# Linear Network Coding for Erasure Broadcast Channel With Feedback: Complexity and Algorithms

Chi Wan Sung, *Member, IEEE*, Kenneth W. Shum, *Member, IEEE*, Linyu Huang, *Member, IEEE*, and Ho Yuet Kwan

***Abstract*— This paper investigates the linear network coding problem for erasure broadcast channel with user feedback. An innovative linear network code is shown to be uniformly optimal for the system. In general, determining the existence of innovative packets is proved to be NP-complete. When the finite field size is larger than the number of users, innovative packets always exist and the problem of finding an innovative encoding vector with smallest Hamming weight is considered. The corresponding decision problem is shown to be NP-complete. Optimal and approximate network coding algorithms for maximizing the sparsity of encoding vectors are designed.**

***Index Terms*— Erasure broadcast channel, innovative encoding vector, sparse network code, computational complexity.**

## I. Introduction

In an erasure broadcast channel, a transmitter needs to send a common message reliably to $K$ users. In each channel use, each user either receives the channel input exactly or experiences an erasure. For this problem, linear network coding [1], [2] has been shown to be a promising solution [3]–[6]. The idea is that a transmitter broadcasts encoded packets that are obtained by linearly 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 broadcast 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. It is intuitively clear that if all the encoded packets are innovative, the *completion time*, measured in terms of number of packet transmissions, is minimized.

Linear network codes for broadcasting can be generated with or without feedback. For example, LT codes [8], Raptor codes [9] and random linear network codes (RLNC) [10] can be used without feedback. These codes, however, do not necessarily produce innovative packets. With feedback, it is suggested in [7] that the Jaggi-Sanders algorithm [11] could be used. While this algorithm is able to find innovative encoding vectors for $q \geq K$, its encoding and decoding complexities are relatively high, as it is not specially 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. Besides, some works focus on minimizing *decoding delay*, where a unit of decoding delay is incurred when a successfully received packet is neither innovative nor instantly decodable [17]–[20].

The excellent performance of linear network coding encourages researchers to consider its practicality. In particular, decoding complexity is an important issue, as mobile devices typically have low computation speed and limited energy. While RLNC is optimal in completion time and does not require any feedback, its high decoding complexity makes it less attractive for practical deployment. One possible way to reduce decoding complexity is to use sparse encoding vectors. For example, a fast algorithm by Wiedemann for solving a system of sparse linear equations can be used for decoding [21]. If all the encoding vectors are $w$-sparse, which means that their Hamming weights are at most $w$, then the complexity for solving an $N \times N$ linear system can be reduced from $O(N^3)$ using Gaussian elimination to $O(wN^2)$ [22]. The Wiedemann algorithm is useful when $N$ is large. When $N$ is moderate, we can implement some sparse representation of matrices, so that even if the usual Gaussian elimination is used,

Manuscript received December 16, 2013; revised January 13, 2016; accepted February 14, 2016. Date of publication March 1, 2016; date of current version April 19, 2016. This work was supported in part by the Research Foundation for Youth Scholars of Sichuan University under Grant 2015SCU11066, in part by the Research Grants Council, Hong Kong, under Project CityU 121713, in part by the University Grants Committee, Hong Kong, under Project AoE/E-02/08. This paper was presented at the 2011 IEEE International Symposium on Information Theory and the 2011 IEEE International Symposium on Network Coding.

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

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

L. Huang was with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong. He is now with the College of Electronics and Information Engineering, Sichuan University, Chengdu 610065, China (e-mail: lyhuang@scu.edu.cn).

H. Y. Kwan was with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong. He is now with the Division of Applied Science and Technology, Community College of City University, Hong Kong (e-mail: hykwan@cityu.edu.hk).

Communicated by M. Langberg, Associate Editor for Coding Theory.

Digital Object Identifier 10.1109/TIT.2016.2536612

the number of additions and multiplications required can be reduced. For other fast methods for solving linear equations over finite fields, we refer the readers to [23] and [24].

Minimizing the completion time and reducing the decoding complexity are equally important in linear network code design for erasure broadcast channels. However, the innovativeness of encoding vectors together with their sparsity has not been thoroughly studied. Given the encoded packets that have been received by the users, the generation of new encoding vectors which are both sparse and innovative is a challenging problem. In this paper, we address this issue, and the main results are listed below:

- When $q < K$, determining the existence of innovative encoding vectors is NP-complete.
- When $q \geq K$, determining whether there is an innovative encoding vector whose Hamming weight is smaller than a given number is NP-complete. An optimal algorithm and an approximate algorithm for maximizing the sparsity of an innovative encoding vector are presented. Both algorithms are able to generate $K$-sparse encoding vectors.

The rest of this paper is organized as follows. We review the literature on complexity in network coding in Section II and some useful notions in complexity theory in Section III. In Section IV, the system model is introduced and the problem is formulated. In Section V, we show that innovative linear network code is uniformly optimal, a concept to be defined later. In Section VI, we characterize the set of innovative encoding vectors and prove that the determination of the existence of an innovative vector for $q < K$ is NP-complete. In Section VII, the sparsity issue is considered. After showing that $K$-sparse innovative vectors always exist if $q \geq K$, we investigate the SPARSITY problem and prove that it is NP-complete. In Section VIII, we present a systematic way to solve MAX SPARSITY using binary integer programming. A polynomial-time approximation algorithm is also constructed. Finally, conclusions are drawn in Section IX.

## II. Related Works on the Complexity Classes of Network Coding and Index Coding Problems

A considerable amount of research has been done on the complexity issues in conventional coding theory (see the survey in [25] for example). For instance, it is shown in [26] and [27] 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. In this section, we give a short survey of existing results on complexity in network coding.

There are several intractable problems in network coding theory. For example, determining the minimum alphabet size for a single-source information flow problem is NP-hard [28]. It is also shown in [28] that for multi-source information flow problem, determining whether we can find a linear network coding solution is NP-complete. Approximating the number of nodes whose demand can be satisfied by a network code is NP-hard [29]. The problem of approximating the capacity of network coding is also a hard problem [30]. The problem of minimizing the number of encoding nodes is proved to be NP-hard in [31] and [32].

In [33], Harvey *et al.* relate the construction of linear network codes to matrix completion. Given a collection of mixed matrices, whose entries belong to a finite field or a set of distinct indeterminates, the problem is to determine whether we can substitute particular values for the indeterminates, such that each of the resulting matrix has maximum rank. If the field size is larger than the number of mixed matrices, it is shown in [33] that a solution to the matrix completion problem can be computed in polynomial time. If the field size is less than or equal to the number of mixed matrices, the problem is NP-complete. Further results on the complexity class of the matrix completion problem is studied in [34].

The index coding problem [35] is closely related to the network coding problem. The objective of index coding is to satisfy the demands of several receivers, each of which has some prior side information, by the least number of broadcast packets. The broadcast is assumed to be noiseless, and the sender is assumed to know all the side information possessed by the receivers. Bar-Yossef *et al.* show that, if there are $n$ data packets and $n$ receivers, and the $i$-th receiver wants the $i$-th packet for $1 \leq i \leq n$, the minimum number of packet transmissions is characterized by the solution to the min-rank of an associated graph, and it is shown by Peeters in [36] that computing the min-rank of a general graph is NP-hard. Tehrani and Dimakis show that in the linear index coding problem over the binary field, computing the encoding coefficients is an NP-hard problem, even if the problem instance is known to be solvable by three packet transmissions [37]. Some variations of index coding can be found in [38] and [39].

## III. Useful Notions in Complexity Theory

Before presenting the broadcast problem, we first define some useful notions in complexity theory, which will be used in this paper. The following definitions are taken from [40]:

*Definition 1:* Let $\{0,1\}^*$ denote the set of all binary strings, and $S$ be a subset of $\{0,1\}^*$. A function $f : \{0,1\}^* \to \{0,1\}$ is said to solve the *decision problem* of $S$ if for every binary string $x$ it holds that $f(x) = 1$ if and only if $x \in S$.

*Definition 2:* For a given $R \subseteq \{0,1\}^* \times \{0,1\}^*$, let $R(x) \triangleq \{y : (x,y) \in R\}$ denote the set of solutions for the binary string $x$. A function $f : \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$ is said to solve the *search problem* of $R$ if for every $x$ the following holds:

$$f(x) \begin{cases} \in R(x) & \text{if } R(x) \neq \emptyset, \\ = \perp & \text{otherwise.} \end{cases}$$

Note that a minimization problem can be regarded as a search problem. By definition, a minimization problem is associated with a value function $V : \{0,1\}^* \times \{0,1\}^* \to \mathbb{R}$. Given $x$, the task is to find $y$ such that $(x,y) \in R$ and $V(x,y)$ is the minimum value of $V(x,y')$ for all $y' \in R(x)$.

The following two definitions concern reductions between two problems:

*Definition 3:* For $S$, $S' \subseteq \{0,1\}^*$, a polynomial-time computable function $f : S \to S'$ is called a *Karp-reduction* of $S$ to $S'$ if, for every binary string $x$, it holds that $x \in S$ if and only if $f(x) \in S'$.

*Definition 4:* For $R,\ R' \subseteq \{0,1\}^* \times \{0,1\}^*$, a pair of polynomial-time computable functions,

$$f : \{0,1\}^* \to \{0,1\}^*,$$
$$g : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*,$$

is called a *Levin-reduction* of $R$ to $R'$ if the function $f$ is a Karp-reduction of $S_R \triangleq \{x : \exists y \text{ s.t. } (x,y) \in R\}$ to $S_{R'} \triangleq \{x' : \exists y' \text{ s.t. } (x',y') \in R'\}$, and for every $x \in S_R$ and $(f(x), y') \in R'$ it holds that $(x, g(x,y')) \in R$.

## IV. System Model and Problem Formulation

Consider a single-hop wireless broadcast system, in which there are one source and $K$ users. The source wants to send $N$ data packets to all the $K$ users. We view each packet as a symbol from an alphabet set $\mathcal{X}$ of size $q$. In other words, the source wants to broadcast $N$ symbols, $P_1, P_2, \ldots, P_N \in \mathcal{X}$. We assume that they are independent random variables, each of which is drawn uniformly at random from $\mathcal{X}$.

We model the transmission as a time-slotted broadcast erasure channel. In time slot $t$, a symbol $X_t \in \mathcal{X}$ is transmitted by the source. The channel output observed by user $k$, denoted by $Y_{k,t}$, is either the same as $X_t$ or equal to a special erasure symbol $e$. A time slot is called a non-erasure slot of user $k$ at time $t$ if $X_t = Y_{k,t}$, and is called an erasure slot of user $k$ otherwise. The channel dynamics is modeled by a stochastic sequence,

$$\Psi \triangleq \big((S_{1,t}, S_{2,t}, \ldots, S_{K,t})\big)_{t=1,2,3,\ldots},$$

where $S_{k,t}$ equals one if $Y_{k,t} = X_t$ or zero if $Y_{k,t} = e$. Assume that $S_{k,t}$'s are all independent of the source symbols $P_1, P_2, \ldots, P_N$. For $k = 1, 2, \ldots, K$, we let $N_k(t, \Psi)$ be the number of non-erasure slots of user $k$ in the first $t$ time slots. We let $\Psi_T$ be the truncated sequence obtained from $\Psi$ by preserving the $KT$ random variables in the first $T$ time slots. After every slot $t$, user $k$ broadcasts $S_{k,t}$ via a control channel without delay and error. We assume that after time $\tau$, the source and all users have the knowledge of $\Psi_\tau$.

Define $\mathcal{Y} \triangleq \mathcal{X} \cup \{e\}$. An $(N, K, q)$ broadcast code is defined by encoding functions

$$f_t : \mathcal{X}^N \times \{0,1\}^{K(t-1)} \to \mathcal{X}, \tag{1}$$

and decoding functions

$$g_{k,t} : \mathcal{Y}^t \times \{0,1\}^{Kt} \to \mathcal{X}^N, \tag{2}$$

where $k = 1, 2, \ldots, K$ and $t = 1, 2, \ldots$.

Given a broadcast code and a realization of the channel dynamics $\Psi$, user $k$ is said to have *download delay* $T_k(\Psi)$ if it is the smallest value of $t$ such that decoding is successful, that is,

$$g_{k,t}(Y_{k,1}, Y_{k,2}, \ldots, Y_{k,t}, \Psi_t) = (P_1, P_2, \ldots, P_N). \tag{3}$$

If decoding is never successful, then we let the download delay be infinity.

The following result gives a lower bound of the download delay of each user:

*Theorem 1: Given any $(N, K, q)$ broadcast code and any channel realization $\Psi$, we have $T_k(\Psi) > \tau$ for all $\tau$ such that $N_k(\tau, \Psi) < N$, for all $k = 1, 2, \ldots, K$.*

*Proof:* Consider a time index $\tau$, where $N_k(\tau, \Psi) < N$. Let $a_1, a_2, \ldots, a_{N_k(\tau)} \le \tau$ be the indices of time slots at which user $k$ experiences no erasure, and let $Y_k \triangleq (Y_{k,a_1}, Y_{k,a_2}, \ldots, Y_{k,a_{N_k(\tau)}})$. Note that

$$\begin{aligned}
&H(P_1, P_2, \ldots, P_N | Y_{k,1}, Y_{k,2}, \ldots, Y_{k,\tau}, \Psi_\tau) \\
&= H(P_1, P_2, \ldots, P_N | Y_k) \\
&= H(P_1, P_2, \ldots, P_N) - \big[H(Y_k) - H(Y_k | P_1, P_2, \ldots, P_N)\big] \\
&\ge H(P_1, P_2, \ldots, P_N) - H(Y_k) \\
&= N \log_2 |\mathcal{X}| - H(Y_k) \\
&\ge N \log_2 |\mathcal{X}| - N_k(\tau, \Psi) \log_2 |\mathcal{X}| \\
&= (N - N_k(\tau, \Psi)) \log_2 q \\
&> 0
\end{aligned}$$

Therefore, the probability that the decoding condition in (3) holds must be strictly less than one. In other words, the download delay of user $k$, $T_k(\Psi)$, must be strictly greater than $\tau$, for all $k$'s. □

*Definition 5:* An $(N, K, q)$ broadcast code is said to be *uniformly optimal* if for any channel realization $\Psi$ and $k = 1, 2, \ldots, K$,

$$T_k(\Psi) = \min\{\tau : N_k(\tau, \Psi) = N\}. \tag{4}$$

If the set $\{\tau : N_k(\tau, \Psi) = N\}$ is empty, we define the minimum as infinity.

The existence of uniformly optimal broadcast code will be investigated in the next two sections.

## V. Linear Network Code

In this paper, we focus on the use of linear network code. The alphabet $\mathcal{X}$ is identified with the finite field $GF(q)$ of size $q$, for some prime power $q$. We define linear network codes formally below:

*Definition 6:* An $(N, K, q)$ broadcast code is said to be a *linear network code* if its encoding functions can be expressed as a linear function of the source packets:

$$f_t(P_1, P_2, \ldots, P_N, \Psi_{t-1}) = x_1 P_1 + x_2 P_2 + \cdots + x_N P_N, \tag{5}$$

where $x_1, x_2, \ldots, x_N \in GF(q)$ are determined by $\Psi_{t-1}$, and the addition and multiplication operations are defined over $GF(q)$.

The vector $\mathbf{x} \triangleq (x_1, x_2, \ldots, x_N) \in GF(q)^N$, as expressed in (5), is called the *encoding vector* of the packet transmitted in slot $t$. Throughout this paper, all vectors are assumed to be column vectors, and we use parenthesis and commas when its components are listed horizontally.

For practical applications, the transmitter can put the encoding vector in the header of the encoded packet. While that incurs some transmission overhead, it can relax the requirement specified in the previous section that every user can listen to the feedback information from all other users.

In other words, the decoding function of user $k$ in (2) can be changed to

$$g_{k,t} : \mathcal{Y}^t \times GF(q)^{Nt} \to \mathcal{X}^N, \tag{6}$$

assuming that the decoder knows the encoding vectors of its received packets.

The *support* of the vector $\mathbf{x}$, denoted by $\text{supp}(\mathbf{x})$, is the set of indices of the non-zero components in $\mathbf{x}$, i.e., $\text{supp}(\mathbf{x}) \triangleq \{i : x_i \neq 0\}$. The *Hamming weight* of $\mathbf{x}$ is defined as the cardinality of $\text{supp}(\mathbf{x})$. An encoding vector that has Hamming weight less than or equal to $w$ is said to be $w$-sparse.

Note that a transmitted packet brings new information to a user if and only if its entropy conditioned on the previously received packets by that user is greater than zero, or equivalently, the new packet is not a function of the previously received packets. In linear-algebraic terms, the condition is that the encoding vector of the new packet does not lie within the span of all previously received encoding vectors of that user.

Suppose that user $k$, for $k = 1, 2, \ldots, K$, 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 transposes of 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. Note that $r_k$ is the rank of $\mathbf{C}_k$. For $k = 1, 2, \ldots, K$, let $V_k$ be the row space of $\mathbf{C}_k$. A vector $\mathbf{x}$ is innovative if it does not belong to $V_k$ for any $k$. Given $K$ encoding matrices $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K$, the set of all innovative encoding vectors is given by

$$\mathcal{I} \triangleq GF(q)^N \setminus \bigcup_{k=1}^{K} V_k. \tag{7}$$

*Definition 7:* A linear network code is said to be *innovative* if for any channel realization $\Psi$, its encoded packet at time $t$ is innovative to all users who have not successfully decoded the source packets yet, that is, those users with indices in $\{k : T_k(\Psi) \geq t\}$.

*Theorem 2: Innovative linear network codes are uniformly optimal.*

*Proof:* With an innovative linear network code, by definition, the packets received by a user who has not successfully decoded all the source packets must all be linearly independent. Therefore, the user is able to decode the source packets once he or she has experienced $N$ non-erasure slots. In other words, (4) holds for all users. Hence the code is uniformly optimal. □

In the next section, we will show that innovative linear network codes exist when $q \geq K$.

## VI. The Innovative Encoding Vector Problem

The existence of innovative linear network code is equivalent to the non-emptiness of the set of encoding vectors $\mathcal{I}$ as defined in (7). It was shown in [7] that $\mathcal{I}$ is non-empty if the finite field size, $q$, is larger than or equal to the number of users, $K$. We present a proof below for the sake of completeness. Afterwards, we show that when $q < K$, determining the existence of innovative vectors is NP-complete.

We begin with a simple lemma, which will be used again in a later section. The proof is trivial and thus omitted.

*Lemma 3: Let $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_K$ be finite subsets of a universal set $\mathcal{U}$. If $K \geq 2$ and $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_K$ contain a common element, then*

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

*Theorem 4 [7]: If $q \geq K$ and the rank of $\mathbf{C}_k$ is strictly less than $N$ for all $k$'s, then $\mathcal{I}$ is non-empty.*

*Proof:* The subspace $V_k$ consists of the $q^{r_k}$ encoding vectors that are *not* innovative to user $k$. Since the zero vector is a common vector of these $K$ subspaces, by Lemma 8, we have

$$|V_1 \cup V_2 \cup \cdots \cup V_K| < \sum_{k=1}^{K} q^{r_k} \leq K q^{N-1} \leq q^N.$$

Therefore, there is at least one innovative encoding vector. □

The condition $q \geq K$ in Theorem 4 cannot be improved in general. The following example shows that the non-existence of an innovative encoding vector for a case when $q = K - 1$.

*Example 1:* Let $q = 3$, $K = 4$, and $N = 3$. The encoding matrices are

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

It can be checked that there is no 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 are also known as the null spaces of $\mathbf{C}_k$'s. 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 $V_k^\perp$. 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 the columns of $\mathbf{C}_k$, we can write $\mathbf{C}_k$ in the following form:

$$[\mathbf{I}_{r_k} | \mathbf{A}_k]\mathbf{P}_k, \tag{8}$$

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.[1] We can take

$$\mathbf{B}_k = [-\mathbf{A}_k^T | \mathbf{I}_{N-r_k}]\mathbf{P}_k. \tag{9}$$

The superscript $^T$ represents the transpose operator. It is straightforward to verify that the product of the matrix in (8)

[1] Recall that a permutation matrix is a square zero-one matrix so that each column and each row contain exactly one "1".

and $\mathbf{B}_k^T$ is a zero matrix. Hence, the $n - r_k$ row vectors in $\mathbf{B}_k$ belong to $V_k^\perp$. Since $\mathbf{B}_k$ contains a permutation of $\mathbf{I}_{N-r_k}$ as a submatrix, the rows of $\mathbf{B}_k$ are linearly independent. As $\dim(V_k^\perp) = n - r_k$, we conclude that the rows of $\mathbf{B}_k$ form a basis of $V_k^\perp$.

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

*Lemma 5: 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, $\mathbf{x} \notin \mathcal{I}$. □

In Appendix A, we give another way of computing a basis of $V_k^\perp$, which is suitable for incremental processing.

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 for a given prime power $q$:

**Problem:** $\mathsf{IEV}_q$

**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 $k = 1, 2, \ldots, K$?

Note that in this formulation, $q$ is a fixed constant while $N$ and $K$ are arbitrary positive numbers, which are provided as part of the input. Without loss of generality, we can assume that the ranks of all the matrices, $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K$, in $\mathsf{IEV}_q$ are strictly less than $N$. Also, we know from Theorem 4 that the answer to $\mathsf{IEV}_q$ is always YES if $K \leq q$. We are interested in the case when $N$ and $K$ grow. The following result shows that the problem is NP-complete:

*Theorem 6: For any fixed prime power $q$, the problem $\mathsf{IEV}_q$ is NP-complete.*

*Proof:* The idea is to Karp-reduce the 3-SAT problem, well-known to be NP-complete [41], to the $\mathsf{IEV}_q$ 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 values 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 construct a Karp-reduction from the 3-SAT problem to the $\mathsf{IEV}_q$ problem with $N = n + 1$ packets and $K = m + 1 + n(q - 2)$ users.

For the $i$-th clause ($i = 1, 2, \ldots, m$), we first construct a $3 \times (n+1)$ matrix $\mathbf{B}_i$. The $j$-th literal ($j = 1, 2, 3$) in the $i$-th clause determines the entries in the $j$-th row of the 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 1, and all other components be 0. 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 components in the $j$-th row of $\mathbf{B}_i$ be 1 and $-1$, respectively, and all other components be 0.

For user $m + 1$, let $\mathbf{B}_{m+1}$ be the $1 \times (n + 1)$ matrix $[\mathbf{0}_n\ 1]$, where $\mathbf{0}_n$ is the zero row vector of length $n$.

Next we consider the remaining $n(q - 2)$ users. (For the special case when $q = 2$, these users do not exist.) For $u = 1, 2, \ldots, n$, let $\mathbf{E}_{u,1}, \mathbf{E}_{u,2}, \ldots, \mathbf{E}_{u,q-2}$ be $1 \times (n+1)$ matrices whose $u$-th components are distinct elements in $GF(q)\backslash\{0, 1\}$ and the $(n + 1)$-st components are all equal to $-1$. Let these $n(q - 2)$ matrices be $\mathbf{B}_i$'s for $i = m + 2, m + 3, \ldots, m + 1 + n(q - 2)$.

Let $\mathbf{C}_i$ be a matrix whose rows form a basis of the orthogonal complement of the row space of $\mathbf{B}_i$. The above procedure defines the encoding matrices for all the $m + 1 + n(q - 2)$ users, and the reduction can be done in polynomial time.

Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ be a Boolean vector and define $\hat{\mathbf{x}} \triangleq (\mathbf{x}, 1)$. 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 $j = 1, 2, \ldots, m + 1 + n(q - 2)$. By Lemma 5, $\hat{\mathbf{x}}$ is not in the row space of $\mathbf{C}_j$ for all $j$. Hence $\hat{\mathbf{x}}$ is also a solution to the derived $\mathsf{IEV}_q$ problem.

Conversely, any solution to the derived $\mathsf{IEV}_q$ problem also yields a solution to the original 3-SAT problem as well. To see this, let $\mathbf{c} = (c_1, c_2, \ldots, c_n, c_{n+1}) \in GF(q)^{n+1}$ be a solution to the derived $\mathsf{IEV}_q$ problem. Note that we must have $c_{n+1} \neq 0$ because of $\mathbf{B}_{m+1}$. Since a non-zero scalar multiplication of $\mathbf{c}$ remains to be a solution to the derived $\mathsf{IEV}_q$ problem, without loss of generality, we can assume that $c_{n+1} = 1$. Due to the last $n(q - 2)$ users, for $i = 1, 2, \ldots, n$, we must have $c_i = 0$ or 1, for otherwise $\mathbf{E}_{i,j}\mathbf{c}$ must be zero for some $j \in \{1, 2, \ldots, q - 2\}$. (More precisely, given any $i$, one and only one of these $q - 2$ vectors is zero.) Let $i$ be an index 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 $\mathsf{IEV}_q$ is clearly in NP, since it is efficiently verifiable. Hence it is NP-complete. □

Note that the above result means that there is no polynomial-time algorithm that can solve $\mathsf{IEV}_q$ when $N$ and $K$ grow, provided that $P \neq NP$. It does not apply to specific settings of $N$ and $K$. For example, if $K \leq q$, $\mathsf{IEV}_q$ becomes trivial to solve, as shown by Theorem 4.

## VII. 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. Recall that when $q \geq K$, innovative vectors always exist. But it is not clear whether $K$-sparse innovative vectors exist. In this section, we prove that when $q \geq K$, $K$-sparse innovative vectors always exist. On the other hand, determining whether there is an innovative vector with Hamming weight less than a certain number is shown to be NP-complete. For this reason, we show that to maximize the sparsity of an innovative vector, we can reduce the problem to the hitting set problem, which

facilitates the development of network coding algorithms in the next section.

### A. Existence of $K$-Sparse Innovative Vector

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

*Lemma 7: For $k = 1, 2, \ldots, K$, let $f_k(\mathbf{x})$ be a non-zero linear polynomial in $L$ variables*

$$f_k(\mathbf{x}) \triangleq \alpha_{k1}x_1 + \alpha_{k2}x_2 + \cdots + \alpha_{kL}x_L,$$

*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 $k$ and $\mathbf{x}^*$ is $K$-sparse.*

*Proof:* Let $\mathcal{S}_l$, where $l = 1, 2, \ldots, L$, 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 identically zero, the union $\bigcup_{l=1}^{L} \mathcal{S}_l$ is equal to $\{1, 2, \ldots, K\}$. 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 $n \neq l$. In this case, $\mathbf{x}^*$ is 1-sparse.

*Case 2:* $|\mathcal{S}_l| < K$ for all $l$. We initialize the encoding vector $\mathbf{x}^*$ to the all-zero vector and assign values to the $L$ variables iteratively, starting from $x_1$ and ending with $x_L$. When we assign a value to $x_t$, for $t = 1, 2, \ldots, L$, we want to choose a value of $x_t^*$ such that the condition

$$f_k(\mathbf{x}^*) \neq 0 \quad \text{for all } k \in \bigcup_{l=1}^{t} S_l \tag{10}$$

is maintained. As $\mathbf{x}^*$ is initialized to the all-zero vector, the condition in (10) is satisfied for $t = 0$ (an empty union is defined as the empty set by convention).

Suppose we have already assigned $x_1^*, x_2^*, \ldots, x_{t-1}^*$ to the first $t-1$ variables. Consider the assignment of $x_t$. If $\mathcal{S}_t \setminus \bigcup_{l=1}^{t-1} S_l$ is empty, then we keep $x_t^* = 0$ and go to the next iteration, i.e., the assignment of the next variable, $x_{t+1}$. Otherwise, if $\mathcal{S}_t \setminus \bigcup_{l=1}^{t-1} \mathcal{S}_l$ is non-empty, consider the equations

$$f_k(x_1^*, \ldots, x_{t-1}^*, x_t, 0, \ldots, 0) = 0, \tag{11}$$

for $k \in \mathcal{S}_t$. Since (11) is a linear equation in a single variable $x_t$, there is one and only one solution to (11) for each $k \in \mathcal{S}_t$. As $|\mathcal{S}_t| < K \leq q$, we can choose $x_t^*$ to be an element in $GF(q)$ such that $f_k(x_1^*, \ldots, x_t^*, 0, \ldots, 0)$ is nonzero for all $k \in \mathcal{S}_t$. After this assignment, the condition in (10) is satisfied.

Upon the termination of the algorithm, we have $f_k(\mathbf{x}^*) \neq 0$ for $k \in \bigcup_{l=1}^{L} \mathcal{S}_l = \{1, 2, \ldots, K\}$. Since there are $K$ inequations in total, $\mathcal{S}_t \setminus \bigcup_{l=1}^{t-1} \mathcal{S}_l$ is non-empty for at most $K$ indices $t$, and there are at most $K$ non-zero assignments to the components of $\mathbf{x}^*$. Hence, $\mathbf{x}^*$ is $K$-sparse. □

The above proof is constructive and algorithmic. The details of implementing the method in the proof of Lemma 7 are presented in Algorithm 1, and we call it the *Sequential Assignment algorithm*. The computational complexity of the Sequential Assignment algorithm in terms of number of multiplications/divisions over $GF(q)$ is shown below:

*Theorem 8: The time complexity of the Sequential Assignment algorithm is $O(KL)$.*

**Algorithm 1** Sequential Assignment Algorithm

**Input:** A $K \times L$ matrix $\mathbf{A} = [\alpha_{kl}]_{k=1,\ldots,K}^{l=1,\ldots,L}$ over $GF(q)$. It is assumed that $\mathbf{A}$ has no zero rows and $q \geq K$.

**Output:** A $K$-sparse vector $\mathbf{x}^*$ over $GF(q)$ such that all components of $\mathbf{A}\mathbf{x}^*$ are nonzero.

1: $x_l^* \leftarrow 0$ for $l = 1, 2, \ldots, L$. // initialize $\mathbf{x}^*$ to $\mathbf{0}$
2: $S_l \leftarrow \{i : \alpha_{il} \neq 0\}$ for $l = 1, 2, \ldots, L$.
3: **if** $\exists l \in \{1, 2, \ldots, L\}$ such that $|S_l| = K$ **then**
4: Let $l$ be an index in $\{1, 2, \ldots, L\}$ such that $|S_l| = K$.
5: $x_l^* \leftarrow 1$.
6: **else**
7: $t \leftarrow 1$.
8: $w_k \leftarrow 0$ for $k = 1, 2, \ldots, K$.
9: $S \leftarrow \emptyset$.
10: **while** $S \neq \{1, 2, \ldots, K\}$ **do**
11: **if** $S_t \not\subseteq S$ **then**
12: $y \leftarrow$ an element in $GF(q)$ such that $y \neq -w_k/\alpha_{k,t}$ for all $k \in S_t$.
13: $w_k \leftarrow w_k + \alpha_{k,t}y$ for $k \in S_t$.
14: $x_t^* \leftarrow y$.
15: $S \leftarrow S \cup S_t$.
16: **end if**
17: $t \leftarrow t + 1$.
18: **end while**
19: **end if**
20: **return** $(x_1^*, x_2^*, \ldots, x_L^*)$.

*Proof:* The main part of Algorithm 1 is the while-loop between Line 10 and Line 18. The variable $w_k$ stores the value of $\sum_{l=1}^{L} \alpha_{kl}x_l^*$, for $k = 1, 2, \ldots, K$, and the variable $S$ stores the value of $\bigcup_{l=1}^{t-1} \mathcal{S}_l$. Since it is assumed that $\bigcup_{l=1}^{L} \mathcal{S}_l = \{1, 2, \ldots, K\}$, the while-loop is repeated at most $L$ times, and we execute Line 12 to Line 15 at most $K$ times. In Line 12, we need to find an element $y$ in $GF(q)$ such that $y$ is not equal to $-w_k/\alpha_{k,t}$ for all $k \in S_t$. This requires no more than $K$ division operations. Therefore, the total complexity of the Sequential Assignment algorithm is $O(KL)$. □

*Theorem 9: If $q \geq K$, there is 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 contains 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 $\mathbf{x}(\mathcal{N})$. By Lemma 5, $\mathbf{x}$ is in $\mathcal{I}$ if $\mathbf{b}_k(\mathcal{N}) \cdot \mathbf{x}(\mathcal{N}) \neq 0$ for all $k$'s. By Lemma 7, if $q \geq K$, we can find a vector $\mathbf{y} \in GF(q)^{|\mathcal{N}|}$ such that $\mathbf{b}_k(\mathcal{N}) \cdot \mathbf{y} \neq 0$ for all $k$'s. Let $\mathbf{x}^* \in GF(q)^N$ be the vector such that $\mathbf{x}^*(\mathcal{N}) = \mathbf{y}$ and other components of $\mathbf{x}^*$ are all zero. Clearly, $\mathbf{x}^*$ belongs to $\mathcal{I}$. It is $K$-sparse, since $|\mathcal{N}| \leq K$. □

The above result shows that if $q \geq K$, the minimum Hamming weight of innovative vectors is bounded above by $K$. This upper bound cannot be further reduced as the following example shows:

*Example 2:* 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 complements 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

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

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 9 shows that we can 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 the 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 MAX 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 $n$?

We have already proven that the answer is always YES if $n \geq K$. We are interested in the case where $n < K$.

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

$$\tilde{\mathbf{b}}_k \triangleq \vee_{i=1}^{N-r_k} \mathbf{b}_{k,i}, \tag{12}$$

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 10: 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 $\tilde{\mathbf{b}}_k(\mathcal{N})$ is not equal to the zero vector 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 7, we can find $\mathbf{y} \in GF(q)^{|\mathcal{N}|}$ such that $\mathbf{b}_{k,j}(\mathcal{N}) \cdot \mathbf{y} \neq 0$ for all $k$'s. Let $\mathbf{x} \in GF(q)^N$ be the vector such that $\mathbf{x}(\mathcal{N}) = \mathbf{y}$ and other components of $\mathbf{x}$ are all zero. Then by Lemma 5, $\mathbf{x} \in \mathcal{I}$.

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 Lemma 5 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 [41].

*Theorem 11:* SPARSITY *is NP-complete.*

*Proof:* We are going to reduce HITTINGSET to an instance of SPARSITY via a Karp-reduction. 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 $\mathscr{C} = \{\text{supp}(\mathbf{b}_1), \text{supp}(\mathbf{b}_2), \ldots, \text{supp}(\mathbf{b}_K)\}$. These $\mathbf{b}_k$'s correspond to the degenerate form of $\mathbf{B}_k$'s in Lemma 5 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 (7). 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 $|\text{supp}(\mathbf{x})| \leq n$. Given the $\mathbf{b}_k$'s obtained via the above reduction, suppose there exists $\mathbf{x} \in \mathcal{I}$ with $|\text{supp}(\mathbf{x})| \leq n$. By Lemma 5, we must have $\mathbf{b}_k \cdot \mathbf{x} \neq 0$ for all $k$'s, which implies $\text{supp}(\mathbf{b}_k) \cap \text{supp}(\mathbf{x}) \neq \emptyset$ 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 $\text{supp}(\mathbf{b}_k) \cap \mathcal{H} \neq \emptyset$ for all $k$'s. Therefore, $\mathbf{B}(\mathcal{H})$ has no zero rows. By Lemma 10, there exists an $\mathbf{x} \in GF(q)^N$ such that supp$(\mathbf{x}) \subseteq \mathcal{H}$. Hence, $|\text{supp}(\mathbf{x})| \leq n$.

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:** MAX SPARSITY

**Instance:** $K$ 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.

We call the minimum Hamming weight among all innovative vectors the *sparsity number*, and denote it by $\omega$. 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, MAX SPARSITY is NP-hard.

On the other hand, if $K$ is held fixed, meaning that the problem size grows only with $N$, then there exists algorithm whose complexity grows polynomially in $N$ to solve MAX SPARSITY. It is proven in [42] and Section VII-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 Lemma 5 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, MAX 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 MAX SPARSITY can be solved via MIN HITTINGSET based on the concept of Levin-reduction.

*Theorem 12:* MAX SPARSITY *can be Levin-reduced to* MIN HITTINGSET.

*Proof:* Given an instance of MAX SPARSITY, we determine $\tilde{\mathbf{b}}_k$ as in (12) for $k = 1, 2, \ldots, K$. Then we form the following instance of MIN HITTINGSET:

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

Let $\mathcal{H}$ be a solution to the above instance. Then $\mathbf{B}(\mathcal{H})$ has no zero rows. By Lemma 10, there exists a vector $\mathbf{x}^* \in \mathcal{I}$ over $GF(q)$ with $\text{supp}(\mathbf{x}^*) \subseteq \mathcal{H}$. Such a vector $\mathbf{x}^*$ can be found by the Sequential Assignment algorithm in polynomial time.

We claim that there does not exist $\mathbf{x}' \in \mathcal{I}$ with Hamming weight $|\text{supp}(\mathbf{x}')| < |\mathcal{H}|$, and thus $|\text{supp}(\mathbf{x}^*)|$ must equal $|\mathcal{H}|$. Suppose there exists such a vector $\mathbf{x}'$. Lemma 10 implies that $\mathbf{B}(\text{supp}(\mathbf{x}'))$ has no zero rows, which in turn implies that $\text{supp}(\mathbf{x}') \cap \text{supp}(\tilde{\mathbf{b}}_k) \neq \emptyset$ for all $k$'s. Then $\text{supp}(\mathbf{x}')$ would be a hitting set with cardinality strictly less than $|\mathcal{H}|$. A contradiction.

The proof is completed by matching the relevant entities and procedures with those in Definition 4. Note that the transformation of a given instance of MAX SPARSITY to an instance of MIN HITTINGSET in essence corresponds to the mapping $f$. A solution to an instance of MIN HITTINGSET, $\mathcal{H}$, corresponds to $y'$. Obtaining $\boldsymbol{x}^*$ from $\mathcal{H}$ by the Sequential Assignment algorithm corresponds to the mapping $g$. □

The above result allows one to solve MAX SPARSITY by means of solving MIN HITTINGSET. The algorithms developed in the next section are based on this idea. Finally, we remark that our approach tries to find a sparse encoding vector for the next transmitted packet. If the broadcast problem is considered as a whole, one should minimize the average Hamming weight of all the transmitted encoded packets. As finding a sparsest vector is already NP-hard, finding a set of sparse vectors for all the transmitted packets can only be harder. Therefore, in this paper, we adopt a greedy approach and only consider finding a sparse vector for the next transmission.

## VIII. NETWORK CODING ALGORITHMS

In this section, we present two algorithms that generate sparse innovative encoding vectors for $q \geq K$. The first one is optimal in terms of sparsity while the second one is an approximate algorithm.

### A. The Optimal Hitting Method

For $q \geq K$, we obtain a sparsest 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 MIN HITTINGSET. We remark that MIN HITTINGSET can be formulated as a binary integer programming (BIP) problem, which can be solved, for example, by the cutting plane method [43]. Alternatively, MIN HITTINGSET can also be solved by the method in [44]. Once $\mathcal{N}$ is found, the non-zero entries in the vector can be obtained by the Sequential Assignment algorithm. We call this procedure for generating an innovative vector with minimum Hamming weight the *Optimal Hitting method*. We summarize the algorithm in Algorithm 2. The correctness of the Optimal Hitting method is guaranteed by Theorem 12.

### B. The Greedy Hitting Method

Step 4 in the Optimal Hitting 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 approximately by the following greedy approach [45]:

- 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 4 of the Optimal Hitting method, the above greedy algorithm can be used to find approximate solutions. We call this modification the *Greedy Hitting* method.

*Theorem 13: The Hamming weight of the encoding vector obtained by the Greedy Hitting method is bounded above by* $\min\{H_N\omega, K\}$*, where* $H_N$ *is the* $N$*-th harmonic number, defined as* $H_N \triangleq \sum_{k=1}^{N} \frac{1}{k}$*, and* $\omega$ *is the sparsity number.*

*Proof:* It is well known that the hitting set problem is just a reformulation of the set covering problem.

**Algorithm 2** The Optimal Hitting Method

**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.

1: Initialize $\mathbf{x}$ as the zero vector.
2: 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.
3: 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 $\mathbf{B}_k$ is regarded as "1" when taking the logical-OR operation.)
4: Solve the corresponding MIN HITTINGSET as shown in Theorem 12 by BIP and obtain $\mathcal{H} \subseteq \{1, 2, \ldots, N\}$.
5: For $k = 1, 2, \ldots, K$, choose a row vector from $\mathbf{B}_k$, say $\hat{\mathbf{b}}_k^T$, such that $\text{supp}(\hat{\mathbf{b}}_k) \cap \mathcal{H} \neq \emptyset$.
6: Determine $\mathbf{y} \in GF(q)^{|\mathcal{H}|}$ such that $\mathbf{y} \cdot \hat{\mathbf{b}}_k(\mathcal{H}) \neq \mathbf{0}$ for $k = 1, 2, \ldots, K$, by the Sequential Assignment algorithm.
7: $\mathbf{x}(\mathcal{H}) \leftarrow \mathbf{y}$.

Therefore, the greedy algorithm is an $H_{|\mathcal{U}|}$ factor approximation algorithm for MIN HITTINGSET, as well as for the set covering problem [46]. As shown in Theorem 12, MAX SPARSITY can be reduced to MIN HITTINGSET, and the sparsity number is equal to the cardinality of the minimum hitting set. Hence, the Greedy Hitting method is also an $H_N$ factor approximation algorithm for MAX SPARSITY.

Since Step 6 in the Greedy Hitting method involves the Sequential Assignment algorithm. By Lemma 7, the encoding vector thus obtained is $K$-sparse. Hence, the encoding vector obtained by the Greedy Hitting method is $K$-sparse. Combining with the upper bound in the previous paragraph, the statement is proved. □

### C. Computational Complexity

Recall that the encoding matrix of each receiver is composed of the encoding vectors of the broadcast packets. If the broadcast packets are encoded by either Optimal Hitting or Greedy Hitting, the encoding vectors are all $K$-sparse. For this reason, we are interested in the case where the rows of the encoding matrices are all $K$-sparse.

*Theorem 14: Suppose the rows of* $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K$ *are all* $K$*-sparse. The time complexities of Optimal Hitting and Greedy Hitting methods are* $O(1.23801^{(N+K)})$ *and* $O(K^2N^2)$*, respectively.*

*Proof:* For the Optimal Hitting 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 $K$-sparse, we can adopt the dual-basis approach in obtaining $\mathbf{B}_k$ as in Appendix A, and guarantee that each $\mathbf{B}_k$ can be obtained in $O(KN^2)$ times. The computational complexity of Step 2 is thus $O(K^2N^2)$. Step 3 involves $O(KN^2)$ operations. In step 4, the MIN HITTINGSET problem shown in Theorem 12 has a complexity of $O(1.23801^{(N+K)})$ [44]. Step 5 requires $O(K)$ operations. Step 6 involves the Sequential Assignment algorithm, which has a complexity of $O(K|\mathcal{H}|)$ according to Theorem 8. Step 7 requires $O(N)$ operations.

Since $|\mathcal{H}| \leq N$, the overall complexity of the Optimal Hitting method is $O(K^2N^2 + 1.23801^{(N+K)}) = O(1.23801^{(N+K)})$. The only difference between the Optimal Hitting and Greedy Hitting method is that the Greedy Hitting method uses a greedy algorithm to approximate the MIN HITTINGSET problem in Step 4. The greedy algorithm takes $O(KN^2)$ operations. Therefore, the overall complexity of the Greedy Hitting method is $O(K^2N^2)$. □

For both Optimal Hitting and Greedy Hitting, all the encoding vectors obtained are $K$-sparse. When these packets are broadcast, a receiver can decode and obtain the source packets by solving a sparse linear system, whose time complexity is $O(\min\{K, N\}N^2)$ [22].

## IX. CONCLUSION

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. We show that innovative linear network code is uniformly optimal in minimizing downloading delay. 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 smaller fields is NP-complete.

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 at most $K$. To find a sparsest innovative vector is proven to be NP-hard via a reduction from the hitting set problem. An exact algorithm based on binary integer programming is described, and a polynomial-time approximation algorithm based on the greedy approach is constructed. We hope that our work increases the understanding of complexity issues in network coding.

## APPENDIX A
## INCREMENTAL METHOD FOR COMPUTING A BASIS OF THE NULL SPACE OF A GIVEN MATRIX

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 $GF(q)$, 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 an $N \times N$ matrix by appending $N - r$ row vectors. These vectors are chosen in a way such that the resulting matrix, denoted by $\tilde{\mathbf{C}}$, is non-singular. 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 Lemma 5, it is innovative to that user if and only if one or more of such inner products are non-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 $\tilde{\mathbf{C}}^{-1} = \tilde{\mathbf{B}}$.

We are going to modify $\tilde{\mathbf{C}}$ by updating its $(r+1)$-st row to $\mathbf{w}^T$. This operation can be expressed algebraically by

$$\tilde{\mathbf{C}} \longleftarrow \tilde{\mathbf{C}} + \mathbf{e}_{r+1}(\mathbf{w} - \mathbf{c}_{r+1})^T, \tag{13}$$

where $\mathbf{e}_{r+1}$ is the column vector with the $(r+1)$-st component equal to 1 and 0 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 0 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 [47], [48, p. 18],

$$\begin{aligned}
&(\tilde{\mathbf{C}} + \mathbf{e}_{r+1}(\mathbf{w} - \mathbf{c}_{r+1})^T)^{-1} \\
&\quad = \tilde{\mathbf{C}}^{-1} - \frac{\tilde{\mathbf{C}}^{-1}\mathbf{e}_{r+1}(\mathbf{w} - \mathbf{c}_{r+1})^T\tilde{\mathbf{C}}^{-1}}{1 + (\mathbf{w} - \mathbf{c}_{r+1})^T\tilde{\mathbf{C}}^{-1}\mathbf{e}_{r+1}} \\
&\quad = \tilde{\mathbf{C}}^{-1} - \frac{\mathbf{b}_{r+1}(\mathbf{w} - \mathbf{c}_{r+1})^T\tilde{\mathbf{C}}^{-1}}{\mathbf{w}^T\mathbf{b}_{r+1}} \\
&\quad = \tilde{\mathbf{C}}^{-1} - \frac{\mathbf{b}_{r+1}(\mathbf{w}^T\tilde{\mathbf{C}}^{-1} - \mathbf{e}_{r+1}^T)}{\mathbf{w}^T\mathbf{b}_{r+1}}.
\end{aligned} \tag{14}$$

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 (14) is a non-zero scalar by construction, so that division of zero would not occur.

The updating procedure can now be performed. $\tilde{\mathbf{C}}$ is updated according to (13) and $\tilde{\mathbf{B}}$ is updated as follows:

$$\tilde{\mathbf{B}} \longleftarrow \tilde{\mathbf{B}} - \frac{\mathbf{b}_{r+1}(\mathbf{w}^T\tilde{\mathbf{B}} - \mathbf{e}_{r+1}^T)}{\mathbf{w}^T\mathbf{b}_{r+1}}. \tag{15}$$

Note that if $\mathbf{w}$ is $\omega$-sparse, the multiplication of $\mathbf{w}^T$ and $\tilde{\mathbf{C}}^{-1}$ in (14) can be done in $O(\omega N)$ operations.

## ACKNOWLEDGEMENTS

The authors would like to thank Prof. Wai Ho Mow and Dr. Kin-Kwong Leung for their stimulating discussions in the early stage of this work. Besides, they are grateful to the associate editor and the anonymous reviewers for their careful reading and valuable suggestions.

## REFERENCES

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

[2] R. Kötter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, 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 *Proc. 40th Annu. Conf. Inf. Sci. Syst.*, Princeton, NJ, USA, Mar. 2006, pp. 864–870.

[4] M. Ghaderi, D. Towsley, and J. Kurose, "Reliability gain of network coding in lossy wireless networks," in *Proc. 27th IEEE Int. Conf. Comp. Commun. (INFOCOMM)*, Phoenix, AZ, USA, 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 *Proc. IEEE Int. Conf. Commun. Workshops*, Dresden, Germany, Jun. 2009, pp. 1–6.

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

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

[8] M. Luby, "LT codes," in *Proc. 43rd Annu. IEEE Symp. Found. Comput. Sci.*, Nov. 2002, pp. 271–282.

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

[10] T. Ho *et al.*, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[11] S. Jaggi *et al.*, "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 *Proc. Int. Symp. Inf. Theory Appl. (ISITA)*, Auckland, New Zealand, Dec. 2008, pp. 1–6.

[13] J. K. Sundararajan, D. Shah, and M. Médard, "A feedback-based adaptive broadcast coding scheme for reducing in-order delivery delay," in *Proc. Netw. Coding, Theory, Appl. (NetCod)*, Lausanne, Switzerland, Jun. 2009, pp. 1–6.

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

[15] L. Lu, M. Xiao, M. Skoglund, L. Rasmussen, G. Wu, and S. Li, "Efficient network coding for wireless broadcasting," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Sydney, NSW, Australia, Apr. 2010, pp. 1–6.

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

[17] L. Keller, E. Drinea, and C. Fragouli, "Online broadcasting with network coding," in *Proc. 4th Workshop Netw. Coding, Theory, Appl. (NetCod)*, Hong Kong, Jan. 2008, pp. 68–73.

[18] R. A. Costa, D. Munaretto, J. Widmer, and J. Barros, "Informed network coding for minimum decoding delay," in *Proc. 5th IEEE Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS)*, Atlanta, GA, USA, Sep./Oct. 2008, pp. 80–91.

[19] P. Sadeghi, D. Traskov, and R. Kötter, "Adaptive network coding for broadcast channels," in *Proc. 5th Workshop Netw. Coding, Theory, Appl. (NetCod)*, Jun. 2009, pp. 80–85.

[20] P. Sadeghi, R. Shams, and D. Traskov, "An optimal adaptive network coding scheme for minimizing decoding delay in broadcast erasure channels," *EURASIP J. Wireless Commun. Netw.*, vol. 2010, Jan. 2010, Art. no. 618016.

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

[22] C. W. Sung, K. W. Shum, and H. Y. Kwan, "On the sparsity of a linear network code for broadcast systems with feedback," in *Proc. IEEE Int. Symp. Netw. Coding*, Beijing, China, Jul. 2011, pp. 1–4.

[23] E. Kaltofen and B. D. Saunders, "On Wiedemann's method of solving sparse linear systems," in *Proc. 9th Int. Symp. Appl. Algebra, Algebraic Algorithms Error-Correcting Codes (AAECC)*, 1991, pp. 29–38.

[24] D. Coppersmith, "Solving linear equations over GF(2): Block Lanczos algorithm," *Linear Algebra Appl.*, vol. 192, pp. 33–60, Oct. 1993.

[25] A. Barg, "Complexity issues in coding theory," *Handbook of Coding Theory*, vol. 1. Amsterdam, The Netherlands: Elsevier, 1998, pp. 649–754.

[26] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg, "On the inherent intractability of certain coding problems (Corresp.)," *IEEE Trans. Inf. Theory*, vol. 24, no. 3, pp. 384–386, May 1978.

[27] 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.

[28] A. R. Lehman and E. Lehman, "Complexity classification of network information flow problems," in *Proc. 15th Annu. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 2004, pp. 142–150.
[29] H. Yao and E. Verbin, "Network coding is highly non-approximable," in *Proc. 47th Annu. Allerton Conf.*, Monticello, IL, USA, Sep./Oct. 2009, pp. 209–213.
[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] M. Langberg, A. Sprintson, and J. Bruck, "Network coding: A computational perspective," *IEEE Trans. Inf. Theory*, vol. 55, no. 1, pp. 147–157, Jan. 2009.
[33] N. J. A. Harvey, D. R. Karger, and K. Murota, "Deterministic network coding by matrix completion," in *Proc. 16th Annu. ACM-SIAM Symp. Discrete Algorithm (SODA)*, Jan. 2005, pp. 489–498.
[34] N. J. A. Harvey, D. R. Karger, and S. Yekhanin, "The complexity of matrix completion," in *Proc. 17th Annu. ACM-SIAM Symp. Discrete Algorithm (SODA)*, Jan. 2006, pp. 1103–1111.
[35] Y. Birk and T. Kol, "Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2825–2830, Jun. 2006.
[36] R. Peeters, "Orthogonal representations over finite fields and the chromatic number of graphs," *Combinatorica*, vol. 16, no. 3, pp. 417–431, 1994.
[37] A. S. Tehrani and A. G. Dimakis, "Finding three transmissions is hard," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Anaheim, CA, USA, Dec. 2012, pp. 2293–2298.
[38] 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, FL, USA, Aug. 2011, pp. 306–310.
[39] S. Y. El Rouayheb, M. A. R. Chaudhry, and A. Sprintson, "On the minimum number of transmissions in single-hop wireless coding networks," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Lake Tahoe, CA, USA, Sep. 2007, pp. 120–125.
[40] O. Goldreich, *P, NP, and NP-Completeness: The Basics of Computational Complexity*. Cambridge, U.K.: Cambridge Univ. Press, 2010.
[41] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA, USA: Freeman, 1979.
[42] H. Y. Kwan, K. W. Shum, and C. W. Sung, "Generation of innovative and sparse encoding vectors for broadcast systems with feedback," in *Proc. IEEE Int. Symp. Inf. Theory*, St. Petersburg, FL, USA, Jul./Aug. 2011, pp. 1161–1165.
[43] A. Schrijver, *Theory of Linear and Integer Programming*. New York, NY, USA: Wiley, 1986.
[44] L. Shi and X. Cai, "An exact fast algorithm for minimum hitting set," in *Proc. 3rd Int. Joint Conf. Comput. Sci. Optim.*, Mar. 2010, pp. 64–67.
[45] U. Feige, "A threshold of ln $n$ for approximating set cover," *J. ACM*, vol. 45, no. 4, pp. 634–652, 1998.
[46] V. V. Vazirani, *Approximation Algorithms*. Berlin, Germany: Springer, 2003.
[47] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes*, 3rd ed. Cambridge, U.K.: Cambridge Univ. Press, 2007, ch. 2.7.1.
[48] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1985.

**Chi Wan Sung** (M'98) received his B.Eng, M.Phil, and Ph.D. degrees in information engineering from the Chinese University of Hong Kong in 1993, 1995, and 1998, respectively. He joined the faculty at City University of Hong Kong in 2000, and is now Associate Professor with the Department of Electronic Engineering. He is an Adjunct Associate Research Professor at University of South Australia, and is on the editorial boards of the *ETRI Journal* and the *Transactions on Emerging Telecommunications Technologies* (ETT). His research interests include wireless communications, resource allocation, network coding, and cloud storage systems.

**Kenneth W. Shum** (M'00) received his B.Eng degree in information engineering from The Chinese University of Hong Kong in 1993, and MSc and PhD degrees in electrical engineering from University of Southern California in 1995 and 2000, respectively. He is now a Research Associate Professor with Institute of Network Coding in The Chinese University of Hong Kong. His research interests include network coding and coding for distributed storage systems.

**Linyu Huang** (M'15) received the B.Eng. degree in electronic information engineering from the University of Electronic Science and Technology of China in 2008, and the Ph.D. degree in electronic engineering from the City University of Hong Kong in 2014, respectively. He joined the College of Electronics and Information Engineering, Sichuan University, as a Lecturer in 2014. His research interests include wireless communications, resource allocation and network coding.

**Ho Yuet Kwan** received the Ph.D. degree in information engineering from The Chinese University of Hong Kong, Shatin, Hong Kong, in 2004. His research interests include network coding and power control in wireless systems.