Binary Error Correcting Network Codes

Qiwen Wang Sidharth Jaggi Shuo-Yen Robert Li
The Chinese University of Hong Kong

***Abstract*—We consider network coding for networks experiencing *worst-case bit-flip* errors, and argue that this is a reasonable model for highly dynamic wireless network transmissions. We demonstrate that in this setup prior network error-correcting schemes ([11], [10]) can be arbitrarily far from achieving the optimal network throughput. We propose a new metric for errors under this model. Using this metric, we prove a new *Hamming-type* upper bound on the network capacity. We also show a commensurate lower bound based on *GV-type* codes that can be used for error-correction. The codes used to attain the lower bound are non-coherent (do not require prior knowledge of network topology). The end-to-end nature of our design enables our codes to be overlaid on classical distributed random linear network codes [6]. Further, we free internal nodes from having to implement potentially computationally intensive link-by-link error-correction.**

I. Introduction

A source Alice wishes to transmit information to a receiver Bob over a network with "noisy" links. Such a communication problem faces several challenges.

The primary challenge we consider is that in highly dynamic (wireless) environments the noise levels on each link might vary significantly across time, and hence be hard to estimate well. This issue of variable noise levels exacerbates at least two other challenges that had been considered settled by prior work.

One, since noise exists in the network, network coding might be dangerous. This is because all nodes mix information, so even a small number of bit-flips in transmitted packets may end up corrupting all the information flowing in the network, causing decoding errors. Prior designs for network error-correcting codes exist (for *e.g.* [11], [10]) but as we shall see they are ineffective against bit-flips in a highly dynamic noise setting. In particular, one line of work (*e.g.* [2], [8], [10]) treats even a single bit-flip in a packet as corresponding to the entire packet being corrupted, and hence results in rates that are too pessimistic – the fundamental problem is that the codes are defined over "large alphabets", and hence are poor at dealing with bit-flip errors. Another line of work (*e.g.* [11]) overlays network coding on link-by-link error correction, but requires accurate foreknowledge of the noise levels on each link to have good performance.

Two, in dynamic settings, the coding operations of nodes in the network may be unknown *a priori*. Under the bit-flip error-model we consider, however, the "transform-estimation" strategy of Ho *et al.* [6] does not work, since any headers pre-specified for this use can also end up being corrupted.

In this work we consider simultaneously the *reliability* and *universality* issues for random linear network coding. Namely, we design end-to-end distributed schemes that allow reliable network communications in the presence of "worst-case" network noise, wherein the erroneous bits can be arbitrarily distributed in different network packets with only the constraint that the total number of bit-flips is bounded from above. Internal network nodes just do linear network coding. Error-correction is only carried out at the receiver(s), which also estimates the linear transform imposed on the source's data by the network[1].

As noted above, our codes are robust to a wide variety of channel conditions – whether the noise bit-flips are evenly distributed among all packets, or even adversarially concentrated among just a few packets, our codes can detect and correct errors up to a network-wide bound on the total number of errors. Naïve implementations of prior codes (for instance, of link-by-link error-correcting codes [11]) that try to correct for worst-case network conditions may result in network codes with much lower rates (see the example in Section I-A below). Thus the naturally occurring diversity of network conditions works in our favour rather than against us.

Also, even though our codes correct *binary* errors rather than errors over larger symbol fields as in prior work, the end-to-end nature of our design enables our codes to be overlaid on classical linear network codes over finite fields (for instance, the random linear network codes of Ho *et al* [6]). Further, we free internal nodes from having to implement potentially computationally intensive link-by-link error-correction.

The main tool used to prove our results is a *transform metric* that may be of independent interest. It is structurally similar to the rank-metric used by Silva *et al.* [10], but has important differences that give our codes the power of universal robustness against binary noise (as opposed to the packet-based noise considered in [10], [8] and [11]).

A. Toy Example

We demonstrate via an example that in networks with worst-case bit-errors, prior schemes have inferior performance compared to our scheme. In Figure I-A the network has C paths (with a total of $2C$ links that might experience worst-case bit-flip errors).

Benchmark 1: If link-by-link error-correction[2] is applied as in [11], *every* link is then required to be able to correct

[1]As is common in coding theory, the upper and lower bounds on error-correction we prove also directly lead to corresponding bounds on error-detection – for brevity we omit discussing error-detection in this work.

[2]Since interior nodes might perform network coding, naïve implementations of end-to-end error-correcting codes are not straightforward – indeed – that is the primary goal of our constructions.

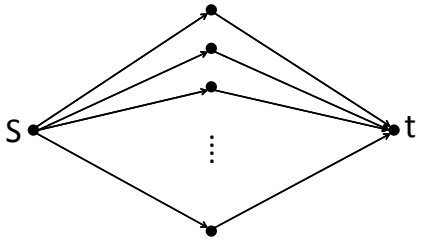

Fig. 1. A network with C parallel paths from the source to the destination. Each internal node performs random linear network coding.

$2Cpn$ *worst-case* bit-flip errors (since all the bit-errors may be concentrated in any single link). Using GV codes ([4], [12]) a rate of $1 - H(4Cp)$ is achievable on each link, and hence the overall rate scales as $C(1-H(4Cp))$. As C increases without bound, the throughput thus actually goes to zero. The primary reason is that every link has to prepare for the worst case number of bit-flips aggregated over the entire network, but in large networks, the total number of bit-flips in the worst-case might be too much for any single link to be able to tolerate.

Benchmark 2: Consider now a more sophisticated scheme, combining link-by-link error correction with end-to-end error-correction as in [10]. Suppose each link can correct $\frac{2nCp}{k}$ worst-case bit-flips, where k is a parameter to be determined such that the rate is optimized. Then at most k links will fail. Overlaying an end-to-end network error-correcting code as in [10] with link-by-link error-correcting codes such as GV codes (effectively leading to a concatenation-type scheme) leads to an overall rate of $(C - 2k)(1 - H(\frac{4Cp}{k}))$. For large C, this is better than the previous benchmark scheme since interior nodes no longer attempt to correct *all* worst-case errors and hence can operate at higher rates – the end-to-end code corrects the errors on those links that do experience errors. Nonetheless, as we observe below, our scheme still outperforms this scheme, since concatenation-type schemes in general have lower rates than single-layer schemes.

Our scheme: The rate achieved by our scheme (as demonstrated in Section III-B) is at least $C(1 - 2H(2p))$. As can be verified, this rate is higher than either of the benchmark schemes.

II. Model

A. Network model

We model our network by a directed acyclic multigraph[3], denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the set of nodes and $\mathcal{E}$ denotes the set of edges. A single source node $s \in \mathcal{V}$ and a set of sinks $\mathcal{T} \subseteq \mathcal{V}$ are pre-specified in $\mathcal{V}$. We denote $|\mathcal{E}|$ and $|\mathcal{T}|$, respectively the number of edges and sinks in the network, by E and S. A directed edge e leading from node u to node v can be represented by the vector (u, v), where u is called the *tail of* e and v is called the *head of* e. In this case e is called an *outgoing edge of* u and an *incoming edge of* v.

[3]Our model also allows non-interfering broadcast links in a wireless network to be modeled via a directed hypergraph – for ease of notation we restrict ourselves to just graphs.

The capacity of each edge is one packet – an length-n vector over a finite field $\mathbb{F}_{2^m}$ – here n and m are design parameters to be specified later. Multiple edges between two nodes are allowed – this allows us to model links with different capacities.[4] As defined in [1], the *network (multicast) capacity*, denoted C, is the minimum over all sinks $t \in \mathcal{T}$ of the mincut of $\mathcal{G}$ from the source s to the sink t. Without loss of generality, we assume there are C edges outgoing from s and incoming edges to t for all sinks $t \in \mathcal{T}$. [5]

B. Code model

The source node s wants to *multicast* a message M to each sinks $t \in \mathcal{T}$. To simplify notation, we consider henceforth just a single sink – our analysis can be directly extended to the multi-sink case. All logarithms in this work are to the base 2, and we use $H(p)$ to denote the *binary entropy function* $-p \log p - (1-p)\log(1-p)$.

Random linear network coding: All internal nodes in the network perform *random linear network coding* [6] over a finite field $\mathbb{F}_{2^m}$. Specifically, each internal node takes uniformly random linear combinations of each incoming packet to generate outgoing packets. That is, let e' and e index incoming and outgoing edges from a node v. The *linear coding coefficient from* e' *to* e is denoted by $f_{e',e} \in \mathbb{F}_q$. Let $\mathbf{Y}_e$ denote the packet (length-n vector over $\mathbb{F}_{2^m}$) transmitted on the edge e. Then $\mathbf{Y}_e = \sum f_{e',e}\mathbf{Y}_{e'}$, where the summation is over all edges e' incoming to the node v, and all arithmetic is performed over the finite field $\mathbb{F}_{2^m}$.

Mapping between $\mathbb{F}_2$ and $\mathbb{F}_{2^m}$: The noise considered in this work is binary in nature. Hence, to preserve the linear relationships between inputs and outputs of the network, we use the mappings given in Lemma 1 in [7]. These map addition and multiplication over $\mathbb{F}_{2^m}$ to corresponding (vector/matrix) operations over $\mathbb{F}_2$. More specifically, a bijection is defined from each symbol (from $\mathbb{F}_{2^m}$) of each packet transmitted on each edge, to a corresponding length-m bit-vector. For ease of notation henceforth, for each edge e and each $i \in \{1, \ldots, n\}$, we use $\mathbf{Y}_e$ and $\mathbf{Y}_e(i)$ solely to denote respectively the length-nm and length-m binary vectors resulting from the bijection operating on packets and their ith symbols, rather than the original analogues over $\mathbb{F}_{2^m}$ traversing that edge e. Separately, each *linear coding coefficient* $f_{e,e'}$ at each node is mapped via a homomorphism to a specific $m \times m$ binary matrix $F_{e,e'}$. The linear mixing at each node is then taken over the binary field – each length-m binary vector $\mathbf{Y}_{e'}(i)$ (corresponding to the binary mapping of the ith symbol of the packet $\mathbf{Y}_{e'}$ over the field $\mathbb{F}_{2^m}$) equals $\sum F_{e',e}\mathbf{Y}_{e'}(i)$. It is shown in [7] that an isomorphism exists between the binary linear operations defined above, and the original linear network code. In what

[4]By appropriate buffering and splitting edges into multiple edges, any network can be approximated into such a network with unit capacity edges.

[5]In cases where the number of outgoing edges from s (or the number of incoming edges to t) is not C, we can add a *source super-node* (or *sink super-node*) with C noiseless *edges* connecting to the original source (or sink) of the network. The change in the number of edges and probability of error on each edge are small compared to those of the original network, so our analysis essentially still applies.

follows, depending on the context, we use the homomorphism to switch between the scalar (over $\mathbb{F}_{2^m}$) and matrix (over $\mathbb{F}_2$) forms of the network codes' linear coding coefficients, and the isomorphism to switch between the scalar (over $\mathbb{F}_{2^m}$) and vector (over $\mathbb{F}_2$) forms of each symbol in each packet.

Noise: We consider "worst-case noise" in this work, wherein an arbitrary number of bit-flips can happen in any transmitted packet, subject to the constraint that no more that a fraction of p bits over all transmitted packets are flipped. The *noise matrix* Z is an $Em \times n$ binary matrix with at most $pEmn$ nonzero entries which can be arbitrarily distributed. In particular, the $m(i-1)+1$ through the mi rows of Z represent the bit flips in the ith packet $\mathbf{Y}_{e_i}$ transmitted over the network. If the $(km+j)$th bit of the length-mn binary vector is flipped (that is, the jth bit of the kth symbol over $\mathbb{F}_{2^m}$ in $\mathbf{Y}_{e_i}$ is flipped), then the $(m(i-1)+j, k)$ bit in Z equals 1, else it equals 0. Thus the noise matrix Z represents the *noise pattern* of the network. To model the noise as part of the linear transform imposed by the network, we add an artificial super-node s' connected to all the edges in the network, injecting noise into each packet transmitted on each edge in the network according to entries of the noise matrix Z.

Source: The source has a set of 2^{Rmn} messages $\{M\}$ it wishes to communicate to each sink, where R is the *rate* of the source. Corresponding to each message M it generates a *codeword* $X(M)$ using the encoders specified in Section III-B (to make notation easier we usually do not explicitly reference the parameter M and instead refer simply to X). This X is represented by a $C \times n$ matrix over $\mathbb{F}_{2^m}$, or alternatively a a $Cm \times n$ matrix over $\mathbb{F}_2$. Each row of this matrix corresponds to a packet transmitted over a distinct edge leaving the source.

Receiver(s): Each sink t receives a batch of C packets. Similarly to the source, it organizes the received packets into a matrix Y, which can be equivalently viewed as a $C \times n$ matrix over $\mathbb{F}_{2^m}$ or a $Cm \times n$ binary matrix. Each sink t decodes the message $\hat{M}$ from the received matrix Y using the decoders specified in Section III-B.

Transfer matrix and Impulse response matrix: Having defined the linear coding coefficients of internal nodes, the packets transmitted on the incoming edges of each sink t can inductively be calculated as linear combinations of the packets on the outgoing edges of s. We denote the $C \times C$ *transfer matrix from the outgoing edges of s to the incoming edges of t by T*, over the finite field $\mathbb{F}_{2^m}$. Alternatively, using the homomorphism described above, T may be viewed as as $Cm \times Cm$ binary matrix.

We similarly define $\hat{T}$ to be the *impulse response matrix*, which is the transfer matrix from the source s to the sink t. Note that T is a sub-matrix of $\hat{T}$, composed specifically of the C columns of T corresponding to the C outgoing edges of s.

In this work we require that every $C \times C$ sub-matrix of $\hat{T}$ is invertible. As noted in, for instance, [9], [6] this happens with high probability for random linear network codes. Alternatively, deterministic designs of network error-correcting codes [2] also have this property.

Using the above definitions the network can thus be abstracted by the equation (1) below as a *worst-case binary-error network channel*.

$$Y = TX + \hat{T}Z. \tag{1}$$

Similar equations have been considered before (for instance in [2], [8], [10]) – the key difference in this work is that we are interested in Z matrices which are fundamentally best defined over the binary field, and hence, when needed, transform the other matrices in (1) also into binary matrices.

Performance of code: The source encoders and channel decoders specified in Section III-B together comprise *worst-case binary-error-correcting network codes*. A *good* worst-case binary-error-correction network channel code has the property that, for all messages M, and noise patterns Z with at most $pEmn$ bit-flips, $\hat{M} = M$. A rate R is said to be *achievable* for the worst-case binary-error channel if, for all sufficiently large n, there exists a good code with rate R.

C. Transform Metric

We first define a "natural" distance function between binary matrices M_1 and M_2 related as $M_1 = M_2 + BZ$ for some matrices B and Z.

Let M_1 and M_2 be arbitrary $a \times b$ binary matrices. Let B be a given $a \times c$ matrix with full column rank. Let $M_1(i)$ and $M_2(i)$ denote respectively the ith columns of M_1 and M_2. We define $d_B(M_1, M_2)$, the *transform distance between M_1 and M_2 in terms of B*, as follows.

Definition 1. *Let $\delta(i)$ denote the minimal number of columns of B that need to be added to $M_1(i)$ to obtain $M_2(i)$. Then the transform distance $d_B(M_1, M_2)$ equals $\sum_{i=1}^{b} \delta(i)$.*

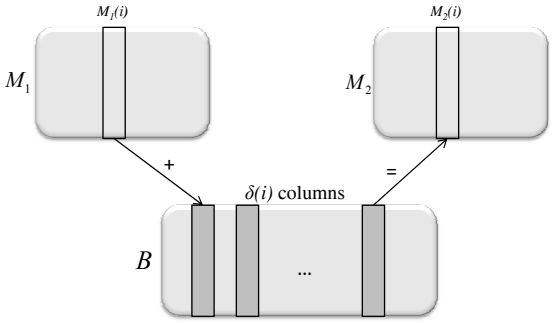

Fig. 2. Transfor metric: the minimal number of columns of B that need to be added to $M_1(i)$ to obtain $M_2(i)$ is $\delta(i)$.

III. Main results

In this section we present our main results. In Theorem 1 in Subsection III-A we present an upper bound on the rates of communication achievable by any code over networks that have "worst-case" bit-flip errors. Our bounding technique is motivated by the corresponding Hamming bound technique in classical coding theory [5] – the main challenge lies in deriving good *lower* bounds for the "volumes of spheres" in the channel model and corresponding metric defined in Section II-C.

In Subsection III-B we discuss schemes that achieve "good" rates of communication over networks that have "worst-case" bit-flip errors. We present three schemes motivated by the well-known Gilbert-Varshamov (GV) bound from classical coding theory [4], [12] – again, the challenge lies in deriving good *upper* bounds on the volumes of spheres in the metric we define. Theorem 2 considers the *coherent* scenario, *i.e.*, when the linear coding coefficients in the network (or at least the transfer matrix T and the impulse response matrix $\hat{T}$) are known in advance to the receiver. We use this setting primarily for exposition, since the proof is somewhat simpler than the proof for the *non-coherent* setting, when no advance information about the topology of the network, the linear coding coefficients used, or T or $\hat{T}$ is known in advance to the receiver. In Theorem 3 we are able to demonstrate that essentially the same rates as in Theorem 2 are still achievable, albeit with an rate-loss that is asymptotically negligible in the block-length n.

As we see below, the functional forms of both the Hamming-type upper bounds and the GV-type lower bounds we derive are structurally very similar to those of the classical Hamming and GV bounds.

A. Hamming-type bound

Theorem 1. *For all p less than $C/(2Em)$ an upper bound on the achievable rate of any code over the worst-case binary-error channel is $1 - H(p)(\frac{E}{C})$.*

Proof: Since each transmitted codeword X is a $Cm \times n$ binary matrix, the number of possible choices of X is at most 2^{Cmn}. But suppose X is transmitted, by the definitions of the worst-case bit-error channel, the received Y lies in the radius-$pEmn$ ball (in the transform metric) $\mathcal{B}_{\hat{T}}(TX, pEmn)$ defined as $\{Y | d_{\hat{T}}(TqX, Y) \leq pEmn\}$. For the message corresponding to X to be uniquely decodable, it is necessary that the balls $\mathcal{B}_{\hat{T}}(TX, pEmn)$ be non-intersecting for each X chosen to be in the codebook. Hence to get an upper bound on the number of codewords that can be chosen, we need to derive a lower bound of the volume of $\mathcal{B}_{\hat{T}}(TX, pEmn)$. Recall that Y equals $TX + \hat{T}Z$. Hence we need to bound from below the number of distinct values of $\hat{T}Z$ for Z with at most $pEmn$ ones.

We consider the case that Z has exactly $pEmn$ ones that are equally distributed among columns of Z – hence every column of Z has pEm ones in it. We now show that, in the worst case, every such distinct matrix Z results in distinct $\hat{T}Z$. Suppose not – in that case there exist distinct Z and Z' with pEm ones in each columns of both matrix such that $\hat{T}Z$ equals $\hat{T}Z'$, *i.e.*, $\hat{T}(Z - Z')$ equals the zero matrix. In particular, for at least some column of Z and Z', say $Z(i)$ and $Z'(i)$, it must be the case that $\hat{T}(Z(i) - Z'(i))$ equals 0. But by assumption each column of both Z and Z' has less that $pEm < C/2$ ones, and hence $Z(i) - Z'(i)$ has less than C ones in it.

We now view $\hat{T}$ and Z as matrices over $\mathbb{F}_{2^m}$. From the argument above, $Z(i) - Z'(i)$ has less than C non-zero elements over $\mathbb{F}_{2^m}$ in it (since an element over $\mathbb{F}_{2^m}$ is zero if and only if each of the m bits in its binary representation is zero). Hence $\hat{T}(Z(i) - Z'(i))$ is a linear combination over $\mathbb{F}_{2^m}$ of strictly less than C columns of $\hat{T}$. But as to the matrix $\hat{T}$ viewed over $\mathbb{F}_{2^m}$, since we are deriving a worst-case upper bound, we can also require that every $C \times C$ sub-matrix of $\hat{T}$ is invertible (as noted before this happens with high probability for random linear network codes). Hence $\hat{T}(Z(i) - Z'(i))$ cannot equal the zero vector, which leads to a contradiction.

Hence the number of distinct values for $\hat{T}Z$ is at least the number of distinct values for Z with at most pEm ones in each column. This equals is at least $\binom{Em}{pEm}^n$, which by Stirling's approximation [3] is at least $2^{H(p)Emn - \log(Em+1)}$. The total number of $Cm \times n$ binary matrices is 2^{Cmn}. Thus an upper bound on the size of any codebook for the worst-case binary-error channel is

$$\frac{2^{Cmn}}{2^{EmnH(p) - \log(Em+1)}} = 2^{(1 - H(p)\frac{E}{C} + \frac{\log(Em+1)}{Cmn})Cmn},$$

which, asymptotically in n, gives the Hamming-type upper bound on the rate of any code as $1 - H(p)\frac{E}{C}$. □

B. Gilbert-Varshamov-type bounds

1) Coherent GV-type network codes: We first discuss the case when the network transfer matrix T and impulse response matrix $\hat{T}$ are known in advance.

Codebook design: Initialize the set $\mathcal{S}$ as the set of all binary $Cm \times n$ matrices. Choose a uniformly random $Cm \times n$ binary matrix X as the first codeword. Eliminate from $\mathcal{S}$ all matrices in the radius-$2pEmn$ ball (in the transform metric) $\mathcal{B}_{\hat{T}}(TX, 2pEmn)$. Then choose a matrix Y' uniformly at random in the remaining set and choose $X' = T^{-1}Y'$ as the second codeword. Now, further eliminate all matrices in the radius-$2pEmn$ ball $\mathcal{B}_{\hat{T}}(TX', 2pEmn)$ from $\mathcal{S}$, choose a random Y' from the remaining set, and choose the third codeword X'' as $X'' = T^{-1}Y''$. Repeat this procedure until the set $\mathcal{S}$ is empty.

Theorem 2. *Coherent GV-type network codes achieve a rate of at least $1 - H(2p)\frac{E}{C}$.*

Proof: For this theorem, we need an upper bound on $\mathcal{B}_{\hat{T}}(TX, 2pEmn)$ (rather than a lower bound on $\mathcal{B}_{\hat{T}}(TX, pEmn)$ as in Theorem 1). Recall that $Y = TX + \hat{T}Z$ The number of different Y, or equivalently, different $\hat{T}Z$, can be bounded from above by the number of different Z. This equals $\sum_{i=0}^{2pEmn} \binom{Emn}{i}$. The dominant term this summation is when i equals $2pEmn$. Hence the summation can be bounded from above by $(2pEmn + 1)\binom{Emn}{2pEmn}$. By Stirling's approximation [3] this is at most $(2pEmn + 1)2^{H(2p)Emn}$.

Thus a lower bound on the size of the codebook for coherent GV-type

$$\frac{2^{Cmn}}{(2pEmn + 1)2^{H(2p)Emn}} = 2^{(1 - H(2p)\frac{E}{C} - \frac{\log(2pEmn+1)}{n})Cmn},$$

which, asymptotically in n, gives the rate of coherent GV-type bound network codes as $1 - H(2p)\frac{E}{C}$. □

2) Non-coherent GV-type network codes: The assumption that T and $\hat{T}$ are known in advance to the receiver is often unrealistic, since the random linear coding coefficients in the network are usually chosen on the fly. Hence we now consider the non-coherent setting, wherein T and $\hat{T}$ are not known in advance. We demonstrate that despite this lack of information the same rates as in Theorem 2 are achievable in the non-coherent setting.

The number of all possible $\hat{T}$ is at most by 2^{CEm} since $\hat{T}$ is a $C \times E$ matrix $\mathbb{F}_{2^m}$ – the crucial observation is that this number is independent of the block-length n. Hence in the non-coherent GV setting, we consider *all* possible values of $\hat{T}$ (and hence T, since it comprises of a specific subset of C columns of $\hat{T}$).

Codebook design: Initialize the set $\mathcal{S}$ as the set of all binary $Cm \times n$ matrices. Choose a uniformly random $Cm \times n$ binary matrix X as the first codeword. For each $C \times E$ matrix $\hat{T}$ (over the field $\mathbb{F}_{2^m}$), eliminate from $\mathcal{S}$ all matrices in the radius-$2pEmn$ ball (in the transform metric) $\mathcal{B}_{\hat{T}}(TX, 2pEmn)$. Then choose a matrix Y' uniformly at random in the remaining set and choose $X' = T^{-1}Y'$ as the second codeword. Now, further eliminate all matrices in the radius-$2pEmn$ ball $\mathcal{B}_{\hat{T}}(TX', 2pEmn)$ from $\mathcal{S}$, choose a random Y' from the remaining set, and choose the third codeword X'' as $X'' = T^{-1}Y''$. Repeat this procedure until the set $\mathcal{S}$ is empty.

Theorem 3. *Non-coherent GV-type network codes achieve a rate of at least* $1 - H(2p)\frac{E}{C}$.

Proof: The crucial difference with the proof of Theorem 2 is in the process of choosing codewords – at each stage of the codeword elimination process, at most $2^{CEm}|\mathcal{B}_{\hat{T}}(TX', 2pEmn)|$ potential codewords are eliminated (rather than $|\mathcal{B}_{\hat{T}}(TX', 2pEmn)|$ potential codewords as in Theorem 2). Hence the number of potential codewords that can be chosen in the codebook is at least

$$\frac{2^{Cmn}}{2^{CEm}(2pEmn+1)2^{H(2p)Emn}}$$

which equals

$$2^{(1-H(2p)\frac{E}{C}-\frac{(\log(2pEmn+1)+E)}{n})Cmn}.$$

As can be verified, asymptotically in n this leads to the same rate of $1 - H(2p)\frac{E}{C}$ as in Theorem 2. □

Note: Our proposed codes via concatenation schemes so that their encoding and decoding complexity grows only polynomially in the block-length (albeit exponentially in network parameters).

C. Scale of Parameters

We now investigate the regime of p wherein our results are meaningful.

Claim 1. *For all* p *less than* $\min(\frac{C}{2Em}, \frac{1}{2^{m+1}})$ *the Hamming-type bounds and GV-type hold.*

Proof: The Hamming-type bound in Theorem 1 requires $pEm < \frac{C}{2}$.

For the GV-type bound in Theorems 2 and 3 to give non-negative rates, $H(2p)\frac{E}{C} < 1$. Hence when p is very small,

$$H(2p)\frac{E}{C} \to 2p(\log 1/(2p))\frac{E}{C} \quad (2)$$

$$< \frac{C}{Em}(\log 1/(2p))\frac{E}{C} = \frac{-\log 2p}{m} \quad (3)$$

$$< 1 \quad (4)$$

where (2) follows from the limiting behaviour of the binary entropy function for small p, (3) is because $p < C/(2Em)$ (our first condition), and (4) is because $p < 2^{-(m+1)}$ (our second condition).

IV. Conclusion

In this work we investigate upper and lower bounds for the performance of end-to-end error-correcting codes for worst-case binary errors. This model is appropriate for highly dynamic wireless networks, wherein the noise-levels on individual links might be hard to accurately estimate. We demonstrate significantly better performance for our proposed schemes, compared to prior benchmark schemes.

References

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

[2] Ning Cai and R. W. Yeung. Network coding and error correction. In *Proc. 2002 IEEE Inform. Theory Workshop*, pages 119–122, Bangalore, India, October 20–25, 2002.

[3] T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley and Sons, 1991.

[4] E. N. Gilbert. A comparison of signalling alphabets. *Bell Systems Technical Journal*, 31:504–522, 1952.

[5] R. W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29:147–160, 1950.

[6] T. Ho, R. Koetter, M. Médard, D. R. Karger, and M. Effros. The benefits of coding over routing in a randomized setting. In *Proc. IEEE Int. Symp. Information Theory*, page 442, Yokohama, Japan, June 29–July 4, 2003.

[7] S. Jaggi, M. Effros, T. Ho, and M. Médard. On linear network coding. In *Proceedings of 42nd Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, 2004.

[8] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Médard. Resilient network coding in the presence of Byzantine adversaries. In *Proc. 26th IEEE Int. Conf. on Computer Commun.*, pages 616–624, Anchorage, AK, May 2007.

[9] R. Koetter and M. Médard. An algebraic approach to network coding. *IEEE Transactions on Information Theory*, 11(5):782–795, October 2003.

[10] R. Kötter and F. R. Kschischang. Coding for errors and erasures in random network coding. *IEEE Transactions on Information Theory*, 54(8):3579–3591, Aug. 2008.

[11] R. W. Yeung L. Song and N. Cai. A separation theorem for single-source network coding. *IEEE Transactions on Information Theory*, 52(5):1861–1871, 2006.

[12] R. R. Varshamov. Estimate of the number of signals in error correcting codes. *Dokl. Acad. Nauk*, 117:739–741, 1957.