Linear Network Code for Erasure Broadcast Channel with Feedback: Complexity and Algorithms

Ho Yuet Kwan, Kenneth W. Shum, Chi Wan Sung

***Abstract*—This paper investigates the construction of linear network codes for broadcasting a set of data packets to a number of users. The links from the source to the users are modeled as independent erasure channels. Users are allowed to inform the source node whether a packet is received correctly via feedback channels. In order to minimize the number of packet transmissions until all users have received all packets successfully, it is necessary that a data packet, if successfully received by a user, can increase the dimension of the vector space spanned by the encoding vectors he or she has received by one. Such an encoding vector is called innovative. To reduce decoding complexity, sparse encoding vectors are preferred, since the sparsity can be exploited when solving systems of linear equations. Generating a sparsest encoding vector with large finite field size, however, is shown to be NP-hard. An approximation algorithm is constructed. For binary field, heuristic algorithms are also proposed.**

***Index Terms*—Erasure broadcast channel, network coding, computational complexity.**

I. Introduction

Broadcasting has been a challenging issue in telecommunications. The challenge mainly comes from how a transmitter can disseminate a common information content to all users/receivers reliably and efficiently via a broadcast channel which could be unstable and error-prone. More specifically, one of the ultimate goal of broadcasting is to provide a transmission scheme such that a common information content or a set of packets can be disseminated with minimum number of transmissions for a sender to complete the whole information content dissemination for all users. This measure is commonly called the *completion time* of a broadcast system.

Several classical approaches provide heuristic solutions to the above issue. With user feedback, automatic repeat request (ARQ) offers reliable retransmissions for the erased packets due to channel impairments. However, such an approach becomes inefficient when the number of users increases, as the users may have entirely distinct needs for the erased packets. Reliable broadcasting can also be achieved without user feedback by forward error correction. With the use of erasure codes, a user can reconstruct the entire set of original packets, provided that the number of erased packets is smaller than a certain threshold. However, the amount of packets that will be erased depends on the channel erasure probability, which is time-varying and hard to predict. That limits the use of erasure codes in broadcasting. To improve upon classical approaches, the approach of linear network coding [1], [2] has been shown to be a promising solution [3]–[6].

The idea of linear network coding for broadcasting is that a transmitter broadcasts to K users encoded packets that are obtained by linear combining the N original packets over the finite field $GF(q)$. An encoding vector specifies the coefficients for the linear combination. An encoded packet together with a header which contains the corresponding encoding vector is broadcasted to all users. It is said to be *innovative to a user* if the corresponding encoding vector is not in the subspace spanned by the encoding vectors already received by that user. It is called *innovative* if it is innovative to all users who have not yet received enough packets for decoding. It is shown in [7] that an innovative packet can always be found if $q \geq K$. Once a user receives any N innovative packets, he or she can decode the N original packets by Gauss-Jordan elimination. Therefore, the generation of innovative packets is vital. Clearly, if all the encoded packets are innovative, the completion time can be minimized. Such a network-coded broadcast scheme is clearly rate-optimal.

Linear network codes for broadcasting can be generated with or without feedback. LT codes [8], Raptor codes [9] and random linear network codes (RLNC) [10] can be used without feedback. By suitably choosing design parameters, innovative packets can be generated by those coding schemes with high probability. LT codes and Raptor codes are generated by an optimized degree distribution. However, they are mainly designed for broadcasting a huge number of packets, and may not be good choices when the number of packets is only moderately large. With feedback, it is suggested in [7] the use of Jaggi-Sanders algorithm [11], which is a general network code generation method and is able to find innovative encoding vectors for $q \geq K$. However, its encoding and decoding complexities are relatively high, as it is not specifically designed for the broadcast application. Therefore, some heuristics have been proposed [12]–[15]. It is suggested in [16] that encoded packets should be *instantly decodable*, in the sense that a new packet can be decoded once it is available at a receiver without waiting for the complete reception of the full set of packets. However, as an instantly decodable packet to all users may

This work was partially supported by a grant from the University Grants Committee of the Hong Kong Special Administrative Region, China (Project No. AoE/E-02/08).

Ho Yuet Kwan is with Department of Electronic Engineering, City University of Hong Kong, Tat Chee Ave, Kowloon Tong, Hong Kong. Email: hykwan@cityu.edu.hk.

Kenneth Shum is with Institute of Network Coding, The Chinese University of Hong Kong, Shatin, Hong Kong. Email: wkshum@inc.cuhk.edu.hk.

Chi Wan Sung is with Department of Electronic Engineering, City University of Hong Kong, Tat Chee Ave, Kowloon Tong, Hong Kong. Email: albert.sung@cityu.edu.hk.

This paper was presented in part in Int. Symp. on Inform. Theory, St. Petersburg, 2011, and in Int. Symp. in Network Coding, Beijing, 2011.

not exist, the completion time is in general larger than that in a system without this extra requirement. With the idea of instantly decodability, some works focus on minimizing *decoding delay*, where a unit of decoding delay is defined as that an encoded packet is successfully received by a user but that packet is not innovative or not instantly decodable to him or her [17]–[20].

The excellent performance of the linear network coding approaches to broadcast encourages researchers to consider its practicality. In fact, the decoding complexity of a linear network code is an important issue in practice. One possible way to reduce decoding complexity is to use sparse encoding vectors. This sparsity property is important, as it can be exploited in the decoding process. For example, a fast algorithm by Wiedemann for solving a system of sparse linear equations can be used [21]. If the Hamming weight of each encoding vector is at most w, the complexity for solving an $N \times N$ linear system can be reduced from $O(N^3)$ using Gaussian elimination to $O(wN^2)$. The Wiedemann algorithm is useful when N is large. When N is moderate, we can implement some sparse representation of matrices, so that even the usual Gaussian elimination is used, the number of additions and multiplications required can be reduced. For other fast methods for solving linear equations over finite field, we refer the readers to [22], [23].

Minimizing the completion time and reducing the decoding complexity are equally important in linear network code design for erasure broadcast channel. However, the innovativeness of encoding vectors together with their sparsity has not been thoroughly studied. Given the encoding vectors which have been received by the users in a broadcast system, how can a transmitter generate encoding vectors which are both sparse and innovative is a challenging problem. In this paper, we address the issue by developing a method called the *Optimal Hitting Method* and its approximation version, called the *Greedy Hitting Method*. Both of them are able to generate sparse and innovative encoding vectors for $q \geq K$. That results in a significant reduction in decoding complexity when compared with their non-sparse counterparts. Furthermore, based on the greedy hitting method, we develop a suboptimal procedure to improve the completion time performance for $q = 2$, where the existence of innovative encoding vectors is not guaranteed. Simulation results show that its performance is nearly optimal.

The rest of this paper is organized as follows. We review the literature on complexity in network coding in Section II. In Section III, the system model is introduced. In Section IV, we consider the innovativeness issue. We characterize innovative encoding vectors by a linear algebra approach and prove that to determine whether there exists an innovative vector for $q = 2$ is NP-complete. In Section V, the sparsity issue is considered. After showing that K-sparse innovative vectors always exist if $q \geq K$, we investigate the problem of finding sparsest innovative vectors, which we call the SPARSITY problem. SPARSITY is proven to be NP-complete. In Section VI, we present a systematic way to solve SPARSITY using binary integer programming. A polynomial-time approximation algorithm is also constructed. In Section VII, our algorithms are compared with some other transmission schemes by simulations. Finally, conclusions are drawn in Section VIII.

II. LITERATURE ON COMPLEXITY CLASSES OF NETWORK CODING PROBLEMS

A considerable amount of research has been done on the complexity issues in conventional coding theory (See the survey in [24] for example). For instance, it is shown in [25] and [26] that the problems of finding the weight distribution and the minimum distance of linear codes are NP-hard. The complexity issues in network coding are less well understood.

For linear network codes, Lehman and Lehman investigate the complexity of a class of network coding problems in [27], and proved that some of the problems are NP-complete. Construction of linear network codes using a technique called matrix completion is considered in [28], and the complexity class of the matrix completion problem is studied in [29]. It is shown in [30] that approximating the capacity of network coding is also a hard problem.

To minimize encoding complexity, Langberg, Sprintson and Bruck divide the nodes in a general network topology into two classes. The nodes in one class forward packets without any coding while the nodes in another class perform network coding. The problem of minimizing the number of encoding nodes is shown to be NP-complete in [31], [32].

El Rouayheb, Chaudhry and Sprintson study the complexity of a related problem called *index coding problem* in [33]. They consider the noiseless broadcast channel, and show that when the coefficient field is binary, the problem of minimizing the number of packet transmissions is NP-hard. A complementary version of the index coding is studied in [34]. It is shown that the complementary index coding is NP-hard, and even obtaining an approximate solution is NP-hard.

In [35], Milosavljevic et al. studies a related system. The users are interested in a common data file but only have partial knowledge of the file. By interactively sending data to each others through a noiseless broadcast channel, the users want to minimize the total amount of data sent through the channel. It is shown in [35] that the optimal rate allocations can be found in polynomial time.

In this paper, the problem setting is similar, except that the channel is modeled as an erasure broadcast channel, and we focus on the innovativeness and sparsity aspects of generating encoding vectors.

III. SYSTEM MODEL

Consider a single-hop broadcast system, in which there are one transmitter and K users. The transmitter wants to send a file to all users via a broadcast channel, which is modeled as a discrete-time broadcast packet erasure channel. At each time, the transmitter broadcasts a packet, which is an element in $GF(q)$. Each user either receives the packet successfully or experiences a packet loss. In other words, the output symbol of a user is either the same as the input symbol or an erasure symbol. The K output symbols of the users are assumed to be independent of one another.

In this paper, we focus on transmission schemes with linear network coding. The file is divided into N equal-size packets. Each packet transmitted by the transmitter is obtained by linearly combining the N packets, with coefficients drawn from $GF(q)$. An N-vector whose components are the N coefficients is said to be the *encoding vector* of that packet. A packet together with a header that contains the corresponding encoding vector is broadcast to K users.

We assume that there is an error-free feedback channel from each user to the transmitter. Upon receiving a packet successfully, a user sends an acknowledgement (ACK) to the transmitter without delay or error. The transmitter keeps track of what each user has received. The next transmitted packet depends on all the previous ACKs from the K users.

Given an encoding vector $\mathbf{x} = (x_1, x_2, \dots, x_N)$[1] of dimension N over $GF(q)$, the *support* of $\mathbf{x}$, denoted by $supp(\mathbf{x})$, is the set of indices of the non-zero components in $\mathbf{x}$, i.e.,

$$supp(\mathbf{x}) \triangleq \{i : x_i \neq 0\}.$$

The *Hamming weight* of $\mathbf{x}$ is defined as the cardinality of $supp(\mathbf{x})$. An encoding vector that has Hamming weight less than or equal to w is said to be w-sparse.

Note that a transmitted packet is useful to a user if its corresponding encoding vector does not lie within the span of all previously received encoding vectors of that user. We say that such an encoding vector is *innovative to that user*. An encoding vector that is innovative to all users is simply said to be *innovative*. It is clear that a rate-optimal solution is to let the transmitter always broadcast innovative packets, provided that innovative encoding vectors always exist.

IV. The Innovative Encoding Vector Problem

Suppose that user k, for $k = 1, 2, \dots, K$, has already received r_k packets whose encoding vectors are linearly independent. Let $\mathbf{C}_k$ be the $r_k \times N$ encoding matrix of user k, whose rows are the r_k encoding vectors. Without loss of generality, we assume that $r_k < N$, for otherwise user k can decode the file successfully and can be omitted from our consideration. A vector $\mathbf{x}$ is innovative if it does not belong to the row space of $\mathbf{C}_k$ for any k. The set of all innovative encoding vectors, $\mathcal{I}$, is given by

$$\mathcal{I} \triangleq GF(q)^N \setminus \bigcup_{k=1}^{K} \text{rowspace}(\mathbf{C}_k). \tag{1}$$

Example: Let $q = 2$, $K = 2$, $N = 4$ and $n = 2$. The two matrices $\mathbf{C}_1$ and $\mathbf{C}_2$ over $GF(2)$ are given by

$$\mathbf{C}_1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \ \mathbf{C}_2 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

The row space of $\mathbf{C}_1$ consists of four vectors [0 0 0 0], [1 1 0 0], [0 0 1 0], and [1 1 1 0]. The row space of $\mathbf{C}_2$ consists of the eight vectors that have even Hamming weight.

[1]Throughout this paper, all vectors are column vectors. For vector x, we use parentheses and commas when its components are listed horizontally, i.e., $(x_1, x_2, \dots, x_N)$.

There are six innovative encoding vectors, and they form the set

$$\mathcal{I} = \{(1,0,0,0), (0,1,0,0), (0,0,0,1), \\ (0,1,1,1), (1,0,1,1), (1,1,0,1)\}.$$

It is well known that $\mathcal{I}$ is non-empty if the finite field size, q, is larger than or equal to the number of users, K [7]. We repeat the simple proof below for the sake of completeness. We begin with a simple lemma, which will be used again in a later section.

Lemma 1. *Let $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_K$ be finite subsets of a universal set $\mathcal{U}$. If $K \geq 2$ and $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_K$ contain a common element, then*

$$|\mathcal{A}_1 \cup \mathcal{A}_2 \cup \ldots \cup \mathcal{A}_K| < |\mathcal{A}_1| + |\mathcal{A}_2| + \ldots + |\mathcal{A}_K|.$$

Proof: Suppose $x \in \mathcal{A}_i$ for all i. Let $\mathcal{A}_i^*$ be the set $\mathcal{A}_i \setminus \{x\}$ for $i = 1, 2, \dots, K$. By applying the union bound,

$$|\mathcal{A}_1^* \cup \mathcal{A}_2^* \cup \ldots \cup \mathcal{A}_K^*| \leq |\mathcal{A}_1^*| + |\mathcal{A}_2^*| + \ldots + |\mathcal{A}_K^*|.$$

This implies

$$|\mathcal{A}_1 \cup \mathcal{A}_2 \cup \ldots \cup \mathcal{A}_K| - 1 \leq |\mathcal{A}_1| - 1 + |\mathcal{A}_2| - 1 + \ldots + |\mathcal{A}_K| - 1.$$

As $K \geq 2$ by hypothesis, we obtain the inequality in the lemma. ■

Theorem 2 ([7]). *If $q \geq K$, then $\mathcal{I}$ is non-empty.*

Proof: For $k = 1, 2, \dots, K$, let V_k be the row space of $\mathbf{C}_k$. The subspace V_k consists of the q^{r_k} encoding vectors which are *not* innovative to user k. Obviously the zero vector is a common vector of these K subspaces. By Lemma 1, the union of these K subspaces contains strictly less than $\sum_{k=1}^K q^{r_k}$ vectors. Since $K \leq q$, we have $\sum_{k=1}^K q^{r_k} \leq Kq^{N-1} \leq q^N$. Therefore there exists at least one encoding vector which is innovative to all users. ■

The condition $q \geq K$ cannot be improved for any prime power q. The following example shows the non-existence of innovative encoding vector for the case where $K = q + 1$. Let U be the ambient space $GF(q)^N$, and V be a subspace of U with dimension $N - 2$. Let $\mathbf{v}_1, \dots, \mathbf{v}_{N-2}$ be a basis of V. The q^2 cosets of V in U form the quotient space U/V, and is isomorphic to a vector space over $GF(q)$. Let $\phi(u) : U \to U/V$ be the canonical mapping from U to the quotient vector space U/V. Because U/V has dimension 2 over $GF(q)$, we can find $q + 1$ vectors in U/V such that none of them is a scalar multiple of the others, i.e., they are pairwise linearly independent. As ϕ is a surjective mapping, we can choose vectors $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_{q+1}$ in U such that, $\phi(\mathbf{u}_1), \phi(\mathbf{u}_2), \dots, \phi(\mathbf{u}_{q+1})$ are pairwise linearly independent in U/V. For $k = 1, 2, \dots, q+1$, define $\mathbf{C}_k$ as the $(N-1) \times N$ matrix whose row vectors are $\mathbf{v}_1, \dots, \mathbf{v}_{N-2}$, and $\mathbf{u}_k$. The row spaces corresponding to the matrices $\mathbf{C}_k$'s satisfy
(i) $\text{rank}(\mathbf{C}_k) = N - 1$ for all k.
(ii) $\text{rowspace}(\mathbf{C}_i) \cap \text{rowspace}(\mathbf{C}_j) = V$ whenever $i \neq j$.
(iii) For $k = 1, 2, \dots, q + 1$, the sets $\text{rowspace}(\mathbf{C}_k) \setminus V$ are mutually disjoint.

The number of elements in the union of the row spaces $\mathbf{C}_k$'s is

$$\begin{aligned}&|V| + \sum_{k=1}^{q+1} |\text{rowspace}(\mathbf{C}_k) \setminus V| \\ &= q^{N-2} + (q+1)(q^{N-1} - q^{N-2}) \\ &= q^N.\end{aligned}$$

Hence the rowspaces of $\mathbf{C}_k$'s cover the whole vector space $GF(q)^N$. Any encoding vector we pick from $GF(q)^N$ is not innovative to at least one user.

As an example, we consider the case $q = 3$ and $K = 4$ and $N = 3$. If the encoding matrices are

$$\begin{bmatrix}1 & 1 & 1\\ 1 & 0 & 0\end{bmatrix}, \begin{bmatrix}1 & 1 & 1\\ 0 & 1 & 0\end{bmatrix}, \begin{bmatrix}1 & 1 & 1\\ 0 & 0 & 1\end{bmatrix}, \begin{bmatrix}1 & 1 & 1\\ 0 & 1 & 2\end{bmatrix},$$

then we cannot find any innovative encoding vector.

The set of innovative encoding vectors, $\mathcal{I}$, can be characterized by the orthogonal complements of the row spaces of $\mathbf{C}_k$'s, which is also known as the null spaces of $\mathbf{C}_k$'s. For $k = 1, 2, \ldots, K$, let V_k be the row space of $\mathbf{C}_k$. Denote the *orthogonal complement* of V_k by $V_k^\perp$,

$$V_k^\perp \triangleq \{\mathbf{v} \in GF(q)^N : \mathbf{x}\cdot\mathbf{v} = 0 \text{ for all } \mathbf{x} \in V_k\},$$

where $\mathbf{x}\cdot\mathbf{v}$ is the inner product of $\mathbf{x}$ and $\mathbf{v}$. We will use the fact from linear algebra that a vector $\mathbf{x}$ is in V_k if and only if $\mathbf{x}\cdot\mathbf{v} = 0$ for all $\mathbf{v} \in V_k^\perp$. Let $\mathbf{B}_k$ be an $(N - r_k) \times N$ matrix whose rows form a basis of $V_k^\perp$. To see whether a vector $\mathbf{x}$ is in V_k, it amounts to checking the condition $\mathbf{B}_k\mathbf{x} = \mathbf{0}$; if $\mathbf{B}_k\mathbf{x} = \mathbf{0}$, then $\mathbf{x} \in V_k$, and vice versa.

There are many different choices for the basis of the orthogonal complement $V_k^\perp$. We can obtain one such choice via the reduced row-echelon form (RREF) of $\mathbf{C}_k$. Suppose we have obtained the RREF of $\mathbf{C}_k$ by elementary row operations. By appropriately permutating its columns, we can write it in the following form:

$$[\mathbf{I}_{r_k}|\mathbf{A}_k]\mathbf{P}_k, \tag{2}$$

where $\mathbf{I}_{r_k}$ is the $r_k \times r_k$ identity matrix, $\mathbf{A}_k$ is an $r_k \times (N - r_k)$ matrix over $GF(q)$, and $\mathbf{P}_k$ is an $N \times N$ permutation matrix. (Recall that a permutation matrix is a square zero-one matrix so that each column and each row contain exactly one "1".) We can take

$$\mathbf{B}_k = [-\mathbf{A}_k^T|\mathbf{I}_{N-r_k}]\mathbf{P}_k^T. \tag{3}$$

The superscript ^{T} represents the transpose operator. It is straightforward to verify that the product of the matrix in (2) and $\mathbf{B}_k^T$ is a zero matrix. Hence, the rows of $\mathbf{B}_k$ are orthogonal to the rows of $\mathbf{C}_k$, and form a basis of $V_k^\perp$.

In the appendix, we give another way of computing a basis, which is suitable for incremental processing.

The following simple result characterizes the set of innovative encoding vectors, $\mathcal{I}$:

Theorem 3. *Given $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K$, an encoding vector $\mathbf{x}$ belongs to $\mathcal{I}$ if and only if $\mathbf{B}_k\mathbf{x} \neq \mathbf{0}$ for all k's.*

Proof: If $\mathbf{B}_k\mathbf{x} \neq \mathbf{0}$, then $\mathbf{x}$ is not in V_k and therefore, is innovative to user k. It is innovative if $\mathbf{B}_k\mathbf{x} \neq \mathbf{0}$ for all k's.

Conversely, if $\mathbf{B}_k\mathbf{x} = \mathbf{0}$ for some k, then $\mathbf{x}$ is in V_k, and hence is not innovative to user k. Therefore, $\mathbf{x} \notin \mathcal{I}$. ■

When the underlying finite field size is small, innovative encoding vectors may not exist. For further investigation of the existence of innovative encoding vectors, we formulate the following decision problem:

Problem: IEV$_q$

Instance: K matrices, $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K$, over $GF(q)$, each of which has N columns.

Question: Is there an N-dimensional vector $\mathbf{x}$ over $GF(q)$ which does not belong to the row space of $\mathbf{C}_k$ for $k = 1, 2, \ldots, K$.

The following result shows that the decision problem is NP-complete for $q = 2$.

Theorem 4. IEV$_2$ *is NP-complete.*

Proof: The idea is to reduce the 3-SAT problem, well-known to be NP-complete [36], to the IEV$_2$ problem. Recall that the 3-SAT problem is a Boolean satisfiability problem, whose instance is a Boolean expression written in conjunctive normal form with three variables per clause (3-CNF), and the question is to decide if there is some assignment of TRUE and FALSE vaules to the variables such that the given Boolean expression has a TRUE value.

Let E be a given Boolean expression with n variables $x_1, \ldots, x_n$, and m clauses in 3-CNF. We want to reduce the 3-SAT problem to the IEV$_2$ problem with $N = n+1$ packets and $K = m + 1$ users.

For the i-th clause ($i = 1, 2, \ldots, m$), we first construct a $3 \times (n+1)$ matrix $\mathbf{B}_i$. If the j-th literal ($j = 1, 2, 3$) in the i-th clause is x_k, then let the k-th component in the j-th row of $\mathbf{B}_i$ be one, and the other components be all zero. Otherwise, if the j-th literal in the i-th clause is $\neg x_k$, then let the k-th and the $(n + 1)$-st component in the j-th row of $\mathbf{B}_i$ be both one, and the remaining components be all zero. Let $\mathbf{C}_i$ be a matrix whose rows form a basis of the orthogonal complement of the row space of $\mathbf{B}_i$. We will use the fact that a vector $\mathbf{v}$ is in the row space of $\mathbf{C}_i$ if and only if $\mathbf{B}_i\mathbf{v} = 0$.

Consider an example with $n = 4$ Boolean variables. From the clause $\neg x_1 \vee \neg x_2 \vee x_3$, we get

$$\mathbf{B}_i = \begin{bmatrix}1 & 0 & 0 & 0 & 1\\ 0 & 1 & 0 & 0 & 1\\ 0 & 0 & 1 & 0 & 0\end{bmatrix}, \ \mathbf{C}_i = \begin{bmatrix}0 & 0 & 0 & 1 & 0\\ 1 & 1 & 0 & 0 & 1\end{bmatrix}.$$

It can be verified that each row in $\mathbf{B}_i$ is orthogonal to the rows in $\mathbf{C}_i$, i.e., the row space of $\mathbf{C}_i$ is the orthogonal complement of the row space of $\mathbf{B}_i$.

For the extra user, user $m+1$, let $\mathbf{B}_{m+1}$ be the $1 \times (n+1)$ matrix $[\mathbf{0}_n 1]$, where $\mathbf{0}_n$ stands for the $1 \times n$ all-zero vector. The problem reduction can be done in polynomial time.

Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ be a Boolean vector and $\hat{\mathbf{x}} = (\mathbf{x}, 1)$. Note that any solution $\mathbf{x}$ to a given 3-SAT problem instance would cause the product $\mathbf{B}_j\hat{\mathbf{x}}$ a non-zero vector for $j = 1, 2, \ldots, m$ and $[\mathbf{0}_n 1]\hat{\mathbf{x}} \neq 0$. Therefore $\hat{\mathbf{x}}$ is not in the row space of $\mathbf{C}_j$ for all j. Hence $\hat{\mathbf{x}}$ is also a solution to the derived IEV$_2$ problem.

Conversely, any solution to the derived IEV$_2$ problem also yields a solution to the original 3-SAT problem as well. Let

$\mathbf{c} = (c_1, c_2, \ldots, c_n, c_{n+1}) \in GF(2)^{n+1}$ be a solution to the derived IEV$_2$ problem. Note that we must have $c_{n+1} = 1$ because of $\mathbf{B}_{m+1}$. Let i be an integer between 1 and m. Since $\mathbf{c}$ is not in the row space of $\mathbf{C}_i$, the product $\mathbf{B}_i\mathbf{c}$ is a non-zero vector. Hence, if we assign TRUE to x_k if $c_k = 1$ and FALSE to x_k if $c_k = 0$, for $k = 1, 2, \ldots, n$, then the i-th clause will have a TRUE value. Since this is true for all i, the whole Boolean expression also has a TRUE value.

The problem IEV$_2$ is clearly in NP, since it is efficiently verifiable. Hence it is NP-complete. ■

From the above result, we know that it is difficult to determine whether there exists an innovative vector for $q = 2$ with a general K. Apart from the problem of existence of an innovative vector, it is also of interest in finding an N-dimensional encoding vector that is innovative to as many users as possible. We state the optimization problem as follows:

Problem: MAX-IEV$_q$

Instance: K matrices $\mathbf{C}_k$ over $GF(q)$, $k = 1, 2, \ldots, K$, and each matrix has N columns.

Objective: Find an N-dimensional vector $\mathbf{x}$ over $GF(q)$ such that the number of users to whom $\mathbf{x}$ is innovative is maximized.

The following result shows the hardness of finding approximate solution to MAX-IEV$_q$:

Theorem 5. *There is no approximation algorithm for* MAX-IEV$_q$ *with an approximation guarantee of* $1 - \epsilon_M$, *assuming P ≠ NP, where* ϵ_M *is a positive constant.*

Proof: Given a Boolean expression in the 3-CNF form, the problem of maximizing the number of clauses that have TRUE values is commonly called the MAX-3-SAT problem. Consider the same reduction described in the proof of Theorem 4. It is clear that the number of clauses that have TRUE values under a given Boolean vector $\mathbf{x}$ is the same as the number of users to whom $\hat{\mathbf{x}} = (\mathbf{x}, 1)$ is innovative, excluding user $m + 1$. Therefore, the reduction is a gap-preserving reduction from MAX-3-SAT to MAX-IEV$_2$. The statement then follows from [37, Corollary 29.8]. ■

V. THE SPARSITY PROBLEM

Decoding complexity is one of the critical issues that could determine the practicality of linear network coding in broadcast erasure channels. One way to reduce the decoding complexity is to generate sparse encoding vectors and apply a decoding algorithm that exploits the sparsity of encoding vectors at receivers. In this section, we focus on the sparsity issues of innovative encoding vectors.

A. Existence of K-sparse innovative vector

In the previous section, it is found that innovative vectors always exist if $q \geq K$. In fact, we can prove a stronger statement that K-sparse innovative vectors always exist under the same condition.

Lemma 6. *For* $k = 1, 2, \ldots, K$, *let* $f_k(\mathbf{x})$ *be a non-zero linear polynomials in L variables*

$$f_k(\mathbf{x}) \triangleq \alpha_{k1}x_1 + \alpha_{k2}x_2 + \cdots + \alpha_{kL}x_L, \; k = 1, 2, \ldots, K,$$

where the coefficients are elements in $GF(q)$. *If* $q \geq K$, *we can always find a vector* $\mathbf{x}^* = (x_1, x_2, \ldots, x_L) \in GF(q)^L$ *such that* $f_k(\mathbf{x}^*) \neq 0$ *for all* k.

We first give a combinatorial proof, and then provide a deterministic algorithm which finds $\mathbf{x}^*$.

Proof: For $k = 1, 2, \ldots, K$, let V_k be the set of vectors $\mathbf{x}$ in $GF(q)^L$ satisfying $f_k(\mathbf{x}) = 0$. The set V_k is a subspace of dimension $L-1$. By Lemma 1, the cardinality of the union of these K subspaces is strictly less than Kq^{L-1} elements, which in turn is less than or equal to the cardinality of the whole space $GF(q)^L$. Thus there exists at least one vector $\mathbf{x}^*$ in $GF(q)^L$ such that $f_k(\mathbf{x}^*) \neq 0$ for all k. ■

Now we give an algorithmic proof of Lemma 6. Let $\mathcal{S}_l$, where $l = 1, \ldots, L$, be the index set such that $k \in \mathcal{S}_l$ if and only if $\alpha_{kl} \neq 0$. Since none of the linear polynomials $f_k(\mathbf{x})$'s are zero, we have $\bigcup_{l=1}^{L} \mathcal{S}_l = \{1, 2, \ldots, K\}$. We distinguish two cases:

Case 1: $|\mathcal{S}_l| = K$ for some l. We can simply let $x_l^* = 1$ and $x_n^* = 0$ for $n \neq l$.

Case 2: $|\mathcal{S}_l| < K$ for all l. We assign values to the variables iteratively. Suppose we have already assign x_t^* to the variable x_t, for $t = 1, 2, \ldots, l-1$. We note that $f_k(x_1^*, \ldots, x_{t-1}^*, x_t, 0, \ldots, 0) = 0$ is a linear equation in a single variable x_t, and thus have only one solution. As $|\mathcal{S}_l| < K \leq q$, the number of elements in $GF(q)$ which satisfy $f_k(x_1^*, \ldots, x_{t-1}^*, x_t, 0, \ldots, 0) = 0$ for some $k \in \mathcal{S}_l$ is strictly less than q. We can choose x_t^* such that

$$f_k(x_1^*, x_2^*, \ldots, x_{t-1}^*, x_t^*, 0, 0, \ldots, 0) \neq 0$$

for all $k \in \mathcal{S}_l$. Upon termination, it is guaranteed that $f_k(x_1^*, x_2^*, \ldots, x_L^*) \neq 0$ for all k. We call this method the *Sequential Assignment* (SA) algorithm. Its computational complexity in terms of number of multiplications/divisions over $GF(q)$ is analyzed as follows: In this algorithm, there are L iterations. In each iteration, we need to find an element in $GF(q)$ that is not a root of any of these K equations. Consider the t-th iteration. For the k-th equation, we need to compute $\alpha_{k,t-1}x_{t-1}^*$ and add it to the accumulated sum $\sum_{j=1}^{t-2} \alpha_{kj}x_j$, which is stored for the next iteration. The root of this equation can then be obtained by a division. Therefore, the total complexity of SA is $O(KL)$.

Example: Let

$$f_1(\mathbf{x}) \triangleq x_1 + 2x_2$$
$$f_2(\mathbf{x}) \triangleq x_2 + 2x_3$$
$$f_3(\mathbf{x}) \triangleq 2x_1 + x_3$$

be $K = 3$ linear polynomials over $GF(3)$. We apply the algorithm in Lemma 6 to find an assignment of $\mathbf{x} = (x_1, x_2, x_3)$ such that $f_1(\mathbf{x})$, $f_2(\mathbf{x})$ and $f_3(\mathbf{x})$ are all non-zero. First of all, the three index sets are $\mathcal{S}_1 = \{1, 3\}$, $\mathcal{S}_2 = \{1, 2\}$, and $\mathcal{S}_3 = \{2, 3\}$. None of them has cardinality three. We proceed as described in the second case. We assign an arbitrary non-zero value to x_1, say $x_1 = 1$, and we can check that $f_1(1, 0, 0) = 1$, $f_2(1, 0, 0) = 0$, $f_3(1, 0, 0) = 2$.

Next, we want to find $x_2 \in GF(3)$ such that

$$f_1(1, x_2, 0) = 1 + 2x_2 \neq 0, \text{ and}$$
$$f_2(1, x_2, 0) = x_2 \neq 0.$$

It turns out that the only choice for x_2 is $x_2 = 2$. After x_2 is fixed, we search for $x_3 \in GF(3)$ such that

$$f_2(1, 2, x_3) = 2 + 2x_3 \neq 0$$
$$f_3(1, 2, x_3) = 2 + x_3 \neq 0.$$

The only choice for x_3 is $x_3 = 0$. Finally, we check that the values of f_1, f_2 and f_3 evaluated at $\mathbf{x} = (1, 2, 0)$,

$$f_1(1, 2, 0) = 2. \ f_2(1, 2, 0) = 1, \ f_3(1, 2, 0) = 2$$

are all non-zero.

By Lemma 6, we prove the following theorem:

Theorem 7. *If $q \geq K$, there exists a K-sparse encoding vector in $\mathcal{I}$.*

Proof: For $k = 1, 2, \ldots, K$, let $\mathbf{b}_k^T$ be an arbitrary row vector in $\mathbf{B}_k$, and let n_k be an arbitrary index such that the n_k-th component of $\mathbf{b}_k$ is non-zero. Form a new index set $\mathcal{N}$ that consists of all n_k's. The cardinality of $\mathcal{N}$ may be less than K since the n_k's may not be distinct. Let $\mathbf{b}_k(\mathcal{N})$ be a truncated vector of $\mathbf{b}_k$, which consists of only the components of $\mathbf{b}_k$ whose indices are in $\mathcal{N}$. Its dimension is equal to $|\mathcal{N}| \leq K$.

Now we show that there exists a vector $\mathbf{x} \in \mathcal{I}$ such that the i-th component of $\mathbf{x}$ is equal to zero if $i \notin \mathcal{N}$. If the i-th component of $\mathbf{x}$ is zero for all $i \notin \mathcal{N}$, then the inner product of $\mathbf{b}_k$ and $\mathbf{x}$ is the same as the inner product of $\mathbf{b}_k(\mathcal{N})$ and $\mathbf{x}(\mathcal{N})$. According to Theorem 3, $\mathbf{x}$ is in $\mathcal{I}$ if $\mathbf{b}_k(\mathcal{N}) \cdot \mathbf{x}(\mathcal{N}) \neq 0$ for all k's. By Lemma 6, we can find such a vector $\mathbf{x}$ if $q \geq K$. Clearly, such a vector has $\mathcal{N}$ as its support, and is hence K-sparse. ■

The above result shows that the minimum Hamming weight of innovative vectors is bounded above by K, assuming $q \geq K$. This upper bound cannot be further reduced as the following example shows: Consider a broadcast system of K users and N packets, where $N \geq K$. Suppose that user k has received a set of uncoded packets $\mathcal{A}_k$. Here we regard $\mathcal{A}_k$ as a subset of $\{1, 2, \ldots, N\}$. Furthermore, suppose that the complement of the $\mathcal{A}_k$'s are mutually disjoint, i.e., $\mathcal{A}_j^c \cap \mathcal{A}_k^c = \emptyset$ for $j \neq k$. In such a scenario, an innovative packet must be a linear combination of at least K packets. For example, let $N = 4$ and $K = 3$. If the encoding matrices of the three users are

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

then an innovative encoding vector must have Hamming weight at least 3. For instance $(1, 1, 1, 0)$ and $(1, 1, 0, 1)$ are innovative, but no vector with Hamming weight 2 or less is innovative.

B. Sparsest Innovative Vectors

Theorem 7 shows that we can always find a K-sparse innovative vector if $q \geq K$. It serves as an upper bound on the minimum Hamming weight of innovative vectors. To further reduce the decoding complexity, it is natural to consider the issue of finding a sparsest innovative encoding vector for given $\mathbf{C}_k$'s. In other words, we want to find a vector in $\mathcal{I}$ that has the minimum Hamming weight for the case where $q \geq K$. We call this algorithmic problem SPARSITY. We state its decision version formally as follows:

Problem: SPARSITY

Instance: A positive integer n and K matrices with N columns, $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K$, over $GF(q)$, where $q \geq K$.

Question: Is there a vector $\mathbf{x} \in \mathcal{I}$ with Hamming weight less than or equal to n?

Given all $\mathbf{C}_k$'s, we can find a basis $\mathbf{B}_k$'s of their corresponding null spaces by the method mentioned in Section IV. For $k = 1, 2, \ldots, K$, let $\mathbf{b}_{k,i}^T$ be the i-th row of $\mathbf{B}_k$. We define

$$\tilde{\mathbf{b}}_k \triangleq \vee_{i=1}^{N-r_k} \mathbf{b}_{k,i}, \tag{4}$$

where $\vee$ denotes the logical-OR operator applied component-wise to the $N - r_k$ vectors, with each non-zero component being regarded as a "1". In other words, the j-th component of $\tilde{\mathbf{b}}_k$ is one if and only if the j-th column of $\mathbf{B}_k$ is nonzero. We define $\mathbf{B}$ as the $K \times N$ matrix whose k-th row is equal to $\tilde{\mathbf{b}}_k^T$. Note that $\mathbf{B}$ is a binary matrix and has no zero rows. For a matrix $\mathbf{A}$ and a subset $\mathcal{N}$ of the column indices of $\mathbf{A}$, let $\mathbf{A}(\mathcal{N})$ be the $K \times |\mathcal{N}|$ submatrix of $\mathbf{A}$, whose columns are chosen according to $\mathcal{N}$. We need the following lemma:

Lemma 8. *Let $\mathcal{N} \subseteq \{1, 2, \ldots, N\}$ be an index set and $q \geq K$. There exists an encoding vector $\mathbf{x} = (x_1, x_2, \ldots, x_N) \in \mathcal{I}$ over $GF(q)$ with supp$(\mathbf{x}) \subseteq \mathcal{N}$ if and only if $\mathbf{B}(\mathcal{N})$ has no zero rows.*

Proof: If $\mathbf{B}(\mathcal{N})$ has no zero rows, then $\tilde{\mathbf{b}}_k(\mathcal{N}) \neq \mathbf{0}$ for all k's. Furthermore, for all k's, there must exist $\mathbf{b}_{k,j}(\mathcal{N}) \neq \mathbf{0}$ for some j. By Lemma 6, we can find $\mathbf{x}(\mathcal{N}) \in GF(q)^{|\mathcal{N}|}$ such that $\mathbf{b}_{k,j}(\mathcal{N}) \cdot \mathbf{x}(\mathcal{N}) \neq 0$ for all k's. Let the components of $\mathbf{x}$ whose indices do not belong to $\mathcal{N}$ be zero. Then by Theorem 3, $\mathbf{x} \in \mathcal{I}$.

Conversely, if $\mathbf{x}$ is an innovative vector with $x_n = 0$ for $n \notin \mathcal{N}$, then $\mathbf{B}(\mathcal{N})$ cannot have zero rows, for if row k of $\mathbf{B}(\mathcal{N})$ is a zero vector, then $\mathbf{B}_k(\mathcal{N})$ is a zero matrix and the k-th inequality in Theorem 3 cannot hold. ■

The NP-completeness of SPARSITY can be established by reducing the hitting set problem, HITTINGSET, to SPARSITY. Recall that a problem instance of HITTINGSET consists of a collection $\mathscr{C}$ of subsets of a finite set $\mathcal{U}$. A *hitting set* for $\mathscr{C}$ is a subset of $\mathcal{U}$ such that it contains at least one element from each subset in $\mathscr{C}$. The decision version of this problem is to determine whether we can find a hitting set with cardinality less than or equal to a given value.

Problem: HITTINGSET

Instance: A finite set $\mathcal{U}$, a collection $\mathscr{C}$ of subsets of $\mathcal{U}$ and an integer n.

Question: Is there a subset $\mathcal{S} \subseteq \mathcal{U}$ with cardinality less than or equal to n such that for each $\mathcal{C} \in \mathscr{C}$ we have $\mathcal{C} \cap \mathcal{S} \neq \emptyset$?

It is well known that HITTINGSET is NP-complete [36].

Example: Let $\mathcal{U} = \{1, 2, 3, 4, 5\}$,

$$\mathscr{C} = \{\{1, 2, 3\}, \{2, 3, 4\}, \{4, 5\}\}$$

and $n = 2$. We can check that the set $\{1, 4\}$ is a hitting set of size $n = 2$.

Theorem 9. SPARSITY *is NP-complete.*

Proof: We are going to reduce HITTINGSET to an instance of SPARSITY via Karp-reduction [38]. Let the cardinality of $\mathcal{U}$ be N. Label the elements of $\mathcal{U}$ by $1, 2, \ldots, N$. We define $\mathscr{C} \triangleq \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$, where K is the number of non-empty subsets in $\mathscr{C}$. For $k = 1, 2, \ldots, K$, form an N-vector $\mathbf{b}_k \in GF(q)^N$ with its i-th component equal to one if i is in $\mathcal{C}_k$ and zero otherwise, i.e., $\mathbf{b}_k$ is the characteristic vector of $\mathcal{C}_k$. Note that $\mathbf{b}_k \neq \mathbf{0}$ and $\mathscr{C} = \{supp(\mathbf{b}_1), supp(\mathbf{b}_2), \ldots, supp(\mathbf{b}_K)\}$. These $\mathbf{b}_k$'s correspond to the degenerate form of $\mathbf{B}_k$'s in Theorem 3 with only one row in $\mathbf{B}_k$. Let $\mathbf{C}_k$ be the encoding matrix of user k, whose row space is the null space of $\mathbf{B}_k$ and $\mathcal{I}$ be the innovative vector set defined in (1). In other words, any instance of HITTINGSET can be represented as an instance of SPARSITY in polynomial time.

It remains to show that there exists a hitting set $\mathcal{H}$ for $\mathscr{C}$ with $|\mathcal{H}| \leq n$ if and only if there exists an $\mathbf{x} \in \mathcal{I}$ with Hamming weight $|supp(\mathbf{x})| \leq n$. Given the $\mathbf{b}_k$'s obtained via the above reduction, suppose there exists $\mathbf{x} \in \mathcal{I}$ with $|supp(\mathbf{x})| \leq n$. By Theorem 3, we must have $\mathbf{b}_k \cdot \mathbf{x} \neq 0$ for all k's, which implies $supp(\mathbf{b}_k) \cap supp(\mathbf{x}) \neq \emptyset$ for all k's. The set $supp(\mathbf{x})$ is therefore a hitting set for the given instance. Conversely, given a hitting set $\mathcal{H}$ for $\mathscr{C}$ with $|\mathcal{H}| \leq n$, by definition $supp(\mathbf{b}_k) \cap \mathcal{H} \neq \emptyset$ for all k's. Therefore, $\mathbf{B}(\mathcal{H})$ has no zero rows. By Lemma 8, there exists an $\mathbf{x} \in GF(q)^N$ such that $supp(\mathbf{x}) \subseteq \mathcal{H}$. Hence, $|supp(\mathbf{x})| \leq n$.

As SPARSITY is verifiable in polynomial time, SPARSITY is in NP. Hence it is NP-complete. ■

Now we define the optimization version of SPARSITY as follows:

Problem: MIN SPARSITY

Instance: A positive integer n and K matrices with N columns, $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K$, over $GF(q)$, where $q \geq K$.

Objective: Find a vector $\mathbf{x} \in \mathcal{I}$ with minimum Hamming weight.

The minimum Hamming weight among all innovative vectors is called the *sparsity number*, and is denoted by ω. It is easy to see that if a polynomial-time algorithm can be found for solving the optimization version of SPARSITY, then that algorithm can be used for solving the decision version of SPARSITY in polynomial time as well. Therefore, MIN SPARSITY is NP-hard.

On the other hand, if K is held fixed, then there exist polynomial-time algorithms to solve MIN SPARSITY. It is proven in [39] and Section V-A that a K-sparse vector exists in $\mathcal{I}$, if $q \geq K$. By listing all vectors in $GF(q)^N$ with Hamming weight less than or equal to K, we can use Theorem 3 to check whether each of them is in $\mathcal{I}$. For each K-sparse encoding vector, we compute the matrix product $\mathbf{B}_k\mathbf{x}$ for $k = 1, 2, \ldots, K$. Each matrix product takes $O(NK)$ finite field operations. The total number of finite field operations for each candidate $\mathbf{x}$ is $O(NK^2)$. After checking all K-sparse encoding vectors, we can then find one with minimum Hamming weight. The number of non-zero vectors in $GF(q)^N$ with Hamming weight no more than K is equal to $\sum_{k=1}^{K} \binom{N}{k}(q-1)^k$. For fixed K and q, the summation is dominated by the largest term $\binom{N}{K}(q-1)^K$ when N is large, which is of order $O(N^K)$. The brute-force method can solve the problem with time complexity of $O(N^K(NK^2))$. As K is held fixed, MIN SPARSITY can be solved in polynomial time in N.

Let MIN HITTINGSET be the minimization version of the hitting set problem, in which we want to find a hitting set with minimum cardinality. The next result shows that MIN SPARSITY can be solved via MIN HITTINGSET based on the concept of Levin-reduction [38].

Theorem 10. MIN SPARSITY *can be reduced to* MIN HITTINGSET *via Levin-reduction.*

Proof: Given an instance of MIN SPARSITY, we determine $\tilde{\mathbf{b}}_k$ as in (4) for $k = 1, 2, \ldots, K$. Then we form the following instance of MIN HITTINGSET:

$$\begin{aligned} &\mathcal{U} = \{1, 2, \ldots, N\}, \\ &\mathscr{C} = \{supp(\tilde{\mathbf{b}}_1), supp(\tilde{\mathbf{b}}_2), \ldots, supp(\tilde{\mathbf{b}}_K)\}. \end{aligned}$$

Let $\mathcal{H}$ be a solution to the above instance. Then $\mathbf{B}(\mathcal{H})$ has no zero rows. By Lemma 8, there exists a vector $\mathbf{x}^* \in \mathcal{I}$ over $GF(q)$ with $supp(\mathbf{x}^*) \subseteq \mathcal{H}$. Such a vector $\mathbf{x}^*$ can be found by the SA algorithm in polynomial time.

We claim that there does not exist $\mathbf{x}' \in \mathcal{I}$ with Hamming weight $|supp(\mathbf{x}')| < |\mathcal{H}|$, and thus $|supp(\mathbf{x}^*)|$ must equal $|\mathcal{H}|$. Suppose there exists such a vector $\mathbf{x}'$. Lemma 8 implies that $\mathbf{B}(supp(\mathbf{x}'))$ has no zero rows, which in turn implies that $supp(\mathbf{x}') \cap supp(\tilde{\mathbf{b}}_k) \neq \emptyset$ for all k's. Then $supp(\mathbf{x}')$ would be a hitting set with cardinality strictly less than $|\mathcal{H}|$. A contradiction. ■

VI. NETWORK CODING ALGORITHMS

In this section, we present algorithms that generate sparse innovative encoding vectors for $q \geq K$. While for the binary transmission cases ($q = 2$), finding an innovative encoding vector may not always be possible, a modification of the algorithm is also proposed for handling such cases.

A. The Optimal Hitting Method

For $q \geq K$, we generate a sparest innovative vector in two steps. First we find an index set $\mathcal{N}$ with minimum cardinality, which determines the support of the innovative encoding vector. This is accomplished by solving the hitting set problem. Once $\mathcal{N}$ is found, the non-zero entries in the vector can be obtained by the SA algorithm.

The hitting set problem can be exactly solved by binary integer programming (BIP), formulated as follows:

$$\omega = \min_{\mathbf{y}} y_1 + y_2 + \ldots + y_N,$$

subject to

$$\mathbf{B}\mathbf{y} \geq \mathbf{1},$$

where

$$\mathbf{B} = \begin{bmatrix} \tilde{\mathbf{b}}_1 \\ \tilde{\mathbf{b}}_2 \\ \vdots \\ \tilde{\mathbf{b}}_K \end{bmatrix}$$

is a $K \times N$ binary matrix, $\mathbf{y} = (y_1, y_2, \ldots, y_N)$ is an N-dimensional binary vector, $\mathbf{1}$ is the K-dimensional all-one vector, and the inequality sign is applied component-wise.

To solve the above problem, we can apply any algorithm for solving BIP in general, for example the cutting plane method. We refer the readers to [40] for more details on BIP.

Example: Let $q = 3$, $K = 3$ and $N = 4$, and the orthogonal complements of V_1, V_2 and V_3 be given respectively by the row spaces of

$$\mathbf{B}_1 = \begin{bmatrix} 1 & 2 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}, \mathbf{B}_2 = \begin{bmatrix} 0 & 2 & 1 & 0 \end{bmatrix},$$
$$\mathbf{B}_3 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 2 \end{bmatrix}.$$

The vectors $\tilde{\mathbf{b}}_k$, for $k = 1, 2, 3$, are

$$\tilde{\mathbf{b}}_1 = [1\ 1\ 0\ 1],\ \tilde{\mathbf{b}}_2 = [0\ 1\ 1\ 0],\ \tilde{\mathbf{b}}_3 = [1\ 0\ 1\ 1].$$

The corresponding instance of MIN HITTINGSET is:

$$\mathcal{U} = \{1, 2, 3, 4\},\ \mathscr{C} = \{\{1, 2, 4\}, \{2, 3\}, \{1, 3, 4\}\}.$$

The solution to both MIN SPARSITY and MIN HITTINGSET can be obtained by solving the following BIP:

$$\min y_1 + y_2 + y_3 + y_4,$$

subject to

$$y_1 + y_2 + y_4 \geq 1,\ y_2 + y_3 \geq 1,\ y_1 + y_3 + y_4 \geq 1,$$
$$y_1, y_2, y_3, y_4 \in \{0, 1\}.$$

One optimal solution is $y_1 = y_2 = 1$ and $y_3 = y_4 = 0$. That means, the sparsity number, ω, is equal to two and $\mathcal{N} = \{1, 2\}$. Furthermore, according to Lemma 8, a 2-sparse innovative encoding vector can be found, for example, by the SA algorithm.

We call the above procedure for generating an innovative vector with minimum Hamming weight the *Optimal Hitting (OH) method.* We summarize the algorithm as follows:

The Optimal Hitting method (OH):

Input: For $k = 1, 2, \ldots, K$, full-rank $r_k \times N$ matrix $\mathbf{C}_k$ over $GF(q)$, where $q \geq K$ and $0 \leq r_k < N$.

Output: $\mathbf{x} = (x_1, x_2, \ldots, x_N) \in \mathcal{I}$ with minimum Hamming weight.

Step 0: Initialize $\mathbf{x}$ as the zero vector.

Step 1: For $k = 1, 2, \ldots, K$, obtain a basis of the null space of $\mathbf{C}_k$. Let $\mathbf{B}_k$ be the $(N - r_k) \times N$ matrix over $GF(q)$ whose j-th row is the j-th vector in the basis.

Step 2: For $k = 1, 2, \ldots, K$, let $\tilde{\mathbf{b}}_k$ be the component-wise logical-OR operations to the $N - r_k$ row vectors of $\mathbf{B}_k$. (Each non-zero component of $\mathbf{B}_k$ are being regarded as a "1" when taking the logical-OR operation.)

Step 3: Solve the corresponding MIN HITTINGSET as shown in Theorem 10 and return $\mathcal{H}$.

Step 4: For $k = 1, 2, \ldots, K$, choose a row vector from $\mathbf{B}_k$, say $\hat{\mathbf{b}}_k^T$, such that $supp(\hat{\mathbf{b}}_k) \cap \mathcal{H} \neq \emptyset$.

Step 5: Determine $\mathbf{x}(\mathcal{H})$ such that $\mathbf{x}(\mathcal{H}) \cdot \hat{\mathbf{b}}_k(\mathcal{H}) \neq \mathbf{0}$ for $k = 1, 2, \ldots, K$, by the SA algorithm.

Example continued: We solve the hitting set problem in Step 3 and obtain $\mathbf{y} = (1, 1, 0, 0)$. Hence, $\mathcal{N} = \{1, 2\}$. In step 4, we choose

$$\hat{\mathbf{b}}_1 = (1, 2, 0, 1), \hat{\mathbf{b}}_2 = (0, 2, 1, 0) \text{ and } \hat{\mathbf{b}}_3 = (1, 0, 0, 2).$$

In Step 5, we obtain $\mathcal{S}_1 = \{1, 3\}$ and $\mathcal{S}_2 = \{1, 2\}$. Note that both $|\mathcal{S}_1|$ and $|\mathcal{S}_2|$ are not equal to 3. We next set $x_1 = 1$, and choose x_2 such that

$$\hat{\mathbf{b}}_1 \cdot (1, x_2, 0, 0) \neq 0$$
$$\hat{\mathbf{b}}_2 \cdot (1, x_2, 0, 0) \neq 0.$$

We can choose $x_2 = 2$ to satisfy these two inequalities simultaneously. The vector $\mathbf{x} = (1, 2, 0, 0)$ is an innovative encoding vector of minimum Hamming weight.

B. The Greedy Hitting Method

Step 2 in the OH method requires solving an NP-hard problem. Therefore, some computationally efficient heuristics should be considered in practice. It is well known that MIN HITTINGSET can be solved efficiently and approximately by the following greedy approach [41]:

- Repeat until all sets of $\mathscr{C}$ are hit:
  - Pick the element that hits the largest number of sets that have not been hit yet.

In Step 3 of the OH method, the above greedy algorithm can be used to find approximate solutions. We call this modification the *Greedy Hitting* (GH) method.

Theorem 11. *The GH method is an H_n factor approximation algorithm for* MIN SPARSITY, *where H_n is the n-th harmonic number, defined as $H(n) \triangleq \sum_{k=1}^{n} \frac{1}{k}$.*

Proof: It is well known that the hitting set problem is just a reformulation of the set covering problem. Therefore, the greedy algorithm is an H_n factor approximation algorithm for MIN HITTINGSET, as well as for the set covering problem [37]. As shown in Theorem 10, MIN SPARSITY can be reduced to MIN HITTINGSET, and the sparsity number is equal to the cardinality of the minimum hitting set. Hence, GH is also an H_n factor approximation algorithm for MIN SPARSITY. ∎

Now we analyze the computational complexity of the GH method. The computation of each $\mathbf{B}_k$ can be reduced to the computation of the RREF of $\mathbf{C}_k$, which takes $O(N^3)$ arithmetic operations. However, if the encoding vectors are ω-sparse, we can adopt the dual-basis approach in obtaining $\mathbf{B}_k$ as in the appendix, and guarantee that each $\mathbf{B}_k$ can be obtained in $O(\omega N^2)$ times. The computational complexity of Step 1 is thus $O(\omega K N^2)$. Step 2 involves $O(KN^2)$ operations. If Step 3 is solved by the greedy algorithm, then it takes $O(KN^2)$ operations. Step 4 requires $O(K)$ operations. Step 5 involves the SA algorithm, which has a complexity of $O(K|\mathcal{H}|)$. Since $|\mathcal{H}| \leq N$, the overall complexity of GH is $O(\omega K N^2)$.

C. Solving Binary Equation Set for $q = 2$

The last step of the GH method involves solving a set of linear inequalities over $GF(q)$. However when $q = 2$, solving a linear inequality of the form $f(\mathbf{x}) \neq 0$ is equivalent to solving the linear equation $f(\mathbf{x}) = 1$. Based on this fact, we now propose a procedure which is called *Solving Binary Equation Set* (SBES), which modifies the SA algorithm so that it is applicable to the case where $q = 2$. Note that the same idea can be applied to cases where q is a prime power satisfying $2 < q < K$.

The heuristic is as follows. We want to find $\mathbf{x}$ such that $\mathbf{Ax} = \mathbf{1}$, where $\mathbf{A}$ is the coefficient matrix of the system of linear equations. The system may be inconsistent and has no solution. Nevertheless, we can disregard some equations and guarantee that at least $rank(\mathbf{A})$ equations are satisfied.

Solving Binary Equation Set Procedure (SBES):

Input: A $K \times N$ matrix $\hat{\mathbf{B}}$ over $GF(2)$ and $\mathcal{N}$, where $\mathcal{N} \subseteq \{1, 2, \dots, N\}$.

Output: $\mathbf{x} = (x_1, x_2, \dots, x_N) \in GF(2)^N$ with support in $\mathcal{N}$.

Step 0: Assign $\mathbf{z} = (z_1, z_2, \dots, z_{|\mathcal{N}|})$ as a zero vector.

Step 1: Delete columns of $\hat{\mathbf{B}}$ whose column indices are not in $\mathcal{N}$. Augment the resulting matrix by adding a K-dimensional all-one column vector to the right-hand side. Let the resulting matrix be denoted by $\mathbf{Q}$.

Step 2: Compute the row echelon form (REF) of $\mathbf{Q}$ and call it $\mathbf{Q}'$.

Step 3: Delete all zero rows in $\mathbf{Q}'$ and any row in $\mathbf{Q}'$ if it has a single "1" in the $(|\mathcal{N}| + 1)$-th entry. The resulting matrix is called $\mathbf{Q}''$. Let the number of pivots in $\mathbf{Q}''$ be ν, and let p_1, $p_2, \dots p_\nu$ be the column indices of the pivot in $\mathbf{Q}''$ listed in ascending order.

Step 4: Execute elementary row operations in $\mathbf{Q}''$ so that $\mathbf{Q}''$ is transformed into its row-reduced echelon form.

Step 5: Set the variables associated with the non-pivot columns to zero. For $i = 1, 2, \dots, \nu$, assign z_{p_i} the value of the i-th entry of the last column in $\mathbf{Q}''$.

Step 6: Assign values to the components of $\mathbf{x}$ such that $\mathbf{x}(\mathcal{N}) = \mathbf{z}$, and $x_i = 0$ if $i \notin \mathcal{N}$.

When applying the GH method to the case where $q = 2$, we replace the SA algorithm in Step 5 of the GH method by the SBES procedure. We call this modification GH with SBES.

Example: Consider $q = 2$, $K = 4$, $N = 5$, $\mathcal{H} = \{1, 3\}$ and

$$\hat{\mathbf{B}} = \begin{bmatrix} \hat{\mathbf{b}}_1^T \\ \hat{\mathbf{b}}_2^T \\ \hat{\mathbf{b}}_3^T \\ \hat{\mathbf{b}}_4^T \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \tag{5}$$

We extract the first and third rows of $\hat{\mathbf{B}}$ and augment it by the all-one column vector, $\mathbf{1}$,

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

In Step 2, we compute the REF of $\mathbf{Q}$

$$\mathbf{Q}' = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

The last row is an all-zero row, representing a redundant equation. The second last row contains a one in the third component, and zero elsewhere, implying that the original system of linear equations cannot be solved. In order to get a heuristic solution, we relax the system by deleting that row, and obtain

$$\mathbf{Q}'' = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

We have $p_1 = 1$ and $p_2 = 2$ in Step 3. The matrix $\mathbf{Q}''$ is already in its RREF. By Step 5, $z_1 = q''(1,3) = 1$ and $z_2 = q''(2,3) = 0$. Finally, we set $x_1 = z_1 = 1$, and $x_3 = z_2 = 0$. So $\mathbf{x} = (1, 0, 0, 0, 0)$ and $\mathbf{x}$ is innovative to all except the last user.

In this example, an innovative vector does exist, but GH with SBES fails to find it. The main reason is that the hitting set subproblem is formulated for the case where $q \geq K$. When q is small, there is no guarantee that a non-empty $\mathcal{I}$ must consist of a vector with support restricted in $\mathcal{H}$. Indeed, we will skip the greedy hitting procedure and simply let $\mathcal{H}$ be the index set of all packets, i.e., $\{1, 2, \dots, N\}$, then it is easy to check that with the same input, the SBES procedure returns $\mathbf{x} = (0, 0, 1, 1, 0)$, which is an innovative vector to all users. We call this modification *Full Hitting* (FH) with SBES, for the reason that the hitting set is chosen as the full index set of the packets. In general, FH with SBES produces encoding vectors that are innovative to more users than GH with SBES, at the expense of higher Hamming weights.

The SBES procedure is indeed the Gauss-Jordan elimination. So the total complexity is $O(NK^2)$. Note that the encoding vectors generated by GH with the SBES procedure may not be innovative when $K > q = 2$; however they are still innovative to a fraction of the K users.

VII. Performance Evaluation

In this section, we evaluate our proposed methods via simulations. We simulate a broadcast system in which a transmitter broadcasts N equal-size packets to K users via a erasure broadcast channel. The whole process is divided into two phases. The transmitter sends all packets one by one without network coding in the first phase. In the second phase, packets are encoded by the encoding vectors generated by the methods we concern and transmitted until all users received enough packets for recovering the N packets. The transmitter encodes packets based on the perfect feedback from the users. A user acknowledges a packet if the packet is received successfully and innovative to that user. The transmitter sets up K matrices $\mathbf{C}_k$, for $k = 1, 2 \dots, K$, where $\mathbf{C}_k$ records the encoding vectors of the encoded packets that have been acknowledged by user k after those previous transmissions. Since the packets are uncoded in the first phase, each row of $\mathbf{C}_k$ contains exactly one nonzero component before the retransmission. In

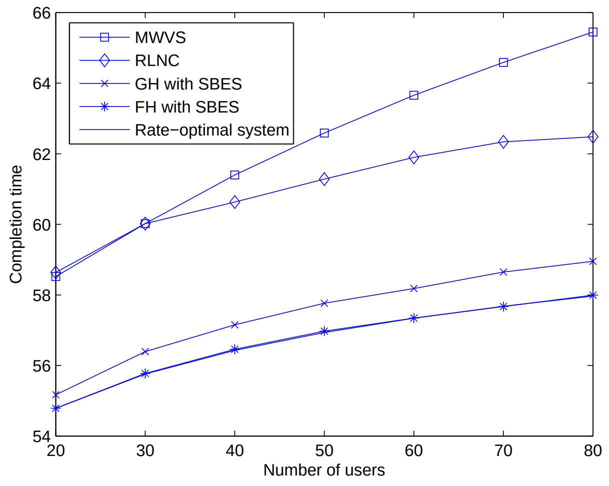

Fig. 1. The completion time for $q = 2$ and $N = 32$

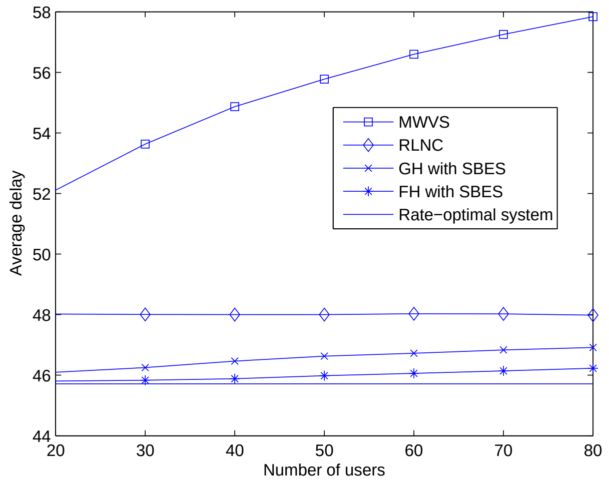

Fig. 2. The average delay for $q = 2$ and $N = 32$

our simulations, each simulation points involves 3000 random realizations and we assume that $N = 32$ and $P_e = 0.3$.

First, we examine the completion time and average delay of a two-phase broadcast system with different encoding schemes. The completion time is defined, from a system perspective, as the average of total number of transmissions for the transmitter to complete an N-packet transmission for all users. The average delay, from a user perspective, is defined as the individual completion time averaged over all users. We only consider the case where $q = 2$, since our proposed methods always generate innovative vectors for $q \geq K$ and are rate-optimal.

Figure 1 shows the completion time performance of the broadcast system with GH with SBES, FH with SBES, the maximum weight vertex search (MWVS) algorithm in [16], where MWVS generates instantly decodable packets, and binary random linear network code (RLNC). In the figure, we include the minimum completion time performance of a broadcast system in a erasure channel as a reference [5]. We remark that the completion time curves of GH and FH with a large finite field size are exactly the same as the optimal curve shown in the figure. For $q = 2$, simulation results show that GH and FH (with SBES) always outperform RLNC and MWVS. More importantly, it is found that the completion time performance of FH with SBES is nearly optimal.

Next, we consider the average delay performance. According to information theory, the best we can do in a (single-user) erasure channel with erasure probability P_e is to have $N/(1 - P_e) \approx 45.7$ transmissions on average. As mentioned before, GH and FH can achieve that limit when $q \geq K$, since they always generate innovative packets. Here we only consider the case where $q = 2$. In Figure 2, the horizontal line at 45.7 serves as a lower bound, which is not tight since innovative vectors do not always exist. Again, both GH and FH (with SBES) outperform RLNC and MWVS.

Intuitively, the delay performance of an encoding scheme is related to its ability to generate innovative vectors. Thus it is meaningful to examine how often the users may receive a non-innovative packet generated by an encoding scheme before they receive a complete set of packets, so that we have a better understanding on its delay performance. We examine the average number of non-innovative packets received by a user for different schemes when $q = 2$ in Figure 3. We observe that due to the random nature of RLNC, the average number of received non-innovative packets per user does not depend on the number of users in the system. This phenomenon agrees with the result in Figure 2 where the average delay performance of RLNC is independent of the number of users as well. In Figure 3, it is found that an encoding scheme with SBES always generates fewer non-innovative packets. It can be interpreted that the idea of SBES is to attempt finding a binary vector that is innovative to as many users as possible. Figure 3 also shows that FH with SBES generates the least amount of non-innovative packets among all encoding schemes that we considered. Note that the hitting set procedure in GH is the source of sparsity of the encoding vectors. However it may limit the choices for finding an innovative vector or solutions to inequality sets in SBES. Therefore the performance of FH is better.

Apart from the delay performance, we are also interested in the decoding complexity of different schemes. As the sparsity of encoding vectors may affect the decoding complexity, it is of interest to know how sparse the encoding vectors generated by different schemes could be. For RLNC with $q = 2$, since we generate each component of an encoding vector with equal probability of zero and one, the average Hamming weight is $N/2 = 16$, which is not shown in Figure 4. For other encoding schemes, it can be observed from Figure 4 that GH and MWVS generate sparse encoding vectors whose average Hamming weight is less than or around 6. The results of GH can be attributed to the hitting set minimization. FH with SBES tries to find vectors that are innovative to as many users as possible, without explicit consideration of the Hamming weight. Therefore, it can achieve a better delay performance at the expense of less sparsity.

Next we evaluate the decoding complexity in terms of the number of additions and multiplications performed. In our

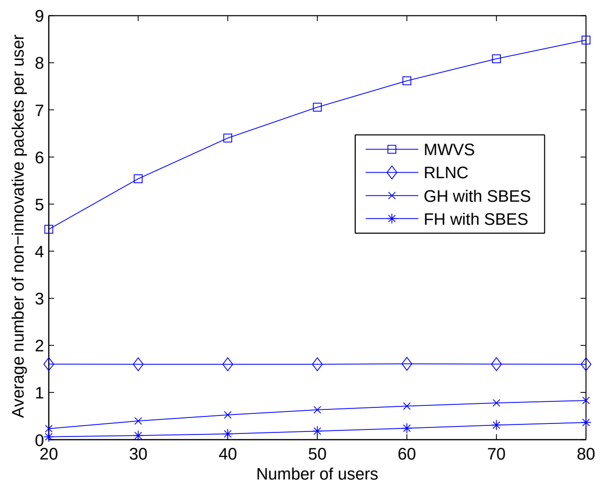

Fig. 3. The average number of non-innovative packets received by a user for $q = 2$ and $N = 32$

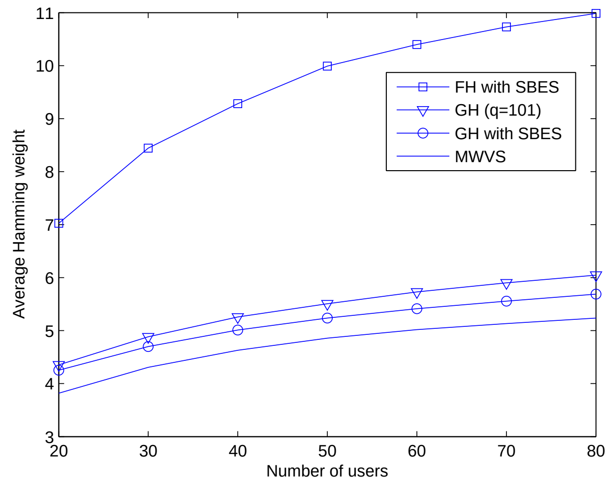

Fig. 4. The average Hamming weight

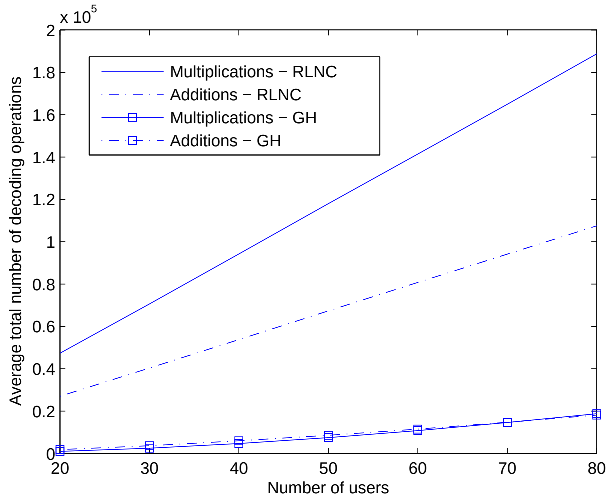

Fig. 5. The average total number of decoding operations for $q = 101$ and $N = 32$

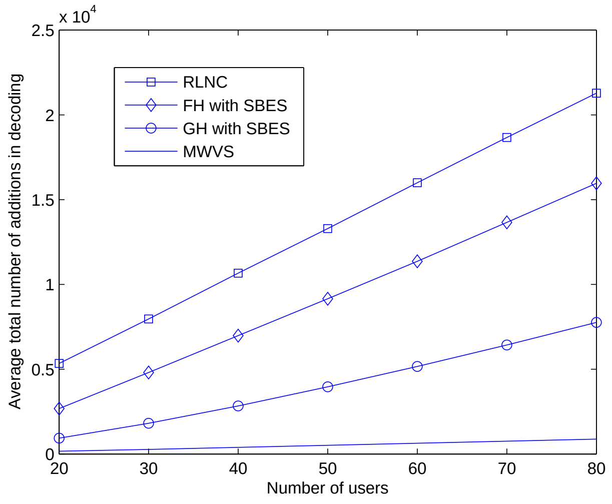

Fig. 6. The average total number of additions in decoding for $q = 2$ and $N = 32$

simulations, an addition operation involving two non-zero operands is counted. A multiplication operation is counted when none of the two operands is 1 or 0. The decoding algorithms in most of the previous work are basically Gauss-Jordan elimination except the instantly decodable schemes in [16], [19]. What we are proposing for decoding is also Gauss-Jordan elimination, but implemented for sparse matrix. Figures 5 and 6 show the average total number of operations for all users in the system with different schemes when $q = 2$ and $q = 101$, respectively. In Figures 4, 5 and 6, we observe that, in general, encoding vectors with larger average Hamming weight results in higher decoding complexity. Figure 5 shows that when $q = 101$, GH yields sparse innovative encoding vectors that results in significant reduction in both the average total number of additions and multiplications when compared with RLNC. Although both GH and RLNC can offer delay-optimal performance, GH involves fewer decoding operations, hence is a preferable choice for the optimal delay performance. In Figure 6, we examine the number of additions involved for different schemes when $q = 2$. We observe that, with instantly decodable MWVS, a receiver enjoys a low decoding complexity at the expense of larger delay. Furthermore, it is found that GH requires significantly fewer computations than RLNC. Especially, the decoding computations involved in GH with SBES is only 40 percent of the decoding computations of RLNC, though GH with SBES outperform RLNC in delay performance. As a result, GH is a good choice in terms of both delay performance and decoding complexity for $q = 2$. However if delay performance is the major concern, FH with SBES, providing nearly optimal delay performance with moderate decoding complexity, should be a good option. In short, both GH and FH offer promising choices for the trade-off between delay performance and decoding complexity in an erasure broadcast system.

VIII. Conclusions

In this paper, we adopt the computational approach to study the linear network code design problem for wireless broadcast systems. To minimize the completion time or to maximize the information rate, the concept of innovativeness plays an important role. While it is well known that innovative encoding vectors always exist when the finite field size, q, is greater than the number of users, K, we prove that the problem of determining their existence over the binary field is NP-complete. Its corresponding maximization version not only is hard to solve, but also is hard to approximate. Nevertheless, we propose a heuristic called FH with SBES, which is numerical shown to be nearly optimal under our simulation settings.

Sparsity of a network code is another issue we have addressed. When $q \geq K$, we show that the minimum Hamming weight within the set of innovative vectors is bounded above by K. To find a vector that achieves the minimum weight, however, is proven to be NP-hard via the reduction from the hitting set problem. An exact algorithm based on BIP is described, and a polynomial-time approximation algorithm based on the greedy approach is constructed.

The performance of our proposed algorithms has been evaluated by simulations. When $q \geq K$, our proposed algorithm is rate-optimal and is effective in reducing decoding complexity. When $q = 2$, our proposed algorithms are able to strike a proper balance between completion time and decoding complexity.

Acknowledgements: The authors would like to thank Prof. Wai Ho Mow and Dr. Kin-Kwong Leung for their stimulating discussions.

Appendix

In this appendix, we illustrate how to compute a basis of the null space *incrementally*. In the application to the broadcast system we consider in this paper, the rows of $\mathbf{C}$ are given one by one. A row is revealed after an innovative packet is received. Given an $r \times N$ matrix $\mathbf{C}$ over $GF(q)$, recall that our objective is to find a basis for the null space of $\mathbf{C}$. The idea is as follows. We first extend $\mathbf{C}$ to a non-singular $N \times N$ matrix by appending $N - r$ row vectors. Let the resulting square matrix be $\tilde{\mathbf{C}}$. Let $\tilde{\mathbf{B}}$ be the inverse of $\tilde{\mathbf{C}}$. By the very definition of matrix inverse, the last $N - r$ columns of $\tilde{\mathbf{B}}$ is a basis for the null space of $\mathbf{C}$.

We proceed by induction. The algorithm is initialized by setting $\tilde{\mathbf{C}} = \tilde{\mathbf{B}} = \mathbf{I}_N$. We will maintain the property that $\tilde{\mathbf{C}}^{-1} = \tilde{\mathbf{B}}$.

Suppose that the first r rows of $\tilde{\mathbf{C}}$ are the encoding vectors received by a user, and $\tilde{\mathbf{C}} = \tilde{\mathbf{B}}^{-1}$. We let $\mathbf{c}_i^T$ be the i-th row of $\tilde{\mathbf{C}}$ and $\mathbf{b}_j$ be the j-th column of $\tilde{\mathbf{B}}$. When a packet arrives, we can check whether it is innovative by taking the inner product of the encoding vector of the new packet, say $\mathbf{w}$, with $\mathbf{b}_{r+1}$, $\mathbf{b}_{r+2}, \ldots, \mathbf{b}_N$. According to Theorem 3, it is not innovative if and only if all such inner products are zero.

Consider the case that $\mathbf{w}$ is innovative. Permute the columns of $\tilde{\mathbf{B}}$, if necessary, to ensure that $\mathbf{w}^T\mathbf{b}_{r+1} \neq 0$. This can always be done, since $\mathbf{w}$ cannot be orthogonal to all the last $N - r$ columns of $\tilde{\mathbf{B}}$. Permute the rows of $\tilde{\mathbf{C}}$ accordingly, so as to ensure that $\tilde{\mathbf{C}}^{-1} = \tilde{\mathbf{B}}$. Then, we modify $\tilde{\mathbf{C}}$ by updating its $(r + 1)$-st row to $\mathbf{w}^T$. This operation can be expressed algebraically by

$$\tilde{\mathbf{C}} \longleftarrow \tilde{\mathbf{C}} + \mathbf{e}_{r+1}(\mathbf{w} - \mathbf{c}_{r+1})^T,$$

where $\mathbf{e}_{r+1}$ is the column vector with the $(r+1)$-st component equal to 1 and zero otherwise. The matrix $\mathbf{e}_{r+1}(\mathbf{w}-\mathbf{c}_{r+1})^T$ is a rank-one matrix, with the $(r+1)$-st row equal to $(\mathbf{w}-\mathbf{c}_{r+1})^T$, and zero everywhere else. The inverse of $\tilde{\mathbf{C}}+\mathbf{e}_{r+1}(\mathbf{w}-\mathbf{c}_{r+1})^T$ can be computed efficiently by the Sherman-Morrison formula [42] [43, p.18],

$$\begin{aligned}
&(\tilde{\mathbf{C}} + \mathbf{e}_{r+1}(\mathbf{w} - \mathbf{c}_{r+1})^T)^{-1} \\
&= \tilde{\mathbf{C}}^{-1} - \frac{\tilde{\mathbf{C}}^{-1}\mathbf{e}_{r+1}(\mathbf{w} - \mathbf{c}_{r+1})^T\tilde{\mathbf{C}}^{-1}}{1 + (\mathbf{w} - \mathbf{c}_{r+1})^T\tilde{\mathbf{C}}^{-1}\mathbf{e}_{r+1}} \\
&= \tilde{\mathbf{C}}^{-1} - \frac{\mathbf{b}_{r+1}(\mathbf{w} - \mathbf{c}_{r+1})^T\tilde{\mathbf{C}}^{-1}}{\mathbf{w}^T\mathbf{b}_{r+1}} \\
&= \tilde{\mathbf{C}}^{-1} - \frac{\mathbf{b}_{r+1}(\mathbf{w}^T\tilde{\mathbf{C}}^{-1} - \mathbf{e}_{r+1}^T)}{\mathbf{w}^T\mathbf{b}_{r+1}}. \qquad (6)
\end{aligned}$$

We have used the facts that $\tilde{\mathbf{C}}^{-1}\mathbf{e}_{r+1} = \mathbf{b}_{r+1}$ and $\mathbf{c}_{r+1}^T\tilde{\mathbf{C}}^{-1} = \mathbf{e}_{r+1}^T$ in the above equations. The denominator of the fraction in (6) is a non-zero scalar by construction, so that division of zero would not occur. We update $\tilde{\mathbf{B}}$ by the expression in (6). Note that if $\mathbf{w}$ is ω-sparse, the multiplication of $\mathbf{w}^T$ and $\tilde{\mathbf{C}}^{-1}$ in (6) can be done in $O(\omega N)$ times.

References

[1] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. on Inf. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[2] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[3] A. Eryilmaz, A. Ozdaglar, and M. Médard, "On delay performance gains from network coding," in *40th Annual Conf. on Inf. Sci. and Systems*, Princeton, Mar. 2006, pp. 864–870.

[4] M. Ghaderi, D. Towsley, and J. Kurose, "Reliability gain of network coding in lossy wireless networks," in *IEEE Int. Conf. on Comp. Comm. (INFOCOMM '08)*, Phoenix, Apr. 2008, pp. 2171–2179.

[5] J. Heide, M. V. Pedersen, F. H. P. Fitzek, and T. Larsen, "Network coding for mobile devices – systematic binary random rateless codes," in *IEEE Int. Conf. Comm. Workshops*, Dresden, Jun. 2009, pp. 1–6.

[6] E. Drinea, C. Fragouli, and L. Keller, "Delay with newtork coding and feedback," in *Proc. IEEE Int. Symp. Inform. Theory*, Seoul, Jun. 2009, pp. 844–848.

[7] M. Durvy, C. Fragouli, and P. Thiran, "Towards reliable broadcasting using ACKs," in *Proc. IEEE Int. Symp. Inform. Theory*, Nice, Jun. 2007, pp. 1156–1160.

[8] M. Luby, "LT codes," in *Proceedings of the 43rd annual IEEE symposium on foundations of computer science*, Nov. 2002, pp. 271–282.

[9] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, 2006.

[10] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, S. Jun, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[11] S. Jaggi, P. Sanders, P. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 1973–1982, Jun. 2005.

[12] J. K. Sundararajan, D. Shah, and M. Médard, "Online network coding for optimal throughput and delay – the three-receiver case," in *Int. Symp. on Inform. Theory and its Application (ISITA)*, Auckland, Dec. 08, pp. 1–6.

[13] ——, "A feedback-based adaptive broadcast coding scheme for reducing in-order delivery delay," in *Network Coding, Theory and Application (NetCod)*, Lausanne, Jun. 2009, pp. 1–6.

[14] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless broadcast using network coding," *IEEE Trans. on Veh. Tech.*, vol. 58, no. 2, pp. 914–925, Feb. 2009.

[15] L. Lu, M. Xiao, M. Skoglund, L. K. Rasmussen, G. Wu, and S. Li, "Efficient network coding for wireless broadcasting," in *IEEE Wireless Comm. and networking conf. (WCNC '10)*, Sydney, Apr. 2010, pp. 1–6.

[16] S. Sorour and S. Valaee, "On minimizing broadcast completion delay for instantly decodable network coding," in *IEEE Int. Conf. Comm. (ICC '10)*, May 2010, pp. 1–5.

[17] L. Keller, E. Drinea, and C. Fragouli, “Online broadcasting with network coding,” in *Proc. of the Fouth Workshop on Network Coding, theory, and applications (NetCod ’08)*, Hong Kong, Jan. 2008, pp. 68–73.

[18] R. Costa, D. Munaretto, J. Widmer, and J. Barros, “Informed network coding for minimum decoding delay,” in *Fifth IEEE Int. Conf. on Mobile Ad-hoc and Sensor Systems (MASS ’08)*, Atlanta, 2008.

[19] P. Sadeghi, D. Traskov, and R. Koetter, “Adaptive network coding for broadcast channels,” in *Fifth Workshop on Network Coding, Theory and Applications (NetCod ’08)*, 2009, pp. 1–6.

[20] P. Sadeghi, R. Shams, and D. Traskov, “An optimal adaptive network coding scheme for minimizing decoding delay in broadcast erasure channel,” *EURASIP J. on Wireless Comm. and Networking*, vol. 2010, 2010, doi:10.1155/2010/61816.

[21] D. H. Wiedemann, “Solving sparse linear equations over finite fields,” *IEEE Trans. Inf. Theory*, vol. 1, no. 32, pp. 54–62, Jan. 1986.

[22] E. Kaltofen and B. D. Saunders, “On Wiedemann’s method of solving sparse linear system,” in *the 9th Int. Symp. on Applied Algebra, Algebraic Algorithms and error-Correcting Codes (AAECC*, T. M. H. F. Mattson and T. R. N. Rao, Eds., vol. 539, 1991.

[23] D. Coppersmith, “Solving linear equations over GF(2): the block Lanczos algorithm,” *Linear algebra and its applications*, vol. 192, pp. 33–60, 1993.

[24] A. Barg, *Handbook of coding theory*. Elsevier Science, 1998, vol. 1, ch. Complexity issues in coding theory, pp. 649–754.

[25] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg, “On the inherent intractability of certain coding problems,” *IEEE Trans. Inf. Theory*, vol. 24, no. 3, pp. 348–386, May 1978.

[26] A. Vardy, “The intractability of computing the minimum distance of a code,” *IEEE Trans. Inf. Theory*, vol. 43, no. 6, pp. 1757–1766, Nov. 1997.

[27] A. R. Lehman and E. Lehman, “Complexity classification of network information flow problems,” in *Proc. of the 15th annual ACM-SIAM Symp. on Discrete algorithms (SODA)*, 2004, pp. 142–150.

[28] N. J. A. Harvey, D. R. Karger, and K. Murota, “Deterministic network coding by matrix completion,” in *Proc. of the 16th Annual ACM-SIAM Symp. on Discrete Algorithm (SODA)*, Jan. 2005, pp. 489–498.

[29] N. J. A. Harvey, D. R. Karger, and S. Yekhanin, “The complexity of matrix completion,” in *Proc. of the 17th Annual ACM-SIAM Symp. on Discrete Algorithm (SODA)*, Jan. 2006, pp. 1103–1111.

[30] 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.

[31] M. Langberg, A. Sprintson, and J. Bruck, “The encoding complexity of network coding,” *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2386–2397, Jun. 2006.

[32] ——, “Network coding: a computational perspective,” *IEEE Trans. Inf. Theory*, vol. 55, no. 1, pp. 147–157, Jan. 2009.

[33] S. Y. El Rouayheb, M. A. R. Chaudhry, and A. Sprintson, “On the minimum number of transmissions in single-hop wireless coding networks,” in *IEEE Information theory workshop (ITW)*, Lake Tahoe, Sep. 2007, pp. 120–125.

[34] M. A. R. Chauhry, Z. Asad, A. Sprintson, and M. Langberg, “On the complementary index coding problem,” in *Proc. IEEE Symp. Int. on Inform. Theory*, St. Petersburg, Aug. 2011, pp. 306–310.

[35] N. Milosavljevic, S. Pawar, S. El Rouayheb, and M. Gastpar, “Optimal deterministic polynomial-time data exchange for omniscience,” arXiv:1108.6046, Aug. 2011.

[36] M. R. Garey and D. S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman and Company, 1979.

[37] V. V. Vazirani, *Approximation Algorithms*. Springer, 2003.

[38] O. Goldreich, *P, NP, and NP-Completeness: The Basics of Computational Complexity*. Cambridge University Press, 2010.

[39] H. Y. Kwan, K. W. Shum, and C. W. Sung, “Generation of innovative and sparse encoding vectors for broadcast systems with feedback,” to appear in *Proc. IEEE Int. Symp. Inform. Theory*, St. Petersburg, Jul. 2011.

[40] A. Schrijver, *Theory of linear and integer programming*. John Wiley & Son, 1986.

[41] U. Feige, “A threshold of log n for approximating set cover,” *Journal of ACM 45*, pp. 634–652, 1998.

[42] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipe*, 3rd ed. Cambridge University Press, 2007, ch. 2.7.1.

[43] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge University Press, 1985.