On the Performance of Distributed LT Codes

Srinath Puducheri, Jörg Kliewer, and Thomas E. Fuja
Department of Electrical Engineering,
University of Notre Dame, Notre Dame, IN 46556, USA
Email: {spuduche, jkliewer, tfuja}@nd.edu

***Abstract*— In this paper we extend previous results on *distributed* LT codes and focus on the frame erasure rate performance of the corresponding block codes with fixed rate. Specifically, we address *modified* LT (MLT) codes which resemble LT codes in both structure and performance. First, an MLT code construction scheme for four sources communicating with a single sink via a common relay is presented. Then, for the fixed rate case and erasure channels at both source-relay and relay-sink links, simulation results are given for the frame erasure rate versus the corresponding erasure probabilities. We observe that MLT codes strongly outperform a scheme where an LT block code for every source is separately transmitted over the relay-sink erasure channel. This is due to the increased codeword length for the MLT code-based schemes.**

I. Introduction

LT codes belong to the general class of digital fountain codes [1], [2], [3], which are sparse graph-based codes for erasure channels. The advantage of such codes compared to traditional channel codes is that in a data transmission scenario with packet erasures (e.g., in a unicast or multicast over communication networks), the number of retransmissions to identify missing data packets is significantly reduced. This is due to the rateless property of such codes where, in principle, an *arbitrary* number of code symbols can be derived from an information block. Thus, the receiver needs only to acknowledge the successful decoding of the information word after a sufficient number of code symbols have been received.

LT codes were developed by Michael Luby [4] as the first practical digital fountain coding scheme. In these codes the information and code symbols are binary strings of length l, and all encoding and decoding processes involve only bitwise XOR operations. LT codes are very efficient as the data length grows, i.e., the fractional overhead of code symbols required (on average) to decode the data decreases as the block size of the data increases.

In this paper we extend previous work [5] on *distributed LT (DLT) codes*, which are obtained by decomposing an LT code into two or more codes. By simply performing a selective XOR of the DLT code symbols, a *modified LT (MLT) code* is obtained, which by design resembles an LT code in both structure and performance. In [5] we introduced a scenario in which two or more independent sources in a network are communicating with a sink through a common relay with limited data processing capability. All sources have the same amount of data to transmit to the sink. In this case, using DLT codes at the sources and a selective XOR at the relay, all the data can be transmitted to the sink by effectively using one big MLT code, rather than several small LT codes for the different sources. Thus, we can take advantage of the benefits of larger data block lengths in the DLT case.

In this paper, we first provide an explicit construction of MLT codes for the case of four source nodes. We then focus on the erasure correction behavior of MLT codes for both lossy source-relay and relay-sink erasure channels. This requires the corresponding LT and MLT codes to have a constant code rate with a fixed number of code symbols. Note that this case is important in a real transmission scenario where the minimal possible code rate must be fixed due to latency constraints and to make economical use of the available transmission resources.

This work was supported in part by the U.S. National Science Foundation under grants CCF 02-05310 and EEC 02-03366, the German Research Foundation (DFG) under grant KL 1080/3-1, and the University of Notre Dame Faculty Research Program.

II. Description of LT Codes

This section reviews the results of [4].

A. Encoding and Decoding Procedures

Assume that the data consists of k information symbols. Then each LT code symbol is generated as follows:

- An integer d (called the *degree* of the code symbol) between 1 and k is randomly chosen according to a specific *degree distribution*, called the *robust soliton distribution* [4].
- From the set of k information symbols, a subset of d symbols is chosen uniformly at random – this constitutes the set of *neighbors* of the code symbol. The neighbors are bitwise XOR-ed to produce the code symbol.

The decoder is assumed to know the degree of every code symbol and the set of neighbors associated with it. The decoder (described in [4]) essentially runs a belief propagation algorithm on the bi-partite graph describing the connections between the information and code symbols, similar to the decoding of low density parity check (LDPC) codes [7].

B. Robust Soliton Distribution

The degree distribution used for constructing LT codes – the robust soliton distribution (RSD) – is constructed such that the decoder can recover the data from slightly more than k code symbols with probability at least $1-\delta$ [4].

Definition 1 ([4]). *For constants $c > 0$ and $\delta \in [0,1]$, the robust soliton distribution $\mu(i)$ is given by*

$$\mu(i) = \frac{\rho(i)+\tau(i)}{\beta}, \quad \textit{for} \quad 1 \le i \le k, \tag{1}$$

$$\text{where} \quad \beta = \sum_{i=1}^{k} \Big(\rho(i)+\tau(i)\Big). \tag{2}$$

Here, $\rho(i)$ (a probability distribution over $1 \le i \le k$) and $\tau(i)$ are given by

$$\rho(i) = \begin{cases} 1/k, & \textit{for} \quad i = 1, \\ 1/(i(i-1)), & \textit{for} \quad 2 \le i \le k, \end{cases} \tag{3}$$

$$\tau(i) = \begin{cases} S/ik, & \textit{for} \quad 1 \le i \le \frac{k}{S}-1, \\ S\ln(S/\delta)/k, & \textit{for} \quad i = \frac{k}{S}, \\ 0, & \textit{otherwise.} \end{cases} \tag{4}$$

The parameter S represents the average number of degree one code symbols and is defined as

$$S = c \cdot \sqrt{k} \cdot \ln\Big(\frac{k}{\delta}\Big). \tag{5}$$

Luby [4] showed that, for a suitably chosen c (independent of k and δ), the decoder can recover the data from $n = k\beta = k + c \cdot \sqrt{k} \cdot \ln^2(k/\delta)$ LT code symbols, with probability at least $1-\delta$. However, as pointed out in [6, p. 592], in practice c can be treated as a free parameter.

III. Two sources: MLT-2 codes

In [5] we addressed the decomposition of LT codes into two DLT-2 codes. By selectively XORing the resulting DLT codewords it was demonstrated that the resulting code resembles an LT code in both structure and performance. We denote such a code a *modified LT* (MLT) code. Consider the network in Fig. 1(a). Nodes S_1 and S_2 are associated with two data blocks $\mathscr{D}_1$ and $\mathscr{D}_2$, respectively, where each node encodes its data block with a DLT code according to a degree distribution p, the so-called *deconvolved soliton distribution* (DSD). These code symbols are transmitted sequentially to node N, which then, as shown in [5], selectively XORs the codewords received from S_1 and S_2 and transmits the resulting codeword with the same length to T. In the following we summarize the steps leading to the derivation of the DSD.

First, we decompose the RSD $\mu(i)$ into two distributions, $\mu'(i)$ and $\mu''(i)$, such that deconvolving $\mu'(i)$ yields a valid probability distribution function. Assume that $\rho(i)$ and $\tau(i)$ are given by (3) and (4), respectively. Then define

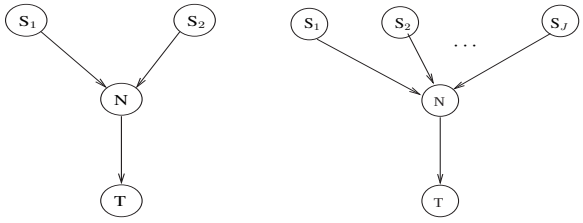

Fig. 1. (a) A two-source single-sink network, (b) A J-source single-sink network

$$\mu'(i) = \begin{cases} 0, & \text{for} \quad i = 1, \\ \frac{\rho(i)+\tau(i)}{\beta'}, & \text{for} \quad 2 \le i \le \frac{k}{S}-1, \\ \frac{\rho(i)}{\beta'}, & \text{for} \quad \frac{k}{S} \le i \le k, \end{cases} \tag{6}$$

with the normalization factor β' given by

$$\beta' = \sum_{i=2}^{k} \rho(i) + \sum_{i=2}^{k/S-1} \tau(i), \tag{7}$$

and

$$\mu''(i) = \begin{cases} \frac{\rho(1)+\tau(1)}{\beta''}, & \text{for} \quad i = 1, \\ \frac{\tau(k/S)}{\beta''}, & \text{for} \quad i = \frac{k}{S}, \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

with the normalization factor β'' given by

$$\beta'' = \rho(1) + \tau(1) + \tau\Big(\frac{k}{S}\Big). \tag{9}$$

Thus, noting that $\beta' + \beta'' = \beta$, from (1) the RSD can be rewritten as

$$\begin{aligned} \mu(i) &= \frac{\beta'}{\beta} \cdot \mu'(i) + \frac{\beta''}{\beta} \cdot \mu''(i) \\ &= \frac{\beta'}{\beta} \cdot \mu'(i) + \left(1 - \frac{\beta'}{\beta}\right) \cdot \mu''(i), \\ &\qquad \text{for} \quad 1 \le i \le k. \end{aligned} \tag{10}$$

The RSD $\mu(i)$ is thus written as a mixture of the distributions $\mu'(i)$ and $\mu''(i)$ with mixing parameter β'/β.

In order to obtain the DSD from the RSD in (10), $\mu'(i)$ is now deconvolved by finding $f(\cdot)$ such that

$$(f * f)(i) = \mu'(i) \quad \text{for} \quad 2 \le i \le k/2+1. \tag{11}$$

After the deconvolution $f(i)$ must be normalized to represent a probability distribution. However, this does not change $f(i)$ much, and we still have $(f*f)(i) \approx \mu'(i)$ for $1 \le i \le k/2+1$ after normalization. Note that a direct deconvolution of $\mu(i)$ is not possible due to a non-zero probability for degree-one symbols, which cannot be represented by convolving two identical distribution with a smallest degree of one. Another reason is that $\mu(i)$ has a spike at $i = k/S$ which would lead to negative values for some i after the deconvolution.

Definition 2 ([5]). *The deconvolved soliton distribution (DSD) $p(\cdot)$ is given by*

$$p(i) = \lambda \cdot f(i) + (1-\lambda) \cdot \mu''(i), \quad \textit{for} \quad 1 \le i \le k/2, \tag{12}$$

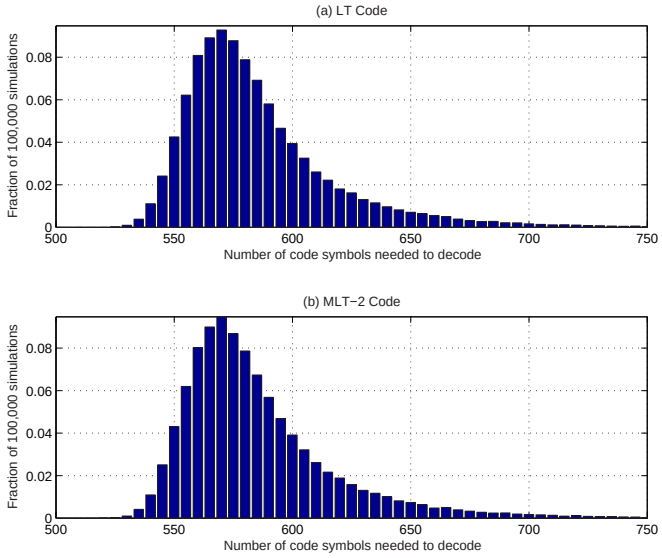

Fig. 2. Histograms of the number of code symbols needed to recover the data, over 1000 simulations – (a) LT code with $k = 500$, (b) Proposed encoding scheme with $|\mathscr{D}_1| = |\mathscr{D}_2| = k/2 = 250$. The parameters of the RSD used for both cases are $c = 0.05$, $\delta = 0.5$.

with the parameter λ given by

$$\lambda = \sqrt{\frac{\beta'}{\beta}}. \tag{13}$$

From (12), we can view $p(i)$ as the mixture of $f(i)$ and $\mu''(i)$ with mixing parameter λ. This property is exploited in the encoding procedure at the relay N, as described in [5].

Since one MLT code symbol is generated from combining two DLT-2 code symbols, we call the resulting code an MLT-2 code. The degree d of an MLT-2 symbol approximately obeys the RSD for $1 \leq d \leq k/2$. We observe from Fig. 2, where histograms for the number of code symbols required to decode k symbols of data are plotted for both an LT and an MLT-2 code, that both constructions have almost identical performance.

IV. Extension to MLT-4 codes

Consider now the situation when $J = 2^M$ sources are communicating with a common sink through a relay, as shown in Fig. 1(b). If we are able to combine all the J sources' data and encode them into a single codeword at the relay, then we can exploit the benefits of the larger data blocklength compared to J LT codes (or $J/2$ MLT-2 codes) being transmitted separately via the relay-to-sink link. Specifically, we address the case of $J = 4$ in the following and consider the resulting MLT-4 codes. To accomplish this, we further decompose DLT-2 codes into two subcodes to obtain DLT-4 codes, which are then used by the four sources. At each instant, the four DLT-4 code symbols from the four sources are selectively XORed at the relay to yield an MLT-4 code.

We start by finding two codes which, when selectively XORed, will yield a DLT-2−like code, i.e., the resulting code and the DLT-2 code have the DSD as their degree distribution. As before, this requires the deconvolution of the target degree distribution, namely the DSD.

A. Deconvolution of the DSD

Assume that each source s_i $(1 \leq i \leq 4)$ has a block $\mathscr{D}_i$ of $\frac{k}{4}$ data symbols to send to the sink T. Direct deconvolution of the DSD is not possible, owing to similar problems as encountered while attempting to deconvolve the RSD because of the "spiky" behavior of the distribution. As a result, we adopt a similar methodology as before, and first split the DSD into two component distributions. The split is along the lines of (12), except for some minor differences.

The difference is that the $f(i)$ component of the DSD has to be slightly modified at $i = \frac{k}{S} - 1$, as otherwise the split-and-deconvolve approach used in Section III yields a negative value at $i = \frac{k}{S} - 2$ for the new distribution. In effect, to "smooth" out $f(i)$, we replace the true value of $f(\frac{k}{S} - 1)$ by a linear interpolation between $f(\frac{k}{S} - 2)$ and $f(\frac{k}{S})$. Thus, we define a new distribution $f^{(new)}(i)$ from $f(i)$ as follows:

$$f^{(new)}(i) = \frac{f(i)}{\gamma}\ , \quad \text{for } 1 \leq i \leq \frac{k}{2}\ ,\ i \neq \frac{k}{S} - 1\ ,$$
$$f^{(new)}\Big(\frac{k}{S} - 1\Big) = \frac{1}{\gamma} \cdot \frac{f(\frac{k}{S}) + f(\frac{k}{S} - 2)}{2}\ , \tag{14}$$

where the normalization factor γ is given by

$$\gamma = \sum_{i \neq \frac{k}{S} - 1} f(i) \ + \ \frac{f(\frac{k}{S}) + f(\frac{k}{S} - 2)}{2}. \tag{15}$$

It turns out that we have not distorted $f(i)$ much except at $i = \frac{k}{S} - 1$, because $f(i)$ is quite small in the neighborhood of $i = \frac{k}{S}$ and hence, γ is very close to one.

We now define $p^{(new)}(i)$ as a mixture of $f^{(new)}(i)$ and $\mu''(i)$ according to

$$p^{(new)}(i) = \lambda \cdot f^{(new)}(i) + (1 - \lambda) \cdot \mu''(i) \tag{16}$$

for $1 \leq i \leq k/2$. This is exactly the same as (12) with $f(i)$ replaced by $f^{(new)}(i)$. The mixing parameter λ is as defined in (13). In the next step $p^{(new)}(i)$ is split into two distributions $p'(i)$ and $p''(i)$ as follows:

$$p'(i) = \begin{cases} 0, & \text{for} \quad i = 1, \\ \frac{1}{\gamma'} \cdot \lambda \cdot f^{(new)}(\frac{k}{S}), & \text{for} \quad i = \frac{k}{S}, \\ \frac{1}{\gamma'} \cdot p^{(new)}(i), & \text{otherwise}, \end{cases} \tag{17}$$

with the normalization factor γ' given by

$$\gamma' = \sum_{i=2}^{\frac{k}{S}-1} p^{(new)}(i) + \lambda \cdot f^{(new)}\Big(\frac{k}{S}\Big) + \sum_{i=\frac{k}{S}+1}^{\frac{k}{2}} p^{(new)}(i)\ , \tag{18}$$

and

$$p''(i) = \begin{cases} \frac{1}{\gamma''} \cdot p^{(new)}(1) & \text{for} \quad i = 1, \\ \frac{1}{\gamma''} \cdot (1 - \lambda) \cdot \mu''(\frac{k}{S}) & \text{for} \quad i = \frac{k}{S}, \\ 0 & \text{otherwise}, \end{cases} \tag{19}$$

with the normalization factor γ'' given by

$$\gamma'' = p^{(new)}(1) + (1-\lambda)\cdot\mu''\Big(\frac{k}{S}\Big)\ . \tag{20}$$

Thus, we have:

$$\begin{aligned} p^{(new)}(i) &= \frac{\gamma'}{\gamma}\cdot p'(i) + \frac{\gamma''}{\gamma}\cdot p''(i) \\ &= \frac{\gamma'}{\gamma}\cdot p'(i) + \left(1-\frac{\gamma'}{\gamma}\right)\cdot p''(i), \\ &\quad \text{for} \quad 1 \le i \le \frac{k}{2}. \end{aligned} \tag{21}$$

We finally proceed to deconvolve $p'(i)$ to obtain $g(i)$ according to

$$g(i) = \begin{cases} \sqrt{p'(2)} & \text{for} \quad i = 1, \\ \frac{p'(i+1) - \sum_{j=2}^{i-1} g(j)\ g(i+1-j)}{2\,g(1)} & \text{for} \quad 2 \le i \le \frac{k}{4}, \\ 0 & \text{for} \quad \frac{k}{4} < i \le k. \end{cases} \tag{22}$$

Owing to the relatively small contribution from the tail of $p'(i)$ for $\frac{k}{4}+2 \le i \le \frac{k}{2}$, and the fact that the above deconvolution yields non-negative values for $g(i)$ even if carried out for all i up to $\frac{k}{2}$, we obtain

$$\sum_{i=1}^{k/4} g(i) \approx 1. \tag{23}$$

We now define the *doubly-deconvolved soliton distribution* as follows:

Definition 3. *The doubly deconvolved soliton distribution (DDSD) $q(\cdot)$ is given by*

$$q(i) = \eta\cdot g(i) + (1-\eta)\cdot p''(i)\,, \quad \textit{for} \quad 1 \le i \le k/4, \tag{24}$$

with the mixing parameter η given by

$$\eta = \sqrt{\frac{\gamma'}{\gamma}}. \tag{25}$$

B. Constructing an MLT-4 code with four sources

The encoding scheme is based on the construction of an MLT-2 code for the two-source network, as developed in [5]. Consider the network of Fig. 1(b) with $J = 4$. The encoding is done as follows:

1) Each source s_i generates a code symbol X_i using the data in $\mathscr{D}_i$ ($i = 1, 2, 3, 4$). The encoding is done in two stages, such that the final degree distribution of X_i is the DDSD $q(\cdot)$.
   a) Each source first randomly selects the component of the DDSD to be used: it chooses either $g(i)$ (with probability η) or $p''(i)$ (with probability $1-\eta$).
   b) The degree d is generated using the selected component distribution and a code symbol X_i of degree d is formed by randomly selecting d neighbors, equiprobably over the $\binom{k/4}{d}$ possible selections.
2) Along with every code symbol X_i, a bit b_i is transmitted to the relay N to indicate the component distribution – a '0' indicates $g(i)$, and a '1' indicates $p''(i)$.
3) The relay computes Y, the code symbol to be transmitted to the sink, in two stages.
   First, the relay computes (Y_1, Y_2): Y_1 from (X_1, X_2, b_1, b_2), and Y_2 from (X_3, X_4, b_3, b_4). The symbol Y_1 is computed as follows:
   - If $b_1 = 0$ and $b_2 = 0$, then $Y_1 = X_1 \oplus X_2$.
   - If $b_1 = 0$ and $b_2 = 1$, then $Y_1 = X_2$.
   - If $b_1 = 1$ and $b_2 = 0$, then $Y_1 = X_1$.
   - If $b_1 = 1$ and $b_2 = 1$, then Y_1 is picked to be either X_1 or X_2 with probability one-half.

   Thus, the relay XORs X_1 and X_2 if and only if they are both derived according to $g(i)$. Likewise Y_2 is computed from (X_3, X_4, b_3, b_4). It can be easily observed that Y_1 and Y_2 obey the DSD for degrees up to $k/4$ (approximately, as we have modified the degree at $i = \frac{k}{S} - 1$). Hence Y_1 and Y_2 may be considered DLT-2 code symbols. Finally, the relay computes Y as a function of Y_1 and Y_2 according to the encoding scheme for the MLT-2 code [5].
4) The symbol Y, which approximately obeys the RSD, is transmitted to the sink T.

The code generated by each source is a *DLT-4 code*, and the final code received by the sink is an *MLT-4 code*. The degree d of an MLT-4 symbol approximately obeys the RSD for $1 \le d \le k/4$.

V. Performance Results

A. Performance of MLT-2 codes for error-free source-relay links

First consider the case of Fig. 1(a) when the source-relay channels are assumed to be error-free, and the relay-sink channel is modeled as a symbol erasure channel with an erasure probability p_{rs}. This is motivated by assuming that the distance from the source node to the relay is quite small, and thus the resulting channel is quite reliable whereas the relay-to-sink link is likely affected by channel errors. Each source has k information symbols. Furthermore, assume that all codes have fixed rate. Such a situation may arise if there is no feedback path from the sink to the sources, e.g., in a broadcast scenario where the intended receivers are potentially unknown to the source; the receivers simply randomly tune in to the broadcast at different times and collect a sufficient number of code symbols in order to decode the data.

We compare two scenarios:

- **Scheme I:** Each source employs an LT code of codeword length n, such that $\frac{k}{n} = R$. The codewords are multiplexed at the relay and transmitted to the sink – thus, a total of $2n$ symbols are transmitted by the relay. A frame erasure is said to occur if the sink cannot recover all the k data symbols of a source from the code symbols it receives.

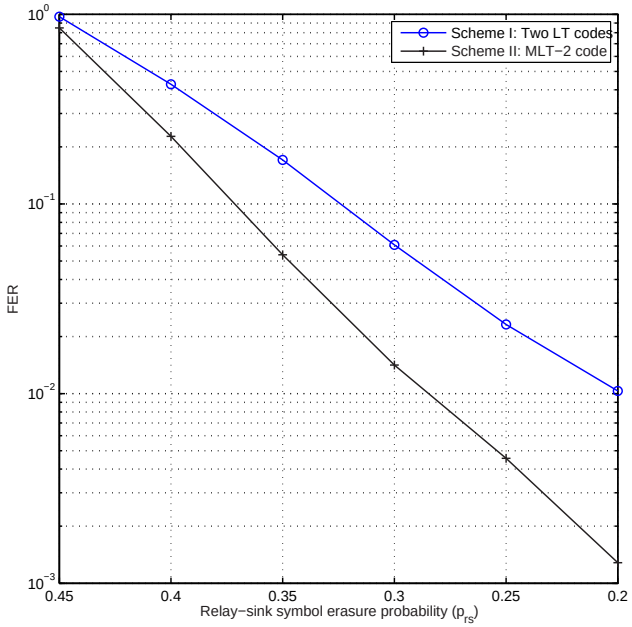

Fig. 3. Frame erasure rate (FER) Curves for LT and MLT-2 codes: $k = 250$, $R = 0.5$, $n = 500$ ($c = 0.05$, $\delta = 0.5$ are the parameters used for the RSD for both codes).

- **Scheme II:** Each source employs a DLT-2 code of length $2n$, such that $\frac{k}{n} = R$. Symbols from the two DLT-2 codewords are selectively XOR-ed at the relay and the resulting MLT-2 codeword of length $2n$ (conveying a total of $2k$ information symbols) is transmitted to the sink. In this case, a frame erasure occurs if the sink cannot recover all the $2k$ data symbols of the two sources from the received codeword.

For both these schemes, the frame erasure rate (FER) of a single source, as seen by the sink, is plotted in Fig. 3 as a function of the symbol erasure rate on the relay-sink channel. It is seen that there is a significant drop in FER in going from Scheme I to II.

B. Performance of MLT-2 codes for lossy source-relay links

Now consider the situation when, in addition to the relay-sink channel, the two source-relay channels are symbol erasure channels, with identical erasure probabilities p_{sr}. The performance of LT and MLT-2 *block* codes under this framework is considered.

Again, as in Section V-A, assume that each source has k symbols of data to transmit. The two schemes in Section V-A are retained for comparison. A minor adaptation to Scheme II is made in that, for the case when a DLT-2 symbol from one source is erased, the relay merely transmits the DLT-2 symbol received from the other source (provided that is not erased) to the sink. As before, the effective rates of both coding schemes on the relay-sink channel are fixed at $R = \frac{k}{n}$.

In Figs. 4, 5 and 6, the FER versus symbol erasure probability p_{rs} on the relay-sink channel is shown for different values of p_{sr}. For comparison, the FER curves of Schemes I and II for the lossless case with $p_{sr} = 0$ are also shown in these figures.

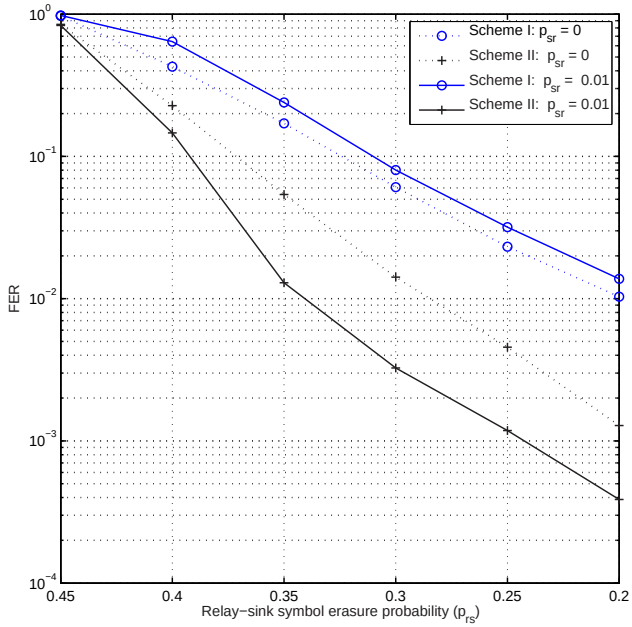

Fig. 4. Frame erasure rate (FER) versus relay-sink erasure probability p_{rs} for LT and MLT-2 codes when the source-relay erasure probability $p_{sr} = 0.01$; $k = 250$, $R = 0.5$, $n = 500$ ($c = 0.05$, $\delta = 0.5$ are the parameters used for the RSD for both codes).

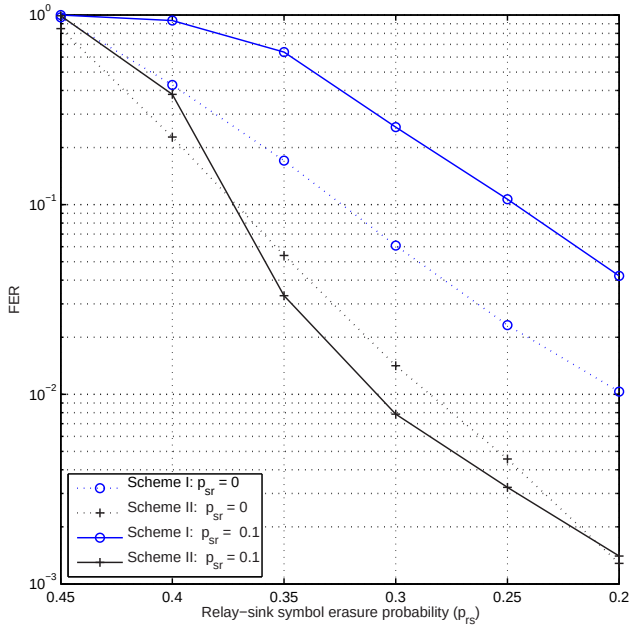

Fig. 5. Frame erasure rate (FER) versus relay-sink erasure probability p_{rs} for LT and MLT-2 codes when the source-relay erasure probability $p_{sr} = 0.1$; $k = 250$, $R = 0.5$, $n = 500$ ($c = 0.05$, $\delta = 0.5$ are the parameters used for the RSD for both codes).

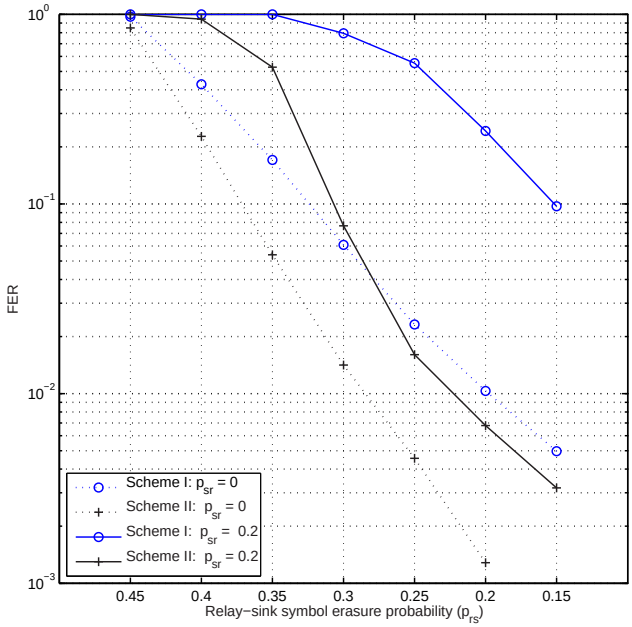

Fig. 6. Frame erasure rate (FER) versus relay-sink erasure probability p_{rs} for LT and MLT-2 codes when the source-relay erasure probability $p_{sr} = 0.2$; $k = 250$, $R = 0.5$, $n = 500$ ($c = 0.05$, $\delta = 0.5$ are the parameters used for the RSD for both codes).

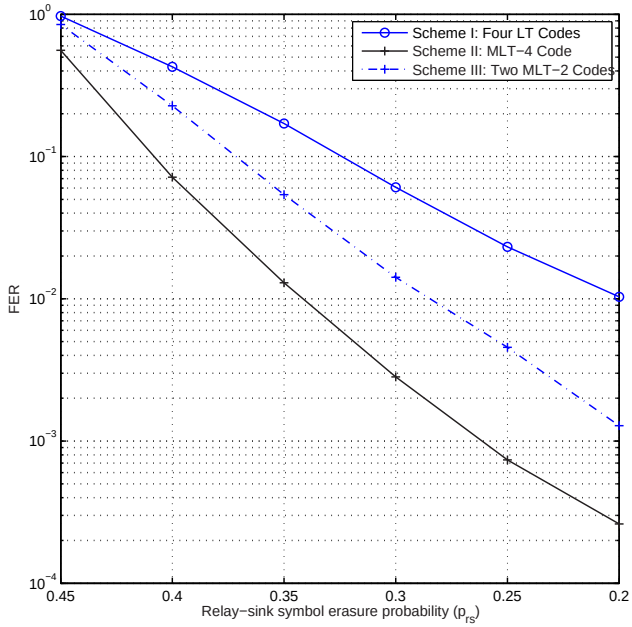

Fig. 7. Frame erasure rate (FER) versus relay-sink erasure probability p_{rs} for LT, MLT-2 and MLT-4 codes: $k = 250$, $R = 0.5$, $n = 500$ ($c = 0.05$, $\delta = 0.5$ are the parameters used for the RSD for all three codes).

Interestingly, for small values of p_{sr} (around 1%) the MLT-2 scheme performs better than for the lossless case. This could be due to the fact that in this regime the relay almost *never* sees an erasure on both the source-relay channels simultaneously. Thus, for a fraction $1 - 2\,p_{sr}$ of the time, when neither source's symbol has been erased, the relay transmits MLT-2 code symbols to the sink, covering the data of both sources. However, for the remaining fraction $2\,p_{sr}$ of the time, it only transmits DLT-2 code symbols, thus covering the data of only one source at any time. This *randomization* between a DLT-2 code (covering one source at a time) and an MLT-2 code (covering both sources) aids the sink in decoding if the mixing parameter $2\,p_{sr}$ is chosen appropriately small.

As p_{sr} grows larger ($\approx 10\%$), the FER for Scheme II eventually starts deteriorating when compared with the small p_{sr} case, as would be expected.

C. Performance of MLT-4 codes

We consider four-source networks wherein a fixed-rate, finite-blocklength constraint is enforced on the coding schemes under comparison (as before with the two-source case). Each source has k data symbols, and the rate of the coding scheme is fixed at R. We only address lossless source-relay links in the following considerations. The schemes under consideration are:

- **Scheme I:** The four sources each encode their k data symbols as n LT code symbols (such that $R = \frac{k}{n}$), which are multiplexed at the relay and sent to the sink.
- **Scheme II:** Each source encodes its data onto $4n$ DLT-4 code symbols (such that $R = \frac{k}{n}$) which are selectively XORed with code symbols from the other sources at the relay to yield MLT-4 code symbols, to be transmitted to the sink. Clearly, the rate of the final MLT-4 code is $R = \frac{4k}{4n}$.
- **Scheme III:** The four sources each encode their k data symbols as $2n$ DLT-2 code symbols ($R = \frac{k}{n}$). At the relay, the DLT-2 code symbols from s_1 and s_2 are combined to give an MLT-2 code over both the sources' data. Likewise, the DLT-2 code symbols from s_3 and s_4 are combined to give another MLT-2 code. The two MLT-2 codes are multiplexed and transmitted to the sink. The rate of these MLT-2 codes is $R = \frac{2k}{2n}$.

Fig. 7 shows the FER versus the relay-sink erasure probability p_{rs} for $k = 250$ and $R = 0.5$. It can be seen that both Schemes II and III have a significantly lower FER than Scheme I. In particular, the MLT-4 code provides a significant improvement in performance over the two MLT-2 codes as well.

D. Comparison with Parent LT Codes

In this section, we present a comparison of MLT codes with the *parent LT codes*[1] from which they were derived. Specifically, we compare the performance of the MLT-2 and MLT-4 codes over a lossy relay-sink link (as discussed above), with that of similar data-size LT codes over a lossy link. The total number of data symbols for both MLT and LT codes is denoted by k. The corresponding FER curves are shown in Fig. 8.

It is seen that the MLT-2 code performs about as well as its parent LT code, whereas, the MLT-4 code performs slightly worse than its parent LT code. This could be attributed to

[1]LT codes of the same information blocklength and the same approximate degree distribution as the MLT codes

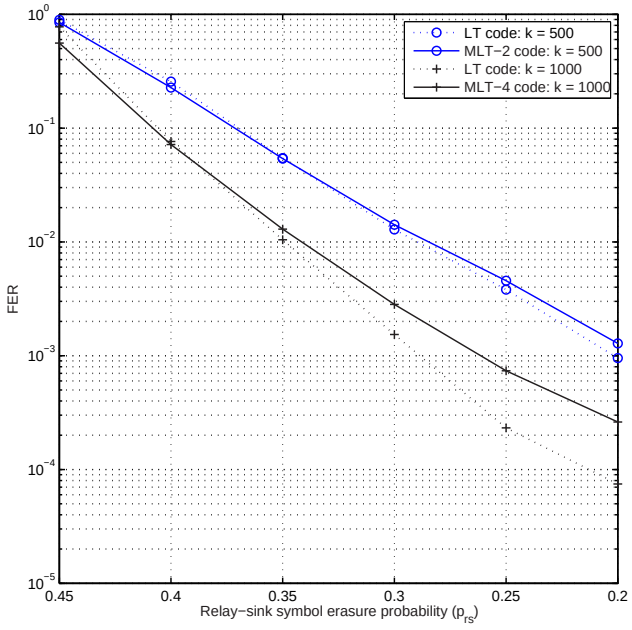

Fig. 8. Frame erasure rate (FER) curves for MLT codes and their parent LT codes ($c = 0.05$, $\delta = 0.5$ are the parameters used for the RSD for all the cases).

the fact that the neighbors of the final MLT-4 code symbol are not picked uniformly from the entire set of data of all the four sources. This inaccuracy is present in the case of an MLT-2 code as well, but for MLT-4 codes, it becomes two-fold: it manifests itself in the construction of a DLT-2 symbol from two DLT-4 symbols, as well as in the construction of an LT code symbol from two DLT-2 symbols.

VI. Conclusions

We have developed a new distributed rateless encoding technique for joint transmission of multiple sources over a common relay. In this scheme the received code symbols at the relay are selectively XORed and the resulting code resembles an LT code in both structure and performance. This paper extends previous work in [5] with further aspects and new results. Besides an explicit derivation of MLT-4 codes we simulated the performance of LT, MLT-2, and MLT-4 block codes with fixed rates. For a symbol erasure channel between relay and sink our results reveal that the MLT-2 scheme yields a lower FER compared to an LT block code for each source transmitted individually over the relay-sink channel; this is due to the larger blocklength that the distributed approach makes possible. For the same reason, the MLT-4 scheme outperforms an MLT-2 code. It is also observed that the FER performance of the MLT-2 scheme is better than that of the LT scheme when there are symbol erasures on the source-relay channel.

References

[1] J. W. Byers, M. Luby, M. Mitzenmacher, "A Digital Fountain Approach to Asynchronous Reliable Multicast," *IEEE J. on Selected Areas in Communications*, vol. 20, no. 8, pp. 1528-1540, August 2002.

[2] J. W. Byers, M. Luby, M. Mitzenmacher, A. Rege, "A Digital Fountain Approach to Reliable Distribution of Bulk Data," *Proc. of ACM SIGCOMM '98*, pp. 56-67, Vancouver, Canada, September 1998

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

[4] M. Luby, "LT Codes," *Proc. of the 43rd Annual IEEE Symp. on Foundations of Comp. Sc.*, pp 271-280, Vancouver, Canada, November 2002

[5] S. Puducheri, J. Kliewer, T. E. Fuja, "Distributed LT codes," *Proc. IEEE Int. Symposium on Inform. Theory (ISIT)*, Seattle, WA, July 2006.

[6] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, 2003

[7] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. IT-8, pp. 21-28, January 1962