

# Multi-access Coded Caching Scheme with Linear Sub-packetization using PDAs

Shanuja Sasi and B. Sundar Rajan  
Indian Institute of Science, Bengaluru  
E-mail: shanuja@iisc.ac.in, bsrajan@iisc.ac.in

**Abstract**—We consider multi-access coded caching problem introduced by Hachem et.al., where each user has access to  $L$  neighboring caches in a cyclic wrap-around fashion. We focus on the deterministic schemes for a specific class of multi-access coded caching problem based on the concept of PDA. We construct new PDAs which specify the delivery scheme for the specific class of multi-access coded caching problem discussed in this paper. For the proposed scheme, the coding gain is larger than that of the state-of-the-art while the sub-packetization level varies only linearly with the number of users. Hence, we achieve a lower transmission rate with the least sub-packetization level compared to the existing schemes.

## I. INTRODUCTION

The drastic increase in the demand for video streaming services is one of the key factors for introducing the coded caching scheme by Maddah-Ali and Niesen [1]. The main objective behind the coded caching scheme is to relieve the traffic burden during peak hours by utilizing the ample cache memories available at the user ends. The set-up consists of a central server having access to a set of  $N$  files of equal size, which is connected through an error-free link to a set of  $K$  users where each user has a cache size of  $M$  files. The proposed  $(K, M, N)$  coded caching scheme operates in two phases. The first is the placement phase where each file is divided into  $F$  equal packets and the cache memories are filled with some of these packets during the off-peak hours. The second phase is the delivery phase when the demands are revealed by the users. Once the demands are known, the server transmits coded symbols of length  $\mathcal{R}$  files over an error-free link to all the users such that all the users are able to meet their demands from the local cache content and the transmitted symbols. The quantity  $\mathcal{R}$  is referred to as the transmission rate.

Coded caching has been extensively studied over the past few years [2]–[9]. The value of  $F$ , which is known as the sub-packetization level, is directly proportional to the complexity of a coded caching scheme. It is well known that there is a tradeoff between the sub-packetization level  $F$  and the transmission rate  $\mathcal{R}$ . In the scheme introduced by Maddah-Ali and Niesen [1], which we refer as MN scheme, the sub-packetization level grows exponentially with respect to the number of users  $K$ , which makes it infeasible for practical implementation. Reducing the sub-packetization level of the coded caching schemes has been a major problem studied during the past few years.

Yan et al. [10], represented a coded caching scheme by an array called Placement Delivery Array (PDA) with an aim

to reduce the sub-packetization level. It is shown that the MN scheme can be represented by a PDA and is optimal among regular PDAs, which is a specific class of PDAs. Although the scheme proposed by Yan et. al. has a lower sub-packetization level compared to the MN scheme at the expense of a slight increase in the transmission rate, the sub-packetization level still increases sub-exponentially with  $K$ . The concept of PDA has been identified as an effective tool to reduce the sub-packetization level and since then various coded caching schemes based on the concept of PDA have been reported [11]–[17].

### A. Multi-access Coded Caching

Unlike the caching models where it is assumed that each user has their own dedicated cache, in this paper, we consider a new model referred to as multi-access coded caching model which was introduced in [18]. In this model, as illustrated in Fig. 1, there is a central server having access to a collection of  $N$  files,  $\mathcal{W} = \{W_0, W_1, W_2, \dots, W_{N-1}\}$ , each of size 1 unit, connected through an error-free link to a set of  $K$  users,  $\mathcal{U} = \{U_0, U_1, \dots, U_{K-1}\}$ . There are  $K$  caches,  $\mathcal{C} = \{C_0, C_1, \dots, C_{K-1}\}$ , each having a storage capacity  $M = N\gamma$  files, where  $\gamma$  is defined as the normalized cache size. Each user can access  $L$  caches in a cyclic wrap-around fashion. The content stored in each cache  $C_\alpha, \alpha \in \{0, 1, \dots, K-1\}$ , is denoted by  $M_\alpha$ . The cache content accessible to the user  $U_\alpha, \alpha \in \{0, 1, \dots, K-1\}$ , is denoted by  $\mathcal{Z}_\alpha$ . The index of the file requested by the user  $U_\alpha, \alpha \in \{0, 1, \dots, K-1\}$ , is denoted by  $d_\alpha$ . The demand vector is denoted by  $\mathbf{d} = (d_0, d_1, \dots, d_{K-1})$ .

Like in the centralized coded caching scheme, multi-access coded caching scheme also operates in two phases: a placement phase and a delivery phase. In the placement phase the caches are filled with parts of the files from the servers' database. In the delivery phase, each user  $U_\alpha$  reveals their demands, which is assumed to be a file from the database. Based on the demand vector and the cache content accessible to each user, the server transmits coded symbols so that each user  $U_\alpha$  can decode the desired file  $W_{d_\alpha}$  using the transmissions as well as the cache content. The overall objective of the multi-access coded caching problem is to obtain placement and delivery schemes so as to minimize the transmission rate, which is defined as the amount of data transmitted by the server in the units of files. The number of users for which each transmission is beneficial is termed as the *coding gain*.

*Notations:*  $[n]$  represents the set  $\{1, 2, \dots, n\}$ ,  $[a, b]$  represents the set  $\{a, a + 1, \dots, b\}$  while  $[a, b)$  represents the set  $\{a, a + 1, \dots, b - 1\}$ . The bit wise exclusive OR (XOR) operation is denoted by  $\oplus$ .  $\lfloor x \rfloor$  denotes the largest integer smaller than or equal to  $x$  and  $\lceil x \rceil$  denotes the smallest integer greater than or equal to  $x$ .  $a|b$  implies  $a$  divides  $b$  and  $a \nmid b$  implies  $a$  does not divide  $b$ , for some integers  $a$  and  $b$ . The transpose of any matrix  $\mathbf{A}$  is represented by  $\mathbf{A}^T$ . For any  $m \times n$  matrix  $\mathbf{A} = (a_{i,j})$ ,  $i \in [0, m - 1]$ ,  $n \in [0, n - 1]$ , the matrix  $\mathbf{A} + b$ , is defined as  $\mathbf{A} + b = (a_{i,j} + b)$ .

## B. Previous Results

The multi-access coded caching scheme was introduced in [18] where the authors have provided a coloring based scheme for the proposed problem. In [19], a new scheme, which we refer as RK scheme, was proposed by mapping of the coded caching problem to the *index coding* problem achieving a transmission rate  $\mathcal{R}_{RK}$  which is less than that achieved in [18].

$$\mathcal{R}_{RK}(\gamma) = \begin{cases} \frac{(K - K\gamma L)^2}{K}, & \forall \gamma \in \left\{ \frac{k}{K} : k \in [0, \lfloor \frac{K}{L} \rfloor] \right\} \\ 0, & \text{for } \gamma = \lfloor \frac{K}{L} \rfloor \frac{1}{K} \end{cases} \quad (1)$$

The sub-packetization level required for RK scheme is  $F_{RK} = \frac{1}{\gamma} \binom{K - K\gamma(L-1) - 1}{K\gamma - 1}$ .

A lower bound  $\mathcal{R}_{lb}$  on the optimal transmission rate-memory trade-off was also derived in [19] for any multi-access coded caching problem when  $L \geq \frac{K}{2}$ . For some special cases, namely for  $L \geq \frac{K}{2}$ , and when  $L = K - 1$ ,  $L = K - 2$ ,  $L = K - 3$  when  $K$  is even, and  $L = K - \frac{K}{g} + 1$  for some positive integer  $g$ , an achievable scheme was proposed separately in [19], which is optimal.

$$\mathcal{R}_{lb}(\gamma) = \begin{cases} K - \left\lfloor \frac{(K-L)(K-L+1)}{2K} \right\rfloor K\gamma, & \text{if } 0 \leq \gamma \leq \frac{1}{K} \\ \frac{(K-L)(K-L+1)}{2K} (2 - K\gamma), & \text{if } \frac{1}{K} \leq \gamma \leq \frac{2}{K} \\ 0, & \text{if } \gamma \geq \frac{2}{K} \end{cases}$$

In [20], the authors have studied two special cases, the first case is when  $K\gamma = 2$ , and the second case is when  $L = \frac{K-1}{K\gamma}$  for any  $K\gamma$ . For  $K\gamma = 2$ , the authors have proposed a novel coded caching scheme, which we call as SPE scheme, that achieves a coding gain that exceeds 2 with a sub-packetization level of  $F_{SPE} = \frac{K(K-2L+2)}{4}$ . The transmission rate achieved using the SPE scheme is

$$\mathcal{R}_{SPE} = \frac{X_1 + X_2}{S}, \quad (2)$$

where  $X_1$ ,  $X_2$  and  $S$  are defined in Eq. (2) in [20]. The coding gain always exceeds 3 and approaches 4 for some values of  $K$  and  $L$ . For the second case considered in [20], i.e., when  $L = \frac{K-1}{K\gamma}$  for any  $K\gamma$ , the authors have provided an achievable scheme which is optimal.

In [21], the authors have proposed an improved scheme for any  $\gamma \in \left\{ \frac{k}{K} : \gcd(k, K) = 1, k \in [1, K] \right\}$ , achieving a rate which is less than or equal to  $\mathcal{R}_{RK}(\gamma)$ .

In [22], the authors have used a *novel transformation* approach to extend the MN scheme to multi-access caching

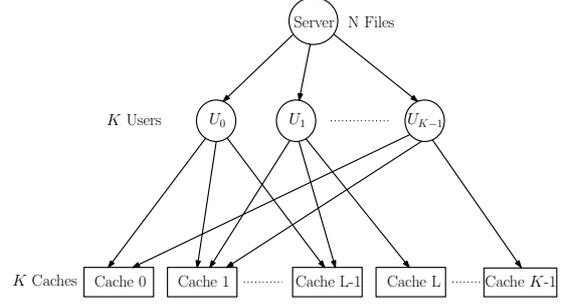


Fig. 1: Multi-access Coded Caching Network [18] consisting of a central server,  $K$  users, and  $K$  caches where each user is connected to  $L$  neighboring caches.

schemes, such that the resulting scheme has the maximum local caching gain and the same coding gain as the related MN scheme. We refer to the scheme proposed in [22] as NT scheme. The coding gain achieved using the NT scheme is  $K\gamma + 1$  with a sub-packetization level of  $F_{NT} = K \binom{K - K\gamma(L-1)}{K\gamma}$ . The transmission rate achieved using the NT scheme is given by

$$\mathcal{R}_{NT} = \frac{K - K\gamma L}{K\gamma + 1}. \quad (3)$$

In [23], the multi-access coded caching problem is divided into a number of special class of index coding problems termed as *Structured Index Coding* problem and using the solutions obtained for the structured index coding problem, a new scheme, which we refer as NK scheme, was proposed for the multi-access coded caching problem which achieves the following rate:

$$\mathcal{R}_{NK} = \frac{\sum_{\mathbf{b} \in \mathcal{B}} \min\{2(K - K\gamma L) + K\gamma - 1 - \bar{\mathbf{b}}, K\}}{F_{NK}(K\gamma + 1)}, \quad (4)$$

where  $\mathcal{B}$  is the collection of all the weak  $K\gamma + 1$  compositions of  $K - K\gamma L - 1$ ,  $\bar{\mathbf{b}}$  denotes the maximum component in the vector  $\mathbf{b} \in \mathcal{B}$  and  $F_{NK} = \frac{1}{\gamma} \binom{K - K\gamma(L-1) - 1}{K\gamma - 1}$  denotes the sub-packetization level. In [24], the authors have proposed a scheme for  $\gamma = \frac{1}{K}$  when  $L \geq \frac{K}{2}$ , which achieves linear sub-packetization level with a slightly higher load than the above schemes. Multi-access coded caching scheme with the number of users not equal to the number of caches was studied in [25] where the authors have identified a special class of resolvable designs called cross resolvable designs.

In [21], we have studied the cases when  $\gamma \in \left\{ \frac{k}{K} : \gcd(k, K) = 1, k \in [1, K] \right\}$ . In this work we have taken up a particular case, when  $\gamma \in \left\{ \frac{k}{K} : k|K, (K - kL + k)|K, k \in [1, K] \right\}$ , which is not covered in [21]. Also, for the cases studied in [21], the worst case sub-packetization level required is  $K^2$  while in this work, the sub-packetization level required is  $K$ .

## C. Our Contributions

Our contributions in this paper are summarized as follows.

- We construct a new class of PDA which we call as *t-cyclic g-regular PDA*, which is used for providing the delivery algorithm in our scheme.
- We prove that the advantage of the proposed scheme is two-fold, in terms of the coding gain as well as the sub-packetization level compared to NT, RK and SPE schemes. The sub-packetization level  $K$  required for our scheme is  $F_{new} = K$ , which varies linearly with the number of users while the sub-packetization level varies sub-exponentially with respect to the number of users in NT, RK and SPE schemes.

## II. MAIN RESULT

We discuss our main result in this section. We characterize our result in Theorem 1.

**Theorem 1.** *Consider a multi-access coded caching scenario with  $N$  files, and  $K$  users, each having access to  $L$  neighboring caches in a cyclic wrap-around way, with each cache having a normalized capacity of  $\gamma$ , where  $\gamma \in \{\frac{k}{K} : k|K, (K - kL + k)|K, k \in [1, K]\}$ . The following transmission rate  $\mathcal{R}_{new}(\gamma)$  is achievable.*

$$\mathcal{R}_{new}(\gamma) = \frac{(K - kL)(K - kL + k)}{2K} \quad (5)$$

Comparison of our scheme with the state-of-the-art is done in Table I. The transmission rate vs  $L$  plot vs  $\gamma$  is obtained for  $K = 24$  in Fig. 2 and the sub-packetization level vs  $L$  vs  $\gamma$  plot is obtained for the same value of  $K$  in Fig. 3. It can be observed that the transmission rate achieved using our scheme for the points considered in Theorem 1 is less compared to that achieved using the RK, and SPE schemes. It can also be observed that the transmission rate achieved using our scheme for all the points considered in Theorem 1, except when  $L = 3, k = 6$  and  $L = 4, k = 4$ , is less compared to that achieved using the NT scheme. The sub-packetization level required is less for our scheme compared to the NT, RK, and SPE schemes. The placement scheme and the delivery algorithm achieving the rate claimed in Theorem 1 is given in Section III.

### A. Comparison with the NT Scheme

The coding gain achieved in our scheme is  $\frac{2K}{K - kL + k}$  while that is  $k + 1$  in the NT scheme. We prove in Lemma 1 that the coding gain achieved in our scheme is more than that achieved using the NT scheme, if  $L > \frac{K(k-1)}{k(k+1)} + 1$ . The sub-packetization level required for our scheme is  $F_{new} = K$  while it is  $F_{NT} = K \binom{K - kL + k}{k}$  for the NT scheme. Hence the sub-packetization level grows linearly with the number of users in our scheme while it grows sub-exponentially with respect to the number of users in the NT scheme. So, the transmission rate achieved in our scheme is less than that achieved using the NT scheme, if  $L > \frac{K(k-1)}{k(k+1)} + 1$ . Moreover, the coding gain increases as  $L$  increases in our scheme while the coding gain is independent of  $L$  in the NT scheme. Hence, as  $L$  increases the gap between the transmission rate between our scheme and the NT scheme increases. Our scheme is better

in terms of both coding gain as well as sub-packetization level compared to the NT scheme.

**Lemma 1.** *For the cases considered in Theorem 1, the coding gain achieved in our scheme,  $\frac{2K}{K - kL + k}$ , is more than the coding gain,  $k + 1$ , achieved using the NT scheme, if  $L > \frac{K(k-1)}{k(k+1)} + 1$ .*

*Proof.* Assume that the coding gain achieved in our scheme is less than or equal to that achieved using the NT scheme if  $L > \frac{K(k-1)}{k(k+1)} + 1$ , i.e.,

$$\begin{aligned} k + 1 &\geq \frac{2K}{K - kL + k} \\ \Rightarrow (k + 1)(K - kL + k) &\geq 2K \\ \Rightarrow Kk + K - k(k + 1)(L - 1) &\geq 2K \\ \Rightarrow k(k + 1)(L - 1) &\leq K(k - 1) \\ \Rightarrow L &\leq \frac{K(k - 1)}{k(k + 1)} + 1 \end{aligned}$$

This contradicts our assumption that  $L > \frac{K(k-1)}{k(k+1)} + 1$ . Therefore the coding gain achieved in our scheme is more than that achieved using the NT scheme, if  $L > \frac{K(k-1)}{k(k+1)} + 1$ .  $\square$

### B. Comparison with the RK Scheme

For the considered cases, the coding gain achieved in our scheme is  $\frac{2K}{K - kL + k}$  while that is  $\frac{K}{K - kL}$  in the RK scheme. We prove in Lemma 2 that our scheme is better in terms of coding gain as compared to the RK scheme. The sub-packetization level required for our scheme is  $F_{new} = K$  while it is  $F_{RK} = \frac{1}{\gamma} \binom{K - K\gamma(L-1)}{K\gamma-1}$  for the RK scheme. Even though the sub-packetization level required for the RK scheme is less compared to the NT scheme, it still grows sub-exponentially with respect to the number of users while the sub-packetization level grows only linearly with the number of users in our scheme. Hence, compared to the RK scheme, our scheme is better in terms of both coding gain and sub-packetization level.

**Lemma 2.** *For the cases considered in Theorem 1, the coding gain achieved in our scheme,  $\frac{2K}{K - kL + k}$ , is greater than or equal to the coding gain,  $\frac{K}{K - kL}$ , achieved using the RK scheme.*

*Proof.* The coding gain achieved in our scheme is less than that achieved using the RK scheme, only if

$$\begin{aligned} \frac{K}{K - kL} &> \frac{2K}{K - kL + k} \\ \Rightarrow K - kL + k &> 2K - 2kL \\ \Rightarrow kL + k &> K \\ \Rightarrow L &> \frac{K}{k} - 1 \end{aligned}$$

If  $L \geq \frac{K}{k}$ , then all the users can access all the sub-files of each file and the transmission rate is zero. Therefore the coding gain achieved in our scheme is greater than or equal to that achieved using the RK scheme, for the cases considered in Theorem 1.  $\square$

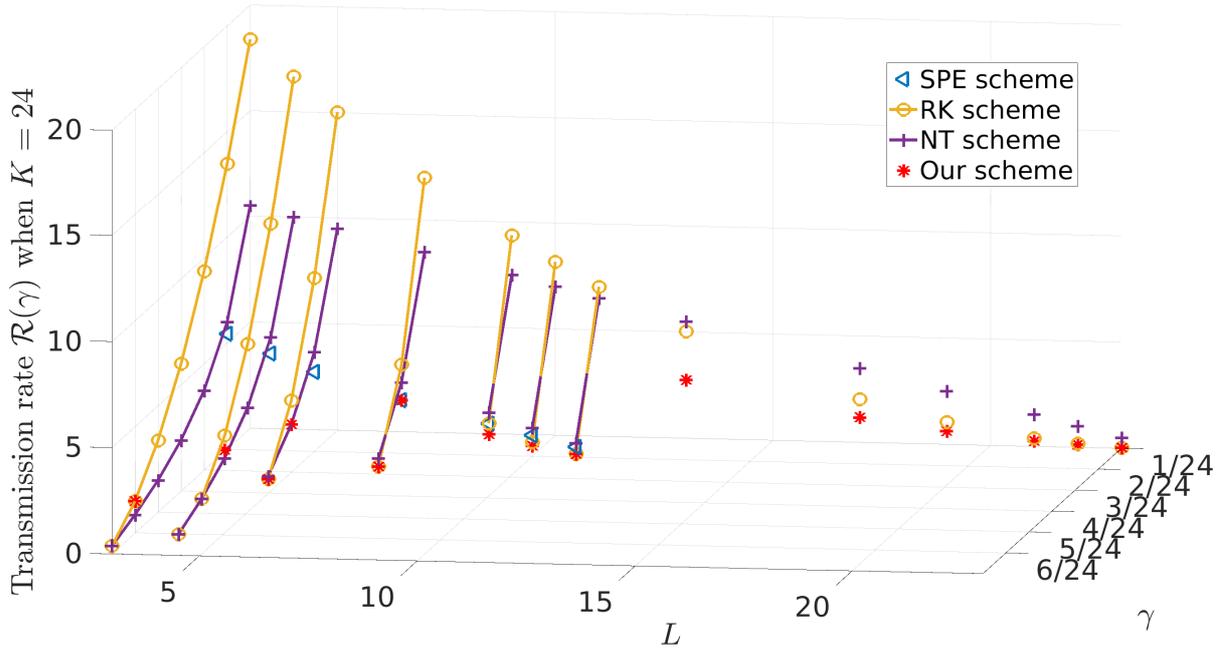


Fig. 2: Transmission rate vs  $L$  vs  $\gamma$  plot when  $K = 24$ .

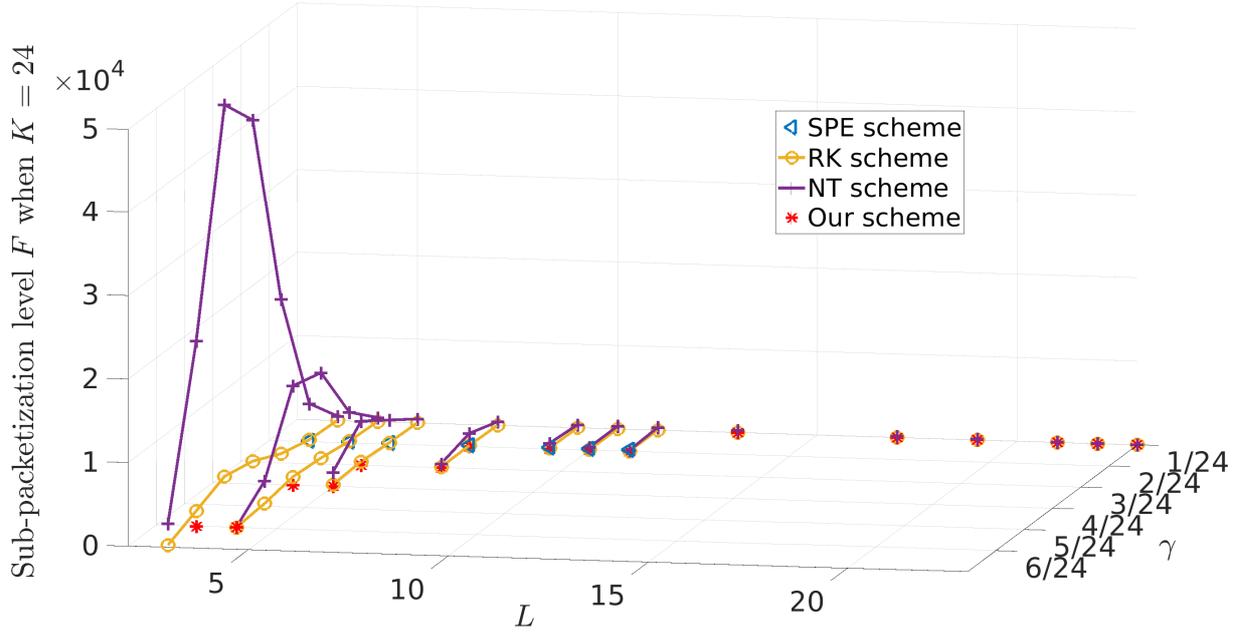


Fig. 3: Sub-packetization level vs  $L$  vs  $\gamma$  plot when  $K = 24$ .

### C. Comparison with the SPE Scheme

For the case when  $k = 2$ , the SPE scheme achieves a coding gain that always exceeds 3 and approaches 4 for some values of  $K$  and  $L$ . For our scheme, since  $\frac{2K}{K-kL+k}$  is an integer (we have assumed that  $K - kL + k$  divides  $K$ ), the coding

gain achieved when  $k = 2$  is always greater than or equal to  $k+2 = 4$  (as proved in Lemma 1). Also, the sub-packetization level required for SPE scheme is  $F_{SPE} = \frac{K(K-2L+2)}{4}$  which is  $\frac{(K-2L+2)}{4}$  times more than the sub-packetization level required for our scheme. Hence, compared to the SPE scheme

	Sub-packetization Level	Coding Gain	Transmission Rate
NT scheme [22]	$K \binom{K-kL+k}{k}$	$k+1$	$\frac{K-kL}{k+1}$
RK scheme [19]	$\binom{K-kL+k-1}{k-1} \frac{K}{k}$	$\frac{K}{K-kL}$	$\frac{(K-kL)^2}{K}$
SPE scheme (for $k=2$ ) [20]	$\binom{K(K-2L+2)}{4}$	3 to 4	$\mathcal{R}_{SPE}$ as in (2)
NK scheme [23]	$\binom{K-kL+k-1}{k-1} \frac{K}{k}$	$\frac{\mathcal{R}_{NK}}{K-kL}$	$\mathcal{R}_{NK}$ as in (4)
Our scheme	$K$	$\frac{2K}{K-kL+k}$	$\frac{(K-kL)(K-kL+k)}{2K}$

TABLE I: Comparison of our scheme with the state-of-the-art when  $\gamma \in \{\frac{k}{K} : k|K, (K-kL+k)|K, k \in [1, K]\}$ .

also, our scheme is better in terms of both coding gain and sub-packetization level for the cases considered in Theorem 1.

#### D. Comparison with the NK Scheme

For NK scheme, the rate expression  $\mathcal{R}_{NK}$  is given by (4). Due to the complexity in the expression, we are not able to compare this with our rate analytically. From Example 2, we observe that the transmission rate when  $K=12, k=2, L=4$  in the NK scheme is slightly less compared to our scheme. However, we gain in terms of the sub-packetization level required at the expense of a slight increase in the rate. The sub-packetization level required for our scheme is  $F_{new} = K$  while it is  $F_{NK} = \frac{1}{\gamma} \binom{K-kL+k-1}{k-1}$  for the NK scheme.

### III. PLACEMENT AND DELIVERY SCHEME

In this section, initially we present our placement scheme. After that we define a new class of PDA which is used for providing our delivery algorithm. Finally we present our delivery scheme to prove Theorem 1.

#### A. Placement Scheme

In the placement phase, we split each file  $W_n, n \in [0, N)$ , into  $K$  disjoint sub-files,  $W_n = \{W_{n,\alpha} : \alpha \in [0, K)\}$ . Each cache  $C_\alpha, \alpha \in [0, K)$ , is filled as follows:

$$M_\alpha = \{W_{n,(k\alpha+j) \bmod K} : j \in [0, k), n \in [0, N)\}.$$

Each cache stores  $k$  sub-files from all the files, where each sub-file is of size  $\frac{1}{K}$ . Hence,  $M = \frac{kN}{K} = N\gamma$ , thus meeting our memory constraint.

The placement is done in such a way that we first create a list of size  $1 \times kK$  by repeating the sequence  $\{0, 1, \dots, K-1\}$ ,  $k$  times, i.e.,  $\{0, 1, \dots, K-1, 0, 1, \dots, K-1, 0, 1, 2, \dots, K-1, \dots\}$ . We fill the caches by taking  $k$  items sequentially from the list. Hence, the first cache is filled with the first  $k$  items, the second cache with the next  $k$  items and so on.

Each user can access  $L$  neighboring caches and each cache stores  $k$  consecutive sub-files of each file. If  $L \geq \lceil \frac{K}{k} \rceil$ , then the user has access to all the sub-files of each of the files. For the case under consideration, each user has access to  $kL$  consecutive sub-files of each file since the content in any consecutive  $L$  caches are disjoint from one another if  $L < \lceil \frac{K}{k} \rceil$ . That is, for each user  $U_\alpha, \alpha \in [0, K)$ , the accessible cache content is  $\{W_{n,(k\alpha+i) \bmod K} : i \in [0, kL), n \in [0, N)\}$ .

Each user's demand of one file among the  $N$  files from the central server is revealed after the placement phase, i.e., the

demand vector  $\mathbf{d}$  is revealed. Our delivery scheme is based on the concept of PDA. So, before presenting the delivery scheme we provide a review on PDAs and after that we define a new class of PDAs which is required for our delivery scheme.

#### B. Review on Placement Delivery Arrays

**Definition 1.** ([10]) For positive integers  $K, F, Z$  and  $S$ , an  $F \times K$  array  $\mathbf{P} = (p_{i,j}), i \in [0, F), j \in [0, K)$ , composed of a specific symbol “ $\star$ ” called star and  $S$  positive integers  $0, 1, \dots, S-1$ , is called a  $(K, F, Z, S)$  placement delivery array (PDA) if it satisfies the following three conditions.

- C1: the symbol  $\star$  occurs exactly  $Z$  times in each column  
C2: each integer occurs at least once in the array  
C3: for any two distinct entries  $p_{i_1, j_1}$  and  $p_{i_2, j_2}$ , such that  $p_{i_1, j_1} = p_{i_2, j_2} = s$  is an integer only if

- a)  $i_1 \neq i_2, j_1 \neq j_2$ , i.e., they lie in distinct rows and distinct columns and  
b)  $p_{i_1, j_2} = p_{i_2, j_1} = \star$ , i.e., the corresponding  $2 \times 2$  sub-array formed by the rows  $i_1, i_2$  and the columns  $j_1, j_2$  must be of the following form  $\begin{pmatrix} s & \star \\ \star & s \end{pmatrix}$  or  $\begin{pmatrix} \star & s \\ s & \star \end{pmatrix}$

**Definition 2.** ([10]) An array  $\mathbf{P}$  is said to be a  $g$ -regular  $(K, F, Z, S)$  PDA, if it satisfies the conditions C1, C3, and the following condition

- C2': Each integer appears  $g$  times in  $\mathbf{P}$  where  $g$  is a constant.

**Theorem 2.** ([10]) For a given  $(K, F, Z, S)$  PDA,  $\mathbf{P} = (p_{i,j}), i \in [0, F), j \in [0, K)$ , we can obtain a  $(K, M, N)$  coded caching scheme having sub-packetization level  $F$  with  $\frac{M}{N} = \frac{Z}{F}$  using Algorithm 1. For any demand vector  $\mathbf{d}$ , the demands of all the users are met at the transmission rate of  $\mathcal{R} = \frac{S}{F}$ .

In a  $(K, F, Z, S)$  PDA  $\mathbf{P}$ , rows correspond to the packets of each file and columns correspond to the users. In any column  $j \in [0, K)$ , if  $p_{i,j} = \star$ , then user  $U_j$  has access to the  $i^{\text{th}}$  packet of all the files. If  $p_{i,j} = s$  is an integer, then it implies the  $i^{\text{th}}$  packet of none of the files is accessible to the user  $U_j$ . The condition C1 implies some  $Z$  packets of all the files are accessible to each user. The condition C2 implies that the number of symbols transmitted by the server is exactly  $S$  since XOR of the requested packets indicated by  $s$  is broadcast by the server during the slot for  $s$ . Hence the transmission rate is  $\frac{S}{F}$ . The condition C3 makes sure that each user can get

the demanded packet, since all the other packets in the coded symbol are available at its cache. The three conditions in the Definition 1 of the PDA guarantees all the users retrieve the requested files.

---

**Algorithm 1** Coded caching scheme based on PDA in [10]

---

```

1: procedure PLACEMENT( $\mathbf{P}, \mathcal{W}$ )
2:   Split each file  $W_n \in \mathcal{W}$  into  $F$  packets, i.e.,  $W_n = \{W_{n,i} : i \in [0, F]\}$ 
3:   for  $j \in [0, K]$  do
4:      $\mathcal{Z}_j \leftarrow \{W_{n,i} : p_{i,j} = \star, \forall n \in [0, N]\}$ 
5:   end for
6: end procedure
7: procedure DELIVERY( $\mathbf{P}, \mathcal{W}, \mathbf{d}$ )
8:   for  $s = 0, 1, \dots, S - 1$  do
9:     Server sends  $\mathcal{Y}_s = \bigoplus_{p_{i,j}=s, i \in [0, F], j \in [0, K]} W_{d_j, i}$ 
10:  end for
11: end procedure

```

---

### C. A New Class of PDAs

In this sub-section we introduce a new class of PDA, termed as  $t$ -cyclic  $g$ -regular  $(K, F, Z, S)$  PDA which is defined in Definition 3.

**Definition 3.** ( $t$ -cyclic  $g$ -regular  $(K, F, Z, S)$  PDA) In a  $g$ -regular  $(K, F, Z, S)$  PDA  $\mathbf{P}$ , if all the  $Z$  stars in each column occur in consecutive rows and if the position of stars in each column in  $\mathbf{P}$  is obtained by cyclically shifting the previous column downwards by  $t$  units, then it is called as  $t$ -cyclic  $g$ -regular  $(K, F, Z, S)$  PDA.

**Example 1.** Consider the following  $12 \times 6$  array  $\mathbf{P}_{12 \times 6}$ .

$$\mathbf{P}_{12 \times 6} = \begin{pmatrix} \star & 0 & 1 & \star & \star & \star \\ \star & 3 & 4 & \star & \star & \star \\ \star & \star & 2 & 0 & \star & \star \\ \star & \star & 5 & 3 & \star & \star \\ \star & \star & \star & 1 & 2 & \star \\ \star & \star & \star & 4 & 5 & \star \\ \star & \star & \star & \star & 0 & 1 \\ \star & \star & \star & \star & 3 & 4 \\ 0 & \star & \star & \star & \star & 2 \\ 3 & \star & \star & \star & \star & 5 \\ 1 & 2 & \star & \star & \star & \star \\ 4 & 5 & \star & \star & \star & \star \end{pmatrix} \quad (6)$$

There are  $Z = 8$  stars in each column and there are 6 integers in the array  $\mathbf{P}_{12 \times 6}$  where each integer  $s \in [0, 5]$  appears 4 times. For each integer  $s \in [0, 5]$ , it can be verified that the condition C3 in Definition 1 is also satisfied by the array  $\mathbf{P}_{12 \times 6}$ . Hence the array  $\mathbf{P}_{12 \times 6}$  represents a 4-regular  $(6, 12, 8, 6)$  PDA. It can be observed that all the 8 stars in each column occur in consecutive rows and the position of stars in each column can be obtained by cyclically shifting the previous column by 2 units down. Therefore, it is a 2-cyclic 4-regular  $(6, 12, 8, 6)$  PDA.

Our delivery algorithm is based on using  $t$ -cyclic  $g$ -regular PDAs. Before providing the general delivery algorithm, we illustrate the idea of using  $t$ -cyclic  $g$ -regular PDAs for solving multi-access coded caching problems with the help of an example.

**Example 2.** Consider the case  $N = K = 12, k = 2, L = 4$ . The server has access to 12 files:  $\mathcal{W} = \{W_n : n \in [0, 12)\}$ , and each file  $W_n, n \in [0, 12)$ , is divided into 12 sub-files:  $W_n = \{W_{n,j} : j \in [0, 12)\}$ . Each cache  $C_\alpha, \alpha \in [0, 12)$ , is filled as  $M_\alpha = \{W_{n, (2\alpha+j) \bmod 12} : j \in \{0, 1\}, n \in [0, 12)\}$ .

Each user  $U_\alpha, \alpha \in [0, 12)$ , has access to all the caches in the set  $\{C_\alpha, C_{(\alpha+1) \bmod 12}, C_{(\alpha+2) \bmod 12}, C_{(\alpha+3) \bmod 12}\}$ . Hence, for each user  $U_\alpha$ , the accessible cache content is  $\mathcal{Z}_\alpha = \{W_{n, (2\alpha+i) \bmod 12} : i \in [0, 8), n \in [0, 12)\}$ .

Consider the 2-cyclic 4-regular  $(6, 12, 8, 6)$  PDA,  $\mathbf{P}_{12 \times 6}$ , obtained in Example 1. For the array  $\mathbf{P}_{12 \times 6}$  with alphabets  $[0, 6) \cup \star$ , define an array  $\mathbf{P}_{12 \times 6} + 6 = (p_{i,j} + 6)$ , where  $\star + 6 = \star$ . The number of occurrences of each integer in  $\mathbf{P}_{12 \times 6} + 6$  is exactly equal to that of  $\mathbf{P}_{12 \times 6}$ . Also, the number of stars in each column in  $\mathbf{P}_{12 \times 6} + 6$  is exactly same as in  $\mathbf{P}_{12 \times 6}$ . Now, we construct a new array

$$\mathbf{P}_{12 \times 12} = (\mathbf{P}_{12 \times 6} \quad \mathbf{P}_{12 \times 6} + 6) = \begin{pmatrix} \star & 0 & 1 & \star & \star & \star & \star & 6 & 7 & \star & \star & \star \\ \star & 3 & 4 & \star & \star & \star & \star & 9 & 10 & \star & \star & \star \\ \star & \star & 2 & 0 & \star & \star & \star & \star & 8 & 6 & \star & \star \\ \star & \star & 5 & 3 & \star & \star & \star & \star & 11 & 9 & \star & \star \\ \star & \star & \star & 1 & 2 & \star & \star & \star & \star & 7 & 8 & \star \\ \star & \star & \star & 4 & 5 & \star & \star & \star & \star & 10 & 11 & \star \\ \star & \star & \star & \star & 0 & 1 & \star & \star & \star & \star & 6 & 7 \\ \star & \star & \star & \star & 3 & 4 & \star & \star & \star & \star & 9 & 10 \\ 0 & \star & \star & \star & \star & 2 & 6 & \star & \star & \star & \star & 8 \\ 3 & \star & \star & \star & \star & 5 & 9 & \star & \star & \star & \star & 11 \\ 1 & 2 & \star & \star & \star & \star & 7 & 8 & \star & \star & \star & \star \\ 4 & 5 & \star & \star & \star & \star & 10 & 11 & \star & \star & \star & \star \end{pmatrix} \quad (7)$$

There are 12 rows,  $0, 1, 2, \dots, 11$ , and 12 columns,  $0, 1, 2, \dots, 11$ , in the PDA  $\mathbf{P}_{12 \times 12}$ . The rows correspond to the sub-files of each file and columns correspond to the users. That is, the  $i^{\text{th}}$  row,  $i \in [0, 12)$ , represents the sub-files  $W_{n,i}, \forall n \in [0, 12)$ , while the  $j^{\text{th}}$  column,  $j \in [0, 12)$ , represents the user  $U_j$ . There are a total of 12 integers in  $\mathbf{P}_{12 \times 12}$ , and each integer occurs 4 times in  $\mathbf{P}_{12 \times 12}$ . All the conditions C1, C2' and C3 are satisfied by the array  $\mathbf{P}_{12 \times 12}$ . Additionally, all the 8 stars in each column occur in consecutive rows and the position of stars in each column can be obtained by cyclically shifting the previous column by 2 units down. Therefore, the array  $\mathbf{P}_{12 \times 12}$  represents a 2-cyclic 4-regular  $(12, 12, 8, 12)$  PDA.

Let the demand vector be  $\mathbf{d} = (d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9, d_{10}, d_{11})$ . We obtain a delivery scheme for the multi-access coded caching scheme with  $K = N = 12, k = 2, L = 4$  and sub-packetization level  $F = 12$ , using the caching scheme given by Algorithm

1 based on the PDA  $\mathbf{P}_{12 \times 12}$ . The following are the coded symbols obtained using Algorithm 1:

$$\begin{aligned}
\mathcal{Y}_0 &= W_{d_0,8} \oplus W_{d_1,0} \oplus W_{d_3,2} \oplus W_{d_4,6} \\
\mathcal{Y}_1 &= W_{d_0,10} \oplus W_{d_2,0} \oplus W_{d_3,4} \oplus W_{d_5,6} \\
\mathcal{Y}_2 &= W_{d_1,10} \oplus W_{d_2,2} \oplus W_{d_4,4} \oplus W_{d_5,8} \\
\mathcal{Y}_3 &= W_{d_0,9} \oplus W_{d_1,1} \oplus W_{d_3,3} \oplus W_{d_4,7} \\
\mathcal{Y}_4 &= W_{d_0,11} \oplus W_{d_2,1} \oplus W_{d_3,5} \oplus W_{d_5,7} \\
\mathcal{Y}_5 &= W_{d_1,11} \oplus W_{d_2,3} \oplus W_{d_4,5} \oplus W_{d_5,9} \\
\mathcal{Y}_6 &= W_{d_6,8} \oplus W_{d_7,0} \oplus W_{d_9,2} \oplus W_{d_{10},6} \\
\mathcal{Y}_7 &= W_{d_6,10} \oplus W_{d_8,0} \oplus W_{d_9,4} \oplus W_{d_{11},6} \\
\mathcal{Y}_8 &= W_{d_7,10} \oplus W_{d_8,2} \oplus W_{d_{10},4} \oplus W_{d_{11},8} \\
\mathcal{Y}_9 &= W_{d_6,9} \oplus W_{d_7,1} \oplus W_{d_9,3} \oplus W_{d_{10},7} \\
\mathcal{Y}_{10} &= W_{d_6,11} \oplus W_{d_8,1} \oplus W_{d_9,5} \oplus W_{d_{11},7} \\
\mathcal{Y}_{11} &= W_{d_7,11} \oplus W_{d_8,3} \oplus W_{d_{10},5} \oplus W_{d_{11},9}.
\end{aligned}$$

Hence, the PDA  $\mathbf{P}_{12 \times 12}$  characterizes the delivery scheme for this example. For the caching scheme generated by a 2-cyclic 4-regular  $(12, 12, 8, 12)$  PDA using Algorithm 1, the coding gain equals to 4, since each transmission benefits 4 users. The coding gain achieved using the NT and RK schemes for this example is 3 while it is 4 for SPE scheme. The sub-packetization level required for NT, RK and SPE schemes are 180, 30 and 18 respectively. The transmission rate achieved using our scheme is 1 while the transmission rate achieved using the NK scheme is 0.755. However the sub-packetization level required for the NK scheme is 30 while for our scheme the sub-packetization level required is 12 which is less than half of that value.

#### D. Delivery Scheme

In this section we provide the delivery scheme to prove Theorem 1. The parts of the file  $W_{d_\alpha}$  available with the user  $U_\alpha, \alpha \in [0, K)$ , are  $kL$  consecutive sub-files, i.e.,  $\{W_{d_\alpha, (k\alpha+i) \bmod K} : i \in [0, kL)\}$ . Hence, the user  $U_\alpha$  should be able to decode all the remaining  $K - kL$  sub-files, i.e.,  $\{W_{d_\alpha, (k\alpha+kL+i) \bmod K} : i \in [0, K - kL)\}$ . To retrieve the remaining sub-files, the transmitted symbols are obtained using the PDA constructed in Algorithm 2 for the case discussed in Theorem 1.

**Theorem 3.** The  $K \times K$  matrix  $\mathbf{P}$  obtained using Algorithm 2 is a  $k$ -cyclic  $\frac{2K}{K-kL+k}$ -regular  $(K, K, kL, \frac{(K-kL)(K-kL+k)}{2})$  PDA.

The proof of Theorem 3 is provided in Section IV.

**Theorem 4.** For a given  $k$ -cyclic  $\frac{2K}{K-kL+k}$ -regular  $(K, K, kL, \frac{(K-kL)(K-kL+k)}{2})$  PDA,  $\mathbf{P} = (p_{i,j}), i \in [0, K), j \in [0, K)$  constructed using Algorithm 2, we can obtain a delivery algorithm using Algorithm 1 for the considered case of multi-access coded problem in Theorem 1, with sub-packetization level  $F = K$ . For any demand vector

$\mathbf{d}$ , the demands of all the users are met at the transmission rate of  $\mathcal{R}_{new} = \frac{(K-kL)(K-kL+k)}{2K}$ .

*Proof.* The proof of this theorem is straightforward from Theorem 2. In a  $t$ -cyclic  $g$ -regular  $(K, F, Z, S)$  PDA  $\mathbf{P}$ , rows correspond to the sub-files of each file and columns correspond to the users. In any column  $j \in [0, K)$ , if  $p_{i,j} = \star$ , then user  $U_j$  has access to the  $i^{\text{th}}$  sub-file of all the files. If  $p_{i,j} = s$  is an integer, then it implies the  $i^{\text{th}}$  sub-file of none of the files is accessible to the user  $U_j$ . The condition C1 implies some  $Z$  sub-files of all the files are accessible to each user. The condition C2 implies that the number of symbols transmitted by the server is exactly  $S$  since XOR of the requested packets indicated by  $s$  is broadcast by the server during the slot for  $s$ . Hence the transmission rate is  $\frac{S}{F}$ . The condition C3 makes sure that each user can get the demanded sub-file, since all the other sub-files in the coded symbol are available at its cache. The three conditions in the Definition 1 of the PDA guarantees all the users retrieve the requested files.  $\square$

**Example 3.** Consider Example 2 with  $K = N = 12, k = 2, L = 4$ . We illustrate each procedure involved in obtaining the matrix  $\mathbf{P}_{12 \times 12}$  using Algorithm 2 for Example 2. The  $\mathbf{A}$  matrix obtained by **procedure 1** in Algorithm 2 corresponding to this example is

$$\mathbf{A} = \begin{pmatrix} \star & 0 & 1 \\ \star & \star & 2 \\ \star & \star & \star \end{pmatrix}.$$

The  $\mathbf{P}_1$  matrix obtained by **procedure 2** in Algorithm 2 corresponding to this example is

$$\mathbf{P}_1 = \begin{pmatrix} \mathbf{A} & \mathbf{A}^T \\ \mathbf{A}^T & \mathbf{A} \end{pmatrix} = \left( \begin{array}{ccc|ccc} \star & 0 & 1 & \star & \star & \star \\ \star & \star & 2 & 0 & \star & \star \\ \star & \star & \star & 1 & 2 & \star \\ \hline \star & \star & \star & \star & 0 & 1 \\ 0 & \star & \star & \star & \star & 2 \\ 1 & 2 & \star & \star & \star & \star \end{array} \right).$$

Using **procedure 3** in Algorithm 2, we obtain the matrix  $\tilde{\mathbf{P}}_1$  as in (6).

Now, using **procedure 4**, the matrix  $\tilde{\mathbf{P}}_2$  obtained is as follows:

$$\tilde{\mathbf{P}}_2 = \tilde{\mathbf{P}}_1 + 6 = \left( \begin{array}{ccc|ccc} \star & 6 & 7 & \star & \star & \star \\ \star & 9 & 10 & \star & \star & \star \\ \star & \star & 8 & 6 & \star & \star \\ \star & \star & 11 & 9 & \star & \star \\ \star & \star & \star & 7 & 8 & \star \\ \star & \star & \star & 10 & 11 & \star \\ \hline \star & \star & \star & \star & 6 & 7 \\ \star & \star & \star & \star & 9 & 10 \\ 6 & \star & \star & \star & \star & 8 \\ 9 & \star & \star & \star & \star & 11 \\ 7 & 8 & \star & \star & \star & \star \\ 10 & 11 & \star & \star & \star & \star \end{array} \right)$$

Finally, the matrix  $\mathbf{P}$  obtained by concatenating  $\tilde{\mathbf{P}}_1$  and  $\tilde{\mathbf{P}}_2$

---

**Algorithm 2**  $k$ -cyclic  $\frac{2K}{K-kL+k}$ -regular  
 $(K, K, kL, \frac{(K-kL)(K-kL+k)}{2})$  PDA  $\mathbf{P}$  Construction, where  
 $\gamma \in \{\frac{k}{K} : k|K, (K-kL+k)|K, k \in [1, K]\}$

---

1: **procedure 1:** CONSTRUCT A SQUARE MATRIX  $\mathbf{A} = (a_{i,j}), i, j \in [0, \frac{K-kL}{k}]$ .  
2:   **for**  $i \in [0, \frac{K-kL}{k}]$  **do**  
3:     **for**  $j \in [0, \frac{K-kL}{k}]$  **do**  
4:       **if**  $j \leq i$  **then**  
5:           $a_{i,j} = \star$   
6:       **else if**  $i = 0$  and  $j > i$  **then**  
7:           $a_{i,j} = j - 1$   
8:       **else**  
9:           $a_{i,j} = a_{i-1,j} + (\frac{K-kL}{k} - i)$   
10:       **end if**  
11:     **end for**  
12:   **end for**  
13: **end procedure 1**  
14: **procedure 2:** OBTAIN A SQUARE MATRIX  $\mathbf{P}_1 = (p_{i,j}), i, j \in [0, \frac{K}{k}]$ .

$$\mathbf{P}_1 = \begin{pmatrix} \mathbf{A} & \mathbf{A}^T & \mathbf{X} & \mathbf{X} & \dots & \mathbf{X} & \mathbf{X} \\ \mathbf{X} & \mathbf{A} & \mathbf{A}^T & \mathbf{X} & \dots & \mathbf{X} & \mathbf{X} \\ \mathbf{X} & \mathbf{X} & \mathbf{A} & \mathbf{A}^T & \dots & \mathbf{X} & \mathbf{X} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \dots & \vdots \\ \mathbf{X} & \mathbf{X} & \mathbf{X} & \mathbf{X} & \dots & \mathbf{A} & \mathbf{A}^T \\ \mathbf{A}^T & \mathbf{X} & \mathbf{X} & \mathbf{X} & \dots & \mathbf{X} & \mathbf{A} \end{pmatrix}$$

where  $\mathbf{X}$  represents a  $\frac{K-kL+k}{k} \times \frac{K-kL+k}{k}$  matrix with all the entries being  $\star$ . The matrix  $\mathbf{P}_1$  is a block matrix having  $\frac{K}{k}$  row and column blocks with blocks of size  $\frac{K-kL+k}{k} \times \frac{K-kL+k}{k}$  as entries.

15: **end procedure 2**  
16: **procedure 3:** FROM THE MATRIX  $\mathbf{P}_1$ , GENERATE A  $K \times \frac{K}{k}$  MATRIX  $\tilde{\mathbf{P}}_1 = (\tilde{p}_{i,j}), i \in [0, K), j \in [0, \frac{K}{k})$ .  
17:   Let  $S_1 = (\frac{(K-kL)(K-kL+k)}{2k^2})$ .  
18:   **for**  $i \in [0, K)$  **do**  
19:     **for**  $j \in [0, \frac{K}{k})$  **do**  
20:       **if**  $k|i$  **then**  
21:           $\tilde{p}_{i,j} = p(\frac{i}{k}, j)$ .  
22:       **else**  
23:           $\tilde{p}_{i,j} = \tilde{p}_{i-1,j} + S_1$ , where  $S_1 + \star = \star$ .  
24:       **end if**  
25:     **end for**  
26:   **end for**  
27: **end procedure 3**  
28: **procedure 4:** NOW, OBTAIN A  $K \times K$  MATRIX  $\mathbf{P}$

$$\mathbf{P} = (\tilde{\mathbf{P}}_1 \quad \tilde{\mathbf{P}}_2 \quad \tilde{\mathbf{P}}_3 \quad \dots \quad \tilde{\mathbf{P}}_k)$$

where each matrix  $\tilde{\mathbf{P}}_t = \tilde{\mathbf{P}}_1 + (t-1)\tilde{S}_1, t \in [2, k], \tilde{S}_1 = \frac{(K-kL)(K-kL+k)}{2k}$ , is a  $K \times \frac{K}{k}$  matrix, with  $\star + (t-1)\tilde{S}_1 = \star$ .

29: **end procedure 4**

---

is as in (7).

**Example 4.** Consider another example with  $N = K = 36, k = 3, L = 9$ . The server has access to 36 files:  $\mathcal{W} = \{W_n : n \in [0, 36)\}$ , and each file  $W_n, n \in [0, 36)$ , is divided into 36 sub-files:  $W_n = \{W_{n,j} : j \in [0, 36)\}$ . Each cache  $C_\alpha, \alpha \in [0, 36)$ , is filled as  $M_\alpha = \{W_{n, (3\alpha+j) \bmod 36} : j \in \{0, 1, 2\}, n \in [0, 36)\}$ .

Each user  $U_\alpha, \alpha \in [0, 36)$ , has access to all the caches in the set  $\{C_{(\alpha+i) \bmod 36} : i \in [0, 9)\}$ . Hence, for each user  $U_\alpha$ , the accessible cache content is  $\mathcal{Z}_\alpha = \{W_{n, (3\alpha+i) \bmod 36} : i \in [0, 27), n \in [0, 36)\}$ .

The  $\mathbf{A}$  matrix obtained by **procedure 1** in Algorithm 2 corresponding to this example is

$$\mathbf{A} = \begin{pmatrix} \star & 0 & 1 & 2 \\ \star & \star & 3 & 4 \\ \star & \star & \star & 5 \\ \star & \star & \star & \star \end{pmatrix}.$$

The  $\mathbf{P}_1$  matrix obtained by **procedure 2** in Algorithm 2 corresponding to this example is

$$\mathbf{P}_1 = \begin{pmatrix} \mathbf{A} & \mathbf{A}^T & \mathbf{X} \\ \mathbf{X} & \mathbf{A} & \mathbf{A}^T \\ \mathbf{A}^T & \mathbf{X} & \mathbf{A} \end{pmatrix} \quad (8)$$

$$= \begin{pmatrix} \star & 0 & 1 & 2 & \star \\ \star & \star & 3 & 4 & 0 & \star \\ \star & \star & \star & 5 & 1 & 3 & \star \\ \star & \star & \star & \star & 2 & 4 & 5 & \star & \star & \star & \star & \star & \star \\ \hline \star & \star & \star & \star & \star & 0 & 1 & 2 & \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star & \star & 3 & 4 & 0 & \star & \star & \star & \star \\ \star & 5 & 1 & 3 & \star & \star & \star \\ \star & 2 & 4 & 5 & \star & \star \\ \hline \star & 0 & 1 & 2 & \star \\ 0 & \star & 3 & 4 & \star \\ 1 & 3 & \star & 5 \\ 2 & 4 & 5 & \star \end{pmatrix}. \quad (9)$$

Using **procedure 3** in Algorithm 2, we obtain the matrix  $\tilde{\mathbf{P}}_1$

as given below.  $\tilde{\mathbf{P}}_1 =$

*	*	0	1	2	*	*	*	*	*	*	*	*	
*	*	6	7	8	*	*	*	*	*	*	*	*	
*	*	12	13	14	*	*	*	*	*	*	*	*	
*	*	*	3	4	0	*	*	*	*	*	*	*	
*	*	*	9	10	6	*	*	*	*	*	*	*	
*	*	*	15	16	12	*	*	*	*	*	*	*	
*	*	*	*	5	1	3	*	*	*	*	*	*	
*	*	*	*	11	7	9	*	*	*	*	*	*	
*	*	*	*	17	13	15	*	*	*	*	*	*	
*	*	*	*	*	2	4	5	*	*	*	*	*	
*	*	*	*	*	8	10	11	*	*	*	*	*	
*	*	*	*	*	14	16	17	*	*	*	*	*	
*	*	*	*	*	*	0	1	2	*	*	*	*	
*	*	*	*	*	*	6	7	8	*	*	*	*	
*	*	*	*	*	*	12	13	14	*	*	*	*	
*	*	*	*	*	*	*	3	4	0	*	*	*	
*	*	*	*	*	*	*	9	10	6	*	*	*	
*	*	*	*	*	*	*	15	16	12	*	*	*	
*	*	*	*	*	*	*	*	5	1	3	*	*	
*	*	*	*	*	*	*	*	11	7	9	*	*	
*	*	*	*	*	*	*	*	17	13	15	*	*	
*	*	*	*	*	*	*	*	*	2	4	5	*	
*	*	*	*	*	*	*	*	*	8	10	11	*	
*	*	*	*	*	*	*	*	*	14	16	17	*	
*	*	*	*	*	*	*	0	1	*	*	*	2	
*	*	*	*	*	*	*	6	7	*	*	*	8	
*	*	*	*	*	*	*	*	*	*	*	12	13	14
0	*	*	*	*	*	*	*	*	*	*	3	4	*
6	*	*	*	*	*	*	*	*	*	*	9	10	*
12	*	*	*	*	*	*	*	*	*	*	15	16	*
1	3	*	*	*	*	*	*	*	*	*	*	5	*
7	9	*	*	*	*	*	*	*	*	*	*	11	*
13	15	*	*	*	*	*	*	*	*	*	*	17	*
2	4	5	*	*	*	*	*	*	*	*	*	*	*
8	10	11	*	*	*	*	*	*	*	*	*	*	*
14	16	17	*	*	*	*	*	*	*	*	*	*	*

(10)

follows:  $\tilde{\mathbf{P}}_2 = \tilde{\mathbf{P}}_1 + 18 =$

*	*	18	19	20	*	*	*	*	*	*	*	*		
*	*	24	25	26	*	*	*	*	*	*	*	*		
*	*	30	31	32	*	*	*	*	*	*	*	*		
*	*	*	21	22	18	*	*	*	*	*	*	*		
*	*	*	27	28	24	*	*	*	*	*	*	*		
*	*	*	33	34	30	*	*	*	*	*	*	*		
*	*	*	*	23	19	21	*	*	*	*	*	*		
*	*	*	*	29	25	27	*	*	*	*	*	*		
*	*	*	*	35	31	33	*	*	*	*	*	*		
*	*	*	*	*	20	22	23	*	*	*	*	*		
*	*	*	*	*	*	*	*	*	*	*	*	*		
*	*	*	*	*	*	*	*	*	*	*	*	*		
*	*	*	*	*	*	*	*	*	*	*	*	*		
*	*	*	*	*	*	*	18	19	*	*	*	*		
*	*	*	*	*	*	*	24	25	*	*	*	*		
*	*	*	*	*	*	*	30	31	*	*	*	*		
*	*	*	*	*	*	*	*	21	18	*	*	*		
*	*	*	*	*	*	*	*	27	24	*	*	*		
*	*	*	*	*	*	*	*	33	30	*	*	*		
*	*	*	*	*	*	*	*	*	*	*	21	*	*	
*	*	*	*	*	*	*	*	29	25	27	*	*	*	
*	*	*	*	*	*	*	*	35	31	33	*	*	*	
*	*	*	*	*	*	*	*	*	*	*	20	22	23	*
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
*	*	*	*	*										

36 =

$$\begin{pmatrix}
 \star & 36 & 37 & 38 & \star \\
 \star & 42 & 43 & 44 & \star \\
 \star & 48 & 49 & 50 & \star \\
 \star & \star & 39 & 40 & 36 & \star \\
 \star & \star & 45 & 46 & 42 & \star \\
 \star & \star & 51 & 52 & 48 & \star \\
 \star & \star & \star & 41 & 37 & 39 & \star \\
 \star & \star & \star & 47 & 43 & 45 & \star \\
 \star & \star & \star & 53 & 49 & 51 & \star \\
 \star & \star & \star & \star & 38 & 40 & 41 & \star & \star & \star & \star & \star & \star \\
 \star & \star & \star & \star & 44 & 46 & 47 & \star & \star & \star & \star & \star & \star \\
 \star & \star & \star & \star & 50 & 52 & 53 & \star & \star & \star & \star & \star & \star \\
 \hline
 \star & \star & \star & \star & \star & 36 & 37 & 38 & \star & \star & \star & \star & \star \\
 \star & \star & \star & \star & \star & 42 & 43 & 44 & \star & \star & \star & \star & \star \\
 \star & \star & \star & \star & \star & 48 & 49 & 50 & \star & \star & \star & \star & \star \\
 \star & \star & \star & \star & \star & \star & 39 & 40 & 36 & \star & \star & \star & \star \\
 \star & \star & \star & \star & \star & \star & 45 & 46 & 42 & \star & \star & \star & \star \\
 \star & \star & \star & \star & \star & \star & 51 & 52 & 48 & \star & \star & \star & \star \\
 \star & 41 & 37 & 39 & \star & \star & \star \\
 \star & 47 & 43 & 45 & \star & \star & \star \\
 \star & 53 & 49 & 51 & \star & \star & \star \\
 \star & 38 & 40 & 41 & \star & \star \\
 \star & 44 & 46 & 47 & \star & \star \\
 \star & 50 & 52 & 53 & \star & \star \\
 \hline
 \star & 36 & 37 & 38 & \star \\
 \star & 42 & 43 & 44 \\
 \star & 48 & 49 & 50 \\
 36 & \star & 39 & 40 \\
 42 & \star & 45 & 46 \\
 48 & \star & 51 & 52 \\
 37 & 39 & \star & 41 \\
 43 & 45 & \star & 47 \\
 49 & 51 & \star & 53 \\
 38 & 40 & 41 & \star \\
 44 & 46 & 47 & \star \\
 50 & 52 & 53 & \star & \star
 \end{pmatrix} \quad (12)$$

Finally, the matrix  $\mathbf{P} = (\tilde{\mathbf{P}}_1 \tilde{\mathbf{P}}_2 \tilde{\mathbf{P}}_3)$  is obtained by concatenating  $\tilde{\mathbf{P}}_1, \tilde{\mathbf{P}}_2$  and  $\tilde{\mathbf{P}}_3$ .

There are 36 rows and columns in the matrix  $\mathbf{P}$ , indexed by  $0, 1, 2, \dots, 35$ . The rows correspond to the sub-files of each file and columns correspond to the users. There are a total of 54 integers in  $\mathbf{P}$ , and each integer occurs 6 times in  $\mathbf{P}$ . All the conditions C1, C2' and C3 are satisfied by the matrix  $\mathbf{P}$ . Additionally, all the 27 stars in each column occur in consecutive rows and the position of stars in each column can be obtained by cyclically shifting the previous column by 3 units down. Therefore, the matrix  $\mathbf{P}$  represents a 3-cyclic 6-regular  $(36, 36, 27, 54)$  PDA.

Using the caching scheme given by Algorithm 1 based on the PDA  $\mathbf{P}$  constructed in this example, we obtain 54 coded symbols. The coding gain achieved using this scheme is 6, since each transmission benefits 6 users. The coding gain achieved using the NT and RK schemes for this example is 4. The sub-packetization level required for NT, and RK schemes are 7920 and 660 respectively.

#### IV. PROOF OF THEOREM 3

In this section, we prove that the matrix  $\mathbf{P}$  constructed using Algorithm 2 is a  $k$ -cyclic  $\frac{2K}{K-kL+k}$ -regular

$(K, K, kL, \frac{(K-kL)(K-kL+k)}{2})$  PDA.

Using **procedure 1** of Algorithm 2, we construct a square matrix  $\mathbf{A}$ . It can be observed from step 5 that all the diagonal entries as well as all the entries below the diagonal are stars. For  $i = 0$ , we consider the  $0^{\text{th}}$  row. The entry corresponding to the  $0^{\text{th}}$  column is a star. From step 6, the entries corresponding to  $1^{\text{st}}$  column to  $(\frac{K-kL}{k})^{\text{th}}$  column are integers from 0 to  $\frac{K-kL}{k} - 1$  respectively. Now, for  $i = 1$ , we consider the  $1^{\text{st}}$  row and the entry corresponding to the  $0^{\text{th}}$  and  $1^{\text{st}}$  columns are stars. The entries corresponding to  $2^{\text{nd}}$  column to  $(\frac{K-kL}{k})^{\text{th}}$  column are integers from  $\frac{K-kL}{k}$  to  $\frac{2(K-kL)}{k} - 2$  respectively. Similarly for any row  $i \in [0, \frac{K-kL}{k}]$ , the entries corresponding to the  $0^{\text{th}}$  column to the  $i^{\text{th}}$  column are stars. The positions corresponding to the rest of the columns ( $(i+1)^{\text{th}}$  column to  $(\frac{K-kL}{k})^{\text{th}}$  column) are filled with  $\frac{K-kL}{k} - i$  different integers which are also different from the integers used for filling the previous rows. There are  $S_1 = \frac{(K-kL)(K-kL+k)}{2k^2}$  integers present in the matrix  $\mathbf{A}$ , since there are  $S_1$  positions available above the diagonal and all those positions are filled with different integers. The number of stars in the  $i^{\text{th}}$  column,  $i \in [0, \frac{K-kL}{k}]$  of the matrix  $\mathbf{A}$ , is  $\frac{K-kL+k}{k} - i$ .

Using **procedure 2** of Algorithm 2, we obtain a block matrix  $\mathbf{P}_1$  with  $\frac{K}{K-kL+k}$  row as well as column blocks, indexed by  $[0, \frac{K}{K-kL+k})$ . Each block is of size  $\frac{K-kL+k}{k} \times \frac{K-kL+k}{k}$ . The  $0^{\text{th}}$  column block is given by

$$\begin{pmatrix}
 \mathbf{A} \\
 \mathbf{X} \\
 \vdots \\
 \mathbf{X} \\
 \mathbf{A}^T
 \end{pmatrix} \quad (13)$$

Each column block,  $i \in [1, \frac{K}{K-kL+k})$  is obtained by cyclically shifting the blocks in the  $0^{\text{th}}$  column block down by  $i$  units. Each block in  $\mathbf{P}_1$  is of size  $\frac{K-kL+k}{k} \times \frac{K-kL+k}{k}$  and there are  $\frac{K}{K-kL+k}$  rows and column blocks present in  $\mathbf{P}_1$ . Hence the matrix  $\mathbf{P}_1$  is of size  $\frac{K}{k} \times \frac{K}{k}$ .

Using **procedure 3** of Algorithm 2, a tall matrix  $\tilde{\mathbf{P}}_1$  is generated from the square matrix  $\mathbf{P}_1$ . The  $K \times \frac{K}{k}$  matrix  $\tilde{\mathbf{P}}_1$  is generated from the  $\frac{K}{k} \times \frac{K}{k}$  matrix  $\mathbf{P}_1$  by adding new  $k-1$  rows after each row in  $\mathbf{P}_1$ . For each  $i \in [0, \frac{K}{k})$ , the  $(ik)^{\text{th}}$  row of the matrix  $\tilde{\mathbf{P}}_1$  is same as the  $i^{\text{th}}$  row of the matrix  $\mathbf{P}_1$  (as in step 21). It is evident from step 23 that the positions of stars in the next  $k-1$  rows, i.e.,  $(ik+1)^{\text{th}}$  row to  $((i+1)k-1)^{\text{th}}$  row, are same as that of the  $(ik)^{\text{th}}$  row of the matrix  $\tilde{\mathbf{P}}_1$ . For the  $(ik+j)^{\text{th}}$  row,  $j \in [1, k)$  of the matrix  $\tilde{\mathbf{P}}_1$  the entries corresponding to columns which do not have stars are filled with integer obtained by adding  $S_1$  to the corresponding elements in the previous row of the matrix  $\tilde{\mathbf{P}}_1$ . This is to make sure that the integers used for filling the  $(ik+j)^{\text{th}}$  row is different from the integers used for filling the  $(ik)^{\text{th}}$  row to  $(ik+j-1)^{\text{th}}$  row.

Using the **procedure 4** of Algorithm 2, we define new matrices  $\tilde{\mathbf{P}}_t, t \in [2, k]$ , where  $\tilde{\mathbf{P}}_t = (\tilde{p}_{i,j} + (t-1)\tilde{S}_1)$ ,  $\tilde{S}_1$  denotes the number of integers present in the matrix  $\tilde{\mathbf{P}}_1$ , and

$\star + (t-1)\tilde{S}_1 = \star$ . Each integer entries in  $\tilde{\mathbf{P}}_t$  is obtained by adding  $(t-1)\tilde{S}_1$  to the corresponding entry in  $\tilde{\mathbf{P}}_1$ . The matrix  $\mathbf{P}$  is obtained by concatenating the  $k$  matrices  $\tilde{\mathbf{P}}_t, t \in [1, k]$ , each of size  $K \times \frac{K}{k}$ . Hence the matrix  $\mathbf{P}$  is of size  $K \times K$ .

**Lemma 3.** *The  $\frac{K}{k} \times \frac{K}{k}$  matrix  $\mathbf{P}_1$  obtained from procedure 2 of Algorithm 2 is a 1-cyclic  $\frac{2K}{K-kL+k}$ -regular  $(\frac{K}{k}, \frac{K}{k}, L, \frac{(K-kL)(K-kL+k)}{2k^2})$  PDA.*

*Proof.* The number of stars present in the  $i^{\text{th}}$  column,  $i \in [0, \frac{K-kL}{k}]$  of the matrix  $\mathbf{A}$ , is  $\frac{K-kL+k}{k} - i$  while the number of stars present in the  $i^{\text{th}}$  column of the matrix  $\mathbf{A}^T$  (the transpose of the matrix  $\mathbf{A}$ ) is  $i+1$ . Hence the number of stars present in the  $i^{\text{th}}$  column,  $i \in [0, \frac{K-kL}{k}]$  of the matrix  $\mathbf{P}_1$ , is  $Z_1 = (\frac{K-kL+k}{k} - i) + (\frac{K}{K-kL+k} - 2)(\frac{K-kL+k}{k}) + (i+1) = L$ . The number of stars in each column of the matrix  $\mathbf{P}_1$  is also  $L$ , since each column block,  $i \in [1, \frac{K}{K-kL+k})$  is obtained by cyclically shifting the blocks in the  $0^{\text{th}}$  column block down by  $i$  units, Hence the condition C1 in Definition 1 is satisfied by the matrix  $\mathbf{P}_1$ .

The number of integers present in the matrix  $\mathbf{A}$  is  $S_1 = \frac{(K-kL)(K-kL+k)}{2k^2}$ . Each integer in the set  $[0, S_1)$  occurs once in the matrix  $\mathbf{A}$  as well as in  $\mathbf{A}^T$ . The block  $\mathbf{A}$  as well as the block  $\mathbf{A}^T$  occur  $\frac{K}{K-kL+k}$  times in the matrix  $\mathbf{P}_1$  (once in each column block). The block  $\mathbf{X}$  contains only stars. Hence, the number of integers present in the matrix  $\mathbf{P}_1$  is  $S_1$  and each integer in the set  $[0, S_1)$  occurs  $\frac{2K}{K-kL+k}$  times in the matrix  $\mathbf{P}_1$ . So, the condition C2' of Definition 2 is satisfied by the matrix  $\mathbf{P}_1$  with  $g = \frac{2K}{K-kL+k}$ .

Now, we need to prove that the condition C3 of Definition 1 is satisfied by the matrix  $\mathbf{P}_1$ . Consider the sub-matrix  $\tilde{\mathbf{A}} = (\tilde{a}_{i,j}), i \in [0, \frac{K-kL}{k}], j \in [0, \frac{2(K-kL)}{k} + 1]$ , present in the matrix  $\mathbf{P}_1$ , where

$$\tilde{\mathbf{A}} = \begin{pmatrix} \mathbf{A} & \mathbf{A}^T \end{pmatrix} \quad (14)$$

All the diagonal entries as well as the entries below the diagonal are stars in the matrix  $\mathbf{A}$ . So, for the transpose matrix  $\mathbf{A}^T$ , all the diagonal entries and the entries above the diagonal are stars. Hence, for any integer  $s \in [0, S_1)$ , suppose that the integer  $s$  is present in the position corresponding to  $i^{\text{th}}$  row and  $j^{\text{th}}$  column of the matrix  $\mathbf{A}$ . Then, for  $i, j \in [0, \frac{K-kL}{k}], j > i$ ,

$$\begin{pmatrix} \tilde{a}_{i,j} & \tilde{a}_{i,i+\frac{K-kL+k}{k}} \\ \tilde{a}_{j,j} & \tilde{a}_{j,i+\frac{K-kL+k}{k}} \end{pmatrix} = \begin{pmatrix} s & \star \\ \star & s \end{pmatrix}. \quad (15)$$

Hence for any integer  $s$  present in the matrix  $\mathbf{A}$ , the sub-matrix given by (14) satisfies the condition C3 in Definition 1.

For the same reason stated for the sub-matrix (14), the sub-matrix  $\begin{pmatrix} \mathbf{A}^T \\ \mathbf{A} \end{pmatrix}$  also satisfies the condition C3 in Definition 1.

Suppose we take any two non-adjacent row blocks in  $\mathbf{P}_1$ , say  $i^{\text{th}}$  and  $j^{\text{th}}$  row blocks,  $i, j \in [0, \frac{K-kL}{k})$ , (the  $0^{\text{th}}$  and the  $(\frac{K}{K-kL+k} - 1)^{\text{th}}$  row blocks are considered as adjacent row blocks). It can be observed that the sub-matrix containing only the blocks  $\mathbf{A}$  and  $\mathbf{A}^T$  in the corresponding column blocks

is of the form

$$\begin{pmatrix} \mathbf{A} & \mathbf{A}^T & \mathbf{X} & \mathbf{X} \\ \mathbf{X} & \mathbf{X} & \mathbf{A} & \mathbf{A}^T \end{pmatrix} \text{ or } \begin{pmatrix} \mathbf{X} & \mathbf{X} & \mathbf{A} & \mathbf{A}^T \\ \mathbf{A} & \mathbf{A}^T & \mathbf{X} & \mathbf{X} \end{pmatrix} \quad (16)$$

Since the matrix  $\mathbf{X}$  contains only stars, it is evident that the sub-matrices given by (16) satisfies the condition C3 in Definition 1.

Suppose we take any two adjacent row blocks in  $\mathbf{P}_1$ , say  $i^{\text{th}}$  and  $(i+1)^{\text{th}}$  row blocks,  $i \in [0, \frac{K}{K-kL+k})$  (the  $0^{\text{th}}$  and the  $(\frac{K}{K-kL+k} - 1)^{\text{th}}$  rows are considered as adjacent row blocks and if  $i = \frac{K}{K-kL+k} - 1$ , then  $i+1$  is considered as 0). It can be observed that the sub-matrix containing only the blocks  $\mathbf{A}$  and  $\mathbf{A}^T$  in the corresponding columns blocks is of the form

$$\mathbf{A}' = (a'_{i,j}) = \begin{pmatrix} \mathbf{A} & \mathbf{A}^T & \mathbf{X} \\ \mathbf{X} & \mathbf{A} & \mathbf{A}^T \end{pmatrix} \quad (17)$$

For any integer  $s \in [0, S_1)$ , suppose that the integer  $s$  is present in the position corresponding to the  $i^{\text{th}}$  row and the  $j^{\text{th}}$  column of the matrix  $\mathbf{A}$ ,  $i, j \in [0, \frac{K-kL}{k}], j > i$ , then

$$\begin{pmatrix} a'_{i,j} & a'_{i,i+l} & a'_{i,j+l} & a'_{i,i+2l} \\ a'_{j,j} & a'_{j,i+l} & a'_{j,j+l} & a'_{j,i+2l} \\ a'_{i+l,j} & a'_{i+l,i+l} & a'_{i+l,j+l} & a'_{i+l,i+2l} \\ a'_{j+l,j} & a'_{j+l,i+l} & a'_{j+l,j+l} & a'_{j+l,i+2l} \end{pmatrix} = \begin{pmatrix} s & \star & \star & \star \\ \star & s & \star & \star \\ \star & \star & s & \star \\ \star & \star & \star & s \end{pmatrix}$$

where  $l = \frac{K-kL+k}{k}$ . Hence, for any integer  $s$  present in the matrix  $\mathbf{A}$ , the sub-matrix given by (17) satisfies the condition C3 in Definition 1.

In short, if we take any integer  $s$  in the matrix  $\mathbf{P}_1$ , the matrix  $\mathbf{P}_1$  satisfies the condition C3 in Definition 1. All the three conditions C1, C2' and C3 are satisfied by the matrix  $\mathbf{P}_1$ . Hence it is a  $\frac{2K}{K-kL+k}$ -regular  $(\frac{K}{k}, \frac{K}{k}, L, \frac{(K-kL)(K-kL+k)}{2k^2})$  PDA.

Now, we need to prove that the matrix  $\mathbf{P}_1$  is  $k$ -cyclic. Consider the  $0^{\text{th}}$  column block in the block matrix  $\mathbf{P}_1$ :

$$\mathbf{C}_0 = \begin{pmatrix} \mathbf{A} \\ \mathbf{X} \\ \vdots \\ \mathbf{X} \\ \mathbf{A}^T \end{pmatrix}. \quad (18)$$

There are  $\frac{K-kL+k}{k}$  columns and  $\frac{K}{k}$  rows in the matrix  $\mathbf{C}_0$ , indexed by  $[0, \frac{K-kL}{k}]$  and  $[0, \frac{K}{k})$  respectively. Recall that all the diagonal entries and the entries below the diagonal are stars in the matrix  $\mathbf{A}$ . For the transpose matrix  $\mathbf{A}^T$ , all the diagonal entries and the entries above the diagonal are stars.



$k$ -cyclic if the position of stars in each column is obtained by shifting the stars in the previous column towards down by  $k$  units. The first  $\frac{K}{k}$  columns in  $\mathbf{P}$  is  $k$ -cyclic since the matrix  $\tilde{\mathbf{P}}_1$  is  $k$ -cyclic. We have assumed that  $k|K$ , hence shifting the position of stars in the  $(\frac{K}{k}-1)^{th}$  column of  $\tilde{\mathbf{P}}_1$  towards down by  $k$  units will result in the pattern of stars in the  $0^{th}$  column of  $\tilde{\mathbf{P}}_1$ . The pattern of stars in the  $0^{th}$  column of  $\tilde{\mathbf{P}}_1$  and  $\tilde{\mathbf{P}}_2$  is the same. Hence shifting the position of stars in the  $(\frac{K}{k}-1)^{th}$  column of  $\mathbf{P}$  towards down by  $k$  units will result in the pattern of stars in the  $(\frac{K}{k})^{th}$  column of  $\mathbf{P}$ . Similarly after every  $\frac{K}{k}$  columns the pattern repeats. So, the PDA  $\mathbf{P}$  is  $k$ -cyclic. This completes the proof.

## V. DISCUSSION

In this work, we have constructed a new class of PDAs which we call as  $t$ -cyclic  $g$ -regular PDA. This class of PDA is used for providing the delivery scheme for multi-access coded caching problems when  $\gamma \in \{\frac{k}{K} : k|K, (K-kL+k)|K, k \in [1, K]\}$ . The sub-packetization level required for our scheme is the least compared to the state-of-the-art schemes. For certain ranges of values of  $L$ , the transmission rate is also less compared to some of the existing schemes. We have obtained a scheme based on PDA only for certain ranges of values of  $\gamma$ . Obtaining a scheme based on PDA with linear sub-packetization for the entire range of  $\gamma$  is an interesting problem to work on.

## ACKNOWLEDGEMENT

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.

## REFERENCES

- [1] M. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2856–2867, 2014.
- [2] K. Wan, D. Tuninetti and P. Piantanida, "On the optimality of uncoded cache placement," in *Information Theory Workshop (ITW), 2016 IEEE*, pp. 161–165.
- [3] Q. Yu, M. A. Maddah-Ali and A. S. Avestimehr, "The Exact Rate-Memory Tradeoff for Caching With Uncoded Prefetching," in *IEEE Transactions on Information Theory*, vol. 64, no. 2, pp. 1281–1296, Feb. 2018.
- [4] S. Jin, Y. Cui, H. Liu and G. Caire, "Uncoded placement optimization for coded delivery," in *IEEE 16<sup>th</sup> International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2018, pp. 1–8.
- [5] J. Gómez-Vilardebó, "Fundamental Limits of Caching: Improved Rate-Memory Tradeoff With Coded Prefetching," in *IEEE Transactions on Communications*, vol. 66, no. 10, pp. 4488–4497, Oct. 2018
- [6] U. Niesen and M. A. Maddah-Ali, "Coded Caching With Nonuniform Demands," in *IEEE Transactions on Information Theory*, vol. 63, no. 2, pp. 1146–1158, Feb. 2017
- [7] E. Ozfatura and D. Guenduez, "Uncoded Caching and Cross-Level Coded Delivery for Non-Uniform File Popularity," 2018 *IEEE International Conference on Communications (ICC)*, Kansas City, MO, 2018, pp. 1–6
- [8] M. A. Maddah-Ali and U. Niesen, "Decentralized Coded Caching Attains Order-Optimal Memory-Rate Tradeoff," in *IEEE/ACM Transactions on Networking*, vol. 23, no. 4, pp. 1029–1040, Aug. 2015
- [9] M. Ji, G. Caire and A. F. Molisch, "Fundamental Limits of Caching in Wireless D2D Networks," in *IEEE Transactions on Information Theory*, vol. 62, no. 2, pp. 849–869, Feb. 2016.

- [10] Q. Yan, M. Cheng, X. Tang and Q. Chen, "On the Placement Delivery Array Design for Centralized Coded Caching Scheme," in *IEEE Transactions on Information Theory*, vol. 63, no. 9, pp. 5821–5833, Sept. 2017.
- [11] M. Cheng, J. Jiang, X. Tang and Q. Yan, "Some Variant of Known Coded Caching Schemes With Good Performance," in *IEEE Transactions on Communications*, vol. 68, no. 3, pp. 1370–1377, March 2020.
- [12] M. Cheng, J. Jiang, Q. Wang and Y. Yao, "A Generalized Grouping Scheme in Coded Caching," in *IEEE Transactions on Communications*, vol. 67, no. 5, pp. 3422–3430, May 2019.
- [13] M. Cheng, J. Jiang, Q. Yan and X. Tang, "Constructions of Coded Caching Schemes With Flexible Memory Size," in *IEEE Transactions on Communications*, vol. 67, no. 6, pp. 4166–4176, June 2019.
- [14] J. Michel and Q. Wang, "Placement Delivery Arrays From Combinations of Strong Edge Colorings," in *IEEE Transactions on Communications*, vol. 68, no. 10, pp. 5953–5964, Oct. 2020.
- [15] C. Shangguan, Y. Zhang and G. Ge, "Centralized Coded Caching Schemes: A Hypergraph Theoretical Approach," in *IEEE Transactions on Information Theory*, vol. 64, no. 8, pp. 5755–5766, Aug. 2018.
- [16] Q. Yan, X. Tang, Q. Chen and M. Cheng, "Placement Delivery Array Design Through Strong Edge Coloring of Bipartite Graphs," in *IEEE Communications Letters*, vol. 22, no. 2, pp. 236–239, Feb. 2018.
- [17] X. Zhong, M. Cheng and J. Jiang, "Placement Delivery Array Based on Concatenating Construction," in *IEEE Communications Letters*, vol. 24, no. 6, pp. 1216–1220, June 2020
- [18] J. Hachem, N. Karamchandani and S. N. Diggavi, "Coded caching for multi-level popularity and access," *IEEE Transactions on Information Theory*, vol. 63, no. 5, pp. 3108–3141, 2017.
- [19] K. S. Reddy and N. Karamchandani, "Rate-memory trade-off for multi-access coded caching with uncoded placement," *IEEE Transactions on Communications*, Vol. 68, No. 6, pp. 3261–3274, 2020.
- [20] B. Serbetci, E. Parrinello and P. Elia, "Multi-access coded caching: gains beyond cache-redundancy," 2019 *IEEE Information Theory Workshop (ITW)*, Visby, Sweden, 2019, pp. 1–5.
- [21] S. Sasi, B. S. Rajan, "An Improved Multi-access Coded Caching with Uncoded Placement," arXiv:2009.05377v2 [cs.IT], Oct 2020.
- [22] M. Cheng, D. Liang, K. Wan, M. Zhang, and G. Caire, "A Novel Transformation Approach of Shared-link Coded Caching Schemes for Multiaccess Networks," arXiv:2012.04483 [cs.IT], Dec 2020.
- [23] K. S. Reddy, N. Karamchandani, "Structured Index Coding Problem and Multi-access Coded Caching," arXiv:2012.04705 [cs.IT], Dec 2020.
- [24] A. A. Mahesh, B. S. Rajan, "A Coded Caching Scheme with Linear Sub-packetization and its Application to Multi-Access Coded Caching," arXiv:2009.10923 [cs.IT], Sep 2020.
- [25] Digvijay Katyay, Pooja Nayak M, and B. Sundar Rajan, "Multi-access Coded Caching Schemes From Cross Resolvable Designs," *IEEE Transactions on Communications*, (Available as early access article in IEEE Xplore).