Optimization of Degree Distribution for Layer-aligned Multipriority Rateless Codes based on Safety Criteria of Ripple

Lien-En Hung, Chun-Kuan Lee, Hsu-Feng Hsiao
Department of Computer Science
National Yang Ming Chiao Tung University
Hsinchu, Taiwan

***Abstract*—Data loss or errors happen frequently on real networks. A common way of addressing this issue is packet retransmission, at the cost of potentially excessive transmission delay. For applications such as multimedia streaming, using channel coding to combat packet loss or errors is a more attractive option. Rateless codes have been an interesting approach due to their lower coding complexity and versatility. Layer-aligned multipriority rateless codes were originally designed for streaming with the capability of unequal error protection. In this paper, we are interested in finding a better degree distribution for such codes. We firstly derive the safety criteria of ripple sizes using a proposed leaping random walk model. In addition, we design an estimate function to predict the ripple size variation for the layer-aligned multipriority rateless codes. To achieve a better degree distribution, we use a genetic algorithm to optimize a multi-objective problem that we have formulated. This enabled us to improve the overall performance of the coding methods. Our simulation results demonstrate that the optimized degree distribution lead to significant improvements in error correction and data recovery rates.**

I. Introduction

Reed-Solomon codes (RS codes) [1] are widely recognized as effective channel coding methods. However, one drawback of such codes is their high complexity , which stems from their calculations in extended Galois fields. Additionally, this complexity limits the number of message symbols that can be processed efficiently.

To address this challenge, Luby Transform Codes (LT codes) [4] were introduced as the first fountain codes [2], also known as rateless codes [3]. The encoding and decoding process of LT codes uses only exclusive-or (XOR) bitwise operations to generate coded symbols that are independent of each other, allowing them to be produced on the fly.

The ripple, as defined in [4], is the set of coded symbols with degree one at each decoding iteration. If the size of the ripple decreases and no degree one coded symbols can be found during the decoding process, the belief-propagation algorithm will terminate. However, this does not mean that the ripple size needs to be very large, as this increases the probability of redundant coded symbols. Therefore, determining a suitable ripple size during the decoding process is an important issue that needs to be considered when using belief-propagation algorithms for LT codes or similar coding structures. The degree distribution $\Omega(x)$, which determines the probability of a coded symbol having degree x, is also a critical factor that influences the recovery performance of LT codes. For each degree x, x message symbols are randomly selected to produce a coded symbol. A degree distribution $\Omega(x)$ that has been used by many rateless code-based methods, including [6], [7], [8], [9], [10], and [25], is provided in [5].

Layer-aligned multipriority rateless codes (LMRC) [9] are specifically designed with an N-cycle layer-aligned overlapping structure for unequal error protection. The decoding of rateless codes is based on the collection of coded symbols with degree one, known as the "ripple." During each iteration, a coded symbol in the ripple is removed, and a message symbol can be decoded. The message symbol is then used to remove the degree from all remaining coded symbols, potentially increasing the size of the ripple. The continuation of the decoding process is heavily influenced by the size of the ripple. If the ripple is empty, the belief-propagation algorithm can be terminated prematurely. However, a large ripple size implies high redundancy in the coded symbols, which ultimately reduces the coding efficiency.

Our study aims to optimize the degree distribution of a branch of rateless codes, namely layer-aligned multipriority rateless codes, based on ripple sizes. First, we model the coding procedure as a specific random process and derive criteria for optimizing ripple sizes. Second, we propose an estimation function for ripple sizes. Third, we formulate multi-objective functions for degree distribution based on the criteria and estimated ripple sizes. To solve the optimization problem, we use a genetic algorithm and develop a principle for selecting a better degree distribution.

In Section II, we describe the proposed method, and in Section III, we present the experimental results. Section IV summarizes the conclusions of our work.

II. Proposed Method

Finding the appropriate ripple size in the decoding process is crucial for determining the degree distribution. To this end, we first discuss the safety criteria for ripple size variation in different decoding phases. Once the safety criteria are established, we design a method to achieve the desired ripple size. We propose an estimation method for the ripple size during the decoding process using LMRC. In addition to the ripple size, we aim to minimize the decoding failure rate at the end of the process.

Figure 1 illustrates the proposed approach for identifying a better degree distribution for LMRC. The parameter δ represents the desired probability of decoding termination during the decoding process, while $\hat{d}$ is the degree distribution we define as $(d_1, d_2, \ldots, d_{max})$, and d_i corresponds to the probability of a coded symbol having a degree of i. R(x) is an estimation of the ripple size in LMRC for a given degree

979-8-3503-3538-5/23/$31.00 ©2023 IEEE

distribution $\hat{d}$ and decoding success rate x. We expect that the ripple size R(x) of $\hat{d}$in LMRC can achieve the safety criteria. We then formulate an objective optimization problem based on the above results and apply the NSGA-II algorithm [11] to obtain the set of degree distributions that come closest to achieving our objectives, as shown in Fig. 2.

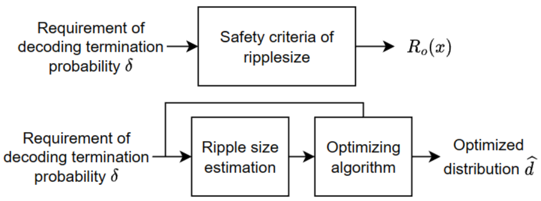

Fig. 1. The concept diagram of the proposed method.

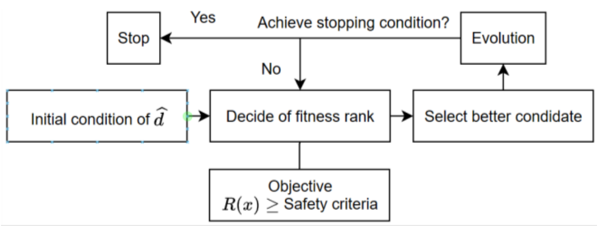

Fig. 2. The process diagram for finding optimizing degree distribution $\hat{d}$.

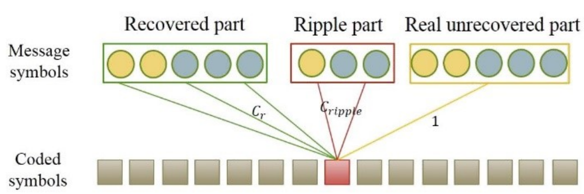

Fig. 3. Example of a coded symbol with degree 6.

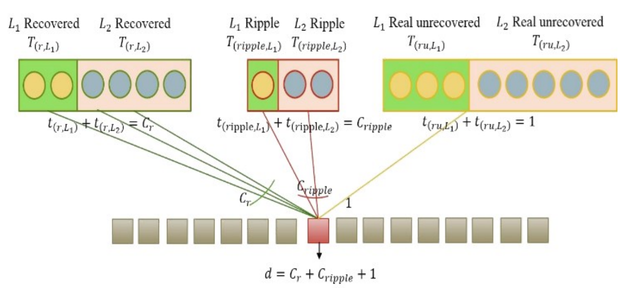

Fig. 4. Example of variables described in 2-layers LMRC.

A. Leaping random walk model and safety criteria for ripple size in LMRC

A good degree distribution has an important property in terms of the expected ripple size, as mentioned in [4]. However, a serious variation in ripple size during the decoding process is not desirable as it increases the likelihood of termination when the ripple size is relatively small [13]. Conversely, the ripple size needs to remain large enough to prevent premature termination of the decoding process. The ideal soliton degree distribution is a counterexample because it is designed with an expected ripple size of one. While the robust soliton degree distribution can increase the ripple size during decoding, its performance is not optimal for lower coding overhead and smaller window sizes. In the following section, we will derive safety criteria for the ripple size, which can be considered as the ideal magnitude during the decoding process.

Shokrollahi and Luby presented a random walk model for the decoding of LT codes in their work [5]. According to their model, the ripple size of the LT code increases or decreases with equal probabilities. To represent this, we introduce the symbol R_x, which denotes the ripple size when the number of recovered message symbols is x. The random walk model can be expressed as follows:

$$R_{x+1} = \begin{cases} R_x + 1 & with\ probability\ \ \dfrac{1}{2} \\ R_x - 1 & with\ probability\ \ \dfrac{1}{2} \end{cases} \quad (1)$$

This model is also known as the symmetric one-dimensional random walk model, and its theory is well established and has been introduced and used in numerous academic literature, such as [21], [22], and [23].

Although this condition is only a heuristic for designing the ripple size, it has been proven to be useful for degree distribution design [5] and has been applied in [13] and [24]. However, the probability of ripple size increase or decrease depends on the amount of unprocessed message symbols, the number of message symbols already recovered, and the ripple size at that time. While the variation in ripple size can be modeled as a random walk, the probabilities involved cannot always be considered constant during the decoding process. By correctly deriving these probabilities, we can determine the probability of ripple decrease and obtain the safety criteria for the ripple size to avoid its decrease to zero based on the number of coded symbols and the user-defined decoding termination probability.

To derive the probability of ripple decrease and the safety criteria for the ripple size, it is necessary to understand the decoding process of LMRC. Initially, all message symbols are divided into three parts: Recovered, Ripple, and Real Unrecovered. Since we define ripple as degree 1 symbols released at each decoding iteration, the symbols in the ripple do not belong to the recovered part at that time. Similarly, the symbols in the ripple do not belong to the real unrecovered part either because their degree is 1. Next, each part is segmented into different slices based on the number of layers in LMRC.

During each decoding iteration, all degree 1 symbols are checked, except for the ones in the recovered part, and added to the ripple. Subsequently, the connection between the symbols in the ripple and their neighboring coded symbols is removed, and all symbols in the ripple are moved to the recovered part. New degree 1 symbols are then checked and added to the ripple, and the decoding process continues until no degree 1 coded symbols remain.

Before we can derive the probability that a coded symbol with degree d can increase the ripple size under different decoding conditions, we need to define some variables. To add an unrecovered symbol to the ripple, one edge of a coding symbol must connect to the real unrecovered part, while the other edges must connect to the ripple and recovered parts. Notably, there must be at least one edge that connects to the ripple part, as we rely on the symbols in the ripple to help release new degree 1 symbols. Therefore, we define two variables C_r and C_{ripple}, which represent the numbers of edges connecting to the recovered part and ripple part, respectively. The example for $C_r = 3, C_{ripple} = 2$ is shown in Fig. 3. And, the relationship between a coded symbol with degree d, C_r and C_{ripple} can be described as

$$d = C_r + C_{ripple} + 1, C_{ripple} \geq 1 \quad (2)$$

We can calculate the probability of a degree d coded symbol that can help add a new unrecovered symbol to the ripple by considering the connection condition (C_r, C_{ripple},1) and the sum of edges, which must be equal to d. We can then derive the probability using the following equation:

$$\sum_{\substack{C_r + C_{ripple} = d-1 \\ C_{ripple} \geq 1}} Probability(C_r, C_{ripple}, 1) \quad (3)$$

The summation calculates the different combinations of edges that connect to the three parts: the recovered part, the ripple part, and the real unrecovered part. The probability function determines the likelihood that the connection condition to the three parts is $(C_r, C_{ripple}, 1)$. However, in LMRC, the different parts are further partitioned into different layers, taking into account the number and weight of message symbols in each layer, as it will affect the probability of a coded symbol choosing a specific number from a specific part and layer. For this purpose, we defined the set

$$\begin{gathered} \hat{T}_r = \left(T_{(r,L_1)}, T_{(r,L_2)}, \dots, T_{(r,L_N)}\right) \\ \hat{T}_{ripple} = \left(T_{(ripple,L_1)}, T_{(ripple,L_2)}, \dots, T_{(ripple,L_N)}\right) \\ \hat{T}_{ru} = \left(T_{(ru,L_1)}, T_{(ru,L_2)}, \dots, T_{(ru,L_N)}\right) \end{gathered} \quad (4)$$

The set $\hat{T}_r$ consists of the number of each layer's message symbols in recovered part. Similarly, $\hat{T}_{ripple}$ and $\hat{T}_{ru}$ consists of the number of different layer's message symbols in ripple part and real unrecovered part, respectively. Note that $T_{(ru,L_i)} = T_{(r,L_i)} - T_{(ripple,L_i)}$. By using the same procedure, we can define the connection condition when the edges connect to different parts and layers as follows

$$\begin{gathered} \hat{t}_r = \left(t_{(r,L_1)}, t_{(r,L_2)}, \dots, t_{(r,L_N)}\right) \\ \hat{t}_{ripple} = \left(t_{(ripple,L_1)}, t_{(ripple,L_2)}, \dots, t_{(ripple,L_N)}\right) \\ \hat{t}_{ru} = \left(t_{(ru,L_1)}, t_{(ru,L_2)}, \dots, t_{(ru,L_N)}\right) \end{gathered} \quad (5)$$

and the limit conditions:

$$\sum_{i=1}^{N} t_{(r,L_i)} = C_r, \quad \sum_{i=1}^{N} t_{(ripple,L_i)} = C_{ripple}, \quad \sum_{i=1}^{N} t_{(ru,L_i)} = 1 \quad (6)$$

Then, we use Fig. 4 to illustrate the relationship between different variables in the 2-layers LMRC. As a result, we can calculate the probability function as follows:

$$probability(C_r, C_{ripple}, 1) = \sum_{C_r} \sum_{C_{ripple}} \sum_{1} WNHD(d, \hat{T}, \hat{t}, \hat{w}) \quad (7)$$

where

$$\begin{gathered} \hat{T} = \{\hat{T}_r, \hat{T}_{ripple}, \hat{T}_{ru}\} \\ \hat{t} = \{\hat{t}_r, \hat{t}_{ripple}, \hat{t}_{ru}\} \\ \hat{w} = \{(w_1, \dots, w_N), (w_1, \dots, w_N), (w_1, \dots, w_N)\} \end{gathered}$$

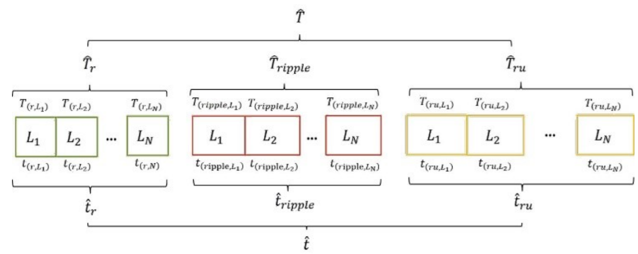

Fig. 5. Relationship between the variables.

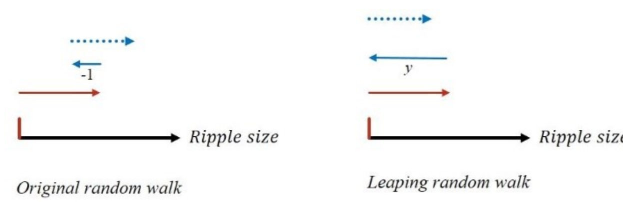

Fig. 6. The diagram to show the difference between normal random walk and leaping random walk

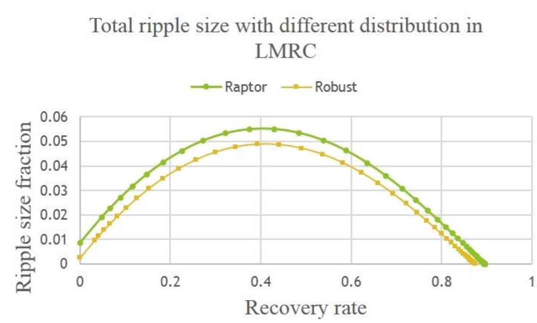

Fig. 7. The variation of total ripple size fraction during decoding process with different distributions.

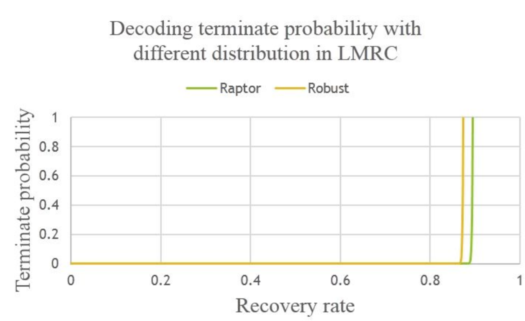

Fig. 8. The terminate probability $F_{[x,x+y]}$ using different distributions.

The first summation in equation (7) takes into account the different connection conditions in the recovered part, while the second summation considers the different connection conditions in the ripple part. Similarly, the third summation considers the connection conditions in the real unrecovered part. To calculate the occurrence probability of a specific connection condition, the WNHD function in (7), we can use the Multivariate Wallenius' Noncentral Hypergeometric Distribution function, provided that we know the number and weight of each part in LMRC. $\hat{T}$ is the set consisted of the number of message symbols for different parts and layers, and $\hat{t}$ is the set, which contains the number of edges that connect to different parts and layers. $\widehat{w}$ is the set which consists of different parts and layers' weight, shown in Fig. 4. In Fig. 5, we illustrate the relationship between the variables in different parts and layers in LMRC.

Finally, the probability $p(d, \hat{T}, \hat{t}, \widehat{w})$ that a coded symbol with degree d, which can help a new symbol added into the ripple in LMRC with parameter $\hat{T}$, $\hat{t}$ and $\widehat{w}$, can be calculated as:

$$p(d, \hat{T}, \hat{t}, \widehat{w}) = \sum_{\substack{C_r + C_{ripple} = d-1 \\ C_{ripple} \geq 1}} Probability(C_r, C_{ripple}, 1) \tag{8}$$

The probability $p(\hat{T}, \hat{t}, \widehat{w})$ that a coded symbol can help add a real unrecovered symbol into the ripple can be derived as:

$$p(\hat{T}, \hat{t}, \widehat{w}) = \sum_{d=1}^{d_{max}} \Omega_d \times p(d, \hat{T}, \hat{t}, \widehat{w}). \tag{9}$$

To model a leaping random walk, we depart from the traditional random walk model. In the leaping random walk, we calculate the probability of a ripple's increase by considering all symbols in the ripple to help a new symbol add into the ripple at once, which is like leaping a distance larger than one. The proposed leaping random walk model is as follows:

$$R_{x+y} = \begin{cases} > 0 & with\ probability \quad p(\hat{T}, \hat{t}, \widehat{w}) \\ Otherwise & with\ probability \quad 1 - p(\hat{T}, \hat{t}, \widehat{w}) \end{cases} \tag{10}$$

where x is the number of total recovered message symbols, and y is the ripple size when the number of recovered message symbols is x. And these two can be calculated as

$$x = \sum_{i=1}^{N} T_{(r, L_i)}, \qquad y = \sum_{i=1}^{N} T_{(ripple, L_i)} \tag{11}$$

Figure 6 shows the difference between normal random walk and leaping random walk. After deriving the leaping random walk model, we need to construct safety criteria for ripple size to ensure decoding can continue at different recovery rates. We can do this by deriving the probability that the ripple size decreases to zero, which can also be regarded as the decoding termination probability. This situation occurs if all coded symbols are unable to help any new unrecovered symbols add to the ripple. Following this rule, when x represents the number of recovered message symbols with ripple size y and the number of coded symbols is N, the probability $F_{[x,x+y]}$ that decoding process terminates during the recovered region x to $x+y$ can be calculated as $F_{[x,x+y]} = \left(1 - p(\hat{T}, \hat{t}, \widehat{w})\right)^N$. We can calculate the decoding termination probability during the decoding process with different degree distributions.

In Fig. 7, we show the ripple size comparison in LMRC with overhead 0.06 and w_1 0.7 while using Raptor code distribution and robust soliton distribution, respectively, with 4000 message symbols. We observe that raptor code distribution shows a larger ripple size during the decoding process. The probability $F_{[x,x+y]}$ was calculated for these two degree distributions, and the result is shown in Fig. 8. We discovered that with larger ripple size in the decoding process, the decoding termination probability $F_{[x,x+y]}$ is smaller. This is observed during the interval [0.8, 1] of the recovery rate.

To obtain the safety criteria for the ripple size in different phases of decoding, we can use the decoding terminate probability formulation if we are given an allowable terminate probability function $\delta(L_j, x)$, defined by the user. The safety criteria for each priority layer is $F_{[x,x+y]} \leq \delta(L_j, x), \hat{T} = \{\hat{T}_r, \hat{R}_{L_j,x}, \hat{T}_{ru}\}$.

Now, we will introduce the estimation of ripple size. Let $s_{i,j}$ be the expected fraction of layer j to which message symbols belong and are recovered when reaching tree level i in AND-OR tree analysis. We can determine that a message symbol has been successfully recovered only when it is connected to an edge carrying the value 1 from a coded symbol. Therefore, we can write the probability as $s_{i,j}$, and we know the probability $p_{i,j}$ [9] that a message symbol belongs to layer j whose value is zero and can be calculated iteratively according to q_i. Also, the probability $p_{i-1,j}$ [9] that an OR-node belongs to layer j at level i-1 is zero. And the recursive formula can be written as:

$$s_{i,j} = 1 - p_{i,j} \approx 1 - e^{N\mu\gamma_w k p_j(-1+q_i)} \tag{12}$$

$$q_i = \sum_{d=0}^{k-1} \left(A_{d+1} \left(\sum_{\forall d: d_1 + \cdots + d_N = d} wdp(d) \cdot \left(1 - \prod_{j=1}^{N} (s_{i-1,j})^{d_j} \right) \right) \right) \tag{13}$$

This recursion formula shows that if the expected fraction $s_{i-1,j}$ of message symbols has already been recovered at tree level i-1, then after belief-propagation decoding, the fraction will increase to $s_{i,j}$. For more details, please refer to [9]. Therefore, the expected fraction $R_{i,j}$ that message symbols from layer j will add to the ripple when the recovery rate of layer j is $s_{i,j}$ can be represented as $R_{i,j} = s_{i+1,j} - s_{i,j}$. The expected total fraction R_i that message symbols add to the ripple after level i can be obtained by adding all layer fractions $R_i = \sum_{j=1}^{N} R_{i,j}$.

B. *Multi-objective problem formulation*

Assuming that the polynomial functions for the estimated and optimized ripple size are $\hat{R}(x)$ and $R_o(x)$, respectively, we can formulate an objective function that evaluates the expected ripple size in a way that satisfies safety criteria as much as possible when the recovery rate is x. This objective function can be expressed as $\hat{R}(x) \geq R_o(x)$.

Our goal is to ensure that the magnitude of the ripple size meets the safety criteria, and minimize the probability that the ripple size will decrease to zero during the decoding process. We need to consider the balance between the part greater than the optimized ripple size and the part smaller. We define the difference function as *difference*(x)=$|\hat{R}(x)-R_o(x)|$, which represents the discrepancy between the estimated and optimized ripple sizes.

Next, we aim to find the degree distribution $\hat{d}$, which is defined as $\hat{d}\equiv(d_1,d_2,\dots,d_{max})$ at different recovery rates x=[x_0,x_1,…, x_{i_f}], where i_f is the final level at which the failure rate of message symbols is the same as the previous level. However, having two objectives that are very similar to each other in terms of multi-objective optimization problems can be counterproductive to the original intention of such problems' design. Therefore, we convert these two objectives into a single objective function to solve this problem.

$$\begin{gathered}\hat{d}=\arg\min_{d} Obj\\ Obj=c_1\times Obj_1-c_2\times Obj_2\end{gathered}\qquad(14)$$

c_1 and c_2 are user defined parameters which can be used to control the importance level for different objective function. Obj_1 and Obj_2 are the summation of *difference*(x) when $\hat{R}(x)<R_o(x)$ and $\hat{R}(x)\geq R_o(x)$.

We use NSGA-II [11] to solve the optimization problem. Since the results will have more than one candidates in the final population, we proposed a simple principle to choose the degree distribution from the final set with the following steps:

1. Sort all candidates based on the total decoding failure rate calculated by the prediction model.
2. Choose the top m% candidates.
3. Select the candidate with minimum value of Obj.

The entire process of finding optimized degree distribution population using NSGA-II is shown in Fig. 9.

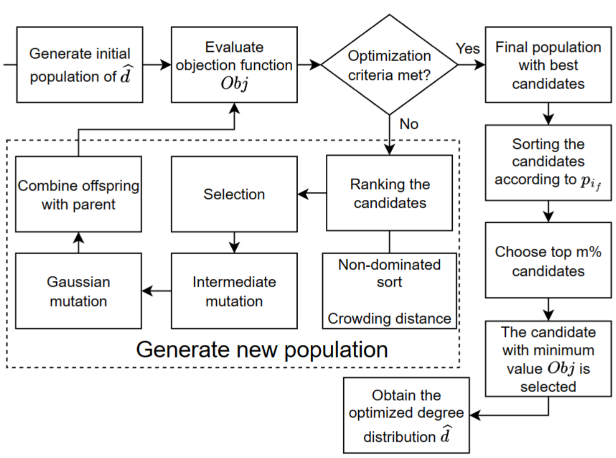

Fig. 9. The entire process of finding optimized degree distribution population.

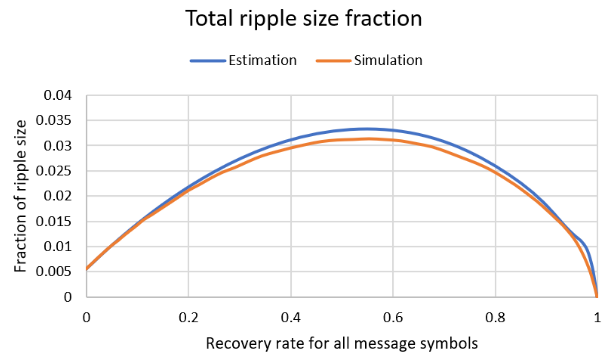

Fig. 10. Compare of Ripple size variation with estimation and simulation.

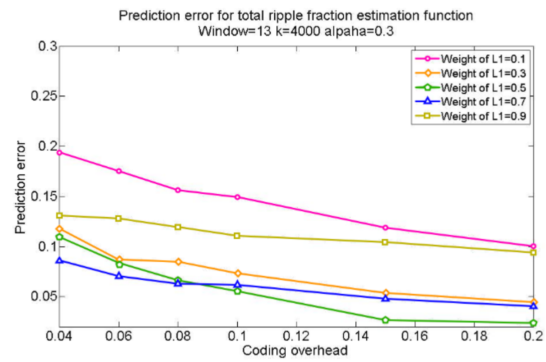

Fig. 11. The estimation error in LMRC.

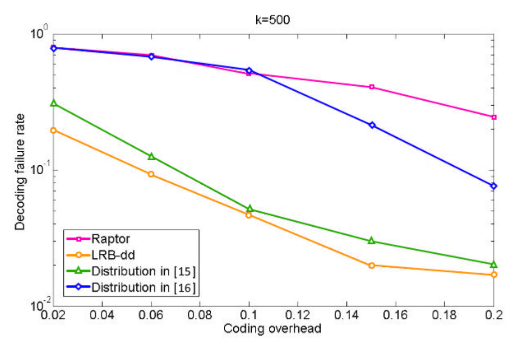

Fig. 12. The average decoding failure rate for different distributions with k=500.

III. THE EXPERIMENT RESULT

In this section, we compare the decoding terminate probability $F_{[x,x+y]}$ calculated by the leaping random walk model and real simulation results between the optimized degree distributions using our proposed method (Leaping random walk based degree distribution, LRB) and other common distributions for LMRC.

We use 2 layers of sliding window, where the window size is 4000 with 13 sliding windows. The proportion of layer 1 and layer 2 $(\alpha_1,\alpha_2)=(0.3,0.7)$; the weight of layer 1 and layer 2 $(w_1,w_2)=(0.7,0.3)$, and the coding overhead $\varepsilon=0.02,0.06,0.1$. c=0.01 and δ=0.05 are used for robust soliton distribution (RSD). Each scenario was simulated 10000 times, and the priority indicator m is 10%. In addition, $\delta(L_j,x)=0.00001$, $c_1=5$, $c_2=1$. The parameters for NSGA-II are as follows. The population size of each generation is 50; the maximum generations is 250; intermediate crossover with crossover probability 0.9 is used, and the mutation method is Gaussian mutation [14] with mutation probability 0.1.

TABLE I. THE AVERAGE RECOVERY RATE WITH DIFFERENT DEGREE DISTRIBUTIONS

	Raptor distribution			**RSD (c=0.01, δ=0.05)**			**Proposed LRB**		
ε \ *Layer*	***both***	***1***	***2***	***both***	***1***	***2***	***both***	***1***	***2***
0.02	0.840	0.971	0.783	0.787	0.947	0.718	0.935	0.994	0.910
0.06	0.905	0.990	0.869	0.875	0.983	0.828	0.961	0.997	0.945
0.10	0.949	0.997	0.929	0.939	0.996	0.914	0.979	0.999	0.971

As seen in Table 1, the average recovery rate of the LRB degree distribution is superior to other distributions in LMRC.

We examined whether the simulation results of the ripple size variation during the decoding process matched the estimation results. Accuracy is an important indicator that determines whether the function can be used to estimate the variation of ripple sizes. For example, in Fig. 10, we show the ripple size variation result with overhead 0.10, $(w1, w2) = (0.5, 0.5)$.

The degree distribution we compare with is the distribution proposed in Raptor codes [5] and used by many unequal error protection methods like [6], [7], [8], [9]. We can observe that the simulation results of the ripple size variation are very close to the estimation results, as shown in Fig. 11.

We also use the proposed degree distribution in the LT codes. Additionally, we examine the performance of different degree distributions in terms of average recovery rates. We compare the average decoding failure rates for the proposed distribution and the other distributions in the literature [15][16]. The results are shown in Fig. 12, which indicates that the coding overhead using our proposed distribution is lower.

IV. CONCLUSION

In this study, we propose a novel method to identify better degree distributions for LMRC. The ripple size that occurs during the decoding process plays a crucial role in determining whether the decoding process can achieve the desired recovery rate. Therefore, we propose a safety criterion for ripple sizes with an allowable termination probability function according to the proposed leaping random walk model. Subsequently, we develop an estimation function for the ripple size variation in LMRC decoding process, utilizing the AND-OR tree analysis. To obtain the optimized degree distribution, we formulate a multi-objective optimization problem and employ the NSGA-II genetic algorithm. We design selection steps to choose a distribution candidate. Our results show that our proposed method accurately estimates the ripple sizes, leading to improved protection strength using the identified degree distribution. Finally, we demonstrate that the found distribution also performs well in LT codes.

REFERENCES

[1] I. S. Reed and G. Solomon, "Polynomial Codes Over Certain Finite Fields," Journal of the Society for Industrial and Applied Mathematics, 8, no. 2, pp. 300-304 ,June 1960.

[2] M. Mitzenmacher, "Digital Fountains: A Survey and Look Forward," IEEE Information Theory Workshop, 2004, pp. 271- 276, 2004.

[3] Palanki, Ravi, and Jonathan S. Yedidia. "Rateless codes on noisy channels," IEEE International Symposium on Information Theory, 2004.

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

[5] A. Shokrollahi, "Raptor Codes," IEEE Transactions on Information Theory, vol.52, pp. 2551-2567, June 2006.

[6] D. Sejdinovic, D. Vukobratovic, A. Doufexi, V. Senk and R. Piechocki, "Expanding Window Fountain Codes For Unequal Error Protection," IEEE Transactions on Communications, 57, no. 9 , pp. 2510-2516, September 2009.

[7] N. Rahnavard, B. N. Vellambi and F. Fekri, "Rateless Codes With Unequal Error Protection Property," IEEE Transactions on Information Theory, 53, no. 4, pp.1521-1532, April 2007. doi:10.1109/TIT.2007.892814.

[8] S. Ahmad, R. Hamzaoui and M. M. Al-Akaidi, "Unequal Error Protection Using Fountain Codes With Applications to Video Communication," IEEE Transactions on Multimedia, 13, no. 1, pp. 92-101, February 2011. doi:10.1109/TMM.2010.2093511.

[9] H-F Hsiao and Y-J Ciou, "Layer-Aligned Multipriority Rateless Codes for Layered Video Streaming," IEEE Transactions on Circuits and Systems for Video Technology, Volume 24. Issue 8, 2014.

[10] Q. Guan and S. Wu, "Research on the Rising Phenomenons in the Bit Error Rate Performances of LT-Based UEP Codes," Machine Learning and Intelligent Communications, 2022.

[11] Deb, Kalyanmoy, et al. "A fast and elitist multiobjective genetic algorithm: NSGA-II." Evolutionary Computation, IEEE Transactions on 6.2, 2002.

[12] K. T. Wallenius, "Biased sampling: The non-central hypergeometric probability distribution," Ph.D. dissertation, Dept. Statistics, Stanford Univ., Stanford, CA, USA, 1963.

[13] Maatouk, Ghid, and Amin Shokrollahi. "Analysis of the second moment of the LT decoder", IEEE Transactions on Information Theory, 2012.

[14] Hinterding, Robert. "Gaussian mutation and self-adaption for numeric genetic algorithms." Evolutionary Computation, 1995., IEEE International Conference on. Vol. 1. IEEE, 1995.

[15] Hansen, Nikolaus, and Andreas Ostermeier. "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. Evolutionary Computation, 1996., Proceedings of IEEE International Conference on.

[16] Chen, Chih Ming, et al. " On the optimization of degree distribution s in LT code with covariance matrix adaptation evolution strategy. Evolutionary Computation (CEC), 2010 IEEE Congress on.

[17] E. Hyytia, T. Tirronen and J. Virtamo, "Optimal Degree Distribution for LT Codes with Small Message Length," IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications, Anchorage.

[18] P. -C. Tsai, C. -M. Chen and Y. -p. Chen, "A novel evaluation function for LT codes degree distribution optimization," 2014 IEEE Congress on Evolutionary Computation (CEC), Beijing, China, 2014

[19] M.G. Luby, M. Mitzenmacher,and M. A. Shokrollahi, “Analysis of Random Processes via And-Or Tree Evaluation,” in Proc. 9th Ann. ACM-SIAM Symp.Discrete Algorithms, pp. 364–373, 1998.

[20] K. -K. Yen, Y. -C. Liao, C. -L. Chen and H. -C. Chang, "Modified Robust Soliton Distribution (MRSD) with Improved Ripple Size for LT Codes," in IEEE Communications Letters, vol. 17, no. 5, pp. 976-979, May 2013.

[21] Sorensen, Jesper H., Petar Popovski, and Jan Østergaard. "Design and analysis of LT codes with decreasing ripple size," Communications, IEEE Transactions on 60.11 (2012): 3191 3197.

[22] J. Rudnick and G. Gaspari, "Elements of the Random Walk: An Introduction for Advanced Students and Researchers," Cambridge University Press, 2004.

[23] Hižak, Jurica, and Robert Logožar. "A derivation of the mean absolute distance in one dimensional random walk," Tehnički glasnik 5.1, 2011

[24] Karande, Shirish S., et al. "Design and analysis of generalized LT codes using colored ripples," IEEE International Symposium on Information Theory, 2008.

[25] P. Wang et al., "Analysis of Rateless Multiple Access Scheme with Maximum Likelihood Decoding in an AWGN Channel," in IEEE Transactions on Wireless Communications, 2023.