Cooperative Coded Caching in Internet of Vehicles Based on Coded Prefetching

Yifan Lin and Congduan Li(✉)

School of Electronics and Communication Enginerring,
Sun Yat-sen University, Shenzhen 518107, China
linyf28@mail2.sysu.edu.cn, licongd@mail.sysu.edu.cn

Abstract. A lot of networks have high temporal variability, which means in the peak hours, the users' requests may exceed the loading capacity, cause congestion. During the off-peak hours, the network resources is underutilized. Caching is a technique to shift the traffic from peak to off-peak hours, by prefetching some content at or near the end users. Coded caching can reduce the peak rate further by jointly optimizing the placement and delivery. In this paper, we consider a novel coded caching model with allowing interaction between cooperative users, and propose a centralized scheme. We use Greedy Constrained Coloring to exploit the multicasting opportunities, and coded prefetching to reduce the redundancy in the cache due to the interaction between users.

Keywords: Network coding · Coded caching · Internet of Vehicles

1 Introduction

Caching is a technique to reduce the traffic rate during the peak hours by placing some contents at or near the end users in advance. Based on it, coded caching, first proposed by Maddah-Ali and Niesen in 2014 [7], achieves a much better performance than the uncoded caching by jointly optimizing the delivery and placement phase, where the multicasting opportunities can be created so we can serve different demands by a single coded message. The well-known Maddah-Ali-Niesen (MAN) scheme was proposed in [7]. After that, various branches of coded caching problems have been studied. The decentralized system model was studied by Maddah-Ali and Niesen [8], in which users pick the contents to cache independently at the placement phase. They also studied the model with nonuniform user demands [10], creating the multicasting opportunities by grouping the users. Ji et al. give the order-optimal delivery rate on the nonuniform demands model [6], by proposing the Random Least Frequently Used placement with Greedy Constrained Coloring delivery (RLFU-GCC) scheme.

Shared caching problem, which allows multiple users to share a single cache memory, is also studied by lots of researchers. Parrinello et al. proposed the optimum caching scheme with uncoded prefetching for a shared caches system [11].

X. Chen et al. (Eds.): IoTaaS 2023, LNICST 585, pp. 396–413, 2025.
https://doi.org/10.1007/978-3-031-70507-6_30

Asadi et al. studied the centralized caching with shared caches in heterogeneous cellular networks [1]. Decentralized coded caching for shared caching is studied by Dutta et al. [2]. Ibrahim et al. studied a D2D model with distinct cache size [4]. These works focus the model that users can be divided into several groups that users in a same group can communicate or share the cache with each other, or the model that all the users can communicate with each other. Ji et al. studied the coded caching system in wireless D2D network [5], where users in a grid network can communicate with other users within a certain distance, and at the delivery phase, the users will be completely served by other users, without the base station. Then based on this model, Çağkan Yapar et al. proposed a optimal scheme with uncoded cache placement and one-shot delivery.

In order to minimize efficiency loss, wireless communication is typically utilized only for the final leg in large scale communication networks. This unique structure allows for improvement of the interface between the physical and data link layers as a means of mitigating architectural inefficiencies within the wireless portion of the network [13].

Multi-access coded caching (MACC) system, which allows single user to access multiple caches, is first studied by Hachem et al. in 2017 [3]. A (K, N, L, M)-MACC system refer to a coded caching system with N files, K users and caches and each user has the access to L caches with a cyclic wrap-around. Actually, our proposed system model is a limited MACC system(see Sect. 2 and Sect. 5) Hachem et al. proposed the MACC system model in [3] with multi-level files popularity settings and a coloring-based scheme. MACC system with uncoded prefetching was studied by Reddy et al. [12]. In [12], the authors proposed a new upper bound of the rate sometimes lower than the rate in [3], an order-optimal rate for MACC system with $L > K/2$, and the exact optimal rate for some special cases. Namboodiri et al. derived a tighter lower bound by adopting the sliding-window subset entropy inequality in the proof [9].

The above MACC systems are symmetric in both user indexing and file indexing [14]. The system model we propose in this paper shares similarities with MACC system but deviates in symmetry in file indexing, constituting a significant divergence from existing system models.

On the other hand, along with the development of 5G, the researches on Internet of Vehicles (IoV) develop rapidly. IoV contains vehicle-to-vehicle (V2V) communication, vehicle-to-RSU (Road side unit) communication and so on. In this paper, we apply the coded caching to the IoV scene, modeling it as a coded caching system with constrained D2D communication. In our model, all the users line up in a row geometrically, corresponding to the single lane model in IoV field. The RSU will act as the base station.

In this paper, we study a MACC-like model and make the following contributions:

– A delivery scheme based on greedy constrained coloring is introduced for our proposed model, which can reduce the rate over the shared link by merging the user cooperation sets to reduce the transmissions times.

– A coded placement technique based on MAN placement in [7] is introduced for our proposed model, which can reduce the memory size requirement with keeping the cache content accessible to users unchanged.
– A new achievable rate is derived based on a scheme with coded placement and graph coloring.

The rest of the paper is organized as follows. Section 2 briefly describes the system model. Section 3 describes our delivery scheme. Section 4 describes our coded placement scheme. Section 5 describes a cut-set type lower bound and shows the performance of our proposed scheme.

2 System Model

As shown in Fig. 1, we consider a centralized model based on the model in [7], K users are connected to a server through a error-free, shared link. The server contains N files of equal size F bits, labeled as $W_1, W_2, \cdots, W_N$. Each user k is equipped with a cache memory Z_k of size MF bits. Each user can access not only its own cache, but also its adjacent users' cache(s), assuming that all the users line up in a row by its label. In the following discussion,we refer a user's adjacent users as its cooperative users. For example, in Fig. 1, user 2 have the access of the caches Z_1 and Z_3, and its own cache Z_2, means that it can access a total of three users' caches.

In particular, the first user and the last user in the row can only access a total of two users' caches. For example, user 1 only have the access of Z_1 and Z_2 due to the fact that there is no user on its left side. Similarly, user 5 only have the access of Z_4 and Z_5. Fig. 2 shows the corresponding IoV model.

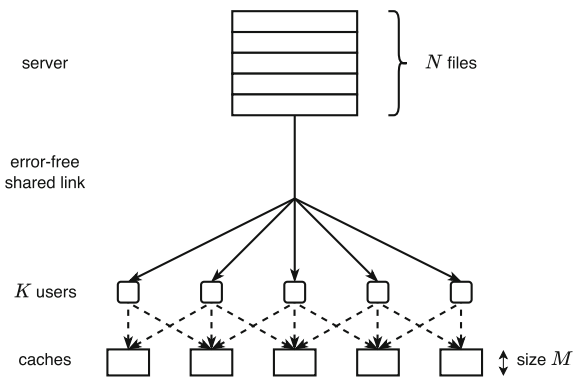

Fig. 1. Caching system considered in this paper. A server containing N files is connected to K users through a shared error-free link. Each user is equipped with a cache of size MF bits, and have the access of its own and its cooperative users' caches. In this figure, $K = N = 5$. User 1 and K only have the access of two caches because they are the "edge" users.

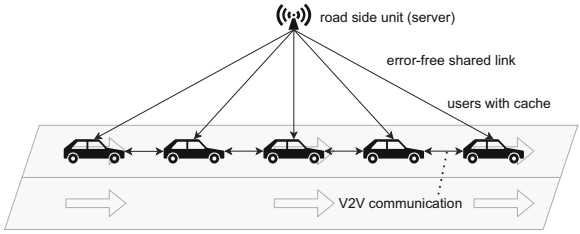

Fig. 2. The IoV model corresponding to the coded caching system model. The road side unit act as the central server and the vehicles act as the users with cache.

To simplify the problem, we assume that users can get contents from its cooperative users' cache *at no cost*.

The operation of a coded caching system will be divided into two phases. In the placement phase, all the caches will be filled without the knowledge of the user demands. In the delivery phase, each user requests exactly one file W_{d_i} from the server. Then the server transmits a series of multicast message through the error-free shared link according to all user demands $\mathbf{d} = (d_1, \cdots, d_K)$. Each user can recover the file required with the transmitted messages and the cached contents it can access. The goal is to design the placement phase and the delivery phase to minimize the peak rate on the shared link.

For ease of reading, the following notations and rules are adopted.

- $W_{i,\mathcal{U}}$: subfile of file W_i, which stored in the caches of all the users in $\mathcal{U}$.
- For any positive integer K and integer $q < K$, let $[K] = \{1, 2, \ldots, K\}$ and $\binom{[K]}{q} = \{\mathcal{D} | \mathcal{D} \subseteq [K], |\mathcal{D}| = q\}$. When $q \leq 0$, $\binom{[K]}{q} = \emptyset$.
- $\mathcal{W}$: set of subfiles.
- Let $\mathcal{W}_1, \mathcal{W}_2$ be two sets of subfiles, $|\mathcal{W}_1| = |\mathcal{W}_2| = n$, $w_{i,j}$ be the j-th subfile in $\mathcal{W}_i$, $\mathcal{W}_1 \oplus \mathcal{W}_2$ donates a set of coded subfiles i.e. $\mathcal{W}_1 \oplus \mathcal{W}_2 = \{w_{1,1} \oplus w_{2,1}, w_{1,2} \oplus w_{2,2}, \ldots, w_{1,n} \oplus w_{2,n}\}$.
- For the calculation of user index in the following discuss, we follow a cyclic wrap-around rule, i.e. user -1, 0 refer to user $K-1$, K, and user $K+1$, $K+2$ refer to user 1, 2, etc.

3 Delivery Phase Design

Our research base on the scheme proposed by Maddah Ali and Niesen in [7], which will be called as the MAN scheme.

In the MAN scheme, we select s users in $[K]$ to form a user cooperation set $\mathcal{S}$. For each user cooperation set, a message

$$X_{\mathcal{S}} = \oplus_{k \in \mathcal{S}} W_{d_u, \mathcal{S} \backslash k}, \tag{1}$$

is generated based on the users' demand and caches, and sent to all users in the set. This ensures that each user in S can decode a subfile of the file it

requested upon receiving the message. After generating and sending the messages to all possible user cooperation sets, all users should be able to obtain the file they requested. In our settings, the delivery phase needs to be redesign to aviod the message redundancy.

Example 1. This example shows that how the message redundancy appears. Consider a system with $K = 8$ users, $N = 8$ files and each user equipped with a cache size of $M = 3$. According to the MAN scheme, the size of the user cooperation set $s = MK/N + 1 = 4$.

Taking the user cooperation set $\mathcal{S} = \{1, 2, 4, 6\}$ as an example, assuming $d_i = W_i$, the message

$$X_{\mathcal{S}} = W_{d_1,\{2,4,6\}} \oplus W_{d_2,\{1,4,6\}} \oplus W_{d_4,\{1,2,6\}} \oplus W_{d_6,\{1,2,4\}},$$

is generated then multicasted to all users in $\mathcal{S}$.

For user 1, it have the access of Z_2, meanwhile the subfile $W_{d_1,\{2,4,6\}}$ is stored in Z_2. Therefore this transmitted message makes no contribution to user 1.

However, for user 4, it still needs $X_{\mathcal{S}}$ to recover the subfile it requested, due to the fact that the subfile $W_{d_4,\{1,2,6\}}$ does not exist in any cache it have the access, i.e. Z_3, Z_4 and Z_5. □

Here we give some definitions.

Definition 1. *For a user $u \in [K]$, let*

$$\mathcal{A}(u) = \begin{cases} \{u+1\} & ,u = 1, \\ \{u-1, u+1\} & ,1 < u < K, \\ \{u-1\} & ,u = K, \end{cases} \tag{2}$$

$$\mathcal{Z}(u) = \mathcal{A}(u) \cup \{u\}, \tag{3}$$

refer to the adjacent user(s) of u and the caches that u can access, respectively.

For a user cooperation set $\mathcal{S}$, let

$$\mathcal{T}_{\mathcal{S}} = \{u \in \mathcal{S} | \mathcal{A}(u) \cap \mathcal{S} = \emptyset\}, \tag{4}$$

$$\mathcal{N}_{\mathcal{S}} = \{u \in \mathcal{S} | \mathcal{A}(u) \cap \mathcal{S} \neq \emptyset\}, \tag{5}$$

refer to its target users set and non-target users set, respectively

For a user $k \in \mathcal{S}$, if none of its adjacent user(s) is the member of $\mathcal{S}$, we say user k is a target user of $\mathcal{S}$, which means it needs the message $X_{\mathcal{S}}$ to recover the file it requested; If any of its adjacent user(s) is the member of $\mathcal{S}$, we say user k is a non-target user of $\mathcal{S}$, which means the message $X_{\mathcal{S}}$ makes no contribution to it.

With the definitions above, we have $\mathcal{T}_{\mathcal{S}} = \{4, 6\}$, $\mathcal{N}_{\mathcal{S}} = \{1, 2\}$ in Example 1.

For each user cooperation set $\mathcal{S}$, due to the fact that the message $X_{\mathcal{S}}$ makes no contribution to the users in $\mathcal{N}_{\mathcal{S}}$, the message generated only for the users in $\mathcal{T}_{\mathcal{S}}$, i.e.

$$X_{\mathcal{S}} = \oplus_{u \in \mathcal{T}_{\mathcal{S}}} W_{d_u, S \setminus \{u\}} \tag{6}$$

Based on the number of target users, the user cooperation sets can be classified into three categories.

1. For the user cooperation sets with $\mathcal{T}_{\mathcal{S}} = \emptyset$, they will be omitted since they make no contribution to any users.
2. For the user cooperation sets with $\mathcal{T}_{\mathcal{S}} = \mathcal{S}$, the corresponding generated messages $X_{\mathcal{S}}$ will be multicasted to their respective target users.
3. For the user cooperation sets with $\mathcal{T}_{\mathcal{S}} \subset \mathcal{S}, \mathcal{T}_{\mathcal{S}} \neq \emptyset$, some messages will be merged into a single one to exploit the multicasting opportunities and reduce the transmission rate on the shared link. The following part in this section describes the specific merging approach.

Our scheme is based on the conflict graph and Greedy Constrained Coloring(GCC). Given

$$\mathbf{S} = \{\mathcal{S} | \mathcal{T}_{\mathcal{S}} \subset \mathcal{S}, \mathcal{T}_{\mathcal{S}} \neq \emptyset\},$$

the conflict graph is defined as follows:

Definition 2. *The conflict graph $\mathcal{H}_{\mathbf{S}}$ is formed by:*

- ***Vertex set**: The set of vertices in $\mathcal{H}_{\mathbf{S}}$ is the set of all user cooperation sets in $\mathbf{S}$. The vertex corresponding to the user cooperation set $\mathcal{S}_i$ is represented by v_i.*
- ***Edge set**: There exists an edge (v_i, v_j) connecting vertices v_i and v_j in $\mathcal{H}_{\mathbf{S}}$ if and only if the following merge condition is NOT satisfied:*

$$\begin{aligned} &\bigwedge_{u \in \mathcal{T}_{\mathcal{S}_i}} \left(\mathcal{Z}(u) \cap \mathcal{N}_{\mathcal{S}_j} \neq \emptyset\right) \\ \wedge &\bigwedge_{u \in \mathcal{T}_{\mathcal{S}_j}} \left(\mathcal{Z}(u) \cap \mathcal{N}_{\mathcal{S}_i} \neq \emptyset\right) \end{aligned}, \tag{7}$$

□

We utilize the GCC algorithm, as presented in Algorithm 1, to assign colors to all the vertices in $\mathcal{H}_{\mathbf{S}}$.

Algorithm 1. Greedy Constrained Coloring

```
Initialize V = Vertex-set(H_S).
while V ≠ ∅ do
    Pick any v ∈ V, and let I = {v}
    for all v' ∈ V\I do
        if There is no edge between v' and I then
            I ← I ∪ v'
        end if
    end for
    Color all the vertices of resulting set I by an unused
     color.
    Let V ← V\I
end while
```

Next, the user cooperation sets of the same color will jointly generate a single message

$$X_\Sigma = \oplus_{\mathcal{S}\in\Sigma}\left[\oplus_{u\in\mathcal{T}_\mathcal{S}} W_{d_u,\mathcal{S}\backslash\{u\}}\right]$$

where Σ is the collection of user cooperation sets $\mathcal{S}$ of the same color. By multicasting this message to all users in $\bigcup_{\mathcal{S}\in\Sigma}\mathcal{T}_\mathcal{S}$, we effectively decrease the amount of message that needs to be transmitted from $|\Sigma|$ to just one.

Theorem 1. *Letting $\chi(\mathcal{H}_\mathbf{S})$ donate the chromatic number of $\mathcal{H}_\mathbf{S}$, $M \in \{0, N/K, 2N/K, \ldots, N\}$, $t = MK/N$, an upper bound of the delivery rate on the shared link*

$$R_{GCC}(M) = \left(\binom{K-t}{t+1} + \chi(\mathcal{H}_\mathbf{S})\right)\frac{1}{\binom{K}{t}}. \tag{8}$$

The first factor is the total number of coded messages we should send over the shared link, each size of $1/\binom{K}{t}$, which is the second factor in $R_{GCC}(M)$. $\binom{K-t}{t+1}$ is the number of user cooperation sets with $\mathcal{T}_\mathcal{S} = \mathcal{S}$. Note that the size of each coded message is also the size of each subfile, normalized by the file size.

The rate $R_{GCC}(M)$ is achieved by our proposed scheme. We formally describe its delivery phase in Algorithm 2. Its placement phase is the same as MAN scheme, described in Algorithm 1 in [7].

Algorithm 2. Coded Caching Delivery

Require: $N, K, M \in \{0, N/K, 2N/K, \ldots, N\}$

1: $s \leftarrow MK/N + 1$
2: $\mathfrak{S} \leftarrow \{\mathcal{S} \subseteq \{1, \ldots, K\} : |\mathcal{S}| = s\}$
3: $X_1 \leftarrow (\oplus_{u\in\mathcal{S}} W_{d_u,\mathcal{S}\backslash\{u\}} : \mathcal{S} \in \mathfrak{S}, l_\mathcal{S} = 0)$
4: $\mathbf{S} \leftarrow \{\mathcal{S} \in \mathfrak{S} : 2 \le l_\mathcal{S} < s\}$
5: Generate the conflict graph $\mathcal{H}_\mathbf{S}$ and color it with GCC.
6: $\mathfrak{E} \leftarrow \{\Sigma$: collection of user cooperation sets with same color.$\}$
7: $X_2 \leftarrow (\oplus_{\mathcal{S}\in\Sigma}\left(\oplus_{u\in\mathcal{T}_\mathcal{S}} W_{d_u,\mathcal{S}\backslash\{u\}}\right) : \Sigma \in \mathfrak{E})$
8: Send X_1 and X_2 to users.

Remark 1. The merge condition (7) consists of two symmetric subconditions. Focus on the first one, it means for each user u in the target users set of $\mathcal{S}_i$, at least one user in $\mathcal{Z}(u)$ should be the member of $\mathcal{N}_{\mathcal{S}_j}$, the non-target users set of $\mathcal{S}_j$.

Focus on a subset $\mathcal{S}'$, for a user $k \in \mathcal{S}', \mathcal{S}' \in \Sigma$, the coded message it received can be rewritten in

$$\begin{aligned} X_\Sigma = & W_{d_k,\mathcal{S}'\backslash\{k\}} \\ & \oplus \bigoplus_{u\in\mathcal{T}_{\mathcal{S}'}\backslash\{k\}} W_{d_u,\mathcal{S}'\backslash\{u\}} \\ & \oplus \bigoplus_{\mathcal{S}\in\Sigma\backslash\{\mathcal{S}'\}} \left(\bigoplus_{u\in\mathcal{T}_\mathcal{S}} W_{d_u,\mathcal{S}\backslash\{u\}}\right) \end{aligned}. \tag{9}$$

The first term is the subfile user k requests. The second term is the subfiles that the other target users in $\mathcal{S}'$ request, which are cached in the cache of user k. The third term is the subfiles that the target users in $\mathcal{S} \in \Sigma \backslash \{\mathcal{S}'\}$ request, which are cached in at least one cache of users in $\mathcal{Z}(k)$ due to

$$\mathcal{N}_{\mathcal{S}} \subseteq \mathcal{S} \backslash \{u\} \wedge \mathcal{Z}(k) \cap \mathcal{N}_{\mathcal{S}} \neq \emptyset \Rightarrow \mathcal{Z}(k) \cap \mathcal{S} \backslash \{u\} \neq \emptyset$$

Example 2. Consider a network with $K = N = 5, M = 2$, then $t = 2, s = 3$. Assume that $d_i = W_i$, we have a total of $\binom{K}{s} = 10$ user cooperation sets. For $\mathcal{S}_1 = \{1,2,3\}, \mathcal{S}_7 = \{2,3,4\}, \mathcal{S}_{10} = \{3,4,5\}$, they are omitted due to $\mathcal{T}_{\mathcal{S}_i} = \emptyset, i = 1,7,10$. For $\mathcal{S}_5 = \{1,3,5\}$ with $\mathcal{T}_{\mathcal{S}_5} = \mathcal{S}_5$, the signal

$$X_{\mathcal{S}_5} = W_{1,\{3,5\}} \oplus W_{3,\{1,5\}} \oplus W_{5,\{1,3\}},$$

is generated and multicasted to user $1, 3$ and 5.

Then for the left six sets, let $\mathbf{S} = \{\mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4, \mathcal{S}_6, \mathcal{S}_8, \mathcal{S}_9\}$, generate the conflict graph $\mathcal{H}_{\mathbf{S}}$ and color it by GCC. The conflict graph and coloring result are shown in Fig. 3.

Hence, the messages of $\mathcal{S}_2$ = and $\mathcal{S}_4$ will be merged, resulting in $\Sigma_1 = \{\mathcal{S}_2, \mathcal{S}_4\}$ and the corresponding message

$$X_{\Sigma_1} = W_{4,\{1,2\}} \oplus W_{1,\{3,4\}}.$$

By multicasting X_{Σ_1} to user 1 and 4, user 1 can recover the subfile $W_{1,\{3,4\}}$ and user 4 can recover the subfile $W_{4,\{1,2\}}$.

Similarly, for $\Sigma_2 = \{\mathcal{S}_3, \mathcal{S}_6\}$ and $\Sigma_3 = \{\mathcal{S}_8, \mathcal{S}_9\}$, the messages

$$X_{\Sigma_2} = W_{5,\{1,2\}} \oplus W_{1,\{4,5\}}$$

$$X_{\Sigma_3} = W_{2,\{4,5\}} \oplus W_{5,\{2,3\}}$$

are generated and multicasted to the corresponding users, respectively.

After all the messages sent over the shared link, user 1 has access to 4 subfiles of W_1 in its own cache:

$$W_{1,\{1,2\}}, W_{1,\{1,3\}}, W_{1,\{1,4\}}, W_{1,\{1,4\}},$$

has access to 3 subfiles of W_1 in user 2's cache:

$$W_{1,\{2,3\}}, W_{1,\{2,4\}}, W_{1,\{2,5\}},$$

can solve $W_{1,\{3,5\}}$ from $X_{\mathcal{S}_5}$, solve $W_{1,\{3,4\}}$ from X_{Σ_1}, solve $W_{1,\{4,5\}}$ from X_{Σ_2}. Therefore user 1 can recover its required file W_1 because it has access to all the subfiles of W_1. Similarly for other users.

Hence all users can recover its required file after four coded messages sent, each size of $1/\binom{K}{t}$. □

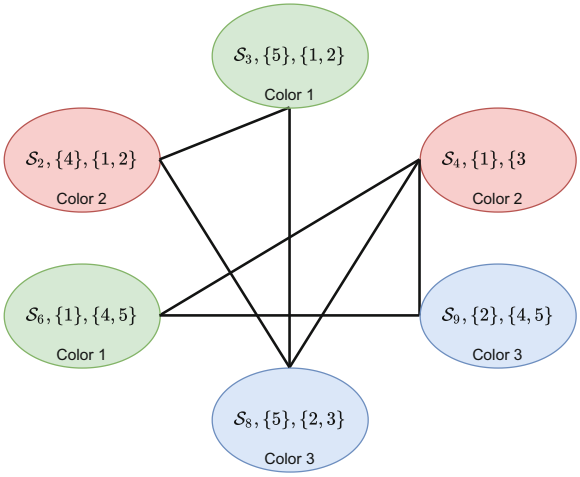

Fig. 3. Conflict graph of Example 2 with coloring by GCC, the vertices are labeled by { the user cooperation set, its target users set, its non-target users set }.

4 Placement Phase Design

In this section we focus on the design of the placement phase and a scheme base on the MAN placement is proposed. In the MAN scheme, each file is devided into $\binom{K}{t}$ subfiles of equal size, where $t = MK/N$. Each subfile will be stored in t users' caches to create more multicast opportunities. In our system model, there is some redundancy in the cache space that each user can access, due to the cache sharing between cooperative users, shown in Example 3.

Example 3. Consider a system with $K = 6$ users, $N = 6$ files, each user equipped with a cache of size $M = 2$, the MAN scheme gives the placement of Table 1, where $n \in \{1, \ldots, 6\}$, each column refer to a user's cache. For the conciseness, when there is no ambiguity, the brackets and commas in the labels of subfiles are omitted, e.g. $W_{n,12}$ means $W_{n,\{1,2\}}$.

For user 2, $W_{n,12}, W_{n,23}, W_{n,13}$ appear twice in the caches it can access, respectively.

Table 1. Placement under MAN scheme in Example 3

User 1	User 2	User 3	User 4	User 5	User 6
$W_{n,12}$	$W_{n,23}$	$W_{n,34}$	$W_{n,45}$	$W_{n,56}$	$W_{n,16}$
$W_{n,13}$	$W_{n,24}$	$W_{n,35}$	$W_{n,46}$	$W_{n,15}$	$W_{n,26}$
$W_{n,14}$	$W_{n,25}$	$W_{n,36}$	$W_{n,14}$	$W_{n,25}$	$W_{n,36}$
$W_{n,15}$	$W_{n,26}$	$W_{n,13}$	$W_{n,24}$	$W_{n,35}$	$W_{n,46}$
$W_{n,16}$	$W_{n,12}$	$W_{n,23}$	$W_{n,34}$	$W_{n,45}$	$W_{n,56}$

Based on the MAN placement scheme, we use **coded prefetching** to reduce the cache size M, on the premise that the contents that each user can access remain unchanged. We propose two schemes for different case.

Let $t = MK/N$, we consider the system that t is an integer, i.e. $M \in 0, N/K, 2N/K, \ldots, N$. Besides, there is no redundancy at the case that $t = 1$, so the following two schemes only consider the case that $t \geq 2$.

4.1 Coded Prefetching Scheme A

Scheme A is suitable for the case where t is small, we first give the result:

$$t'_A = t * \left(1 - \frac{\binom{K-4}{t-2} + \binom{K-4}{t-3}}{\binom{K-1}{t-1}}\right) \tag{10}$$

i.e. we can almost achieve the rate $R_{GCC}(M), M = tN/K$ at $M' = t'_A N/K$. In this scheme, we need to send some extra packets in delivery phase. The rate-memory tradeoff

$$R_A(M') = R_{GCC}(M) + \frac{\binom{K-4}{t-2}}{\binom{K}{t}}. \tag{11}$$

Here is the describe of the scheme A. After placing the subfiles according to the MAN scheme, we proceed to modify the cache contents through two steps:

- **Step 1**: For each user i, delete the subfiles $\mathcal{W}_1(i) = (W_{n,\{i-2,i,i+1\}\cup\mathcal{X}})$ and $\mathcal{W}_2(i) = (W_{n,\{i-2,i-1,i\}\cup\mathcal{X}})$, then place the coded subfiles

$$\mathcal{W}_1(i) \oplus \mathcal{W}_2(i) = (W_{n,\{i-2,i,i+1\}\cup\mathcal{X}} \oplus W_{n,\{i-2,i-1,i\}\cup\mathcal{X}}),$$

  where $\mathcal{X} \in \binom{[K]\backslash\{i-2,i-1,i,i+1\}}{t-3}$. For the case where $t = 2$, Step 1 is omitted.

  In Step 1, we delete $2\binom{K-4}{t-3}$ subfiles, then place $\binom{K-4}{t-3}$ coded subfiles.

- **Step 2**: For each user i, delete the subfiles $\mathcal{W}_3(i) = (W_{n,\{i,i+1\}\cup\mathcal{X}})$ and $\mathcal{W}_4(i) = (W_{n,\{i-2,i\}\cup\mathcal{X}})$, then place the coded subfiles

$$\mathcal{W}_3(i) \oplus \mathcal{W}_4(i) = (W_{n,\{i,i+1\}\cup\mathcal{X}} \oplus W_{n,\{i-2,i\}\cup\mathcal{X}}),$$

  where $\mathcal{X} \in \binom{[K]\backslash\{i-2,i-1,i,i+1\}}{t-2}$.

In Step 2, we delete $2\binom{K-4}{t-2}$ subfiles, then place $\binom{K-4}{t-2}$ subfiles. In delivery phase, we should send $\mathcal{W}_4(2)_{d_1} \oplus \mathcal{W}_3(K)_{d_K}$ to the "edge" user i.e. user 1 and user K.

The proof of the fact that the contents that each user can access remain unchanged after these two steps is shown in Appendix A.

Note that in MAN scheme we place $\binom{K-1}{t-1}$ subfiles in each cache, and the extra messages we send in delivery phase is of size $\binom{K-4}{t-2}/\binom{K}{t}$, we have (10) and (11).

4.2 Coded Prefetching Scheme B

Scheme B is suitable for bigger t, we first give the result:

$$t'_B = t * \left(1 - \frac{\binom{K-2}{t-2}}{2\binom{K-1}{t-1}}\right) \tag{12}$$

i.e. we can achieve the rate $R_{GCC}(M), M = tN/K$ at $M' = t'_B N/K$. The rate-memory tradeoff

$$R_B(M') = R_{GCC}(M). \tag{13}$$

Here is the describe of the scheme B. In Scheme B, we first divide each file W_n into two parts of equal size W_n^1, W_n^2 and place them in two steps.

- **Step 1**: Place all the files W_n^1 in MAN scheme, which fills half of the cache. For each user i numbered even, $i \neq K$, delete the subfiles $\mathcal{W}_5(i) = (W^1_{n,\{i-1,i,i+1\}\cup\mathcal{X}}), \mathcal{W}_6(i) = (W^1_{n,\{i-1,i\}\cup\mathcal{Y}}), \mathcal{W}_7(i) = (W^1_{n,\{i,i+1\}\cup\mathcal{Y}})$, then place the coded subfiles

$$\mathcal{W}_6(i) \oplus \mathcal{W}_7(i) = (W^1_{n,\{i-1,i\}\cup\mathcal{Y}} \oplus W^1_{n,\{i,i+1\}\cup\mathcal{Y}}),$$

  where $\mathcal{X} \in \binom{[K]\backslash\{i-1,i,i+1\}}{t-3}, \mathcal{Y} \in \binom{[K]\backslash\{i-1,i,i+1\}}{t-2}$. If K is even, then for user K, delete the subfiles $W^1_{n,\{K-1,K\}\cup\mathcal{Q}}$, where $\mathcal{Q} \in \binom{[K]\backslash\{K-1,K\}}{t-2}$.

In Step 1, each user numbered even can still access the subfiles deleted from adjacent users' caches. Each user j numbered odd can access $\mathcal{W}_6(j+1)$ and $\mathcal{W}_7(j-1)$ so it can recover $\mathcal{W}_7(j+1)$ and $\mathcal{W}_6(j-1)$. Therefore after Step 1, the contents(W_n^1) that each user can access remain unchanged for each user.

- **Step 2**: Place all the files W_n^2 in MAN scheme, which fills rest half of the cache. For each user i numbered odd, $i \neq 1, K$, delete the subfiles $\mathcal{W}_5(i) = (W^2_{n,\{i-1,i,i+1\}\cup\mathcal{X}}), \mathcal{W}_6(i) = (W^2_{n,\{i-1,i\}\cup\mathcal{Y}}), \mathcal{W}_7(i) = (W^2_{n,\{i,i+1\}\cup\mathcal{Y}})$, then place the coded subfiles

$$\mathcal{W}_6(i) \oplus \mathcal{W}_7(i) = (W^2_{n,\{i-1,i\}\cup\mathcal{Y}} \oplus W^2_{n,\{i,i+1\}\cup\mathcal{Y}}),$$

  where $\mathcal{X} \in \binom{[K]\backslash\{i-1,i,i+1\}}{t-3}, \mathcal{Y} \in \binom{[K]\backslash\{i-1,i,i+1\}}{t-2}$. For user 1, delete the subfiles $W^2_{n,\{1,2\}\cup\mathcal{Q}}$, where $\mathcal{Q} \in \binom{[K]\backslash\{1,2\}}{t-2}$. If K is odd, then for user K, delete the subfiles $W^2_{n,\{K-1,K\}\cup\mathcal{P}}$, where $\mathcal{P} \in \binom{[K]\backslash\{K-1,K\}}{t-2}$.

Step 2 is a repetition of Step 1 for each user numbered odd and W_n^2. Therefore after Step 2, the contents(W_n^2) that each user can access remain unchanged for each user.

$\binom{K-2}{t-2}/2$ subfiles are deleted for each user, so we have (12) and (13) proved.

Table 2. Placement under Scheme A in Example 3

User 1	User 2	User 3	User 4	User 5	User 6
$W_{n,13}$	$W_{n,24}$	$W_{n,35}$	$W_{n,46}$	$W_{n,15}$	$W_{n,26}$
$W_{n,14}$	$W_{n,25}$	$W_{n,36}$	$W_{n,14}$	$W_{n,25}$	$W_{n,36}$
$W_{n,15} \oplus W_{n,12}$	$W_{n,26} \oplus W_{n,23}$	$W_{n,13} \oplus W_{n,34}$	$W_{n,24} \oplus W_{n,45}$	$W_{n,35} \oplus W_{n,56}$	$W_{n,46} \oplus W_{n,16}$
$W_{n,16}$	$W_{n,12}$	$W_{n,23}$	$W_{n,34}$	$W_{n,45}$	$W_{n,56}$

Table 3. Placement under Scheme B in Example 3

User 1	User 2	User 3	User 4	User 5	User 6
$W^1_{n,12}$		$W^1_{n,34}$		$W^1_{n,56}$	$W^1_{n,16}$
$W^1_{n,13}$	$W^1_{n,24}$	$W^1_{n,35}$	$W^1_{n,46}$	$W^1_{n,15}$	$W^1_{n,26}$
$W^1_{n,14}$	$W^1_{n,25}$	$W^1_{n,36}$	$W^1_{n,14}$	$W^1_{n,25}$	$W^1_{n,36}$
$W^1_{n,15}$	$W^1_{n,26}$	$W^1_{n,13}$	$W^1_{n,24}$	$W^1_{n,35}$	$W^1_{n,46}$
$W^1_{n,16}$	$W^1_{n,12} \oplus W^1_{n,23}$	$W^1_{n,23}$	$W^1_{n,34} \oplus W^1_{n,45}$	$W^1_{n,45}$	
	$W^2_{n,23}$		$W^2_{n,45}$		$W^2_{n,16}$
$W^2_{n,13}$	$W^2_{n,24}$	$W^2_{n,35}$	$W^2_{n,46}$	$W^2_{n,15}$	$W^2_{n,26}$
$W^2_{n,14}$	$W^2_{n,25}$	$W^2_{n,36}$	$W^2_{n,14}$	$W^2_{n,25}$	$W^2_{n,36}$
$W^2_{n,15}$	$W^2_{n,26}$	$W^2_{n,13}$	$W^2_{n,24}$	$W^2_{n,35}$	$W^2_{n,46}$
$W^2_{n,16}$	$W^2_{n,12}$	$W^2_{n,23} \oplus W^2_{n,34}$	$W^2_{n,34}$	$W^2_{n,45} \oplus W^2_{n,56}$	$W^2_{n,56}$

Table 2 and Table 3 shows the placement under scheme A and B in Example 3, respectively. In scheme A, we should send $W_{d_1,26} \oplus W_{d_6,16}$ to user 1 and 6 in delivery phase.

Finally we choose the scheme with lower rate, i.e.

$$R_{final}(M) = \min\{R_A(M), R_B(M)\}.$$

5 Performance Analysis

In this section, we show the performance of our proposed scheme.

As mentioned earlier, our proposed system model can be considered a MACC system with L values between 2 and 3. The key differentiation from existing MACC systems is the asymmetrical network topology.

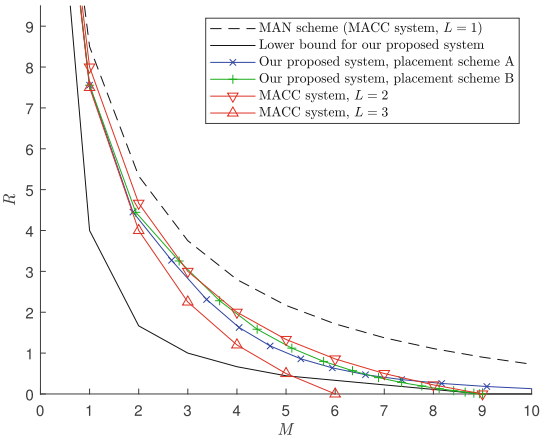

Fig. 4. Rate R on the shared link in the delivery phase as a function of memory size M for $N = K = 18$. Scheme A performs better at lower M. Scheme B performs better at bigger M and reaches zero at $t = N/2 = 9$.

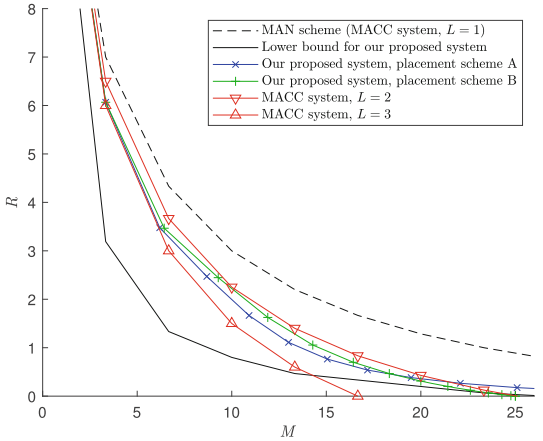

Fig. 5. Rate R on the shared link in the delivery phase as a function of memory size M for $N = 50, K = 15$. Scheme A performs better at lower M. Scheme B performs better at bigger M and reaches zero at $t = N/2 = 25$.

As shown in Fig. 4, the delivery rate of our scheme (system) is between the rate of MACC system with $L = 2$ and $L = 3$ in [3], which matches the system settings: most of users have the access to three caches as the MACC system with $L = 3$, but the bottleneck of the system is the edge users, who only have the access to two caches as the MACC system with $L = 2$. The placement scheme A performs better at small M. With the increase of M, the rate of placement scheme B decreases faster than scheme A and achieves 0 rate at $M = N/2$. Fig. 5 shows the case of $N = 50, K = 15$.

The lower bound is

$$R(M) \geq \max_{s\in\{1,\ldots,\min\{N,K\}\}} \left(s - \frac{\min(s+1,K)}{\lfloor N/s \rfloor} M\right). \tag{14}$$

The proof of the lower bound is shown in Appendix B

6 Conclusion

In this paper, we proposed a novel model of coded caching inspired by the V2V scene, allowing communication between cooperative users. Based on this model, we designed a scheme with coded prefetching to reduce the redundancy in the cache contents and Greedy Constrained Coloring to exploit the multicasting opportunities. We showed its performance, comparing to the Multi-access Coded Caching system.

Acknowledgement. This work was supported by the National Science Foundation of China (NSFC) with grant no. 62271514 and the Science, Technology and Innovation Commission of Shenzhen Municipality with grant no. JCYJ20210324120002007, and ZDSYS20210623091807023.

A Proof of Placement Scheme A

A.1 Proof of Step 1

Here we show the proof of the fact that after step 1, the contents that each user can access remain unchanged.

Table 4. Subfiles placed in two users after Step 1 in Scheme A.

User i	User $i+1$
$\mathcal{W}_1(i) \oplus \mathcal{W}_2(i)$	$W_1(i+1) \oplus \mathcal{W}_2(i+1)$
$\mathcal{W}_n 2(i+1)$	$\mathcal{W}_1(i)$

For two users $i, i+1$, shown in Table 4, $\mathcal{W}_1(i)$ is cached in the cache of user $i+1$, $\mathcal{W}_2(i+1)$ is cached in the cache of user i. Table 5 shows which users will be and will not be included in the labels of the subfiles in these sets. Due to the fact

$$\mathcal{W}_1(i) \cap \mathcal{W}_1(i+1) = \emptyset, \tag{15a}$$
$$\mathcal{W}_1(i) \cap \mathcal{W}_2(i+1) = \emptyset, \tag{15b}$$
$$\mathcal{W}_2(i) \cap \mathcal{W}_2(i+1) = \emptyset, \tag{15c}$$

both of them can recover $\mathcal{W}_2(i)$ and $\mathcal{W}_1(i+1)$.

Table 5. Labels in the subfiles sets in Step 1, Scheme A.

set	contains	not contains
$\mathcal{W}_1(i)$	$i-2, i, i+1$	i
$\mathcal{W}_2(i)$	$i-2, i-1, i$	$i+1$
$\mathcal{W}_1(i+1)$	$i-1, i, i+1$	i
$\mathcal{W}_2(i+1)$	$i-1, i, i+1$	$i+2$

The label of subfile in $W_1(i)$ must contains i, and the label of subfile in $W_1(i+1)$ must not contains i, therefore we have (15a). The label of subfile in $W_2(i+1)$ must contains $i-1$, and the label of subfile in $W_1(i)$ must not contains $i-1$, therefore we have (15b). The label of subfile in $W_2(i+1)$ must contains $i+1$, and the label of subfile in $W_2(i)$ must not contains $i+1$, therefore we have (15c).

Therefore after step 1, the contents that each user can access remain unchanged.

A.2 Proof of Step 2

Table 6. Subfiles placed in three users after Step 2 in Scheme A.

User $i-1$	**User i**	**User $i+1$**
$\mathcal{W}_3(i-1) \oplus \mathcal{W}_4(i-1)$	$\mathcal{W}_3(i) \oplus \mathcal{W}_4(i)$	$\mathcal{W}_3(i+1) \oplus \mathcal{W}_4(i+1)$
part of $\mathcal{W}_4(i+1)$	$\mathcal{W}_3(i-1)$	$\mathcal{W}_3(i)$

For user $i, i \neq 1, K$, shown in Table 6 it can access $\mathcal{W}_3(i)$ in the cache of user $i+1$, then recover $\mathcal{W}_4(i)$ from the coded subfiles. It can access $\mathcal{W}_3(i-1)$ in its own cache, then recover $\mathcal{W}_4(i-1)$ from the coded subfiles in the cache of user $i-1$. Then it can access $\mathcal{W}_4(i+1)$ in the cache of user $i-1$, and recover

Table 7. Labels in the subfiles sets in Step 2, Scheme A.

set	contains	not contains
$\mathcal{W}_3(i-1)$	$i-i, i$	$i-3, i-2$
$\mathcal{W}_3(i)$	$i, i+1$	$i-2, i-1$
$\mathcal{W}_3(i+1)$	$i+1, i+2$	$i-1, i$
$\mathcal{W}_4(i-1)$	$i-3, i-1$	$i-2, i$
$\mathcal{W}_4(i)$	$i-2, i$	$i-1, i+1$
$\mathcal{W}_4(i+1)$	$i-1, i+1$	$i, i+2$

$\mathcal{W}_3(i+1)$ from the coded subfiles in the cache of user $i+1$. Table 7 shows which users will be and will not be included in the labels of the subfiles in these sets.

We should prove the fact

$$\mathcal{W}_3(i) \cap \mathcal{W}_3(i+1) = \emptyset, \tag{16a}$$
$$\mathcal{W}_3(i) \cap \mathcal{W}_4(i+1) = \emptyset, \tag{16b}$$
$$\mathcal{W}_4(i+1) \cap \mathcal{W}_3(i-1) = \emptyset, \tag{16c}$$

The label of subfile in $W_3(i)$ must contains i, and the label of subfile in $W_3(i+1)$ must not contains i, therefore we have (16a). The label of subfile in $W_3(i)$ must contains i, and the label of subfile in $W_4(i+1)$ must not contains i, therefore we have (16b). The label of subfile in $W_3(i-1)$ must contains i, and the label of subfile in $W_4(i+1)$ must not contains i, therefore we have (16c). Besides, although $\mathcal{W}_4(i+1) \cap \mathcal{W}_4(i-1) \neq \emptyset$, user i can first recover $\mathcal{W}_4(i-1)$ then it can access all subfiles in $\mathcal{W}_4(i+1)$.

For user 1, it does not have the access of $\mathcal{W}_4(2)$ because user 0 does not exist. For user K, it does not have the access of $\mathcal{W}_3(K)$ because user $K+1$ does not exist. We send

$$X = \mathcal{W}_4(2)_{d_1} \oplus \mathcal{W}_3(K)_{d_K}$$

to user 1 and K in delivery phase, with some reordering of the subfiles.

User 1 can access $\mathcal{W}_3(K)_{d_1}$ in its own cache so it can solve $\mathcal{W}_4(2)_{d_1}$. User K can access $\mathcal{W}_4(2)_{d_K}$ except the subfiles in

$$\mathcal{W}_8 = \mathcal{W}_4(2)_{d_K} \cap \mathcal{W}_4(K)_{d_K} = W_{d_K,\{2,K-2,K\}\cup\mathcal{X}}$$

where $\mathcal{X} \in \binom{[K]\setminus\{K-2,K-1,K,1,2,3\}}{t-3}$.

Let

$$\mathcal{W}_9 = W_{d_K,\{1,2,K\}\cup\mathcal{X}}$$

, note that $\mathcal{W}_9 \subset \mathcal{W}_3(K)_{d_K}$. Here $\mathcal{W}_9$ contains the subfiles coded with the subfiles in $\mathcal{W}_8$ stored in the cache of user K. When constructing message X, we should aviod the subfiles in $\mathcal{W}_8 \oplus \mathcal{W}_9$ appear in X by reordering the subfiles in $\mathcal{W}_4(2)_{d_1}$ and $\mathcal{W}_3(K)_{d_K}$, to make sure that user K can solve $\mathcal{W}_9$, then solve $\mathcal{W}_8$.

We should prove that the number of subfiles in $\mathcal{W}_9$ does not exceed half of the number of subfiles in $\mathcal{W}_3(K)_{d_K}$, i.e.

$$\binom{K-6}{t-3} \leq \binom{K-4}{t-2}/2 \tag{17}$$

where $0 < t < K$, t and K are integer. Expand (17) we have

$$K^2 - 5K + 2t^2 - 2Kt + 12 \geq 0 \tag{18}$$

Let $g(K,t)$ be the left-hand side of (18), $g(K,t)$ reaches its minimum value at $K = 5, t = 2.5$. It can be easily proved that $g(K,t) \geq 0$ when $4 \leq K \leq 6, 2 \leq t \leq 4$. Therefore we have (17) proved.

A.3 Proof of the Independence of Step 1 and Step 2

In the last part of the proof, we show that we modify the different subfiles in Step 1 and Step 2. We proof the fact

$$\mathcal{W}_1(i) \cap \mathcal{W}_3(i) = \emptyset, \tag{19a}$$
$$\mathcal{W}_1(i) \cap \mathcal{W}_4(i) = \emptyset, \tag{19b}$$
$$\mathcal{W}_2(i) \cap \mathcal{W}_3(i) = \emptyset, \tag{19c}$$
$$\mathcal{W}_2(i) \cap \mathcal{W}_4(i) = \emptyset, \tag{19d}$$

The users will be and will not be included in the labels of the subfiles in these sets are shown in Table 5 and Table 7. The label of subfile in $W_1(i)$ must contains $i-2$, and the label of subfile in $W_3(i)$ must not contains $i-2$, therefore we have (19a). The label of subfile in $W_1(i)$ must contains $i+1$, and the label of subfile in $W_4(i)$ must not contains $i+1$, therefore we have (19b). The label of subfile in $W_2(i)$ must contains $i-2$, and the label of subfile in $W_3(i)$ must not contains $i-2$, therefore we have (19c). The label of subfile in $W_2(i)$ must contains $i-1$, and the label of subfile in $W_4(i)$ must not contains $i-1$, therefore we have (19d).

B Proof of the Lower Bound

Let $s \in \{1, \ldots, \min\{N, K\}\}$ and select arbitrary s users, called $\mathcal{U}_s$. Let $\mathcal{L}(\mathcal{U}_s)$ be the caches that users in $\mathcal{U}_s$ can access and $L(\mathcal{U}_s) = |\mathcal{L}(\mathcal{U}_s)|$, i.e. the number of the caches that users in $\mathcal{U}_s$ can access, we have

$$L(\mathcal{U}_s) \geq s + 1. \tag{20}$$

Note that $L(\mathcal{U}_s) = s + 1$ only when $\mathcal{U}_s = [s]$ or $\mathcal{U}_s = [K] \backslash [K - s]$. For any other case, $L(\mathcal{U}_s) > s + 1$.

For the users in $\mathcal{U}_s$, there exist a user demand and a corresponding input to the shared link, say X_1, such that X_1 and $\mathcal{L}(\mathcal{U}_s)$ determine the file $W_1, \ldots, W_s$. Similarly, there exist a user demand and a corresponding input to the shared link, say X_2, such that X_2 and $\mathcal{L}(\mathcal{U}_s)$ determine the file $W_{s+1}, \ldots, W_{2s}$. Continue in the same manner selecting $X_3, \ldots, X_{\lfloor N/s \rfloor}$, we have $X_1, \ldots, X_{\lfloor N/s \rfloor}$ and $\mathcal{L}(\mathcal{U}_s)$ determine the files $W_1, \ldots, W_{s\lfloor N/s \rfloor}$.

Consider the cut separating $X_1, \ldots, X_{\lfloor N/s \rfloor}$ and $\mathcal{L}(\mathcal{U}_s)$ from the corresponding users. By the cut-set bound,

$$\lfloor N/s \rfloor R(M) + L(\mathcal{U}_s)M \geq s\lfloor N/s \rfloor.$$

Solving for R and optimizing over all possible s and $\mathcal{U}_s$, we have (14) proofed.

References

1. Asadi, B., Ong, L.: Centralized caching with shared caches in heterogeneous cellular networks. In: 2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), pp. 1–5 (2019). https://doi.org/10.1109/SPAWC.2019.8815401
2. Dutta, M., Thomas, A.: Decentralized coded caching for shared caches. IEEE Commun. Lett. **25**(5), 1458–1462 (2021). https://doi.org/10.1109/LCOMM.2021.3052237
3. Hachem, J., Karamchandani, N., Diggavi, S.N.: Coded caching for multi-level popularity and access. IEEE Trans. Inf. Theory **63**(5), 3108–3141 (2017). https://doi.org/10.1109/TIT.2017.2664817
4. Ibrahim, A.M., Zewail, A.A., Yener, A.: Device-to-device coded-caching with distinct cache sizes. IEEE Trans. Commun. **68**(5), 2748–2762 (2020). https://doi.org/10.1109/TCOMM.2020.2970950
5. Ji, M., Caire, G., Molisch, A.F.: Fundamental limits of caching in wireless D2D networks. IEEE Trans. Inf. Theory **62**(2), 849–869 (2016). https://doi.org/10.1109/TIT.2015.2504556
6. Ji, M., Tulino, A.M., Llorca, J., Caire, G.: Order-optimal rate of caching and coded multicasting with random demands. IEEE Trans. Inf. Theory **63**(6), 3923–3949 (2017). https://doi.org/10.1109/TIT.2017.2695611
7. Maddah-Ali, M.A., Niesen, U.: Fundamental limits of caching. IEEE Trans. Inf. Theory **60**(5), 2856–2867 (2014). https://doi.org/10.1109/TIT.2014.2306938
8. Maddah-Ali, M.A., Niesen, U.: Decentralized coded caching attains order-optimal memory-rate tradeoff. IEEE/ACM Trans. Networking **23**(4), 1029–1040 (2015). https://doi.org/10.1109/TNET.2014.2317316
9. Namboodiri, K.K.K., Rajan, B.S.: Improved lower bounds for multi-access coded caching. IEEE Trans. Commun. **70**(7), 4454–4468 (2022). https://doi.org/10.1109/TCOMM.2022.3174890
10. Niesen, U., Maddah-Ali, M.A.: Coded caching with nonuniform demands. IEEE Trans. Inf. Theory **63**(2), 1146–1158 (2017). https://doi.org/10.1109/TIT.2016.2639522
11. Parrinello, E., Ünsal, A., Elia, P.: Fundamental limits of coded caching with multiple antennas, shared caches and uncoded prefetching. IEEE Trans. Inf. Theory **66**(4), 2252–2268 (2020). https://doi.org/10.1109/TIT.2019.2955384
12. Reddy, K.S., Karamchandani, N.: Rate-memory trade-off for multi-access coded caching with uncoded placement. IEEE Trans. Commun. **68**(6), 3261–3274 (2020). https://doi.org/10.1109/TCOMM.2020.2980817
13. Tang, Y., Heydaryan, F., Luo, J.: Distributed coding in a multiple access environment. Found. Trends. Netw. **12**(4), 260–412 (2018). https://doi.org/10.1561/1300000063
14. Tian, C.: Symmetry, demand types and outer bounds in caching systems. In: 2016 IEEE International Symposium on Information Theory (ISIT), pp. 825–829 (2016). https://doi.org/10.1109/ISIT.2016.7541414