Coded Caching with Multiple File Requests

Yi-Peng Wei Sennur Ulukus
Department of Electrical and Computer Engineering
University of Maryland College Park, MD 20742
ypwei@umd.edu *ulukus@umd.edu*

***Abstract*—We study a two-phase caching network consisting of one server with N files connected to K users through an error-free shared link. Each user has a cache memory which can store M files in the placement phase. In the delivery phase, each user requests L files, and the server transmits the messages accordingly. Using the message sent by the server combined with the cache memory, each user reconstructs the L files they requested. In this work, we focus on the case $L > 1$, i.e., the case of multiple file requests. We adopt the symmetric batch caching scheme and propose a general delivery scheme. To prove the optimality of the proposed general delivery scheme, we apply two converse techniques. The first converse technique is for general coding schemes and is obtained through virtual user construction. The second converse technique is for vector linear coding schemes and is obtained using an interference alignment point of view. With these two converse techniques, we characterize either the unconstrained optimal coding rate, or optimum linear coding rate, with symmetric batch caching for certain cases.**

I. Introduction

Consider a two-phase caching network [1] consisting of one server with N files connected to K users through an error-free shared link. Each user has their local cache memory which can store M files. The two phases are the placement phase and the delivery phase. In the placement phase, the network traffic load is low. Each user can access the whole N files in the server and fill their cache memory in advance. In the delivery phase, the network traffic load is high. Each user requests L files from the server, and the server delivers messages through the error-free shared link to K users. The request of each user is unknown a priori in the placement phase. Each user reconstructs the L files they requested by the messages sent from the server and the side information stored in their cache memory. The objective is to minimize the traffic load in the delivery phase due to the high traffic load in this phase.

The two-phase caching network is first studied in [1], with the assumption that in the delivery phase, each user requests one file, i.e., $L = 1$. Reference [1] proposes *symmetric batch caching* for the placement phase. Combined with the coded multicasting in the delivery phase, reference [1] shows that global caching gain can be obtained. Using a cut-set bound analysis, order optimality of rate-memory trade off is shown for the worst-case file requests in [1]. *Independent and identical caching* is proposed in [2] to account for the decentralized nature of practical caching networks. Both symmetric batch caching and independent and identical caching are uncoded cache placement schemes [3], [4], i.e., each user stores a subset of the bits of the original files. For uncoded cache placement, with $N \geq K$, by using index coding converse bound [5], reference [3] shows the optimality of rate-memory trade off for the worst-case file requests. In addition, reference [4] shows the optimality for arbitrary N, K and M not only for the worst-case file requests but also for the average case. For coded placement, order optimality results can be found in [6] and references therein.

In the delivery phase, each user can request more than one file, i.e., $L > 1$. This case is first studied in [7] which adopts symmetric batch caching as in [1]. For the delivery phase, [7] treats each different file request as a different index coding problem, and generalizes the achievability scheme for multiple unicast index coding in [8] to group casting index coding. Reference [7] shows the order optimality for the worst-case file request with a multiplicative constant 18 based on a cut-set bound analysis. Then, reference [9] shows the order optimality for the worst-case file request with multiplicative constant 11 by improving the converse bound through Han's inequality. Reference [9] adopts symmetric batch caching as in [1] and applies the delivery scheme in [1] L times.

In this work, we also adopt symmetric batch caching as in [1], and focus on exact optimality as opposed to order optimality. We propose a general delivery scheme for multiple file requests. To show the optimality of the delivery scheme, we use two converse techniques. The first technique is for general coding schemes and is inspired by the converse in [4]. The second technique is for vector linear coding schemes using an interference alignment point of view and is inspired by [10], [11]. With these two converse techniques, we determine either unconstrained optimal coding rate, or optimal linear coding rate with symmetric batch caching for certain cases. If LK different files are requested, we characterize the optimal coding rate. For $L = 2$ and $K = 3, 4$, when each subfile is cached only at one user (i.e., $t = 1$), we characterize either the unconstrained optimal coding rate, or the optimal linear coding rate.

II. System Model and Problem Setting

We consider a caching network (see Fig. 1) consisting of one server and K users. The server connects to the K users through an error-free shared link. The server has N files denoted by $W_1, W_2, \ldots, W_N$. Each file is of size F bits. Each user has a local cache memory Z_k of size MF bits for some real number $M \in [0, N]$. There are two phases in this network, a placement

This work was supported by NSF Grants CNS 13-14733, CCF 14-22111, and CNS 15-26608.

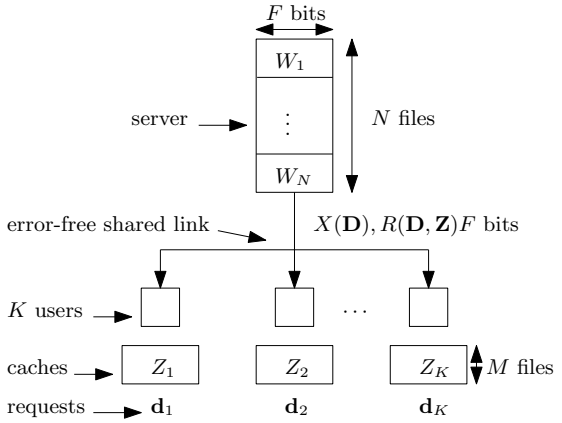

Fig. 1. Caching network.

phase and a delivery phase. In the placement phase, user k can access all the N files and fill its cache memory Z_k. Therefore, $Z_k = \phi_k(W_1, W_2, \ldots, W_N)$, where $\phi_k : \mathbb{F}_2^{NF} \to \mathbb{F}_2^{MF}$. In the delivery phase, each user requests L files out of the N files. Let us denote each user's request by an $L \times 1$ vector $\mathbf{d}_k = (d_{k,1}, \ldots, d_{k,L})^{\mathrm{T}}$, where $d_{k,i}$ is the index of the file, i.e., $d_{k,i} \in \{1, 2, \ldots, N\},\ 1 \le i \le L$. A request matrix $\mathbf{D}$ of size $L \times K$ is formed accordingly. The server outputs $X(\mathbf{D})$ of size RF bits through the error-free shared link to the K users, where R refers to the load of the network in the delivery phase. A rate R is said to be achievable if each user k can decode the L files it requested by utilizing $X(\mathbf{D})$ and Z_k.

Note that in the placement phase, which L files will be requested by each user is unknown in advance. Therefore, the cache memory Z_k is determined before knowing $\mathbf{D}$. In addition, we focus on the uncoded cache placement as in [4], which means that each user k chooses MF bits out of NF bits to fill its cache memory Z_k. An example of coded cache placement is provided in [1, Appendix]. We denote the minimum achievable rate in the delivery phase by $R^*(\mathbf{D}, \mathbf{Z})$, where $\mathbf{Z} = (Z_1, \ldots, Z_K)$. An average rate for a given cache placement $\mathbf{Z}$ is defined as $R^*(\mathbf{Z}) = \mathbb{E}_{\mathbf{D}}[R^*(\mathbf{D}, \mathbf{Z})]$, by assuming that each user chooses the L files equally likely from the N files. The minimum rate is defined as $R^* = \min_{\mathbf{Z}} R^*(\mathbf{Z})$.

If $L = 1$, reference [4] shows that *symmetric batch caching* originally proposed in [1] attains the minimum rate R^*. We summarize the symmetric batch caching here. Given each user has cache memory size $M = \frac{tN}{K}$, where $t \in \{1, \ldots, K\}$, for each file W_i, we partition the file into $\binom{K}{t}$ non-overlapping and equal-size subfiles, and denote $[K] = \{1, 2, \ldots, K\}$ and $\mathcal{T} = \{T : T \subset [K], |T| = t\}$, where $|\cdot|$ means the cardinality of a set. Note $|\mathcal{T}| = \binom{K}{t}$. We label the subfiles of W_i as $W_{i,T}$, where $T \in \mathcal{T}$. Equivalently, $W_i = \cup_{T \in \mathcal{T}} W_{i,T}$ and $W_{i,T} \cap W_{i,T'} = \emptyset$ if $T \neq T'$. In the placement phase, user k places the subfile $W_{i,T}$ into the cache memory Z_k if $k \in T$. Thus, user k gets $\binom{K-1}{t-1}$ subfiles of each file W_i. We denote the symmetric batch caching with parameter t as $\mathbf{Z}^t_{\mathrm{sym}}$.

In this work, we adopt the symmetric batch caching, $\mathbf{Z}^t_{\mathrm{sym}}$, and study $R^*(\mathbf{D}, \mathbf{Z}^t_{\mathrm{sym}})$. Also, we denote $R^*_l(\mathbf{D}, \mathbf{Z}^t_{\mathrm{sym}})$ as the minimum achievable rate confined to vector linear coding schemes. We propose a general delivery scheme. We characterize $R^*(\mathbf{D}, \mathbf{Z}^t_{\mathrm{sym}})$ if LK distinct files are requested by the users. For $L = 2$, $t = 1$, $K = 3, 4$, we characterize $R^*(\mathbf{D}, \mathbf{Z}^t_{\mathrm{sym}})$ for certain request matrices $\mathbf{D}$, while we characterize $R^*_l(\mathbf{D}, \mathbf{Z}^t_{\mathrm{sym}})$ for all other request matrices.

III. Proposed Delivery Scheme

To illustrate the delivery scheme, we introduce the following vector space representation. In the placement phase, we partition each file into $\binom{K}{t}$ subfiles. We view each subfile as a vector, and regard each subfile as the basis of the vector space. Since the subfiles are linearly independent, with all the subfiles, we have a $N\binom{K}{t}$-dimensional vector space. Note that in the delivery phase, we do not further sub-packetize the subfiles. Therefore, our delivery scheme can also be viewed as scalar linear coding. We consider the vector space over $\mathbb{F}_2$.

Let $\mathcal{S} = \{S : S \subset [K], |S| = t + 1\}$. For every $S \in \mathcal{S}$ and $1 \le l \le L$, a candidate delivery message is as follows:

$$Y_{S,l} = \bigoplus_{s \in S} W_{d_{s,l}, S\setminus\{s\}}. \tag{1}$$

Here, $d_{s,l}$ identifies the index of the lth file user s requests. Since $|S \setminus \{s\}| = t$, it identifies the subfile $W_{d_{s,l}, S\setminus\{s\}}$ owned by these t users. Note that among the $t+1$ terms on the right hand side of (1) only one term is unknown to each user s. By sending the candidate delivery message $Y_{S,l}$, each user s in S can decode a subfile of $W_{d_{s,l}}$ they requested.

For $L = 1$, the delivery scheme proposed in [1] is to go over all the sets in $\mathcal{S}$ and send out all the candidate delivery message given in (1). This results in $\binom{K}{t+1}$ transmissions. However, reference [4] shows that going over all the sets in $\mathcal{S}$ is unnecessary. Instead, [4] goes over $\mathcal{S}' = \{S : S \subset [K], S \cap \mathcal{U} \neq \emptyset, |S| = t + 1\}$, where $\mathcal{U}$ corresponds to the leaders defined in [4]. In [4], the number of transmissions is reduced to $\binom{K}{t+1} - \binom{K-|\mathcal{U}|}{t+1}$.

For $L > 1$, we know that sending out all the candidate delivery messages by going over all the sets in $\mathcal{S}$ and $1 \le l \le L$ is sufficient to satisfy all the requests. To reduce the traffic load, if the candidate delivery message can be obtained through a linear combination of the sent messages, then the server does not need to send this candidate delivery message. Since each candidate delivery message given in (1) has a vector representation, the necessary delivery messages consist of the messages in the maximal linearly independent subset of the candidate delivery messages formed by going over all the sets in $\mathcal{S}$ and $1 \le l \le L$.

We use an example to illustrate the proposed delivery scheme. Let $N = 4$, $K = 4$, $M = 1$, and $L = 2$. Then, $t = 1$. To simplify the notation, let A, B, C and D denote the four files. By applying symmetric batch caching, $\mathbf{Z}^1_{\mathrm{sym}}$, we have

$$Z_k = (A_k, B_k, C_k, D_k), \tag{2}$$

where $k = 1, 2, 3, 4$. Suppose the request matrix is as follows

$$\mathbf{D} = \begin{pmatrix} A & A & A & B \\ B & C & C & C \end{pmatrix}, \tag{3}$$

which means user 1 requests files A and B, user 2 requests files A and C and so on. If the server sends out all the candidate delivery messages as [9, Lemma 1], then this results in $R = 3$ normalized traffic load, i.e., all the 12 candidate delivery messages are sent out. If the server applies the delivery scheme in [4] twice, then it results in $R = \frac{5}{2}$ normalized traffic load, i.e., only 10 candidate delivery messages are sent out. By applying the delivery scheme proposed here, only 9 candidate delivery messages are sent out resulting in $R = \frac{9}{4}$ normalized traffic load.

Specifically, by treating each subfile as the basis of the vector space, we have

$$A_w \to \mathbf{e}_w, \qquad w = 1,2,3,4 \tag{4}$$
$$B_x \to \mathbf{e}_{x+4}, \qquad x = 1,2,3,4 \tag{5}$$
$$C_y \to \mathbf{e}_{y+8}, \qquad y = 1,2,3,4 \tag{6}$$
$$D_z \to \mathbf{e}_{z+12}, \qquad z = 1,2,3,4 \tag{7}$$

where $\mathbf{e}_i, i = 1,2,\ldots,16$ are the standard bases of $\mathbb{F}_2^{16}$. Since file D does not appear in the request matrix given in (3), in the following we only consider the subspace spanned by $\{\mathbf{e}_i, i = 1,2,\ldots,12\}$. We represent the candidate delivery messages as follows:

$$Y_{\{1,2\},1} = A_2 \oplus A_1 \to (1,1,0,0,0,0,0,0,0,0,0,0), \tag{8}$$
$$Y_{\{1,3\},1} = A_3 \oplus A_1 \to (1,0,1,0,0,0,0,0,0,0,0,0), \tag{9}$$
$$Y_{\{1,4\},1} = A_4 \oplus B_1 \to (0,0,0,1,1,0,0,0,0,0,0,0), \tag{10}$$
$$Y_{\{2,3\},1} = A_3 \oplus A_2 \to (0,1,1,0,0,0,0,0,0,0,0,0), \tag{11}$$
$$Y_{\{2,4\},1} = A_4 \oplus B_2 \to (0,0,0,1,0,1,0,0,0,0,0,0), \tag{12}$$
$$Y_{\{3,4\},1} = A_4 \oplus B_3 \to (0,0,0,1,0,0,1,0,0,0,0,0), \tag{13}$$
$$Y_{\{1,2\},2} = B_2 \oplus C_1 \to (0,0,0,0,0,1,0,0,1,0,0,0), \tag{14}$$
$$Y_{\{1,3\},2} = B_3 \oplus C_1 \to (0,0,0,0,0,0,1,0,1,0,0,0), \tag{15}$$
$$Y_{\{1,4\},2} = B_4 \oplus C_1 \to (0,0,0,0,0,0,0,1,1,0,0,0), \tag{16}$$
$$Y_{\{2,3\},2} = C_3 \oplus C_2 \to (0,0,0,0,0,0,0,0,0,1,1,0), \tag{17}$$
$$Y_{\{2,4\},2} = C_4 \oplus C_2 \to (0,0,0,0,0,0,0,0,0,1,0,1), \tag{18}$$
$$Y_{\{3,4\},2} = C_4 \oplus C_3 \to (0,0,0,0,0,0,0,0,0,0,1,1). \tag{19}$$

Here, (8)-(13) are the candidate delivery messages for the requests of files (A,A,A,B), and (14)-(19) are the candidate delivery messages for the requests of files (B,C,C,C). Reference [9] sends out all the 12 candidate delivery messages in (8)-(19) achieving a normalized traffic load $R = 3$. In [4], if we choose users $\{1,4\}$ as the leaders for the requests of files (A,A,A,B), and choose users $\{1,2\}$ as the leaders for the requests of files (B,C,C,C), then we do not need to send out candidate messages $Y_{\{2,3\},1}$ and $Y_{\{3,4\},2}$, since

$$Y_{\{2,3\},1} = A_3 \oplus A_2 \tag{20}$$
$$= (A_3 \oplus A_1) \oplus (A_2 \oplus A_1) \tag{21}$$
$$= Y_{\{1,3\},1} \oplus Y_{\{1,2\},1}, \tag{22}$$
$$Y_{\{3,4\},2} = C_4 \oplus C_3 \tag{23}$$
$$= (C_4 \oplus C_2) \oplus (C_3 \oplus C_2) \tag{24}$$
$$= Y_{\{2,4\},2} \oplus Y_{\{2,3\},2}. \tag{25}$$

That is, if the server applies the delivery scheme in [4] twice, then the normalized traffic load is $R = \frac{5}{2}$.

For our general delivery scheme, in addition to $Y_{\{2,3\},1}$ and $Y_{\{3,4\},2}$, we also do not send out $Y_{\{1,3\},2}$, since

$$Y_{\{1,3\},2} = B_3 \oplus C_1 \tag{26}$$
$$= (A_4 \oplus B_3) \oplus (B_2 \oplus C_1) \oplus (A_4 \oplus B_2) \tag{27}$$
$$= Y_{\{3,4\},1} \oplus Y_{\{1,2\},2} \oplus Y_{\{2,4\},1}. \tag{28}$$

Therefore, sending out the other 9 candidate delivery messages in (8)-(19) is sufficient. Our proposed delivery scheme achieves normalized traffic load of $R = \frac{9}{4}$.

By using our proposed delivery scheme, the following rate

$$R(\mathbf{D}, \mathbf{Z}_{\text{sym}}^t) = \frac{\text{rank}\,(\mathbf{Y})}{\binom{K}{t}} \tag{29}$$

is achievable in general, where $\mathbf{Y}$ is an $L\binom{K}{t+1} \times n$ matrix, with $n = N\binom{K}{t}$, where each row vector of matrix $\mathbf{Y}$ consists of vector representation of candidate delivery messages given in (1) by treating each subfile as a standard basis of $\mathbb{F}_2^n$.

IV. Converse

We consider two converse techniques in this work. The first technique is for general coding schemes and is inspired by [4]. The second technique is for vector linear coding schemes using an interference alignment point of view and is inspired by [10], [11]. We use two examples to illustrate these two converse techniques.

A. Converse for General Coding Schemes

This converse builds on the virtual user construction, which is a generalization from [4, Appendix A]. The virtual user construction is related to the request matrix $\mathbf{D}$. It starts from the set consisting of K users, i.e., at the beginning let $\mathcal{U} = [K]$. We remove user $k \in \mathcal{U}$ if user k requests the same set of files as an other user $k' \in \mathcal{U}$. Now, we construct the virtual user according to $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$. Initially, the cache memory of the virtual user, denoted by Z_v, is empty. We fill Z_v with the cache memory of user u_1, i.e., Z_{u_1}. Then, we fill Z_v with the cache memory of user u_2, but if the subfiles cached in Z_{u_2} are the files requested by previous user, i.e., $\mathbf{d}_{u_1}$, then we do not fill these subfiles into Z_v. We continue this process, i.e., filling Z_v with the cache memory of user u_k, but if the subfiles in Z_{u_k} are the files requested by one of the previous users, i.e., $\mathbf{d}_{u_1}, \ldots, \mathbf{d}_{u_{k-1}}$, then not filling these subfiles into Z_v. Note that the virtual user does not have a memory size constraint. Note also that there are $n!$ ways to build the virtual user.

Let us use an example to show the virtual user construction and the converse bound it provides. Consider a caching network with $N = 4$, $K = 3$, $t = 1$, and the request matrix is

$$\bar{\mathbf{D}} = \begin{pmatrix} A & A & B \\ B & C & D \end{pmatrix}. \tag{30}$$

By applying symmetric batch caching, $\mathbf{Z}_{\text{sym}}^1$, we have

$$Z_k = (A_k, B_k, C_k, D_k), \tag{31}$$

where $k = 1, 2, 3$.

According to the request matrix $\bar{\mathbf{D}}$ given in (30), we have $\mathcal{U} = \{1, 2, 3\}$, since no two users request exactly the same set of files. We fill Z_v according to the order of user 2, user 3 then user 1. Due to user 2, Z_v contains subfiles (A_2, B_2, C_2, D_2). For user 3, since subfiles (A_3, C_3) are requested by previous user, i.e., $\mathbf{d}_2$, we only fill subfiles (B_3, D_3) into Z_v. Finally, for user 1, since subfiles (A_1, B_1, C_1, D_1) are requested by one of the previous users, we do not fill any new subfiles into Z_v. After the construction of the virtual user, we construct the converse bound as follows.

Assume that by receiving $X(\bar{\mathbf{D}})$ each user can reconstruct the files they requested. Since the virtual user has more side information than user 2, by receiving $X(\bar{\mathbf{D}})$ the virtual user can also reconstruct files A and C. This also shows that $X(\bar{\mathbf{D}})$ contains information of subfiles (A_1, A_3, C_1, C_3). After the virtual user obtains files A and C, together with the files in Z_v, the virtual user has more side information than user 3. Therefore, the virtual user can also reconstruct files B and D. This also shows that $X(\bar{\mathbf{D}})$ contains information of subfiles (B_1, D_1). According to this virtual user construction, we know that $X(\bar{\mathbf{D}})$ at least needs to contain $6 \times \frac{F}{3} = 2F$ bits, since information of subfiles $(A_1, A_3, C_1, C_3, B_1, D_1)$ are from $X(\bar{\mathbf{D}})$, and each subfile is of size $\frac{F}{3}$ bits. By applying the delivery scheme proposed in Section III, we note that normalized traffic load $R = 2$ is also achievable. Therefore, we have $R^*(\bar{\mathbf{D}}, \mathbf{Z}^1_{\text{sym}}) = 2$.

Note that if we construct the virtual user according to the order of user 1, user 2 then user 3, then we have subfiles $(A_1, B_1, C_1, C_2, D_1, D_2, D_3)$ in Z_v. From this construction, we can only show that the traffic load must be greater than $\frac{5}{3}F$, since it only implies that $X(\bar{\mathbf{D}})$ contains information of subfiles $(A_2, A_3, B_2, B_3, C_3)$. Therefore, different virtual user constructions may result in a different converse bounds.

We remark that for $L = 1$, the virtual user construction in [4, Appendix A] always results in a tight converse. However, for $L > 1$, the virtual user construction may not always give a tight converse. To see this, consider a different request matrix as follows:

$$\tilde{\mathbf{D}} = \begin{pmatrix} A & A & B \\ B & C & C \end{pmatrix}. \tag{32}$$

In this case, no matter which ordering we use to construct the virtual user, we can only show that the normalized traffic load $R \geq \frac{5}{3}$, while the proposed achievable scheme achieves $R = 2$, leaving a gap between the two.

B. Converse for Vector Linear Coding Schemes

For the request matrix given in (32), if we are confined to vector linear coding scheme, then we have a tighter converse bound. The request matrix given in (32) is equivalent to the following index coding problem [10, Sec. II], [11]. User 1 with side information (A_1, B_1, C_1) requests subfiles (A_2, A_3, B_2, B_3). User 2 with side information (A_2, B_2, C_2) requests subfiles (A_1, A_3, C_1, C_3). User 3 with side information (A_3, B_3, C_3) requests subfiles (B_1, B_2, C_1, C_2). Since we apply the symmetric batch caching, $\mathbf{Z}^t_{\text{sym}}$, each subfile has the same size, here $\frac{F}{3}$ bits. Therefore, the upper bound of the symmetric capacity of the index coding problem can be used to lower bound the traffic load of the caching network [3].

Let $R^{\text{IC}}_{A_2}$ denote the rate to transmit message A_2 in the corresponding index coding problem. For the caching network, A_2 refers to the subfile. For the corresponding index coding problem, A_2 represents the message. The superscript IC refers to the rate for the index coding problem. For user 1, we have

$$\begin{aligned} nR^{\text{IC}}_{A_2} &= H(A_2) && (33)\\ &= I(A_2; S^n, A_1, B_1, C_1) \\ &\quad + H(A_2|S^n, A_1, B_1, C_1) && (34)\\ &\leq I(A_2; S^n, A_1, B_1, C_1) + o(n) && (35)\\ &= I(A_2; S^n|A_1, B_1, C_1) + o(n), && (36)\\ &= H(S^n|A_1, B_1, C_1) \\ &\quad - H(S^n|A_1, B_1, C_1, A_2) + o(n) && (37) \end{aligned}$$

where S^n in (34) refers to the received symbols in the index coding problem, (35) is due to Fano's inequality, and (36) is due to the independence between A_2 and (A_1, B_1, C_1). Since we only use vector linear coding schemes, (37) implies

$$R^{\text{IC}}_{A_2} + \dim(\mathbf{V}_{A_3,B_2,B_3,C_2,C_3}) \leq \dim(\mathbf{V}_{A_2,A_3,B_2,B_3,C_2,C_3}), \tag{38}$$

where $\mathbf{V}_{A_3,B_2,B_3,C_2,C_3}$ denotes the vector space spanned by the messages $(A_3, B_2, B_3, C_2, C_3)$.

For the left hand side of (38), we further have

$$\begin{aligned} &R^{\text{IC}}_{A_2} + \dim(\mathbf{V}_{A_3,B_2,B_3,C_2,C_3}) \\ &= R^{\text{IC}}_{A_2} + \dim(\mathbf{V}_{A_3}) + \dim(\mathbf{V}_{B_2,B_3,C_2,C_3}) && (39)\\ &= R^{\text{IC}}_{A_2} + \dim(\mathbf{V}_{A_3}) + \dim(\mathbf{V}_{B_2}) + \dim(\mathbf{V}_{B_3}) \\ &\quad + \dim(\mathbf{V}_{C_2,C_3}) && (40)\\ &\geq R^{\text{IC}}_{A_2} + \dim(\mathbf{V}_{A_3}) + \dim(\mathbf{V}_{B_2}) + \dim(\mathbf{V}_{B_3}) \\ &\quad + \dim(\mathbf{V}_{C_2}) && (41)\\ &\geq 5R^{\text{IC}} && (42) \end{aligned}$$

To guarantee that user 1 decodes correctly, we have $\dim(\mathbf{V}_{A_3} \cap \mathbf{V}_{B_2,B_3,C_2,C_3}) = 0$, since user 1 requests A_3 and has no side information about (B_2, B_3, C_2, C_3). Together with the fact that for vector spaces A and B,

$$\dim(A) + \dim(B) = \dim(A \cup B) + \dim(A \cap B), \tag{43}$$

we have (39). Similarly, $\dim(\mathbf{V}_{B_2} \cap \mathbf{V}_{B_3,C_2,C_3}) = 0$ and $\dim(\mathbf{V}_{B_3} \cap \mathbf{V}_{C_2,C_3}) = 0$, due to the requests of user 1 and the lack of side information of user 1. Therefore, we have (40). By using (43) and the fact that dimension is a nonnegative quantity, we have (41). To get (42), by applying Fano's inequality we have

$$\begin{aligned} nR^{\text{IC}}_{A_3} &\leq I(A_3; S^n|A_1, A_2, B_1, B_2, B_3, C_1, C_2, C_3) + o(n) && (44)\\ &= H(S^n|A_1, A_2, B_1, B_2, B_3, C_1, C_2, C_3) + o(n). && (45) \end{aligned}$$

Therefore,

$$R_{A_3}^{\mathrm{IC}} \leq \dim(\mathbf{V}_{A_3}) \tag{46}$$

Since we consider the symmetric capacity upper bound, we remove the subscript of R^{IC} to simplify the notation.

For the right hand side of (38), we have

$$\begin{aligned}&\dim(\mathbf{V}_{A_2,A_3,B_2,B_3,C_2,C_3}) + \dim(\mathbf{V}_{C_1})\\ &\quad= \dim(\mathbf{V}_{A_2,A_3,B_2,B_2,C_1,C_2,C_3}) &(47)\\ &\quad\leq \dim(\mathbf{V}_{A_1,A_2,A_3,B_1,B_2,B_3,C_1,C_2,C_3}) = 1. &(48)\end{aligned}$$

To guarantee that user 2 decodes correctly, we have $\dim(\mathbf{V}_{C_1} \cap \mathbf{V}_{A_3,B_3,C_3}) = 0$. Also, to guarantee that user 3 decodes correctly, we have $\dim(\mathbf{V}_{C_1} \cap \mathbf{V}_{A_2,B_2,C_2}) = 0$. Together with (43) and the fact that for vector spaces A, B and C,

$$\begin{aligned}\dim((A \cup B) \cap C) = \dim(A \cap C) + \dim(B \cap C)\\ - \dim((A \cap B) \cap C),\end{aligned} \tag{49}$$

we have (47). By using (43) and the fact that dimension is nonnegative, we have (48). Finally, combining (38), (42), and (48), we have

$$5R^{\mathrm{IC}} \leq 1 - \dim(\mathbf{V}_{C_1}). \tag{50}$$

With (46), we have $R^{\mathrm{IC}} \leq \frac{1}{6}$. Since the upper bound for the symmetric index coding capacity is $\frac{1}{6}$ and each subfile has the same size of $\frac{F}{3}$ bits, this results in the lower bound of $2F$ bits traffic load for the cache network. By applying the delivery scheme proposed in Section III, we know that normalized traffic load $R = 2$ is achievable. Thus, $R_l^*(\tilde{\mathbf{D}}, \mathbf{Z}_{\mathrm{sym}}^1) = 2$.

V. Tight Results

A. Distinct File Requests

If the users request LK distinct files in the delivery phase, then

$$R^*(\mathbf{D}, \mathbf{Z}_{\mathrm{sym}}^t) = L\frac{\binom{K}{t+1}}{\binom{K}{t}}. \tag{51}$$

We start with the converse bound. Since LK distinct files are requested, this means that each file is requested by at most one user. Therefore, we construct the virtual user according to the set $\mathcal{U} = [K]$, and fill Z_v according to the order of user 1, user 2 to user K. Assume that by receiving $X(\mathbf{D})$, each user can reconstruct the file they requested. Since the virtual user has more side information than user 1, by receiving $X(\mathbf{D})$ the virtual user can also reconstruct the files user 1 requested. This implies that $X(\mathbf{D})$ should contain at least $L\frac{\binom{K-1}{t}}{\binom{K}{t}}F$ bits of information. For user 2, this implies that $X(\mathbf{D})$ should contain another $L\frac{\binom{K-2}{t}}{\binom{K}{t}}F$ bits of information, since different L files are requested. Continuing this process up to user K, the normalized traffic load is at least

$$L\left(\frac{\binom{K-1}{t}}{\binom{K}{t}} + \frac{\binom{K-2}{t}}{\binom{K}{t}} + \cdots + \frac{\binom{t}{t}}{\binom{K}{t}}\right) = L\frac{\binom{K}{t+1}}{\binom{K}{t}}, \tag{52}$$

where the equality holds by the binomial theorem.

For the delivery scheme, since every candidate delivery message occupies at least one new dimension, by going over all the sets in $\mathcal{S}$, and $1 \leq l \leq L$, we observe that all the candidate delivery message are linearly independent. Therefore, $\mathrm{rank}(\mathbf{Y})$ in (29) is $L\binom{K}{t+1}$, and we have a tight result for the case of distinct file requests.

B. Reduction to One File Request

For $L = 1$, our general delivery scheme has the same result as in [4], i.e., if $L = 1$, then

$$R^*(\mathbf{D}, \mathbf{Z}_{\mathrm{sym}}^t) = \frac{\binom{K}{t+1} - \binom{K-n}{t+1}}{\binom{K}{t}}, \tag{53}$$

where n is the number of different files in $\mathbf{D}$.

For the converse bound, since n different files are requested, we have $|\mathcal{U}| = n$. By using similar arguments as in Section V-A, we have that the normalized traffic load is at least

$$\frac{\binom{K-1}{t} + \binom{K-2}{t} + \cdots + \binom{K-n}{t}}{\binom{K}{t}} = \frac{\binom{K}{t+1} - \binom{K-n}{t+1}}{\binom{K}{t}}. \tag{54}$$

For the delivery scheme, by going over all the sets in $\mathcal{S}$, we have $\binom{K}{t+1}$ candidate deliveries. Reference [4, Lemma 1] shows that some set of candidate delivery messages can be obtained through linear combination of other set of candidate delivery messages, which results in $\binom{K}{t+1} - \binom{K-n}{t+1}$ messages delivered. Therefore, we have $\mathrm{rank}(\mathbf{Y}) \leq \binom{K}{t+1} - \binom{K-n}{t+1}$. Combining with (54), we have (53).

C. Two File Requests

In Section V-A, we allow arbitrary L but require distinct file requests, while in Section V-B, we allow arbitrary file requests but require $L = 1$. Here, we consider $L = 2$, and allow arbitrary file requests. These results are obtained through the delivery scheme presented in Section III. To characterize $R^*(\mathbf{D}, \mathbf{Z}_{\mathrm{sym}}^t)$, we use the converse technique presented in Section IV-A, while for $R_l^*(\mathbf{D}, \mathbf{Z}_{\mathrm{sym}}^t)$, we use the converse technique presented in Section IV-B.

We first consider $L = 2$, $t = 1$ and $K = 3$. We list all the possible cases of file requests here. Other cases not listed here can be obtained through permutations and relabeling.

$$\mathbf{D}_1 = \begin{pmatrix} A & B & C \\ D & E & F \end{pmatrix}, R^*(\mathbf{D}_1, \mathbf{Z}_{\mathrm{sym}}^1) = \frac{6}{3}. \tag{55}$$

$$\mathbf{D}_2 = \begin{pmatrix} A & A & B \\ C & D & E \end{pmatrix}, R^*(\mathbf{D}_2, \mathbf{Z}_{\mathrm{sym}}^1) = \frac{6}{3}. \tag{56}$$

$$\mathbf{D}_3 = \begin{pmatrix} A & A & A \\ B & C & D \end{pmatrix}, R^*(\mathbf{D}_3, \mathbf{Z}_{\mathrm{sym}}^1) = \frac{5}{3}. \tag{57}$$

$$\mathbf{D}_4 = \begin{pmatrix} A & A & B \\ B & C & D \end{pmatrix}, R^*(\mathbf{D}_4, \mathbf{Z}_{\mathrm{sym}}^1) = \frac{6}{3}. \tag{58}$$

$$\mathbf{D}_5 = \begin{pmatrix} A & A & C \\ B & B & D \end{pmatrix}, R^*(\mathbf{D}_5, \mathbf{Z}_{\mathrm{sym}}^1) = \frac{6}{3}. \tag{59}$$

$$\mathbf{D}_6 = \begin{pmatrix} A & A & A \\ B & B & C \end{pmatrix}, R^*(\mathbf{D}_6, \mathbf{Z}_{\mathrm{sym}}^1) = \frac{5}{3}. \tag{60}$$

$$\mathbf{D}_7 = \begin{pmatrix} A & A & B \\ B & C & C \end{pmatrix}, R_l^*(\mathbf{D}_7, \mathbf{Z}_{\text{sym}}^1) = \frac{6}{3}. \quad (61)$$

$$\mathbf{D}_8 = \begin{pmatrix} A & A & A \\ B & B & B \end{pmatrix}, R^*(\mathbf{D}_8, \mathbf{Z}_{\text{sym}}^1) = \frac{4}{3}. \quad (62)$$

Note that $\mathbf{D}_1$ is a case of distinct file requests as shown in Section V-A. For $\mathbf{D}_2$, if we construct the virtual user according to the order of user 3, user 2, and user 1, then we have a tight converse result. However, if we construct the virtual user according to the order of user 1, user 2, and user 3, then we can only show that the normalized traffic load must be greater than $\frac{5}{3}$. For $\mathbf{D}_7$, no matter which ordering we use to construct the virtual user, we can only show that the normalized traffic load $R \geq \frac{5}{3}$. Instead, by using the converse technique in Section IV-B, we characterize the optimal linear coding rate for $\mathbf{D}_7$ as $R_l^*(\mathbf{D}_7, \mathbf{Z}_{\text{sym}}^1) = \frac{6}{3}$.

We then consider $L = 2$, $t = 1$, and $K = 4$. The following cases are all the representative cases for $K = 4$.

$$\mathbf{D}_1' = \begin{pmatrix} A & B & C & D \\ E & F & G & H \end{pmatrix}, R^*(\mathbf{D}_1', \mathbf{Z}_{\text{sym}}^1) = \frac{12}{4}. \quad (63)$$

$$\mathbf{D}_2' = \begin{pmatrix} A & A & B & C \\ D & E & F & G \end{pmatrix}, R^*(\mathbf{D}_2', \mathbf{Z}_{\text{sym}}^1) = \frac{12}{4}. \quad (64)$$

$$\mathbf{D}_3' = \begin{pmatrix} A & A & A & B \\ C & D & E & F \end{pmatrix}, R^*(\mathbf{D}_3', \mathbf{Z}_{\text{sym}}^1) = \frac{11}{4}. \quad (65)$$

$$\mathbf{D}_4' = \begin{pmatrix} A & A & B & B \\ C & D & E & F \end{pmatrix}, R^*(\mathbf{D}_4', \mathbf{Z}_{\text{sym}}^1) = \frac{11}{4}. \quad (66)$$

$$\mathbf{D}_5' = \begin{pmatrix} A & A & B & C \\ B & D & E & F \end{pmatrix}, R^*(\mathbf{D}_5', \mathbf{Z}_{\text{sym}}^1) = \frac{12}{4}. \quad (67)$$

$$\mathbf{D}_6' = \begin{pmatrix} A & A & C & D \\ B & B & E & F \end{pmatrix}, R^*(\mathbf{D}_6', \mathbf{Z}_{\text{sym}}^1) = \frac{12}{4}. \quad (68)$$

$$\mathbf{D}_7' = \begin{pmatrix} A & A & A & A \\ B & C & D & E \end{pmatrix}, R^*(\mathbf{D}_7', \mathbf{Z}_{\text{sym}}^1) = \frac{9}{4}. \quad (69)$$

$$\mathbf{D}_8' = \begin{pmatrix} A & A & A & B \\ B & C & D & E \end{pmatrix}, R^*(\mathbf{D}_8', \mathbf{Z}_{\text{sym}}^1) = \frac{11}{4}. \quad (70)$$

$$\mathbf{D}_9' = \begin{pmatrix} A & A & A & C \\ B & B & D & E \end{pmatrix}, R^*(\mathbf{D}_9', \mathbf{Z}_{\text{sym}}^1) = \frac{11}{4}. \quad (71)$$

$$\mathbf{D}_{10}' = \begin{pmatrix} A & A & A & A \\ B & B & C & D \end{pmatrix}, R^*(\mathbf{D}_{10}', \mathbf{Z}_{\text{sym}}^1) = \frac{9}{4}. \quad (72)$$

$$\mathbf{D}_{11}' = \begin{pmatrix} A & A & A & C \\ B & B & B & D \end{pmatrix}, R^*(\mathbf{D}_{11}', \mathbf{Z}_{\text{sym}}^1) = \frac{10}{4}. \quad (73)$$

$$\mathbf{D}_{12}' = \begin{pmatrix} A & A & A & B \\ B & B & C & D \end{pmatrix}, R^*(\mathbf{D}_{12}', \mathbf{Z}_{\text{sym}}^1) = \frac{10}{4}. \quad (74)$$

$$\mathbf{D}_{13}' = \begin{pmatrix} A & A & B & B \\ C & C & D & E \end{pmatrix}, R^*(\mathbf{D}_{13}', \mathbf{Z}_{\text{sym}}^1) = \frac{11}{4}. \quad (75)$$

$$\mathbf{D}_{14}' = \begin{pmatrix} A & A & B & C \\ B & C & D & E \end{pmatrix}, R^*(\mathbf{D}_{14}', \mathbf{Z}_{\text{sym}}^1) = \frac{11}{4}. \quad (76)$$

$$\mathbf{D}_{15}' = \begin{pmatrix} A & A & B & D \\ B & C & C & E \end{pmatrix}, R_l^*(\mathbf{D}_{15}', \mathbf{Z}_{\text{sym}}^1) = \frac{12}{4}. \quad (77)$$

$$\mathbf{D}_{16}' = \begin{pmatrix} A & A & A & B \\ B & C & C & D \end{pmatrix}, R^*(\mathbf{D}_{16}', \mathbf{Z}_{\text{sym}}^1) = \frac{10}{4}. \quad (78)$$

$$\mathbf{D}_{17}' = \begin{pmatrix} A & A & A & B \\ B & C & D & C \end{pmatrix}, R_l^*(\mathbf{D}_{17}', \mathbf{Z}_{\text{sym}}^1) = \frac{11}{4}. \quad (79)$$

$$\mathbf{D}_{18}' = \begin{pmatrix} A & A & B & B \\ C & C & D & D \end{pmatrix}, R^*(\mathbf{D}_{18}', \mathbf{Z}_{\text{sym}}^1) = \frac{10}{4}. \quad (80)$$

$$\mathbf{D}_{19}' = \begin{pmatrix} A & A & B & C \\ B & C & D & D \end{pmatrix}, R^*(\mathbf{D}_{19}', \mathbf{Z}_{\text{sym}}^1) = \frac{10}{4}. \quad (81)$$

$$\mathbf{D}_{20}' = \begin{pmatrix} A & A & A & B \\ B & B & C & C \end{pmatrix}, R_l^*(\mathbf{D}_{20}', \mathbf{Z}_{\text{sym}}^1) = \frac{9}{4}. \quad (82)$$

$$\mathbf{D}_{21}' = \begin{pmatrix} A & A & A & A \\ B & B & B & C \end{pmatrix}, R^*(\mathbf{D}_{21}', \mathbf{Z}_{\text{sym}}^1) = \frac{8}{4}. \quad (83)$$

$$\mathbf{D}_{22}' = \begin{pmatrix} A & A & A & A \\ B & B & B & B \end{pmatrix}, R^*(\mathbf{D}_{22}', \mathbf{Z}_{\text{sym}}^1) = \frac{6}{4}. \quad (84)$$

Here, $\mathbf{D}_1'$ corresponds to distinct file requests as shown in Section V-A. For $\mathbf{D}_{19}'$, if we construct the virtual user according to the order of user 2, user 3, user 1 and user 4, then we have a tight converse result. However, if we construct the virtual user according to the order of user 1, user 2, user 3, and user 4, we can only show that the normalized traffic load must be greater than $\frac{9}{3}$. For $\mathbf{D}_{15}'$, $\mathbf{D}_{17}'$ and $\mathbf{D}_{20}'$, we can only characterize the optimal linear coding rate. The converse bound derivation is similar to that in Section IV-B, since $\tilde{\mathbf{D}}$ is a sub-matrix of these matrices through some elementary operations.

VI. Conclusion

For the two-phase caching network with multiple file requests, we adopt symmetric batch caching and propose a general delivery scheme. The general delivery scheme is based on sending only those candidate linear combinations which provide new linearly independent equations. The general delivery scheme is shown to be optimal for certain cases through two converse techniques. The first technique is based on virtual user construction and the second converse technique uses an interference alignment point of view.

References

[1] M. A. Maddah-Ali and U. Niesen. Fundamental limits of caching. *IEEE Trans. on Inf. Theory*, 60(5):2856–2867, May 2014.

[2] M. A. Maddah-Ali and U. Niesen. Decentralized coded caching attains order-optimal memory-rate tradeoff. *IEEE/ACM Trans. on Networking*, 23(4):1029–1040, Aug. 2015.

[3] K. Wan, D. Tuninetti, and P. Piantanida. On the optimality of uncoded cache placement. In *IEEE ITW*, Sep. 2016.

[4] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr. The exact rate-memory tradeoff for caching with uncoded prefetching. In *IEEE ISIT*, Jun. 2017.

[5] F. Arbabjolfaei, B. Bandemer, Y.-H. Kim, E. Şaşoğlu, and L. Wang. On the capacity region for index coding. In *IEEE ISIT*, Jul. 2013.

[6] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr. Characterizing the rate-memory tradeoff in cache networks within a factor of 2. In *IEEE ISIT*, Jun. 2017.

[7] M. Ji, A. M. Tulino, J. Llorca, and G. Caire. Caching and coded multicasting: Multiple groupcast index coding. In *IEEE GlobalSIP*, Dec. 2014.

[8] K. Shanmugam, A. G. Dimakis, and M. Langberg. Local graph coloring and index coding. In *IEEE ISIT*, Jul. 2013.

[9] A. Sengupta and R. Tandon. Improved approximation of storage-rate tradeoff for caching with multiple demands. *IEEE Trans. on Comm.*, 65(5):1940–1955, Feb. 2017.

[10] H. Maleki, V. R. Cadambe, and S. A. Jafar. Index coding – An interference alignment perspective. *IEEE Trans. on Inf. Theory*, 60(9):5402–5432, Sep. 2014.

[11] H. Sun and S. A. Jafar. Index coding capacity: How far can one go with only shannon inequalities? *IEEE Trans. on Inf. Theory*, 61(6):3041–3055, Jun. 2015.