Near Real-time Rateless Coding with a Constrained Feedback Budget

Morteza Hashemi and Ari Trachtenberg
Dept. of Electrical and Computer Engineering, Boston University, USA

Abstract

We present a *near real-time* rateless coding approach based on the constrained use of a feedback channel. The specific feedback used in our scheme is based on a measure of distance between a received word and the symbols already decoded at the receiver, with successive feedbacks providing accumulating information about the state of the decoder. Our encoder, in turn, optimizes its choice of constituent source symbols in each encoding according to the decoder information and not uniformly at random as is typically done. We provide an upper-bound on the error probability of our approach when used with a maximum-likelihood decoder, and simulation results demonstrate that the coding performance can be tuned according to the feedback budget available.

This work appeared as:

- M. Hashemi and A. Trachtenberg, "Near Real-Time Rateless Coding with a Constrained Feedback Budget", 52nd Annual Allerton Conference on Communication, Control, and Computing, October 2014.

I. Introduction

Traditional rateless codes, such as Luby-Transform (LT) [1] and Raptor codes [2], can often provide reliable communication over an unreliable channel, although they have high coding overhead for short block lengths [3]. These codes also have a well-known all-or-nothing decoding property (the so-called "waterfall" phenomenon), where an abrupt jump in the ratio of decoded input symbols occurs when the coding is almost complete. In order to improve the performance of rateless codes in such circumstances, there has been increasing interest in incorporating the use of a feedback channel. For example, a decoder for Real-Time (RT) oblivious [4] and Shifted-LT (SLT) [5] codes sends the number of recovered symbols back to the transmitter, and this feedback is used to modify the degree distribution at the transmitter.

In this paper, we propose a coding scheme, called *Real-time LT (RLT)*, that uses *distance-type* feedback messages. An RLT encoder utilizes the distance between a received encoding symbol and the set of already decoded symbols at the receiver to learn about the connectivity of the decoder's graph. Specifically, the encoder estimates the probability that each input symbol has been decoded (at the decoder), and these estimates are then used by the encoder to optimize the choice of input symbols within each transmission. Our choice of input symbols is based on a greedy approach that attempts to maximize the decoding progress per transmission, with the result that the expected number of input symbols decoded per received symbol stays mostly uniform throughout the transmission process, an almost *real-time* decoding performance [4]. To make the analysis of RLT codes more tractable, we further suggest a sub-optimal coding scheme, dubbed *Nonuniform-LT (NLT)* codes, wherein input symbols are chosen independently according to a nonuniform selection distribution. For ease of exposition, we base our code construction upon the Robust Soliton (RS) degree distribution as used in LT codes [1], although any degree distribution can be used.

Our previous work [6] has focused on very light-weight feedback messages (i.e., only 1 bit per reception), and this work attempts to generalize that result to larger feedback budgets that may occur in other applications; in so doing, we also allow the decoder to specific control *when* feedback occurs. We demonstrate this feedback flexibility in simulations, supported by improved upper bounds on decoding failures.

The rest of this paper is organized as follows. In Section II, we present a brief background on rateless coding, including a brief literature review. Section III motivates our work and provides our RLT code design methodology, followed by a description of Nonuniform LT codes and their analysis in Section IV. In Section V we compare RLT codes with the current state-of-the-art feedback-based rateless codes, and we conclude with Section VI.

II. Background

In this section, we provide a brief background on rateless coding schemes.

A. Preliminaries

Encoding: In a rateless coding framework, we assume that the encoder has k input symbols to transmit to one or several receivers over an erasure broadcast channel. In the LT codes [1], the encoder first picks a coding degree d according to the Robust Soliton distribution defined based on distributions ρ_k and τ_k:

$$\rho_k(i) = \begin{cases} \frac{1}{k} & i = 1; \\ \frac{1}{i(i-1)} & i = 2, ..., k; \end{cases}$$

and τ_k that is defined as follows: let $R = c \ \ln\left(\frac{k}{\delta}\right)\sqrt{k}$ for some constant c, then:

$$\tau_k(i) = \begin{cases} \frac{R}{ik} & i \le k/R - 1; \\ \frac{R}{k}\ln(R/\delta) & i = k/R; \\ 0 & \text{otherwise.} \end{cases}$$

The Robust Soliton distribution $\Omega_k(i)$ is the normalized sum of $\rho_k(i)$ and $\tau_k(i)$, i.e., :

$$\Omega_k(i) = \frac{\rho_k(i) + \tau_k(i)}{\sum_{j=1}^{k} \rho_k(j) + \tau_k(j)} \quad i = 1, ..., k. \tag{1}$$

Next, d input symbols are chosen uniformly at random without replacement, and their sum over an appropriate finite field forms an encoding symbol transmitted over the channel. Indices of the d selected input symbols are made available (i.e., as meta-information) to the decoder. d constituent input symbols are usually referred to as *neighbors* of the encoding symbol.

Decoding: At the decoder side, the k input symbols can be recovered with high probability from a set of received encoding symbols with a cardinality slightly larger than k. For instance, LT codes support recovery of input symbols from $k+O\left(\sqrt{k}\ln^2(k/\delta)\right)$ encoding symbols with probability $1-\delta$. To this end, the widely-used low complexity peeling decoder [7] is employed. The peeling decoder finds all encoding symbols with degree 1, whose neighbor can be immediately recovered. The recovered input symbols are then excluded from all encoding symbols that have them as neighbors. The decoder proceeds iteratively until all input symbols are recovered or there exists no more encoding symbols with degree 1, in which case decoding fails.

B. Related work

There have been several works to incorporate different types of feedback messages into the traditional rateless codes. Feedback messages in the Real-Time (RT) oblivious codes [4] contain information about the number of recovered symbols at the decoder side. The RT encoder then uses the feedback information to gradually increase the degree of encoding symbols. A similar-in-spirit methodology has been used in the Shifted LT (SLT) codes [5], wherein feedback messages notify about the number of recovered symbols to the transmitter. The degree distribution is then shifted according to the number of recovered symbols. The SLT scheme uses the same encoding/decoding procedure as LT, though it considerably improves the performance of LT codes for short block lengths. Similarly, in Growth codes [8], the receiver feeds information about the number of recovered symbols back to the transmitter, and then the transmitter increases the coding degree to enhance the probability of transmitting innovative symbols.

Another type of feedback message involves the encoder requesting a particular coding degree according to the state of decoder [9]. Such a coding scheme adds flexibility to the decoder to control the decoding progress. It has been shown that to fully recover k input symbols, at most $1.236k$ error-free transmissions are required by this method. Yet another type of feedback contains the identity of recovered symbols [10]. Such information is then used by the encoder to redesign the degree distribution for subsequent transmissions. More recently, the authors in [11] have proposed a heuristic to use a "hybrid" feedback-based rateless, called LT-AF codes, in which the receiver alternates between two types of feedback messages: the first type of feedback contains information about the number of decoded symbols as in the SLT and RT codes, while the receiver requests a specific input symbol through the second type of feedback.

We distinguish the RLT codes from previous works by two key aspects. First, the type of feedback in the RLT codes is based on *distance-type* feedback. We generalize the use of distance messages in [6] by feeding *all distance* messages back every so often. In other words, after collecting a pre-defined number of encoding symbols, the decoder initiates a single feedback message that contains distance information. The encoder learns about the connectivity of the decoding graph by accumulating information across all feedbacks.

Second, in contrast to previous works that are based on uniform symbol selection, we define an optimization framework to select neighbors of encoding symbols such that they result in a maximum decoding progress per transmission. We provide a relaxed version of this optimization based on a nonuniform symbol selection at the encoder. Note that unequal error protection (UEP) rateless codes [12] are also built upon a nonuniform symbol selection distribution that is set based on the importance of source symbols (i.e., more important data are more probable to be chosen), whereas our nonuniform selection distribution is calculated on-the-fly based on feedback information.

III. Near Real-time Rateless Codes

Previous rateless codes with feedback were designed based on modifying the degree distribution according to feedback messages, e.g., by shifting the degree distribution, or by explicitly increasing the degree. However, when a degree d is picked, d input symbols are selected uniformly at random. To enhance the coding performance, input symbols can be selected based on connectivity of the decoder's graph (i.e., state of individual input symbol).

A natural feedback approach involves sending the *identity* of recovered symbols back to the encoder. If the encoder stores all encoding symbols, it can then recover distance information from this feedback, and thus learn about the connectivity of the decoder's graph. However, storing all encoding symbols may be costly, especially when the forward channel is highly lossy. Moreover, it should be noted that initially, when not so many symbols are decoded, this feedback provides little information about the decoding state.

In this paper, we propose distance-type feedbacks that provide information about the decoder's graph through a uniform feedback transmission (i.e., one feedback for every s encoding symbols). In this scheme, the encoder stores only encoding symbols between consecutive feedbacks, resulting in a light-weight encoder. As we shall explain, neighbor selection based on distance information can incorporate the "history" of transmissions to accelerate decoding progress.

A. Distance-type feedback

Definition. *Given an encoding symbol y and a set of recovered symbols $\mathcal{C}$,* ***distance*** *between y and the set $\mathcal{C}$ is the number of neighbors of y that remain undecoded right after receiving y.*

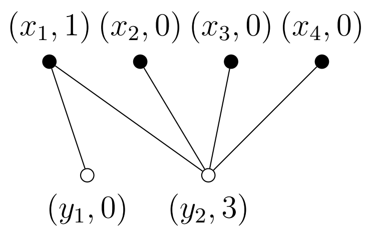


Fig. 1. Distance graph labeling: A label (x_i, q_i) implies that the input symbol x_i has been decoded with the probability of q_i up to the current state. Labels of check nodes y_i are defined to be the number of neighbors of y_i with a label of less than 1.

Example: Suppose that input symbols x_1, .., x_4 are encoded and transmitted in the following order: $y_1 = x_1 + x_2$, $y_2 = x_1 + x_4$, $y_3 = x_4$, and $y_4 = x_1 + x_2$. Based on distance definition, y_1 and y_2 have distance 2, while y_3 and y_4 have distance 0 (x_4, x_1 and x_2 are decoded after receiving y_3 and y_4).

From the definition of distance feedback, it is clear that some of the distance messages provide explicit information about the decoding state at the decoder. For instance, distance 0 happens when all neighbors of the encoding symbol have already been decoded, or can be decoded by an XOR operation using the incoming encoding symbol, and thus, the encoder has full knowledge that all neighbors corresponding to the symbol with distance 0 are decoded. Moreover, if the distance of an encoding symbol is equal to its degree (i.e., number of neighbors), then the transmitter infers that none of the symbol's neighbors are recovered at the decoder side. On the other hand, some distance feedbacks provide implicit information in the sense that the encoder would not be able to uniquely label neighbors of an encoding symbol as decoded or not-yet-decoded. In such cases, the encoder assigns a probability of having been decoded to the input symbols. For instance, if the encoding symbol $y = x_1 + x_2 + x_3 + x_4$ has distance 2 with the current state of decoder, the transmitter infers that each input symbol $x_1, ..., x_4$ has been decoded with the probability of $\frac{2}{4}$. These probabilities would be updated after receiving a new feedback message.

In order to limit the number of feedback transmission, we assume that the receiver is limited to send one feedback after collecting every s encoding symbols, and that each feedback message contains distance information corresponding to encoding symbols received between consecutive feedbacks. Such a mechanism provides a uniform feedback transmission that prevents congestion on the back channel. Moreover, it adds an additional flexibility to the coding scheme in that the parameter s can be set to any arbitrary value depending on the feedback channel available. In this model, we assume that the cost of feedback is mostly associated with the number of transmissions, and not the length of each message.

Processing distance information: In order to process distance information, the encoder constructs a bipartite graph wherein input symbols are placed on the top, and encoding symbols at the bottom as shown in Fig. 1. The input symbols' labels correspond to their probability of having been decoded. Each encoding symbol y_i is also associated with a label that represents the number of neighbors of y_i with labels less than 1. Specifically, assume that after the t-th feedback, $n_i^{(t)}$ neighbors of y_i are labeled 1 (i.e, they have been decoded). The label of y_i denoted by $l_i^{(t)}$, is then calculated as:

$$l_i^{(t)} = d_i - n_i^{(t)}, \tag{2}$$

where d_i is the degree of y_i. We assume that the t-th feedback message contains the distance value of $f^{(t)}$ corresponding to the encoding symbol $y_i = \sum_{j \in A_i} x_j$. The label of a constituent symbol x_j is then estimated as:

$$q_j^{(t)} = \max \left\{ q_j^{(t-1)}, \frac{l_i^{(t-1)} - f^{(t)}}{l_i^{(t-1)}} \right\}. \tag{3}$$

In (2), the encoder excludes the recovered symbols (i.e., $q_j = 1$) from the labeling process and then it uniformly "shares" the distance between remaining constituent symbols as in (3). For instance, if the encoding symbol $y = x_1 + x_2 + x_3 + x_4$ has distance 2 with the current state of decoder, and the encoder

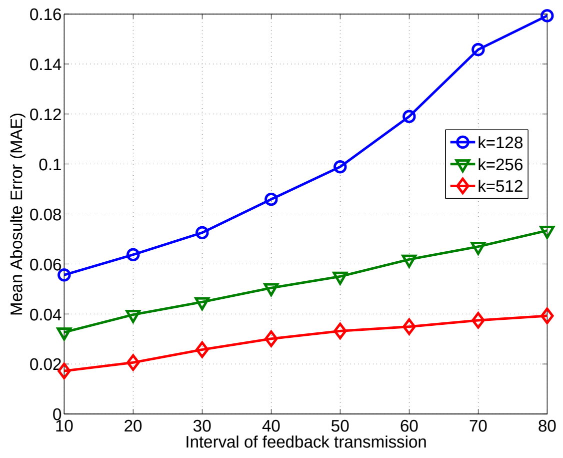


Fig. 2. Mean Absolute Error (MAE) for the estimated probability of having been decoded as the interval of feedback transmission (parameter s) increases.

has already assigned label of 1 to x_1 (i.e., x_1 has been decoded), then the encoder uniformly shares the distance of 2 between the remaining symbols (i.e., x_2, x_3, and x_4) and updates their probability of having been decoded to $\frac{3-2}{3} = \frac{1}{3}$.

We use the Mean Absolute Error (MAE) quantity to investigate the accuracy of the estimated probability of having been decoded against the actual decoder state. In particular, assume that the variable b_j represents the state of the symbol j at the decoder such that $b_j = 1$ if the symbol j has been decoded and $b_j = 0$ otherwise. MAE is then defined as:

$$\text{MAE} = \frac{1}{k}\sum_{j=1}^{k}|q_j - b_j|,$$

in which q_j's are estimated using (3). Fig. 2 shows the MAE quantity averaged over all feedback messages transmitted as the interval of feedback transmission (i.e., parameter s) increases. The results (averaged over 1000 times run) illustrate that decreasing the interval of feedback transmission (i.e., higher feedback rate) decreases the estimation error, and that the labeling mechanism provides a low-error estimation of the decoder state.

Next, we formulate an optimization framework that incorporates the probability values q_j's in the neighbor selection of encoding symbols.

B. Maximum decoding progress

The encoder estimates the probability of having been decoded q_j's based on distance feedbacks. Constituent symbols within an encoding symbol are then selected in such a way that they provide maximum decoding progress, which is recovering exactly one input symbol per transmission.

Example: Assume that we want to transmit a codeword with degree 1 based on the current value of q_j's. Transmitting the next encoding symbol consisting of only the individual input symbol j, would be innovative with the probability of $1 - q_j$. Intuitively, to achieve a maximum decoding progress per transmission, the encoder needs to transmit an input symbol j^* where: $j^* = \arg\max_j \ (1 - q_j) = \arg\min_j \ q_j$.

From the example, one can see that if the encoder is restricted to transmit only degree one symbols, then this coding scheme prevents the coupon collector phenomenon [13], in that the recovered symbols would have $q = 1$, and thus, they will never be chosen for subsequent transmissions. For the general case, the encoder first picks the coding degree d based on the Robust Soliton distribution, and then d input symbols are chosen to provide a maximum probability of decoding a single input symbol per transmission.

Definition. *For a given input symbol x and a set of $d-1$ input symbols A, the Decoding Probability function $DP(x,A)$ is defined as the probability of immediate decoding the input symbol x transmitting $y = x + \sum_{j \in A} x_j$.*

In order to decode an input symbol x_i within a transmission, the transmitted symbol with the degree of d should include the undecoded symbol x_i and $d-1$ already decoded symbols. Symbol x_i is not decoded with the probability of $1-q_i$, and $d-1$ symbols belonging to the set A have already been decoded with the probability of $\prod_{j \in A} q_j$. Therefore, at each step to transmit a symbol of degree d, the encoder chooses d input symbols (x_i^*, A^*) satisfying:

$$(x_i^*, A^*) = \underset{(x_i,A)}{\arg\max}\ DP(x_i, A) = \underset{(x_i,A)}{\arg\max}\,(1-q_i) \prod_{\substack{j \in A \\ j \neq i}} q_j. \tag{4}$$

In order to solve this optimization, the encoder picks an input symbol that maximizes the value of $(1-q_i)$, combined with $d-1$ symbols with largest q_j's (ties are broken randomly). This scheme is a "soft" version of generating encoding symbols when the encoder knows exactly which symbols are decoded.

C. Distance feedback properties

Cumulative feedback information: Distance-type feedback has a favorable property in that it accumulates information across all received feedback messages. Specifically, assume that there exist k input symbols at the encoder, and after receiving a new feedback message, say the t-th feedback, the encoder updates the probability vector $\mathbf{q}^{(t)} = \left(q_1^{(t)}, q_2^{(t)}, ..., q_k^{(t)}\right)$, wherein $q_j^{(t)}$ is the probability of having been decoded for the input symbol j. The encoder updates probability values corresponding to the encoding symbol's neighbors, and other probability values remain unchanged. This update mechanism allows the encoder to accumulate information across all feedback messages, noting that in previous feedback schemes (e.g., sending number of recovered symbols in [4, 5]), a new feedback makes previously received feedback information obsolete.

Decoder's graph information: In most of previous works, feedback messages fail to provide information about the state of individual input symbols at the decoder side. For instance, sending the number of recovered symbols does not make the decoded symbols known to the encoder. The authors in [10] use feedback messages that contain the *identity* of recovered symbols, wherein each symbol is labeled as either decoded or not-yet decoded at the encoder; in other words, there are binary labels to distinguish between symbols. Likewise, light-weight feedbacks in the Delete-and-Conquer codes [6] provide information on decoded symbols. In these schemes, all of the undecoded symbols are considered to be the same from the encoder's perspective as they have the same probability of being included in subsequent transmissions. However, the key point is that based on the history of transmissions (i.e., connectivity of the decoder's graph), input symbols are not the same in terms of decoding progress.

Distance feedbacks can distinguish between undecoded symbols such that neighbors of an encoding symbol with a smaller degree have a higher probability of being included in subsequent transmissions. This feature is desirable as it provides faster decoding progress compared with when neighbors of a high-degree symbol are included in a transmission. For example, consider two encoding symbols $y_1 = x_1 + x_2 + x_3$ and $y_2 = x_4 + x_5$, and assume that the next transmission is of degree one. If symbol x_1, x_2, or x_3 is used in the next transmission, the decoder can only recover a single input symbol, whereas sending either x_4 or x_5 results in recovering two symbols. In the following we show that distance-type feedback prioritizes neighbors of encoding symbols with a smaller degree. From (3), label of an input symbol j is related to:

$$q_j = \frac{l_i - f}{l_i} = 1 - \frac{f}{l_i},$$

where l_i represents the degree of the encoding symbol i. Note that by increasing the coding degree l_i (for a fixed value of f), the probability $1-q_j$ of selecting the undecoded neighbor j decreases. Therefore, distance feedbacks assign higher selection probability to neighbors of low-degree encoding symbols, which boosts the decoding progress.

Distance feedbacks provide information about the decoder's graph, but it should be noted that one can envision other schemes to learn about the decoding graph. For instance, the decoder can send the whole decoding graph back to the encoder, and thus, the encoder would have full knowledge about the state of input symbols at the decoder's side. This method, however, incurs high communication overhead on the back channel.

IV. Maximum Likelihood Decoder Failure Probability

In the previous section, we established an optimization framework that can achieve maximum decoding progress per transmission. For the purpose of analysis, however, it is challenging to work with the solution in (4). Therefore, we suggest a relaxed version of the RLT codes, referred as Nonuniform LT (NLT), in which symbols are selected *independently* according to a nonuniform selection distribution. Next, we derive an upper bound on the decoder failure probability of the NLT codes. Our simulation results show that the RLT codes can achieve substantially better performance than the NLT codes, and thus the calculated upper bound can serve as a bound for the RLT codes as well.

A. Nonuniform Rateless Coding

We use the probability values q_j to define a nonuniform symbol selection distribution as follows:

$$p_j := \frac{\max(q_j, 1-q_j)}{\sum_i \max(q_i, 1-q_i)}, \tag{5}$$

in which, p_j is the probability of selecting the input symbol j as the neighbor of an encoding symbol. Therefore, the encoder first picks a coding degree d based on the Robust Soliton distribution[1], then with the probability of p_j an individual symbol j is selected as one of d symbols. Note that this symbol selection distribution assigns a higher weight to those symbols that the encoder has less uncertainty about them. In other words, the encoder assigns higher selection probability on symbols with q_j close to 1 (i.e., the symbol has been decoded almost surely (a.s.)) or symbols with q_j close to 0 (i.e., the symbol has not been decoded a.s.).

The nonuniform symbol selection p approximates the behavior of the optimized symbol selection in (4), as the symbol j is more likely to be a solution for the optimization problem in (4) if it has either q_j close to 1 or q_j close to 0, which is qualitatively the same scheme as in the distribution p defined in (5).

B. Maximum-Likelihood Decoder Failure Probability

In this section, we derive an upper bound on the maximum likelihood (ML) decoder failure probability of the NLT codes with feedback. The ML decoding over the binary erasure channel (BEC) is equivalent to recovering k information (input) symbols (or for simplicity we assume that each symbol contains a single bit) from n received output bits. Assume that x is a row vector containing k input bits and y is the vector of n collected output bits. Matrix $H = [h_{i,j}]$ is an $n \times k$ adjacency matrix of the decoder's graph, where an entry $h_{i,j}$ is equal to 1 if the i-th output node has the j-th input node as a neighbor. The ML decoder is then equivalent to solving a system of linear equations in the form of:

$$Hx^T = y^T. \tag{6}$$

Equation (6) has a unique solution if and only if the matrix H has full rank. To calculate an upper bound on the decoder failure probability of the NLT codes, we assume that the decoder is restricted to transmit a single feedback when it successfully collects m encoding symbols. After the feedback transmission,

[1]The use of the Robust Soliton distribution is arbitrary and any distribution can be used.

input symbols are selected nonuniformly according to (5). Note that our analysis can be extended to more than one feedback message in a straightforward manner, and that the derived upper bound for a single feedback serves as a bound for more than one feedback as well.

Theorem 1 Assume that the NLT encoder uses the degree distribution $\Omega_k(d)$, and that the decoder collects n encoding symbols. The decoder is allowed to transmit a single feedback message when it successfully collects m symbols. An upper bound on the ML decoder failure probability of the NLT codes is given by:

$$p_e \leq \min\left\{1, \sum_{w=1}^{k} \binom{k-1}{w-1} \left(\sum_d \Omega_k(d) \frac{\sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s}\binom{k-w}{d-s}}{\binom{k}{d}}\right)^m \left(\sum_d \Omega_k(d) \sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s}\binom{k-w}{d-s}\left(\frac{2}{k}\right)^d\right)^{n-m}\right\}. \tag{7}$$

Proof: Assume that $I = \{i_1, i_2, ..., i_w\}$ is the set of indices in which the input vector x is equal to 1, implying that x has Hamming weight w. Let p_e be the probability that an input bit i (for an arbitrary $i \in \{1, 2, ..k\}$) can not be decoded under the ML decoding rule. Similar to [12], we have:

$$\begin{aligned} p_e &= \Pr\left\{\exists x \in GF(2^k), x(i) = 1 : \quad Hx^T = 0^T\right\} \\ &\leq \sum_{\substack{x \in GF(2^k) \\ x(i)=1}} \Pr\left\{Hx^T = 0^T\right\}. \end{aligned}$$

In order to calculate $\Pr\{Hx^T = 0^T\}$, we consider the rows of matrix H before and after receiving the feedback separately. First, let's denote the rows of H before feedback by H_1. Rows in H_1 are generated independently according to the degree distribution $\Omega_k(d)$. Thus, considering a random binary vector R_1, we have:

$$\Pr\{H_1 x^T = 0^T\} = \left(\Pr\{R_1 x^T = 0^T\}\right)^m.$$

The event $R_1 x^T = 0^T$ happens if and only if R_1 has an even number of 1's in those indices of i in which $x(i)$ is equal to 1 as well (i.e., $i \in I$). Given that row R_1 has degree d, we have:

$$\Pr\{R_1 x^T = 0^T\} = \frac{\sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s}\binom{k-w}{d-s}}{\binom{k}{d}}.$$

Because each row of H_1 has degree d with the probability of $\Omega_k(d)$ we have:

$$\Pr\{H_1 x^T = 0^T\} = \left(\sum_d \Omega_k(d) \frac{\sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s}\binom{k-w}{d-s}}{\binom{k}{d}}\right)^m. \tag{8}$$

Next, let's consider the rows of H after the feedback, which are denoted by H_2. After receiving the feedback, neighbors of output symbols are selected according to the nonuniform selection distribution p defined in (5). Considering a row R_2 belonging to H_2, the probability of choosing an even number of input symbols from the indices in which $x(i)$ is equal to 1 is as follows:

$$\Pr\{R_2 x^T = 0^T\} = \sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \Big(\sum_{A \in F_s} \prod_{i \in A} p_i\Big)\Big(\sum_{B \in G_{d-s}} \prod_{j \in B} p_j\Big).$$

F_s is the set of all subsets of I with the cardinality of s, and G_{d-s} is the set of all subsets of complement of I with $d-s$ elements. Defining A_s^* and B_{d-s}^* as:

$$A_s^* = \arg\max_{A \in F_s} \prod_{i \in A} p_i \quad \text{and} \quad B_{d-s}^* = \arg\max_{B \in G_{d-s}} \prod_{j \in B} p_j,$$

we have:

$$\Pr\{R_2 x^T = 0^T\}$$
$$\leq \sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s} p_{i_1} p_{i_2} ... p_{i_s} \times \binom{k-w}{d-s} p_{j_1} p_{j_2} ... p_{j_{d-s}},$$

wherein $A_s^* = \{i_1, i_2, ..., i_s\}$ and $B_{d-s}^* = \{j_1, j_2, ..., j_{d-s}\}$. Recall that $\forall j$:

$$p_j = \frac{\max(q_j, 1-q_j)}{\sum_i \max(q_i, 1-q_i)} \leq \frac{1}{\sum_i \frac{1}{2}},$$

which results in the following inequality:

$$\Pr\{R_2 x^T = 0^T\} \leq \sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s} \binom{k-w}{d-s} \left(\frac{1}{\sum_{i=1}^{k} \frac{1}{2}} \right)^d$$
$$= \sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s} \binom{k-w}{d-s} \left(\frac{2}{k} \right)^d .$$

Similar to the first part, we have:

$$\Pr\{H_2 x^T = 0^T\} =$$
$$\left(\sum_d \Omega_k(d) \sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s} \binom{k-w}{d-s} \left(\frac{2}{k} \right)^d \right)^{n-m} . \quad (9)$$

(8) and (9) lead to the theorem statement. ■

Corollary 1. Assuming $m = \frac{n}{2}$, the upper bound on the ML decoder failure probability of the NLT codes is tighter than the corresponding bound of the LT codes.

Proof: From [12], the ML decoder failure probability of the LT codes is upper-bounded by:

$$p_e \leq \min \left\{ 1, \sum_{w=1}^{k} \binom{k-1}{w-1} \right.$$
$$\left. \left(\sum_d \Omega_k(d) \frac{\sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s} \binom{k-w}{d-s}}{\binom{k}{d}} \right)^n \right\}$$

From (7) and in order to show that the NLT codes have a tighter upper bound, the following inequality should hold:

$$\left(\sum_d \Omega_k(d) \sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s} \binom{k-w}{d-s} \left(\frac{2}{k} \right)^d \right)^{\frac{n}{2}}$$
$$\leq \left(\sum_d \Omega_k(d) \frac{\sum_{s=0,2,..,2\lfloor \frac{d}{2} \rfloor} \binom{w}{s} \binom{k-w}{d-s}}{\binom{k}{d}} \right)^{\frac{n}{2}} .$$

Or equivalently: $\left(\frac{2}{k}\right)^d \leq \frac{1}{\binom{k}{d}} \Leftrightarrow \frac{2^d k!}{d!(k-d)!} \leq k^d$. The last inequality holds $\forall k$ and $\forall d \geq 4$. Thus, independent of the choice of the degree distribution $\Omega_k(d)$ and $\forall d \geq 4$, NLT provides a smaller upper bound on the ML decoder failure probability compared with the LT codes. ■

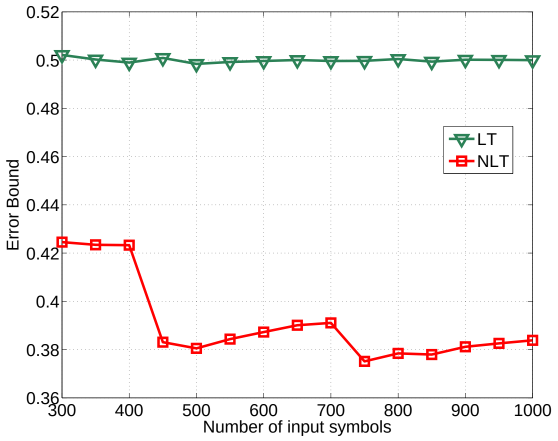


Fig. 3. Comparison of upper bounds on the ML decoder failure probability for the LT and NLT codes ($c = 0.5$, $\delta = 0.1$ in the Robust Soliton distribution).

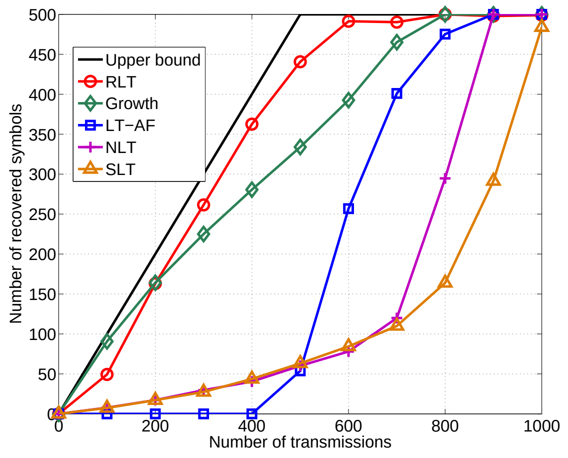
From the derived upper bound one can conclude that by making the degree distribution denser the upper bound on the ML failure probability decreases. But, a denser degree distribution incurs higher encoding/decoding computational complexity. In Fig. 3, we numerically compare the upper bound (normalized with respect to the powers $\frac{n}{2}$) of the LT and NLT codes with the Robust Soliton distribution in which d can take all values (i.e., d is not restricted to be greater than or equal to 4) . The results illustrate that for the general case NLT still provides a tighter upper bound on the decoder failure probability, and that the upper bound of the NLT codes decays faster than the LT codes as the block length k increases.

V. Simulation results

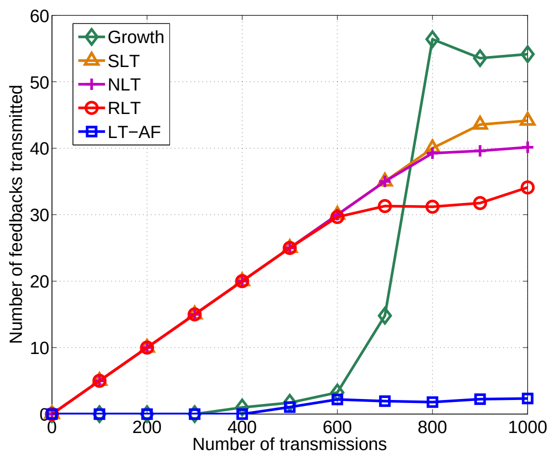
In this section we compare the performance of the RLT codes against its relaxed version Nonuniform LT (NLT) codes, and against the Shifted LT (SLT), Growth, and LT-AF codes of Variable Node with Maximum Degree (LT-AF+VMD) in [11]. The parameters of the Robust Soliton distribution are set as $c = 0.5$ and $\delta = 0.1$, and each simulation is run 1000 times.

A. Intermediate performance

For real-time applications, it is desirable that input symbols experience a small decoding delay, meaning that the decoder should be able to recover some symbols before the decoding process is almost complete. An RLT encoder maximizes the probability of generating encoding symbols that include only a single unrecovered symbol, and thus, with each transmission the decoder should be able to decode an input symbol with high probability. Intermediate performance of different codes for the block length of $k = 500$ is shown in Fig. 4(a), which illustrates that the RLT codes enjoy a near real-time decoding compared with other schemes. Furthermore, one can notice that the NLT codes achieve a better performance than the SLT codes. In these simulations, we configure the SLT, RLT, and NLT decoders to transmit one feedback message after receiving every 20 forward messages (i.e., $s = 20$). Fig. 4(b) shows the amount of feedback transmitted by each coding scheme. The results show that the RLT codes achieve a near real-time performance with a uniform (i.e., constant rate) feedback transmission. A constant-rate feedback is desirable as it prevents congestion on the back channel.



(a) Intermediate performance for $k = 500$



(b) Number of feedbacks transmitted

Fig. 4. (a) Intermediate performance for $k = 500$. SLT, RLT and NLT decoders are set to transmit one feedback after receiving every 20 encoding symbols. (b) Number of feedback messages transmitted.

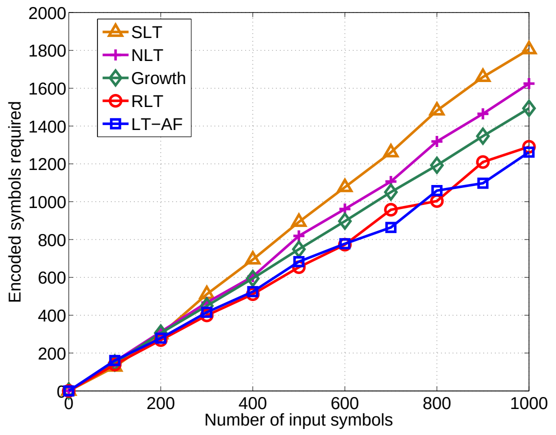


Fig. 5. Total number of transmissions for successful recovery of all symbols.

B. Coding overhead

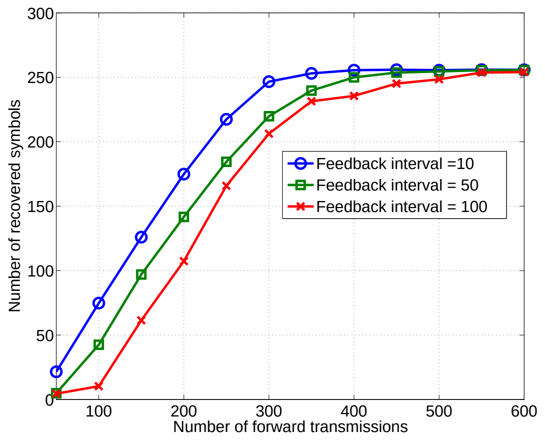
We consider coding schemes that are able to recover all k input symbols from $(1+\epsilon)k$ error-free transmissions. The coding overhead (i.e., total number of forward transmissions) required by different rateless codes is shown in Fig. 5. The results illustrate that the RLT codes have similar performance to the LT-AF codes, while it supports full recovery of input symbols with less forward transmissions compared with other schemes.

C. Tunable performance

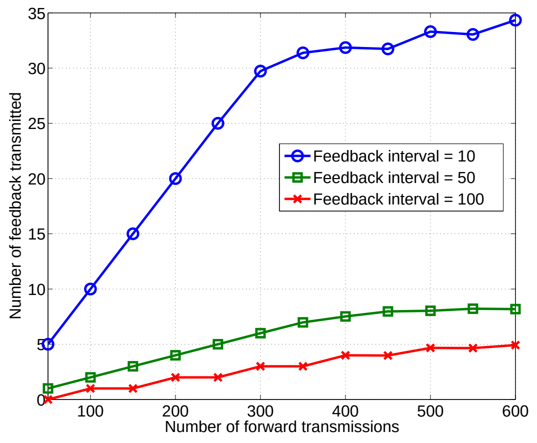
In most of previous works, decoder usually does not have full control over the total amount of feedback transmitted, or the point at which a feedback message is initiated. For instance, in [10], the point at which the feedback is initiated needs to be optimized. In this context, an RLT decoder can adjust not only the point of feedback transmission but also the rate of feedback transmission based on the characteristics of the feedback channel. Fig. 6(a) shows the intermediate performance of the RLT codes with the block length of $k = 256$ and different feedback transmission interval (i.e, inverse of the feedback transmission rate). The results illustrate that the real-time feature of the RLT codes can be tuned based on the available feedback rate. Fig. 6(b) shows the amount of feedback transmitted vs. the number of forward messages received by the decoder for $k = 256$ input symbols. In terms of total coding overhead, the performance of the RLT codes can be tuned based on the rate of the feedback transmission as well. Fig. 7(a) shows the performance of the RLT codes as the feedback interval increases and Fig. 7(b) depicts the number of feedback messages transmitted as the feedback interval increases. The results demonstrate that by increasing the feedback interval, more forward transmissions are needed, whereas the number of feedback messages decreases, which implies an intrinsic trade-off between the number of forward and feedback transmissions.

VI. Conclusion

In this paper, we proposed the *near Real-time* LT (RLT) codes based on the distance feedback and a greedy code construction to optimize the selection of neighbors of encoding symbols, which results in a near real-time decoding performance and reduced coding overhead. The RLT codes add two essential features to the rateless coding process in that the decoder has full control over the *rate* of feedback and the *points* of feedback transmission. As a relaxed version of the RLT codes, we presented the Nonuniform LT (NLT) codes built based on a nonuniform neighbor selection distribution. Our analysis show that the NLT codes provide a tighter upper bound on the maximum likelihood decoder failure probability compared with the LT codes. In future works, we plan to characterize maximum performance gains that a fixed limited feedback budget provides.

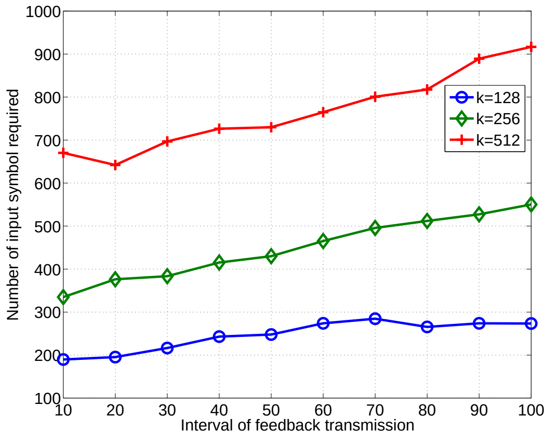


(a) Intermediate performance for $k = 256$

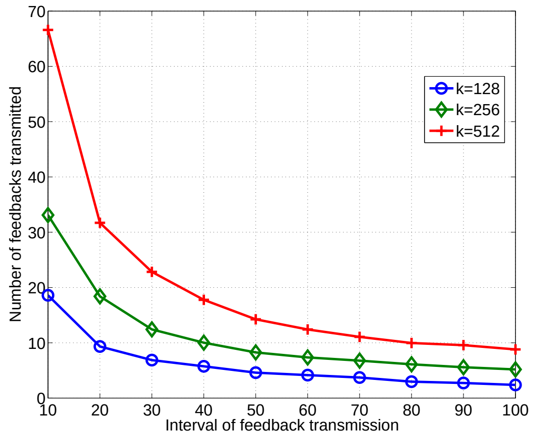


(b) Number of feedbacks transmitted

Fig. 6. (a) Intermediate performance of the RLT codes for $k = 256$ (b) Number of feedback messages transmitted vs. the number of forward transmissions



(a) Total number of forward transmissions required



(b) Number of feedbacks transmitted

Fig. 7. (a) Total number of forward transmissions required by the RLT codes (b) Number of feedback messages transmitted as the interval of feedback increases.

Acknowledgment

The authors would like to thank Dr. Yuval Cassuto for his valuable suggestions and comments.

References

[1] Michael Luby. LT codes. In *Annual Symposium on Foundations of Computer Science*, pages 271–280, 2002.
[2] Amin Shokrollahi. Raptor codes. *IEEE Trans. Inf. Theory*, 52(6):2551–2567, 2006.
[3] Esa Hyytia, Tuomas Tirronen, and Jorma Virtamo. Optimal degree distribution for LT codes with small message length. In *IEEE INFOCOM*, pages 2576–2580, 2007.
[4] Amos Beimel, Shlomi Dolev, and Noam Singer. RT oblivious erasure correcting. *IEEE/ACM Trans. on Networking*, 15(6):1321–1332, 2007.
[5] Andrew Hagedorn, Sachin Agarwal, David Starobinski, and Ari Trachtenberg. Rateless coding with feedback. In *IEEE INFOCOM*, pages 1791–1799, 2009.
[6] Morteza Hashemi, Ari Trachtenberg, and Yuval Cassuto. Delete-and-conquer: Rateless coding with constrained feedback. In *51th Annual Allerton Conference. IEEE*, 2013.
[7] Michael G Luby, Michael Mitzenmacher, Amin Shokrollahi, Daniel A Spielman, and Volker Stemann. Practical loss-resilient codes. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 150–159. ACM, 1997.
[8] Abhinav Kamra, Vishal Misra, Jon Feldman, and Dan Rubenstein. Growth codes: Maximizing sensor network data persistence. In *ACM SIGCOMM*, volume 36, pages 255–266, 2006.
[9] Yuval Cassuto and Amin Shokrollahi. On-line fountain codes for semi-random loss channels. In *Information Theory Workshop (ITW), 2011 IEEE*, pages 262–266, 2011.
[10] Jesper Hemming Sørensen, Toshiaki Koike-Akino, and Philip Orlik. Rateless feedback codes. In *IEEE International Symposium on Information Theory*, pages 1767–1771, 2012.
[11] Ali Talari and Nazanin Rahnavard. Robust LT codes with alternating feedback. *Computer Communications*, 2014.
[12] Nazanin Rahnavard, Badri N Vellambi, and Faramarz Fekri. Rateless codes with unequal error protection property. *Information Theory, IEEE Transactions on*, 53(4):1521–1532, 2007.
[13] Rajeev Motwani. *Randomized algorithms*. Cambridge university press.