 3\}, \{4, 5, 6\}\} \text{ and } \mathcal{P}_2 = \{\{1, 4, 5\}, \{2, 3, 6\}\}$$

For a given resolvable design (X, \mathcal{A}) if $|X| = v$, $|\mathcal{A}| = b$, block size is k and number of parallel classes is r , then there are exactly $\frac{b}{r}$ blocks in each parallel class. Since the blocks in each parallel class are disjoint, therefore number of blocks in each parallel class is $= \frac{b}{r} = \frac{v}{k}$.

A. Cross Resolvable Design (CRD)

Definition 3 (Cross Intersection Number): For any resolvable design (X, \mathcal{A}) with r parallel classes, the i^{th} cross intersection number, μ_i where $i \in \{2, 3, \dots, r\}$, is defined as the cardinality of intersection of i blocks drawn from any i distinct parallel classes, provided that, this value remains same ($\mu_i \neq 0$), for all possible choices of blocks.

For instance, in Example 1, $\mu_2 = 1$, as the intersection of any 2 blocks drawn from 2 distinct parallel classes is always at exactly one point. But we cannot define μ_3 as the intersection of 3 blocks drawn from 3 distinct parallel classes takes elements from the set $\{0, 1\}$.

Definition 4 (Cross Resolvable Design): For any resolvable design (X, \mathcal{A}) , if there exist at least one $i \in \{2, 3, \dots, r\}$ such that the i^{th} cross intersection number μ_i exists, then the resolvable design is said to be a Cross Resolvable Design (CRD). For a CRD the maximum value for i for which μ_i exists is called the Cross Resolution Number (CRN) for that CRD. A CRD with the CRN equal to r is called a Maximal Cross Resolvable Design (MCRD).

Note that the resolvable design in *Example 2* is not a CRD as μ_2 does not exist.

Example 3: For the resolvable design (X, \mathcal{A}) with

$$X = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}, \text{ and}$$

$$\mathcal{A} = \{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9\}, \{1, 4, 7\}, \{2, 5, 8\}, \{3, 6, 9\}\}.$$

The parallel classes are

$$\mathcal{P}_1 = \{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9\}\} \text{ and}$$

$$\mathcal{P}_2 = \{\{1, 4, 7\}, \{2, 5, 8\}, \{3, 6, 9\}\}$$

and $\mu_2 = 1$.

Example 4: For the resolvable design (X, \mathcal{A}) with $X = \{1, 2, 3, 4, 5, 6, 7, 8\}$, and $\mathcal{A} = \{\{1, 2, 3, 4\}, \{5, 6, 7, 8\}, \{1, 2, 5, 6\}, \{3, 4, 7, 8\}, \{1, 3, 5, 7\}, \{2, 4, 6, 8\}\}$, the parallel classes are

$$\mathcal{P}_1 = \{\{1, 2, 3, 4\}, \{5, 6, 7, 8\}\},$$

$$\mathcal{P}_2 = \{\{1, 2, 5, 6\}, \{3, 4, 7, 8\}\}, \text{ and}$$

$$\mathcal{P}_3 = \{\{1, 3, 5, 7\}, \{2, 4, 6, 8\}\}.$$

In this case $\mu_2 = 2$ and $\mu_3 = 1$.

Example 5: For the resolvable design (X, \mathcal{A}) with

$$X = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}, \text{ and}$$

$$\mathcal{A} = \{\{1, 2, 3, 4, 5, 6\}, \{7, 8, 9, 10, 11, 12\}, \{1, 2, 3, 7, 8, 9\}, \{4, 5, 6, 10, 11, 12\}\}$$
 the parallel classes are

$$\mathcal{P}_1 = \{\{1, 2, 3, 4, 5, 6\}, \{7, 8, 9, 10, 11, 12\}\}, \text{ and}$$

$$\mathcal{P}_2 = \{\{1, 2, 3, 7, 8, 9\}, \{4, 5, 6, 10, 11, 12\}\}.$$

We have $\mu_2 = 3$.

Example 6: Consider the resolvable design (X, \mathcal{A}) with $X = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, and

$$\mathcal{A} = \{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9\}, \{1, 4, 7\}, \{2, 5, 8\}, \{3, 6, 9\}, \{1, 5, 9\}, \{2, 6, 7\}, \{3, 4, 8\}, \{1, 6, 8\}, \{2, 4, 9\}, \{3, 5, 7\}\}.$$
 The parallel classes are

$$\mathcal{P}_1 = \{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9\}\},$$

$$\mathcal{P}_2 = \{\{1, 4, 7\}, \{2, 5, 8\}, \{3, 6, 9\}\},$$

$$\mathcal{P}_3 = \{\{1, 5, 9\}, \{2, 6, 7\}, \{3, 4, 8\}\}, \text{ and}$$

$$\mathcal{P}_4 = \{\{1, 6, 8\}, \{2, 4, 9\}, \{3, 5, 7\}\}.$$

Here $\mu_2 = 1$ and μ_3, μ_4 does not exist.

From Example 6 one can see that μ_r need not always exist for a CRD.

Lemma 1: For any given CRD (X, \mathcal{A}) with r parallel classes and any cross intersection number μ_i for $i \in \{3, 4, \dots, r\}$ we have, $\mu_{i-1} = \mu_i \frac{v}{k}$.

Proof: From any i parallel classes, let us choose a block from each parallel class denoted as $\mathcal{A}_{1,1}, \mathcal{A}_{2,1}, \mathcal{A}_{3,1}, \dots, \mathcal{A}_{i,1}$. Let

$$\mathcal{A}_{1,1} \cap \mathcal{A}_{2,1} \cap \mathcal{A}_{3,1} \cap \dots \cap \mathcal{A}_{i,1} = \mathcal{M}_1$$

$$\mathcal{A}_{1,1} \cap \mathcal{A}_{2,1} \cap \mathcal{A}_{3,1} \cap \dots \cap \mathcal{A}_{i,2} = \mathcal{M}_2$$

⋮

$$\mathcal{A}_{1,1} \cap \mathcal{A}_{2,1} \cap \mathcal{A}_{3,1} \cap \dots \cap \mathcal{A}_{i, \frac{v}{k}} = \mathcal{M}_{\frac{v}{k}}$$

From the definition of cross resolvable design,

$$|\mathcal{M}_1| = |\mathcal{M}_2| = |\mathcal{M}_3| = \dots = |\mathcal{M}_{\frac{v}{k}}| = \mu_i.$$

It is also easy to see that, any $\mathcal{M}_k \cap \mathcal{M}_l = \phi$ where $k, l \in \{1, 2, \dots, \frac{v}{k}\}$. Now,

$$\mathcal{M}_1 \cup \mathcal{M}_2 \cup \dots \cup \mathcal{M}_{\frac{v}{k}} = \mathcal{A}_{1,1} \cap \mathcal{A}_{2,1} \cap \mathcal{A}_{3,1} \cap \dots \cap \mathcal{A}_{i-1,1}$$

$$\cap \left\{ \bigcup_{l \in [\frac{v}{k}]} \mathcal{A}_{i,l} \right\} \\ = \mathcal{A}_{1,1} \cap \mathcal{A}_{2,1} \cap \mathcal{A}_{3,1} \cap \dots \cap \mathcal{A}_{i-1,1}$$

since $\bigcup_{l \in [\frac{v}{k}]} \mathcal{A}_{i,l} = X$. We have $|\mathcal{M}_1 \cup \mathcal{M}_2 \cup \dots \cup \mathcal{M}_{\frac{v}{k}}| = \mu_i \frac{v}{k}$ and hence

$$|\mathcal{A}_{1,1} \cap \mathcal{A}_{2,1} \cap \mathcal{A}_{3,1} \cap \dots \cap \mathcal{A}_{i-1,1}| = \mu_i \frac{v}{k} \text{ leading to } \mu_{i-1} = \mu_i \frac{v}{k}. \blacksquare$$

III. PROPOSED SCHEME

Given a cross resolvable design (X, \mathcal{A}) with v points, r parallel classes, b blocks of size k each, $b_r \stackrel{\text{def}}{=} \frac{b}{r}$ blocks in each parallel class, we choose some $z \in \{2, 3, \dots, r\}$ such that μ_z exists. Let \mathcal{A}_j denote the j^{th} block in \mathcal{A} , assuming some ordering on the blocks of \mathcal{A} . We associate a coded caching problem with $K = \binom{r}{z} \left(\frac{b}{r}\right)^z$ number of users, N files in server database, b number of caches, $\frac{M}{N} = \frac{k}{v}$ fraction of each file at each cache and subpacketization level v . There is a one to one correspondence between the blocks of the CRD and the caches. A user is connected to distinct z caches such that these z caches correspond to z blocks from distinct parallel classes. We denote the set of K users \mathcal{K} as, $\mathcal{K} := \{U_H : |H| = z\}$ where, H is a z sized set containing cache indices from distinct parallel classes.

A. Placement Phase

In the caching placement phase, we split each file $W_i, \forall i \in [N]$ into v non-overlapping subfiles of equal size i.e.,

$$W_i = (W_{i,k} : k \in X), \quad i = 1, 2, \dots, N.$$

The placement is as follows. In the j^{th} cache, the indices of the subfiles stored in \mathcal{Z}_j is the j^{th} block in the design. We assume symmetric batch prefetching i.e.,

$$\mathcal{Z}_j = \{W_{i,k} : k \in \mathcal{A}_j\}, \quad \forall i \in \{1, 2, \dots, N\}, \forall j \in \{1, 2, \dots, b\}$$

Therefore the total number of subfiles for each file in any cache is block size k of the resolvable design i.e., $\frac{M}{N} = \frac{k}{v}$.

Let $\frac{M'}{N}$ denote the fraction of a file each user has access to. We have

$$\frac{M'}{N} \\ = \sum_{i=1}^z \frac{|\mathcal{A}_i|}{v} - \sum_{1 \leq i_1 < i_2 \leq z} \frac{|\mathcal{A}_{i_1} \cap \mathcal{A}_{i_2}|}{v} + \dots + (-1)^{t+1} \\ \sum_{1 \leq i_1 < \dots < i_t \leq z} \frac{|\mathcal{A}_{i_1} \cap \dots \cap \mathcal{A}_{i_t}|}{v} + \dots + (-1)^{z+1} \frac{|\mathcal{A}_1 \cap \dots \cap \mathcal{A}_z|}{v}$$

where \mathcal{A}_i , $i \in [z]$ are z blocks from z distinct parallel classes. Using Lemma 1 we get,

$$\frac{M'}{N} = \sum_{i=1}^z \frac{M}{N} - \sum_{1 \leq i_1 < i_2 \leq z} \frac{\mu_2}{v} + \dots + (-1)^{t+1} \sum_{1 \leq i_1 < \dots < i_t \leq z} \frac{\mu_t}{v} + \dots + (-1)^{z+1} \frac{\mu_z}{v},$$

which simplifies to

$$\frac{M'}{N} = \frac{zM}{N} + \sum_{t=2}^z (-1)^{t+1} \binom{z}{t} \frac{\mu_t}{v}.$$

From the above expression it is clear that for cross resolvable design $M' \neq zM$. All the cases considered in [8], [10] corresponds to the case that $M' = zM$, i.e., the size of the memory that a user has access to is an integer multiple of the size of the cache. From this it follows that the cases considered in [8] do not intersect with the cases considered in this paper.

Lemma 2: The number of users having access to any particular subfile is exactly $\binom{r}{z} (b_r^z - (b_r - 1)^z)$.

Proof: There are $\binom{r}{z}$ possible ways of choosing z parallel classes out of r parallel classes. Fix some point $l \in [v]$. In each parallel class there will be $(b_r - 1)$ blocks that does not contain l . So there are totally $(b_r - 1)^z$ users which do not have access to l . Hence the number of users which have access to l is given by $(b_r^z - (b_r - 1)^z)$. So, taking all possible combinations of z parallel classes we have $\binom{r}{z} (b_r^z - (b_r - 1)^z)$. ■

Lemma 3: The number of users having access to a subfile of a specific cache is exactly $\binom{r-1}{z-1} (b_r)^{z-1}$.

Proof: Any user is connected to z parallel classes. First we fix a subfile accessible to the user by fixing a cache accessible to the user. Once we fix a cache we also fix a parallel class. Then other $z - 1$ parallel classes can be chosen in $\binom{r-1}{z-1}$ ways. Since there are b_r blocks in each of these parallel classes, we have total number of users equal to $\binom{r-1}{z-1} (b_r)^{z-1}$. ■

B. Delivery Phase

For delivery, we first arrange the users in lexicographical order of their indices S , establishing a one-to-one correspondence with the set $\{1, 2, \dots, K\}$. At the beginning of the delivery phase, each user requests one of the N files and let the demand vector be denoted by $\mathbf{d} = (d_1, d_2, \dots, d_K)$. To derive an upper bound on the required transmission rate, we focus our attention on the worst case scenario i.e., each user requests for distinct files. The delivery steps are presented as an algorithm in **Algorithm 1**.

The proof of correctness of the **Algorithm 1** is given in the Appendix. We have

Theorem 1: For N files and K users each with access to z caches of size M in the considered caching system, if it $N \geq K$ and for the distinct demands by the users, the proposed scheme achieves the rate R given by

$$R = \frac{\mu_z \binom{b_r}{2}^z \binom{r}{z}}{v}$$

Proof: The first **for** loop of the delivery algorithm runs $\binom{r}{z}$ times. The second **for** loop of the delivery algorithm

Algorithm 1 Algorithm for Delivery

```

1: for  $u = 1$  to  $u = \binom{r}{z}$  do
2:   Choose a set of  $z$  out of  $r$  parallel classes
3:   which is different from the sets chosen before.
4:   Let this set be
        $\mathcal{P}_1 = \{C_{1,1}, C_{1,2}, \dots, C_{1,i_1}, \dots, C_{1,b_r}\},$ 
        $\mathcal{P}_2 = \{C_{2,1}, C_{2,2}, \dots, C_{2,i_2}, \dots, C_{2,b_r}\},$ 
        $\vdots$ 
        $\mathcal{P}_z = \{C_{z,1}, C_{z,2}, \dots, C_{z,i_z}, \dots, C_{z,b_r}\}.$ 
5:   for  $v = 1$  to  $v = \binom{b_r}{2}^z$  do
6:     Choose a pair of blocks from each of the  $z$ 
7:     parallel classes  $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_z$ . This set of  $2z$ 
8:     blocks must be different from the ones chosen
9:     before. Let the chosen set be
        $\{C_{1,i_1}, C_{1,j_1}, C_{2,i_2}, C_{2,j_2}, \dots, C_{z,i_z}, C_{z,j_z}\}$ 
10:    where,  $i_s, j_s \in [b_r]$  and  $j_s \neq i_s, \forall s \in [z]$ .
11:    There are  $2^z$  users corresponding to the  $2z$  blocks
12:    chosen above. Denoting this set of user indices by
13:     $\mathcal{X}$ , we have  $\mathcal{X} = \{(C'_1, C'_2, \dots, C'_z) : C'_s \in$ 
        $\{C_{s,i_s}, C_{s,j_s}\}\}$ 
14:    Calculate: Calling the user connected to the
15:    set of caches  $\{C_{1,a_1}, C_{2,a_2}, \dots, C_{z,a_z}\}$ ,
16:    where  $a_k \in \{i_k, j_k\}$ ,  $k = 1, 2, \dots, z$ , to be
17:    the  $m^{\text{th}}$  user calculate the set  $f_m$  as  $f_m = C_{1,e_1} \cap$ 
        $C_{2,e_2} \cap \dots \cap C_{z,e_z}$ 
18:    where  $e_k = \{i_k, j_k\} \setminus a_k$ . We have  $|f_m| = \mu_z$ . Let
19:     $f_m := \{y_{m,1}, y_{m,2}, \dots, y_{m,\mu_z}\}$ .
20:    Calculate  $f_m$  as above for all the  $2^z$  users in  $\mathcal{X}$ .
21:    Transmit: Now do the following  $\mu_z$  transmissions
        $\bigoplus_{m \in \mathcal{X}} W_{d_m, y_{m,s}}, \forall s \in [\mu_z]$ 
22:    Note that there are  $\mu_z$  transmissions for  $\mathcal{X}$ .
23:  end for

```

runs $\binom{b_r}{2}^z$ times. The transmit step of the delivery algorithm runs μ_z times. So we see that totally there are $\mu_z \binom{b_r}{2}^z \binom{r}{z}$ transmissions and subpacketization level is v . Hence, the result from the definition of rate. ■

Lemma 4: The number of users benefited in each transmission, known in the literature as coding gain and denoted by g , is given by $g = 2^z$.

Proof: From second **for** of the delivery algorithm it can be observed that, there are totally 2^z users benefited from a transmission. So the coding gain by definition is 2^z . ■

IV. PERFORMANCE COMPARISON

For both the MaN and the SPE schemes, the number of users and the number of caches are equal. Whereas, for the schemes proposed in this paper the number of users K and the number of caches b need not be same. So, when we compare our scheme with the MaN scheme, we will keep the number of caches and fraction of each file stored at each cache

TABLE I
COMPARISON BETWEEN MaN AND PROPOSED SCHEME FOR THE CLASS OF CROSS RESOLVABLE DESIGN DERIVED FROM AFFINE GEOMETRY FOR THE CASE $z = 2$, WHERE q IS A PRIME OR PRIME POWER AND $m \geq 2$

Parameters	MaN Scheme	Proposed Scheme
Number of Caches	$\frac{q(q^m - 1)}{q - 1}$	$\frac{q(q^m - 1)}{q - 1}$
Number of caches a user has access to	1	2
Fraction of each file at each cache $\left(\frac{M}{N}\right)$	$\frac{1}{q}$	$\frac{1}{q}$
Number of Users (K)	$\frac{q(q^m - 1)}{q - 1}$	$\frac{q^3(q^m - 1)(q^{m-1} - 1)}{2(q - 1)^2}$
Subpacketization level (F)	$\left(\frac{q(q^m - 1)/q - 1}{(q^m - 1)/q - 1}\right)$	q^m
Rate (R)	$\frac{(q^m - 1)(q - 1)}{q^m + q - 2}$	$\frac{q(q^m - 1)(q^{m-1} - 1)}{8}$
Rate per user $\left(\frac{R}{K}\right)$	$\frac{(q - 1)^2}{q(q^m + q - 2)}$	$\frac{(q - 1)^2}{4q^2}$
Gain (g)	$\frac{q^m - 1}{q - 1} + 1$	4

same. When we compare our scheme with the SPE scheme, the number of caches and the number of caches a user is connected to is kept same.

A. Comparison of Our Schemes Obtained From CRDs From Affine Resolvable BIBDs

In this subsection, we focus on the schemes obtained using the resolvable designs from affine resolvable balanced incomplete block designs (BIBDs) [15](Chapter 5, Resolvable BIBDs) which are CRDs. We will compare the resulting schemes with the MaN scheme. One such infinite class of affine resolvable BIBDs is derived from affine geometry. Such CRDs exists for all q and m , where q is a prime or a power of a prime number and $m \geq 2$. For any such q and m , the CRD resulting from an affine resolvable BIBD has the number of points $v = q^m$, the number of points in a block $k = q^{m-1}$, the number of blocks $b = \frac{q(q^m - 1)}{q - 1}$, the number of parallel classes $r = \frac{q^m - 1}{q - 1}$. It is known that any two blocks drawn from different parallel classes always intersect at exactly $\frac{k^2}{v}$ points [15] i.e., $z = 2$ and $\mu_z = q^{m-2}$.

For the multi-access coded caching scheme from the CRDs with parameters q and m , we have $K = (b_r)^z \binom{r}{z} = \frac{q^3(q^m - 1)(q^{m-1} - 1)}{2(q - 1)^2}$ users, $b = \frac{q(q^m - 1)}{q - 1}$ caches each having a cache size of $\frac{M}{N} = \frac{k}{v} = \frac{1}{q}$ and $\frac{M'}{N} = \frac{(2q - 1)}{q^2}$.

1) *Comparison With the MaN Scheme:* Since $\frac{bM}{N} = \frac{b}{q} = \frac{q^m - 1}{q - 1}$ is a integer, we have the corresponding MaN scheme with N files, $\frac{q(q^m - 1)}{q - 1}$ number of users and each user has $\frac{1}{q}$ fraction of each file stored at its corresponding cache. The two schemes have been compared by keeping the number of caches and fractions of each file at each cache equal in Table I. It is seen that our scheme performs better than the MaN scheme in terms of the number of users supported and the subpacketization levels at the cost of increased rate and decreased gain.

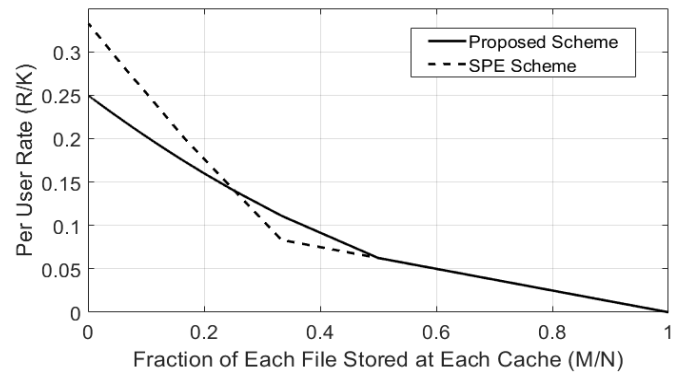


Fig. 2. Per user rate for the codes in Table II.

2) *Comparison With the SPE Scheme:* For comparison with SPE scheme we will use $m = 2$. The special case of $m = 2$ is the class of CRDs from affine planes. For the subpacketization value in SPE scheme to be an integer, given by $\frac{K(K - 2z + 2)}{4}$ [8], where K is the number of users and z is the number of caches a user has access to, we consider the SPE scheme with N files, $q(q + 1)$ users, $\frac{M}{N} = \frac{2}{q(q + 1)}$ fraction of each file stored at each cache and each user has access to exactly $z = 2$ caches, which gives a subpacketization level equal to $\frac{q(q - 1)(q + 1)(q + 2)}{4}$. Since $\frac{b(b - 2z + 2)}{4} = \frac{q(q - 1)(q + 1)(q + 2)}{4}$ is an integer, we have the comparable SPE scheme with N files, $q(q + 1)$ number of users. The comparison is given in Table II and also shown in Fig.3. The rate expression in [8] is complicated (given in Theorem 1, page 2 of [8]). So for comparison with the proposed scheme we plot the rate versus the number of users for the two schemes in Fig. 3. In Fig.2 we plot the per user rate $\frac{R}{K}$ against the fraction of each file stored at each cache $\frac{M}{N}$ and see that our scheme performs better than the SPE scheme for small cache sizes and matches the performance at large cache sizes.

Another such infinite class of affine resolvable BIBDs is derived from Hadamard matrices. Such CRDs exists for all m ,

TABLE II
COMPARISON BETWEEN SPE AND PROPOSED SCHEME FOR THE CLASS OF CRDs FROM AFFINE PLANES WHERE q IS A PRIME OR PRIME POWER

Parameters	SPE Scheme	Proposed Scheme
Number of Caches	$q(q + 1)$	$q(q + 1)$
Number of Caches a user has access to (z)	2	2
Number of Users (K)	$q(q + 1)$	$\frac{q^3(q+1)}{2}$
Fraction of each file at each cache ($\frac{M}{N}$)	$\frac{2}{q(q+1)}$	$\frac{1}{q}$
Fraction of each file each user has access to	$\frac{4}{q(q+1)}$	$\frac{2q-1}{q^2}$
Subpacketization level	$\frac{q(q-1)(q+1)(q+2)}{4}$	q^2
Rate (R)	See Fig. 3	See Fig.3
Gain (g)	between 3 and 4	4

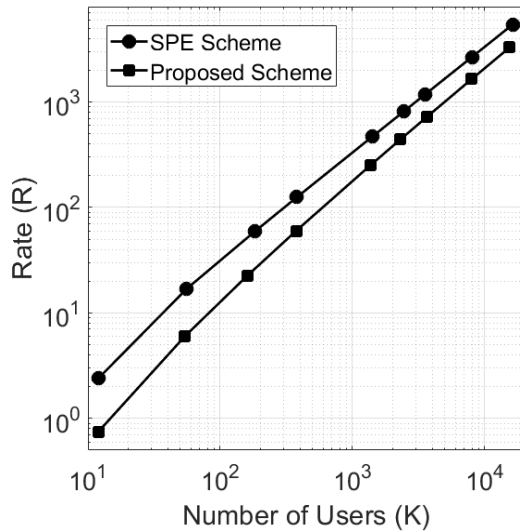


Fig. 3. Rate R for the schemes in Table II.

if there exists a Hadamard matrix of order $4m$. For an m , the CRD resulting from an affine resolvable BIBD has the number of points $v = 4m$, the number of points in a block $k = 2m$, the number of blocks $b = 2(4m - 1)$, the number of parallel classes $r = 4m - 1$. It is known that any two blocks drawn from different parallel classes always intersect at exactly $\frac{k^2}{v}$ points [15] i.e., $z = 2$ and $\mu_z = m$.

For the multi-access coded caching scheme from the CRDs with parameter q and m , with N files we have $K = (b_r)^z \binom{r}{z} = 4(2m - 1)(4m - 1)$ users, $b = 2(4m - 1)$ caches each having a cache size of $\frac{M}{N} = \frac{k}{v} = \frac{1}{2}$ and $\frac{M'}{N} = \frac{3}{4}$.

Since $\frac{bM}{N} = \frac{b}{2} = 4m - 1$ is an integer, we have the corresponding MaN scheme with N files, $2(4m - 1)$ number of users and each user has $\frac{1}{2}$ fraction of each file stored at its corresponding cache. The two schemes have been compared by keeping the number of caches and fractions of each file at each cache equal in Table III. It is seen that our scheme performs better than the MaN scheme in terms of the number

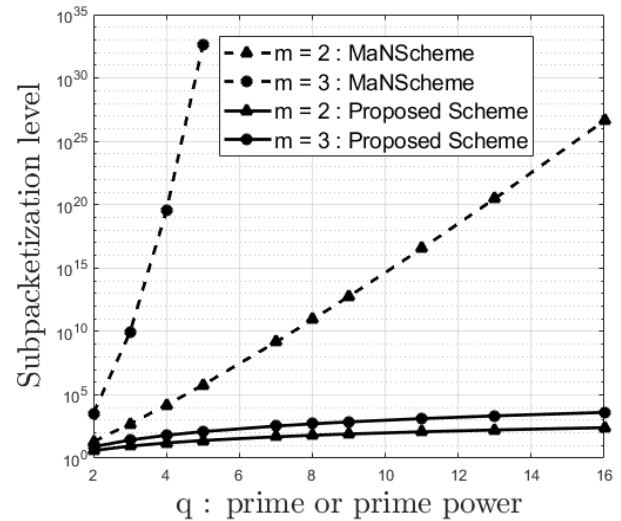


Fig. 4. Comparing subpacketization level for $m = 2$ and $m = 3$ between MaN and Proposed scheme for the class of resolvable designs derived from affine resolvable BIBD's, where q is a prime or prime power.

of users supported and the subpacketization level at the cost of increased rate and decreased gain.

Fig. 4 shows the subpacketization v versus the parameter q . The plot in Fig. 5 shows the per user rate $\frac{R}{K}$ versus the fraction of each file stored at each cache $\frac{M}{N}$. Since $\frac{M}{N} = \frac{1}{q}$ and $\frac{R}{K} = \frac{(q-1)^2}{(q)(q^m+q-2)}$ in case of MaN scheme, $\frac{R}{K} = \frac{(q-1)^2}{4q^2}$ in case of proposed scheme is a function of q , we plot $\frac{R}{K}$ vs $\frac{M}{N}$ keeping m constant for different values of q , where q is a prime or prime power number. The points between any two achievable $(\frac{M}{N}, \frac{R}{K})$ pairs through this scheme, can be obtained as well through memory sharing. When the cross intersection number is minimal, i.e., $\mu_2 = 1$, we see that our scheme performs better than MaN scheme.

B. Two Examples Outperforming the MaN Scheme

In this subsection, we present two instances of our schemes using CRDs (not from affine planes) which outperform the

TABLE III
COMPARISON BETWEEN MaN AND PROPOSED SCHEME FOR THE CLASS OF CROSS RESOLVABLE DESIGN DERIVED FROM HADAMARD MATRICES FOR THE CASE $z = 2$

Parameters	MaN Scheme	Proposed Scheme
Number of Caches	$2(4m - 1)$	$2(4m - 1)$
Number of caches a user has access to	1	2
Fraction of each file at each cache $\left(\frac{M}{N}\right)$	$\frac{1}{2}$	$\frac{1}{2}$
Number of Users (K)	$2(4m - 1)$	$4(2m - 1)(4m - 1)$
Subpacketization level (F)	$\binom{2(4m-1)}{4m-1}$	$4m$
Rate (R)	$\frac{(4m-1)}{4m}$	$\frac{(2m-1)(4m-1)}{4}$
Rate per user $\left(\frac{R}{K}\right)$	$\frac{1}{8m}$	$\frac{1}{16}$
Gain (g)	$4m$	4

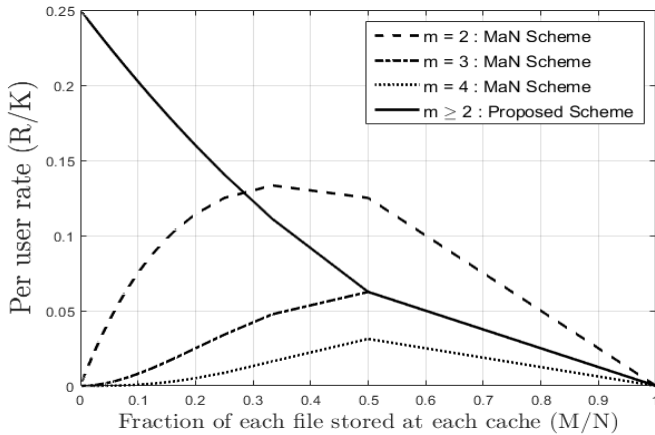


Fig. 5. Performance Analysis between MaN and Proposed scheme for the class of cross resolvable design derived from affine resolvable BIBD's for the case $z = 2$, where q is a prime or prime power and $m \geq 2$.

MaN scheme in all aspects, namely in rate, gain as well as in subpacketization level simultaneously. The first instance is our scheme obtained using the CRD given in Example 3 and the second one is that obtained from the CRD given in Example 4. The performances of these two schemes in comparison with the comparable MaN schemes is presented in Table IV. The performance of our scheme for these two instances is shown pictorially in Fig. 6

C. Comparison With SPE Scheme

In this subsection, we show some examples (not necessarily from affine planes) for the comparison between two schemes.

Example 7: Consider the resolvable design with parameters $v = 8$, $b = 8$, $r = 4$, $k = 4$ and $\mu_2 = 2$ specified as follows. $X = \{1, 2, 3, 4, 5, 6, 7, 8\}$, and

$\mathcal{A} = \{\{1, 2, 3, 4\}, \{5, 6, 7, 8\}, \{1, 2, 5, 6\}, \{1, 3, 5, 7\}, \{2, 4, 6, 8\}, \{3, 4, 7, 8\}, \{1, 4, 5, 8\}, \{2, 3, 6, 7\}\}$. The parallel classes are

$$\mathcal{P}_1 = \{\{1, 2, 3, 4\}, \{5, 6, 7, 8\}\},$$

$$\mathcal{P}_2 = \{\{1, 2, 5, 6\}, \{3, 4, 7, 8\}\},$$

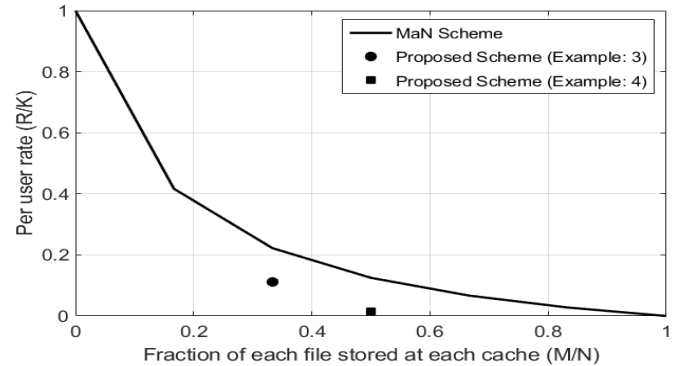


Fig. 6. Pictorial representation of Table IV.

$$\mathcal{P}_3 = \{\{2, 4, 6, 8\}, \{1, 3, 5, 7\}\} \text{ and}$$

$$\mathcal{P}_4 = \{\{1, 4, 5, 8\}, \{2, 3, 6, 7\}\}.$$

The comparison between the SPE scheme and proposed scheme in Table V shows that more users can be supported in a multi-access setup, and certain choices of cross resolvable designs can yield better subpacketization levels and even better gains than the comparable SPE scheme at the cost of increased storage in each cache.

V. SCHEMES FROM RESOLVABLE DESIGNS ($z = 1$)

So far, we have considered the multi-access coded caching problems from CRDs where a user is associated with z , $z \in \{2, 3, \dots, r\}$ distinct caches such that these z caches correspond to z blocks from distinct parallel classes. In this section we describe multi-access coded caching schemes from resolvable designs that are not CRD. For such resolvable designs we define $z = 1$. We see that this case is nothing but the case where each user is associated with its own dedicated cache, i.e., the number of users is equal to the number of blocks.

For a given resolvable design (X, \mathcal{A}) with v points, r parallel classes, b blocks of size k each, b_r blocks in each parallel class, now we have the coded caching problem with N files

TABLE IV
COMPARISON BETWEEN MaN AND PROPOSED SCHEME

Parameters	CRD of Example 3		CRD of Example 4	
	MaN Scheme	Proposed Scheme	MaN Scheme	Proposed Scheme
Number of Caches (b)	6	6	6	6
Number of Caches a user has access to	1	2	1	3
Number of Users (K)	6	9	6	8
Subpacketization level (F)	15	9	20	8
Fraction of each file at each cache ($\frac{M}{N}$)	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{2}$
Fraction of each file each user has access to	$\frac{1}{3}$	$\frac{5}{9}$	$\frac{1}{2}$	$\frac{7}{8}$
Rate (R)	$\frac{4}{3}$	1	$\frac{3}{4}$	$\frac{1}{8}$
Gain (g)	3	4	4	8

TABLE V
COMPARISON BETWEEN SPE AND THE PROPOSED SCHEME

Parameters	CRD of Example 5		CRD of Example 4	
	SPE Scheme	Proposed Scheme	SPE Scheme	Proposed Scheme
Number of Caches (b)	8	8	6	6
Number of Caches a user has access to	2	2	2	2
Number of Users (K)	8	24	6	12
Subpacketization level (F)	12	8	6	8
Fraction of each file at each cache ($\frac{M}{N}$)	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{2}$
Fraction of each file each user has access to	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{2}{3}$	$\frac{3}{4}$
Rate (R)	$\frac{7}{6}$	$\frac{3}{2}$	$\frac{1}{2}$	$\frac{3}{4}$
Gain (g)	$\frac{25}{7}$	4	4	4

in server database, b be the number of caches, $K = \binom{r}{z} b_r^z = r b_r = b$ users, $\frac{M}{N} = \frac{k}{v}$ fraction of each file at each cache and subpacketization level v .

A. Placement and Delivery Phase

The placement phase remains the same as described in Section II for schemes from CRDs and the Delivery steps are same as in as **Algorithm 1** with taking $\mu_1 = k$. It can be checked that the proof of correctness of **Algorithm 1** holds for the case included with $z = 1$ and $\mu_1 = k$. Now Theorem 1 and Lemma 4 can be restated for the case $z = 1$ as follows with the proofs remaining valid.

Theorem 2: For multi-access coded caching schemes from resolvable designs with $z = 1$, N files and K users each with access to one cache of size M in the considered caching system, with $N \geq K$, the proposed scheme achieves the worst case rate R with distinct demands given by $R = \frac{r k \binom{b_r}{z}}{v}$ and

the number of users benefited in each transmission i.e gain, is 2.

B. Performance Comparison: MaN Scheme Vs Schemes From Resolvable Designs From Affine Planes With $z = 1$

Consider the coded caching schemes from the resolvable design derived from the affine plane treating them with $z = 1$. We have $b = K = n(n+1)$ and $\frac{M}{N} = \frac{1}{n}$. Since $\frac{KM}{N} = n+1$ is an integer, we have the corresponding MaN scheme with N files, $n(n+1)$ number of users and each user having $\frac{1}{n}$ fraction of each file stored at its corresponding cache. Comparison of proposed scheme with MaN scheme is given in Table VI.

C. Comparison of Proposed Scheme for Different Values of z

In this subsection, we analyze the proposed scheme for different values of $z \in \{2, 3, \dots, r\}$. We will analyze a CRD

TABLE VI
COMPARISON BETWEEN MaN AND PROPOSED SCHEME FOR THE CLASS OF RESOLVABLE DESIGNS DERIVED FROM AFFINE PLANES WITH $z = 1$

Parameters	MaN Scheme	Proposed Scheme
Number of Caches	$n(n+1)$	$n(n+1)$
Fraction of each file at each cache $\left(\frac{M}{N}\right)$	$\frac{1}{n}$	$\frac{1}{n}$
Number of Users (K)	$n(n+1)$	$n(n+1)$
Subpacketization level (F)	$\binom{n(n+1)}{n+1}$	n^2
Rate (R)	$\frac{(n+1)(n-1)}{n+2}$	$\frac{(n+1)(n-1)}{2}$
Gain (g)	$n+2$	2

TABLE VII
COMPARISON FOR DIFFERENT VALUES OF z FOR A RESOLVABLE DESIGN GIVEN IN EXAMPLE 8

Parameters	$z = 1$	$z = 2$	$z = 3$
Number of Caches (b)	9	9	9
Subpacketization level (F)	27	27	27
Number of Users (K)	9	27	27
Fraction of each file at each cache $\left(\frac{M}{N}\right)$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$
Fraction of each file each user has access to $\left(\frac{M'}{N}\right)$	$\frac{1}{3}$	$\frac{5}{9}$	$\frac{19}{27}$
Rate (R)	3	3	1
Rate per user $\left(\frac{R}{K}\right)$	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{27}$
Gain (g)	2	4	8

for different values of z with respect to rate per user $\left(\frac{R}{K}\right)$.

$$\frac{\left(\frac{R}{K}\right)_z}{\left(\frac{R}{K}\right)_{z-1}} = \frac{1}{2} \left(1 - \frac{k}{v}\right), \quad z \in \{3, 4, \dots, r\} \quad (1)$$

$$\frac{\left(\frac{R}{K}\right)_z}{\left(\frac{R}{K}\right)_{z-1}} = \frac{\mu_2}{2k} \left(\frac{v}{k} - 1\right), \quad z = 2 \quad (2)$$

Example 8: Consider the resolvable design with parameters $v = 27$, $b = 9$, $r = 3$, $k = 9$, $\mu_2 = 3$ and $\mu_3 = 1$. In Table VII, we compare the resolvable design for different values of z .

Example 9: Consider the resolvable design with parameters $v = 16$, $b = 8$, $r = 4$ and $k = 8$ with $X = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16\}$, and $\mathcal{A} = \{1, 2, 3, 4, 5, 6, 7, 8\}, \{9, 10, 11, 12, 13, 14, 15, 16\}, \{1, 2, 3, 4, 9, 10, 11, 12\}, \{5, 6, 7, 8, 13, 14, 15, 16\}, \{1, 2, 5, 6, 9, 10, 13, 14\}, \{3, 4, 7, 8, 11, 12, 15, 16\}, \{1, 3, 5, 7, 9, 11, 13, 15\}, \{2, 4, 6, 8, 10, 12, 14, 16\}$.

The parallel classes are

$$\mathcal{P}_1 = \{\{1, 2, 3, 4, 5, 6, 7, 8\}, \{9, 10, 11, 12, 13, 14, 15, 16\}\},$$

$$\mathcal{P}_2 = \{\{1, 2, 3, 4, 9, 10, 11, 12\}, \{5, 6, 7, 8, 13, 14, 15, 16\}\},$$

$$\mathcal{P}_3 = \{\{1, 2, 5, 6, 9, 10, 13, 14\}, \{3, 4, 7, 8, 11, 12, 15, 16\}\},$$

$$\mathcal{P}_4 = \{\{1, 3, 5, 7, 9, 11, 13, 15\}, \{2, 4, 6, 8, 10, 12, 14, 16\}\}$$

It can be easily observed that $\mu_2 = 4$, $\mu_3 = 2$ and $\mu_4 = 1$. In Table VIII, we compare the resolvable design for different values of z .

From Table VII, we can see that though one would expect more z , to give more improvement in overall rate, this need not be the case. For $z = 1$ and for $z = 2$ the overall rate remains same, though gain improves from 2 to 4. This is because the number of users supported improved from 9 to 27. Comparing in terms of rate per user, in both the examples we see that as z increases, rate per user decreases. At the same time, it need not be the case that increasing value of z , yields increase in number of users that can be supported. This is evident from Table VIII. We see a drop in number of users supported from 32 to 16 when z changes from $z = 3$ to $z = 4$.

D. Subpacketization

Lemma 5: For a given resolvable design with parameters v , b , r , k and for the chosen value of z , with K being the number of users supported, the subpacketization level v is given by

$$v = k \left(\frac{K}{r}\right)^{1/z}. \quad (3)$$

Proof: Since for a given resolvable design and for a chosen value of z , number of users K is equal to $K = \binom{r}{z} (b_r)^z = \binom{r}{z} \left(\frac{v}{k}\right)^z$ rearranging which gives (3). ■

From the above lemma, it is not hard to see that the subpacketization level v grows slower than the number of users K i.e., the subpacketization level growth is sublinear with K for the different classes of cross resolvable designs discussed in this paper.

VI. DISCUSSION

We have identified a special class of resolvable designs called cross resolvable designs which lead to multi-access coded caching schemes. While combinatorial designs have been used in the literature for coded caching problems ours is the first work to use them for multi-access coded caching. Our results indicate that using CRDs in multi-access setups can help attain gains beyond cache redundancy at low subpacketization levels while supporting a large number of users. Our scheme outperforms MaN scheme in terms of rate per user, gains and subpacketization simultaneously. It can perform better than SPE scheme in terms of users supported, rate per user and subpacketization levels which are important design parameters for any coded caching scheme. Our scheme also supports a wide range of choices of KM/N as opposed to SPE scheme. We have shown that the schemes presented in the paper using resolvable designs from affine planes perform better than the MaN scheme for large memory sizes using the metric per-user-rate. This is the only class of resolvable designs that we could identify which is cross resolvable. It will be interesting to construct or identify new cross resolvable designs and study the performance of the resulting multi-access coded caching schemes.

TABLE VIII
COMPARISON FOR DIFFERENT VALUES OF z FOR A RESOLVABLE DESIGN GIVEN IN EXAMPLE 9

Parameters	$z = 1$	$z = 2$	$z = 3$	$z = 4$
Number of Caches (b)	8	8	8	8
Subpacketization level (F)	16	16	16	16
Number of Users (K)	8	24	32	16
Fraction of each file at each cache $\left(\frac{M}{N}\right)$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
Fraction of each file each user has access to $\left(\frac{M'}{N}\right)$	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{7}{8}$	$\frac{15}{16}$
Rate (R)	2	$\frac{3}{2}$	$\frac{1}{2}$	$\frac{1}{16}$
Rate per user $\left(\frac{R}{K}\right)$	$\frac{1}{4}$	$\frac{1}{16}$	$\frac{1}{64}$	$\frac{1}{256}$
Gain (g)	2	4	8	16

APPENDIX

PROOF OF CORRECTNESS OF THE DELIVERY ALGORITHM

The proof of correctness of the delivery phase given by **Algorithm 1** is provided by the sequence of the following three lemmas.

Lemma 6: Let Y_m , $m \in \mathcal{K}$ be the set of indices of subfiles accessible to the m^{th} user. It is easily seen that $|Y_m| = \frac{vM'}{N}$. The set of subfiles which is available with every user in \mathcal{X} , other than m is $\bigcap_{t \in \mathcal{X} \setminus m} Y_t$, $\forall m \in \mathcal{X}$. Consider the transmission corresponding to the set \mathcal{X} and a user $m \in \mathcal{X}$ as in **Algorithm 1**. Then the following equality holds.

$$f_m = \bigcap_{t \in \mathcal{X} \setminus m} Y_t, \forall m \in \mathcal{X} \quad (4)$$

Proof: In **Algorithm 1** consider the combination of $2z$ blocks (caches)

$C_{1,i_1}, C_{1,j_1}, C_{2,i_2}, C_{2,j_2}, \dots, C_{z,i_z}, C_{z,j_z}$, where $i_s, j_s \in [b_r]$ and $j_s \neq i_s, \forall s \in [z]$ and the m^{th} user which has access to z blocks $C_{1,a_1}, C_{2,a_2}, \dots, C_{z,a_z}$, where $a_s \in \{i_s, j_s\}$, $s \in [z]$. The sequence of equations (5) to (13), as shown at the bottom of the next page, constitute the proof for (4). ■

Lemma 7: At the end of each transmission corresponding to set \mathcal{X} in **Algorithm 1**, each user is able to decode one sub-file.

Proof: Consider a user m , $m \in \mathcal{X}$ and any user m' , $m' \neq m$, $m' \in \mathcal{X}$. The set $f_{m'}$ represents the set of subfiles which is available with every user in \mathcal{X} , other than m' . From Lemma 6 we know that the user m has access to all the subfiles in $f_{m'}$. Consider the transmission $\bigoplus_{m \in \mathcal{X}} W_{d_m, y_{m,s}}$, $s \in [\mu_z]$ corresponding to set \mathcal{X} . The m^{th} user is able to get the

subfile $W_{d_m, y_{m,s}}$ from this transmission since it has every other subfile $W_{d_{m'}, y_{m',s}}$, $\forall m' \in \mathcal{X}$, $m' \neq m$. ■

Lemma 8: At the end of all the transmissions in **Algorithm 1** each user is able get all subfiles of the demanded file W_{d_m} , $m \in \mathcal{K}$.

Proof: In **Algorithm 1** it can noted that there are in total $(b_r - 1)^z$ transmissions from which the m^{th} user gets $\mu_z(b_r - 1)^z$ subfiles. Now consider a combination of $2z$ blocks (caches) given as $C_{1,i_1}, C_{1,j_1}, C_{2,i_2}, C_{2,j_2}, \dots, C_{z,i_z}, C_{z,j_z}$, where $i_s, j_s \in [b_r]$ and $j_s \neq i_s, \forall s \in [z]$, and the m^{th} user having access to z blocks (caches), $C_{1,a_1}, C_{2,a_2}, \dots, C_{z,a_z}$, where $a_s \in \{i_s, j_s\}$, $\forall s \in [z]$. We have $Y_m = C_{1,a_1} \cup C_{2,a_2} \cup \dots \cup C_{z,a_z}$ and the m^{th} user is able to receive subfile indices of the demanded file W_{d_m} from the transmission corresponding to $2z$ caches (considered above) given by

$$\bigcap_{\substack{l_s = \{i_s, j_s\}, \forall s \in [z] \\ (l_1, l_2, \dots, l_z) \neq (a_1, a_2, \dots, a_z)}} \{C_{1,l_1} \cup C_{2,l_2} \cup \dots \cup C_{z,l_z}\}$$

which, using Lemma 6 is same as $\{C_{1,e_1} \cap C_{2,e_2} \cap \dots \cap C_{z,e_z}\}$ where, $e_s = \{i_s, j_s\} \setminus a_s$, $\forall s \in [z]$. In order to find subfile indices that m^{th} user get from all $(b_r - 1)^z$ transmissions, we have to vary the value of e_s such that $e_s \neq a_s$, $\forall s \in [z]$.

The m^{th} user is able to receive subfile indices given by $\bigcup_{\substack{e_s = 1, \\ e_s \neq a_s \\ \forall s \in [z]}}^{b_r} \{C_{1,e_1} \cap C_{2,e_2} \cap \dots \cap C_{z,e_z}\}$. In addition to this

using Y_m , the m^{th} user gets the subfile indices shown in the sequence of expressions numbered (14) to (20), as shown at the top of the page 13. Notice that the last expression in (17) is the union of all the blocks in a parallel class. So from the property of resolvable designs the above set is equal to the set

$$\bigcap_{t \in \mathcal{X} \setminus m} Y_t = \bigcap_{\substack{l_s = \{i_s, j_s\}, \forall s \in [z] \\ (l_1, l_2, \dots, l_z) \neq (a_1, a_2, \dots, a_z)}} \{C_{1, l_1} \cup C_{2, l_2} \cup \dots \cup C_{z, l_z}\} \quad (5)$$

$$= \left\{ \bigcap_{\substack{l_s = \{i_s, j_s\}, \\ \forall s \in [2, z]}} \{C_{1, e_1} \cup C_{2, l_2} \cup \dots \cup C_{z, l_z}\} \right\} \cap \left\{ \bigcap_{\substack{l_s = \{i_s, j_s\}, \\ \forall s \in [2, z] \\ (l_2, l_3, \dots, l_z) \neq \\ (a_2, a_3, \dots, a_z)}} \{C_{1, a_1} \cup C_{2, l_2} \cup \dots \cup C_{z, l_z}\} \right\}$$

$$= C_{1, e_1} \cap \left\{ \bigcap_{\substack{l_s = \{i_s, j_s\}, \\ \forall s \in [2, z] \\ (l_2, l_3, \dots, l_z) \neq \\ (a_2, a_3, \dots, a_z)}} \{C_{1, a_1} \cup C_{2, l_2} \cup C_{3, l_3} \cup \dots \cup C_{z, l_z}\} \right\} \quad (6)$$

$$= C_{1, e_1} \cap \left\{ C_{1, a_1} \cup \left\{ \bigcap_{\substack{l_s = \{i_s, j_s\}, \\ \forall s \in [2, z] \\ (l_2, l_3, \dots, l_z) \neq \\ (a_2, a_3, \dots, a_z)}} \{C_{2, l_2} \cup C_{3, l_3} \cup \dots \cup C_{z, l_z}\} \right\} \right\} \quad (7)$$

$$= C_{1, e_1} \cap \left\{ \bigcap_{\substack{l_s = \{i_s, j_s\}, \\ \forall s \in [2, z] \\ (l_2, l_3, \dots, l_z) \neq \\ (a_2, a_3, \dots, a_z)}} \{C_{2, l_2} \cup C_{3, l_3} \cup \dots \cup C_{z, l_z}\} \right\} \quad (8)$$

$$= \{C_{1, e_1} \cap C_{2, e_2}\} \cap \left\{ \bigcap_{\substack{l_s = \{i_s, j_s\}, \\ \forall s \in [3, z] \\ (l_3, l_4, \dots, l_z) \neq \\ (a_3, a_4, \dots, a_z)}} \{C_{3, l_3} \cup C_{4, l_4} \cup \dots \cup C_{z, l_z}\} \right\} \quad (9)$$

(proceeding as to arrive at the previous step)

$$= \{C_{1, e_1} \cap C_{2, e_2} \cap C_{3, e_3}\} \cap \left\{ \bigcap_{\substack{l_s = \{i_s, j_s\}, \\ \forall s \in [4, z] \\ (l_4, l_5, \dots, l_z) \neq \\ (a_4, a_5, \dots, a_z)}} \{C_{4, l_4} \cup C_{5, l_5} \cup \dots \cup C_{z, l_z}\} \right\} \quad (10)$$

$$\vdots \quad (11)$$

$$= \{C_{1, e_1} \cap C_{2, e_2} \cap \dots \cap C_{z-1, e_{z-1}}\} \cap \left\{ \bigcap_{\substack{l_z = \{i_z, j_z\}, \\ (l_z) \neq (a_z)}} \{C_{z, l_z}\} \right\} \quad (12)$$

$$= \{C_{1, e_1} \cap C_{2, e_2} \cap \dots \cap C_{z, e_z}\} = f_m. \quad (13)$$

$$Y_m \cup \left\{ \bigcup_{\substack{e_s = 1, \\ e_s \neq a_s \\ \forall s \in [z]}}^{b_r} \{C_{1,e_1} \cap C_{2,e_2} \cap \dots \cap C_{z,e_z}\} \right\} \quad (14)$$

$$= \{C_{1,a_1} \cup C_{2,a_2} \cup \dots \cup C_{z,a_z}\} \cup \bigcup_{\substack{e_s = 1, \\ e_s \neq a_s \\ \forall s \in [z-1]}}^{b_r} \left\{ C_{1,e_1} \cap C_{2,e_2} \cap \dots \cap \left\{ \bigcup_{\substack{e_z = 1 \\ e_z \neq a_z}}^{b_r} C_{z,e_z} \right\} \right\} \quad (15)$$

$$\supseteq \{C_{1,a_1} \cup C_{2,a_2} \cup \dots \cup C_{z-1,a_{z-1}}\} \cup \bigcup_{\substack{e_s = 1, \\ e_s \neq a_s \\ \forall s \in [z-1]}}^{b_r} \{C_{1,e_1} \cap C_{2,e_2} \cap \dots \cap C_{z,a_z}\} \cup \bigcup_{\substack{e_s = 1, \\ e_s \neq a_s \\ \forall s \in [z-1]}}^{b_r} \left\{ C_{1,e_1} \cap C_{2,e_2} \cap \dots \cap \left\{ \bigcup_{\substack{e_z = 1 \\ e_z \neq a_z}}^{b_r} C_{z,e_z} \right\} \right\}$$

$$= \{C_{1,a_1} \cup C_{2,a_2} \cup \dots \cup C_{z-1,a_{z-1}}\} \cup \bigcup_{\substack{e_s = 1, \\ e_s \neq a_s \\ \forall s \in [z-1]}}^{b_r} \left\{ C_{1,e_1} \cap C_{2,e_2} \cap \dots \cap \left\{ \bigcup_{e_z = 1}^{b_r} C_{z,e_z} \right\} \right\} \quad (16)$$

$$= \{C_{1,a_1} \cup C_{2,a_2} \cup \dots \cup C_{z-1,a_{z-1}}\} \cup \bigcup_{\substack{e_s = 1, \\ e_s \neq a_s \\ \forall s \in [z-1]}}^{b_r} \{C_{1,e_1} \cap C_{2,e_2} \cap \dots \cap C_{z-1,e_{z-1}}\} \quad (17)$$

(Continuing with similar steps we get),

$$\supseteq \{C_{1,a_1} \cup C_{2,a_2} \cup \dots \cup C_{z-2,a_{z-2}}\} \cup \bigcup_{\substack{e_s = 1, \\ e_s \neq a_s \\ \forall s \in [z-2]}}^{b_r} \{C_{1,e_1} \cap C_{2,e_2} \cap \dots \cap C_{z-2,e_{z-2}}\} \quad (18)$$

⋮

$$\supseteq \{C_{1,a_1} \cup C_{2,a_2}\} \cup \left\{ \bigcup_{\substack{e_s = 1, \\ e_s \neq a_s \\ \forall s \in [2]}}^{b_r} \{C_{1,e_1} \cap C_{2,e_2}\} \right\} \quad (19)$$

$$\supseteq \{C_{1,a_1}\} \cup \left\{ \bigcup_{\substack{e_1 = 1, \\ e_1 \neq a_1}}^{b_r} C_{1,e_1} \right\} \quad (20)$$

containing all the subfile indices of all the files and therefore it also contains all the subfiles of the demanded file W_{d_m} . ■

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] Q. Yan, M. Cheng, X. Tang, and Q. Chen, "On the placement delivery array design for centralized coded caching scheme," *IEEE Trans. Inf. Theory*, vol. 63, no. 9, pp. 5821–5833, Sep. 2017.
- [3] L. Tang and A. Ramamoorthy, "Coded caching schemes with reduced subpacketization from linear block codes," *IEEE Trans. Inf. Theory*, vol. 64, no. 4, pp. 3099–3120, Apr. 2018.
- [4] J. Wang, M. Cheng, Q. Yan, and X. Tang, "Placement delivery array design for coded caching scheme in D2D networks," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3388–3395, May 2019.
- [5] Q. Yan, M. Wigger, and S. Yang, "Placement delivery array design for combination networks with edge caching," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 1555–1559.
- [6] N. Hachem, S. Karamchandani, and N. Diggavi, "Coded caching for multi-level popularity and access," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 3108–3141, May 2017.
- [7] K. S. Reddy and N. Karamchandani, "On the exact rate-memory trade-off for multi-access coded caching with uncoded placement," in *Proc. Int. Conf. Signal Process. Commun. (SPCOM)*, Bangalore, India, Jul. 2018, pp. 1–5.
- [8] B. Serbetci, E. Parrinello, and P. Elia, "Multi-access coded caching: Gains beyond cache-redundancy," in *Proc. IEEE Inf. Theory Workshop*, Visby, Gotland, Aug. 2019, pp. 1–5.
- [9] K. S. Reddy and N. Karamchandani, "Rate-memory trade-off for multi-access coded caching with uncoded placement," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, 2019, pp. 1232–1236.
- [10] K. Srinivas Reddy and N. Karamchandani, "Rate-memory trade-off for multi-access coded caching with uncoded placement," 2019, *arXiv:1909.01756*. [Online]. Available: <http://arxiv.org/abs/1909.01756>
- [11] E. Parrinello, A. Ünsal, and P. Elia, "Coded caching with shared caches: Fundamental limits with uncoded prefetching," 2018, *arXiv:1809.09422*. [Online]. Available: <http://arxiv.org/abs/1809.09422>

- [12] A. M. Ibrahim, A. A. Zewail, and A. Yener, "Coded placement for systems with shared caches," in *Proc. ICC - IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, May 2019, pp. 1–6.
- [13] B. Asadi and L. Ong, "Centralized caching with shared caches in heterogeneous cellular networks," in *Proc. IEEE 20th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Cannes, France, Jul. 2019, pp. 1–5.
- [14] L. Tang and A. Ramamoorthy, "Coded caching for networks with the resolvability property," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Barcelona, Jul. 2016, pp. 420–424.
- [15] D. R. Stinson, *Combination Designs: Constructions Analysis*. Cham, Switzerland: Springer, 2003.
- [16] S. Agrawal, K. V. Sushena Sree, and P. Krishnan, "Coded caching based on combinatorial designs," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, Jul. 2019, pp. 1227–1231.
- [17] S. Agrawal, K. V. Sushena Sree, and P. Krishnan, "Coded caching based on combinatorial designs," 2019, *arXiv:1901.06383*. [Online]. Available: <http://arxiv.org/abs/1901.06383>



Digvijay Katyal was born in Uttarakhand, India. He received the M.Tech. degree in communication and networks from the Department of Electrical Communication Engineering, Indian Institute of Science, Bengaluru, in 2020. He is currently working as a Senior Engineer at Samsung Semiconductor India R&D Centre. His primary research interests include coded caching, D2D communication, and wireless communication.



Pooja Nayak Muralidhar was born in Kerala, India. She received the B.Tech. degree in electronics and communication engineering from the National Institute of Technology, Calicut, in 2016. She is currently pursuing the Ph.D. degree with the Department of Electrical Communication Engineering, Indian Institute of Science, Bengaluru. She worked as an Engineer at Schneider Electric India Pvt. Ltd. Her primary research interests include coded caching and coded computation. She is a recipient of the Prime Minister's Research Fellowship (2020).



B. Sundar Rajan (Fellow, IEEE) was born in Tamil Nadu, India. He received the B.Sc. degree in mathematics from Madras University, Madras, India, in 1979, the B.Tech. degree in electronics from the Madras Institute of Technology, Madras, in 1982, and the M.Tech. and Ph.D. degrees in electrical engineering from IIT Kanpur, Kanpur, in 1984 and 1989, respectively.

He was a Faculty Member of the Department of Electrical Engineering, IIT Delhi, New Delhi, from 1990 to 1997. He has been a Professor with the Department of Electrical Communication Engineering, Indian Institute of Science, Bengaluru, since 1998. His primary research interests include space-time coding for MIMO channels, distributed space-time coding and cooperative communication, coding for multiple-access and relay channels, and network coding. He is a fellow of the Indian National Academy of Engineering, the Indian National Science Academy, the Indian Academy of Sciences, and the National Academy of Sciences, India. He is a J.C. Bose National Fellow (2016–2020) and a member of the American Mathematical Society. He is a recipient of Prof. Rustum Choksi Award by I.I.Sc for Excellence in Research in Engineering for the year 2009, the IETE Pune Center's S.V.C Aiya Award for Telecom Education in 2004, and the Best Academic Paper Award at IEEE WCNC 2011. He has served as the Technical Program Co-Chair of IEEE Information Theory Workshop (ITW'02) held in Bengaluru, in 2002. He was an Editor of IEEE WIRELESS COMMUNICATIONS LETTERS (2012–2015), an Associate Editor of Coding Theory for IEEE TRANSACTIONS ON INFORMATION THEORY (2008–2011 and 2013–2015), and an Editor of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS (2007–2011).