Network Error Correction From Matrix Network Coding

Kwang Taik Kim
Communication and Networking Group,
Samsung Advanced Institute of Technology
Yongin, Republic of Korea
Email: kwangtaik.kim@samsung.com

Chan-Soo Hwang
ASSIA, Inc.
Redwood City, CA 94065
Email: chwang@assia-inc.com

Vahid Tarokh
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
Email: vahid@seas.harvard.edu

***Abstract*—We present matrix network coding methods that are naturally amenable to a distributed implementation method, i.e., do not require the knowledge of network topology, and that are suitable for network error correction. First, the Singleton bound can be K-fold increased by employing a $K \times K$ matrix coefficient. Moreover, we prove that matrix network coding outperforms linear network coding, since it corrects more errors than linear network coding, while the amount of header overhead per packet can be kept the same by reducing the finite field size. This comes from the fact that the finite field size of matrix network coding required to guarantee the sufficient decoding probability is much smaller than linear network coding. Secondly, matrix network coding is refinable in the sense that, by receiving a larger number of network coded packets, larger error correction capabilities are achieved. Simulation results show that matrix network coding can provide $0.7 - 2$[dB] more coding gain than the linear network coding schemes.**

I. Introduction

Recently, wireless ad-hoc networks have received much attention, because of their applications to sensor networks and wireless local-area network (LAN) meshes. Such ad-hoc networks can improve the coverage of LANs and enable networking among wireless sensors and handheld devices. For such consumer applications, a wireless ad-hoc network needs to accommodate multiple data-flows that have different source/destination pairs. Each data-flow takes multiple hops over the air from a source node to a destination node, thus requiring intermediate nodes for relaying. Usually, such relaying protocol has been implemented using ad-hoc routing protocols, for example, ad-hoc on demand vector routing (AODV) in WLAN mesh using IEEE 802.11s standards [14]. Unfortunately, such ad-hoc routing schemes may not provide enough performance and robustness to support consumer-grade handheld devices. This is because multiple data-flows would intersect at a certain relay node, the ad-hoc routing protocols may suffer from the bottleneck effect at the relay node. In addition, since handheld devices would frequently move in and out of the network, ad-hoc routing protocols may not support sufficient robustness to such topology changes and may increase error rates. One of the candidates for improving ad-hoc routing protocols is network coding [4].

The concept of network coding was first invented by Ahlswede *et al.* who proved that it is suboptimal to restrict the network nodes to perform only routing [1]. Li *et al.* [8], and Koetter and Médard [7] showed that the multicast capacity can be achieved by only allowing linear operations in the intermediate nodes. Later, Ho *et al.* [6] showed that the linear network coding coefficients can be randomly selected, while the decoding probability approaches to 1 exponentially as the field size increases. On the other hand, a simple XOR network coding scheme was used in an IEEE 802.11 mesh network in [10]. It has been shown that this network coding significantly improves the throughput of both multicast and unicast traffic. In this paper, we explore the other benefit of network coding: the robustness to channel errors.

In network coding, intermediate nodes between various source/destination pairs typically transmit a linear combination (computed over finite field $\mathbb{F}_{2^m}$) of their various received data streams. If N packets $P_1, P_2 \cdots, P_N$ defined over finite field $\mathbb{F}_{2^m}$ are network coded, then N linear combinations of these packets are usually sufficient to decode $P_1, \cdots, P_N$ at the receiver. When a destination node receives more than N linear combinations of $P_1, \cdots, P_N$, this additional information can be used for error correction to improve the robustness to channel errors. Yeung and Cai [12] considered this issue and then further explored in [13]. These papers establish upper and lower bounds on the error correction capability of network codes analogous to the singleton and Gilbert-Varshamov bounds. In their work it is assumed that the network topology is known, nonetheless they do not provide concrete constructions that can be used in practice (over arbitrary network topologies). This motivates our approach in this paper, where we introduce matrix network coding schemes to enable network error corrections. We will remove the restriction on the knowledge of network topology in designing our network coding methods by using random matrix coding coefficients, similarly as random linear network coding in [6].

The outline of this paper is the following: In Section II, we review random linear network coding and discuss its limitations in network error correction. We then present the mathematical model of our system and introduce matrix network coding in Section III. In Section IV, we explain the encoding and decoding algorithms for the matrix network coding. We then compare the random linear network coding scheme and the proposed matrix network coding scheme in Section V. This comparison shows that matrix network coding

can correct more errors than random linear network coding in the presence of redundant received packets. We also show that the amount of header overhead per packet is kept the same, because the arithmetic required for matrix network coding can be performed over a smaller finite field. Moreover, when there are no redundant received packets and no transmission errors, matrix network coding and random linear network coding perform the same in terms of the decoding probability. This suggests that the proposed network error correction methods are robust to channel errors and refinable in the sense that the reception of a larger number of network coded packets improves the error correction performance. Finally in Section VI, we provide simulation results, which shows that the proposed matrix network coding scheme provides $0.7-2$[dB] coding gain over the random linear network coding schemes.

II. Problem Background

We consider a network with source nodes $S_1, S_2, \cdots, S_N$ and destinations $T_1, T_2, \cdots, T_M$. These source and destination nodes can change from time to time. Other nodes in the network may act as relay nodes. The main scenario of interest here is a multicast scenario where the packets $P_1, P_2, \cdots, P_N$, respectively, generated at nodes $S_1, S_2, \cdots, S_N$ must be received at all underlying destination nodes. However, our methods can also be applied to unicast scenarios (where $M = N$ and P_i is to be received at T_i for $i = 1, 2, \cdots, N$) or even mixed scenarios where some of the underlying packets must be sent in multicast and others in unicast mode. The channels between nodes can be either modeled as discrete, Gaussian or fading channels. Although our proposed construction method will apply to all these scenarios, in wireless scenarios Rayleigh or Rician fading channels may be of higher interest.

Existing network coding literature focuses on transmission of linear combinations of packets arriving at intermediate nodes. In other words if we assume without loss of generality that the input packets to a node are $P_1, P_2, \cdots, P_k$. These packets can be thought of as vectors of length L each defined over the finite field $\mathbb{F}_{2^m}$ (Equivalently, this means that each packet is of length Lm bits). Then the transmitted packet by this node is given by $\sum_{i=1}^{k} \alpha_i P_i$, where α_i's are elements of the finite field $\mathbb{F}_{2^m}$. Note that the same approach can be applied to perform network coding in analog domain.

A. Decoding/Error Correction in Random Linear Network Coding

Clearly in random linear network coding any packet received at the destination node is a linear combination of source packets $P_1, P_2, \cdots, P_N$. Assuming no errors, in order to reconstruct $P_1, P_2, \cdots, P_N$ at least N independent linear combinations of these packets are needed. Specifically, let us assume the reception of N linear combinations of packets $P_1, P_2, \cdots, P_N$ given by

$$P_o^l = \sum_{i=1}^{N} \alpha_i^l P_i \tag{1}$$

for $l = 1, 2, \cdots, N$, where $\alpha_i^l \in \mathbb{F}_{2^m}$ for $i, l = 1, 2, \cdots, N$. Let the j-th element of packets P_i and P_o^l be denoted by $P_i(j)$ and $P_o^l(j)$, respectively. Then $P_i(j),\ P_o^l(j) \in \mathbb{F}_{2^m}$ for $j = 1, 2, \cdots, L$. It follows from the above equation that

$$P_o^l(j) = \sum_{i=1}^{N} \alpha_i^l P_i(j). \tag{2}$$

Let $\tilde{P}_o(j) = \left(P_o^1(j), \cdots, P_o^N(j)\right)$, $\tilde{P}(j) = (P_1(j), \cdots, P_N(j))$ and

$$\Lambda = \begin{pmatrix} \alpha_1^1 & \alpha_1^2 & \cdots & \cdots & \alpha_1^N \\ \alpha_2^1 & \alpha_2^2 & \cdots & \cdots & \alpha_2^N \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \alpha_N^1 & \alpha_N^2 & \cdots & \cdots & \alpha_N^N \end{pmatrix}. \tag{3}$$

Then

$$\tilde{P}_o(j) = \tilde{P}(j)\Lambda.$$

This is a system of linear equations in $\mathbb{F}_{2^m}$. Assuming that the matrix Λ is invertible, the above equation can be solved to give

$$\tilde{P}(j) = \tilde{P}_o(j)\Lambda^{-1},$$

for $j = 1, 2, \cdots, L$.

Clearly in the above, if one of the symbols of $\tilde{P}_o(j)$ is received in error, then this error will effect some of the values of $\tilde{P}(j)$. One way that this may be corrected is by requiring that each packet $P_i,\ i = 1, 2, \cdots, N$ by itself is a coded packet encoded using a codebook $\mathcal{C}_i$. Then, if some of the values $P_i(j), j = 1, 2, \cdots, L$ are computed incorrectly, then these errors may be corrected by decoding using $\mathcal{C}_i$.

Another possibility is when K linear combinations of $P_1, P_2, \cdots, P_N$ are present at the receiver. Let

$$P_o^l = \sum_{i=1}^{N} \alpha_i^l P_i$$

for $l = 1, 2, \cdots, K$ be present at the receiver. Let

$$G = \begin{pmatrix} \alpha_1^1 & \alpha_1^2 & \cdots & \cdots & \alpha_1^K \\ \alpha_2^1 & \alpha_2^2 & \cdots & \cdots & \alpha_2^K \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \alpha_N^1 & \alpha_N^2 & \cdots & \cdots & \alpha_N^K \end{pmatrix},$$

(and as in the above) $\tilde{P}_o(j) = \left(P_o^1(j), \cdots, P_o^K(j)\right)$ and $\tilde{P}(j) = (P_1(j), \cdots, P_N(j))$. Then we have

$$\tilde{P}_o(j) = \tilde{P}(j)G.$$

This is equivalent to the case when $\tilde{P}(j)$ is encoded using a linear code defined over $\mathbb{F}_{2^m}$ whose generator matrix is G. When $K > N$, in theory one must be able to use the above linear code for error correction even if $P_1, P_2, \cdots, P_N$ are uncoded packets. Suppose next that $P_1, P_2, \cdots, P_N$ are coded packets using codebooks $\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_N$, respectively. Then the information bits of $P_1, P_2, \cdots, P_N$ can be presented as encoded by a two-dimensional product code whose vertical codebooks are $\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_N$, and in the horizontal direction

each row is encoded by the linear code whose generator matrix is G.

B. Main Challenges With Random Linear Network Coding

There are a number of issues in the above approach to network correction:

- First, let us consider the case that packets $P_1, P_2, \cdots, P_N$ are uncoded. Suppose, for example, that $K = N+1$. Then the minimum distance d of the linear code generated by G (from the MDS bound) satisfies $d \leq K - N + 1 \leq 2$. Since a linear code can only guarantee correction of all the error patterns of up to $t \leq \frac{d-1}{2}$ errors, then in this case, G does not offer any error correction capability at all.
- Even if $K \geq N+2$ in the above, again the code generated by G must have a good minimum d if any reasonable error correction capability is to be achieved, since only up to $t \leq \frac{d-1}{2}$ can be corrected.
- In most practical cases, the network topology is not known. This is particularly the case in mobile wireless systems. In such cases, the network designer has no control over the structure of G at the receiver. The packets received at the destination node are linear combinations of source packets with the coefficients depending on the path that the packet has traveled between the source node and destination node, and on the linear coefficients chosen at any intermediate node. The matrix G also depends on the number of packets received at the receiver. It is thus hard to guarantee a large minimum distance for the linear code generated by G, when the structure of G can not be controlled.
- Since the structure of G is not a priori known, it may be hard to decode the linear code generated by G.
- In order to guarantee in the above that the matrix Λ is invertible with high probabilities, the size of the underlying finite field $\mathbb{F}_{2^m}$ must chosen to be large [6]. However, this creates two new issues of its own:
  - If the bit error probability is $p \geq 0$, then the symbol error rate for $P_i(j)$ is easily seen to be $1-(1-p)^m$. This means that with larger finite field size, the probability of symbol error in the above increases.
  - Since the values α_i^j in the above must be transmitted in the header information, larger finite field sizes produce larger header overheads.

This motivates us to propose matrix network coding, a new paradigm for network coding that addresses some of the above issues.

III. Matrix Network Coding

In this section, we depart from this point of view and introduce a new method which we refer to as matrix network coding. For generality, we present our technique for packets defined over any arbitrary finite field $\mathbb{F}_{2^m}$, but in most practical applications we would be only interested in binary packet symbols (defined over the binary field $\mathbb{F}_2$).

To this end, let the underlying source packets $P_1, P_2, \cdots, P_N$ be defined over $\mathbb{F}_{2^m}$ and all have length $L = pq$. In matrix network coding each packet P_i is represented as a sequence of length q of p-dimensional vectors $(\mathbf{P}_i(1), \mathbf{P}_i(2), \cdots, \mathbf{P}_i(q))$, where

$$\mathbf{P}_i(j) = \left(P_i\left((j-1)p+1\right), P_i\left((j-1)p+2\right), \cdots, P_i(jp)\right)$$

for $j = 1, 2, \cdots, q$. Each node s performs matrix network coding as follows: Suppose that packets $P_s^1, P_s^2, \cdots, P_s^k$ arrive at node s. As in the above, each packet $P_s^i,\ i = 1, 2, \cdots, k$ can be written as q blocks of consecutive p symbols each

$$P_s^i = \left(\mathbf{P}_s^i(1), \mathbf{P}_s^i(2), \cdots, \mathbf{P}_s^i(q)\right),$$

where

$$\mathbf{P}_s^i(j) = \left(P_s^i\left((j-1)p+1\right), P_s^i\left((j-1)p+2\right), \cdots, P_s^i(jp)\right)$$

for $j = 1, 2, \cdots, q$. Node s chooses k matrices $A_1, A_2, \cdots, A_k$ of size $p \times p$ referred to as combining coefficients. Then it computes

$$\mathbf{P}_o(j) = A_1 P_s^1(j) + A_2 P_s^2(j) + \cdots + A_k P_s^k(j) \qquad (4)$$

for $j = 1, 2, \cdots, q$. The output packet at node s is then given by

$$P_o = \left(\mathbf{P}_o(1), \mathbf{P}_o(2), \cdots, \mathbf{P}_o(q)\right).$$

In the above, we first require the underlying combining coefficients to be *invertible*. This assumption is purely for technical analysis reasons and does not reduce the generality of our approach.

If $p = 1$ in the following, the scheme reduced to random linear network coding. Thus we can assume without any loss of generality that $p \geq 2$. As in random linear network coding it can be shown that any transmitted packet P by any node in the network and/or that received at the destination node is of the form

$$P = \left(\mathbf{P}(1), \mathbf{P}(2), \cdots, \mathbf{P}(q)\right),$$

where

$$\mathbf{P}(j) = B_1\mathbf{P}_1(j) + B_2\mathbf{P}_2(j) + \cdots + B_N\mathbf{P}_N(j),$$

for all $j = 1, 2, \cdots, q$, and for some $p \times p$ matrices $B_1, \cdots, B_N$. We say that P is a matrix linear combination of $P_1, \cdots, P_N$ with matrix network coding coefficients $B_1, \cdots, B_N$. It will be assumed without loss of generality that each intermediate (relaying) node chooses its combining coefficients such that for each output packet all non-zero matrix coding coefficients are invertible.

Suppose that at the destination node K $(\geq N)$ matrix linear combinations of $P_1, P_2, \cdots, P_N$ are received. For $j = 1, 2, \cdots, q$, let

$$\mathbf{P}_o^l(j) = \sum_{i=1}^{N} B_i^l \mathbf{P}_i(j)$$

for $l = 1, 2, \cdots, K$ be present at the receiver. Let

$$G = \begin{pmatrix} B_1^1 & B_1^2 & \cdots & \cdots & B_1^K \\ B_2^1 & B_2^2 & \cdots & \cdots & B_2^K \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ B_N^1 & B_N^2 & \cdots & \cdots & B_N^K \end{pmatrix}, \quad (5)$$

(and as in the above) $\tilde{\mathbf{P}}_o(j) = \left(\mathbf{P}_o^1(j), \cdots, \mathbf{P}_o^K(j)\right)$ and $\tilde{\mathbf{P}}(j) = (\mathbf{P}_1(j), \cdots, \mathbf{P}_N(j))$. Then we have

$$\tilde{\mathbf{P}}_o(j) = \tilde{\mathbf{P}}(j)G.$$

If $K = N$ and the matrix G in the above is invertible, then

$$\tilde{\mathbf{P}}(j) = \tilde{\mathbf{P}}_o(j)G^{-1},$$

for $j = 1, 2, \cdots, q$. This means that for each j, the vectors $\mathbf{P}_1(j), \cdots, \mathbf{P}_N(j)$ can be computed. Thus the input packets $P_1, \cdots, P_N$ can be reconstructed assuming no transmission errors.

More interestingly, in general the equation $\tilde{\mathbf{P}}_o(j) = \tilde{\mathbf{P}}(j)G$ is equivalent to the case when $\tilde{\mathbf{P}}(j)$ is encoded using a linear code defined over $\mathbb{F}_{2^m}$ whose generator matrix is G. The above linear code can in theory be used for error correction even if $P_1, P_2, \cdots, P_N$ are uncoded packets.

Let the messages of sources $S_1, S_2, \cdots, S_N$ be $M_1, M_2, \cdots, M_N$, respectively. We assume that message M_i, $i = 1, 2, \cdots, N$ is encoded into P_i, $i = 1, 2, \cdots, N$ using codebook $\mathcal{C}_i$, $i = 1, 2, \cdots, N$. Note that this formulation also covers the uncoded case where some of $\mathcal{C}_i$ represent uncoded transmission. The packets P_i's are matrix network coded over a wireless network. More specifically, the information bits of $P_1, P_2, \cdots, P_N$ are being encoded with a two-dimensional product code whose vertical codebooks are $\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_N$, and in the horizontal direction $\mathbf{P}_1(j), \cdots, \mathbf{P}_N(j)$ are encoded by a linear code whose generator matrix is G for $j = 1, 2, \cdots, q$. The question that we have to address is how to decode this product code.

Note that, in (4), we assumed that the same matrices $A_1, \cdots, A_k$ are used for matrix network coding for $j = 1, 2, \cdots, q$. It is also possible to choose potentially different encoding matrices for different values of $j = 1, 2, \cdots, q$. Such a generalization is obvious, and the methods and results established below can be easily generalized to cover this approach as well.

IV. Network Error Correction From Matrix Network Coding

We next pay attention to network error correction, when matrix network coding is employed. Decoding of product codes where all the constituent vertical codes are the same (e.g. $\mathcal{C}_1 = \mathcal{C}_2 = \cdots = \mathcal{C}_N$) has been extensively studied in the literature [5], [9], [11]. In fact, many powerful classes of capacity achieving codes, for example, have the structure of such product codes. If the constituent codes of a product code are systematic and have simple MAP decoding algorithms, then the product code can be iteratively decoded. A simpler way to decode the product code is by first decoding the rows and then the columns of the product code. This is less complex than full MAP decoding of the product code, or even the approximate MAP decoding (e.g. using iterative decoding), but may not provide as much decoding gain.

If the network is a fixed network, then it may be possible to architect matrix linear combining coefficients at each node such that the matrix G corresponds to a code with a good structure and lend itself to a low complexity algebraic decoder, or MAP or approximate MAP decoder. However in mobile networks, where the network topology and underlying channels change frequently, it is a formidable task to pre-construct G in advance. In such cases, it is reasonable to assume that G is a randomly constructed matrix known at the receiver without any particular structure. In particular in most cases, it can not be assumed that G is systematic. Additionally, since different sources may transmit using different encoders, it may not be assumed that $\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_N$ are the same. Thus, we have to decode a general product code with potentially distinct vertical constituent encoders per each column and with a horizontal linear code with a randomly generated generator matrix G. However, it can be assumed that the vertical constituent codes $\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_N$ have simple decoding algorithm (either hard decision, soft decision or MAP decoding). This assumption is reasonable since each one of these codes is to begin with selected for encoding (at their respective sources) for its reasonable performance and implementation complexities.

Much is not known about decoding a linear code with an arbitrary generator matrix G, except that in general this is NP-hard [2]. Fortunately in our case, we usually have small values of p and $K-N$. Additionally, in most cases, we choose the ground field to be $\mathbb{F}_2$. This is reasonable since (as we will discuss in the next section), for example, in the case of matrix network coding with $p = 8$ over $\mathbb{F}_2$, the probability of failure equivalent to random linear network coding over $\mathbb{F}_{2^{128}}$ is achieved.

The code G, then, corresponds to a linear binary code of length Kp and dimension Np. It is well-known that any finite field linear code (and in particular any binary linear code) has a trellis representations [3], [9]. This trellis representation has at most $\min(2^{Np}, 2^{(K-N)p})$ states. If $(K-N)p$ is reasonably small, then this trellis representation can be used to produce log-likelihood probabilities for the symbols of $P_1, P_2, \cdots, P_N$ [9]. These log-likelihood then in turn can be used for MAP decoding of $\mathcal{C}_i$. However if $(K-N)p$ is not small, then this process becomes too complex. For example, if $K = N + 2$ and $p = 8$, then the trellis may have as many as 65536 states, which may be too complex to implement.

A. Sphere Decoding Algorithm For Channels with Gaussian Noise

Another approach that seems reasonable is based on a generalization of sphere decoder [11] using list decoding. We advocate this method of decoding for Gaussian noise channels. These not only include AWGN, but also include Rayleigh and Rician fading channels.

Suppose that $\mathbf{R}(j), j = 1, 2, \cdots, q$ is the received word corresponding to $\tilde{\mathbf{P}}_o(j)$ defined over $\mathbb{F}_{2^m}$ corrupted by Gaussian noise. Let $m = 1$ for simplicity. A generalization of $m \geq 2$ is obvious, and the methods and results established below can be easily generalized to cover the $m \geq 2$ cases.

- We first compute a parity check matrix H for the generator matrix G.
- Fix an integer w depending on the probability of bit error P_{err}. The value
$$w = \max\left(\left\lfloor \frac{p(K-N)}{2} \right\rfloor, \lfloor KpP_{\text{err}} + 3\sigma_{\text{err}} \rfloor + 1\right)$$
is an example of a reasonable choice in most cases, where $\sigma_{\text{err}} = \sqrt{KpP_{\text{err}}(1-P_{\text{err}})}$. This number is chosen, since, assuming independent bit error probability P_{err}, the average number of bit errors is KpP_{err} in packets of length Kp bits. The variance of the number of errors is $KpP_{\text{err}}(1-P_{\text{err}})$, so w is larger than the mean plus three standard deviations of number of bit errors, and thus captures a good portion of most probable outcomes of Bernoulli process of the error.
- For each $j = 1, 2, \cdots, q$,
  - Given $\mathbf{R}(j)$, we produce log-likelihood probabilities for the elements of Kp dimensional $\tilde{\mathbf{P}}_o(j)$.
  - Find the w locations of $\tilde{\mathbf{P}}_o(j)$ with the lowest absolute value of log-likelihoods. Assume that these elements are in locations $1 \leq i_1 < i_2 < \cdots < i_w \leq Kp$ of $\tilde{\mathbf{P}}_o(j)$.
  - Let $\mathcal{R}$ denote the vector formed by replacing the elements in all other locations (except $i_1 < i_2 < \cdots < i_w$) of $\mathbf{R}(j)$ with 1 if the corresponding log-likelihood is positive and with 0 if the corresponding log-likelihood is negative. In other words $\mathcal{R}$ is formed by performing hard decisions on $\mathbf{R}(j)$ in locations other than $i_1 < i_2 < \cdots < i_w$.
  - Consider all 2^w vectors $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_{2^w}$ formed by replacing the elements of $\mathcal{R}$ in locations $i_1 < i_2 < \cdots < i_w$ with all possible binary numbers of length w (i.e. binary expansions of $0, 1, \cdots 2^w - 1$).
  - Identify those vectors $\mathcal{R}_i,\ i = 1, 2, \cdots, 2^w$ that satisfy $\mathcal{R}_i H^T = 0$. These represent those $\mathcal{R}_i$ that are codewords of G. Without loss of generality (by possible relabeling) assume that these vectors are $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_u$ for some $0 \leq u \leq 2^w - 1$ ($u = 0$ if none of the $\mathcal{R}_i$ satisfies $\mathcal{R}_i H^T = 0$).
  - For each $\mathcal{R}_i,\ i = 1, 2, \cdots, u$, compute the input bit vector $v_i,\ i = 1, 2, \cdots, u$ that, when encoded with the generator matrix G, produces the codeword $\mathcal{R}_i$.
  - If $u > 0$, then assuming that the only possible values of $\tilde{\mathbf{P}}(j)$ are $v_i,\ i = 1, 2, \cdots, u$ and using the received vector $\mathbf{R}(j)$ compute the log-likelihood of bits of $\tilde{\mathbf{P}}(j)$. In other words, we compute $p_l^1 = P\left(\mathbf{R}(j) | v_i, i = 1, 2, \cdots u \text{ with } l\text{-th bit equal to } 1\right)$, and $p_l^0 = P\left(\mathbf{R}(j) | v_i, i = 1, 2, \cdots u \text{ with } l\text{-th bit equal to } 0\right)$. The LLR of the l-th bit of $\tilde{\mathbf{P}}(j)$ is $\log(p_l^1/p_l^0)$.
  - If $u = 0$, then set the values of the log-likelihoods of all the bits of $\tilde{\mathbf{P}}(j)$ to zeroes.
- The log-likelihoods of all elements of $P_i,\ i = 1, 2, \cdots, N$ are now computed. Using this and a (MAP, soft, or hard) decoder for $\mathcal{C}_i$, the message of each source can be computed.

One can save some processing energy by generalizing the above decoding algorithm to binary symmetric channels as we will discuss in the next section.

B. Coset Decoding Algorithm For Binary Symmetric Channels

One of the potential benefits of network error correction may be in energy efficient communications. For example, one may use codes with simple encoders at the sources and efficient hard decision decoders at the destinations in order to save energy, and rely on network error correction to reduce the probability of error. In order to further save energy in the decoding process, the receiver may choose to perform hard decision decoding on the received word. Thus, we give a procedure for hard decision decoding based on coset decoding.

Suppose that $\mathbf{R}(j),\ j = 1, 2, \cdots, q$ is hard decision received. This can be formed, for example, by applying hard decision to the received signal. Then $\mathbf{R}(j), j = 1, 2, \cdots, q$ can be modeled by the output of a binary symmetric channel with transition probability P_{err} upon the input $\tilde{\mathbf{P}}_o(j)$.

- We first compute a parity check matrix H for the generator matrix G. Let $\mathcal{C}$ denote the codebook generated by G. We list all the $2^{p(K-N)}$ cosets $\mathbf{x}_i + \mathcal{C},\ i = 1, 2, \cdots, 2^{p(K-N)}$ with $\mathbf{x}_i$ selected to be a binary string with minimal Hamming weight in the coset $\mathbf{x}_i + \mathcal{C}$. We list $\mathbf{x}_i H^T$ for $i = 1, 2, \cdots, 2^{p(K-N)}$.
- For each $j = 1, 2, \cdots, q$,
  - Given $\mathbf{R}(j)$, compute the syndrome vector $\mathbf{R}(j)H^T$. Find the unique vector $\mathbf{x}_\star$ (between all $\mathbf{x}_i,\ i = 1, 2, \cdots, 2^{p(K-N)}$) such that $\mathbf{R}(j)H^T = \mathbf{x}_\star H^T$.
  - Compute a vector $\mathcal{R} = \mathbf{R}(j) + \mathbf{x}_\star$. Clearly this is a codeword of G.
  - Compute the input bit vector v_j that, when encoded with the generator matrix G, produces the codeword $\mathcal{R}$. Let this be the estimate of $\tilde{\mathbf{P}}(j)$.
- The estimates of all bits of $P_i,\ i = 1, 2, \cdots, N$ are now computed. Using this and a hard decision decoder for $\mathcal{C}_i$, the message of each source can be computed.

V. Comparison of Matrix Network Coding and Random Linear Network Coding

In this section, we will establish the following theorem:

Theorem 1. *For the same amount of header overhead per packet, matrix network coding outperforms random linear network coding in the following senses:*

- *Matrix network coding corrects more errors than random linear network coding, particularly when p is large.*
- *When there are no transmission errors, matrix network coding and random linear network coding perform the same.*

- *The arithmetic required for matrix network coding is performed over a smaller finite field which corresponds to less hardware complexity. However, although in matrix network coding more multiplications are needed, the total computation complexity of matrix network coding is slightly smaller.*

Proof: Proof will be provided in the following subsections. ∎

A. Maximum Network Error Correction Capability

We will show that matrix network coding error correction is superior to random linear network coding error correction.

Proof: Suppose that K matrix linear combination (respectively linear combinations) of the transmitted packets are received in the matrix (respectively random linear) network coding scenario outlined above. The generator matrix G in the above produces an $[Kp, Np]$ linear code (respectively a $[K, N]$ linear code). By the MDS bound G can have minimum distance as high as $Kp - Np + 1$ (respectively $K - N + 1$). This means that G could guarantee correction of all the error patterns of as much as $(K-N)p/2$ (respectively $(K-N)/2$) errors. Since $p \geq 2$, matrix network coding can correct more errors than random linear network coding. ∎

B. Header Overhead

In random linear network coding error correction, each network coded packet $P = \sum_{k=1}^{N} \alpha_i P_i$ must have a header that contains the coefficients $\alpha_i, i = 1, 2, \cdots, N$. Since each $\alpha_i \in \mathbb{F}_{2^m}$, as many as mN bits can be required for each header. Let us next consider the header in matrix network coding. If the coefficients of the underlying matrices are defined over $\mathbb{F}_{2^{m_1}}$, then a similar argument as in the above shows that in general as many as $p^2 m_1 N$ bits may be required to include all the possible matrix coefficients in the header of each packet. In fact, as many as

$$N \log_2 S(p, m_1) < p^2 m_1 N$$

bits are needed, where $S(p, m)$ is the number of $p \times p$ non-singular matrices over $\mathbb{F}_{2^{m_1}}$, i.e., $S(p, m_1) = \sum_{k=1}^{p} \log_2 \left(2^{m_1 p} - 2^{m_1 (k-1)}\right)$. So it seems for larger p, the header must be substantially larger than the random linear network coding case. However, as we will discuss below, this is not the case. In order to show this, let us keep the header overhead of matrix network coding $\left(N \log_2 S(p, m_1) < N m_1 p^2\right)$ and random linear network coding Nm at the same level. Thus we must choose $m \simeq m_1 p^2$.

The next main question is, given the same header overhead, if we will have approximately the same probability of failure. The following section addresses this issue.

C. Decoding Performance Under No Transmission Errors

Let us first recall the rational for choosing a finite field $\mathbb{F}_{2^m}$ for $m > 1$ for random linear network coding. As in the above, consider the reception of N linear combinations of packets $P_1, P_2, \cdots, P_N$ given by (1) for $l = 1, 2, \cdots, N$, where $\alpha_i^l \in \mathbb{F}_{2^m}$ for $i, l = 1, 2, \cdots, N$. We define $P_i(j), P_o^l(j) \in \mathbb{F}_{2^m}$ for $j = 1, 2, \cdots, L$ as in the above. It follows (2) that, defining $\tilde{P}_o(j) = \left(P_o^1(j), \cdots, P_o^N(j)\right)$, $\tilde{P}(j) = (P_1(j), \cdots, P_N(j))$, and Λ in (3) as in the above, we have $\tilde{P}_o(j) = \tilde{P}(j)\Lambda$. This is a system of linear equations in $\mathbb{F}_{2^m}$. We would like the matrix Λ to be invertible, so that the above equation can be solved to give $\tilde{P}(j) = \tilde{P}_o(j)\Lambda^{-1}$ for $j = 1, 2, \cdots, L$. If elements of Λ are chosen at random in $\mathbb{F}_{2^m}$ according to a uniform i.i.d. distribution, then the following result is well-known:

Lemma 1. *Suppose that elements of Λ are chosen at random according to a uniform i.i.d. distribution. Then Λ is not singular with probability*

$$\prod_{k=1}^{N} \left(1 - 2^{-m(N-k+1)}\right). \tag{6}$$

From Lemma 1, if m and N are both small, then the chances of failure of the above algorithm is not small. For example, if $m = 1$ and $N = 2$, then the chances of failure is $\frac{5}{8}$. Thus a larger finite field size is selected in order to reduce the probability of failure, i.e., Λ not being invertible.

Let us next consider the probability of failure of matrix network coding over a finite field $\mathbb{F}_{2^{m_1}}$ and compute m_1 so that a similar level of protection can be achieved. To this end, suppose that at a destination node $K(= N)$ matrix linear combinations of $P_1, P_2, \cdots, P_N$ are received. Using the same notation, for $j = 1, 2, \cdots, q$, we have $\mathbf{P}_o^l(j) = \sum_{i=1}^{N} B_i^l \mathbf{P}_i(j)$ for $l = 1, 2, \cdots, N$. Letting Ω be given by (5) with $K = N$, $\tilde{\mathbf{P}}_o(j) = \left(\mathbf{P}_o^1(j), \cdots, \mathbf{P}_o^N(j)\right)$, and $\tilde{\mathbf{P}}(j) = (\mathbf{P}_1(j), \cdots, \mathbf{P}_N(j))$, we have

$$\tilde{\mathbf{P}}_o(j) = \tilde{\mathbf{P}}(j)\Omega.$$

If the matrix Ω in the above is invertible, then

$$\tilde{\mathbf{P}}(j) = \tilde{\mathbf{P}}_o(j)\Omega^{-1}$$

for $j = 1, 2, \cdots, q$, and for each j the vectors $\mathbf{P}_1(j), \cdots, \mathbf{P}_N(j)$ can be computed.

Lemma 2. *Suppose that in a matrix network coding scenario, each node chooses invertible $p \times p$ combining coefficients defined over $\mathbb{F}_{2^{m_1}}$ according to a uniform i.i.d. distribution over all such matrices, while making sure that all the non-zero matrix network coding coefficients at the output are non-singular. Letting*

$$S(p, m_1) = \prod_{i=0}^{p-1} \left(2^{pm_1} - 2^{im_1}\right), \tag{7}$$

the probability κ that the matrix Ω is invertible satisfies

$$\kappa \geq \prod_{i=1}^{N-1} \left(1 - \frac{2^{ip^2 m_1}}{S(p, m_1)^N}\right). \tag{8}$$

Proof: The number of non-singular $p \times p$ matrices is easily seen to be (7). The means that the number of all possible arrangements for Ω is at most $S(p, m_1)^{N^2}$. We now count the number of possibilities that produce non-singular matrices Ω. The first nonsingular block of matrices $B_1^1, B_1^2, \cdots, B_1^N$

of Ω can be arbitrarily selected, so there are $S(p,m_1)^N$ possibilities. Since Ω is linearly independent, none of the rows of second block $B_2^1, B_2^2, \cdots, B_2^N$ of Ω can be linear combinations of the first block. This means that none of the rows of second block $B_2^1, B_2^2, \cdots, B_2^N$ of Ω can attain any of the $2^{m_1 p}$ possible linear combinations of rows of $B_1^1, B_1^2, \cdots, B_1^N$. Since there are p possible rows in the second block, at most $2^{m_1 p^2}$ possibilities is ruled out and the second block can take on at least $S(p,m_1)^N - 2^{m_1 p^2}$ possibilities. Continuing in this way, we observe that the third block of Ω has only rows that are not linear combinations of rows of the first and second blocks. Thus at least there are $S(p,m_1)^N - 2^{2m_1 p^2}$ possibilities. Continuing in this manner, we observe that there are at least

$$S(p,m_1)^N \prod_{i=1}^{N-1} \left(S(p,m_1)^N - 2^{i m_1 p^2} \right) \tag{9}$$

possible values for Ω to be non-singular. Thus dividing (9) by $S(p,m_1)^{N^2}$ yields the lower bound for the probability κ, as specified in (8). This proves the lemma. ■

We can now prove the second result of Theorem 1:

Lemma 3. *For the same header overhead, assuming reception of N copies of N network coded packets and no transmission errors, the probability that matrix inversion fails in both matrix network coding and random linear network coding is the same as p grows large.*

Proof: For the same overhead amount in the above, we must have

$$N \log_2 S(p,m_1) = Nm.$$

This gives $S(p,m_1) = 2^m$. It is also easy to see that $m_1 p(p-1) \leq \log_2 S(p,m_1) \leq m_1 p^2$, with $\log_2 S(p,m_1) \simeq m_1 p^2$ as p grows large. It follows that (6) and the right hand side of (8) are the same as p grows large. ■

The following corollary is also of interest:

Corollary 1. *If the underlying $p \times p$ combining matrices are arbitrarily selected (and not necessarily be required to be non-singular), then the header overhead remains the same as p grows large.*

Proof: We have shown in the above that $\log_2 S(p,m_1) \simeq m_1 p^2$. This is equal to the header that requires to describe an arbitrary $p \times p$ matrix defined over $\mathbb{F}_{2^{m_1}}$. This proves the result. ■

It is easy to see that even if $m_1 = 1$, for reasonable values of $p \geq 4$, the conclusions of Lemma 3 holds.

D. Computation Complexity

Now we are ready to prove the third result of Theorem 1.

Proof: To this end, multiplication in the finite field of size $\mathbb{F}_{2^m}$ requires approximately about $\mathcal{O}(m^2)$ binary operations. In addition, $r \times r$ matrix inversion requires about $\mathcal{O}(r^3)$ finite field multiplications. Therefore, since the matrix Λ has size $N \times N$ and matrix Ω has size $Np \times Np$, random linear network coding requires $\mathcal{O}(N^3 m^2)$ binary operation, whereas matrix coding requires $\mathcal{O}(N^3 p^3 m_1^2)$. If we replace $m = m_1 p^2$, these correspond to $\mathcal{O}(N^3 p^4 m_1^2)$ versus $\mathcal{O}(N^3 p^3 m_1^2)$. Thus the complexity is slightly smaller. ■

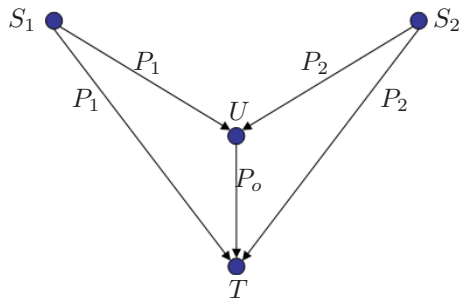

Fig. 1. 4-node network with two source nodes S_1, S_2, one intermediate node U, and one destination node T, where $P_o = (\mathbf{P}_o(1), \mathbf{P}_o(2), \cdots, \mathbf{P}_o(q))$ and $\mathbf{P}_o(j) = B_1\mathbf{P}_1(j) + B_2\mathbf{P}_2(j)$, $j = 1, 2, \cdots, q$ for MNC; and $P_o = \alpha_1 P_1 + \alpha_2 P_2$ for RLNC.

VI. Simulations

Since the network error correction capability is described as only maximum for matrix network coding (MNC) and random linear network coding (RLNC), we provide simulation results to document the actual performance of MNC with the sphere decoding algorithms outlined above and that of RLNC. The simulations do not attempt to characterize precisely the network error correction performance for networks with various sizes and topologies, but seek to give an idea of the performance of MNC for a simple network.

Our experiments are run on a 4-node fixed network with two source nodes S_1, S_2, one destination node T, and one intermediate node U. Each source transmits an uncoded packet of length L bits, where L is a power of 2. MNC or RLNC is done at the intermediate node U. We assume that the noisy versions of transmitted uncoded packets P_1 and P_2 from sources S_1 and S_2 are heard both at the destination node T and the intermediate node U. We further assume that the intermediate node U can decode P_1 and P_2 perfectly. This is illustrated in Figure 1.

In the matrix network correction scheme, the intermediate node U applies MNC to packets P_1 and P_2, respectively, with matrix coefficients B_1 and B_2 over the binary field $\mathbb{F}_2$, i.e., the matrix network coded packet $\mathbf{P}(j)$ is of the form

$$\mathbf{P}(j) = B_1\mathbf{P}_1(j) + B_2\mathbf{P}_2(j),\ j = 1, 2, \cdots, q,$$

where

$$P_i = (\mathbf{P}_i(1), \cdots, \mathbf{P}_i(q)),\ i = 1, 2.$$

The underlying modulation for packets P_1, P_2, and matrix network coded packets $\mathbf{P}(j)$ is assumed to be BPSK (where bits 0 and 1 are mapped into real numbers 1 and -1, respectively). It is assumed that the modulated waveforms of P_1, P_2, and $\mathbf{P}(j)$ are corrupted by real Gaussian noise $\mathcal{N}(0, \sigma^2)$ with mean zero and variance σ^2, and they are present at the destination node T. The signal to noise power ratio (SNR) is then equal to $\frac{1}{\sigma^2}$. At the destination node T, the sphere decoding algorithm is then executed.

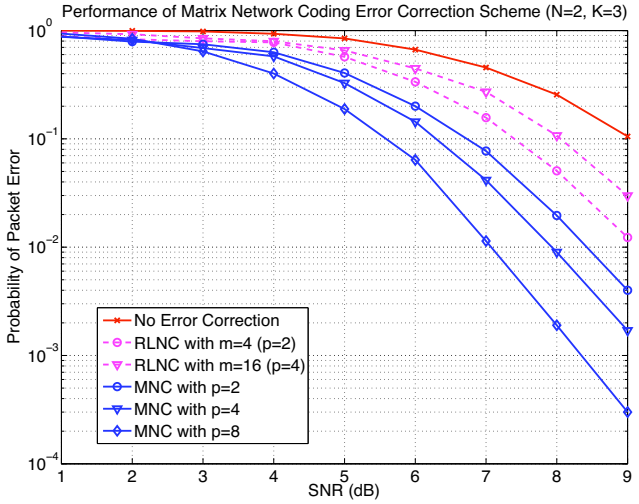

Fig. 2. The error correction performance of the matrix network coding and the random linear network coding schemes with $N = 2,\ K = 3$, and $L = 16$ [bits] for $p = 2, 4, 8$.

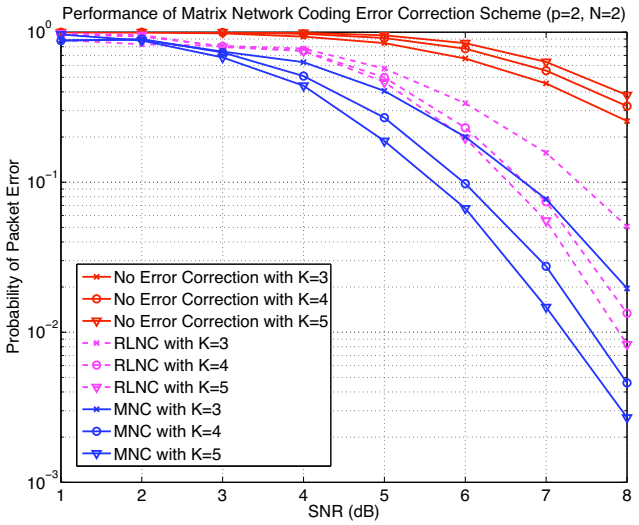

Fig. 3. The error correction performance of the matrix network coding and the random linear network coding schemes with $p = 2,\ N = 2$, and $L = 16$ [bits] for $K = 3, 4, 5$.

We also consider the error correction scheme of RLNC as a comparison. The intermediate node U performs RLNC of P_1 and P_2 over $\mathbb{F}_{2^m}$ and produces $\alpha_1 P_1 + \alpha_2 P_2$ with $\alpha_1, \alpha_2 \in \mathbb{F}_{2^m}$, where $m = p^2$ is chosen so that the decoding performance under no transmission errors is almost the same between MNC and RLNC, while their header overheads are kept at the same level. Again the BPSK modulation is performed as in the above. At the destination node T, hard decisions are made on symbols of P_1, P_2 and $\alpha_1 P_1 + \alpha_2 P_2$ producing $\hat{P}_1$, $\hat{P}_2$, and $\hat{P}_3$, respectively. These are then used for the network error correction. This is the approach commonly proposed in the literature. In our case as discussed above, this gives a $[3, 2]$ code, since for every pair of corresponding symbols of P_1 and P_2, we have three corresponding coded symbols of P_1, P_2, and $\alpha_1 P_1 + \alpha_2 P_2$. This code can not correct any errors as its minimum Hamming distance is at most 2 (from the MDS bound). However it can detect one symbol error, since, whenever $\hat{P}_3 \neq \alpha_1 \hat{P}_1 + \alpha_2 \hat{P}_2$, we can conclude that an error has happened. Here we assume no CRC or other check mechanisms to know which packet was received correctly or not. If P_1, P_2, and $\alpha_1 P_1 + \alpha_2 P_2$ is received wrong, the destination node does not know which packet is correctly received and which is not. The first case is considered as no error correction. In the second case, the destination node T uses the sphere decoding algorithm to harness the diversity effect, and thus corrects more errors than the hard decisions that is used in the first case.

We used $p = 2, 4, 8$, and $L = 16$ [bits]. In this case, we produced 100 pairs of random matrix network coding coefficients or random linear network coding coefficients at intermediate node U. For each pair of coefficients, we produced $1,000$ random pairs of packets for P_1 and P_2. For decoding, we use the sphere decoder described above in Section IV-A.

The probabilities of packet error for MNC and RLNC with $N = 2$, $K = 3$, and $L = 16$ [bits] for various $p = 2, 4, 8$ are shown in Figure 2. On average, MNC provides a coding gain of $2.5 - 3.5$ [dB] more than the no error correction case, and that of $0.7 - 2$ [dB] more than RLNC, even if the sphere decoding algorithm helps the RLNC scheme harness the diversity effect by using information from relay node U. The gap of a coding gain between MNC and either no error correction or RLNC becomes larger as p becomes larger, because the singleton bound can be p-fold increased by employing $p \times p$ matrix coefficients; the radius of the sphere decoder also becomes larger; and thus it can capture more candidates for the transmitted codewords.

As the code rate becomes lower from $R = \frac{N}{K} = \frac{2}{3}$ to $\frac{2}{5}$, i.e., by receiving a larger number of network coded packets, larger error correction capabilities are achieved from both MNC and RLNC with the sphere decoder. This is shown in Figure 3. Note that the no error correction case performs worse as the destination node T receives a larger number of network coded packets, because T does not have CRC or other check mechanisms as previously mentioned.

VII. Conclusions

This paper investigates matrix network coding, which improves the error-correcting capability of random linear network coding by employing matrix coding coefficients. Despite the increase in the dimension of the coding coefficient, the header overhead is kept the same as the random linear network coding scheme, because matrix network coding requires a smaller field size to guarantee the same decoding probability under no transmission errors. This paper shows that the $p \times p$ matrix network coding can provide up to p times larger minimum distance as compared to random linear network coding. We also present the sphere decoding scheme for matrix network

coding. Numerical results show that, when $N+1$ linear combinations of N information packets are received, the use of sphere decoding along with matrix network coding provides around $0.7-2$ [dB] coding gain as compared to the random linear network coding schemes. On the other hand, when less than the N linear combinations of N information packets are received, the proposed sphere decoding algorithm can be modified to enable the decoding N information packets, provided that there exists redundancy in these packets (i.e., if these packets are themselves streams of coded symbols).

REFERENCES

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

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

[3] G. D. Forney and M. D. Trott, "The dynamics of group codes: state spaces, trellis diagrams, and canonical encoders," *IEEE Trans. on Inf. Theory*, vol. 39, no. 5, pp. 1491–1513, Sept. 1993.

[4] C. Fragouli and E. Soljanin, "Network coding applications," *Foundations and Trends in Networking*, vol. 2, no. 1, pp. 1–133, 2007.

[5] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inf. Theory*, vol. 42, pp. 429–445, Mar. 1996.

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

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

[8] 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.

[9] J. Lodge, R. Young, P. Hoeher, and J. Hagenauer, "Separable MAP filters for the decoding of product and concatenated codes," *Proc. of IEEE ICC*, pp. 1740–1745, May 1993.

[10] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: practical wireless network coding," *Proc. of ACM SIGCOMM*, 2006.

[11] R. M. Pyndiah, "Near-optimum decoding of product codes: block turbo codes," *IEEE Trans. on Commun.*, vol. 46, no. 8, pp. 1003–1010, Aug. 1998.

[12] R. W. Yeung and N. Cai, "Network error correction, Part I: Basic concepts and upper bounds," *Commun. in Informations and Systems*, vol. 6, no. 1, pp. 19–36, 2006.

[13] R. W. Yeung and N. Cai, "Network error correction, Part II: Lower bounds," *Commun. in Informations and Systems*, vol. 6, no. 1, pp. 37–54, 2006.

[14] IEEE P802.11s/D3.0, "Part 11: Wireless LAN MAC and PHY specifications, Amendment 10: Mesh Networking," March 2009