

Advances in Iterative Decoding and Maximum Likelihood Decoding for the Packet Network with Comparisons to Fountain Codes over the Erasure Channel

J. Cai, M. Tomlinson, C. Tjhai, M. Ambroze, and M. Ahmed
Fixed and Mobile Communications Research
University of Plymouth
PL4 8AA, United Kingdom
email: {jcai,ctjhai,mtomlinson,mambroze,mahmed}@plymouth.ac.uk

Abstract

In this paper, we propose a novel ML decoding algorithm – *the In-place Algorithm* in conjunction with a *Product Packetisation* method for the congested Internet Network modeled as a Binary Erasure Channel (BEC). Any code can be used with this algorithm and we give results for cyclic codes constructed from BCH codes. Existing codes and decoding algorithms are compared in terms of performance and decoding complexity. It is shown that a significant performance improvement can be achieved. In general, ML decoding on the Erasure channel has a complexity of $O(N^3)$ or $O(N^2)$ depending on the algorithm. It is shown that this complexity can be reduced to $O(N^{1.5})$ for the network channel by using the product packetisation method. With an analysis in performance, ratelessness can be achieved by acks.

1 Introduction

Network transmission is based on packet transmission. Multicast and broadcast are typical examples of packet-based communications. One simple way to get around packet loss is to employ a protocol in which receiving parties acknowledge received packets. This solution requires multiple rounds of communications which is inappropriate for many situations, especially in multicasting and broadcasting. In those cases, we must limit the amount of feedback to the senders and the number of redundant packets sent to receivers. Erasure codes rely on error correcting principle which can recover the lost packets at the receiver without the need for retransmission.

A model for the Internet transmission is the Binary Erasure Channel (BEC). This channel was introduced by Elias [2] in 1955. A packet is lost due to the network congestion with probability p , and then the BEC has the capacity $1 - p$. Elias proved that there exist codes of rate R for any $R < 1 - p$ that can be used to transmit over channels of capacity $1 - p$. For an erasure channel a rate R code encodes a message of size K into a transmission of $K/R = N$, so that the original message can be recovered from erased positions of the N bit message.

In a milestone paper, Luby *et. al.* [3] proposed the first realization of a class of erasure codes – LT codes, which are termed rateless and are generated on the fly as needed. In [4], Shokrollahi introduced the idea of Raptor codes which adds an outer code to LT codes. Raptor codes have been established in order to solve

the error floors exhibited by the LT codes.

In this paper we consider stand alone BCH codes instead of LT codes, as they do not show an error floor. Low-density parity-check (LDPC) codes have been studied [1] to [9] for application to the BEC. Gallager's soft-decoding algorithm [10], was implemented [1] for LDPC codes over the BEC. Capacity-achieving degree distribution codes for the binary erasure channel have been introduced in [1], [7] and [8]. Finite-length analysis was used to find better LDPC codes for the BEC [9]. For ML decoding on the BEC, three families of algorithms have been proposed: structured Gaussian elimination [12], whose computational complexity is $O(N^3)$, the conjugate gradient and Lanczos algorithms [12] and [13], both of which require about $O(N^2)$ operations, and the Wiedemann [14] algorithm with $O(N^2)$. The Lanczos algorithm incorporating structured Gaussian elimination was proposed in [15].

The paper is organized as follows. Starting with a superposition of the erased bits on the parity-check matrix in section 2.1, we review the performance of the iterative decoding algorithms, for the BEC, in section 2.2, principally the Recovery Algorithm and the Guess Algorithm and an improved algorithm [6]– the Multi-Guess Algorithm. In section 2.3, we also describe a Maximum Likelihood decoding algorithm, the In-place Algorithm, which is based upon the Gaussian Elimination Algorithm [12]. In section 3, we propose a new packetisation format – A Product Packetisation, which provides the basis for a more efficient and quicker decoder. Using the In-place algorithm, the computational complexity is $O(N^{1.5})$. The conclusions are given in

2 Preliminaries

2.1 Matrix Representation of erased bits

We use the parity check matrix H to represent the code. Considering an (N, K) binary linear block code, we denote a codeword as $\mathbf{x} = \{x_1, x_2, \dots, x_N\}$. After being transmitted over the erasure channel with erasure probability ϵ , the received vector can be divided into a transmitted sub-sequence and an erased sub-sequence, denoted as $\mathbf{y} = \{y_1, y_2, \dots, y_{L_\epsilon}\}$ and $\mathbf{y}_\epsilon = \{y_{\epsilon 1}, y_{\epsilon 2}, \dots, y_{\epsilon L_\epsilon}\}$ respectively, where $L_\epsilon + L_\epsilon = N$.

Corresponding to the parity check matrix of the code, we can generate an erasure matrix M_ϵ ($L_\epsilon \times N$) which contains the positions of the erased bits in H . Then we denote the set of erased bits i that participate in each parity check row by $E_i^h = \{j : M_{\epsilon(ij)} = 1\}$ with h standing for “horizontal” and the number of erased bits in E_i^h is denoted by $|E_i^h|$. Similarly we define the set of checks in which bit j participates, $E_j^v = \{i : M_{\epsilon(ij)} = 1\}$ with v standing for “vertical”, and the number of erased bits in E_j^v is denoted by $|E_j^v|$. Let $\mathbf{E}^h = \{E_i^h \mid i \in \{1, 2, \dots, L_\epsilon\}\}$ and $\mathbf{E}^v = \{E_j^v \mid j \in \{1, 2, \dots, N\}\}$. The matrix representation is shown in Fig. 1, where an “x” represents an erasure.

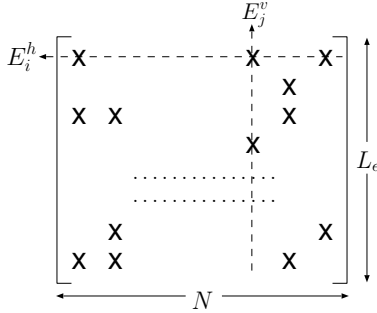


Fig. 1. A matrix representation (M_ϵ) of the erased bits

2.2 Review of Iterative Decoding Algorithms

In [1], the Recovery Algorithm is described, which is also called message-passing algorithm, and is equivalent to Gallager’s soft-decoding algorithm, an iterative decoding algorithm. The decoding complexity can be shown to be $O(N^2)$. The algorithm is briefly outlined below:

Recovery Algorithm

- Step 1* Generate the M_ϵ and obtain the $\mathbf{E}^h$.
- Step 2* For $i \in \{1, 2, \dots, L_\epsilon\}$, if $|E_i^h| = 1$, we replace the value in the bit position i with the XOR of the unerased bits in that check equation. Then we remove the erasure from the erasure matrix.

- Step 3* Continue from step 2 until all the erased bits are solved or the decoding cannot continue further.

If there exists erasures in stopping sets, the Recovery Algorithm stalls. In this case, an algorithm [11] with the term Guess Algorithm is applied by performing several “guesses” of the unsolved erased bits, which is similar to the Maxwell decoder [19] which works as a message passing decoder.

Guess Algorithm

- Step 1* Run the decoder with Recovery Algorithm until it fails due to stopping set(s).
- Step 2* In order to break the stopping set, when $|E_i^h| = 2$, we guess one of the erased symbols and update the erasure matrix M_ϵ and $\mathbf{E}^h$.
- Step 3* Continue from step 1 until all the erased symbols are solved or the decoding cannot continue further. If the decoder cannot continue, declare a decoder failure and exit.
- Step 4* Create a list of 2^g solutions, where g is the number of guesses made. From the list c_{out_k} , $k \in \{1, 2, \dots, 2^g\}$, pick the one that satisfies $Hc_{\text{out}_k}^T = \mathbf{0}$.

Obviously, compared to the Recovery Algorithm, the complexity of this algorithm increases with g . Usually, we limit the number of guesses to a small number g_s . If after g_s guesses, the decoding still cannot be finished, a decoding failure is declared.

In [6], we proposed an improved iterative decoding algorithm, which is called “Multi-Guess Algorithm”. For sparse codes, the Multi-Guess Algorithm can approach optimal performance with significant improvements. Consider the computational complexity, we calculate the minimum number of guesses, denoted by $\min(g)$ before the decoding. The calculation lemma was given in [6]. For the Multi-Guess Algorithm, a whole row is guessed. A crucial row c is defined as follows:

- 1) $c \in \omega_\delta$, where ω_δ is the set of all equations with $|E_i^h| = \delta$ where $i \in \{1, 2, \dots, L_\epsilon\}$, and $\delta =$ number of unknown positions.
- 2) $\sum_{j \in E_c^h} |E_j^v|$ is maximized over c in ω_δ

The Multi-Guess Algorithm is given below:

Multi-Guess Algorithm

- Step 1* Run the decoder with Guess Algorithm until $|E_i^h| > 2$ for $i = 1, \dots, L_\epsilon$.
- Step 2* Evaluate the value of $\min(g)$. If $\min(g) > g_s$, the decoding declares a failure and exits.
- Step 3* Group the rows as ω_δ .
- Step 4* Find the row which is on the basis of the highest value of $|E_j^v|$ with the value of $|E_i^h| = \delta$ and guess all erased bits in that row. (There will be at most $2^{\delta-1}$ guesses.)
- Step 5* Guess one bit p with $|E_p^v| = 1$ in each of the independent rows.
- Step 6* Update M_ϵ , $\mathbf{E}^h$ and $\mathbf{E}^v$. Continue the decoding

from step 3 to step 5 until all the erased bits are solved or the decoding cannot continue further.

Obviously, the decoding complexity can grow exponentially with the number of guesses. In practice, improved performance is achieved with $O(N^2)$ complexity. With higher complexity, there is a better tradeoff with respect to performance by using the ML decoding algorithm.

2.3 Optimal ML Decoding Algorithm

Let $\mathbf{x}'$ denote the received vector, where $\mathbf{x}' = \mathbf{y} \cup \mathbf{y}_\epsilon$. We now devise a reduced complexity algorithm to decode the erased bits by solving the equation 1 using the Gaussian Reduction method [12].

$$H\mathbf{x}'^T = \mathbf{0}. \quad (1)$$

According to [9], optimal decoding is equivalent to solving the linear system, shown in equation (1). If the equation (1) has a unique solution, an optimal algorithm is possible. Gaussian Reduction algorithm is considered as an optimal algorithm over the BEC, but has a complexity of $O(N^3)$. We propose a reduced complexity Gaussian Reduction algorithm – In-place Algorithm [5] by eliminating the column-permutations required. This algorithm is stated as follows:

In-place Algorithm

Step 1 The codeword is received and y_ϵ are substituted in positions of erased bits in H . Starting with one of the erased symbols, y_{ϵ_s} , the first equation containing this symbol is flagged that it will be used for the solution of y_{ϵ_s} . This equation is subtracted from all other equations containing y_{ϵ_s} and not yet flagged to produce a new set of equations. The procedure repeats until either non flagged equations remain containing y_{ϵ_s} (in which case a decoder failure is declared) or no erased symbols remain that are not in flagged equations.

Step 2 Let $y_{\epsilon_{\text{last}}}$ be the erased symbols at the last flagged equations. In the latter case, starting with $y_{\epsilon_{\text{last}}}$ this equation is solved to find y_{last} and this equation is unflagged. This coefficient is substituted back into the remaining flagged equations containing y_{last} . The procedure now repeats with the second from last flagged equation now being solved for $y_{\epsilon_{\text{last}-1}}$. This equation is unflagged and followed by back substitution of $y_{\text{last}-1}$ for $y_{\epsilon_{\text{last}-1}}$ in the remaining flagged equations.

Compared with the Gaussian Elimination Algorithm, the In-place Algorithm complexity remains $O(N^3)$, but the multiplicative constant is significantly reduced. The memory required for storing swapped positions is eliminated by only performing flag and addition, which means we do not have to convert the M_ϵ matrix to a leading diagonal matrix. The least complexity ML algorithm [18] which requires $O(N^{2.376})$ in complexity

is not guaranteed to produce a solution and incurs considerable overheads in implementation.

3 Product Packetisation

In this section, we describe a product packetisation structure. In a traditional way, files sent over the Internet are partitioned into packets, as in the applications of LT codes and Raptor codes. Each packet is either received without error or not in which case it is erased. Standard file-transfer protocols simply partition a file into n packets, then repeatedly transmit each packet until all of the packets have been received.

The term Product Packetisation is used to describe the arranging of the encoded bits in a format similar to a product code. The encoded bits are written into n columns and read out in rows for transmission, as illustrated in Fig. 2. With this packetisation structure, short, powerful algebraic codes can be used for Internet transmission.

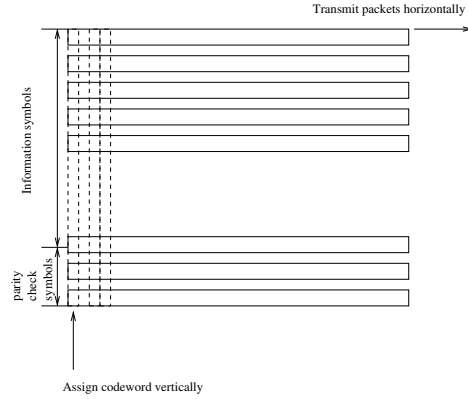


Fig. 2. A Product Packetisation Structure

We designed the structure as a square matrix of size $n \times n = N$, where N is the size of the encoded sequence. Although the structure is flexible, there is a complexity/performance compromise which depends on code block size. We have found that, in this case, choosing a square matrix structure gives a good complexity/performance compromise. In Fig. 3, the shaded area represents a packet erasure. In the decoding process, each column is decoded separately. As the erasures are in the same positions, for all codewords, the In-place Algorithm has to be applied only to the first column. For each of the other $n - 1$ columns, the same erasure correcting equations are used, but with the coordinates provided by that column. This reduces the complexity of the ML decoder to $O(n^3) = O(N^{3/2})$ since $n \doteq \sqrt{N}$. This coding scheme can be made rateless by acknowledging successful decoding once enough packets have been received. Due to the powerful error correction codes used, not all the packets are necessary for successful decoding. Compared with fountain codes using the message-passing algorithm

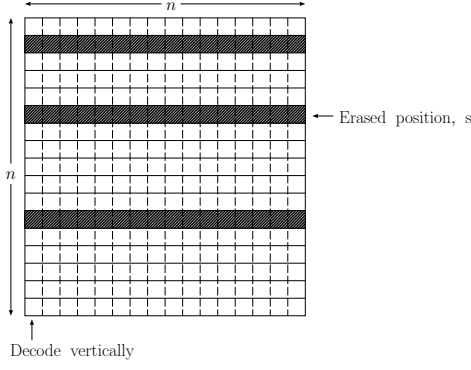


Fig. 3. An example of received packets

with complexity $O(N^2)$, the new algorithm is not only able to correct more erasures but is also faster and more efficient.

4 Simulation Results

We have evaluated the performance of LT codes and cyclic codes for $N = 225$ and $N = 3969$. For $N = 225$, the LT code with the recovery decoder produces very poor performance. In this case, we have used the ML decoder. The LT code (225, 105) was divided into 15 packets with the packet-size of 15 bits and the cyclic code (225, 105) was arranged in the product packetisation format with the same packet-size¹. The cyclic code is capable of correct decoding up to and including 4 packets erased. As $1+x^{15}$ divides $1+x^{225}$, the decoder for this cyclic code can be implemented with the reduced complexity decoder. As shown in Fig 4, the performance of the cyclic code (225, 105) is significantly better than that of the LT (225, 105) code. When $p = 0.1$, the result of the cyclic code (225, 105) is over 2 orders of magnitude better than that of the LT (225, 105).

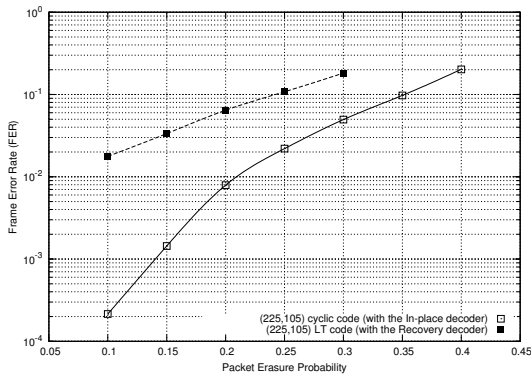


Fig. 4. FER vs the erasure probability of erased packets for the cyclic code (225, 105) and the LT (225, 105) code over the Erasure Channel, both using ML decoding

¹The cyclic code (225, 105) has a generator polynomial $g(x) = 1 + x^{15} + x^{30} + x^{60} + x^{120}$, which is a replication of the BCH (15, 7, 5) code with $g(x) = 1 + x + x^2 + x^4 + x^8$.

To verify the decoding complexity calculation, we implemented a longer code, the cyclic code (3969, 2016) and the LT (3969, 2016) code. The cyclic code² (3969, 2016) is based upon the generator polynomial of the BCH (63, 32, 12) code. The simulated performance is shown in Fig. 5. On a standard 1GHz PC, the LT (3969, 2016) code took 775ms to encode each codeword and 141ms to decode each codeword for $p = 0.4$ (using the Recovery Decoder). In contrast, the cyclic code (3969, 2016) took only 5ms to encode each codeword and 8ms to decode each codeword using the In-place decoder for $p = 0.4$ (the decoding time for this decoder is independent of the value of p).

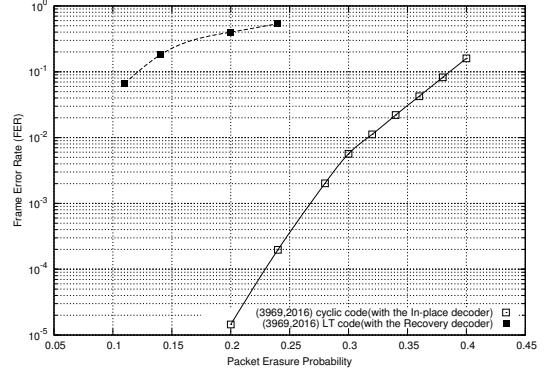


Fig. 5. FER vs the erasure probability of erased packets for the cyclic code (3969, 2016) and the LT (3969, 2016) code over the Erasure Channel

5 Conclusions

In this paper, we presented a novel coding and decoding algorithm for packet networks modelled as a Binary Erasure Channel (BEC). The algorithm combines a ML decoder with a product packetisation structure. In the past iterative decoding algorithms, although non-optimum, have been used as a good tradeoff between performance and computational complexity. With the new decoding algorithm, decoding is ML and the computational complexity is reduced to $O(N^{1.5})$. This is less complexity than the case for iterative decoders. Also, compared with LT codes, the new decoding algorithm with algebraic codes achieves a significant improvement in performance.

References

- [1] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 569 – 584, Feb. 2001.
- [2] P. Elias, "Coding for two noisy channels," in *Proc. 3rd London Symp. Information Theory*, 1955, pp.61 – 76.

²The cyclic code (3969, 2016) has a generator polynomial $g(x) = 1 + x^{63} + x^{441} + x^{567} + x^{756} + x^{882} + x^{945} + x^{1008} + x^{1134} + x^{1197} + x^{1260} + x^{1323} + x^{1638} + x^{1764} + x^{1890} + x^{1953}$

- [3] M. Luby, "LT-Code," *Proceeding of the 43rd Annual IEEE Symposium on the Foundations of Computer Science*, pp. 271 – 280, 2002.
- [4] M. A. Shokrollahi, S. Lassen and M. Luby, "Multi-stage code generator and decoder for communication systems," U.S. Patent application number 20030058958, Dec. 2001.
- [5] M. Tomlinson, J. Cai, Cen Tjhai, M. Ambroze and M. Z. Ahmed, "System for correcting deleted or missing symbols," U. K. Patent application number 0428042.6, Dec. 2004.
- [6] J. Cai, Cen Tjhai, M. Tomlinson, M. Ambroze and M. Z. Ahmed, "An Efficient Solution to Packet Loss : Erasure Correcting Codes," in *The Fourth IASTED International Conference CSN*, 2005, pp.224 – 229.
- [7] M. A. Shokrollahi, "Capacity-achieving sequences," *IMA Volumes in Mathematics and its Applications*, vol.123, pp.153 – 166, 2000.
- [8] P. Oswald and M. A. Shokrollahi, "Capacity-achieving sequences for the erasure channel," *IEEE Trans. Inform. Theory*, vol. 48, pp.3019 – 3028, Dec. 2002.
- [9] C. Di, D. Proietti, I. E. Telatar, T. Richardson, and R. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Tran. Inform. Theory*, vol. 48, pp. 1570 – 1579, June 2002.
- [10] R. G. Gallager, "Low Density Parity Check Codes," Cambridge, MA: MIT Press, 1963.
- [11] H. Pishro-Nik and Faramarz Fekri, "On decoding of low-density parity-check codes over the binary erasure channel," *IEEE Trans. Inform. Theory*, vol. 20, pp. 439 – 454, March 2004.
- [12] A. M. Odlyzko, "Discrete logarithms in finite fields and their cryptographic significance," in *Advances in Cryptology: Proceeding of Eurocrypt'84*, T. Beth, N. Cot, and I. Ingemarssen, Eds. Berlin, Germany: Springer-Verlag, 1985, vol. 209, pp. 224 – 314.
- [13] D. Coppersmith, A. Odlyzko and R. Schroepfel, "Discrete logarithms in $GF(p)$," *Algorithmica*, vol. 1, pp 1–15, 1986.
- [14] D. Wiedemann, "Solving sparse linear equations over finite fields", *IEEE Trans. Inform. Theory*, vol IT-27, pp. 54–62, Jan. 1986
- [15] P. L. Montgomery, "A block Lanczos algorithm for finding dependencies over $GF(2)$," in *Advances in Cryptology : Proceeding of Eurocrypt'95*, L. Cl Guillou and J. J. Quisquater, eds, vol 921, pp. 106–120, Springer-verlag, 1995.
- [16] W. W. Peterson, "Error Correcting Codes," The MIT Press 1961.
- [17] F. J. MacWilliams and N. J. A. Sloane, "The Theory of Error Correcting Codes," North-Holland 1977.
- [18] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions", in *Proc. 19th ACM Symp. Theory of Computing*, 1987, pp. 1–6.
- [19] C. Measson, R. Urbanke, A. Montanari and T. Richardson, "Life Above Threshold: List Decoding to Area Theorem and MSE", in the ITW 2004, San Antonio, Texas, USA, Oct. 2004.