Data Dissemination With Side Information and Feedback

Mingjun Dai, Kenneth W. Shum, *Member, IEEE*, and Chi Wan Sung, *Member, IEEE*

***Abstract*—Index coding (IC), which can be regarded as a special class of network coding, deals with the problem of sending a number of packets to a group of receivers, each of which requests one packet and may have some other packets in its cache. This paper generalizes the IC problem in that both the packet requested by a receiver and the packets in its cache can be linear combinations of the packets. To minimize the number of transmissions required, a heuristic algorithm based on the idea of partitioning the users into coding groups is designed. To realize this idea, a polynomial time algorithm to determine whether a set of users form a coding group over the binary field or a field with a size larger than the number of users is constructed. For users that form a coding group, the corresponding encoding vector can be also found. A lower bound is derived in order to evaluate the performance of the heuristic algorithm. Numerical results show that the number of transmissions required by the heuristic algorithm and the lower bound both grow roughly linearly with the number of users, and the heuristic algorithm outperforms some benchmark algorithms.**

***Index Terms*—Index coding, network coding, data dissemination, coded side information, broadcast relay channel.**

I. Introduction

DATA dissemination belongs to the broadcasting scenarios, where one access point (AP) or base station (BS) sends packets to a number of users. Different users may have different packet requests and the sender wants to meet all their requests as soon as possible [1]. For example, in a waiting room of an airport or railway station, some passengers are watching online television program, some are surfing a popular webpage, while others are downloading files, all through WiFi.

Due to the broadcast nature of wireless communication, each user is able to overhear the signals intended to other nearby users [2]. Although these overheard data packets are not requested by the user, they may serve as side information for the decoding of data packets arriving later. Utilization of overheard data packets is essential in enabling network coding (NC) techniques [3]. For example, in the COPE system [4] for wireless mesh networks, the user nodes are set in promiscuous mode and store the captured signals for a window of time, so that the user nodes can leverage opportunistic coding by sending the bitwise XOR of two or more packets. In the problem of index coding (IC) [5], which is motivated by increasing throughput in satellite communications, the ground stations may capture some data packets sent from the satellite which are not intended to them. Then the satellite can exploit the opportunity of transmitting some mixtures of the data packets to the ground stations, with the aim of minimizing the total transmission time.

The IC problem has received a considerable amount of attention recently. It is shown in [6] that it is closely related to NC, and in [7] that it can be formulated in terms of interference alignment. IC from an information-theoretic point of view is studied in [8]. Bounds on broadcast completion time, derived in [9] and [10], show that nonlinear IC outperforms linear IC. By linear IC, we mean that the data packets are regarded as vectors over a finite field, and we mix the data packets by linearly combining them. Though linear IC, in general, underperforms nonlinear IC [9], it enjoys practical simplicity. From now on, we restrict ourselves to linear IC. The term "IC" will refer to linear IC unless stated otherwise.

The IC problem is shown to be equivalent to determining the *minrank* of a side-information graph [8], which has been proven to be NP-hard [11]–[13]. On one hand, characterizations of the maximum transmission efficiency of the IC problem can be found in [14]–[16], which try to approximate the broadcast capacity and the corresponding performance bound of the IC problem. In addition, the works in [17]–[19] identify some special side-information graphs whose optimal code can be found efficiently. On the other hand, in order to apply IC to practical systems, some efficient sub-optimal algorithms are proposed in [20]–[25].

In all of the above works, the side information is in uncoded form. The objective of this paper is to show that side information in coded form is also useful in reducing the total completion time, where we confine *coded packets* to linear combinations of the basic uncoded packets over some finite

Manuscript received February 19, 2013; revised December 9, 2013 and April 20, 2014; accepted June 6, 2014. Date of publication June 30, 2014; date of current version September 8, 2014. This work was supported in part by the University Grants Committee of the Hong Kong Special Administrative Region, China (AoE/E-02/08), from the Research Grants Council of the Hong Kong Special Administrative Region, China (CityU 121713), from Natural Science Foundation of China (61301182, 61372078), in part from the National 973 project (2013CB336700), from Natural Science Foundation of Guangdong Province (S2013040016857), from Specialized Research Fund for the Doctoral Program of Higher Education from The Ministry of Education (20134408120004), from Yumiao Engineering from Education Department of Guangdong Province (2013LYM_0077), from the Open Research Fund of the State Key Laboratory of Integrated Services Networks, Xidian University (ISN15-06), and from Natural Science Foundation of Shenzhen University (00036107). The associate editor coordinating the review of this paper and approving it for publication was G. Zussman.

M. Dai is with the College of Information Engineering, Shenzhen University, Shenzhen 518060, China. He is also with The State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China (e-mail: mjdai@szu.edu.cn).

K. W. Shum is with the Institute of Network Coding, The Chinese University of Hong Kong, Shatin, Hong Kong (e-mail: wkshum@inc.cuhk.edu.hk).

C. W. Sung is with the Department of Electronic Engineering, College of Science and Engineering, City University of Hong Kong, Kowloon, Hong Kong (e-mail: albert.sung@cityu.edu.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TWC.2014.2330571

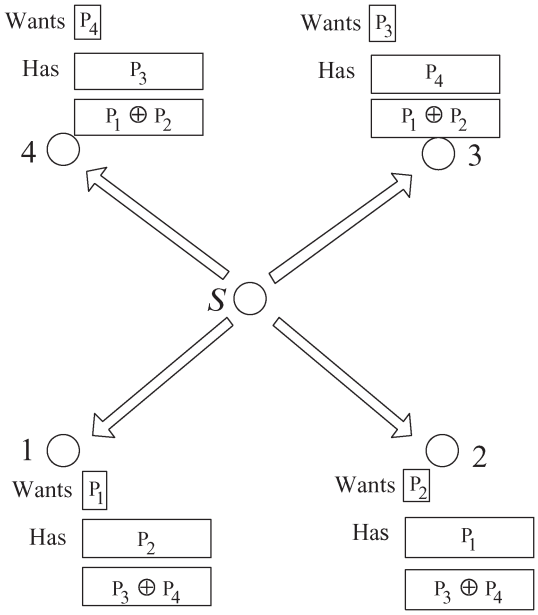

Fig. 1. Example of the index coding problem.

field. As the source node is transmitting both coded and uncoded data packets, coded data packets are available to a user if the user captures all packets from the source node. Consider the example shown in Fig. 1. Coding is performed over GF(2). There are four users and four packets. User i wants to download basic uncoded packet P_i, for $i = 1, 2, 3, 4$. Suppose that user 1 has P_2 and $P_3 + P_4$, user 2 has P_1 and $P_3 + P_4$, user 3 has P_4 and $P_1 + P_2$, user 4 has P_3 and $P_1 + P_2$. If the packets in their cache are not used at all, we need four packet transmissions (P_1, P_2, P_3, and P_4) to satisfy the users. If only the uncoded packets in their cache are used, two is enough ($P_1 + P_2$ and $P_3 + P_4$). If all packets in their cache are used, the source only needs to transmit one coded packet $P_1 + P_2 + P_3 + P_4$. This example illustrates that the total completion time can be further reduced if coded packets are cached and used.

The requested and cached packets in some applications are indeed coded. We list some examples below: 1) In a WiFi network, retransmission process is needed to counteract the packet erasure effect. To increase throughput, coded packet transmissions are considered in [1]. Although in their system model, receivers do not store coded packets in their cache for future use, it is conceptually simple to extend their model so that receivers can keep track of their received coded packets so as to further boost the system throughput of a WiFi network. 2) In wireless distributed (cloud) storage systems [26], [27], coding methods such as erasure codes and regenerating codes are normally used to encode the message into coded packets for storing in the storage nodes. If a certain storage node loses some packets due to disk failure, a repair process is initiated. The storage node will request the lost packets, which are coded. The packets that are not lost, which are also coded, can be used as side information to speed up the repair process. 3) In the broadcast relay channel (BRC) [28] as described in a later section, coded packets as side information and requests naturally come into play under practical packet erasure scenario.

In this paper, we generalize the IC problem by assuming that the sender and the users possess coded packets and the users requests coded packets. The resultant problem is referred to as the *generalized IC* (GIC) problem. Our proposed heuristic solution involves two components. First, we derive a criterion that judges whether a coded packet can satisfy a given group of users if this packet is successfully received by all members in this group. Here, *satisfying a group of users* means that each user in this group is able to decode his/her requested packet. We call such group of users a *coding group*. An efficient algorithm is proposed to determine whether a given group of users can form a coding group over the binary field or over a field with large size. Second, we propose a heuristic method to partition the users into disjoint coding groups. A heuristic solution based on these two components is then given to solve the GIC problem. To evaluate its effectiveness, we derive a lower bound for the number of transmissions required. Numerical results show that the number of transmissions required by our proposed method and the lower bound both grow roughly linearly with the number of users. Besides, in more practical complex topologies, e.g., the BRC with packet erasures, numerical results also confirm the superiority of our method over some benchmark methods.

The idea of partitioning the users into several groups and treating each group separately can also be found in some existing works. Instantly decodable network code (see e.g. [11], [20], [25]) is a heuristic in which a user only uses one of the packets transmitted by the source node to decode the required packet. No linear combination of the transmitted packets is used. Under the heuristic of instant decodability, the IC problem is formulated as a clique partitioning problem on a graph called the side information graph. In [29], the IC problem is formulated using a bipartite graph, and like what we assume in this paper, a packet may be requested by two or more users.

All of the aforementioned works on IC consider uncoded packets as side information and uncoded packet requests, and can be considered special cases of our problem formulated in Section II. Our formulation is an extension of our previous work in [30] from the binary field to a general finite field. A theoretical lower bound on the completion time is presented in Section III. A heuristic algorithm to solve the GIC problem based on the notion of coding group is given in Section IV. Its computational complexity is analyzed in Section V. A series of numerical study is performed in Section VI. In Section VII, we apply the proposed heuristic algorithm to the broadcast relay channel with packet erasures. In Section VIII, we conclude the paper and give directions for future investigations.

II. Problem Formulation

Consider a group of M users, denoted by $\mathcal{M} \triangleq \{1, \ldots, M\}$, and N source packets, $P_1, P_2 \ldots, P_N$. The packets are regarded as elements in the finite field GF(q) for some prime power q. Suppose that each user requests a packet. In the original IC problem, the requested packets are uncoded packets. In our formulation, it is helpful to make a slight generalization and suppose that each user requests a *coded packet*. By a coded packet we mean a linear combination of the N packets

$\sum_{j=1}^{N} c_j P_j$ with coefficients c_j taken from GF(q). The vector formed by the N coefficients is called the *coding vector* of this packet. We assume without loss of generality that all coding vectors are non-zero vectors, because the packet corresponding to the zero coding vector is the zero packet and conveys no information.

For $i \in \mathcal{M}$, the coding vector of the packet requested by user i is called the *request vector*, and is denoted by $\boldsymbol{r}_i$. If the request vector is the j-th standard basis vector, $\boldsymbol{e}_j \triangleq (\underbrace{0,\ldots,0,1}_{j},0,\ldots,0)$, then the requested packet is the packet P_j. All vectors in this paper are column vectors unless stated otherwise. Besides, we use the notation $(x_1, x_2, \ldots, x_n)$ to denote a column vector.

Each user is assumed to have some coded packets as side information stored in his cache. For $i \in \mathcal{M}$, suppose that there are K_i coded packets stored in the cache of user i. These K_i coded packets are specified by an $N \times K_i$ matrix over GF(q), called the *coding matrix* $\boldsymbol{C}_i$ of user i, whose columns are the coding vectors of these K_i coding vectors. Without loss of generality, we assume that $\boldsymbol{C}_i$ is of full rank for all i. It is assumed that there is a perfect feedback channel from each of the users to the sender, so that the sender knows the side information and the requests of the users.

To satisfy the demand of all the M users, a sender, denoted by S, broadcasts packets according to the requests, $\boldsymbol{r}_i$'s, and cached information, $\boldsymbol{C}_i$'s, of the users. For generality, S may or may not have all the N uncoded packets in its cache. Similar to each user, S is assumed to have K_S coded packets and its coding matrix is denoted by $\boldsymbol{C}_S$. We assume that all packets broadcast by S can be received correctly by all users.

We focus on the case where each user cannot construct its requested packet by the packets in its cache. Mathematically, this is equivalent to the following conditions:

$$\boldsymbol{r}_i \text{ is not in span}(\boldsymbol{C}_i),\ \forall i \in \mathcal{M}, \tag{1}$$

where span($\boldsymbol{M}$) represents the column span of matrix $\boldsymbol{M}$. This condition can also be expressed in terms of the rank function. We introduce the notation that, for matrices $\boldsymbol{M}_1, \boldsymbol{M}_2, \ldots, \boldsymbol{M}_k$ having the same number of rows, we let rank($\boldsymbol{M}_1, \boldsymbol{M}_2, \ldots, \boldsymbol{M}_k$) be the rank of the concatenated matrix $[\boldsymbol{M}_1 \boldsymbol{M}_2 \cdots \boldsymbol{M}_k]$. Then (1) is equivalent to

$$\text{rank}(\boldsymbol{C}_i, \boldsymbol{r}_i) = \text{rank}(\boldsymbol{C}_i) + 1,\ \forall i \in \mathcal{M}. \tag{1'}$$

We also focus on the case where the sender has enough information to satisfy all users, namely

$$\boldsymbol{r}_i \in \text{span}(\boldsymbol{C}_S, \boldsymbol{C}_i),\ \forall i \in \mathcal{M}, \tag{2}$$

or equivalently

$$\text{rank}(\boldsymbol{C}_S, \boldsymbol{C}_i) = \text{rank}(\boldsymbol{C}_S, \boldsymbol{C}_i, \boldsymbol{r}_i),\ \forall i \in \mathcal{M}. \tag{2'}$$

The condition in (2), or in (2'), means that the coding vector of the packet requested by user i lies within the sum space of the column spaces of $\boldsymbol{C}_S$ and $\boldsymbol{C}_i$, and thus can be constructed by the packets held by the sender S and user i.

In this paper, we consider the optimization problem of minimizing the number of packets broadcast by S so as to satisfy the demand of all users. If the sender transmits L coded packets, the coding vectors of these L packets can be concatenated as an $N \times L$ matrix, which can be written in the form $\boldsymbol{C}_S \boldsymbol{E}$ for some $K_S \times L$ matrix $\boldsymbol{E}$. The optimization is to find the smallest integer L such that there exists a $K_S \times L$ matrix $\boldsymbol{E}$ that satisfies the following condition:

$$\text{rank}(\boldsymbol{C}_S \boldsymbol{E}, \boldsymbol{C}_i) = \text{rank}(\boldsymbol{C}_S \boldsymbol{E}, \boldsymbol{C}_i, \boldsymbol{r}_i),\ \forall i \in \mathcal{M}. \tag{3}$$

This condition means that each user can decode the requested packet based on the L broadcast packets and its K_i cached packets. By the assumption in (2'), we can choose the $K_S \times K_S$ identity matrix as a feasible solution, and thus the minimization problem is well defined. Note that this formulation includes the original IC problem as a special case, and is thus NP-hard.

Example 1: Let $N = M = 5$. Suppose that user i requests packet i, for $i = 1, 2, \ldots, 5$, i.e., $\boldsymbol{r}_i = \boldsymbol{e}_i$. The size of the finite field in this example could be any prime power. Suppose that user 1 has packet P_2, user 2 has packet $P_1 + P_5$, user 3 has packets P_1 and P_4, user 4 has packet $P_1 + P_2 + P_3$, and user 5 has packets P_2 and $P_1 + P_3$, as side information. The coding matrices of the sender and the five users are shown below:

$$\boldsymbol{C}_S = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},\ \boldsymbol{C}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix},\ \boldsymbol{C}_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix},$$

$$\boldsymbol{C}_3 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix},\ \boldsymbol{C}_4 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix},\ \boldsymbol{C}_5 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

The sender can broadcast three coded packets $P_1 + P_2$, $P_3 + P_4$, and P_5 in order to satisfy the demands of all users. User 5 can obtain the requested packet directly without any information processing. For users 1 to 4, the requested packets can be recovered as follows,

$$\begin{aligned} &\text{User 1}: \ P_1 = (P_1 + P_2) - P_2, \\ &\text{User 2}: \ P_2 = (P_1 + P_2) + P_5 - (P_1 + P_5), \\ &\text{User 3}: \ P_3 = (P_3 + P_4) - P_4, \\ &\text{User 4}: \ P_4 = (P_1 + P_2) + (P_3 + P_4) - (P_1 + P_2 + P_3). \end{aligned}$$

We will show in the next section that the source node has to send at least three coded packets in this example, and hence three packet transmissions are optimal.

There is a geometric interpretation of the GIC problem. From the side information, a user can compute any packet whose coding vector belongs to the subspace generated by the columns of the corresponding coding matrix. Hence, each user has the knowledge of all vectors in a subspace of GF$(q)^N$. To satisfy the demand of user i, the source node needs to find a coding vector belonging to the subspace spanned by the request vector

$\boldsymbol{r}_i$ and the columns of $\boldsymbol{C}_i$, but not the subspace spanned by the column of $\boldsymbol{C}_i$. The task of the source node is to find the minimum number of coding vectors to satisfy all users.

III. A Lower Bound on Completion Time

In the original IC problem, for all $i \in \mathcal{M}$, user i requests P_i and his cached packets are all uncoded. The side information possessed by the users can be represented by an undirected graph G. User i is represented by vertex v_i. Two vertices v_i and v_j are connected by an edge if user i caches P_j while user j caches P_i. A graph so constructed is called the *side information graph*. It is shown in [8] that the minimum number of packet transmissions needed to satisfy all users' demands by a linear index code is lower bounded by the maximum cardinality of an independent set in G. The lower bound follows from the fact that the users in an independent set have no mutual information and have to be served separately.

For the GIC problem, we generalize the concept of independent set. For a non-empty subset $\mathcal{U}$ of $\mathcal{M}$, we define $\boldsymbol{R}(\mathcal{U})$ to be the $N \times |\mathcal{U}|$ matrix obtained by concatenating the request vectors associated with the users in $\mathcal{U}$. Similarly, we let $\boldsymbol{C}(\mathcal{U})$ denote the matrix obtained by concatenating the coding matrices associated with the users in $\mathcal{U}$.

Consider a non-empty subset of users $\mathcal{U}$. If we assume hypothetically that they can cooperate by sharing their side information, then they can jointly compute any vector in the column space of $\boldsymbol{C}(\mathcal{U})$. Even if they cooperate, we need to send at least

$$\rho(\mathcal{U}) \triangleq \text{rank}\left(\boldsymbol{R}(\mathcal{U}), \boldsymbol{C}(\mathcal{U})\right) - \text{rank}\left(\boldsymbol{C}(\mathcal{U})\right)$$

packets to the group of users $\mathcal{U}$. For each $i \in \mathcal{M}$, we have $\rho(\{i\}) = 1$ by the assumption in (1), namely $\boldsymbol{r}_i$ does not belong to the column space of $\boldsymbol{C}_i$.

Theorem 1: For $\mathcal{U} \subseteq \mathcal{M}$, the minimum number of packet transmissions required to satisfy the demands of users in $\mathcal{U}$, denoted by $L^*(\mathcal{U})$, is lower bounded by $\rho(\mathcal{U})$. In particular, we have

$$L^*(\mathcal{M}) \geq \rho(\mathcal{U}) \tag{4}$$

for all $\mathcal{U} \subseteq \mathcal{M}$.

Proof: For each subset $\mathcal{U}$ of users, if they cooperate in decoding, the number of required packet transmissions is no less than $\rho(\mathcal{U})$. If they do not cooperate, the number of packet transmissions should be larger than or equal to $\rho(\mathcal{U})$.

For the inequality in (4), we note that if the demands of all users are satisfied, then the demands of the users in $\mathcal{U}$ are satisfied. Hence $L^*(\mathcal{M}) \geq L^*(\mathcal{U}) \geq \rho(\mathcal{U})$. ■

The previous theorem states that for any set of users $\mathcal{U}$, the value of $\rho(\mathcal{U})$ is a lower bound on the total completion time. As the number of subsets of users grows exponentially with the total number of users, M, it is not practical to exhaustively search for a subset of $\mathcal{M}$ which yields the tightest lower bound in (4). In the rest of this section, we develop a heuristic algorithm to compute a lower bound on $L^*(\mathcal{M})$. We will make use of the monotone and submodular properties of the rank function. Let $\boldsymbol{A}$ be an $m \times n$ matrix, and for each subset $\mathcal{S}$ of $\{1, 2, \ldots, n\}$, let $\boldsymbol{A}_{\mathcal{S}}$ denote the submatrix of $\boldsymbol{A}$ obtained by retaining the columns indexed by $\mathcal{S}$. The monotone property of rank function says that $\text{rank}(\boldsymbol{A}_{\mathcal{S}}) \leq \text{rank}(\boldsymbol{A}_{\mathcal{T}})$ whenever $\mathcal{S} \subseteq \mathcal{T} \subseteq \{1, 2, \ldots, n\}$, and the submodular property states that

$$\text{rank}(\boldsymbol{A}_{\mathcal{S}}) + \text{rank}(\boldsymbol{A}_{\mathcal{T}}) \geq \text{rank}(\boldsymbol{A}_{\mathcal{S}\cap\mathcal{T}}) + \text{rank}(\boldsymbol{A}_{\mathcal{S}\cup\mathcal{T}}),$$

for all subsets $\mathcal{S}$ and $\mathcal{T}$ of $\{1, 2, \ldots, n\}$.

The next lemma lists some properties of the function ρ.

Lemma 2:

(i) For all $\mathcal{U} \subseteq \mathcal{M}$, we have $0 \leq \rho(\mathcal{U}) \leq |\mathcal{U}|$.

(ii) If $\rho(\mathcal{U}) = |\mathcal{U}|$, then $\rho(\mathcal{U}') = |\mathcal{U}'|$ for all subsets $\mathcal{U}' \subseteq \mathcal{U}$.

Proof: (i) Since $\boldsymbol{C}(\mathcal{U})$ is a submatrix of the concatenated matrix $[\boldsymbol{R}(\mathcal{U})\ \boldsymbol{C}(\mathcal{U})]$, according to the monotone property of rank function, we get $\rho(\mathcal{U}) \geq 0$. We have

$$|\mathcal{U}| + \text{rank}\left(\boldsymbol{C}(\mathcal{U})\right) \geq \text{rank}\left(\boldsymbol{R}(\mathcal{U})\right) + \text{rank}\left(\boldsymbol{C}(\mathcal{U})\right) \tag{5}$$

$$\geq \text{rank}\left(\boldsymbol{R}(\mathcal{U}), \boldsymbol{C}(\mathcal{U})\right), \tag{6}$$

where (5) is due to the fact that $\boldsymbol{R}(\mathcal{U})$ has $|\mathcal{U}|$ columns, and the inequality in (6) follows from the submodular property of the rank function. This proves that $\rho(\mathcal{U}) \leq |\mathcal{U}|$.

(ii) If $\rho(\mathcal{U}) = |\mathcal{U}|$, then the request vectors of $\mathcal{U}$ must be linearly independent and none of them lies in the linear span of $\boldsymbol{C}(\mathcal{U})$, implying that both (5) and (6) hold with equality. This implies that, for any subset $\mathcal{U}'$ of $\mathcal{U}$, the request vectors of $\mathcal{U}'$ are also linearly independent, and do not lie in the linear span of $\boldsymbol{C}(\mathcal{U}')$. Hence, equality holds in (5) and (6) with $\mathcal{U}$ replaced by any subset $\mathcal{U}'$ of $\mathcal{U}$. This proves that $\rho(\mathcal{U}') = |\mathcal{U}'|$. ■

For any given set of users $\mathcal{U}$, the first part of previous lemma states that $\rho(\mathcal{U})$ is upper bounded by the cadinality of $\mathcal{U}$. In the extreme case where $\rho(\mathcal{U}) = |\mathcal{U}|$, we say that the users in $\mathcal{U}$ are *independent*. This represents the case that cooperation among the users in $\mathcal{U}$ does not help in reducing the total completion time.

The second part of Lemma 2 says that if $\mathcal{U}$ is a set of independent users, then any subset $\mathcal{U}'$ of $\mathcal{U}$ is also independent. Furthermore, each user as a group of size one is independent. A set of users $\mathcal{U}$ is said to be *maximally independent* if it is not properly contained in another set of independent users.

Based on the idea of maximally independent set of users, we describe a heuristic method to compute a lower bound (we name it as *heuristic lower bound*) as follows. We first initialize $\mathcal{U}$ to be an independent set of users, and try to add more users to it in a greedy manner. The users are first sorted in an ascending order according to the number of packets in their cache. If two users have the same number of packets in their cache, the tie is broken arbitrarily. We initialize $\mathcal{U}$ to be $\{x_0\}$, where x_0 is the first user in the list. Then, we pick the users one by one according to the order specified above. After picking a user, say x, we replace $\mathcal{U}$ by $\mathcal{U} \cup \{x\}$ if $\rho(\mathcal{U} \cup \{x\}) = |\mathcal{U}| + 1$; otherwise leave $\mathcal{U}$ unchanged. As we run the above algorithm, the property that the subset $\mathcal{U}$ of users is independent is kept invariant.

Let the resulting subset be $\mathcal{U}^*$ at the end of the algorithm. It can be proved that the subset $\mathcal{U}^*$ is maximally independent. Indeed, suppose that y is a user not in $\mathcal{U}^*$ and $\mathcal{U}^* \cup \{y\}$ is an independent set of users, then all non-empty subsets of $\mathcal{U}^* \cup \{y\}$

are independent by Lemma 2, but y should have been added to $\mathcal{U}$ when we run the algorithm. This contradicts that $\mathcal{U}^* \cup \{y\}$ is independent.

To illustrate the algorithm, we apply this lower bound to Example 1. We first list the users according to the number of side-information packets. If two users have the same number of packets in the cache, the user with smaller user index is picked first in the algorithm. Hence, we order the users as: user 1, user 2, user 4, user 3, and user 5. The set $\mathcal{U}$ is initialized to be $\{1\}$. Since $\rho(\{1,2\}) = 1$, user 2 is not added to the set $\mathcal{U}$ in the second iteration. We then check that $\rho(\{1,4\}) = 2$, and update $\mathcal{U}$ to {1,4}. Next, we compute $\rho(\{1,4,3\}) = 2$ and see that users 1, 3, and 4 are not independent. Thus user 3 is discarded. Finally, we compute $\rho(\{1,4,5\}) = 3$ and conclude that {1, 4, 5} is a maximally independent set of users. In other words, even if users 1, 4, and 5 are allowed to jointly decode their packets, the source needs to send at least three packets to satisfy the demand of them. So, if no cooperation among the users is allowed, the total number of transmitted packets is also at least equal to three. The completion time in Example 1 is therefore no less than three, and the three transmitted packets shown in Example 1 thus achieve the optimal completion time.

IV. A Heuristic Algorithm Based on Coding Group

We say that a coded packet is *decodable* to a user if the user can decode the requested packet upon receiving this packet. In linear-algebraic terms, a coded packet with coding vector $\boldsymbol{v}$ is decodable to user i if $\boldsymbol{r}_i$ is contained in the subspace spanned by $\boldsymbol{v}$ and the columns of $\boldsymbol{C}_i$. A group of users is said to form a *coding group* if there is a common coded packet which is decodable to all members in this group. In other words, a packet with coding vector $\boldsymbol{v}$ is decodable to users in a group of users $\mathcal{U}$ if for all $i \in \mathcal{U}$, (i) the vector $\boldsymbol{v}$ is contained in the linear span of $\boldsymbol{r}_i$ and the columns of $\boldsymbol{C}_i$, and (ii) $\boldsymbol{v}$ is not contained in the linear span of the columns of $\boldsymbol{C}_i$. Note that when the concept of coding group is applied to the original IC problem with uncoded side information, the coding groups are precisely the cliques in the side-information graph.

By the assumption in (2), there is at least one decodable packet to each individual user. Hence each user forms a coding group of size 1. A coding group is called *maximal* if it is not a subset of a larger coding group. Note that a subset of a coding group is also a coding group. For example, in the example in Fig. 1, {1, 2} and {3, 4} are coding groups, and indeed they are maximal coding groups. In this section, to tackle the GIC problem formulated in Section II, we present a heuristic algorithm based on the notion of coding group. In particular, we propose an algorithm to partition the users into disjoint coding groups in the first subsection, where we assume that there is a fast method to check whether a given subset of users forms a coding group. Such a method together with its efficient encoding algorithm will be presented in the next subsection.

A. Heuristic Algorithm for Solving the Problem in Section II

In this subsection, we first give an algorithm to find a maximal coding group. This algorithm will be used by a recursive algorithm to partition the users into disjoint coding groups. The number of such coding groups is then the number of packets transmissions required by our scheme.

1) Finding a Maximal Coding Group: Given a sender with its coding matrix, a set of users together with their corresponding coding matrices and request vectors, we can obtain a maximal coding group as follows: We first initialize a variable set, $\mathcal{U}$, to be the set of all users and a variable set, $\mathcal{B}$, to be the empty set. Then we remove from $\mathcal{U}$ a user v who has the smallest number of cached packets, and put v into $\mathcal{B}$. We repeatedly remove the user with the smallest number of cached packets from $\mathcal{U}$. Add this user to $\mathcal{B}$ if he/she forms a coding group with the users already in $\mathcal{B}$. Repeat the above steps until $\mathcal{U}$ becomes empty. Upon the termination of the algorithm, the set $\mathcal{B}$ is a maximal coding group.

Note that we add users one by one into $\mathcal{B}$. Intuitively, it is less likely that a user with few cached packets can be added into $\mathcal{B}$, assuming that some users are already in $\mathcal{B}$. To maximize the size of a coding group, it is more reasonable to start from a user with the smallest number of cached packets. That explains why our heuristic algorithm is designed as such.

Algorithm 1 Partition($\mathcal{M}, \boldsymbol{C}_S, \boldsymbol{C}_i, \boldsymbol{r}_i$ for $i \in \mathcal{V}$)

Input: The set of users $\mathcal{M}$, the coding matrix of the sender $\boldsymbol{C}_S$, the coding matrices $\boldsymbol{C}_i$ and the request vector $\boldsymbol{r}_i$ of user $i \in \mathcal{M}$.
Output: A partition of $\mathcal{M}$ into coding groups
1: **if** $\mathcal{M}$ is the empty set **then**
2: return $\emptyset$
3: **end if**
4: Initialize $\mathcal{B} \leftarrow \emptyset, \mathcal{U} \leftarrow \mathcal{M}$.
5: **repeat**
6: Choose $r \in \mathcal{U}$ which has the smallest rank$(\boldsymbol{C}_r)$
7: **if** $\{r\} \cup \mathcal{B}$ form a coding group **then**
8: $\mathcal{B} \leftarrow \mathcal{B} \cup \{r\}$
9: **end if**
10: $\mathcal{U} \leftarrow \mathcal{U} \setminus \{r\}$
11: **until** $\mathcal{U}$ is empty
12: Let $\boldsymbol{v}$ be the coding vector for the coding group $\mathcal{B}$.
13: For user $j \in \mathcal{M} \setminus \mathcal{B}$, augment the coding matrix $\boldsymbol{C}_j$ by the coding vector $\boldsymbol{v}$
14: return $\{\mathcal{B}\} \cup$ Partition1$(\mathcal{M} \setminus \mathcal{B}, \boldsymbol{C}_S, \boldsymbol{C}_j, \boldsymbol{r}_j$ for $j \in \mathcal{M} \setminus \mathcal{B})$

2) Partitioning Users to Several Coding Groups: Given an algorithm for finding a maximal coding group, a recursive algorithm solving the problem posed in Section II can be obtained. We first obtain a maximal coding group $\mathcal{B}$ and let $\boldsymbol{v}$ be the coding vector for this group. Next, we augment the coding matrices of the users in $\mathcal{M} \setminus \mathcal{B}$ by $\boldsymbol{v}$ and repeat the procedure for these users. The pseudo-code for the recursive algorithm is given in Algorithm 1. Line 4 to line 11 in Algorithm 1 is the procedure for finding a maximal coding group. After the "repeat-until" loop, $\mathcal{B}$ is a maximal coding group. Note that the coding matrices of the users who do not belong to $\mathcal{B}$ is

augmented with the coding vector for $\mathcal{B}$, as shown in lines 12 and 13.

B. Characterization and Efficient Encoding for Coding Group

We first re-write the criterion for forming a coding group in a convenient form, and then propose an efficient algorithm for checking this criterion and finding the corresponding encoding vector, which is needed in our heuristic algorithm described in the previous subsection.

Theorem 3: The users in $\mathcal{J} \subseteq \mathcal{M}$ form a coding group if and only if there exists $\boldsymbol{x}_S \in \text{GF}(q)^{K_S}$, $a_i \in \text{GF}(q) \setminus \{0\}$ and $\boldsymbol{x}_i \in \text{GF}(q)^{K_i}$ for $i \in \mathcal{J}$, such that

$$\boldsymbol{C}_S \boldsymbol{x}_S + \boldsymbol{C}_i \boldsymbol{x}_i = a_i \boldsymbol{r}_i \tag{7}$$

holds for all $i \in \mathcal{J}$.

Proof: When the matrix $\boldsymbol{E}$ in (3) is a column vector $\boldsymbol{x}_S$, i.e., when $K_S = 1$, then the condition in (3) can be equivalently formulated as in the theorem. We note that the value of a_i is required to be non-zero, in order to guarantee that the vector $\boldsymbol{r}_i$ is contained in the linear span of the columns of $\boldsymbol{C}_i$ and $\boldsymbol{C}_S$. ■

The above condition amounts to solving a system of linear equations over GF(q), with a_i's and the components of $\boldsymbol{x}_S$ and $\boldsymbol{x}_i$'s as variables, with the subtlety that a_i's are constrained to be non-zero. If the values of a_i's are fixed, then checking whether there is a solution reduces to a standard problem in linear algebra and can be efficiently done. A naive method to determine whether $\mathcal{J}$ is a coding group is to enumerate all $(q-1)^J$ options of a_i's, and for each option check whether (7) holds for all i, where J denotes the size of $\mathcal{J}$, i.e., $J \triangleq |\mathcal{J}|$. Its time complexity, however, grows exponentially with the group size, J. For this reason, we now present an efficient algorithm that can reduce the time complexity to polynomial. Throughout this paper, the complexity is measured in terms of the number of finite-field operations unless otherwise stated.

We define the *left null space* of a matrix $\boldsymbol{M}$ as the set of all column vectors $\boldsymbol{x}$ such that $\boldsymbol{x}^T \boldsymbol{M} = \boldsymbol{0}^T$, where $\boldsymbol{0}$ is the zero column vector. The left null space of $\boldsymbol{M}$ is also called the *orthogonal complement* of span($\boldsymbol{M}$). The *right null space* of a matrix $\boldsymbol{M}$ is the set of all column vectors $\boldsymbol{y}$ such that $\boldsymbol{M}\boldsymbol{y} = \boldsymbol{0}$. Let $x_1, x_2, \ldots, x_{K_S}$ be the components of $\boldsymbol{x}_S$ and let $\boldsymbol{s}_k$ be the k-th column of $\boldsymbol{C}_S$ for $k = 1, 2, \ldots, K_S$.

The proposed algorithm is divided into four steps.

1) *Find a basis for the left null space of $\boldsymbol{C}_i$ for $i \in \mathcal{J}$*: We first augment $\boldsymbol{C}_i$ by adding $N - K_i$ column vectors to its right so that the resultant $N \times N$ matrix, denoted by $\bar{\boldsymbol{C}}_i$, is of full rank. We then compute the inverse of $\bar{\boldsymbol{C}}_i$. By removing the top K_i rows in $\bar{\boldsymbol{C}}_i^{-1}$, we obtain the $(N - K_i) \times N$ matrix $\boldsymbol{B}_i$. By the definition of matrix inverse, $\boldsymbol{B}_i \boldsymbol{C}_i$ is a zero matrix, and thus the rows of $\boldsymbol{B}_i$ form a basis of the left null space of $\boldsymbol{C}_i$.
2) *Express a_i as a function of $\boldsymbol{x}_S$*: Let $\boldsymbol{d}_i$ be the column vector $(x_1, x_2, \ldots, x_{K_S}, -a_i)$. According to (7), we want to find $\boldsymbol{d}_i$ such that $[\boldsymbol{C}_S \ \boldsymbol{r}_i]\boldsymbol{d}_i$ is in span($\boldsymbol{C}_i$), which can also be expressed as

$$\boldsymbol{B}_i [\boldsymbol{C}_S \ \boldsymbol{r}_i] \boldsymbol{d}_i = \boldsymbol{0}. \tag{8}$$

According to the assumption in (1), $\boldsymbol{B}_i \boldsymbol{r}_i$ cannot be the zero vector. We hence can row-reduce $\boldsymbol{B}_i[\boldsymbol{C}_S \ \boldsymbol{r}_i]$ to a matrix, denoted by $\boldsymbol{E}_i$, whose last column is $(1, 0, 0, \ldots, 0)$, i.e., the top-right corner of $\boldsymbol{E}_i$ is 1, and the entries below it are all zero. From the first row of $\boldsymbol{E}_i$, we can then express a_i as a linear function of $\boldsymbol{x}_S$ as

$$a_i = \boldsymbol{h}_i^T \boldsymbol{x}_S, \tag{9}$$

where $[\boldsymbol{h}_i^T \ 1]$ is the first row of $\boldsymbol{E}_i$.

For example, if $\boldsymbol{B}_i[\boldsymbol{C}_S \ \boldsymbol{r}_i]$ can be row-reduced to

$$\boldsymbol{E}_i = \begin{bmatrix} 0 & 1 & 2 & 3 & 1 \\ 2 & 1 & 3 & 1 & 0 \\ 2 & 0 & 3 & 2 & 0 \end{bmatrix}, \tag{10}$$

then $a_i = x_2 + 2x_3 + 3x_4$.

3) *Characterize the potential solution space of $\boldsymbol{x}_S$*: Let $\boldsymbol{E}_i^-$ be the matrix obtained by removing the first row and the last column of $\boldsymbol{E}_i$. For example, from the matrix $\boldsymbol{E}_i$ in (10), we have

$$\boldsymbol{E}_i^- = \begin{bmatrix} 2 & 1 & 3 & 1 \\ 2 & 0 & 3 & 2 \end{bmatrix}. \tag{11}$$

According to (8), any vector $\boldsymbol{x}_S$ satisfying $\boldsymbol{E}_i^- \boldsymbol{x}_S = \boldsymbol{0}$ and $a_i = \boldsymbol{h}_i^T \boldsymbol{x}_S \neq 0$ can satisfy the demand of user i. Hence, the right null space of $\boldsymbol{E}_i^-$ defines a set of potential solutions for user i.

Consider the users in $\mathcal{J}$ together. Let $\boldsymbol{E}^-$ denote the concatenated matrix formed by stacking $\boldsymbol{E}_i^-$ for all $i \in \mathcal{J}$ vertically. The potential solution space of $\boldsymbol{x}_S$, denoted by $\mathcal{W}$, is the right null space of $\boldsymbol{E}^-$. If $\mathcal{W}$ contains only the zero vector, then $\mathcal{J}$ is not a coding group since in this case $\boldsymbol{x}_S = \boldsymbol{0}$ and hence $a_i = 0$ for all i, which violates the condition in Theorem 3.

Suppose $\mathcal{W}$ contains some nonzero vectors. Let $D > 0$ be the dimension of $\mathcal{W}$. A vector $\boldsymbol{x}_S$ in $\mathcal{W}$ can be expressed as

$$\boldsymbol{x}_S = \boldsymbol{G}\boldsymbol{y}, \tag{12}$$

for some $K_S \times D$ matrix $\boldsymbol{G}$ and column vector $\boldsymbol{y}$ in GF(q)^D. We remark that when $\boldsymbol{E}^-$ is a degenerate empty matrix, we let $\boldsymbol{G}$ be the $K_S \times K_S$ identity matrix. Let $\boldsymbol{H}$ be the $K_S \times J$ matrix which has $\boldsymbol{h}_i$'s as its columns. According to (9) and Theorem 3, a vector $\boldsymbol{x}_S \in \mathcal{W}$ is a solution if

$$\boldsymbol{H}^T \boldsymbol{x}_S \tag{13}$$

has no zero entries.

4) *Solve for coefficients a_i*: Let $\boldsymbol{a}$ be the J-dimensional column vector whose components are a_i for $i \in \mathcal{J}$, and let $\boldsymbol{F}$ be the $J \times D$ matrix $\boldsymbol{H}^T \boldsymbol{G}$. We need to find $\boldsymbol{y} \in$ GF(q)^D such that

$$\boldsymbol{a} = \boldsymbol{F}\boldsymbol{y} = \boldsymbol{H}^T \boldsymbol{G}\boldsymbol{y} \tag{14}$$

has no zero entries. There are three cases to be considered:

Case 1 ($q = 2$): If the binary field is used, i.e., $q = 2$, then the only choice for a_i is $a_i = 1$ for all $i \in \mathcal{J}$. Determining whether $\mathcal{J}$ forms a coding group then amounts to check whether a system of linear equations (as shown in (14) by setting $\boldsymbol{a}$ to be the all-one vector) has solutions over GF(2).

Case 2 ($q \geq J$): The situation is more involved for a general field size. We only need to consider the case where $\boldsymbol{F}$ has no zero rows, for otherwise there cannot be a solution and $\mathcal{J}$ cannot be a coding group. The following result shows that a solution can always be found if $q \geq J$:

Theorem 4: If $q \geq J$ and $\boldsymbol{F}$ has no zero rows, then there exists $\boldsymbol{y} \in \text{GF}(q)^D$ such that $\boldsymbol{Fy}$ has no zero entries.

Proof: Let $\boldsymbol{f}_i^T$ denote the i-th row of $\boldsymbol{F}$, for $i = 1, 2, \ldots, J$. For each i, as $\boldsymbol{f}_i$ is not zero, there are exactly q^{D-1} choices of $\boldsymbol{y}$ in $\text{GF}(q)^D$ such that $\boldsymbol{f}_i^T\boldsymbol{y}$ is zero. If $J = 1$, the theorem is obviously true. Suppose that $J \geq 2$. We count the number of $\boldsymbol{y}$ such that $\boldsymbol{y} \neq \boldsymbol{0}$ and $\boldsymbol{f}_i^T\boldsymbol{y} = 0$ for at least one i. By the union bound, we obtain

$$\begin{aligned}&\left|\left\{\boldsymbol{y} \in \text{GF}(q)^D : \boldsymbol{y} \neq \boldsymbol{0} \text{ and } \exists i,\ \boldsymbol{f}_i^T\boldsymbol{y} = 0\right\}\right| \\ &\quad \leq \sum_{i \in \mathcal{J}} \left|\left\{\boldsymbol{y} \in \text{GF}(q)^D : \boldsymbol{y} \neq \boldsymbol{0} \text{ and } \boldsymbol{f}_i^T\boldsymbol{y} = 0\right\}\right| \\ &\quad = J(q^{D-1} - 1) \leq q^D - 2 < q^D.\end{aligned}$$

The first inequality in the previous line follows from the hypothesis that $2 \leq J \leq q$. Therefore, we can find at least one vector $\boldsymbol{y}$ in $\text{GF}(q)^D$ such that $\boldsymbol{f}_i^T\boldsymbol{y}$ is nonzero for all $i \in \mathcal{J}$. ■

Example 2: Suppose $M = N = 3$, $q = 3$, $\boldsymbol{r}_i = \boldsymbol{e}_i$ for $i = 1$, 2, 3, and

$$\boldsymbol{C}_S = \begin{bmatrix}1 & 1\\ 1 & 2\\ 0 & 0\end{bmatrix},\ \boldsymbol{C}_1 = \begin{bmatrix}1 & 0\\ 0 & 1\\ 2 & 2\end{bmatrix},\ \boldsymbol{C}_2 = \begin{bmatrix}1\\ 1\\ 0\end{bmatrix},\ \boldsymbol{C}_3 = \begin{bmatrix}1 & 1\\ 0 & 2\\ 2 & 2\end{bmatrix}.$$

We want to check whether users 1 to 3 form a coding group. We first find bases for the left null spaces of the three coding matrices:

$$\boldsymbol{B}_1 = [1 \quad 1 \quad 1],\ \boldsymbol{B}_2 = \begin{bmatrix}2 & 1 & 0\\ 0 & 0 & 1\end{bmatrix},\ \boldsymbol{B}_3 = [1 \quad 0 \quad 1].$$

In step 2, we write

$$\begin{aligned}&[1 \quad 1 \quad 1]\begin{bmatrix}1 & 1 & 1\\ 1 & 2 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ -a_1\end{bmatrix} = 0,\\ &\begin{bmatrix}2 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 1 & 0\\ 1 & 2 & 1\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ -a_2\end{bmatrix} = \begin{bmatrix}0\\ 0\end{bmatrix},\\ &[1 \quad 0 \quad 1]\begin{bmatrix}1 & 1 & 0\\ 1 & 2 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ -a_3\end{bmatrix} = 0.\end{aligned}$$

After some row reductions, we obtain

$$\boldsymbol{E}_1 = [2 \quad 0 \quad 1],\ \boldsymbol{E}_2 = \begin{bmatrix}0 & 1 & 1\\ 0 & 0 & 0\end{bmatrix},\ \boldsymbol{E}_3 = [1 \quad 1 \quad 1],$$

and hence

$$a_1 = 2x_1, \tag{15}$$

$$a_2 = x_2, \tag{16}$$

$$a_3 = x_1 + x_2. \tag{17}$$

Then we characterize the potential solution space by stacking all the $\boldsymbol{E}_j^-$'s. It happens that all of them are degenerate empty matrices, so the potential solution space, $\mathcal{W}$, is $\text{GF}(3)^2$ with dimension D equal to 2. $\boldsymbol{G}$ can be chosen as the 2×2 identity matrix.

It remains to solve for $\boldsymbol{a}$. Combining (15), (16), and (17), we obtain

$$\begin{bmatrix}a_1\\ a_2\\ a_3\end{bmatrix} = x_1\begin{bmatrix}2\\ 0\\ 1\end{bmatrix} + x_2\begin{bmatrix}0\\ 1\\ 1\end{bmatrix}.$$

Since $q = J = 3$ in this example, non-zero solutions to a_i's always exist. For example, we may choose $x_1 = x_2 = 1$. Then we have $(a_1, a_2, a_3) = (2, 1, 2)$ and $\boldsymbol{x}_S = (x_1, x_2) = (1, 1)$.

While Theorem 4 guarantees the existence of a solution when $q \geq J$, we still need a systematic way to find it. In the following, we present an algorithm to find such a solution. For ease of presentation, we let f_{id} be the (i, d)-entry of $\boldsymbol{F}$ and $f_i(\boldsymbol{y})$ be the i-th component of $\boldsymbol{Fy}$. In the following, we suppose that $q \geq J$.

For $d = 1, 2, \ldots, D$, define

$$\mathcal{I}_d \triangleq \{i : f_{id} \neq 0\}$$

to be the set comprising the indices of the non-zero entries in the d-th column of $\boldsymbol{F}$. Since $\boldsymbol{F}$ has no zero rows, we have $\cup_{d=1}^{D}\mathcal{I}_d = \mathcal{J}$. We want to find $\tilde{\boldsymbol{y}} \triangleq (\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_D)$ such that $\boldsymbol{F}\tilde{\boldsymbol{y}}$ has no zero entries. We distinguish two cases:

a) There exists d such that $\mathcal{I}_d = \mathcal{J}$. We can simply choose $\tilde{y}_d = 1$ and $\tilde{y}_g = 0$ for all $g \neq d$.

b) $|\mathcal{I}_d| < J$ for all d. We determine the values of $\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_D$ iteratively one by one. Suppose we have already determined $\tilde{y}_d$ for $d = 1, 2, \ldots, n-1$. Note that

$$f_i(\tilde{y}_1, \ldots, \tilde{y}_{n-1}, y_n, 0, \ldots, 0) = 0 \tag{18}$$

is a linear equation with respect to a single variable y_n for $i \in \mathcal{I}_n$. It thus has only one solution. Since $|\mathcal{I}_n| < J \leq q$, the number of elements in $\text{GF}(q)$ which satisfy (18) for some $i \in \mathcal{I}_n$ is strictly less than q. We hence can choose $\tilde{y}_n$ such that $f_i(\tilde{y}_1, \ldots, \tilde{y}_{n-1}, \tilde{y}_n, 0, \ldots, 0) \neq 0$ for all $i \in \mathcal{I}_n$. After $\tilde{y}_d$ for all d have been determined, we can guarantee that $f_i(\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_D) \neq 0$ for all $i \in \mathcal{J}$.

After running the foregoing algorithm, if the users in $\mathcal{J}$ form a coding group, the vector $\boldsymbol{x}_S$ can be obtained by substituting $\tilde{\boldsymbol{y}}$ into (12).

Case 3 ($2 < q < J$): For $2 < q < J$, we may solve for the coefficients a_i's by exhaustive search over all combinations of $(a_1, \ldots, a_J)$'s, or solving a system of multi-variable polynomials by Groebner basis. An illustration of the method of Groebner basis is given in the Appendix.

Remark:

1) After receiving the broadcast packet, each user in $\mathcal{J}$ can then decode the requested packet by Gaussian elimination, which has complexity $O(N^3)$.
2) For engineering purpose, the case $q = 2$ plays a special role, as operations over GF(2) can be done in a fast manner. If transmission efficiency is important, one may choose a larger field size, e.g. $q \geq M$, so that the chance that a set of users forms a coding group will be higher. For the rest of this paper, we assume either $q = 2$ or $q \geq M$.

V. Complexity Analysis

In this section, we analyze the time complexity of the procedure of checking whether a given set of users forms a coding group, followed by the analysis of the overall heuristic algorithm. Lastly, we analyze the time complexity of the heuristic lower bound.

A. Checking Coding Group—An Intermediate Step

We now analyze the time complexity of checking whether a given subset of users, $\mathcal{J}$, forms a coding group. In the procedure described in the previous section, there are four steps. The analyses of the first three steps are relatively straightforward and are shown below:

- Step 1 requires transforming $\bar{\boldsymbol{C}}_i$ for all $i \in \mathcal{J}$ to row reduced echelon form (RREF) and hence has time complexity $O(JN^3)$.
- Step 2 requires first computing $\boldsymbol{E}_i$ and then reducing the last column of $\boldsymbol{E}_i$ to $(1, 0, 0, \ldots, 0)$ for all $i \in \mathcal{J}$. The major computation burden lies in matrix multiplication and hence has complexity $O(JN^3)$.
- In Step 3, to characterize $\mathcal{W}$, we need to transform $\boldsymbol{E}^-$ into RREF. To do that, we reduce the concatenated matrix $\boldsymbol{E}^-$ row by row. Since there are at most JN rows, the complexity is $O(JN \times K_S^2) = O(JN^3)$.

The analysis for Step 4 is divided into two cases, namely $q = 2$ and $q \geq J$:

Case 1 ($q = 2$): We can set $a_i = 1$ for all $i \in \mathcal{J}$ and then solve the $J \times D$ system of linear equations. If $J \leq D$, there will be J^2 row operations. Each row operation requires $O(D)$ steps. Therefore, the complexity is $O(J^2D)$. If $J > D$, there will be JD row operations, since there are at most D pivots. Again, each row operation requires $O(D)$ steps. Therefore, the complexity is $O(JD^2)$. Combining the two results, the complexity is $O(\min\{J, D\}JD) = O(\min\{J, N\}JN)$.

Case 2 ($q \geq J$): We run the iterative procedure described in the previous section, which involves D iterations. During iteration n, an element in GF(q) that is not a solution to any of the linear (18) for $i \in \mathcal{I}_n$ needs to be determined. We can first find the root to $f_i = 0$ as shown in (18) for each $i \in \mathcal{J}$. There are at most J distinct values. We mark them as "1" in a binary array of length q. We then choose a symbol from GF(q) which is different from all those J values. This can be done by finding the first zero entry of the binary array, which requires examining at most J entries of the array. The time complexity is therefore $O(J)$. Totally, there are D iterations. The total complexity is $O(JD) = O(JN)$.

Summarizing the above four steps, we obtain that the overall complexity of determining whether a group of J users forms a coding group is $O(JN^3)$ for both $q = 2$ and $q \geq J$. For $2 < q < J$, it is not known whether $\tilde{\boldsymbol{y}}$ can be determined in polynomial time.

B. Heuristic Algorithm With Incremental Implementation

Now we analyze the time complexity of the whole heuristic algorithm, as stated in Algorithm 1. Note that the repeat-until loop needs to be executed $O(M^2)$ times. A naive implementation of repeatedly checking the coding group would require a time complexity of $O(M^3N^3)$. In the following, we analyze its time complexity by incrementally executing the algorithm, which has lower time complexity than the naive implementation.

We claim that lines 6 to 10 in the repeat-until loop can be carried out incrementally. To do that, assume a temporary coding group $\mathcal{B}$ has been formed, the corresponding $\boldsymbol{E}^-$ is already in RREF, and $\mathcal{U}$ is as defined in the algorithm. We now test whether user $u \in \mathcal{U}$ can join $\mathcal{B}$. We perform the procedure described in the previous section from step 1 through the middle of step 3 on user u to obtain $\boldsymbol{E}_u^-$. We then construct matrix $\boldsymbol{A}$ by concatenating $\boldsymbol{E}^-$ with $\boldsymbol{E}_u^-$. We then perform elementary row operations on $\boldsymbol{A}$ to transform it into RREF through row reducing the rows of $\boldsymbol{E}_u^-$. We then try to determine $\boldsymbol{y}$ which satisfies the condition in (14). If it can be found, we update $\boldsymbol{E}^-$ to be the RREF resultant of $\boldsymbol{A}$. Otherwise, we test another user in $\mathcal{U}$. All the above process is implemented incrementally except the process of determining $\boldsymbol{y}$. In other words, only Step 4 requires $O(M^2)$ executions while the other steps require only $O(M)$ executions.

Based on the analysis in the previous subsection, we can see that for $q = 2$, the time complexity of Algorithm 1 is $O(M^2N^3 + \min\{M, N\}M^3N)$, which can be simplified as $O(\max\{M, N\}M^2N^2)$. For $q \geq J$, the time complexity is $O(M^2N^3 + M^3N) = O((N^2 + M)M^2N)$. It is interesting to note that while the complexities in checking whether a given set of users forms a coding group for $q = 2$ and $q \geq J$ are the same, Algorithm 1 has different complexities for the two cases. For example, when M grows quadratically with N, Algorithm 1 has complexities $O(M^4)$ and $O(M^{3.5})$ for $q = 2$ and $q \geq J$, respectively.

C. Heuristic Lower Bound With Incremental Implementation

Now we analyze the time complexity of the heuristic lower bound as stated in the end of Section III.

In the first step, the users are sorted and the time complexity is $O(M \log M)$. In the second step, a maximal independent set is obtained. To do this, we add users one by one to $\mathcal{U}$. Before adding a user to $\mathcal{U}$, $\rho(\mathcal{U})$ needs to be computed, which involves the computation of the ranks of $(\boldsymbol{R}(\mathcal{U}), \boldsymbol{C}(\mathcal{U}))$ and $\boldsymbol{C}(\mathcal{U})$. Recall that computing the rank of an $N \times K$ matrix can be done by Gaussian elimination, which has time complexity

$O(\min(K,N)KN)$. Since $\boldsymbol{C}(\mathcal{U})$ has N rows and at most $|\mathcal{U}|N$ columns, computing its rank has complexity $O(|\mathcal{U}|N^3)$, or simply $O(MN^3)$. Therefore, a naive implementation of repeatedly checking the above criterion would require a time complexity of $O(M^2N^3)$. The overall time complexity is $O(M\log M + M^2N^3) = O(M^2N^3)$.

In the following, similar to the case in the previous subsection, we consider the computation of the heuristic lower bound by incrementally executing the rank checking operation, which has lower time complexity than the naive implementation.

We claim that the rank computation for adding an element to $\mathcal{U}$ can be carried out incrementally. We illustrate this by taking the computation of rank$(\boldsymbol{C}(\mathcal{U}))$ as an example. Initially, $|\mathcal{U}| = 1$. We use Gaussian elimination to transform $\boldsymbol{C}(\mathcal{U})$ to column echelon form (CEF), which has time complexity $O(N^3)$. Let $\boldsymbol{T}$ denote the resultant matrix, which is a lower triangular matrix. If we add another element, say x, into $\mathcal{U}$, we need to compute rank$(\boldsymbol{T}, \boldsymbol{C}_x)$, which can be done by transforming it to CEF. Since $\boldsymbol{T}$ is already in CEF, we only need to perform column reduction on $\boldsymbol{C}_x$. This again has complexity $O(N^3)$. The above procedure is repeated until $\mathcal{U}$ is a maximal set of independent users and the corresponding time complexity is then $O(MN^3)$. Together with the sorting in the first step, the overall time complexity of computing the heuristic lower bound is $O(M\log M + MN^3)$.

VI. Numerical Studies

As Theorem 4 shows, a sufficient condition to utilize our proposed heuristic algorithm is $q \geq M$. In this section, given the number of users M, to utilize the algorithm, we define such a q that is slightly larger than M by $q_M \triangleq 2^{\lceil \log_2(M) \rceil}$, where $\lceil x \rceil$ denotes the smallest integer greater than or equal to x. To show the advantage of our proposed scheme, we perform two sets of numerical studies, which are illustrated in the next two subsections, respectively. The first merely focuses on the probability of forming coding groups while the second focuses on the number of transmissions to satisfy the demand of a group of users.

We assume the sender possesses all the uncoded packets, i.e., $\boldsymbol{C}_S$ is equal to the identity matrix, while the users' cached packets are obtained at random. To generate $\boldsymbol{C}_i$, we first pick a random number K_i between 0 and $N-1$. Let $\boldsymbol{C}_i$ be chosen uniformly at random over all full-rank $N \times K_i$ matrices over GF(q). Two cases are considered: $q = 2$ and $q = q_M$.

A. Probability of Forming Coding Groups

We vary N from 4 to 32 with interval 4. For each N, we vary M from 2 to N, with the corresponding value of q_M calculated as described in the first paragraph of this section. We set $\boldsymbol{r}_i = \boldsymbol{e}_i$ (the j-th standard basis vector) for all $i \in \mathcal{M}$. For each pair of (N, M), we check whether all the M users form a coding group and compare the corresponding probability, denoted by p_q for $q = 2$ or $q = q_M$, by averaging over 1000 random $\boldsymbol{C}_i$'s realizations. The obtained simulation data is shown in Table I, where the last column shows the ratio of probabilities of forming coding groups between $q = 2$ and $q = q_M$. We can observe that for each pair of (N, M), increasing the field size q from 2 to q_M indeed reaps higher probability of forming coding groups, since the last column is always greater than 1. Note that there are some blank entries in the last column, since in these cases, the ratio is infinity. This also applies to the blank rows for certain combinations of (N, M), say (8,7) and (8,8). Those blank rows are not shown for clarity.

TABLE I
Probability of Forming Coding Groups for $q = 2$ and $q = q_M$

N	M	q_M	p_2	p_{q_M}	$\frac{p_{q_M}}{p_2}$
4	3	4	0.187	0.248	1.33
4	4	4	0.045	0.099	2.2
8	3	4	0.150	0.193	1.29
8	4	4	0.030	0.054	1.8
8	5	4	0.003	0.022	7.33
8	6	8	0.002	0.006	3
12	3	4	0.150	0.198	1.32
12	4	4	0.024	0.057	2.38
12	5	8	0.006	0.014	2.33
12	6	8	0	0.004	
16	3	4	0.143	0.178	1.24
16	4	4	0.035	0.049	1.4
16	5	8	0.005	0.011	2.2
16	6	8	0.001	0.002	2
20	3	4	0.132	0.170	1.29
20	4	4	0.030	0.048	1.6
20	5	8	0.006	0.010	1.67
20	6	8	0.001	0.002	2
24	3	4	0.160	0.186	1.16
24	4	4	0.034	0.050	1.47
24	5	8	0.008	0.012	1.5
24	6	8	0.004	0.005	1.25
28	3	4	0.158	0.178	1.13
28	4	4	0.039	0.047	1.21
28	5	8	0.003	0.008	2.67
28	6	8	0.001	0.002	2
32	3	4	0.160	0.177	1.11
32	4	4	0.038	0.051	1.34
32	5	8	0.006	0.013	2.17
32	6	8	0.000	0.001	

B. Number of Transmissions

The setting is identical to the previous subsection except that we fix $M = N$ and use our proposed Algorithm 1 to the M users. By averaging over 1000 random $\boldsymbol{C}_i$'s realizations, the average number of transmissions obtained by Algorithm 1 and the lower bound in Section III for $q = 2$ and $q = q_M$ is shown in Fig. 2.

Fig. 2 shows that increasing the finite field size from 2 to q_M reduces the number of transmissions. The curves of both Algorithm 1 and the lower bound in Section III grows roughly linearly with M, though with different slopes.

VII. Application in BRC

In this section, we consider an application scenario, in which our results can be applied. Nowadays, popular live online video has emerged. The delivery of popular video clips often consumes a lot of bandwidth. Normally, data blocks for these video clips reach a user via a series of intermediate relay nodes. If intermediate relay nodes cache the data blocks, then they may use the cached data blocks to serve other nearby users with less taken-up resource [31]–[35]. The broadcast relay channel

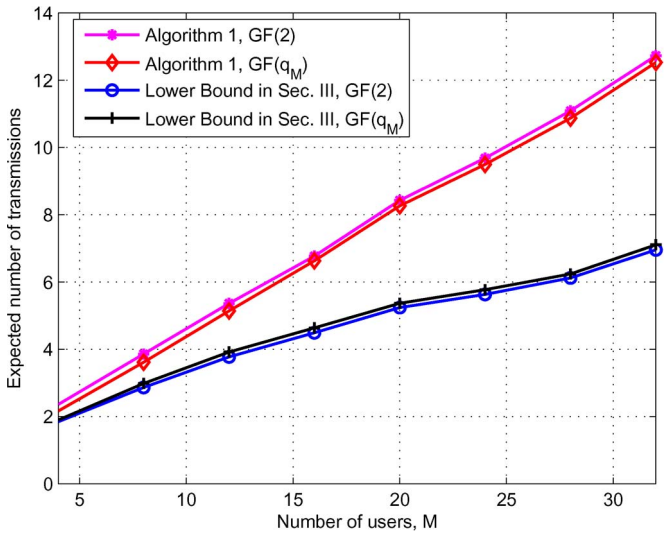

Fig. 2. Average No. of transmissions for $q = 2$ or $q = q_M$.

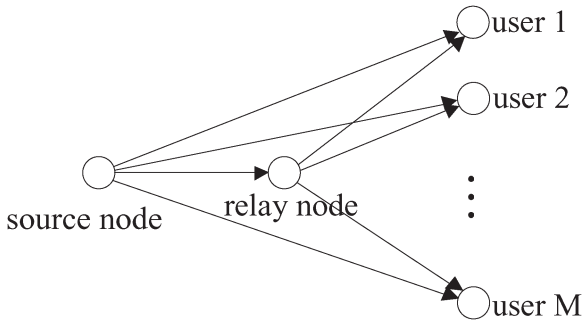

Fig. 3. Diagram of the BRC model.

(BRC), as shown in Fig. 3, serves as a basic building block for such an application scenario. The BRC is a kind of dual to the multiple-access relay channel (MARC), in which the directions of data transmissions are reversed. The work in this section is closely related to our earlier work in [36]. For more details on the MARC channel, we refer the readers to [37], which discusses optimal resource allocation methods and capacities for MARC.

In this section, we apply our Algorithm 1 to the relay-destinations hop of the BRC with practical packet erasure assumption. This setting can be easily extended to the application of user-defined two-step collaborative files downloading [38] among adjacent users: In the first step, the users download the files from an AP under packet erasures. In the second step, under the environment of no packet erasure, the users share information among themselves with each of them serving as the transmitter according to certain scheduling algorithm.

A. Problem Setting

As shown in Fig. 3, there are one source node, one relay node, and M users in BRC. We assume that the relay is close to the users and far away from the source, so that we can assume that all relay-user links are erasure-free. The source-relay link and all source-user links experience independent packet erasures (across links and over time) with the same erasure probability $\epsilon \in (0, 1)$. Initially, the source holds N packets $P_1, \ldots, P_N$, while the relay and the users have no packet in their cache. The number of users, M, is equal to the number of packets, N. Similar to Section VI, we also define such a q that is slightly larger than M by $q_M \triangleq 2^{\lceil \log_2(M) \rceil}$. We assume that user i wants to download packet P_i from the source with the help of the relay, $i \in \mathcal{M}$. In other words, request vector $\boldsymbol{r}_i = \boldsymbol{e}_i$ for all $i \in \mathcal{M}$.

B. Description of All Schemes and the Lower Bound

We compare the performance of three schemes (one uncoded and two coded) and a lower bound for one of the coded scheme. The two coded schemes are both based on Algorithm 1.

All the three schemes operate in phases. In phase I of all schemes, the source transmits all the uncoded packets one by one. In other words, phase I consists of N transmissions. Since there may be packet erasures, in general, we need more phases to complete the whole transmissions. Consider the ***uncoded scheme*** first. It operates in three phases. In phase II, the source simply repeatedly transmits the required packets until it receives an ACK for each packet from either the target user or the relay. In phase III, if any user has not obtained its requested packet, the relay will transmit it to him/her. For this simple uncoded scheme, analytical expression for its expected transmission time can be obtained. Since the transmissions for different users are independent, it suffices to consider the transmission for one user. Let X be the random variable representing the number of transmissions for one user. We have the following equation:

$$E[X] = 1 + \epsilon \left[(1 - \epsilon) + \epsilon E[X] \right]. \tag{19}$$

There is a probability of ϵ that the user is unable to receive the packet in the first transmission and requires retransmissions. If the relay has received the packet in the first transmission (with probability $(1 - \epsilon)$), then it takes one retransmission to complete the process. Otherwise, both the relay and the user have not received it, and the whole process needs to start again, meaning that, another $E[X]$ retransmissions are needed. Solving the above equation for $E[X]$ and multiplying it by the number of users, M, we obtain

$$ME[X] = M \frac{1 + \epsilon - \epsilon^2}{1 - \epsilon^2}. \tag{20}$$

For both coded schemes, the transmission is divided into three phases: As described before, the source transmits all the uncoded packets one by one during phase I. Then phase II starts by letting the source retransmit coded packets until the relay can take over the source's role, i.e., until condition (2) is satisfied where $\boldsymbol{C}_S$ is substituted by the coding matrix of the relay, which is denoted by $\boldsymbol{C}_R$. In this case, phase III starts by letting the relay serve as the sender, apply Algorithm 1, and send the corresponding coded packets for each obtained coding group. The difference between these two coded schemes lies in phase II. In the following, we describe the operation in phase II for each coded scheme.

In the ***first coded scheme***, called ***Scheme A***, the source linearly mixes the requested packets in a random manner in phase II. More specifically, each coefficient is chosen uniformly at random from GF(q). The source then sends the resultant coded packets.

In the ***second coded scheme***, called ***Scheme B***, the source computes maximal coding group according to lines 5 to 11 of Algorithm 1 with a slight modification. We replace line 6 by the following: "Choose $r \in \mathcal{U}$ which has the *largest* rank($[\boldsymbol{C}_r\ \boldsymbol{C}_R]$)." Note that there are two changes. First, $\boldsymbol{C}_r$ is replaced by $[\boldsymbol{C}_r\ \boldsymbol{C}_R]$. It means that we hypothetically let all users have the coded packets in the relay, and try to satisfy these hypothetic users. Note that when they are satisfied, the relay can take over the source's role and starts phase III. Second, when picking users to form a coding group, we start from the one who has the largest rank rather than the smallest rank, since this heuristic will potentially find a coding group with a larger cardinality and the transmission process may enter phase III sooner.

We also compare the schemes with a ***lower bound for Scheme B***. To obtain this lower bound, we proceed in the same manner as in the first and second phases of Scheme B. After the second phase, we compute the lower bound on the number of remaining retransmissions as in Section III. Then we add the number of packet retransmissions in the second phase with the value of the lower bound. In the figures to be presented in the next subsection, we simply label its performance curves by "Lower Bound".

C. Performance Criterion for Comparison

Let T be the total number of *retransmissions*. In other words, T represents the number of transmissions after phase I. We want to compare the expected number of retransmissions, $E[T]$, of the above three transmission schemes and the lower bound for $q = 2$ and $q = q_M$.

For the uncoded scheme, we have a closed-form expression for its expected number of retransmissions:

$$E[T] = ME[X] - N \tag{21}$$

$$= M\frac{1+\epsilon-\epsilon^2}{1-\epsilon^2} - M \tag{22}$$

$$= \frac{M\epsilon}{1-\epsilon^2}, \tag{23}$$

where (22) is obtained from (20).

D. Numerical Comparison

We set $M = 16$ and obtain Fig. 4. For the uncoded scheme, each data point is obtained analytically according to (23). For all other schemes, each data point is obtained by averaging over 1000 random source to users/relay channel realizations.

We can see that Scheme B performs much better than the uncoded scheme. The largest improvement is obtained when $\epsilon = 0.6$, at which the percentage reduction in T is almost 50%. Besides, the coding operation in phase II is essential, since Scheme A performs even worse than the uncoded scheme when

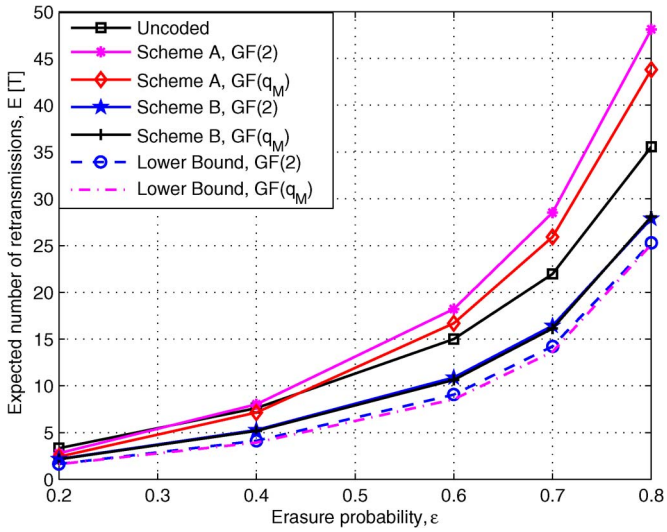

Fig. 4. Comparison of three schemes and the lower bound in the broadcast relay channel in terms of the expected number of retransmission.

$\epsilon \geq 0.5$. Using a larger field size, Scheme B can have better performance, but the gain is insignificant. The main reason is that the number of users is too small and the gain cannot be demonstrated.

Since it is very time consuming to simulate a system of large value of M. We use a trick to get a rough idea of the performance difference between Scheme B with $q = 2$ and $q = q_M$. To do that, we use phase I merely to generate the coding matrices of each user. We always assume that all users are not satisfied after phase I. In other words, for $i = 1, 2, \ldots, N$, packet i can only possibly be received by users other than user i. That means after phase I, we still have M users in the system. With this modification, we have

$$E[T] = M\left[(1-\epsilon) + \epsilon E[X]\right] \tag{24}$$

$$= M\left[(1-\epsilon) + \epsilon\frac{1+\epsilon-\epsilon^2}{1-\epsilon^2}\right] \tag{25}$$

$$= \frac{M}{1-\epsilon^2}, \tag{26}$$

where (24) is due to the fact that there are always M unsatisfied users and for each user, it takes one retransmission if the relay has received his requested packet and $E[X]$ retransmissions, on average, if not. The performance results for the three schemes and the lower bound are plotted in Fig. 5. The phenomenon is similar to that in the previous figure except that the gap between $q = 2$ and $q = q_M$ for Scheme B becomes larger and the gap between the uncoded scheme and Scheme B also becomes larger, both due to the fact that more users create more coding opportunities.

Both Figs. 4 and 5 indicate that the gap between Scheme B and the lower bound is narrow. This shows that the room for improving phase III of Scheme B is small.

VIII. Conclusion and Future Direction

We investigate linear NC construction for the GIC problem. We derive the criterion of forming a coding group, and propose

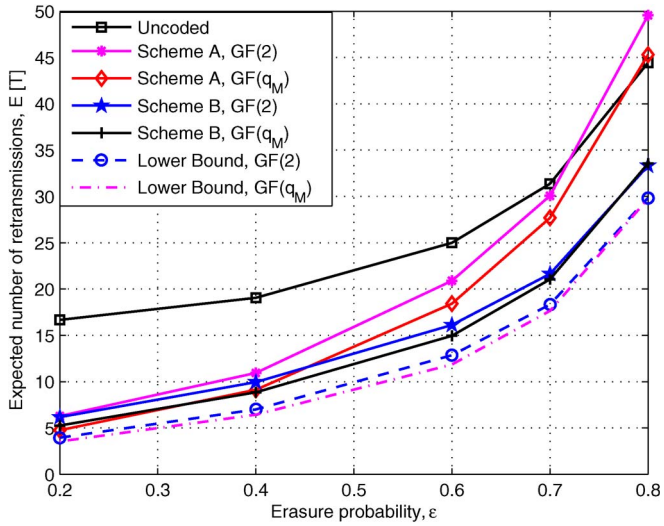

Fig. 5. Comparison of three schemes and the lower bound in the broadcast relay channel in terms of the expected number of retransmission (All users in Phase I are not satisfied).

an efficient algorithm for the corresponding checking and encoding. We further propose a heuristic method to partition the users into disjoint coding groups and a simple lower bound on the number of transmissions required. Numerical studies show three facts: First, large field size outperforms small field size in general for the coded schemes. Second, the performance of the optimal scheme is bounded between two functions which grow roughly linearly with the number of users. Third, for BRC the scheme with coded retransmission outperforms the uncoded one significantly.

In the future, we may consider extending this GIC problem to retransmission with delay constraint, and each user may request multiple packets. Note that there is a tradeoff between transmission efficiency and delay (Sometimes the received packets should be always useful at their reception instant) [25].

Appendix A
Solving System of Inequations by Groebner Basis

In the fourth step in the algorithm for determining a coding group in Section IV, we need to find a vector $(y_1, y_2, \ldots, y_D)$ in GF$(q)^D$ satisfying

$$f_i(y_1, y_2, \ldots, y_D) \neq 0$$

for $i = 1, 2, \ldots, J$, where f_i is a polynomial over GF(q) in $y_1, \ldots, y_D$ of degree 1. We will call it a *system of inequations*. This system of inequations can be solved efficiently by the method given in Section IV when $q = 2$ or when $q \geq J$. In this appendix, we outline the method of elimination using Groebner basis for general field size q through an explicit example.

Consider the following system of six inequations in variables $\boldsymbol{y} = (y_1, y_2, y_3)$.

$$y_2 + 2y_3 \neq 0 \tag{27}$$
$$y_2 + 4y_3 \neq 0 \tag{28}$$
$$y_1 + 2y_2 \neq 0 \tag{29}$$
$$y_1 + 3y_3 \neq 0 \tag{30}$$
$$4y_1 + y_2 + 2y_3 \neq 0 \tag{31}$$
$$3y_1 + 2y_2 \neq 0. \tag{32}$$

The coefficients are elements in GF(5). Since the number of inequations is larger than the field size, Theorem 4 does not apply. We use the following two facts. Firstly, in a finite field of size q, a field element α is nonzero if and only if $\alpha^{q-1} = 1$. Hence, an inequation $f_j(\boldsymbol{y}) \neq 0$ is equivalent to $(f_i(\boldsymbol{y}))^{q-1} = 1$. Secondly, if $\boldsymbol{y}$ is a solution, any nonzero scalar multiple of $\boldsymbol{y}$ is also a solution. From this observation we can assume without loss of generality that the value of one of f_i's is equal to 1. In the following, we assume that the value of the last one is equal to one, i.e., $3y_1 + 2y_2 = 1$.

We transform the system of inequations to

$$(y_2 + 2y_3)^4 - 1 = 0 \bmod 5 \tag{33}$$
$$(y_2 + 4y_3)^4 - 1 = 0 \bmod 5 \tag{34}$$
$$(y_1 + 2y_2)^4 - 1 = 0 \bmod 5 \tag{35}$$
$$(y_1 + 3y_3)^4 - 1 = 0 \bmod 5 \tag{36}$$
$$(4y_1 + y_2 + 2y_3)^4 - 1 = 0 \bmod 5 \tag{37}$$
$$3y_1 + 2y_2 - 1 = 0 \bmod 5. \tag{38}$$

Let I be the ideal generated by the six polynomials on the left-hand sides of (33) to (38) in the polynomial ring GF$(5)[y_1, y_2, y_3]$. Let G be a Groebner basis of I with respect to the lexicographical order defined by $y_1 \succ y_2 \succ y_3$. The Groebner basis generates the ideal I, and has many desirable properties. We will use Groebner basis to eliminate the variables. We refer the readers to the book [39] for more details.

Using SINGULAR [40], a software for computational commutative algebra, we find that G consists of the following four polynomials

$$g_1(\boldsymbol{y}) = y_3^4 + y_3^3 + y_3^2 + y_3 \tag{39}$$
$$g_2(\boldsymbol{y}) = y_2^2(y_3 + 3) + y_2\left(4y_3^3 + 3y_3^2 + 2y_3 + 2\right) - y_3^3 - y_3^2 - y_3 - 1 \tag{40}$$
$$g_3(\boldsymbol{y}) = y_2^3 - 2y_2^2 + \left(y_3^3 - 2y_3^2 - y_3 - 1\right) y_2 + 2y_3^3 + 2y_3^2 + 2y_3 + 2 \tag{41}$$
$$g_4(\boldsymbol{y}) = y_1 - y_2 - 2. \tag{42}$$

The first polynomial $g_1(\boldsymbol{y})$ in the Groebner basis involves variable y_3 only, while $g_2(\boldsymbol{y})$ and $g_3(\boldsymbol{y})$ involve only y_2 and y_3. We can solve for y_3 by the first polynomial, then solve for y_2 by the second and third polynomial, and finally determine the value of y_1 by the last polynomial.

From $y_3^4 + y_3^3 + y_3^2 + y_3 = 0$, we see that $y_3 = 0$ is one possibility. Substitute $y_3 = 0$ into $g_2(\boldsymbol{y}) = 0$ and $g_3(\boldsymbol{y}) = 0$, we obtain

$$3y_2^2 + 2y_2 - 1 = 0$$

and

$$y_2^3 - 2y_2^2 - y_2 + 2,$$

and check that $y_2 = 2$ and $y_2 = 4$ are the common solutions. From $g_4(\boldsymbol{y}) = 0$, y_1 can be calculated by $y_1 = y_2 + 2$. Therefore $\boldsymbol{y} = (4, 2, 0)$ and $\boldsymbol{y} = (1, 4, 0)$ are solutions to the system of inequations (27) to (32). For example, we can verify that if $y_1 = 4$, $y_2 = 2$, and $y_3 = 0$, we have

$$\begin{aligned}(2) + 2(0) &= 2 \bmod 5\\ (2) + 4(0) &= 2 \bmod 5\\ (4) + 2(2) &= 3 \bmod 5\\ (4) + 3(0) &= 4 \bmod 5\\ 4(4) + (2) + 2(0) &= 3 \bmod 5\\ 3(4) + 2(2) &= 1 \bmod 5.\end{aligned}$$

For $y_3 = 2, 3, 4$, we can repeat the above procedure and solve for $\boldsymbol{y}$ in the same manner. (We remark that there exist solutions to the inequations (27) to (32) with $y_3 = 1$ as well, for example, $\boldsymbol{y} = (0, 2, 1)$. However, none of them satisfies the equality in (38), and hence is not picked up by the elimination method by Groebner basis.)

REFERENCES

[1] A. Cohen, E. Biton, J. Kampeas, and O. Gurewitz, "Coded unicast downstream traffic in a wireless network: Analysis and WiFi implementation," *EURASIP J. Signal Process.*, vol. 25, no. 2, pp. 1–20, May 2013.

[2] F. Sun, T. M. Kim, A. J. Paulraj, E. de Carvalho, and P. Popovski, "Cell-edge multi-user relaying with overhearing," *IEEE Commun. Lett.*, vol. 17, no. 6, pp. 1160–1163, Jun. 2013.

[3] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[4] S. Katti *et al.*, "XORs in the air: Practical wireless network coding," in *Proc. SIGCOMM*, Pisa, Italy, Sep. 2006, pp. 243–254.

[5] Y. Birk and T. Kol, "Informed-source coding-on-demand (ISCOD) over broadcast channels," in *Proc. IEEE INFOCOM*, San Francisco, CA, USA, Mar. 1998, pp. 1257–1264.

[6] S. El Rouayheb, A. Sprintson, and C. Georghiades, "On the index coding problem and its relation to network coding and matroid theory," *IEEE Trans. Inf. Theory*, vol. 56, no. 7, pp. 3187–3195, Jul. 2010.

[7] H. Maleki, V. Cadambe, and S. Jafar, "Index coding: An interference alignment perspective," in *Proc. IEEE Int. Symp. Inf. Theory*, Cambridge, MA, USA, Jul. 2012, pp. 2236–2240.

[8] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol, "Index coding with side information," *IEEE Trans. Inf. Theory*, vol. 57, no. 3, pp. 1479–1494, Mar. 2011.

[9] E. Lubetzky and R. Stav, "Nonlinear index coding outperforming the linear optimum," *IEEE Trans. Inf. Theory*, vol. 55, no. 8, pp. 3544–3551, Aug. 2009.

[10] N. Alon, A. Hassidim, E. Lubetzky, U. Stav, and A. Weinstein, "Broadcasting with side information," in *Proc. 49th IEEE Symp. Found. Comput. Sci.*, 2008, pp. 823–832.

[11] S. El Rouayheb, M. A. R. Chaudhry, and A. Sprintson, "On the minimum number of transmissions in single-hop wireless coding networks," in *Proc. IEEE ITW*, Lake Tahoe, NV, USA, Sep. 2007, pp. 120–125.

[12] M. Langberg and A. Sprintson, "On the hardness of approximating the network coding capacity," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 1008–1014, Feb. 2011.

[13] R. Peeters, "Orthogonal representations over finite fields and the chromatic number of graphs," *Combinatorica*, vol. 16, no. 3, pp. 417–431, 1996.

[14] A. Blasiak, R. Kleinberg, and E. Lubetsky, "Index Coding Via Linear Programming," 2011, arXiv:1004.1379.

[15] K. Shanmugam, A. G. Dimakis, and M. Langberg, "Local graph coloring and index coding," in *Proc. IEEE Int. Symp. Inf. Theory*, Istanbul, Turkey, Jul. 2013, pp. 1152–1156.

[16] F. Arbabjolfaei, B. Bandemer, Y. H. Kim, E. Sasoglu, and L. Wang, "On the Capacity Region for Index Coding," in *Proc. IEEE Int. Symp. Inf. Theory*, 2013, pp. 962–966.

[17] S. H. Dau, V. Skachek, and Y. Chee, "Optimal index codes with near-extreme rates," in *Proc. IEEE Int. Symp. Inf. Theory*, Cambridge, MA, USA, Jul. 2012, pp. 2241–2245.

[18] Y. Berliner and M. Langberg, "Index coding with outerplanar side information," in *Proc. IEEE Int. Symp. Inf. Theory*, St. Petersburg, Russia, Jul. 2011, pp. 806–810.

[19] L. Ong and C. K. Ho, "Optimal index codes for a class of multicast networks with receiver side information," in *IEEE Proc. ICC*, Ottawa, ON, Canada, Jun. 2012, pp. 2223–2228.

[20] M. A. R. Chaudhry and A. Sprintson, "Efficient algorithms for index coding," in *Proc. IEEE INFOCOM*, Phoenix, AZ, USA, Apr. 2008, pp. 1–4.

[21] S. Sorour and S. Valaee, "An adaptive network coded retransmission scheme for single-hop wireless multicast broadcast services," *IEEE/ACM Trans. Netw.*, vol. 19, no. 3, pp. 869–878, Jun. 2011.

[22] M. A. R. Chaudhry, Z. Asad, A. Sprintson, and M. Langberg, "On the complementary index coding problem," in *Proc. IEEE Int. Symp. Inf. Theory*, St. Petersburg, Russia, Jul. 2011, pp. 244–248.

[23] E. Chlamtác and I. Haviv, "Linear index coding via semidefinite programming," in *Proc. ACM-SIAM*, Kyoto, Japan, Jan. 2012, pp. 1–5.

[24] C. Zhan, V. C. S. Lee, J. Wang, and Y. Xu, "Coding-based data broadcast scheduling in on-demand broadcast," *IEEE Trans. Wireless Commun.*, vol. 10, no. 11, pp. 3774–3783, Nov. 2011.

[25] S. Sorour, "Completion delay minimization for instantly decodable network coding," Ph.D. dissertation, Univ. Toronto, Toronto, ON, Canada, 2011.

[26] C. Gong and X. Wang, "On partial downloading for wireless distributed storage networks," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 3278–3288, Jun. 2012.

[27] R. Zeng, Y. Jiang, C. Lin, Y. Fan, and X. Shen, "A distributed fault/intrusion-tolerant sensor data storage scheme based on network coding and homomorphic fingerprinting," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 10, pp. 1819–1830, Oct. 2012.

[28] F. Sun, E. de Carvalho, P. Popovski, and C. Thai, "Coordinated direct and relay transmission with linear non-regenerative relay beamforming," *IEEE Signal Process. Lett.*, vol. 19, no. 10, pp. 680–683, Oct. 2012.

[29] A. S. Tehrani, A. G. Dimakis, and M. J. Neely, "Bipartite index coding," in *Proc. IEEE Int. Symp. Inf. Theory*, Cambridge, MA, USA, Jul. 2012, pp. 2246–2250.

[30] K. W. Shum, M. Dai, and C. W. Sung, "Broadcasting with coded side information," in *Proc. 2nd Int. Workshop Netw. Coding Wireless Relay Netw.*, Sydney, NSW, Australia, Sep. 2012, pp. 89–94.

[31] M. J. Neely, A. S. Tehrani, and Z. Zhang, "Dynamic index coding for wireless broadcast networks," in *IEEE Proc. INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 316–324.

[32] R. Liu *et al.*, "Cooperative caching scheme for content oriented networking," *IEEE Commun. Lett.*, vol. 17, no. 4, pp. 781–784, Apr. 2013.

[33] J. Y. Kim, G. M. Lee, and J. K. Choi, "Efficient multicast schemes using in-network caching for optimal content delivery," *IEEE Commun. Lett.*, vol. 17, no. 5, pp. 1048–1051, May 2013.

[34] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femto-caching and device-to-device collaboration: A new architecture for wireless video distribution," *IEEE Commun. Mag.*, vol. 51, no. 4, pp. 142–149, Apr. 2013.

[35] A. Aijaz, H. Aghvami, and M. Amani, "A survey on mobile data offloading: Technical and business perspectives," *IEEE Wireless Comm.*, vol. 20, no. 2, pp. 104–112, Apr. 2013.

[36] M. Dai, H. Y. Kwan, and C. W. Sung, "Linear network coding strategies for the multiple-access relay channel with packet erasures," *IEEE Trans. Wireless Commun.*, vol. 12, no. 1, pp. 218–227, Jan. 2013.

[37] P. Puducheri-Sundaravaradhan and T. E. Fuja, "Capacity and coding for two common wireless erasure relay networks with optimal bandwidth allocation," *IEEE Trans. Wireless Commun.*, vol. 11, no. 12, pp. 4308–4317, Dec. 2012.

[38] M. H. Firooz and S. Roy, "Collaborative downloading in VANET using network coding," in *Proc. IEEE ICC*, Ottawa, ON, Canada, Jun. 2012, pp. 4584–4588.

[39] D. A. Cox, J. B. Little, and D. B. O'Shea, *Using Algebraic Geometry*. New York, NY, USA: Springer-Verlag, 2005.

[40] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann, SINGULAR—A Computer Algebra System for Polynomial Computations, 2011. [Online]. Available: http://www.singular.uni-kl.de

Authors' photographs and biographies not available at the time of publication.