# Decoding of LT-Like Codes in the Absence of Degree-One Code Symbols

Nadhir I. Abdulkhaleq and Orhan Gazi

**Luby transform (LT) codes were the first practical rateless erasure codes proposed in the literature. The performances of these codes, which are iteratively decoded using belief propagation algorithms, depend on the degree distribution used to generate the coded symbols. The existence of degree-one coded symbols is essential for the starting and continuation of the decoding process. The absence of a degree-one coded symbol at any instant of an iterative decoding operation results in decoding failure. To alleviate this problem, we proposed a method used in the absence of a degree-one code symbol to overcome a stuck decoding operation and its continuation. The simulation results show that the proposed approach provides a better performance than a conventional LT code and memory-based robust soliton distributed LT code, as well as that of a Gaussian elimination assisted LT code, particularly for short data lengths.**

**Keywords: Rateless coding, LT erasure codes, Degree-one, Tanner graph, pattern recognition.**

Manuscript received Feb. 17, 2016; revised May 9, 2016; accepted May 16, 2016.

Nadhir I. Abdulkhaleq (corresponding author, c1282650@student.cankaya.edu.tr) and Orhan Gazi (o.gazi@cankaya.edu.tr) are with the Department of Electronics and Communication, Cankaya University, Ankara, Turkey.

## I. Introduction

Rateless codes were introduced by Luby and Byers in 1998 [1], and are an efficient invention that provides a reliable solution to the problems of an automatic repeat request protocol, particularly in erasure channels. Luby transform (LT) codes were the first rateless erasure codes proposed in the literature. After the invention of LT codes, Shokrollahi used a cascading scheme consisting of a pre-code such as an LDPC block code followed by the well-known LT code, forming a new generation of rateless codes called Raptor codes [2]. LT codes are considered capacity approaching erasure codes. Rateless codes produce a limitless amount of encoded symbols from $k$ source symbols. The receiver is capable of decoding when the number of received symbols is slightly greater than $k$. Because the transmission continues until the receiver obtains $k$ symbols, the rate of the code is not fixed, and hence the term “rateless” is used for these codes. For fixed rate codes such as block codes, the rate is chosen to overcome the worst channel conditions. This occasionally brings about unnecessary overhead when a channel is in a good state. On the other hand, rateless codes bring about a comfortable flexibility for the code rate, that is, for the transmission overhead [3].

Luby transform codes have small encoding and decoding complexities, and $k$ source symbols can be recovered using a message passing algorithm with an average decoding complexity on the order of $k \cdot \ln(k/\delta)$ operations for a probability of successful decoding of $1 - \delta$. In addition, LT codes are universal and fit any erasure channel, and their performance improves as the overhead length increases [1], [4].

For the generation of LT codes, the source file is first sliced into $k$ source symbols $(u_1, u_2, \ldots, u_k)$. A source symbol can be a single bit or a group of bits. A distribution, $\Omega(d)$, called a

  
http://dx.doi.org/10.4218/etrij.16.0116.0107

degree distribution is used to generate a random digit $d$, which is called a degree. For each generated degree, a set of source symbols are uniformly chosen and exclusively ORed to form an encoded symbol $C_i$. The generated symbols are transmitted over a binary erasure channel. The channel erases some of these transmitted symbols, and the amount of loss varies according to the erasure probability of the channel, that is, $\alpha$. The decoder can start decoding if the degree information and set of neighbors of each encoding symbol are available at the receiver [5]. When $N$ coded symbols are received such that at least a single coded symbol is of degree one, a message-passing algorithm can be utilized for a decoding operation. The message-passing algorithm works in an iterative manner among the coded symbols in order to recover $k$ source symbols. The decoder either successfully decodes all source symbols or declares a failure, which depends on the selection of information symbols while forming the coded symbols. The decoder fails if a coded symbol of degree one cannot be found before the recovery of all information symbols. The degree distribution, which affects the encoding and decoding efficiency, plays a critical role in the design of LT codes [6].

The rapid surge of data flows over the Internet has encouraged researchers to focus on fountain codes with low encoding and decoding complexities when compared to previously known classical codes, such as such as Reed-Solomon block erasure codes [7]. Significant progress has been made on the design of LT codes. Some studies have been conducted to enhance the degree distribution to improve the overall performance of the code. A good degree distribution should provide a sufficient number of degree-one coded symbols. Moreover, probabilities of low degree values should be high enough that the decoding can continue until the last information symbol is recovered. The ripple size is the number of degree-one coded symbols available in each decoding step. In addition, the degree distribution should provide a large probability value for a high value degree. This is necessary to guarantee that all information symbols are used during the encoding operation.

A robust soliton distribution (RSD), one of the distributions satisfying all of the above mentioned constrains, is used to generate the degree values for coded symbols. Studies on improving the RSD are available in the literature. A reshaped ripple size with an adjusted RSD was proposed by Yen and others [8], where the ripple size is controlled in a manner such that its value is as close as possible to data length $k$ during the BP decoding. In this way, the successful decoding rate is increased compared to that of a conventional RSD. Lu and others [9] proposed a decoding algorithm called LT-W. Wiedemann's method was also applied to the LT decoding process to extend its decoding ability [9]. The authors proved that this approach reduces the packet overhead and supports the efficiency of the original LT decoding process. In addition, memory-based RSD (MBRSD) was investigated by Hayajneh and others [10]. For MBRSD, a degree-one coded symbol choses the information symbol whose degree is the maximum of all information symbols during the encoding operation. In this way, the complexity of the decoding process is reduced. In addition, MBRSD decreases the error floor compared to that of a conventional LT code, particularly when combined with the decreased ripple size approach of Sorenson [3].

In all of the previous works mentioned above, the decoders can suffer from the absence of a degree-one code symbol at any stage and declare a failure. We focused on this issue and looked for some ways to continue the decoding operation even in the absence of degree-one coded symbols.

As the main motivation of this paper, we present new pattern-based decoding operations in which we intend to prove that the use of some of the patterns in a Tanner graph connection decoding operation can still be continued even in the absence of a degree-one coded symbol, which is the reason for the declaration of a decoding failure for conventional LT decoders.

The rest of this paper is organized as follows. In Section II, background information for conventional LT codes are provided. In Section III, the proposed pattern-based approach for the decoding of fountain-like codes in the absence of a degree-one information symbol is discussed. The simulation results for the error rate and decoding probability performance of LT-RSD, LT-MBRSD, and the proposed LT-PR are presented in Section IV. Finally, some concluding remarks are drawn in Section V.

## II. Background

In this section, we provide brief information regarding the encoding and decoding procedure of LT codes. Detailed information on the encoding and decoding of LT codes can be found in [1], [3], and [4].

### 1. Encoding

The first step of LT encoding is to divide a data file into packets. A degree $d$ is then generated according to the distribution, and the encoded packet is generated by XORing the $d$ source packets, which are chosen uniformly at random. A packet can contain a single bit or a group of bits.

Mathematically, the encoding operation can be illustrated as

$$C = uG, \tag{1}$$

where $u$ represents the information packets, and $G$ is a generator matrix, whose column size is a constant number and

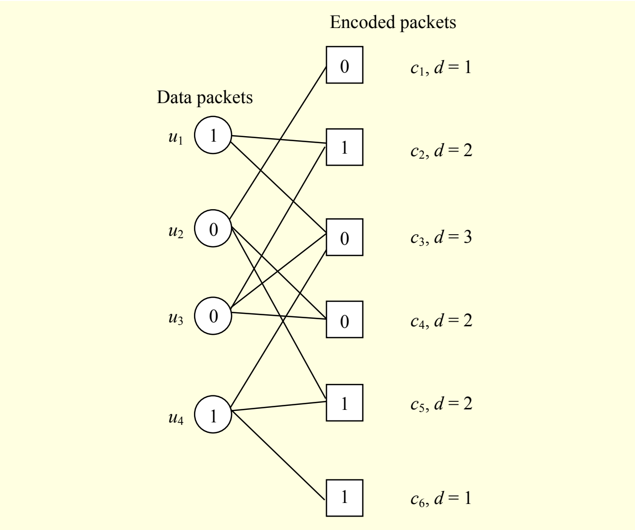


Fig. 1. Tanner graph representation of LT code for $k$ = 4 and $N = 6$.

its row size is a variable number whose value depends on the time instant at which an acknowledgement signal is sent from the receiver side. In addition, $G$ is a binary matrix. The 1s in the columns of $G$ indicate the information packets chosen during the generation of the corresponding coded packet. The number of 1s in a column of $G$ represents the total number of information packets chosen to generate the coded packet, that is, the degree of the coded packet.

Example: An information file is divided into four packets (bits), which are (1 0 0 1), and using a degree generator, the columns of $G$ can be formed as

$$G = \begin{bmatrix} 0\,1\,1\,0\,0\,0\cdots \\ 1\,0\,0\,1\,1\,0\cdots \\ 0\,1\,1\,1\,0\,0\cdots \\ 0\,0\,1\,0\,1\,1\cdots \end{bmatrix}.$$

Then, using $C = uG$, the coded packets are found as

$$C = [0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \cdots]. \tag{2}$$

Tanner graphs are used to graphically illustrate the encoding and decoding operations of LT codes. The connections between the data packets are usually shown on the left-hand side, and the code packets are usually shown on the right-hand side. In (2), assuming that only six code packets are generated, a Tanner graph illustration can be given as in Fig. 1.

After the reception of a sufficient number of encoded packets, the decoder starts decoding the received packets through the use of the message-passing algorithm, which includes a message exchange between the left- and right-hand sides of the Tanner graph. For a data file consisting of $k$ source packets, the decoder will achieve an average of $O\left(k \cdot \ln(k/\delta)\right)$ packet operations. The $k$ source symbols can be recovered from any $k + O\left(\sqrt{k}\,\ln^2(k/\delta)\right)$ encoded symbols with a probability of $1-\delta$, where $\delta$ represents the decoder failure probability [5].

The probability distribution $\Omega(d)$ of the degrees on the right-hand side is a precious part of the design for LT codes. A number of distributions have been proposed in the literature. Two preliminary distributions were studied in [10] and [11].

### A. Ideal Soliton Degree Distribution

The ideal soliton distribution is given as

$$\rho(d) = \begin{cases} \dfrac{1}{k} & \text{for } d = 1, \\ \dfrac{1}{d(d-1)} & \text{for } d = 2, 3, \ldots, k, \end{cases} \tag{3}$$

which has not been practically adapted in the literature. This distribution may result in some of the data symbols not being covered by the code symbols, and this distribution supplies only a single degree-one code symbol to the ripple size budget in each decoding step, which makes such a distribution unstable and fragile.

### B. Robust Soliton Degree Distribution (RSD)

Improvements in ideal soliton distributions have been made, and a robust soliton distribution (RSD) was introduced. For RSD, two further parameters have been introduced: a constant $c \in (0,1)$, and δ representing the probability of the decoding failure. In addition, the expected number of degree-one symbols is calculated as

$$S = c.\, \ln\left(\frac{k}{\delta}\right)\sqrt{k}. \tag{4}$$

Moreover, with this new parameter, the function $\tau(d)$ is introduced as

$$\tau(d) = \begin{cases} \dfrac{S}{k}\dfrac{1}{d} & \text{for } d = 1, 2, \ldots, \dfrac{k}{S} - 1, \\ \dfrac{S}{k}\ln\left(\dfrac{S}{\delta}\right) & \text{for } d = \dfrac{k}{S}, \\ 0 & \text{for } d > \dfrac{k}{S}. \end{cases} \tag{5}$$

A robust soliton distribution is then formed using (3) and (5) as

$$\Omega(d) = \frac{\rho(d) + \tau(d)}{\beta}, \tag{6}$$

where $\beta = \sum_d \rho(d) + \tau(d)$. This distribution is successful in building a strong infrastructure for a worthy encoding-decoding performance.


ETRI Journal, Volume 38, Number 5, October 2016
http://dx.doi.org/10.4218/etrij.16.0116.0107

### 2. Decoding

The decoder uses the benefits of an encoding operation to extract the information symbols in an iterative manner through the use of message passing between both sides of the Tanner graph, as shown in Fig. 1. The operation of the decoder can be summarized in four steps [10]:

1) Find a coded symbol $C_n$ connected to a single data symbol $u_k$ (The decoder declares a failure if there is no such coded symbol).
2) Decode $u_k$ as $C_n$.
3) Take the XOR of $u_k$ and coded symbols that are connected to $u_k$.
4) Release all $u_k$ connections.
5) Repeat (1)–(4) until all data symbols are recovered, or declare a failure if there is no longer a degree-one coded symbol.

## III. Improvements in LT Code Design

It is apparent from the above decoding steps that the existence of a degree-one coded symbol is vital for the continuation of the decoding operations. If a degree-one coded symbol cannot be found during any iteration, the decoding process halts and the decoder declares a failure. In this paper, we propose a method called LT with pattern recognition, that is, LT-PR, for the decoding of LT codes even in the absence of degree-one coded symbols. The performance improvement of the existing LT-like codes through the use of our suggested approach is also inspected.

### 1. Memory Based RSD (MBRSD)

For MBRSD [10], a classical encoding operation of an LT code is modified, introducing a memory unit into the system to keep track of the data symbols connected to degree-one coded symbols. The aim of MBRSD is to decrease the complexity of the decoding operations. Because a memory unit is introduced into the system while choosing the data symbols, the distribution used while selecting the data symbols is no longer uniform. The encoding operation for MBRSD can be outlined as follows:

**Algorithm 1.** Grouping algorithm

1) Generate a degree $d$ from the right-hand side distribution, similar to RSD.
2) If $d = 1$, choose the data symbol with the highest instantaneous degree without a replacement. If $d \neq 1$, choose $d$ uniformly distributed data symbols.
3) Perform an XOR of the chosen $d$ data symbols to generate and transmit the code symbol $C_n$.
4) Repeat steps 1–3 until an acknowledgement signal is received.

### 2. Proposed Code Design: LT with Pattern Recognition (LT-PR)

As mentioned previously, the absence of a degree-one code symbol at any instant of the iteration results in a decoding failure. In LT encoding, coded symbols are formed by taking the XOR of randomly selected data symbols. When a degree-one symbol is found, the data symbol connected to a degree-one symbol is decoded directly. In addition, the resolved data symbol is added to the other coded symbols that contain the data symbol in their XOR formation. In this way, the degrees of the coded symbols are reduced.

This can be mathematically explained as follows: if

$$c_i = u_l \text{ and } c_m = u_l \oplus u_n \oplus \cdots,$$

then $c_m$ is reduced as

$$c_m = c_m \oplus c_i.$$

This operation can be generalized for coded symbols having degrees of more than one. Let $\boldsymbol{c}_i$ be a coded symbol such that $\deg(\boldsymbol{c}_i) > 1$, where $\deg(\cdot)$ is a degree function whose output is the degree value of coded symbol $\boldsymbol{c}_i$. Let us define the run-set as $\mathrm{rs}(\boldsymbol{c}_i) = \{u_l, u_s, \ldots\}$, which is the set of data symbols used while generating $\boldsymbol{c}_i$. For the coded symbols $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$, if

$$\mathrm{rs}(\boldsymbol{c}_i) \subset \mathrm{rs}(\boldsymbol{c}_j), \tag{7}$$

then $\boldsymbol{c}_j$ can be reduced to

$$\boldsymbol{c}_j = \boldsymbol{c}_j \oplus \boldsymbol{c}_i, \tag{8}$$

after which the degree of $\boldsymbol{c}_j$ is reduced to

$$\deg(\boldsymbol{c}_j) = \deg(\boldsymbol{c}_j) - \deg(\boldsymbol{c}_i). \tag{9}$$

This is the motivation of our approach. In other words, if we cannot find a degree-one code symbol, we can then look for code symbols of degree-two, degree-three, and so on and try to reduce higher-degree symbols using lower-degree symbols in (7) through (9). That is, we look for some code patterns in other code patterns by paying attention to the degrees and run-set of the code symbols.

Pattern searching can be conducted by using the connections between the data and coded symbols of the Tanner graph, or by inspecting the columns of the generator matrix of the LT encoder.

Example:

Assume that the generator matrix of an LT code is as given in (10)

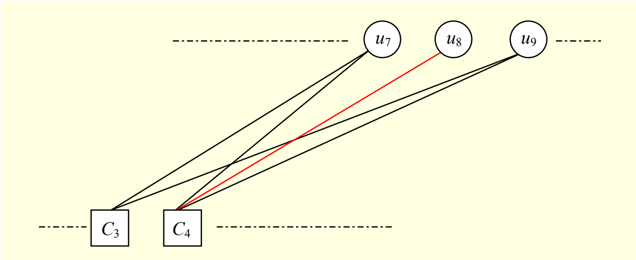


Fig. 2. Tanner graph connections for $C_3$ and $C_4$.

$$G^{\mathrm{T}} = \begin{bmatrix} 0100000001 \\ 1000000100 \\ 0000001010 \\ 0000001110 \\ 0100010000 \\ 0001000001 \\ 1000100000 \\ 1001100000 \\ 0000000000 \\ 0000000101 \end{bmatrix}_{10\times10} . \quad (10)$$

The number of 1s in each row of $G^{\mathrm{T}}$ in (10) shows the degree of the coded symbols. Then, the degrees of the code symbols are found as follows.

$$\deg(C_1)=2 \quad \deg(C_2)=2 \quad \deg(C_3)=2 \quad \deg(C_4)=3$$
$$\deg(C_5)=2 \quad \deg(C_6)=2 \quad \deg(C_7)=2 \quad \deg(C_8)=3$$
$$\deg(C_9)=2 \quad \deg(C_{10})=2$$

Here, it can be seen that there is no degree-one code symbol. A conventional LT decoder using belief propagation declares a failure at the beginning of the decoding procedure. However, when the run-set of the code symbols is inspected we can see that

$$\begin{aligned} &\mathrm{rs}(C_3)=\{u_7,u_9\}, \\ &\mathrm{rs}(C_4)=\{u_7,u_8,u_9\}, \\ &\mathrm{rs}(C_3)\subset \mathrm{rs}(C_4). \end{aligned} \quad (11)$$

The connections for code symbols $C_3$ and $C_4$ in the Tanner graph are shown in Fig. 2.

Using (11), $C_4$ can be simplified using

$$C_4 = C_4 \oplus C_3,$$

reducing the degree of $C_4$ to

$$\deg(C_4)=\deg(C_4)-\deg(C_3)\rightarrow \deg(C_4)=3-2\rightarrow 1,$$

which is only of degree one. Because a degree-one coded symbol is again available, we can continue decoding using a conventional approach.

In addition, a similar relation is also available for the code symbols $C_7$ and $C_8$, that is, $\mathrm{rs}(C_7) \subset \mathrm{rs}(C_8)$ and $C_8 = C_8 \oplus C_7$. Thus, it can be concluded that even in the absence of degree-one code symbols, it may be possible to continue with the decoding by searching for some coded patterns in other patterns. The simplest approach may be searching for degree-two coded patterns inside degree-three coded patterns.

A Tanner graph gives us an image of these patterns, which can help us continue on with the decoding process. Let us define some of the terms used in a Tanner graph. Code and data nodes are the points in a Tanner graph where the edges are connected to each other. A cycle is a set of nodes and edges such that a node can be reached from any of the other nodes by tracing the connecting edges. For instance, in Fig. 2, the nodes labelled by $C_3$, $C_4$, $u_7$, and $u_9$, and the black edges connecting all of these nodes, form a cycle. If two code nodes are in the same cycle, then a coded symbol with a smaller degree representing one of these nodes can be used to reduce the degree of the coded symbol representing another node.

The decoding approach using Tanner graphs can also be interpreted using the generator matrix of an LT code. Let $\mathrm{r}_i$ and $\mathrm{r}_j$ be two rows of $G^{\mathrm{T}}$, and $d(\mathrm{r}_i)$ and $d(\mathrm{r}_j)$ be the number of 1s in rows $\mathrm{r}_i$, and $\mathrm{r}_j$, respectively. Here, $\deg(C_i) = d(\mathrm{r}_i)$ and $\deg(C_j) = d(r_j)$. If $d(\mathrm{r}_i \oplus \mathrm{r}_j) < d(\mathrm{r}_i)$ or $d(\mathrm{r}_i \oplus \mathrm{r}_j) < d(\mathrm{r}_j)$, then $C_i$ or $C_j$ can be simplified using either $C_i = C_i \oplus C_j$ or $C_j = C_j \oplus C_j$. While choosing $C_i$ or $C_j$ for simplification, we can use the following criterion:

$$C_k = \underbrace{\mathrm{argmin}}_{k}\left| d(\mathrm{r}_i \oplus \mathrm{r}_j) - d(\mathrm{r}_k)\right|,\ k \in \{i, j\}\ .$$

If $d(\mathrm{r}_i \oplus \mathrm{r}_j) > d(\mathrm{r}_i)$ and $d(\mathrm{r}_i \oplus \mathrm{r}_j) > d(\mathrm{r}_j)$, then $C_i$ or $C_j$ cannot be used to simplify each other. For a simple illustration, we can consider algorithm 2. With this algorithm, we search for only the coded symbols that differ by a single packet in their run sets.

**Algorithm 2.**

1) Assume that $G$ is an $N \times N$ binary generator matrix, and $N$ coded symbols are received.
2) Check the existence of a degree-one coded symbol. If degree-one coded symbol $C_i$ exists, which also means that a row containing a single 1 in $G^{\mathrm{T}}$ exists, that is, $G_{ij} = 1$, resolve the corresponding data symbol. Set $G_{ij} = 0$, $i = 1, \ldots, N$, that is, set the corresponding column elements to 0, and remove the row containing a single 1 from $G^{\mathrm{T}}$. This decoding approach using a generator matrix is equivalent to the Tanner graph decoding method.
3) If there is no degree-one code symbol, then calculate $d(\mathrm{r}_i \oplus \mathrm{r}_j)$, $i = 1, j = 1, 2, \ldots, N$.
4) If $d(\mathrm{r}_i \oplus \mathrm{r}_j) = 1$ is found in step 3, proceed as in step 2. Otherwise, increment the value of $i$, that is, $i = i + 1$, and repeat step 3.

Without checking for more complex patterns, the easy


http://dx.doi.org/10.4218/etrij.16.0116.0107

method provided in algorithm 2 improves the performance of LT decoders significantly. We provide our experimental results in the next section.

## IV. Simulation Results

We checked the improvement of the state-of-the-art MBRSD when employing the proposed approach. We used data lengths of $k = 32$ and $k = 256$ for our simulations. For the right-hand side degree distributions, RSD with parameters $c = 0.02$ and $\delta = 0.1$ were used. The simulations were run until 100 erroneous frames were received. This means that the number of transmitted packets changed for every rate. For our proposed approach, we tried the technique described in algorithm 2. In Fig. 3, the bit-error-rate performances of the LT code using deferent decoding approaches assisted by belief propagation (BP) are presented. Comparisons were conducted among our proposed pattern recognition assisted BP (PR-BP); a regular BP, that is, MRSD assisted BP [10]; and the well-known Gaussian elimination assisted BP [12]. It is clear from Fig. 3 that an LT code with RSD employing the proposed method outperforms the conventional BP-RSD and BP-MBRSD methods for all rates applied. For the BER performance, our proposed method achieved a similar score to that of the BP-GE-RSD [13]. For regular BP, to recover the $k$ source symbols from any $N$ encoding symbols with a probability of $1 - \delta$, an average of $O\left(k \ln(k/\delta)\right)$ symbol operations were required [1]. On the other hand, the required number of additive operations for BP-GE is on the order of $O(k^2)$ [14]. For our proposed method, the additive complexity is on the order of $O(1*m)$, where $1 \le l \le n$ and $1 \le m \le n$. When $n = k + \epsilon$ and $l = m = n$, the worst additive complexity of our proposed method is similar to that of the BP-GE approach. However, it is obvious that the probabilistic average complexity of our proposed approach is less than that of the BP-GE method. From this point of view, it is clear that the proposed method is more efficient in terms of the computational complexity.

For a larger data length of $k = 256$, the BER performances of an LT code using BP-RSD, BP-MBRSD, BP-PR-RSD, and BP-PR- MBRSD employing algorithm 2 are shown in Fig. 4. It is clear from Fig. 4 that the proposed approach enhances the performances of LT-like codes at all rates. This improvement is due to the removal of a decoding block owing to the absence of degree-one coded symbols.

Another criterion for the performance of rateless codes is the decoding success or failure rate. The decoding success rate is a measure of the decoder performance and is defined as the ratio of total number of correctly decoded packets to the total number of transmitted packets. The simulation was performed

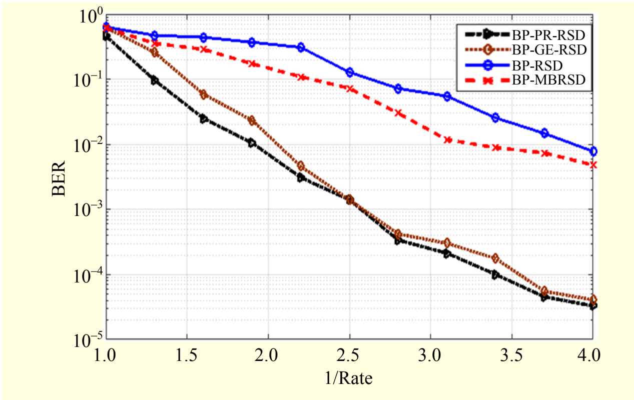


Fig. 3. BER performances for an LT code using BP-RSD, BP-MBRSD, BP-GE-RSD, and BP-PR-RSD for $k = 32$ using RSD with parameters $c = 0.02$ and $\delta = 0.1$ with an erasure probability of $\alpha = 0.02$.

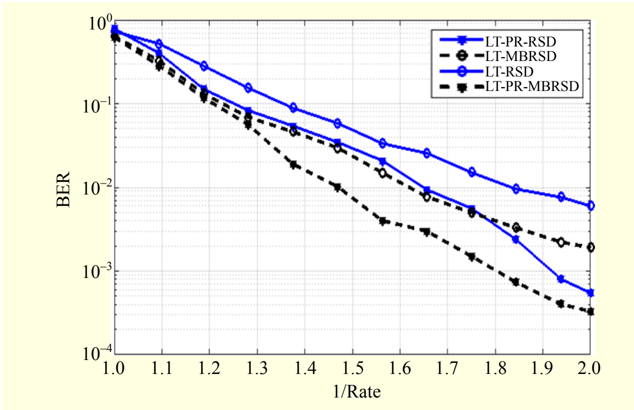


Fig. 4. BER performances for an LT code using BP-RSD, BP-MBRSD, BP-PR-RSD, and BP-PR-MBRSD for $k = 256$ using RSD with parameters $c = 0.02$ and $\delta = 0.1$ with an erasure probability of $\alpha = 0.02$.

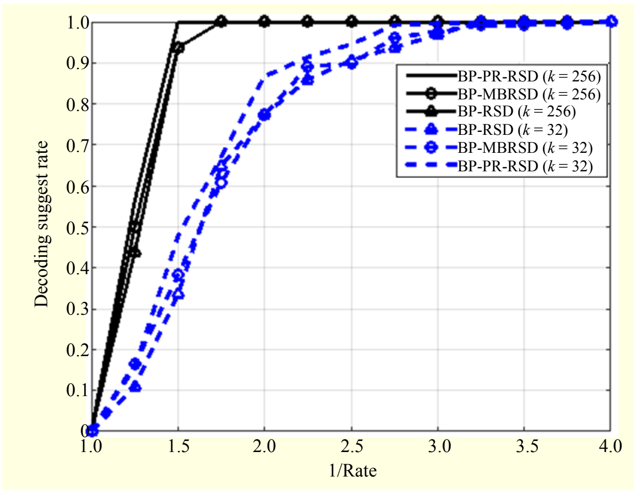


Fig. 5. Decoding probability performance curves for an LT code using BP-RSD, BP-MBRSD, and BP-PR-RSD for $k = 32$ and 256 using RSD with parameters $c = 0.02$ and $\delta = 0.1$ with an erasure probability of $\alpha = 0.02$.

for two data lengths of $k$ = 32 and $k$ = 256. The performance graph is shown in Fig. 5. For data length $k$ = 256, the BP-PR-RSD achieves 100% decoding success at a rate of 1.5, whereas the BP-MBRSD needs an additional rate of 0.25 to achieve the same performance level. For data length $k$ = 32, LT-PR-RSD achieves 100% decoding success at a rate of 2.75, whereas MBRSD achieves the same performance level at a rate of 3.25.

## V. Conclusion

Absence of degree-one code symbols during the decoding of LT codes results in decoding failure. The decoding operation can even halt at the beginning of the iteration. To alleviate this problem, we proposed an approach for the decoding of LT-like codes such that, even in the absence of degree-one code symbols, the decoding operation can be continued and a stuck decoding operation can be overcome, resulting in a better level of performance. In addition, the complexity of the proposed approach is negligible, and the method can even be used for real-time applications. The simulation results show that, with the proposed approach, better performance levels for LT and LT-MBRSD codes are obtained when compared to their performances without the proposed method.

## References


[1] J.W. Byers et al., "A Digital Fountain Approach to Reliable Distribution of Bulk Data," *Proc. ACM Conf. Appl., Technol., Archiectures Protocls comput. Commun.*, Vancouver, Canada, Aug. 1998, pp. 56–67.

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

[3] J.H. Sorensen, P. Popovski, and J. Ostergaard, "Design and Analysis of LT Codes with Decreasing Ripple Size," *IEEE Trans. Commun.*, vol. 60, no. 11, June 2012, pp. 3191–3197.

[4] D.J.C. MacKay, "Information Theory, Inference, and Learning Algorithm," in *Computer Modern*, 3rd ed., Cambridge, UK: Cambridge University Press, 2004, pp. 591–596.

[5] M. Luby, "LT Codes," *Proc. Annu. IEEE Symp. Found. Comput. Sci.*, Vancouver, Canada, Nov. 16–19, 2002, pp. 271–280.

[6] Z. Zhiliang et al., "Performance Analysis of LT Codes with Different Degree Distribution," *Int. Workshop Chaos-fractals Theories Appl.*, Liaoning, China, Oct. 18–21, 2012, pp. 142–146.

[7] G. Chunmei and B. Xueyao, "Performance Analysis and Parameter Optimizing Rules of LT Codes," *J. China Commun.*, vol. 7, no. 4, 2010, pp. 103–107.

[8] K. Yen et al., "Modified Robust Soliton Distribution (MRSD) with Improved Ripple Size for LT Codes," *IEEE Commun. Lett.*, vol. 17, no. 5, May 2013, pp. 976–979.

[9] L. Haifeng et al., "LT-W: Improving LT Decoding with Wiedemann Solver," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, Dec. 2013, pp. 7887–7897.

[10] K.F. Hayajneh, S. Yousefi, and M. Valipour, "Improved Finite-Length Luby-Transform Codes in the Binary Erasure Channel," *J. IET Commun.*, vol. 9, no. 8, 2015, pp. 1122–1130.

[11] D.J.C. MacKay, "Fountain Codes," *IEE Proc. Commun.*, vol. 152, no. 6, Dec. 2005, pp. 1062–1068.

[12] V. Bioglio et al., "On the Fly Gaussian Elimination for LT Codes," *IEEE Commun. Lett.*, vol. 13, no. 12, Dec. 2009, pp. 953–955.

[13] H.Y. Cheong et al., "Belief Propagation Decoding Assisted on-the-Fly Gaussian Elimination for Short LT Codes," *Cluster Comput.*, vol. 19, no. 1, Mar. 2016, pp. 309–314.

[14] S.J. Kim, K.R. Ko, and S.Y. Chung, "Incremental Gaussian Elimination Decoding of Raptor Codes over BEC," *IEEE Commun. Lett.*, vol. 12, no. 4, Apr. 2008, pp. 307–309.


**Nadhir I. Abdulkhaleq** received his BS degree in electrical engineering-aircraft board equipment from Military Technical College, Cairo, Egypt in 1990, and his MS degree in electrical engineering-communication and radar from Military College of Engineering, Baghdad, Iraq in 1995. From 1995 to 2004, he worked as a lecturer in military institutes. Since 2004, he has been lecturer at the Technology Institute-Middle Technical University, Baghdad, Iraq. He earned a scholarship from the Ministry of Higher Education and Scientific Research in Iraq to obtain his PhD in electronic and communication engineering from Cankaya University, Ankara, Turkey, which he started in 2013.

**Orhan Gazi** obtained his BS, MS and PhD degreed in electrical and electronics engineering from the Middle East Technical University, Ankara, Turkey in 1996, 2001, and 2007, respectively. His research interests include channel coding, digital communication, and information theory. He is currently working in the Electronics and Communication Engineering Department of Cankaya University as an associate professor.



http://dx.doi.org/10.4218/etrij.16.0116.0107