MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Iterative Quantization Using Codes On Graphs

Emin Martinian and Jonathan S. Yedidia

TR2003-120 September 2003

Abstract

We study codes on graphs combined with an iterative message passing algorithm for quantization. Specifically, we consider the binary erasure quantization (BEQ) problem which is the dual of the binary erasure channel (BEC) coding problem. We show that duals of capacity achieving codes for the BEC yield codes which approach the minimum possible rate for the BEQ. In contrast, low density parity check codes cannot achieve the minimum rate unless their density grows at least logarithmically with block length. Furthermore, we show that duals of efficient iterative decoding algorithms for the BEC yield efficient encoding algorithms for the BEQ. Hence our results suggest that graphical models may yield near optimal codes in source coding as well as in channel coding and that duality plays a key role in such constructions.

Proceedings of the 41st Annual Allerton Conference on Communications, Control, and Computing, October 2003

Publication History:–

1. First printing, TR-2003-120, September 2003

Iterative Quantization Using Codes On Graphs

Emin Martinian
Massachusetts Institute of Technology
Cambridge, MA 02139
emin@allegro.mit.edu

Jonathan S. Yedidia
Mitsubishi Electronic Research Labs
Cambridge, MA 02139
yedidia@merl.com

Abstract

We study codes on graphs combined with an iterative message passing algorithm for quantization. Specifically, we consider the binary erasure quantization (BEQ) problem which is the dual of the binary erasure channel (BEC) coding problem. We show that duals of capacity achieving codes for the BEC yield codes which approach the minimum possible rate for the BEQ. In contrast, low density parity check codes cannot achieve the minimum rate unless their density grows at least logarithmically with block length. Furthermore, we show that duals of efficient iterative decoding algorithms for the BEC yield efficient encoding algorithms for the BEQ. Hence our results suggest that graphical models may yield near optimal codes in source coding as well as in channel coding and that duality plays a key role in such constructions.

1 Introduction

Researchers have discovered that error correction codes defined on sparse graphs can be iteratively decoded with low complexity and vanishing error probability at rates close to the Shannon limit. Based on the close parallels between error correction and data compression, we believe that similar graphical codes can approach the fundamental limits of data compression with reasonable complexity. Unfortunately, the existing suboptimal channel decoding algorithms for graphical codes generally fail unless the decoder input is already near a codeword. Since this is usually not the case in source coding, either a new type of graph or a new suboptimal algorithm (or both) is required.

Before developing iterative quantization techniques it is worth investigating the potential gains of such an approach over existing systems. For asymptotically high rates, when compressing a continuous source with finite moments relative to mean square error (MSE) distortion, entropy coded scalar quantization (ECSQ) is 1.53 dB from the rate-distortion limit [1]. For moderate rates the gap is larger: in quantizing a Gaussian source relative to MSE distortion, ECSQ systems are 1.6–3.4 dB away from the rate-distortion limit. For these parameters, trellis coded quantization (TCQ) using a 256-state code with optimal quantization has a gap of 0.5–1.4 dB [2]. For higher rates, sources with larger tails (*e.g.*, a source with a Laplacian distribution), or sources with memory, the gaps are larger. Thus for memoryless sources quantized at moderate rates, new codes have the potential to improve performance by the noticeable margin of a few decibels. More generally, the codes on graphs paradigm may prove valuable in a variety of scenarios involving speech, audio, video and other complicated sources.

To illustrate possible approaches to developing graphical codes we focus on the binary erasure quantization (BEQ) problem which is the dual of the binary erasure channel (BEC) coding problem. First we describe the BEQ problem model in Section 2. Next, in Section 3 we present our main result for the BEQ: duals of low density parity check codes can be analyzed, encoded, and decoded by dualizing the corresponding techniques for the BEC. Specifically, by dualizing capacity achieving codes for the BEC we obtain rate-distortion approaching codes for the BEQ. Finally, we close with some concluding remarks in Section 4.

2 Quantization Model

Vectors and sequences are denoted with an arrow (*e.g.*, $\vec{x}$). Random variables or random vectors are denoted using the sans serif font (*e.g.*, x or $\vec{\mathsf{x}}$). We consider the standard (memoryless) data compression problem and represent an instance of the problem with the tuple $(\mathcal{S}, p_{\mathsf{s}}(s), d(\cdot,\cdot))$ where $\mathcal{S}$ represents the source alphabet, $p_{\mathsf{s}}(s)$ represents the source distribution, and $d(\cdot,\cdot)$ represents a distortion measure. Specifically, a source $\vec{\mathsf{s}}$ consists of a sequence of n random variables s_1, s_2, ..., s_n each taking values in $\mathcal{S}$ and generated according to the distribution $p_{\vec{\mathsf{s}}}(\vec{s}) = \prod_{i=1}^{n} p_{\mathsf{s}}(s_i)$. A rate R encoder $f(\cdot)$ maps $\vec{\mathsf{s}}$ to an integer in $\{1, 2, \ldots, 2^{nR}\}$, and the corresponding decoder $g(\cdot)$ maps the resulting integer into a reconstruction $\vec{\mathsf{q}}$. Distortion between the source $\vec{\mathsf{s}}$ and the reconstruction $\vec{\mathsf{q}}$ is measured via $D = \frac{1}{n}\sum_{i=1}^{n} d(\mathsf{s}_i, \mathsf{q}_i)$.

Shannon derived the minimum possible rate required by any data compression system operating with distortion D. The so-called rate-distortion function is given by the formula

$$R(D) = \min_{p_{\mathsf{q}|\mathsf{s}}(q|s): E[d(\mathsf{s},\mathsf{q})] \le D} I(\mathsf{s};\mathsf{q}) \tag{1}$$

where $I(\cdot;\cdot)$ denotes mutual information and $E[\cdot]$ denotes expectation.

2.1 Binary Erasure Quantization

To highlight connections between error correction and data compression, we consider the binary erasure quantization (BEQ) problem where the source vector consists of ones, zeros, and "erasures" represented by the symbol $*$. Neither ones nor zeros may be changed, but erasures may be quantized to either zero or one. Practically, erasures may represent source samples which are missing, irrelevant, or corrupted by noise and so do not affect the distortion regardless of the value they are assigned. Formally, the BEQ problem with erasure probability e corresponds to

$$\mathcal{S} = \{0, 1, *\} \tag{2a}$$

$$p_{\mathsf{s}}(s) = \frac{1-e}{2}\cdot\delta(s) + \frac{1-e}{2}\cdot\delta(s-1) + e\cdot\delta(s-*) \tag{2b}$$

$$d(a,b) = 0 \text{ if } a = * \text{ or } a = b, \text{ and 1 otherwise.} \tag{2c}$$

It is straightforward to show that for $D = 0$ the distribution

$$p_{\mathsf{q}|\mathsf{s}}(q|s) = \delta(q-s) \text{ if } s \in \{0,1\}, \text{ and } \frac{1}{2}\cdot\delta(q) + \frac{1}{2}\cdot\delta(q-1) \text{ if } s = *. \tag{3}$$

optimizes (1) and yields the value of the rate-distortion function at $D = 0$:

$$R_{\text{BEQ}}(D = 0) = 1 - e. \tag{4}$$

3 Codes For Erasure Quantization

It is well-known that the encoder for a quantizer serves a similar function to the decoder for an error correcting code in the sense that both take a vector input (*i.e.*, a source to quantize or channel output to decode) and map the result to bits (*i.e.*, the compressed source or the transmitted message). The decoder for a quantizer can similarly be identified with the encoder for an error correcting code in the sense that both take bits as input and produce a vector (*i.e.*, a source reconstruction or a channel input). Thus it is natural to investigate whether swapping the encoder and decoder for a good error correcting code such as a low density parity check (LDPC) code produces a good quantizer.

3.1 LDPC Codes Are Bad Quantizers

One benefit of studying the BEQ problem is that it demonstrates why low density parity check (LDPC) codes are inherently unsuitable for quantization. Specifically, consider an LDPC code like the one illustrated in Fig. 1 using Forney's normal graph notation [3]. If all the variables connected to a given check are not erased, then there is an even chance that no code symbol can match the source in that position and thus the distortion will be positive regardless of the code rate. Thus, as stated in Theorem 1 and proved in Appendix A, successful decoding is asymptotically unlikely unless the density of *every* parity check matrix for the code increases logarithmically with the block length.[1]

Theorem 1. *Let $\mathcal{C}_{(n)}$ be a sequence of linear codes of length n and fixed rate R such that the probability that binary erasure quantization using $\mathcal{C}_{(n)}$ of a random source sequence with $e \cdot n$ erasures will succeed with zero distortion is bounded away from 0 as $n \to \infty$. Then regardless of the values of R and e, the degree of the parity-check nodes in any parity-check graph representation of $\mathcal{C}_{(n)}$ must increase at least logarithmically with n.*

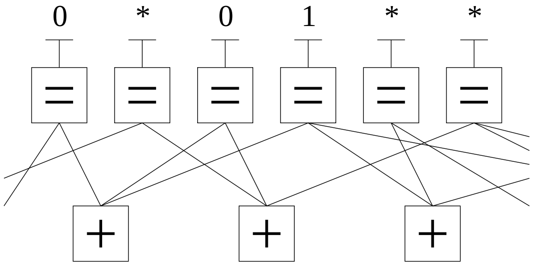

Figure 1: Using an LDPC code for binary erasure quantization. The boxes with = signs are repetition nodes: all edges connected to an = box must have the same value. The boxes with + signs are check nodes: the modulo 2 sum of the values on edges connected to a + box must be 0. The source consists of 0's, 1's, and erasures represented by $*$'s. Erasures may be quantized to 0 or 1 while incurring no distortion. A non-zero distortion must be incurred for the source shown above since the left-most check cannot be satisfied.

The poor performance of LDPC codes for quantization may seem surprising in light of their excellent properties in channel coding, but it has long been recognized that good

[1] We may expect the density of a code to increase as it approaches the *capacity* or *rate-distortion function*, but a code whose density also increases with *block length* seems undesirable.

codes for error correction and quantization may be different. The former is essentially a packing problem where the goal is to place as many codewords as possible in a given space such that the codewords are far apart and can be distinguished despite noise. The latter is a covering problem where the goal is to place as few codewords as possible in a given space such that every point in space is near at least one codeword.

From any good error correcting code (respectively data compression code) it is easy to obtain another code which is almost as good at error correction (resp. data compression) but terrible at source coding (resp. channel coding). For example, removing half the codewords has an asymptotically negligible effect on error correction since it only decreases the rate by $1/n$ and only increases robustness. But, removing half the codewords can dramatically hinder source coding since half the time the source may be very far from the nearest codeword. Conversely, doubling the number of codewords has an asymptotically negligible effect on data compression since the rate only increases by $1/n$ while the distortion may decrease slightly. But doubling the number of codewords can be catastrophic for error correction if it drastically reduces the distance between codewords.

3.2 Dual LDPC Codes

Many researchers have explored duality relationships between error correction and source coding. Such work demonstrates that often a good solution for one problem can be obtained by dualizing a good solution to the other. Continuing in this tradition, we study the properties of dual LDPC codes for binary erasure quantization.

Formally, a length n binary linear code $\mathcal{C}$ is a subspace of the n dimensional vector space over the binary field and the dual code $\mathcal{C}^\perp$ is the subspace orthogonal to $\mathcal{C}$. For LDPC codes, the code $\mathcal{C}$ is usually specified by the parity check matrix $\mathbf{H}$ representing the constraint that $\mathbf{H}\vec{x}^T = 0$ if and only if $\vec{x}$ is a codeword. To obtain the dual code $\mathcal{C}^\perp$ we can recall that the generator $\mathbf{G}^\perp$ of $\mathcal{C}^\perp$ is exactly $\mathbf{H}$. If the code $\mathcal{C}$ is represented by a normal graph as in Fig. 1, then the graph of the dual code $\mathcal{C}$ can be obtained by swapping $+$ and $=$ nodes [3]. In dualizing the code graph in this manner it may be useful to note that while the graph of $\mathcal{C}$ obtained from $\mathbf{H}$ represents a syndrome former for $\mathcal{C}$, the dualized graph represents an encoder for $\mathcal{C}^\perp$.

For example, Fig. 2 is obtained by dualizing the code graph in Fig. 1. Notice that while the original code cannot quantize the source with distortion 0, the dual code can. Intuitively, the advantage of a low density encoder structure is that it provides a simple representation of a basis which can be used to construct the desired vector. In the following sections, we investigate the properties of dual LDPC codes for quantization with both optimal quantization and iterative quantization.

3.3 Optimal Quantization/Decoding and Duality

The following theorem (proved in Appendix A) demonstrates the dual relationship between channel decoding and source quantization using optimal decoding/quantization algorithms.

Theorem 2. *A channel decoder for the code $\mathcal{C}$ can correctly decode every received sequence with the erasure pattern*[2] *$\vec{e}$ if and only if a quantizer for the code $\mathcal{C}^\perp$ can successfully quantize every*[3] *source sequence with the erasure pattern $\vec{e}^\perp = 1 - \vec{e}$.*

[2] If symbol i is erased (resp. unerased) then $e_i = 1$ (resp. $e_i = 0$) in our notation for erasure patterns.

[3] Note that some source sequences (*e.g.*, the all zero sequence) can be successfully quantized using

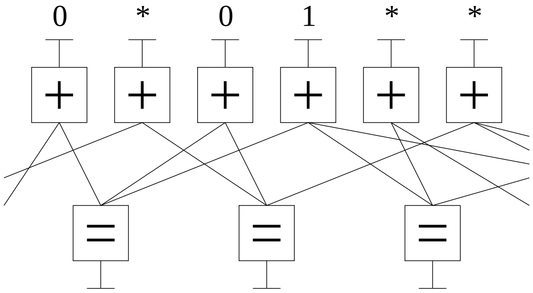

Figure 2: Using the dual of an LDPC code for binary erasure quantization. Choosing values for the variables at the bottom produces a codeword. The values for each sample of the resulting codeword are obtained by taking the sum modulo 2 of the connected variables. In contrast to Fig. 1, this structure can successfully match the source with no distortion if the bottom 3 variables are set to 0, 0, 1.

From this result we immediately obtain the following Corollary.

Corollary 1. *Let $\mathcal{C}_{(n)}$ be a sequence of linear codes which achieves the capacity of a binary erasure channel with erasure probability* e *using optimal decoding. The sequence $\mathcal{C}_{(n)}^{\perp}$ obtained by taking the duals of $\mathcal{C}_{(n)}$ achieves the minimum rate for $D = 0$ for the BEQ with erasure probability $\mathrm{e}^{\perp} = 1 - e$ using optimal quantization.*

The statement and proof of the two preceding results contain a curious duality between erased/known symbols in source coding and known/erased symbols in channel coding. A similar duality exists between a likelihood ratio, λ, and its Fourier transform $\Lambda = \frac{1-\lambda}{1+\lambda}$ used in dualizing the sum-product algorithm [3, pp. 545–546]. Specifically, the Fourier transform maps known/erased likelihood ratios to erased/known likelihood ratios.

3.4 Iterative Decoding/Quantization and Duality

In the following we first review the intuition behind iterative erasure decoding algorithms and describe the particular decoding algorithm we consider in Table 1. Next we outline the intuition behind a similar approach for iterative quantization and precisely describe our quantization algorithm in Table 2. Finally, we show that these algorithms are duals.

3.4.1 Iterative Erasure Decoding

Many iterative message-passing decoding algorithms are essentially based on the following idea. The outgoing message on edge i of a $+$ node is the modulo-2 sum of all incoming messages (excluding edge i) with the proviso that if any incoming message (excluding edge i) is $*$ then the outgoing message is also $*$. For an $=$ node, the outgoing message on edge i is $*$ only if all other incoming messages are $*$, otherwise the outgoing message is the same as the known incoming message or messages. These message-passing rules can

$\mathcal{C}^{\perp}$ regardless of $\vec{e}^{\perp}$. Similarly, a system which decodes ambiguous received sequences to the all zero sequence may succeed even when many erasures occur. Thus to obtain the desired equivalence between correct decoding and successful quantization we define correct decoding (resp. successful quantization) as being able to deduce the transmitted codeword (resp. a codeword matching non-erased positions of the source) for every possible received sequence (resp. source) with the the erasure pattern $\vec{e}$ (resp. $\vec{e}^{\perp}$).

be interpreted as determining the outgoing message on edge i by applying the following "sum" and "product" formulas to all other incoming messages. [4]

$$\begin{array}{c||c|c|c} + & 0 & 1 & * \\ \hline\hline 0 & 0 & 1 & * \\ \hline 1 & 1 & 0 & * \\ \hline * & * & * & * \end{array} \quad \text{and} \quad \begin{array}{c||c|c|c} \times & 0 & 1 & * \\ \hline\hline 0 & 0 & \# & 0 \\ \hline 1 & \# & 1 & 1 \\ \hline * & 0 & 1 & * \end{array} \tag{5}$$

It is well-known that such algorithms yield optimal decoding on a tree and also perform well on graphs with cycles provided appropriate scheduling and initialization rules are selected. Initializing all messages to $*$ and using sequential or parallel schedules are common choices. For the purpose of proving theorems, we consider a sequential schedule in the ERASURE-DECODE algorithm of Table 1.

Table 1: An algorithm for iteratively decoding data with erasures.

```
ERASURE-DECODE($\mathbf{H}$,$\vec{y}$)
while $\vec{y}$ has at least one erased sample do
    if $\exists$ row $i$ of $\mathbf{H}$ (i.e., a check) connected to exactly one erased variable $y_j$ then
        Set $y_j$ to be the XOR of all unerased bits in the check
    else
        return FAIL
    end if
end while
Set $\vec{x}$ to the message variables obtained from $\vec{y}$
return the message variables $\vec{x}$
```

3.4.2 Iterative Erasure Quantization

The message-passing rules in (5) can also be applied to the BEQ problem for graphs without cycles provided some form of tie-breaking is used. Specifically, some variables will receive erasure messages even after the algorithm has completed. Such variables can be arbitrarily chosen to be either 0 or 1 and still produce a valid quantization. For example in quantizing the source $(*, *, 1)$ with a (3,2) single parity check code, both $(1, 0, 1)$ and $(0, 1, 1)$ are equally valid results and this tie can broken arbitrarily.

On a graph with cycles, however, generalizing this approach by initializing all unknown messages to $*$ usually fails. For example, on the dual of a Gallager code or a code like the one represented in Fig. 2 such an initialization rule leads to all messages being erased at every step of the algorithm. To perform effective tie-breaking, we need to distinguish between variables which can be arbitrarily set to 0 or 1 and variables which have not yet received enough information to be determined.

One way to distinguish between these cases is to denote the former as erasures with the symbol $*$ and the latter as null messages with the symbol $\varnothing$, and initialize all messages to $\varnothing$. With this initialization, we can use the following "sum" and "product" rules: [5]

$$\begin{array}{c||c|c|c|c} + & 0 & 1 & * & \varnothing \\ \hline\hline 0 & 0 & 1 & * & \varnothing \\ \hline 1 & 1 & 0 & * & \varnothing \\ \hline * & * & * & * & * \\ \hline \varnothing & \varnothing & \varnothing & * & \varnothing \end{array} \quad \text{and} \quad \begin{array}{c||c|c|c|c} \times & 0 & 1 & * & \varnothing \\ \hline\hline 0 & 0 & \# & 0 & 0 \\ \hline 1 & \# & 1 & 1 & 1 \\ \hline * & 0 & 1 & * & \varnothing \\ \hline \varnothing & 0 & 1 & \varnothing & \varnothing \end{array} \tag{6}$$

[4] In the product rule for erasure decoding, the symbol # denotes a contradiction which is impossible if only erasures and no errors occurred.

[5] In the product rule for erasure quantization, the symbol # denotes a contradiction. If a contradiction is generated then quantizing the given source with no distortion is impossible and the algorithm fails.

Specifically, the outgoing message from a + node in a graph like Fig. 2 is computed by combining incoming messages from all other edges with the + rule. The outgoing message from an = node is computed by combining incoming messages from all other edges with the × rule. Whenever an = node has all incoming messages being $*$, the value of the node is arbitrary. This tie can be broken by arbitrarily choosing a value of 0 or 1 provided the tie is broken consistently. Essentially, the requirement of consistent tie-breaking can be interpreted as a constraint on the message-passing schedule: tie-breaking information for a given tie should be propagated through the graph before other ties are broken.

In order to provide a precise algorithm for the purpose of proving theorems, we consider the ERASURE-QUANTIZE in Table 2 based on applying the rules in (6) with a sequential schedule and all tie-breaking collected into step 8.

Table 2: An algorithm for iteratively quantizing a source with erasures.

```
ERASURE-QUANTIZE($\mathbf{G}$,$\vec{z}$)
 1: while $\vec{z}$ has at least one unerased sample do
 2:    if $\exists$ row $i$ of $\mathbf{G}$ (i.e., a variable) connected to exactly one unerased check $z_j$ then
 3:       Reserve message variable $i$ to later satisfy $z_j$ and erase check $z_j$
 4:    else
 5:       return FAIL
 6:    end if
 7: end while
 8: Arbitrarily set all unreserved message variables
 9: Set reserved variables to satisfy the corresponding checks starting from the last reserved
    variable and working backward to the first reserved variable
10: return message variables $\vec{w}$
```

3.4.3 Iterative Algorithm Duality

Our main results regarding iterative quantization are the following three theorems stating that ERASURE-QUANTIZE works correctly, can be analyzed in the same manner as the ERASURE-DECODE algorithm, and works quickly:

Theorem 3. *For any linear code with generator matrix* $\mathbf{G}$*,* ERASURE-QUANTIZE($\mathbf{G}, \vec{z}$) *either fails in step 5 or else returns* $\vec{w}$ *such that* $\vec{w}\mathbf{G}$ *matches* $\vec{z}$ *in all unerased positions.*

Theorem 4. *Consider a linear code with parity check matrix* $\mathbf{H}$ *and its dual code with generator matrix* $\mathbf{G}^{\perp} = \mathbf{H}$*. The algorithm* ERASURE-DECODE($\mathbf{H}, \vec{y}$) *fails in step 5 if and only if the algorithm* ERASURE-QUANTIZE($\mathbf{G}^{\perp}, \vec{z}$) *fails in step 5 where* $\vec{y}$ *has erasures specified by* $\vec{e}$ *and* $\vec{z}$ *has erasures specified by* $\vec{e}^{\perp} = 1 - \vec{e}$*.*

Theorem 5. *The algorithm* ERASURE-QUANTIZE($\mathbf{G}, \vec{z}$) *runs in time* $\mathcal{O}(n \cdot d)$ *where* n *is the length of* $\vec{z}$ *and* d *is the maximum degree of the graph corresponding to* $\mathbf{G}$*.*

These results (proved in Appendix A) imply that the parallel structure between erasure decoding and erasure quantization allows us to directly apply virtually every result from the analysis of one to the other. For example, these theorems combined with the analysis/design of irregular LDPC codes achieving the capacity of the binary erasure channel [4] immediately yield the following Corollary:

Corollary 2. *There exists a sequence of linear codes which can be efficiently encoded and decoded that achieves the rate-distortion function for binary erasure quantization.*

4 Concluding Remarks

In this paper we demonstrated how codes on sparse graphs combined with iterative decoding can achieve the Shannon limit for binary erasure quantization. The main contribution of our algorithm is in recognizing the role of tie-breaking, scheduling, and initialization in iterative quantization. The key insight in our analysis is the strong dual relationship between error correction and quantization for codes on graphs and their associated decoding/quantization algorithms (both optimal and iterative). We conjecture that the main task in designing iterative message-passing algorithms for more general quantization problems lies in designing appropriate tie-breaking, scheduling, and initialization rules for such scenarios and exploiting similar dual relationships to channel decoding.

A Proofs

Proof of Theorem 1: Consider quantizing a random source and choose some $c > 0$ and let d be the smallest integer such that at least $c \cdot n$ parity checks have degree at most d. For each such parity check, the probability that all variables in the check are not erased is at least $(1-e)^d$. Hence the probability that the check cannot be satisfied is at least $(1/2) \cdot (1-e)^d$. Since there are $c \cdot n$ such checks, the probability that at least one check cannot be satisfied is

$$\begin{aligned}
\Pr[\text{encoding failure}] &\geq 1 - \left[1 - (1/2) \cdot (1-e)^d\right]^{c \cdot n} && (7)\\
&\geq 1 - \left[1 - (1/2 - e/2)^d\right]^{c \cdot n} && (8)\\
&= 1 - \exp\left\{c \cdot n \ln\left[1 - (1/2 - e/2)^d\right]\right\} && (9)\\
&\geq 1 - \exp\left\{-c \cdot n \cdot (1/2 - e/2)^d\right\} && (10)\\
&= 1 - \exp\left\{-\exp\left[\ln c + \ln n + d \ln(1/2 - e/2)\right]\right\}. && (11)
\end{aligned}$$

Hence for the probability of decoding failure to become small, d must grow at least logarithmically with n for every $c > 0$. Note that this argument applies to any parity-check graph representation of the code. □

Proof of Theorem 2: We will show that unique channel decoding is possible if and only if the matrix equation $\mathbf{M}\vec{x} = \vec{y}$ has a solution (where $\mathbf{M}$ will be defined shortly). Similarly, we will show that source quantization is possible for every $\vec{z}$ if and only if the matrix equation $\vec{w}\mathbf{M} = \vec{z}$ has a solution for every $\vec{z}$. By demonstrating that both conditions are satisfied if and only if the same matrix $\mathbf{M}$ has rank n, we will prove the desired result.

Assume that all erasures occur in the last $|\vec{e}|$ positions (*i.e.*, $\vec{e} = 0^{n-|\vec{e}|}\ 1^{|\vec{e}|}$). [6] This incurs no loss of generality since the coordinates of $\mathcal{C}$ can always be permuted accordingly and the theorem applied to the permuted code and its permuted dual code. Let $\vec{x}$ represent the transmitted signal and let $\vec{y}$ denote the received signal. Optimal decoding corresponds to finding a vector which is a codeword of $\mathcal{C}$ and consistent with the unerased received values. The requirement that $\vec{x}$ is a codeword corresponds to the equation $\mathbf{H}\vec{x} = 0$ where $\mathbf{H}$ is the parity check matrix of $\mathcal{C}$. The requirement that $\vec{x}$ is consistent

[6] We use $|\vec{a}|$ to denote the number of non-zero values in $\vec{a}$ (*i.e.*, the weight of $\vec{a}$) and $b^c \triangleq \underbrace{(b\ b \ldots b)}_{\text{c times}}$.

with the received unerased data corresponds to the equation $(\mathbf{I}_{n-|\vec{e}|}\ \mathbf{0})\vec{x} = \vec{y}\,_1^{n-|\vec{e}|}$ where $\mathbf{I}_t$ represents a t-by-t identity matrix and $\vec{y}\,_i^j$ represents the sub-vector $(y_i, y_{i+1}, \ldots, y_j)$. Thus successful decoding is possible if and only if the equation

$$\begin{pmatrix} \mathbf{I}_{n-|\vec{e}|} & \mathbf{0} \\ \multicolumn{2}{c}{\mathbf{H}} \end{pmatrix} \vec{x} = \begin{pmatrix} \vec{y}\,_1^{n-|\vec{e}|} \\ 0 \end{pmatrix} \tag{12}$$

has a unique solution. According to well-known properties of linear algebra, uniqueness is equivalent to the matrix in (12) having full column rank (*i.e.*, rank n). Note that existence of a solution is guaranteed since a codeword was sent and no errors occurred.

Let $\vec{z}$ represent the source to be quantized with erasure pattern $\vec{e}^{\perp}$. Since we assumed that all erasures in $\vec{e}$ occurred in the last $|\vec{e}|$ samples, the dual erasure pattern $\vec{e}^{\perp}$ has all erasures occurring in the first $|\vec{e}^{\perp}|$ positions (*i.e.*, $\vec{e}^{\perp} = 1^{n-|\vec{e}|}\ 0^{|\vec{e}|}$). Optimal decoding corresponds to finding a vector $\vec{w}$ which is a codeword of $\mathcal{C}$ and consistent with the unerased received values. The former requirement corresponds to the equation $\vec{v}\mathbf{G}^{\perp} = \vec{w}$ where $\mathbf{G}^{\perp}$ is the generator matrix of $\mathcal{C}^{\perp}$ and $\vec{v}$ is a binary vector of appropriate dimension. The latter requirement corresponds to the equation $\vec{u}\left(\mathbf{I}_{|\vec{e}^{\perp}|}\ \mathbf{0}\right) + \vec{w} = \vec{z}$ where $\vec{u}$ is a binary vector chosen to ensure that the first $|\vec{e}^{\perp}|$ positions (*i.e.*, the erased positions) match regardless of $\vec{w}$. Thus successful decoding is possible for every $\vec{z}$ if and only if a solution exists for

$$\begin{pmatrix} \vec{u} & \vec{v} \end{pmatrix} \begin{pmatrix} \mathbf{I}_{|\vec{e}^{\perp}|} & \mathbf{0} \\ \multicolumn{2}{c}{\mathbf{G}^{\perp}} \end{pmatrix} = \vec{z} \tag{13}$$

for every $\vec{z}$. According to well-known properties of linear algebra, existence of a solution for every $\vec{z}$ is equivalent to the matrix in (13) having full column rank (*i.e.*, rank n). Note that uniqueness of a solution is neither guaranteed not required since quantization is successful if at least one solution is found.

Noting that $\mathbf{G}^{\perp} = \mathbf{H}$ and $|\vec{e}^{\perp}| = n - |\vec{e}|$ completes the proof since these conditions imply that the matrices in (12) and (13) are identical. □

Proof of Theorem 3: For the algorithm to exit the while loop and reach step 8, every unerased element of z_i must have been erased in step 3 and assigned a reserved message variable. After a variable is reserved all its checks must be erased. Since checks can never changed from erased to unerased, a reserved variable can never again be selected in step 2 and thus a variable can never be reserved more than once.

Thus after the while loop, each unerased position in $\vec{z}$ has a corresponding reserved variable. Hence there exists an assignment of the message variables which result in a codeword matching $\vec{z}$ in the unerased positions. This assignment could be computed via brute-force by solving he corresponding system of linear equations, but in Theorem 5 we show that this step can be computed more efficiently. □

Proof of Theorem 4: The proof relies on the following invariant for steps 1–7 of both algorithms:

$$\forall j,\ y_j \text{ is erased if and only if } z_j \text{ is unerased.} \tag{14}$$

This condition is trivially true before the algorithm begins and forms the base case for a proof by induction. We assume that (14) holds at iteration i of steps 1–7 and show that it must also hold at iteration $i+1$.

First, (14) implies that the outcome of step 1 is the same for each algorithm. Next, since $\mathbf{G}^{\perp} = \mathbf{H}$ the tests in step 1 and step 2 of ERASURE-DECODE($\mathbf{H}, \vec{y}$) and ERASURE-QUANTIZE($\mathbf{G}^{\perp}, \vec{z}$) yield the same result. Finally, at step 3, y_j is unerased while z_j is erased. Therefore, by induction, condition (14) is true at every iteration and ERASURE-DECODE($\mathbf{H}, \vec{y}$) fails at step 5 if and only if ERASURE-QUANTIZE($\mathbf{G}^{\perp}, \vec{z}$) fails at step 5. □

Proof of Theorem 5: The while loop executes at most n times. Therefore step 1 requires at most $\mathcal{O}(n)$ operations. Consider storing the number of variables with exactly one unerased check in a data structure which supports insertion and removal in constant time (*e.g.*, a hash table). We can initialize the data structure with $\mathcal{O}(d \cdot n)$ operations. Removing an element in steps 2 and 3 and updating the data structure to account for step 3 requires $\mathcal{O}(d)$ operations. Thus steps 1 through 8 require $\mathcal{O}(d \cdot n)$ operations and all that remains is to bound the running time of step 9.

Denote the first reserved variable by $v_{j(1)}$, the second reserved variable by $v_{j(2)}$ and so on to $v_{j(|e|)}$. As described in step 9, we first assign a value to $v_{j(|e|)}$ and work backward. Specifically, we set $v_{j(i)}$ to the modulo-2 sum of $z_{j(i)}$ and all message variables connected to $z_{j(i)}$ (except $v_{j(i)}$). This is possible for $z_{j(|e|)}$ since no other reserved variable could be connected to $z_{j(|e|)}$.[7] Similarly, $z_{j(|e|-1)}$ must be connected to only unreserved variables as well as perhaps to $v_{j(|e|)}$ and therefore a value can be determined for $v_{j(|e|-1)}$. Thus, by induction we can determine every v_j.

Adding up the operations computed for each step yields a running time of $\mathcal{O}(d \cdot n)$. □

Acknowledgment

The authors wish to thank G. D. Forney, Jr. for many helpful comments on the manuscript.

References

[1] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Transactions on Information Theory*, vol. 44, pp. 2325–2383, October 1998.

[2] M. W. Marcellin and T. R. Fischer, "Trellis coded quantization of memoryless and Gauss-Markov sources," *IEEE Transactions on Communications*, vol. 38, pp. 82–93, January 1990.

[3] G. D. Forney, Jr., "Codes on graphs: normal realizations," *IEEE Transactions on Information Theory*, vol. 47, pp. 520–548, Feb 2001.

[4] P. Oswald and A. Shokrollahi, "Capacity-achieving sequences for the erasure channel," *IEEE Transactions on Information Theory*, vol. 48, pp. 3017–3028, December 2002.

[5] A. Braunstein, M. Mezard, M. Weight, and R. Zecchina, "Constraint satisfaction by survey propagation." `arXiv:cond-mat/0212451` (http://arXiv.org).

[7] If $z_{j(|e|)}$ was connected to another reserved variable v, that would imply v was reserved when connected to $z_{j(|e|)}$ which was unerased as well as z which must also have been unerased. This contradicts step 2 in ERASURE-QUANTIZE.