Rank Deficient Decoding of Linear Network Coding

*Zhiyuan Yan**, *Hongmei Xie**, and *Bruce W. Suter*†

*Department of Electrical and Computer Engineering, Lehigh University, Bethlehem, PA 18015, USA
†Air Force Research Laboratory, Rome, NY 13441, USA

***Abstract*—Since all packets in linear network coding are subject to linear combinations, in all existing network coding schemes, a full rank of received packets is required to start decoding. This requirement unfortunately results in long delays and low throughputs. In this work we propose two classes of rank deficient decoders that work for rank deficient received packets. Within either class, different decoding strategies have been proposed for tradeoffs between delay/throughput and data accuracy. The decoders of the first class take advantage of the sparsity inherent in data and produce the data vectors with the smallest Hamming weight. Since these decoders have high complexities, we propose a class of decoders with polynomial complexities based on linear programming. Both classes of decoders can recover data from fewer received packets and hence achieve higher throughputs and shorter delays than the full rank decoder.**

***Index Terms*—Linear network coding, rank deficient decoding, linear programming**

I. Introduction

Due to its promise of significant throughput gains as well as other advantages, network coding [1]–[3] is already used or considered for a wide variety of wired and wireless networks (see, for example, [4]–[8]). One significant drawback of network coding is that a full rank of received packets at the receiver nodes of a multicast (or a unicast) is needed before decoding can start, leading to long delays and low throughputs, especially when the number of packets of a session is large. This is particularly undesirable for applications with stringent delay requirements.

Aiming to solve this problem, we propose rank deficient decoding for linear network coding, which can start even when the received packets are not full rank. By reformulating the decoding problem of network coding in a different fashion, the decoding problem reduces to a collection of syndrome decoding problems. Solving these syndrome decoding problems, rank deficient decoding leads to smaller delays and higher throughputs, at the expense of possible decoding errors. Specifically, we propose two classes of rank deficient decoders with different complexities. The decoders of the first class, called Hamming norm (HN) decoders, take advantage of the sparsity inherent in data and produce the data vectors with the smallest Hamming weight. Since the HN decoders have high complexities for large size systems, we propose a class of decoders based on linear programming, referred to as linear programming (LP) decoders. Considering linear programming relaxation of the Hamming norm decoders and solving them by using standard linear programming procedures, the linear programming decoders have polynomial complexities and are much more affordable. Both classes of decoders recover data from fewer received packets and hence achieve higher throughputs and shorter delays than the full rank decoder. Since these decoders could produce erroneous outputs, within each class several different decoding strategies have been proposed for different tradeoffs between delay/throughput and data accuracy, and they include the full rank decoder of network coding as a special case.

In the literature, there are two related different approaches to dealing with the synergy of network coding and compressive sensing, and they also aim for different applications. Our work is quite different from both existing approaches. Above all, our reformulation of the decoding problem in network coding is novel, and this reformulation was not considered in the open literature to the best of our knowledge. One approach was proposed in [9], where statistical property of data blocks are taken advantage of to alleviate the "all-or-nothing" drawback of network coding in distributed storage systems. In this approach, random linear network coding is used to encode coded blocks in distributed storage networks. Hence, this approach is not directly comparable to our work, which focuses on the decoding issue of linear network coding in general and applies to a wide variety of applications. The other approach [10], [11] aims to take advantage of the statistical correlation of data generated by distributed sensor networks. A salient feature of this approach is that in theory data are real values and linear combinations are now performed over the real (or complex) field. The rationale for this is that the real representation of data is a more natural one for sensor networks [10], [11]. In practice, data are represented in a finite precision system. It has been shown that information loss due to finite precision grows with the network size [12]. In contrast, in our work network coding remains over some finite fields, and hence our scheme does not suffer the information loss due to finite precision as the approach in [10], [11]. Thus, the full rank decoder remains the most relevant previous work, and henceforth we compare our rank deficient decoders with the full rank decoder only.

II. Rank Deficient Decoding

A. System Model

In this work, we treat all packets as N-dimensional row vectors over some finite field GF(q), where q is a prime power. Also, we focus on linear network coding (LNC) only, which was shown to be optimal in most cases [2]. Finally, we assume that the network is error-free, and error control (see, for example, [13]–[16]) is not embedded in network coding.

Suppose a source node of a unicast or multicast injects a collection of n data packets (or row vectors over GF(q)), $\mathbf{X}_0, \mathbf{X}_1, \cdots, \mathbf{X}_{n-1}$, into the network. At any sink node, m packets (or row vectors over GF(q)), $\mathbf{Y}_0, \mathbf{Y}_1, \cdots, \mathbf{Y}_{m-1}$, are received, where $\mathbf{Y}_i = \sum_{j=0}^{n-1} a_{i,j}\mathbf{X}_j$ for $i = 0, 1, \cdots, m-1$ and

$a_{i,j} \in \text{GF}(q)$. Since the sink node can locally generate more linear combinations of $\mathbf{Y}_0, \mathbf{Y}_1, \cdots, \mathbf{Y}_{m-1}$, it is assumed that $\mathbf{Y}_0, \mathbf{Y}_1, \cdots, \mathbf{Y}_{m-1}$ are linearly independent, which implies that $m \leq n$. That is, the $m \times n$ matrix $\mathbf{A} = [a_{i,j}]$, often called the global coding kernel matrix, has a rank of m.

B. Full Rank Decoder

Let us further denote the matrices $\left[\mathbf{X}_0^T\, \mathbf{X}_1^T\, \cdots\, \mathbf{X}_{n-1}^T\right]^T$ and $\left[\mathbf{Y}_0^T\, \mathbf{Y}_1^T\, \cdots\, \mathbf{Y}_{m-1}^T\right]^T$ as $\mathbf{X}$ and $\mathbf{Y}$, respectively, where T is the matrix transpose operator. Since $\mathbf{Y} = \mathbf{AX}$, the sink node can recover the transmitted data packets by reversing the encoding of the data packets by the network. This is easily achievable when $m = n$, as the sink node can recover the data packets by computing $\mathbf{X} = \mathbf{A}^{-1}\mathbf{Y}$. Thus, the decoding in network coding starts only after the sink node has received n linearly independent combinations of the transmitted data packets. The required number of linearly independent packets received by the sink node leads to longer delays and lower throughputs, which may be undesirable for some applications.

C. Rank Deficient Decoding

We can formulate the data recovery problem at the sink node in a different way. Let us consider symbol l of $\mathbf{Y}_i$, and we have $Y_{i,l} = \sum_{j=0}^{n-1} a_{i,j} X_{j,l}$ for $i = 0, 1, \cdots, m-1$ and $l = 0, 1, \cdots, N-1$. Let us denote the column vectors $(Y_{0,l}\, Y_{1,l}\, \cdots\, Y_{m-1,l})^T$ and $(X_{0,l}\, X_{1,l}\, \cdots\, X_{n-1,l})^T$ as $\mathbf{V}_l$ and $\mathbf{W}_l$, respectively. Clearly, we have $\mathbf{V}_l = \mathbf{AW}_l$ for $l = 0, 1, \cdots, N-1$. The sink node can recover the data packets if it can obtain $\mathbf{W}_l$ from

$$\mathbf{V}_l = \mathbf{AW}_l \text{ for } l = 0, 1, \cdots, N-1. \tag{1}$$

Eq. (1) shows that the data recovery problem at the sink node can be viewed as N parallel decoding problems in Eq. (1), each corresponding to one symbol in the packet (or row vector). These N parallel decoding problems are equivalent to the decoding problem of linear network coding.

This reformulated problem is related to two well known decoding problems. First, if we treat the $m \times n$ matrix $\mathbf{A}$ as a parity check matrix for a linear block code of length n and dimension $n - m$, the decoding problem in Eq. (1) is closely related to a syndrome decoding problem. That is, the sink node needs to recover $\mathbf{W}_l$ based on the syndrome $\mathbf{V}_l$. Second, if we treat $\mathbf{W}_l$ as a data vector and $\mathbf{A}$ a measurement matrix, this is analogous to the decoding problem in compressive sensing.

D. Hamming Norm Decoders

Since the data recovery problem at any sink node is equivalent to a collection of parallel problems in Eq. (1), we focus on one such problem. In other words, we try to solve $\mathbf{V} = \mathbf{AW}$ for $\mathbf{W}$, where $\mathbf{V}$ and $\mathbf{W}$ are m- and n-dimensional column vectors, respectively, and $\mathbf{A}$ remains an $m \times n$ matrix with full rank ($m \leq n$).

For a linear block code of length n and dimension $n - m$ with a parity check matrix $\mathbf{A}$, $\mathbf{V} = \mathbf{AW}$ can be viewed as a syndrome of the received vector $\mathbf{W}$. It is well known that for a linear block code, the syndromes have a one-to-one correspondence with its cosets, each of which is of size q^{n-m}. In other words, all vectors in a coset lead to the same syndrome. Thus, solving $\mathbf{V} = \mathbf{AW}$ for $\mathbf{W}$ is equivalent to finding a vector within a coset.

If no side information is available, we can make a decision within the coset by taking advantage of some inherent properties of the data vector. In this work, we proceed by relying on the sparsity of the data vector, which is well justified in many applications. That is, the proposed Hamming norm decoders produce the vector with the smallest Hamming weight in the coset.

As is common in the compressive sensing literature, we consider two possible scenarios for sparsity. First, when $\mathbf{W}$ is sparse, we use a vector with the smallest Hamming weight in the coset corresponding to $\mathbf{V}$ as the estimate of $\mathbf{W}$. Second, suppose that $\mathbf{\Phi W}$ is sparse for a known nonsingular $n \times n$ matrix $\mathbf{\Phi}$. Since $\mathbf{V} = \mathbf{AW} = \mathbf{A\Phi^{-1}\Phi W}$, we can treat $\mathbf{V}$ as a syndrome for the linear block code defined by $\mathbf{A\Phi^{-1}}$. Thus, in this scenario, we first select a vector with the smallest Hamming weight in the coset of the code defined by $\mathbf{A\Phi^{-1}}$ corresponding to $\mathbf{V}$, and then produce an estimate of $\mathbf{W}$ by multiplying the selected vector with $\mathbf{\Phi^{-1}}$. In both scenarios, the key step is to select a vector with the smallest Hamming weight in the coset corresponding to the given syndrome. Thus, we assume $\mathbf{W}$ is sparse without loss of generality.

In coding theory terminology, a vector with the smallest Hamming weight among a coset is called a leader of the coset. Note that some coset leaders may not be unique, when more than one vector in the coset has the smallest Hamming weight. In this case, either the coset leader is selected among these vectors at random or a list of all potential leaders is the output.

We remark that this problem is closely related to but different from the syndrome decoding problem in classic coding theory. In our decoding, a vector or a list of vectors with the smallest Hamming weight in the coset corresponding to the given syndrome is the estimate of the data vector. In the syndrome decoding problem, a coset leader is often considered as an estimate of the error vector. However, the key step in both problems is to select a vector or a list of vectors with the smallest Hamming weight in the coset corresponding to the given syndrome.

Thus, we have the following sufficient condition for successful decoding:

Lemma 1. *The minimum Hamming distance of the linear block code defined by* $\mathbf{A}$*, denoted by* $d_H(\mathbf{A})$*, satisfies* $d_H(\mathbf{A}) \leq m + 1$*. When the Hamming weight of* $\mathbf{W}$*, denoted by* $w_H(\mathbf{W})$*, is less than half of the minimum Hamming distance of the linear block code defined by* $\mathbf{A}$*, that is* $w_H(\mathbf{W}) < \frac{d_H(\mathbf{A})}{2}$*,* $\mathbf{W}$ *can be recovered by syndrome decoding.*

Proof: The first part is due to the Singleton bound on the minimum Hamming distance of linear block codes. The second part holds because it is well known that a coset leader with Hamming weight less than $\frac{d_H(\mathbf{A})}{2}$ is unique. ■

When $\mathbf{W}$ is not a unique coset leader, there are two possibilities. First, when the Hamming weight of $\mathbf{W}$ is minimal in its coset, either $\mathbf{W}$ has a probability to be selected when coset leaders are chosen at random or $\mathbf{W}$ is one of the possible

vectors produced by the decoder, depending on whether the decoder needs to generate only one vector or a list of vectors. Second, when the Hamming weight of $\mathbf{W}$ is not minimal, a wrong vector will be produced by the Hamming norm decoder.

E. Decoding Strategies

Possible outcomes of the full rank decoder are failure or success. In contrast, the proposed Hamming norm decoders may produce wrong decisions. Analogous to classical error control coding, the preference between decoding failures and decoding errors varies from one application to another. For instance, for applications with stringent delay constraints, partially correct data packets may be more desirable than decoding failures. For other applications such as cloud storage, data integrity may be a top priority than delays, especially when packet retransmission is possible. Hence, it is necessary to consider a wide range of decoding strategies so as to offer different tradeoffs between delay/throughput and accuracy.

Two extreme strategies are natural and straightforward. One extreme, called the error-free (EF) decoder, is similar to the full rank decoder in the sense that it decodes only if decoding success is guaranteed by Lemma 1. The other extreme, referred to as the best-effort (BE) decoder, always tries to decode with available received packets. The error-free and best-effort decoders represent the most conservative and the most aggressive strategies, respectively.

We also devise a family of decoding strategies that fills the gap between these two extremes based on one observation about error control codes. For an (n, k) perfect code over GF(2), we have $\sum_{i=0}^{t} \binom{n}{i} = 2^{n-k}$, where $t = \left\lfloor \frac{d_H(\mathbf{A})-1}{2} \right\rfloor$. In other words, all coset leaders are unique and have Hamming weight up to t. However, since most codes are not perfect and some allowance needs to be made. Hence, we devise a greedy-l decoding strategy: decodes only if $\sum_{i=0}^{cw-l} \binom{n}{i} = 2^{n-k}$, where cw is the maximal possible Hamming weight of $\mathbf{W}$. The parameter l represents how aggressive the decoder is: for the same code defined by $\mathbf{A}$, the greater l is, the more aggressive the decoder. In fact, one can use different l values to approach the two extremes, the best-effort and error-free strategies.

F. Linear Programming Decoders

Since both the computational complexity and the memory requirement of the Hamming norm decoders grow exponentially with the size of $\mathbf{A}$, we also adopt a linear programming (LP) approach. Since $\mathbf{A}$ is not necessarily sparse, we formulate the problem based on that for binary linear block code with high-density polytopes in [17].

Let $f_0, f_1, \ldots, f_{n-1}$ be the variables representing the code bits of $\mathbf{W}$, and $\mathbf{V} = (v_0, v_1, \ldots, v_{m-1})^T$ be the syndrome received. For each check node $j \in \mathcal{J}$, let $T_j^E = \{0, 2, 4, \ldots, 2\lfloor |N(j)|/2 \rfloor\}$ for $v_j = 0$, and $T_j^O = \{1, 3, 5, \ldots, 2\lfloor (|N(j)| - 1)/2 \rfloor + 1\}$ for $v_j = 1$. Then the linear programming formulation for the syndrome decoding is to minimize $\sum_{i=0}^{n-1} f_i$ subject to the linear constraints in [17, (14)–(19)] except that $T_j = T_j^E$ if $v_j = 0$, and $T_j = T_j^O$ if $v_j = 1$. In contrast, $T_j = T_j^E$ in [17, (14)–(19)]. In addition, we add a linear constraint to narrow down the optimal solutions:

$$\sum_{i=0}^{n-1} f_i \leq cw.$$

Linear programming may produce non-integral results, in which case two approaches are considered. The first is to round off the real values into integers, which are compared with the original data to compute decoding error or success rate, and we call this approach LP I. The other, referred to as LP II, is to declare decoding failure. Both LP I and LP II are applicable to all greedy as well as the BE strategies.

III. Simulation Results

To illustrate the advantages of the proposed rank deficient decoders, we present some numerical simulation results with the following settings. Network coding is carried out over GF(2). We assume each session (or generation) consists of $n = 8$ packets of length $N = 8$ bits such that the transmission matrix has a constant column weight of $cw = 2$. The matrix $\mathbf{A}$ is generated randomly, with each element being 0 or 1 with equal probability. For each iteration, as the number of (linearly independent) received packets m increases from 1 to 15, the proposed decoders as well as the full rank decoder are used to decode, and their decoding success, failure, or error on both packet and bit levels are recorded. For each decoder, its packet- and bit-level success, failure, or error rate is obtained by averaging over 100,000 generations.

We note that such small values for n and N are chosen so that the complexities of the Hamming norm decoders are manageable. We also note that in this setting, the data sparsity is manifested as an upper bound on the column weights in the transmitted data packets. We also have simulation results assuming other deterministic or stochastic manifestations of data sparsity, such as an upper bound on the row weights in the transmitted data packets, or the bits in the transmitted data packets being i.i.d. binary Bernoulli random variables with probability p ($p < 1/2$). Due to limited space, the simulation results for these other manifestations are omitted, but the proposed rank deficient decoders demonstrate similar advantages regardless of the manifestation of data sparsity.

In Fig. 1 and Fig. 2, respectively, the packet- and bit-wise fraction of decoding success, failures, and errors of Hamming norm decoders are represented by green, yellow, and red bars. Similarly, Fig. 3 and Fig. 4, respectively, compare the packet- and bit-wise fraction of decoding success, failures, and errors of linear programming decoders. In all figures, for each value of m, the six bars represent, from left to right, the full rank, error-free, greedy-(-1), greedy-0, greedy-1, and best-effort strategies, respectively. In order to measure and compare the throughput and delay of linear network coding with these decoders, the average minimum numbers of packets required to achieve a packet success rate (PSR) of 1 or a bit success rate (BSR) of 0.95 are compared in Table I.

The simulation results confirm our claims about rank deficient decoders. The full rank decoder can recover data packets only when $m \geq n = 8$ and recovers no packet when $m < 8$. In contrast, our rank deficient decoders recover a greater fraction

TABLE I
AVERAGE NUMBER OF PACKETS REQUIRED TO ACHIEVE 100% PSR AND 95% BSR

Strategy	FR	EF	greedy-(-1)			greedy-0			greedy-1			BE		
			HN	LP I	LP II	HN	LP I	LP II	HN	LP I	LP II	HN	LP I	LP II
100% PSR	9.60	8.84	8.12	8.44	8.45	7.57	8.19	8.22	7.44	8.17	8.21	7.44	8.17	8.21
95% BSR	9.60	8.84	8.05	8.15	8.18	7.40	7.66	7.74	7.17	7.58	7.67	7.17	7.58	7.67

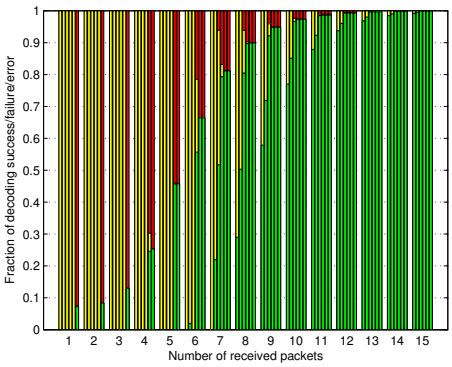

Fig. 1. Packet-level performance of different strategies using the HN decoders

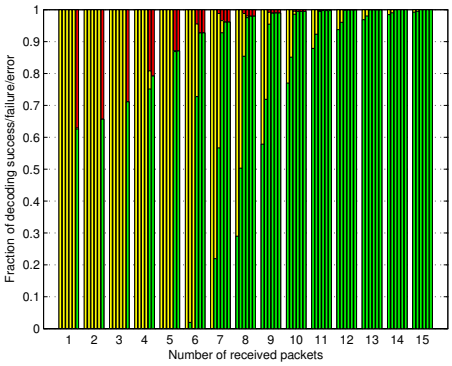

Fig. 2. Bit-level performance of different strategies using the HN decoders

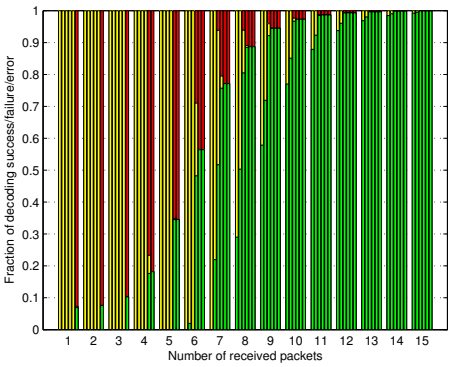

Fig. 3. Packet-level performance of different strategies using LP I

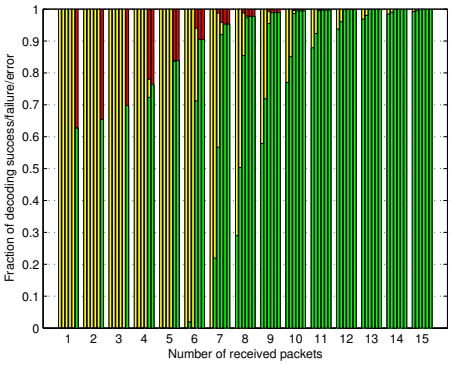

Fig. 4. Bit-level performance of different strategies using LP I

of data packets when $m \geq n = 8$, and recover a substantial fraction of data packets even when $m < 8$.

The proposed decoders provide a wide range of tradeoffs between delay/throughput and decoding errors. Just like the full rank decoder, the error-free strategy does not produce any decoding errors. Nevertheless, it outperforms the full rank decoder significantly for $m < n$. For instance, when $m = 7$, the error-free strategy recovers over 20% of the packets, while the full rank decoder cannot recover anything. At the other extreme, the performance of the best-effort strategy improves when m grows. For instance, when $m = 1$, it recovers around 10% of the packets and 70% of the bits. However, when $m = 7$, it recovers over 80% of the packets and 96% of the bits in the session. The greedy-l strategies fill the gap between the two extremes.

There is a difference between packet- and bit-level performances. For the full rank and error-free strategies, their packet- and bit-level performances are the same, because their decoding strategies depend on $\mathbf{A}$ only, and are the same for all l's in Eq. (1). For the other four strategies, since their decoding strategies depend on $\mathbf{A}$ as well as $\mathbf{V}_l$, their packet- and bit-level performances are different. Of course, their bit-level decoding success fractions are better than their respective packet-level decoding success fractions. This is because a packet-level decoding success requires bit-level decoding successes for all l's in Eq. (1).

Compared with the full rank decoder, the average minimum numbers of packets required for success decoding for the error-free and best-effort strategies are approximately 10% and 20% smaller, respectively. Assuming that the received packets arrive in a uniform interval, this means that throughputs achieved by the error-free and best-effort strategies are roughly 10% and 20%, respectively, higher than the full rank decoder. The actual advantage may be more significant, because it takes longer to receive a linearly independent packet when more received packets already exist.

As expected, the linear programming decoders perform slightly worse than the Hamming norm decoders. However, the performance difference is negligible when the number of received packets is large.

REFERENCES

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

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

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

[4] C. Gkantsidis and P. R. Rodriguez, "Network coding for large scale content distribution," *Proceedings of 2005 IEEE Infocom*, vol. 4, pp. 2235–2245, March 2005.

[5] S. Deb, M. Médard, and C. Choute, "Algebraic gossip: a network coding approach to optimal multiple rumor mongering," *IEEE Trans. Info. Theory*, vol. 52, no. 6, pp. 2486–2507, June 2006.

[6] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Médard, "The importantance of being opportunistic: practical network coding for wireless environments," *Proc. Allerton Conf. Commun., Control and Computing*, September 2005.

[7] J.-S. Park, M. Gerla, D. S. Lun, Y. Yi, and M. Médard, "Codecast: a network-coding-based ad hoc multicast protocol," *IEEE Wireless Commun. Magazine*, vol. 13, no. 5, pp. 76–81, October 2006.

[8] Z. Liu, C. Wu, B. Li, and S. Zhao, "Uusee: Large-scale operational on-demand streaming with random network coding," *Proc. IEEE Int. Symp. on Computer Communications*, pp. 1–9, March 2010.

[9] H. Chen, "Distributed file sharing: Network coding meets compressed sensing," *Proceedings of First International Conference on Communications and Networking in China (ChinaCom'06)*, pp. 1–5, October 2006.

[10] S. Katti, S. Shintre, S. Jaggi, D. Katabi, and M. Médard, "Real network coding," *Proceedings of Forty-Fifth Annual Allerton Conference on Communication, Control, and Computing*, pp. 389–395, September 2007.

[11] N. Nguyen, D. Jones, and S. Krishnamurthy, "Netcompress: Coupling network coding and compressed sensing for efficient data communication in wireless sensor networks," *Proceedings of 2010 IEEE Workshop on Signal Processing Systems (SiPS 2010)*, pp. 356–361, October 2010.

[12] S. Shintre, S. Katti, S. Jaggi, B. K. Dey, D. Katabi, and M. Médard, "'Real' and 'complex' network codes: Promises and challenges," *Fourth Workshop on Network Coding, Theory and Applications (NetCod 2008)*, pp. 1–6, January 2008.

[13] N. Cai and R. W. Yeung, "Network coding and error correction," in *Proc. IEEE Information Theory Workshop*, Bangalore, India, October 2002, pp. 20–25.

[14] Z. Zhang, "Linear network error correction codes in packet networks," *IEEE Trans. Info. Theory*, vol. 54, no. 1, pp. 209–218, January 2008.

[15] R. Kötter and F. R. Kschischang, "Coding for errors and erasures in random network coding," *IEEE Trans. Info. Theory*, vol. 54, no. 8, pp. 3579–3591, August 2008.

[16] H. Mahdavifar and A. Vardy, "Algebraic list-decoding on the operator channel," in *Proc. IEEE Int. Symp. Info. Theory*, Austin, USA, June 2010, pp. 1193–1197.

[17] J. Feldman, M. J. Wainwright, and D. R. Karger, "Using linear programming to decode binary linear codes," *IEEE Trans. Info. Theory*, vol. 51, no. 3, pp. 954–972, March 2005.