LT codes on Partial Erasure Channels

Carolyn Mayer
Department of Mathematics
University of Nebraska - Lincoln
Lincoln, Nebraska 68588–0130
Email: cmayer@huskers.unl.edu

Christine A. Kelley
Department of Mathematics
University of Nebraska - Lincoln
Lincoln, Nebraska 68588–0130
Email: ckelley2@math.unl.edu

***Abstract*—LT codes are a class of rateless codes designed for data dissemination on erasure channels. In this paper, we present a decoder for LT codes on partial erasure channels, which were recently introduced for multi-level read storage channel applications. We compare the efficiency of LT codes on these channels to those on the q-ary Erasure Channel (QEC).**

I. Introduction

Luby Transform (LT) codes [1] are a family of universal fountain codes designed to transmit data on channels with erasures. Packets that are functions of the original file are generated randomly with the goal that as few packets of data are transmitted as necessary so that the receiver can recover all of the data. An extension of LT codes, Raptor codes were introduced in [2], [3] and have linear time encoding and decoding. In Raptor codes, input symbols are precoded prior to being encoded by an LT code. Researchers have since extended the ideas of LT and Raptor codes to other channels, such as the Binary Symmetric Channel (BSC) and binary Additive White Gaussian Noise (AWGN) Channel [3]–[5].

Motivated by applications in non-volatile memory multi-level read channels, the q-ary Partial Erasure Channel (QPEC) was introduced in [6], and analysis of iterative decoding of Low-Density Parity-Check (LDPC) codes on the QPEC was presented. In this channel, an erased symbol is known to belong to a set of M possible symbols, thus some partial information is available. More recently, the Multilevel Erasure Channel (MEC) and q-ary Bit-Measurement Channel (QBMC) were introduced to model applications in which partial erasures of symbols over a $q = 2^k$-ary alphabet correspond to erasures at the bit level in the binary representation of the symbol [7], [8]. Multistage decoding of LDPC codes on the MEC and QPEC is also addressed in [7].

In this paper, we look at LT codes on these partial erasure channels. Assuming a similar encoding strategy, one could simply disregard partially erased symbols (i.e. "packets") and generate more packets until enough are available to retrieve the entire file. However, we propose a code design and decoder that allow for some partial erasure correction to occur, thereby limiting the need to generate unnecessary symbols. The decoder combines the ease of the LT decoder with a second layer of belief propagation decoding.

This paper is organized as follows. Background on the QPEC, MEC, and QBMC is given in Section 2. In Section 3 we present a two-phase decoding algorithm for LT codes on these channels. In Section 4 we obtain the expected number of symbols required to be generated for these LT codes. Section 5 presents density evolution analysis of the LT decoder on the QPEC.

II. Preliminaries

Let $q = p^k$ for some prime p. We consider codes over $GF(q)$. In a standard q-ary Erasure Channel (QEC), a q-ary input symbol will either be received without error (with probability $1-\varepsilon$) or erased (with probability ε). When a symbol is erased, it is known that an erasure has occured. The QEC with $q = 2$ is known as the Binary Erasure Channel (BEC). Partial erasure channels are generalizations of erasure channels in which not all information is removed during an erasure event. After a partial erasure, sets of possible symbols remain.

A. Partial Erasure Channels

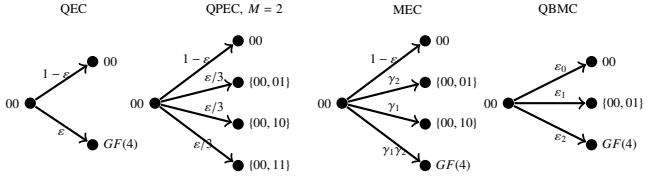

Fig. 1. Transition probabilities seen by $x = 00 \in GF(4)$ for the QEC, QPEC with $M = 2$, MEC, and QBMC.

The QPEC [9] with (partial) erasure probability ε is a generalization of the QEC in which, given a q-ary input, the QPEC outputs either the original input (with probability $1-\varepsilon$) or a partial erasure (with probability ε). A partial erasure is a set containing the original input as well as $M-1$ other q-ary symbols. Each of the $\binom{q-1}{M-1}$ possible erasures for a symbol occur with equal probability.

The MEC [7] is a partial erasure channel in which erasure events correspond to one or more bit erasures in the binary representation of a 2^k-ary symbol. The i-th bit has erasure probability γ_i for $i = 1, \ldots, k$. The probability of a (partial or full) erasure event occuring is

$$\varepsilon = \sum_{B \subseteq [k]} \left(\prod_{i \in B} \gamma_i \prod_{i \notin B} (1 - \gamma_i) \right).$$

The definition of the MEC may be extended to p^k-ary symbols. However, we will focus on 2^k-ary symbols in this paper. When

$\gamma_1 = \gamma_2 = \cdots = \gamma_k = \gamma$, we call the channel the *constrained* MEC. For the constrained MEC, $\varepsilon = 1 - (1-\gamma)^k$.

The QBMC [8] has erasure events corresponding to bursts of erasures starting with the rightmost bit in the binary representation of a 2^k-ary symbol. A burst of length i occurs with probability ε_i for $i = 0, \ldots, k$. The probability of a (partial or full) erasure event occuring is $1 - \varepsilon_0 = \sum_{i=1}^{k} \varepsilon_i$. Figure 1 shows a comparison of the QEC, QPEC with $M = 2$, MEC, and QBMC for $q = 4$.

B. LT Encoding Process

As in [1], [10], encoding symbols are generated as follows:

1) Choose a degree d independently for each encoding symbol according to a given degree distribution.
2) Randomly choose d of the k information symbols. For LT codes on the QPEC with $q = p^k$, take the sum of the chosen symbols $\pmod p$. For LT codes on the MEC and QBMC with $q = 2^k$, take the bitwise sum of the chosen symbols.

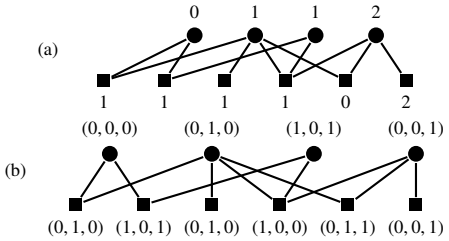

Fig. 2. Encoding graphs (a) on the QPEC with $q = 3$ and $M = 2$ and (b) on the MEC or QBMC with $q = 8$.

Figure 2 shows possible encoding graphs on the QPEC and on the MEC or QBMC with degree distribution $p(1) = \frac{1}{3}$, $p(2) = \frac{1}{2}$, and $p(3) = \frac{1}{6}$. Circular nodes represent message symbols and square nodes represent encoding symbols. The degree distribution used in the encoding process greatly influences the performance and efficiency of the LT decoder [1].

III. LT Decoding Process

We now introduce a two-phase decoding process for LT codes on partial erasure channels. "Phase I" mirrors the original LT process on the QEC [1], and "Phase II" allows for further decoding when the Phase I algorithm gets stuck.

Decoding: Phase I

1) Each message node starts in an *uncovered* state.
2) At the first iteration, each degree one encoding node is *released* to *cover* its neighboring message node (i.e. the message node is assigned the value or intersection of sets of values associated with the released encoding node neighbors). These encoding nodes are removed from the decoding graph.
3) The set of covered message nodes not yet *processed* forms a *ripple*.
4) If a message node in the ripple has an associated set of cardinality one, then the message symbol represented by the node is the value of the symbol in the set. Randomly select a message node in the ripple with an associated set of cardinality one, if such a node exists. *Process* the selected message node by subtracting its associated value from each of its neighboring encoding nodes. Remove the message node and its incident edges.
5) Release any degree one encoding nodes. Continue processing nodes in the ripple with an associated set of cardinality one and releasing degree one encoding nodes.

An example of Phase I decoding for the QPEC is shown in Figure 3. If all message symbols have been recovered at the end of Phase I, decoding is complete. If there are still message symbols to be recovered, note that there are two possibilities: either the ripple is empty and a decoder failure results, or the ripple is non-empty, all message nodes in the ripple are covered by sets of size greater than 1, and all of the encoding nodes remaining have degree greater than 1. Note that the first case (an empty ripple) is what characterizes decoder failure for LT codes on the QEC.

We will use $\mathcal{N}(e)$ to denote the neighborhood of an encoding node e, and $\mathcal{N}_R(e)$ to denote the subset of $\mathcal{N}(e)$ contained in the ripple. Let E_R be the set of encoding nodes with $\mathcal{N}(e) = \mathcal{N}_R(e)$; that is, E_R contains all of the encoding nodes that have all of their neighbors in the ripple. Denote by I_m (resp., I_e), the set of symbols associated with message node m (resp., encoding node e). For sets U and V, their sum is the set $\{u + v : u \in U \text{ and } v \in V\}$, and their difference is defined similarly.

Decoding: Phase II

1) For each encoding node $e \in E_R$, send I_m along each incident edge $x_m x_e$ for each $m \in \mathcal{N}(e)$.
2) For each edge $x_m x_e$ in the previous step return $I_e - \sum_{m' \in (\mathcal{N}(e) \setminus m)} I_{m'}$ along $x_e x_m$.
3) Update I_m to be
$$I_m \cap \bigcap_{e \in E_R} \left(I_e - \sum_{m' \in (\mathcal{N}(e) \setminus m)} I_{m'} \right).$$
4) If $|I_m| = 1$ for any m, process x_m and return to Phase I.

Recall that in each partial erasure channel model, the transmitted symbol must be contained in the set of symbols associated with a partially erased node. That is, I_e contains the original sum of the values of the neighboring symbols of e. If a message node m is processed, then the single element in I_m associated with the message node is the correct information symbol represented by that node. Note that at the start of decoding, $|I_e| > 1$ only if the encoding symbol e is partially erased, and $|I_e|$ may increase or decrease as decoding iterations proceed. Moreover, $I_m = \emptyset$ at the start of decoding for each message node m, and once $I_m \neq \emptyset$, the size $|I_m|$ decreases monotonically with increasing decoding iterations.

When all message nodes are processed (i.e. message symbols recovered), decoding is complete. Recall that in the LT process [1], there is a one-to-one correspondence between an encoding symbol covering a message symbol and recovering a message symbol. However, this correspondence no longer holds on the QPEC, MEC, and QBMC.

Example III.1. *LT Decoding on the QPEC*

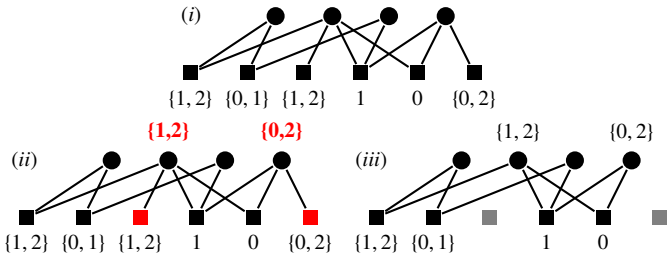

Fig. 3. Phase I decoding on the QPEC ($q = 3, M = 2$), starting with a possible decoding graph (i) for the encoding graph in Figure 2(a). After Step (iii), Phase II begins.

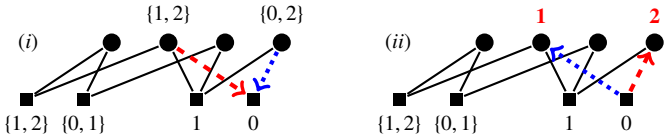

Fig. 4. Phase II decoding following the Phase I decoding in Fig. 3. After Step (ii), the decoder returns to Phase I.

Fig. 3 shows Phase I decoding on the QPEC with $q = 3$ and $M = 2$, and Fig. 4 shows an example of the Phase II decoding process after Phase I decoding ends in Step (iii) of Fig. 3. In Step (i) (resp., Step (ii)) of Fig. 4, $\{1,2\}$ (resp., $0-\{1,2\} = \{1,2\}$) is sent along the dashed edge. At the fourth message node, $\{1,2\}$ is intersected with $\{0,2\}$, and the symbol 2 is recovered. Simultaneously, in Step (i) (resp., Step (ii)) of Fig. 4 , $\{0,2\}$ (resp., $\{0,1\}$) is sent along the dotted edge, leading to the recovery of the symbol 1 at the second message node. Decoding now returns to Phase I as in Figure 5. Note that decoding would be unsuccessful if the second encoding node had started with the partial erasure $\{1,2\}$ instead of $\{0,1\}$.

For the MEC and QBMC, if bitwise sums are taken during the encoding process, then we may decode bits separately. To do this, create k copies of the decoding graph, $G_1, G_2, \ldots, G_k$. In G_i, discard message symbols in which bit i was erased. The usual LT decoding algorithm may be applied to each graph. Figure 6 shows a possible overall decoding graph on the MEC along with the corresponding component graph for its first bit.

Proposition III.2. *On the MEC or QBMC, the probability of decoding success is the same using the single overall graph with two-phase decoder or with component graphs each with the standard LT decoder.*

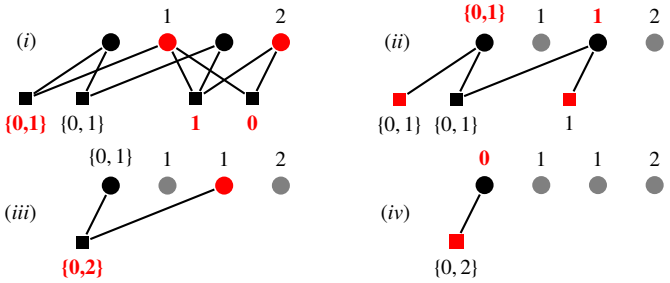

Fig. 5. Successful Phase I decoding following Phase II decoding in Fig. 4.

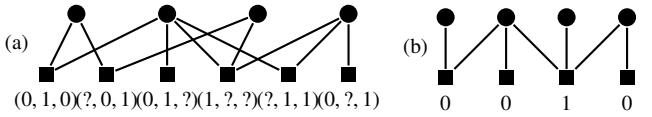

Fig. 6. (a) A possible decoding graph on the MEC from the encoding graph in Figure 2(b), and in (b) the corresponding decoding graph for the first bit.

Proof. If the graph starts with no degree one encoding nodes, then failure occurs both when using a single graph and when using the multiple graphs. Otherwise, failure using the single decoding graph occurs if: (i) in Phase I, the ripple empties before each message symbol is recovered, or (ii) in Phase II, no message node cardinality is reduced to one after any number of iterations. In case (i), message nodes previously in the ripple are successfully recovered and processed, but no more encoding nodes are reduced to cardinality one. The degrees of the encoding nodes are the same in the individual bit graphs of the multiple graph case. After the same message symbols are recovered in the multiple graph case, no encoding node will have low enough degree to be released. In case (ii), some bits are not recovered for message nodes in the ripple. Thus, for at least one of the individual bit graphs, a message node from the ripple in the single graph case will either not reach the ripple (if adjacent encoding symbols are erased), or its processing will not reduce an encoding node to degree one. Thus, if a failure occurs on the single graph, then a failure would occur if individual component decoding graphs were used instead. The other direction is similar. □

IV. Number of Encoding Symbols

The number of symbols that must be received for a given allowable failure probability δ of the decoder depends on the degree distribution chosen for the encoding symbols. For n message symbols on the QEC, the All-At-Once distribution requires $n \cdot \ln(n/\delta)$ encoding symbols, and the Robust Soliton distribution requires $n + O(\ln^2(n/\delta)\sqrt{n})$ encoding symbols [1]. The design of LT degree distributions was studied in [11].

If each component has failure probability at most δ when decoding componentwise on the MEC, then the overall decoding process has failure probability at most $1-(1-\delta)^k$. In order to have overall probability of failure at most δ_{total}, we may set $\delta \le 1-(1-\delta_{total})^{\frac{1}{k}}$ and determine how many symbols must be received by each component for the given degree distribution.

Proposition IV.1. *For the MEC with $\gamma_i = \gamma$ for $i = 1, \ldots, k$, an average of at most*

$$\sum_{t=N}^{\infty} t \cdot (1-\gamma)^{tk} \left[\left(\sum_{j=0}^{t-N} \binom{t}{j} \frac{\gamma^j}{(1-\gamma)^j} \right)^k - \left(\sum_{j=0}^{t-N-1} \binom{t-1}{j} \frac{\gamma^j}{(1-\gamma)^{j+1}} \right)^k \right]$$

symbols are required to be transmitted in order to decode successfully using component decoding graphs, where N is the number of encoding symbols required to be received by each component for the specified LT code degree distribution.

γ	.005	.1	.25	.45
$E(Z)$	3.0590	3.9506	5.2425	7.7521
$N/(1-\gamma)^k$	3.0608	4.5725	9.4815	32.7846

TABLE I
$E(Z)$ ON THE CONSTRAINED MEC AND $\frac{N}{(1-\gamma)^k}$ FOR $N = 3, k = 4$, AND VARIOUS γ.

Proof. Let X_i be the number of symbols needed for channel i, and let $Z = \max(X_1, \ldots, X_k)$. Then

$$E(Z) = \sum_{t=N}^{\infty} tP(Z = t) = \sum_{t=N}^{\infty} t\left(\prod_{i=1}^{k} P(X_i \leq t) - \prod_{i=1}^{k} P(X_i \leq t-1)\right)$$

If N encoding symbols must be received on each channel, then $P(X_i \leq t)$ is the probability that at most $t - N$ of the first t generated symbols are erased, so

$$P(X_i \leq t) = \sum_{j=0}^{t-N} \binom{t}{j} \gamma_i^j (1-\gamma_i)^{t-j},$$

and $E(\max(X_1, \ldots, X_k) =$

$$\sum_{t=N}^{\infty} t\left[\prod_{i=1}^{k} \sum_{j=0}^{t-N} \binom{t}{j} \gamma_i^j (1-\gamma_i)^{t-j} - \prod_{i=1}^{k} \sum_{j=0}^{t-N-1} \binom{t-1}{j} \gamma_i^j (1-\gamma_i)^{t-1-j}\right].$$

For the constrained MEC, this becomes $E(\max(X_1, \ldots, X_k)) =$

$$\sum_{t=N}^{\infty} t(1-\gamma)^{tk}\left[\left(\sum_{j=0}^{t-N} \binom{t}{j} \frac{\gamma^j}{(1-\gamma)^j}\right)^k - \left(\sum_{j=0}^{t-N-1} \binom{t-1}{j} \frac{\gamma^j}{(1-\gamma)^{j+1}}\right)^k\right]$$

□

Note that on the MEC, an expected $\frac{N}{1-\varepsilon}$ symbols must be generated before N are received with no (full or partial) erasures. As noted in [7], on the constrained MEC, $1-\varepsilon = (1-\gamma)^k$, and thus an expected $\frac{N}{(1-\gamma)^k}$ symbols must be generated before N are received with no (full or partial) erasures.

In Table I, the second row shows the expected number of symbols to be generated on the constrained MEC for $N = 3$, $k = 4$, and various γ. The third row shows the expected number of symbols to be generated before 3 are received with no (full or partial) erasures. Note that as the bit erasure probability γ increases, partial erasures become more common. Since any partial erasure is treated as a full erasure on the QEC, more outputs are treated as erasures on the QEC than on the MEC.

Another estimate for the required number of symbols to be generated can be found by finding the least integer M such that the expected number of symbols received given that M are sent is at least N for each component. The expected number of symbols received by the component with the most erasures (note that this might not be the component with the highest erasure probability) given that M are sent is

$$\sum_{(i_1, i_2, \cdots, i_k)} \left(\left(\prod_{j=1}^{k} \binom{M}{i_j}\right) \gamma^{\sum_{j=1}^{k} i_j} (1-\gamma)^{\sum_{j=1}^{k} (M-i_j)} \min_{j=1,\ldots,k} (M - i_j)\right),$$

where $(i_1, i_2, \cdots, i_k)$ is a k-tuple with integer entries between 0 and M.

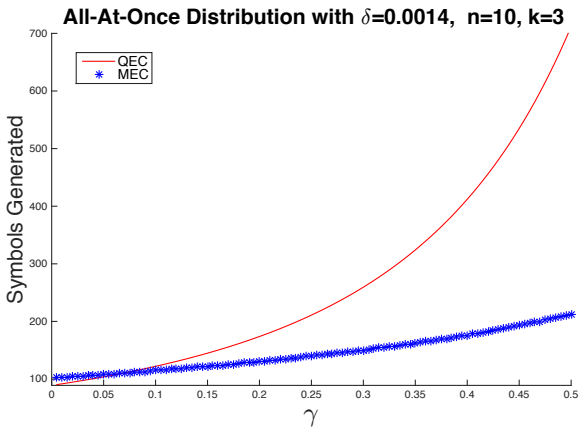

Fig. 7. Upper Bound on Expected Number of Symbols Generated using the All-At-Once Distribution with $\delta_{total} = 0.0014$, n=10, and $k = 3$.

If decoding is successful on the QEC, then decoding will also be successful on the MEC without generating additional encoding symbols. Thus the expected number of encoding symbols that must be generated for successful decoding on the MEC may be bounded above by the minimum of the expected number needed for the corresponding QEC and the number found in the Proposition IV.1.

Example IV.2. *Consider an LT Code with $q = p^3$, $n = 10$ message symbols, and allowable probability of decoding failure $\delta_{total} = 0.0014$. Using the All-At-Once distribution, approximately 89 intact encoding symbols must be received on the QEC. In order for each component to have decoding failure at most $1 - (1 - \delta_{total})^{\frac{1}{k}}$, approximately 100 symbols must be received by each component on the MEC. Figure 7 shows the upper bound given by Proposition IV.1 for the expected number of symbols that must be generated on the MEC for various γ as well as the expected number of encoding symbols that must be generated on the corresponding QEC.*

Lemma IV.3. *The QBMC has no gain (in terms of number of symbols generated) over the corresponding QEC.*

Proof. Note that if any erasure event occurs on the QBMC, then the rightmost bit is erased. Decoding is successful exactly when decoding of the rightmost bit of each symbol is successful. The effective channel for the rightmost bit is a BEC with erasure probability $\sum_{i=1}^{k} \varepsilon_i = 1 - \varepsilon_0$, and an expected $\frac{N}{\varepsilon_0}$ symbols must be generated. This is the same as the expected number of symbols to be generated before N are received with no (full or partial) erasures. □

V. Density Evolution Analysis

Since LT codes on the QPEC can not be analyzed using component graphs as in the MEC or QBMC, we give an approximation of density evolution (DE) equations for the two-phase decoder on the QPEC. DE is an asymptotic analysis that tracks how message probability density functions evolve as iterations progress, and is based on the degree distribution of the graph. The analysis assumes that messages received at each node are independent, (i.e. the graph is cycle free). DE

analysis of LDPC codes on the QPEC was done in [6]. In contrast to DE equations for LDPC codes that have variable and constraint nodes, these DE equations must be adapted for the encoding and message nodes in the LT code framework.

Using the notation in [6], let $\mathcal{S}_1, \ldots, \mathcal{S}_{2^q-1}$ denote the nonempty subsets of $GF(q)$, and let $\mathcal{I}_i$ be an ordered list of i (not necessarily distinct) indices from $1, \ldots, 2^q - 1$. Let $\chi_t(\mathcal{I}_i)$ indicate if the sets indexed in $\mathcal{I}_i$ lead to information set S_t at a message node. That is, $\chi_t(\mathcal{I}_i)$ is 1 if $\bigcap_{j\in\mathcal{I}_i} S_j$ is S_t, and 0 otherwise. Similarly, let $\eta_t(\mathcal{I}_i)$ indicate if the sets indexed in $\mathcal{I}_i$ lead to information set S_t at an encoding node. That is, $\eta_t(\mathcal{I}_i)$ is 1 if the sumset of sets in $\mathcal{I}_i$ is S_t, and 0 otherwise. Let $\theta_t = 1$ if $|\mathcal{S}_t| = 1$, and 0 otherwise. Denote by $p^{\ell}_{m\to e}(\mathcal{S}_t)$ (resp., $p^{\ell}_{e\to m}(\mathcal{S}_t)$) the probability that $\mathcal{S}_t$ is sent from a message node to an encoding node (resp., from an encoding node to a message node) in iteration ℓ.

We now derive DE equations for Phase II as an approximate analysis for the two-phase decoder. While the expected performance of LT codes is optimized for the Ideal Soliton distribution, in practice the performance is better when encoding is done using a Robust Soliton degree distribution with an expected ripple size greater than one [1]. Let λ_i (resp., ρ_i) be the fraction of edges incident to a message node (resp., encoding node) of degree i. For the Ideal or Robust Soliton degree distribution,

$$\lambda_i = e^{-d_\rho/R}\frac{(d_\rho/R)^{i-1}}{(i-1)!}, \qquad \rho_i = \frac{niE_i}{\sum_i niE_i},$$

where $R = \frac{\#\text{ message symbols}}{\#\text{ encoding symbols}}$ and E_i is the probability of an encoding node having degree i in the distribution given in [1], d_ρ is the average degree of encoding nodes given by $\frac{\sum_i niE_i}{n}$ [12], and n is the number of encoding nodes (assumed to be large for the DE analysis). For $\ell \geq 1$,

$$p^{\ell}_{m\to e}(\mathcal{S}_t) = \sum_{i\geq 2,\lambda_i\neq 0} \lambda_i \sum_{\mathcal{I}_{i-1}} \left(\prod_{j\in\mathcal{I}_{i-1}} p^{\ell-1}_{e\to m}(\mathcal{S}_j)\right)\cdot \chi_t(\mathcal{I}_{i-1}),$$

$$p^{\ell}_{e\to m}(\mathcal{S}_t) = \rho_1 p^0_{e\to m}(t)(1-\theta_t) + \frac{1-\varepsilon}{q}\theta_t + \varepsilon \sum_{i\geq 2,\rho_i\neq 0} \rho_i \sum_{\mathcal{I}_{i-1}} \left(\prod_{j\in\mathcal{I}_{i-1}} p^{\ell}_{m\to e}(\mathcal{S}_j)\right)\cdot \eta_t(\mathcal{I}_{i-1}),$$

and the initial conditions when $\ell = 0$ are

$$p^0_{e\to m}(S_t) = \begin{cases} \frac{\varepsilon}{\binom{q}{M}} & \text{if } |\mathcal{S}_t| = M \\ (1-\varepsilon)\frac{1}{q} & \text{if } |\mathcal{S}_t| = 1 \\ 0 & \text{else} \end{cases}.$$

At the end of iteration ℓ, the probability of decoding failure is given by $1 - \sum_{t:|S_t|=1} p^{\ell}_{m\to e}(S_t)$, i.e. failure occurs if the cardinality of the information set sent by some message node is never one. The threshold of the DE process is defined as

$$\epsilon^* = \arg\max_{\varepsilon}\left\{\left(1 - \sum_{|S_t|=1} p^{\ell}_{m\to e}(\mathcal{S}_t)\right) \to 0 \text{ as } \ell\to\infty\right\},$$

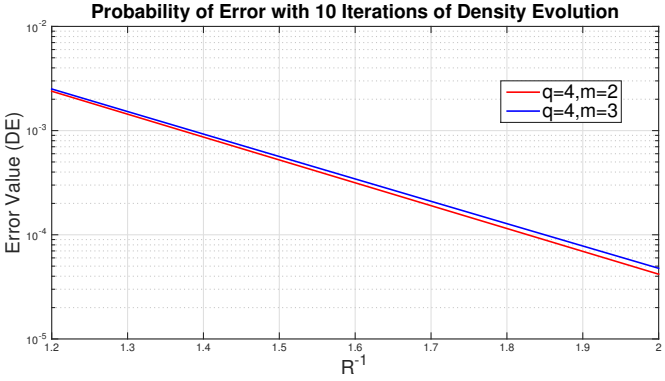

Fig. 8. Performance of LT codes on QPEC for $\varepsilon = 0.1$ using a modified Ideal Soliton distribution.

and gives a lower bound on the decoding threshold for the two-phase decoder. While degree one encoding nodes are essential to the LT process, their presence results in a non-vanishing term in $p^{\ell}_{e\to m}(\mathcal{S}_t)$. Thus DE analysis for LT codes may be hindered by an error floor [12]. This may be overcome by using an outer code, as is done with Raptor codes [3].

Figure 8 shows the asymptotic performance of LT codes on the QPEC with $q = 4$, $M = 2$ or 3, and $\varepsilon = 0.1$. The plot shows the error rate decreasing as a function of R. Due to the complexity of the DE on the QPEC, the DE equations were run for 10 iterations using a modification of the Ideal Soliton distribution without high degree encoding nodes. The performance curves are similar for these parameters due to the degree one encoding node terms. Optimizing the degree distribution for LT codes on partial erasure channels, and in particular on the QPEC, is a subject for future work.

References

[1] M. Luby, "LT codes", *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, 2002, pp. 271-280.

[2] A. Shokrollahi, S. Lassen, M. Luby, "Multi-stage Code Generator and Decoder for Communication Systems", U.S. Patent Application 20030058958, Dec. 2001.

[3] A. Shokrollahi, "Raptor Codes", *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2551-2567 June 2006.

[4] J. H. Sørensen, T. Koike-Akino, P. Orlik, J. Østergaard, and P. Popovski, "Ripple Design of LT Codes for BIAWGN Channels", Mitsubishi Electric Research Laboratories, TR2014-007, 2014.

[5] O. Etesami and A. Shokrollahi, "Raptor Codes on Binary Memoryless Symmetric Channels", *IEEE Transactions on Information Theory*, vol. 52, no. 5, pp. 2033-2051, May 2006.

[6] R. Cohen and Y. Cassuto, "Iterative Decoding of LDPC Codes over the q-ary Partial Erasure Channel", *IEEE Transactions on Information Theory*, vol. 62, no. 5, pp. 2658-2672, May 2016.

[7] C. Mayer, K. Haymaker, C.A.Kelley, "Coding for multilevel and partial erasure channels", submitted.

[8] R. Cohen and Y. Cassuto "LDPC Codes for the q-ary Bit Measurement Channel", *2016 9th International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*, Brest, 2016, pp. 261-265.

[9] R. Cohen and Y. Cassuto, "LDPC codes for partial-erasure channels in multi-level memories", *2014 IEEE International Symposium on Information Theory*, Honolulu, HI, 2014, pp. 2097-2101.

[10] A. Khisti, "Tornado Codes and Luby Transform Codes", 2003.

[11] J. H. Sørensen, P. Popovski, and J. Østergaard, "Design and Analysis of LT Codes with Decreasing Ripple Size", *IEEE Transactions on Communications*, vol. 60, no. 11, pp. 3191-3197, November 2012.

[12] G. Joshi, J.B. Rhim, J. Sun, D. Wang, "Fountain Codes", Project Report 6.451, December 2010.