Compressed Network Coding Vectors

Mahdi Jafari, Lorenzo Keller, Christina Fragouli and Katerina Argyraki
EPFL, Lausanne, Switzerland
Email: mahdi.jafarisiavoshani@epfl.ch, lorenzo.keller@epfl.ch, christina.fragouli@epfl.ch, katerina.argyraki@epfl.ch

***Abstract*— In networks that employ network coding, two main approaches have been proposed in the literature to allow the receivers to recover the source information: (i) use of coding vectors, that keep track of the linear combinations the received packets contain, and (ii) subspace coding, that dispenses of the need to know the linear combinations, since information is conveyed from the choice of subspaces alone. Both these approaches impose the strong requirement that all source packets get potentially combined. We here present a third approach that relaxes this assumption, and is thus not a special case from either of the previous two. This relaxation allows to employ compressed coding vectors to efficiently convey the coding coefficients, without altering the operation of intermediate network nodes. We develop optimal designs for such vectors.**

I. Introduction

There has been a growing consensus in the research community that network coding is a promising technique to be applied in networking applications, such as wireless networks and content distribution networks. Network coding has intermediate network nodes perform combinations of the source data. Practical networks being subject to random delays, synchronization errors, and even packet erasures, nodes failures, and topology changes, it is not viable to assume that the linear combinations performed at the intermediate nodes are deterministically known at the receivers.

Two approaches have been proposed in the literature to address this. The first has a coding vector appended to each packet [2]. This vector keeps track of the linear combination of the source packets the coded packet contains. The receivers use this information to solve a system of linear equations and recover the original data. The second approach uses subspace coding [4]. The information is conveyed by a subspace that the source selects; the receivers to decode simply need to decide which was the sent subspace. In this case the receiver needs no information about the linear combinations that the network nodes perform to decode. Both these approaches divide the source packets into generations and allow combining only among packets in the same generation. As far as we know, these are the only two approaches currently proposed.

The first approach comes at the cost of the coding vectors overhead. This overhead would be acceptable for large packets, however, in wireless applications, where packets are much shorter, it can very fast become prohibitive. Even in wired networks, the tradeoff between a larger generation, which employs longer coding vectors, and a smaller generation, which may not allow mixing of packets and reduce the network coding benefits, is a subject of research in the community.

From an information theoretic point of view, the second approach, subspace coding, results in higher information rates for short packet length, but as the packet length increases, achieves the same information rate as having the coding vectors overhead [3]. Moreover, it is very challenging to design subspace codes for multisource network coding, where the information sources that insert data in the network are not co-located, as is the case for several applications. We discuss such examples in Section II.

In this paper we present a third approach that is not a special case of the previous two. Our approach employs shortened or compressed coding vectors to efficiently convey the coding coefficients. The observation our approach leverages is that, the classic design of coding vectors allows potentially *all* source packets to get combined together; however, for some networks, this is too strong a requirement (see Section II for examples), and results in too low an information rate. In our approach we thus propose to employ coding vectors that allow at most m source packets to get combined. This naturally occurs in some applications, where for example only source packets originating from neighboring nodes get combined. We can also artificially restrict the number of source packets that get combined, by appending to each coded packet a few bits to count the number of source packets it contains. Note that, the receiver will eventually still need to solve a set of n linear equations to retrieve the source data; our approach only shortens the coding vectors that convey these linear combinations.

Our design problem can now be stated as follows. Given a generation that contains n source packets, each receiver is going to observe packets that contain linear combinations of *at most* m source packets. We want to design coding vectors that allow us, by receiving each combined packet, to determine which linear combination of the source packets it contains. The classical coding vectors design would utilize coding vectors of length n. In this paper we explore what, under our assumptions, is the smallest length r of coding vectors we need to employ, and how can we select them. A key point of our design is that we require the intermediate node operation to be oblivious to the coding vectors employed, and in particular, to not perform compression operations.

For m much smaller than n, our approach can also be viewed as compressing the classical coding vectors, and our problem can be cast in a compressed sensing framework.

This work was supported by the Swiss National Science Foundation under award PP002-110483, by the EU projects NetReFound (FP6-IST-034413) and N-CRAVE (FP7 ICT-2007-215252) and by the Hasler Foundation project number 2072.

Moreover, in this case, solving the set of n linear equations at the receiver becomes more efficient, since we can take advantage of the low density of the linear combinations to decode with belief propagation techniques.

The paper is organized as follows. Section II introduces our notation and reviews the existing approaches. Section III presents our approach, and Section IV concludes the paper.

II. Approaches in the literature and problem statement

Consider a dissemination protocol where the nodes in the network perform linear network coding, *i.e.*, linearly combine their incoming packets. One or multiple sources, not necessarily collocated, produce independent information packets, that we will call source packets. The source packets get divided into sets called generations. Source packets belonging in the same generation are allowed to get combined together, as they traverse the network. Assume that each generation contains n source packets $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. Each such packet consists of ℓ symbols over some finite field $\mathbb{F}_q$.

The classical coding vector approach appends to each source packet $\boldsymbol{x}_i$ a coding vector $\boldsymbol{p}_i^C$. Initially, the sources employ $\boldsymbol{p}_i^C = \boldsymbol{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0) \in \mathbb{F}_q^n$, *i.e.*, $\boldsymbol{e}_i$ has zeros everywhere and 1 is at the ith position. Thus the packets sent by the sources are of the form

$$[\boldsymbol{e}_i \mid \boldsymbol{x}_i], \tag{1}$$

where we assumed without loss of generality that the coding vector is placed at the beginning of the packet. Intermediate network nodes perform linear combinations of their received packets. In general a packet propagating in the network will have the form

$$\boldsymbol{p} \triangleq [\boldsymbol{p}^C \mid \boldsymbol{p}^I], \tag{2}$$

where $\boldsymbol{p}^I \in \mathbb{F}_q^\ell$ is a linear combination of source packets (we call this sometimes information vector), and $\boldsymbol{p}^C \in \mathbb{F}_q^n$ is the coding vector that contains the linear coefficients for the combined source packets.

Each receiver that receives n packets with linearly independent coding vectors can recover the original source information. To do so, the receiver solves the linear equations

$$\begin{bmatrix} \boldsymbol{p}_1^I \\ \boldsymbol{p}_2^I \\ \vdots \\ \boldsymbol{p}_n^I \end{bmatrix} = \underbrace{\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}}_{\boldsymbol{A} \in \mathbb{F}_q^{n \times n}} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_n \end{bmatrix}, \tag{3}$$

where the ith row of matrix $\boldsymbol{A}$ is the coding vector corresponding to received packet $\boldsymbol{p}_i$. Since the receiver collects n linearly independent coding vectors, the matrix $\boldsymbol{A}$ is full rank, and thus the original packets can be recovered.

This approach comes at the overhead of the coding vectors, that can fast become impractical, as we following illustrate.

Example 1: Consider a sensor network consisting of 100 nodes, each sending a message to a sink. To implement network coding using coding vectors over a field of size $q = 2^4$ we would need to use 50 Bytes of each packet simply for the coding vectors. In the TinyOs operating system [18], which is perhaps the most popular for sensor nodes, a typical frame length allows approx. 30 bytes for data transmissions. Thus clearly this is not a viable approach. ■

TABLE I

Coding for two sources.

$\mathcal{C}_2/\mathcal{C}_1$	π_1	π_2	π_3
π_4	$\pi_1 + \pi_4$	$\pi_2 + \pi_4$	$\pi_3 + \pi_4$
π_5	$\pi_1 + \pi_5$	$\pi_2 + \pi_5$	$\pi_3 + \pi_5$
π_6	$\pi_1 + \pi_6$	$\pi_2 + \pi_6$	$\pi_3 + \pi_6$

Subspace coding dispenses of the need to convey coding vectors. In this scheme, neither the receiver nor the sources know the mixing matrices $\boldsymbol{A}$ in (3), *i.e.*, the specific set of linear operations. Sources can only communicate information using subspaces which are unaffected by the linear operations performed on them. Hence, each source uses a subspace codebook, *i.e.*, maps each message to a set of vectors that span a different subspace. This approach is optimal in terms of achievable information rates when the length of the packets is small but as the length increases it results in the same information rate as the coding vectors approach [3]. Moreover, as the following example illustrates, code design is not trivial when multiple sources insert data in the network.

Example 2: We here argue that designing subspace codes for the case where the sources are not collocated is challenging. Consider the case where n sources employ codebooks $\mathcal{M}_i$ consisting of subspaces of the vector space $\mathbb{F}_q^\ell$, *i.e.*,

$$\mathcal{C}_i = \{\pi_j^{(i)} : \pi_j^{(i)} \subseteq \mathbb{F}_q^\ell,\ 1 \leq j \leq |\mathcal{M}_i|\}, \quad i = 1, \ldots, n.$$

To transmit information to the sink, source i maps a measured value to one such subspace π and inserts in the network $\dim(\pi)$ vectors that span π. In relaying information towards the sink, the sensor linearly combines all packets it has received (including that generated by itself) and transmits the combined packet to the next relays towards to the sink. As a result, the sink will observe vectors from the union of subspaces inserted by all the sources. In particular, if source i inserts the subspace π_i, the sink will observe vectors from the subspace $\pi_1 + \pi_2 + \cdots + \pi_n$. Using the knowledge of the codebooks $\{\mathcal{C}_i\}$, it needs to decode the source data.

To be able to correctly decode at the receiver, we need to ensure that every combination of source data results in a *distinct* union subspace. We call this the identifiability property. Assume for simplicity we have two source nodes, S_1 using the codebook $\mathcal{C}_1 = \{\pi_1,\ \pi_2,\ \pi_3\}$, while S_2 the codebook $\mathcal{C}_2 = \{\pi_4,\ \pi_5,\ \pi_6\}$. Table I summarizes all outcomes. For this code to be identifiable, we want all (or some) entries in Table I to correspond to distinct subspaces. For example, $\pi_1 + \pi_4$ should be a distinct subspace from $\pi_2 + \pi_5$.

This problem is hard to solve even for the case of two sources, and a very small codebook (in our example each node transmits only 3 values). Designing such a code for multiple sources is clearly a challenging task. ■

In both the previous approaches, a common underlying assumption is that, all source packets may get combined in the network. Given that clearly this can be too strong a requirement for many practical networks, we here relax it, and require that each coded packets contains a linear combination of at most m out the n source packets. This allows us to use coding vectors whose length grows sub-linearly with n, resulting in a more efficient network communication. In the following we in turn discuss, how can we design such coding vectors, and how do we utilize them in decoding, *i.e.*, how we can retrieve the linear coefficients of the combined source packets. We also discuss what is the smallest required length, and what are the benefits we can expect to get.

III. Compressing the coding vectors

A. Code design

Consider a network performing linear network coding, where each coded packet contains the linear combination of at most m source packets. For m much smaller than n, the classical coding vectors become sparse. We can thus compress them, by replacing them with shorter vectors, that still allow the receivers to extract the original coding vectors and decode the sources messages. Our construction utilizes properties of algebraic error correcting codes, and proceeds as follows.

Select a linear code $\mathcal{C} = [n, k, d]_q$ where $d = \min(2m + 1, n+1)$ with k as large as possible. Consider the $r \times n$ parity check matrix $\mathbf{H}_\mathcal{C}$ where $r \triangleq n - k$. As coding vector, assign to source packet $\boldsymbol{x}_i$ the ith column of the matrix $\mathbf{H}_\mathcal{C}$, which we will denote as $\boldsymbol{h}_i$. That is,

$$\boldsymbol{h}_i = \boldsymbol{e}_i \cdot \mathbf{H}_\mathcal{C}^T. \tag{4}$$

We call these vectors *compressed coding vectors*. Thus the sources insert to the network the packets

$$[\boldsymbol{h}_i \mid \boldsymbol{x}_i]. \tag{5}$$

Intermediate nodes linearly combine their received packets. The coded packets propagating in the network will now have the form

$$\boldsymbol{p} \triangleq [\hat{\boldsymbol{p}}^C \mid \boldsymbol{p}^I], \tag{6}$$

where $\hat{\boldsymbol{p}}^C \in \mathbb{F}_q^r$ denotes the compressed coding vector appended to packet $\boldsymbol{p}$. This is related to the classical coding vector $\boldsymbol{p}^C$ that describes the linear transform from the source packets as

$$\hat{\boldsymbol{p}}^C = \boldsymbol{p}^C \cdot \mathbf{H}_\mathcal{C}^T. \tag{7}$$

If m packets are allowed to be combined, with m much smaller than the length n of the coding vector $\boldsymbol{p}^C$, this can be viewed as compressing the sparse vector $\boldsymbol{p}^C$, and hence the compressed coding vector terminology.

The reason this construction enables receivers to decode follows from a well known property the columns of matrix $\mathbf{H}_\mathcal{C}$ satisfy. If a code $\mathcal{C}$ has minimum distance d, then any set of $d-1$ columns of the matrix $\mathbf{H}_\mathcal{C}$ are linearly independent [5]. Moreover, given that at most m source packets get combined, $\text{wt}(\boldsymbol{p}^C) \leq m$ where $\text{wt}(\cdot)$ denotes the Hamming weight of a vector, the number of non-zero elements. The following lemma states that we will be able to recover the original coding vectors from the compressed ones.

Lemma 1: There is an injective map between $\boldsymbol{p}^C$, $\text{wt}(\boldsymbol{p}^C) \leq m$, and $\hat{\boldsymbol{p}}^C$ related by (7).

Proof: For two $\boldsymbol{p}_1^C \neq \boldsymbol{p}_2^C$ where $\text{wt}(\boldsymbol{p}_1^C) \leq m$ and $\text{wt}(\boldsymbol{p}_2^C) \leq m$ we have $\text{wt}(\boldsymbol{p}_2^C - \boldsymbol{p}_1^C) \leq \min(2m, n)$. But the minimum distance of $\mathbf{H}_\mathcal{C}$ is $\min(2m + 1, n + 1)$ so $(\boldsymbol{p}_2^C - \boldsymbol{p}_1^C) \cdot \mathbf{H}_\mathcal{C}^T \neq 0$ which leads to $\hat{\boldsymbol{p}}_1^C \neq \hat{\boldsymbol{p}}_2^C$. ∎

Example 3: Suppose the number of packets in every generation is $n = 15$ and each packet in the network contains linear combinations of at most $m = 2$ packets which leads to $d = 2m+1 = 5$. Let also $q = 2^4$. The code $\mathcal{C}$ can be chosen to be the Reed-Solomon code with parameters $\mathcal{C} = [15, 11, 5]_{2^4}$. The parity check matrix of $\mathcal{C}$ can be written as follows

$$\mathbf{H}_\mathcal{C} = \begin{bmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{15-1} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(15-1)} \\ 1 & \alpha^3 & \alpha^6 & \cdots & \alpha^{3(15-1)} \\ 1 & \alpha^4 & \alpha^8 & \cdots & \alpha^{4(15-1)} \end{bmatrix},$$

where α is a primitive element of $\mathbb{F}_{2^4}$. Each column of $\mathbf{H}_\mathcal{C}$ can be assigned to one of $n = 15$ source packets.

B. Decoding

Upon receiving a packet $\boldsymbol{p}$ with compressed coding vector $\hat{\boldsymbol{p}}^C$, the receiver needs to recover the original coding vector to construct the system of linear equation in (3).

In our construction, the problem of finding the original coding vector $\boldsymbol{p}^C$ from the compressed coding vector $\hat{\boldsymbol{p}}^C$ reduces to a decoding problem. In the coding theory terminology, we need to find the error vector having access only to the syndrome of a received vector. More formally, we may write

$$\begin{aligned} \text{find} \quad & \boldsymbol{p}^C \\ \text{subject to} \quad & \text{wt}(\boldsymbol{p}^C) \leq m, \\ & \boldsymbol{p}^C \cdot \mathbf{H}_\mathcal{C}^T = \hat{\boldsymbol{p}}^C. \end{aligned} \tag{8}$$

This problem is in general NP-complete [6]. However, coding theory identifies instances that accept efficient encoding and decoding algorithms, and we leverage these constructions.

Note that, it is sufficient to find what are the non-zero positions of $\boldsymbol{p}^C$. If we know the non-zero positions, using the knowledge of the matrix $\mathbf{H}_\mathcal{C}$, we can uniquely recover the linear coefficients in the original coding vectors. The following lemma from coding theory formalizes this observation [8].

Lemma 2: Let $\mathcal{C}$ be a linear code in $\mathbb{F}_q^n$ with parity check matrix $\mathbf{H}_\mathcal{C}$. Assume a codeword $\boldsymbol{x}$ is sent and a word $\boldsymbol{y}$ is received, with error vector $\boldsymbol{e}$, where $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{e}$. Suppose we know a set J with at most $d(\mathcal{C}) - 1$ elements that contains the set of error positions; *i.e.*, non-zero elements of $\boldsymbol{e}$. Then the error vector $\boldsymbol{e}$ is the unique solution of the following linear equations: $\hat{\boldsymbol{e}}\mathbf{H}^T = \boldsymbol{y}\mathbf{H}^T$ and $\hat{e}_j = 0$ for all $j \notin J$.

The above lemma shows that we can reduce the problem of recovering the original coding vector to the problem of finding the non-zero positions of the original coding vector.

One approach to achieve this is through exhaustive search. For small values for m and n this in a fast computer can be

TABLE II

TIME FOR EXHAUSTIVE SEARCH IN SECONDS. EXPERIMENTS ARE RUN ON A SINGLE CORE OF AN INTEL CENTRINO DUO2, AT 3 GHZ.

n/m	2	3	4
15	0.00018	0.0020	0.017
31	0.00097	0.024	0.48
63	0.0047	0.24	10.4

feasible, as Table II illustrates. However, there are $\binom{n}{m}$ possible m-sets of non-zero positions to consider. This number grows exponentially in n when $\frac{m}{n}$ converges to a non-zero number.

A more practical approach is to use some known algebraic codes for $\mathcal{C}$ like BCH code, Reed-Solomon code [14], Goppa code [13], algebraic geometry codes [15], etc., to recover the original coding vectors efficiently. For all of the codes mentioned above there exists a version of the Berlekamp-Massey algorithm [9], [10] which allows the receivers to find the location of non-zero elements of original coding vectors as well as their values, using only the syndrome.

The Berlekamp-Massey algorithm consists of three stages that can be briefly summarized as follows. The first stage is the calculation of syndrome which in our approach we have it for free, since the received compressed coding vectors are equivalent to syndromes of original coding vectors. The second stage is to find the error locator polynomial which is defined as following

$$\lambda(z) \triangleq \prod_{r=1}^{\tau}(1-\alpha^{i_r}z) = \sum_{r=0}^{\tau}\lambda_i z^r,$$

where $i_1, \ldots, i_\tau$ are the non-zero elements of $\boldsymbol{p}^C$, $\tau \leq t$, and α is a primitive nth root of unity. Finally, receivers find the roots of $\lambda(z)$ to find the location of non-zero components of $\boldsymbol{p}^C$ and using Lemma 2 can retrieve the original coding vectors.

C. Benefits

Using the compressed coding vectors method, the length of coding vectors reduces from n to $r = n - k$. The following lemma shows the optimality of our construction in terms of r.

Lemma 3: The proposed construction leads to the shortest length possible for compressed coding vectors.

Proof: The problem of finding the shortest representation for a sparse vector of length n and sparsity at most m over $\mathbb{F}_q$ is equivalent to the problem of designing the highest rate code with length n and minimum distance $2m+1$ over $\mathbb{F}_q$. Indeed, if one could find a smaller representation for the compresses coding vectors that still lets recovering the original coding vectors, *i.e.*, problem 8 is solvable, this implies that there exist a higher rate code with the specified parameters. ■

We now examine what is the required size r for different cases. From the Singleton bound for code $\mathcal{C}$ we have

$$k \leq \ n - d + 1 = \ n - \min(2m, n),$$

so for $\frac{n}{2} \leq m \leq n$ we have $k = 0$ which implies that we can select w.l.o.g. the full rank $n \times n$ parity matrix $\mathbf{H}_\mathcal{C}$ to be the identity matrix. In this case, we recover the usual network coding with coding vectors $\{\boldsymbol{e}_i\}$ appended to the sources packets, and there is no benefit from our approach.

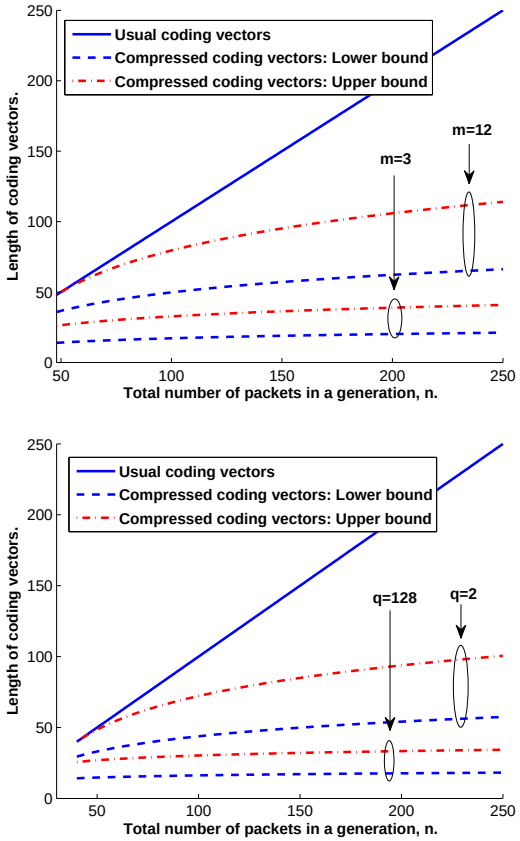

Fig. 1. Bounds on the length of the compressed coding vectors, r. Upper figure: r as a function of the number of packets in a generation n, when $m = 2$ and $m = 20$ sources get combined, $q = 2$. Lower figure: r as a function of n when $m = 10$ sources get combined, for two values of field size $q = 2$ and $q = 128$.

From the Gilbert-Varshamov bound [7] we have an upper bound for the length of compressed coding vectors r that for the case $m < \frac{n}{4}$ can be simplified to

$$r \leq nH_q\left(\frac{d-1}{n}\right) = nH_q\left(\frac{2m}{n}\right), \tag{9}$$

where $H_q(\delta) = \delta\log_q(q-1) - \delta\log_q\delta - (1-\delta)\log_q(1-\delta)$ is the q-ary entropy function. Also, the Sphere packing bound leads to a lower bound on the length of compressed coding vectors where for $m < \frac{n}{2}$ we can simplify it to obtain

$$\begin{aligned} r &\geq nH_q\left(\frac{d-1}{2n}\right) - \frac{1}{2}\log_q\left(4(d-1)\left(1-\frac{d-1}{2n}\right)\right) \\ &= nH_q\left(\frac{m}{n}\right) - \frac{1}{2}\log_q\left(8m\left(1-\frac{m}{n}\right)\right). \end{aligned} \tag{10}$$

From (9) and (10), for fixed values of m, and as the number of source packets grows, we have

$$m\log_q n + \mathcal{O}(1) \leq r \leq 2m\log_q n + \mathcal{O}(1),$$

So using the proposed method, we can reduce the growth of coding vectors from $\mathcal{O}(n)$ to $\mathcal{O}(m \log n)$.

Example 4: Using a table of the best codes known (from [5] and [12]), we can see for example that, there exist binary linear codes of length $n = 127$ with redundancy $r = 35$ and minimum distance $d = 2m + 1 = 11$, which is in fact a shortened version of $[128, 93, 11]_2$ Goppa code [13]. Thus in a network with 127 source packets in each generation if at most $m = 5$ source vectors get combined, we need to use coding vectors of length $r = 35$ instead of $n = 127$.

In the previous example, it is assumed that the network nodes perform binary network coding; *i.e.*, nodes only XOR the packets. However, if the field size is increased, a shorter compressed coding vectors can be used as it is shown in the Example 5. In fact the code used in Example 4 is not a MDS[1] code while the Reed-Solomon code in Example 5 is MDS.

Example 5: It is known that the Reed-Solomon code [14] is a linear code $[n, k, d]_q$ where $n = q - 1$ and $k + d - 1 = n$. To compare the length of compressed coding vectors resulting from the Reed-Solomon code as in Example 4, consider a field of size $q = 2^7$, which leads to $n = 127$. Also set $d = 2m + 1 = 11$ for the redundancy; the length of compressed coding vectors equals $r = n - k = d - 1 = 10$ where the ratio of the compressed coding vector length to the original coding vector length is much lower than that of Example 4. Compared to the case of classical coding vectors, the coding vector headers decrease from 112 Bytes to only 9 Bytes.

D. Effect on Rate

A natural question to ask is, if we restrict the number of combined packets, how does this affect the observed multicasting rate. This clearly depends not only on the value of m, but also on the network topology and on the subsets of m packets that get combined. For example, for some networks, no coding is required to achieve the min-cut rate for all receivers. It is also easy to come up with specifically constructed examples, where we cannot achieve the min-cut rate unless all source packets get combined.

However, as we argued in Section II, in many situations, such as the case of multisource wireless networks, it is not practical to allow all possible linear combinations to occur. Additionally, in preliminary experiments we are performing, we see that the number of actually combined packets depends on the distance from the receiver and the number of sources in the vicinity, and can be much smaller than the total number of sources in the network, as sources that are topologically separated may very rarely have their packets combined.

One possible way to abstract this problem is the following. Consider again the linear equations that the receiver needs to solve in (3) and assume that the nonzero elements in the $n \times n$ matrix $\boldsymbol{A}$ are chosen uniformly at random with probability m/n. For each nonzero position, a uniformly at random nonzero coefficient from the field $\mathbb{F}_q$ is then selected. This will result in a sparse matrix $\boldsymbol{A}$, where each row will have on the average m nonzero elements from $\mathbb{F}_q$ per row (coding vector). From [16], we have the following lemma.

Lemma 4: For every $c \geq 0$ there exist a constant a_c such that for the random matrix $\boldsymbol{A} \in \mathbb{F}_q^{n \times n}$, $n > e^c$, with $m = \log n - c$ we have

$$\Pr[\text{Rank}(\boldsymbol{A}) < n] \leq \frac{a_c}{q}.$$

Proof: See Corollary 2.4 [16]. ∎

Work in [17] extended the above lemma and showed that for $m > \log n$ the probability that the matrix $\boldsymbol{A}$ is not full rank approaches zero polynomially fast with n. The above argument shows that m should be at least order $\mathcal{O}(\log n)$ to let receivers end up with full rank matrix $\boldsymbol{A}$ with high probability.

IV. Conclusions

In this paper we have presented a novel approach for practical network coding, that uses shortened coding vectors as compared to the classical approach. We showed that we can reduce the length of the coding vectors from n to $\mathcal{O}(m \log n)$ where n is the number of source packets injected in the network, if each coded packet contains the combination of at most m source packets.

[1] Maximum distance separable.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow", *IEEE Transactions on Information Theory*, Volume 46, Issue 4, Page(s) 1204–1216, 2000.

[2] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding", *Allerton Conference on Communication, Control, and Computing*, Monticello, IL, October 2003.

[3] M. Jafari, S. Mohajer, C. Fragouli and S. Diggavi, "Capacity of non-coherent network coding", *IEEE International Symposium on Information Theory*, Seoul, Korea, June 2009.

[4] R. Koetter and F. Kschischang, "Coding for errors and erasures in random network coding", *IEEE Transactions on Information Theory*, Volume 54, Issue 8, Page(s) 3579–3591, August 2008.

[5] F. J. MacWilliams and N. J. A. Sloane, "The theory of error-correcting codes", North-Holland Mathematical Library, 1977.

[6] E. R. Berlekamp, R. J. McEliece, and H. C. van Tilborg, "On the inherent intractability of certain coding problems", *IEEE Transactions on Information Theory*, Volume 24, Issue 3, May 1978.

[7] R. E. Blahut, "Algebraic codes for data transmission", Cambridge University Press, 2003.

[8] Edited by V. S. Pless and W. C. Huffman, "Handbook of coding theory", North-Holland Mathematical Library, 1998.

[9] E. R. Berlekamp, "Nonbinary BCH decoding", *IEEE Transaction of Information Theory*, Volume 14, Issue 2, 1968.

[10] J. L. Massey, "Shift-Register synthesis and BCH decoding", *IEEE Transaction on Information Theory*, Volume 15, Issue 1, 1969.

[11] N. Patterson, "Algebraic Decoding of Goppa Codes", *IEEE Transaction on Information Theory*, Volume 21, Issue 2, Page(s) 203–207, 1975.

[12] M. Grassl, "Bounds on the minimum distance of linear codes", *Online available at http://www.codetables.de*, Accessed on 2009-01-09.

[13] V. D. Goppa, "A new class of linear error-correcting codes", *Probl. Peredach. Inform*, Volume 6, Issue 3, Page(s) 24–30, 1970.

[14] I. S. Reed and G. Solomon, "Polynomial codes over certain finite field", *Journal of the Society for Industrial and Applied Mathematics (SIAM)*, Volume 8, Page(s) 300–304, 1960.

[15] V. D. Goppa, "Codes associated with divisors", *Probl. Peredachi Inform.*, Volume 13, Page(s) 33–39, 1977. Translation: *Probl. Inform. Transmission*, Volume 13, Page(s) 22–26, 1977.

[16] J. Blomer, R. Karp, and E. Welzl, "The Rank of Sparse Random Matrices over Finite Fields", *Random Structures Algorithms 10*, Page(s) 407–419.

[17] P. Pakzad, C. Fragouli, and A. Shokrollahi, "Coding Schemes for Line Networks", *IEEE International Symposium on Information Theory*, Adelaide, Australia, 2005.

[18] Tinyos: http://www.tinyos.net/.