PAPER

A Practical Optimization Framework for the Degree Distribution in LT Codes

Chih-Ming CHEN[†], *Nonmember*, **Ying-ping CHEN**[†a)], *Member*, **Tzu-Ching SHEN**[†], *and* **John K. ZAO**[†], *Nonmembers*

SUMMARY LT codes are the first practical *rateless* codes whose reception overhead totally depends on the degree distribution adopted. The capability of LT codes with a particular degree distribution named *robust soliton* has been theoretically analyzed; it asymptotically approaches the optimum when the message length approaches infinity. However, real applications making use of LT codes have finite number of input symbols. It is quite important to refine degree distributions because there are distributions whose performance can exceed that of the robust soliton distribution for short message length. In this work, a practical framework that employs evolutionary algorithms is proposed to search for better degree distributions. Our experiments empirically prove that the proposed framework is robust and can customize degree distributions for LT codes with different message length. The decoding error probabilities of the distributions found in the experiments compare well with those of robust soliton distributions. The significant improvement of LT codes with the optimized degree distributions is demonstrated in the paper.

key words: *LT codes, degree distribution, forward error correction, evolutionary algorithms, digital fountain*

1. Introduction

Luby transform (LT) codes [1] proposed by Michael Luby in 2002 are the first practical implementation of digital fountain codes. Two important features are considered in LT codes. First, the degree of each encoding symbol is decided by a particular probability distribution. Second, *belief propagation* algorithm is employed on the receiver side for decoding with less computational cost. Due to the features, the performance of LT codes totally depends on the adopted degree distribution. Based on a mathematical analysis, two degree distribution forms called *soliton distributions* were also presented in the proposal [1]. One of them has optimal performance in ideal case and the other is asymptotically close to optimal when message length approaches infinity. However, in practice, the message is usually divided into a finite, or moderate, number of input symbols according to different applications. Hence, efforts are needed to find better degree distributions for LT codes with a small number of input symbols.

In the present paper, an LT code optimization framework utilizing evolutionary algorithms is proposed to search for good degree distributions. Evolutionary algorithms are heuristic search technologies that solve problems by mimicking certain phenomena observed in nature. By developing the framework, we not only expect to find degree distributions with better performance than robust soliton distributions but also provide a tool that can customize degree distributions for different purposes such as lower error rate or lower reception overhead.

The remainder of the paper is organized as follows. Section 2 gives the related work on improving LT codes by refining degree distributions. Section 3 introduces the detailed coding mechanism of LT codes and two evaluation approaches of degree distributions. In Sect. 4, our optimization framework is proposed and described in detail. Several optimization results with different requirements are represented in Sect. 5, and the optimized distributions are compared with the original design of LT codes in Sect. 6. Finally, Sect. 7 concludes this paper.

2. Related Work

LT codes have been applied to many applications and are embedded as a basic component to develop other advanced rateless codes. Improving the performance of LT codes is therefore important if not necessary. Robust soliton distribution has been proven near optimal when the number of input symbols approaches infinity. Thus, most research topics focused on how to improve the performance of LT codes with a short message length [2], [3]. Reference [2] presented an approach to obtain the optimal degree distributions for a code length smaller than 30, while applications with such a scale can easily be handled by Gaussian elimination [4], [5]. To optimize LT codes with a greater size of input symbols, Hyytiä [6] made the first attempt to introduce heuristic search algorithms to optimize degree distributions for LT codes with code length 100, while good degree distributions are currently most in need for the range of the input symbol size from hundreds to ten thousands. Aiming at this range, our preliminary studies [7], [8] made use of evolutionary algorithms on the optimization of degree distributions. A related work [9] focused on maximizing the intermediate recovery rate by using multi-objective optimization algorithms. In these studies, real transmission was simulated to evaluate the performance of a particular degree distribution. The approach is feasible but not efficient. Based on a similar idea for optimization, a practical framework integrated with a more efficient evaluation approach is introduced in this paper to enhance the performance of LT codes

Manuscript received January 15, 2013.
Manuscript revised May 13, 2013.
[†]The authors are with the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan.
a) E-mail: ypchen@nclab.tw
DOI: 10.1587/transcom.E96.B.2807

and to enable the use of LT codes in the range of message lengths from hundreds to ten thousands.

3. Overview of LT Codes

Before the proposed optimization framework is described, an overview of the LT codes will be given in this section as a background. First, the encoding and decoding procedures are described in Sect. 3.1. Section 3.2 shows the two forms of *soliton degree distributions* whose performance has been confirmed. Furthermore, evaluating the quality of a degree distribution is a necessity in evolutionary algorithms. Section 3.3 introduces two methods for evaluating the decoding error probability of LT codes.

3.1 Encoding and Decoding

To apply LT codes, information messages in general are divided into k fragments of identical length called *input symbols*. In contrast, the generated code words are called *encoding symbols*. Before an encoding symbol is generated, a degree d is chosen at random according to the adopted degree distribution $\Omega(d)$, where $1 \le d \le k$ and $\sum_{d=1}^{k} \Omega(d) = 1$. The degree d is the number of distinct input symbols involved in composing an encoding symbol. d input symbols, also named *neighbors* of the encoding symbol, are then chosen uniformly at random and accumulated by XOR operations.

At the receiver side, when n encoding symbols are received, where n is slightly larger than k, belief propagation is used to reconstruct the source data step by step. All encoding symbols are initially covered in the beginning. For the first step, all encoding symbols with only one neighbor can be directly released to recover their unique neighbor. When an input symbol has been recovered but has yet to be processed, it will be queued in a set called the *ripple*. At each subsequent step, an unprocessed input symbol is chosen from the ripple and then removed from all the encoding symbols which have the chosen input symbol as a neighbor. If encoding symbols with only one remaining neighbor are generated after the removal, the releasing step repeats and may produce new members of the ripple to maintain a stable size of the ripple. Maintaining a sufficient size of the ripple is crucial because the decoding process fails when the ripple becomes empty with covered input symbols. In other words, more encoding symbols are required to resume the decoding process until all input symbols are recovered.

3.2 Soliton Distribution

The behavior of LT codes is determined by the degree distribution, $\Omega(d)$, and the number of received encoding symbols, n. The overhead $\varepsilon = n/k - 1$ measures the performance of LT codes and depends on the given degree distribution. Based on a theoretical analysis, Luby proposed the ideal soliton distribution of which the overhead is 0 in the ideal case.
Ideal soliton distribution $\rho(d)$:

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

Ideal soliton distribution guarantees that all the release probabilities are identical to $1/k$ at each decoding step. Hence, there is exactly one expected ripple generated at each step when the number of encoding symbols $n = k$. After k steps, the source data can be recovered.

However, ideal soliton distribution works poorly in practice. Belief propagation may be suspended by a small variance of the stochastic encoding/decoding process because the expected size of the ripple is only one at any moment. According to the random walk theory, the probability with which a random walk of length k deviates from its mean by more than $\ln(k/\delta)\sqrt{k}$ is at most δ, a baseline of the ripple size that must be maintained to complete the decoding process. Hence, in the same paper by Luby, a modified version called *robust soliton distribution* was also proposed.
Robust soliton distribution $\mu(d)$:

$$S = c \cdot \ln(k/\delta)\sqrt{k}$$

$$\tau(d) = \begin{cases} S/dk & \text{for} \quad d = 1, \ldots, k/S - 1 \\ S\ln(S/\delta)/k & \text{for} \quad d = k/S \\ 0 & \text{for} \quad d = k/S + 1, \ldots, k \end{cases}. \tag{2}$$

$$\beta = \sum_{d=1}^{k} (\rho(d) + \tau(d)) \tag{3}$$

$$\mu(d) = \frac{\rho(d) + \tau(d)}{\beta} \text{ for } d = 1, \ldots, k \tag{4}$$

Variables $c > 0$ and δ are two parameters for tuning the characteristic of robust soliton distribution. The distribution raises the expected ripple size from one to $\ln(k/\delta)\sqrt{k}$. Robust soliton distribution hence can ensure a successful decoding probability of at least 1-δ when $n = k \cdot \beta = k + O(\ln^2(k/\delta)\sqrt{k})$ encoding symbols are received.

Robust soliton distribution is practical based on an asymptotic analysis. However, in most situations, source data cannot be divided into an infinite number of pieces, and as a consequence, the behavior of LT codes do not exactly match the analytical result, especially when $k \ll \infty$. Robust soliton distribution gives a continuous upper bound of the overhead for LT codes with any k size, and there is still room to reduce the overhead by customizing degree distributions according to different numbers of input symbols.

3.3 Evaluation of Degree Distributions

An intuitive indicator to evaluate degree distributions is the reception overhead, ε. However, the encoding process of LT codes is stochastic, and uncertainty exists in the transmission channel. A successful decoding cannot be guaranteed in the condition of a constant overhead. To consider the overhead as a random variable, a large amount of simulations can be used to estimate the distribution and the expected value of ε, but a huge computational cost is required for obtaining precise results. Therefore, an alternative approach used in the proposed framework is to evaluate

the decoding error probability of LT codes with a particular reception overhead. In 2004, an effective evaluation method [10] was proposed for LT codes with a finite message length. Dynamic programming was utilized to construct the distribution of the ripple size during the decoding process. In short, the decoding error probability of LT codes can be deterministically evaluated by this method when the number of input symbols k and a constant overhead are given. Unfortunately, the computation considers a 3-dimensional matrix of size k^3. Even when several reduction techniques are applied, the computational complexity is still $O(k^3 \log^2(k) \log\log(k))$. Subsequently, a new method for a rigorous analysis on LT codes has been proposed in 2006 [11]. The difference in this approach is to make an assumption that the number of received symbols is a random variable with mean n. Based on the assumption, a fast evaluation with time complexity $O(k^2 \log(k))$ has been presented. Both the approaches are feasible to serve as an evaluation function in the framework and the second one with a lower computational cost was implemented in our experiments for efficiency. The approach can be denoted as function $f(k, \Omega, \varepsilon)$ which evaluates the error probability of LT codes when the code length is k, the degree distribution is Ω, and overhead ε has been received.

4. Optimization Framework

The paper employs optimization algorithms to search for good degree distributions for LT codes. The whole framework consists of three components, including an evolutionary algorithm, a distribution translation, and an evaluation function. This section will illustrate each of components and the cooperation between them.

4.1 Evolutionary Algorithms

Evolutionary computation [12], [13] is a branch of computational intelligence and widely used to solve real-world problems like searching, pattern design, and optimization. The key idea of evolutionary algorithms is to emulate the most important mechanism in natural, "survival of the fittest." In the emulation, each individual is represented by an encoded string like chromosomes in creatures. The representation form could be binary, integer, or real number strings depending on the problem's decision variables. These individuals are initialized randomly and evaluated according to the given objective function. Individuals with better fitness are selected with higher probabilities as survivors, which can reproduce offspring in the next generation. The reproduction process includes basic evolutionary operators such as crossover and mutation. Offspring individuals are then created to maintain the population usually of the same size as initialization. The evolution continues until some stopping condition is satisfied, which may be the fact that an individual with certain expected fitness is found or that a predefined number of generations for evolution is achieved.

There are many different evolutionary algorithms, and they can be roughly classified according to the types of decision variables. The optimization target in this work is the degree distribution of LT codes, which consists of probabilities and can be represented as real numbers. For the variable type, *covariance matrix adaption evolution strategy* (CMA-ES) [14] is introduced as the solution in this work. CMA-ES is a famous variant of *evolution strategies* (ES) [15], [16], which is an important family in evolution computation. Evolution strategy has been developed since early 1960s for solving optimization problems in industry. Decision variables in evolution strategy are denoted as a real number vector, and totally μ decision vectors are randomly initialized to form the initial population. For each generation, new λ decision vectors are sampled from the decision space by various operators, and then μ individuals with best fitness are selected as survivors to form the new population. Evolution strategy is denoted as $(\mu + \lambda)$-ES if the survivors are selected from the union of parents and offspring; (μ, λ)-ES selects survivors from only the λ offspring individuals. Evolution strategy can be summarized as several basic steps including new offspring sampling, fitness evaluation, and survival selection. As a variant of evolution strategy, CMA-ES has a similar structure described by the pseudo code presented in Algorithm 1, in which n is the dimensionality, and some strategic parameters are used to control the evolutionary operators. An important property of evolution strategies is self-adaptation, a mechanism that iteratively tunes the strategic parameters before the reproduction of new offspring. Inheriting the mechanism from ES, only the input parameters need to be decided for the use of CMA-ES because CMA-ES can adapt the other algorithmic parameters during the optimization process. Therefore, we can focus on the problem itself and employ CMA-ES as a black-box optimization tool. CMA-ES is adopted in this study not only because it is a popular real-valued parameter optimization method in evolutionary computation but also because its search ability has been theoretically analyzed and empirically verified on certain classic optimization problems, such as Ackley's function, Griewank's function, and Rastrigin's function. More details and variants of CMA-ES can be found in the literature [17]–[19]. The source code for the present study can be downloaded at [20].

Algorithm 1 Procedure of CMA-ES

Input parameters: μ, λ
Strategic parameters: m, σ, C
While not terminated
1: Sample new λ offspring $x_i \in \mathbb{R}^n$ for $i = 1, \ldots, \lambda$;
2: Evaluate the fitness value of each individual x_i;
3: Rank and select the best μ individuals;
4: Update m, σ and C according to the selected individuals;

4.2 Individual Representation

An essential step to employ evolutionary algorithms in general is to encode the decision variables of the problem at hand. More specifically, such an encoding, irrelevant to

communication coding, is a function that maps solution instances of a problem onto particular representations called *genotype*. The relationship can be understood as that various life types are decided by their own chromosome composition. Different encoding functions may influence the design of evolutionary operators and the size of the search space. The degree distribution is the optimization target in the present work and can be intuitively represented as a real-number vector of length k, depending on the size of input symbols. However, the challenge here is the number of decision variables, i.e., the problem dimensionality. Evolutionary algorithms have been well developed to handle dimensions from ten to hundreds, even a thousand. However, a problem of higher dimensionality means a larger search space has to be explored, and finding the optimal solution is usually harder. In other words, a huge computational cost is required to achieve the optimization for such problems of high dimensionality. To consider a successful decoding process, encoding symbols with degree one are necessary for triggering the belief propagation algorithm. Hence, a feasible degree distribution must have a nonzero probability on degree one. In contrast, the probabilities of the other degrees are allowed to be zeros to construct a workable distribution. As a result, an alternative representation called *sparse degree distribution* is suggested to reduce the number of dimensions. A sparse degree distribution has nonzero values on a set of discrete degrees which are user-defined and called *tags*. The representation has been widely adopted in the literature [7]–[9], [21]. In fact, all degrees should be considered as tags for searching for the global optimal distribution, but it is not practical when k is greater than 100 due to the optimization cost. Therefore, the proposed framework tries to find near-optimal distributions based on a given set of integers, i.e., tags. To choose appropriate tags is another interesting issue goes beyond the scope of this paper. A related investigation can be found in [22]. The investigation [22] shows that the functionality of a degree distribution can be approximated by a sparse distribution and the effect of approximation depends on the density of the tags. Based on the result, in this paper, Fibonacci numbers smaller than the source data length are suggested as tags to form the decision variable vector v and the degree distribution, represented as *sparse degree distribution* $\omega(d)$:

$$\omega(d) = \begin{cases} v(i) & d = \text{the } i\text{-th Fibonacci number} \\ 0 & \text{otherwise} \end{cases}, \qquad (5)$$

where k is the size of input symbols. The formula shows how to translate a real vector into a degree distribution, and it is what the distribution translator does in Fig. 1. For an example of $k = 12$, the nonzero probability is only allowed on degrees of tags: $\{1, 2, 3, 5, 8\}$. The decision variable vector $v = (0.2, 0.3, 0.3, 0.1, 0.1)$ represents the distribution $\omega = (0.2, 0.3, 0.3, 0, 0.1, 0, 0, 0.1, 0, 0, 0, 0)$. The authors would like to emphasize that Fibonacci sequence is feasible but not the only feasible setting to be selected as tags. The reason to choose tags in this way is for the consistency between different k's such that the optimized results of degree distributions can be systematically compared.

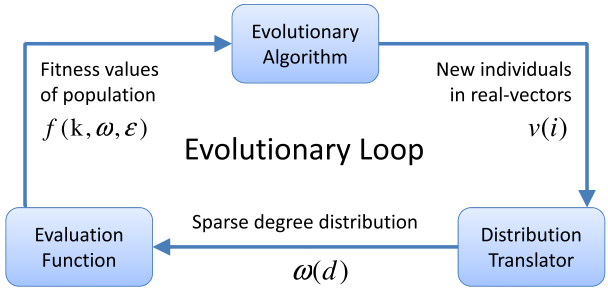


Fig. 1 Basic components and evolutionary loop adopted in the proposed framework.

4.3 Optimization Flow

Figure 1 shows the flow chart of the evolutionary loop, in which each optimization component has been introduced. The three components are independent and can be replaced with any feasible, equivalent module. The first block includes an evolutionary algorithm which is the core of the framework and CMA-ES is employed in this paper. At the beginning, the input parameters λ and μ are given and then the initial population consists of random real vectors are created for the first generation. The next block is the distribution translator to translate a real vector into a reasonable degree distribution as aforementioned. Usually, the initial random vectors and later, new individuals, $v(i)$, created by general-purpose evolutionary operators would not be valid probability distributions. For this reason, a normalization step is needed to normalize arbitrary real-number vectors to be probability distributions before the representation translation. Such an operation is easy and does not change the feasibility of the proposed framework, although the computational complexity may be slightly increased. After normalization, the real vectors, $v(i)$, are translated into the proper form of sparse degree distributions, $\omega(d)$, according to the expression (5. The last block denotes the evaluation function. Two evaluation methods for degree distributions of LT codes are introduced in Sect. 3.3. The second one [11] with a lower computational complexity is implemented as the function $f(k, \omega, \varepsilon)$ to evaluate the error probability of the individuals as their fitness. Finally, the fitness information feeds back to the evolutionary algorithm for selecting the survivors for next generation. After a fixed number of iterations, the best individual discovered by the algorithm will be recorded as the final result.

5. Experiments

In the present work, a general optimization framework that searches for good degree distributions for LT codes is proposed. The proposed framework can be operated with ease because a small number of parameters need be determined. The first part of this section will explain the parameters of our experiments and present the optimization results. In the

second part, a comparison between full and sparse degree distributions is given to demonstrate that the degree reduction is feasible and practical. Note that all the experimental results in this section are averaged over 30 independent runs because of the stochastic nature of evolutionary algorithms.

5.1 Optimization Results

To ensure the readers to be able to reproduce the experiment, all the detailed settings will be clearly stated. CMA-ES is the evolutionary algorithm employed in this work, and the source code of CMA-ES in Matlab is available on the webpage of CMA-ES [20]. CMA-ES has the ability to self-adapt the algorithmic parameters, and the default, recommended parameter values except for the maximum number of function evaluations are used in this study. The evolutionary process will terminate when the maximum number of function evaluations is reached. The value controls the expected computation cost for each optimization run and roughly corresponds to the execution time because the evaluation function is usually considered the most expensive part in evolutionary optimization. This limitation is identical to the use of maximum number of generation because the population size is fixed in the present work.

Figure 2 presents the results of optimization experiments for sizes 100, 400, 700, and 1000 of input symbols. The input parameters of CMA-ES are given as $\mu = 5$ and $\lambda = 10$. The maximum number of function evaluation is 5000, the evaluation overhead is 0.1, and distributions with sparse degree are adopted. Based on our observation, the value 0.1 is suitable for an length of input symbols ranging from 100 to 1000 such that the experimental results can be clearly plotted in the same figure. The users make use of our framework can customize the value for different requirements. The plotted data are averaged result over 30 independent runs of the optimization experiment. Partial information of the standard derivation is presented in the figure for a clear display. The variation of the standard derivation indicates the convergence of searching. As the evolution proceeds, the average error probabilities decrease and converge to different levels according to k. The results conform to the property of LT codes that a greater length of message brings a lower reception overhead; in other words, a lower error probability can be achieved for a constant overhead. In Fig. 3, the optimized sparse degree distributions obtained with CMA-ES are plotted. Although the degree distribution seems similar to robust soliton distributions, they are in fact quite different from robust soliton distributions because there exist many tags with zero probability omitted in the figure. The performance for LT codes with sparse tags will be examined and shown to be comparable to that of LT code with full tags in Sect. 6.

As for the overhead, we can examine the experimental results presented in Figs. 4 and 5, in which $k = 1000$ and $\varepsilon \in \{0.02, 0.05, 0.1, 0.2\}$. According to Fig. 4, for $\varepsilon \in \{0.02, 0.05\}$, the error probability cannot be appropriately reduced. This means that it is extremely hard to search

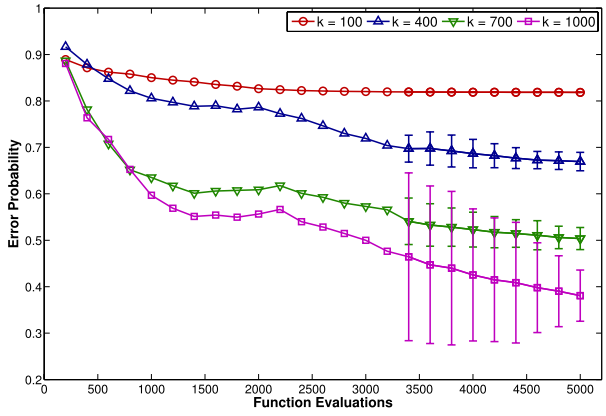


Fig. 2 Error probability variation during the optimization processes for different input symbol sizes. For a clear display, partial information of the standard derivation is presented.

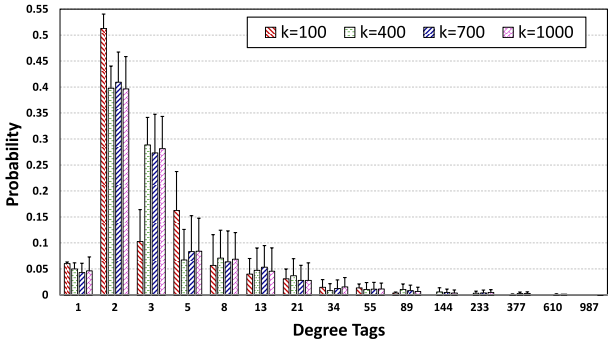


Fig. 3 Optimized sparse degree distributions obtained by CMA-ES for different input symbol sizes.

for good degree distributions with such a condition or perhaps LT codes cannot work properly with such overheads. When $\varepsilon \in \{0.1, 0.2\}$, the proposed framework starts to reduce the error probability significantly. The degree distributions corresponding to low error probabilities can be found at earlier stages for a greater overhead. Figure 5 shows the obtained degree distributions and indicates the relationship between the error probability and the degree distribution.

We can observe that the obtained degree distributions are similar but not identical. There is little difference between them, especially between cases $\varepsilon = 0.02$ and $\varepsilon = 0.2$. Such a phenomenon reveals that the proposed optimization framework is able to search for good degree distributions for the given overhead. The framework is effective to seek the lower bound of decoding error probability for given overheads and helps researchers to explore the limitation of LT codes. The outcome is also consistent with the results presented in our previous study [7], in which we used transmission simulation to evaluate the overhead of a degree distribution and found that LT codes might start to work properly when ε was close to or greater than 0.1 for $k = 1000$.

5.2 Full Degrees

In the second experiment, the optimization of degree distri-

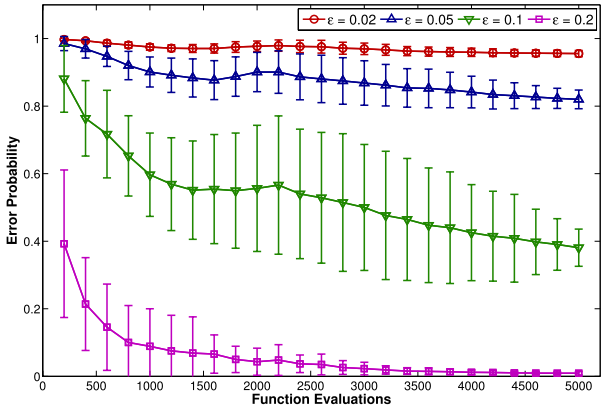


Fig. 4 Error probability variation during the optimization processes for different reception overheads.

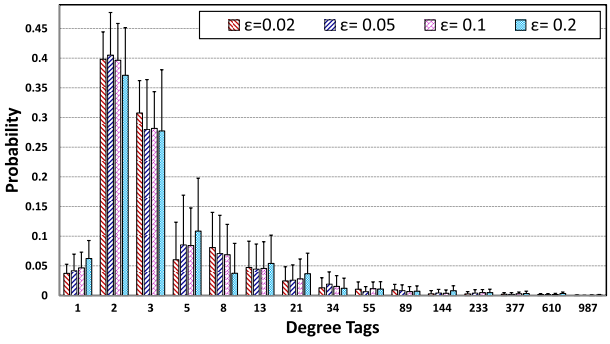


Fig. 5 Optimized sparse degree distributions obtained by CMA-ES for different reception overheads.

butions composed of different types of tags is investigated. To decrease the dimensionality, the use of sparse degree distributions is suggested. Such a representation effectively reduces the search space, but the possible degree distributions are apparently limited by not considering all the degrees. The intention of this experiment is to examine whether the effectiveness of the proposed framework is impacted by the adoption of sparse degree distributions. Figure 6 presents the comparison between sparse and full degree distributions. The full degree distribution is defined as a probability distribution with values at each degree from 1 to k. Only one experiment in which $k = 100$ was conducted because of the very high computational cost to optimize full degree distributions. The maximum number of function evaluation hence is set to 30000 in this experiment. It is clear that the optimization of full degree distributions converges slowly and reaches almost the same error probability at the final stage. Figure 7 shows the obtained full degree distribution after the optimization. To compare with the sparse degree distribution in Fig. 3, the distributions seem quite different, while a series of examinations presented in the next section demonstrates a similar level of performance.

6. Investigation into the Optimization Results

In the previous section, several degree distributions with

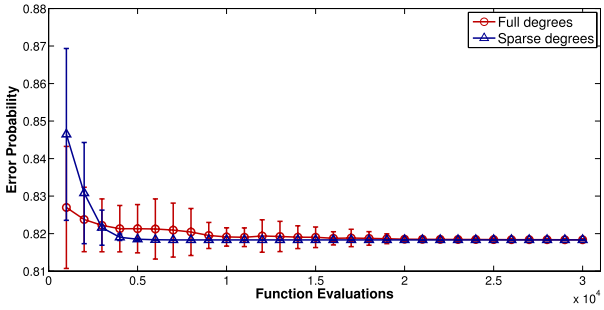


Fig. 6 Optimization results of full and sparse tags are compared. More function evaluations are required for full tags.

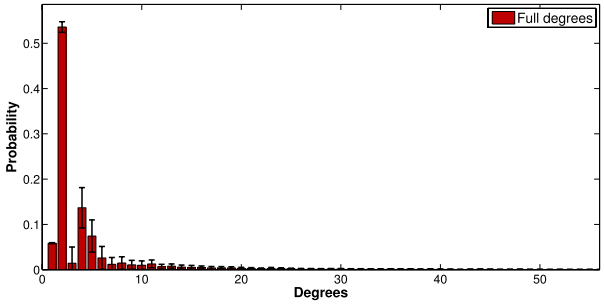


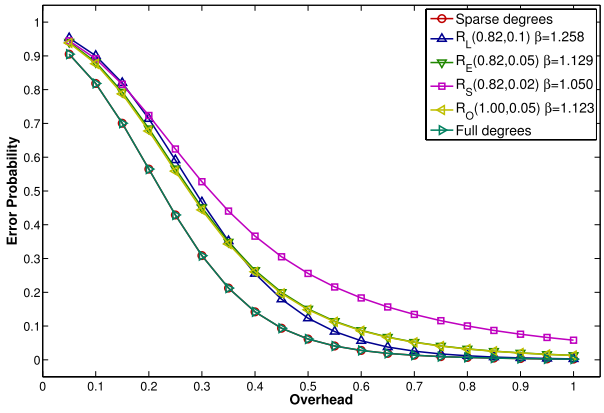
Fig. 7 Full degree distributions optimized by CMA-ES. Partial degree values with 99% probability density are plotted.

a low error probability were found with a given reception overhead. Table 1 lists the best degree distributions and the corresponding error probabilities in the 30 independent runs of our experiments. In the table, the entries that are not applicable are left blank. Furthermore, some tags with zero probability after optimization give us an empirical evidence that the use of sparse degree distributions is valid. These degree distributions will be further examined with different evaluation methods and compared with robust soliton distributions. As Eq. (3), the characteristic of robust soliton distributions is determined by two parameters, δ and c. Luby gave the theoretical analysis [1] to show that LT codes with such robust soliton distributions can ensure a successful decoding process with a failure probability δ when $k \cdot \beta$ input symbols are received. Thus, the three robust soliton distributions are chosen with a fixed δ that equals the optimization results in Table 1 and different c values to make their corresponding β satisfying the three scenarios, larger than (R_L), smaller than (R_S), and roughly equivalent (R_E) to 1.1. The control group was selected to show robust soliton distributions with different properties. For a fair comparison, the proposed optimization framework was also utilized to minimize the error probability of robust soliton distribution for LT codes. Decision variables in the case are two parameters of robust soliton, δ and c. The optimized robust soliton distribution is denoted as (R_O) and joins the control group.

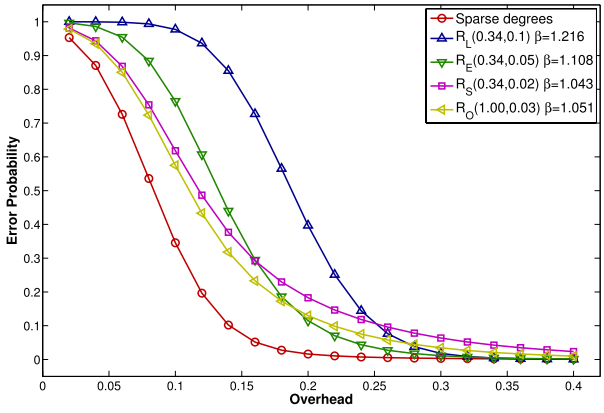
In the first evaluation method [11], the error probability is exactly calculated when a reception overhead was received on average. Each degree distribution is compared by an investigation consisting of a reception ratio sweep.

Table 1 The best degree distributions discovered in each 30-run experiment.

Tags	k=100	k=400	k=700	k=1000			
	$\varepsilon = 0.1$	$\varepsilon = 0.1$	$\varepsilon = 0.1$	$\varepsilon = 0.02$	$\varepsilon = 0.05$	$\varepsilon = 0.1$	$\varepsilon = 0.2$
1	0.0579	0.0310	0.0225	0.0178	0.0211	0.0244	0.0191
2	0.5368	0.4903	0.5064	0.5014	0.4965	0.4592	0.5483
3	0.0481	0.1692	0.1301	0.1448	0.1419	0.2498	0.0359
5	0.2332	0.1096	0.1868	0.1963	0.1909	0.0212	0.2591
8	0.0009	0.1034	0.0165	0.0000	0.0394	0.1628	0.0120
13	0.0618	0.0000	0.0734	0.0684	0.0092	0.0103	0.0017
21	0.0322	0.0638	0.0104	0.0383	0.0730	0.0032	0.0874
34	0.0094	0.0000	0.0263	0.0040	0.0000	0.0470	0.0000
55	0.0163	0.0152	0.0000	0.0000	0.0000	0.0035	0.0002
89	0.0035	0.0000	0.0204	0.0227	0.0210	0.0000	0.0193
144		0.0162	0.0001	0.0000	0.0003	0.0116	0.0017
233		0.0000	0.0002	0.0000	0.0000	0.0000	0.0092
377		0.0013	0.0069	0.0055	0.0062	0.0058	0.0000
610			0.0000	0.0000	0.0000	0.0002	0.0051
987				0.0000	0.0000	0.0000	0.0000
Error prob.	0.8183	0.6547	0.4836	0.9488	0.8000	0.3456	0.0057

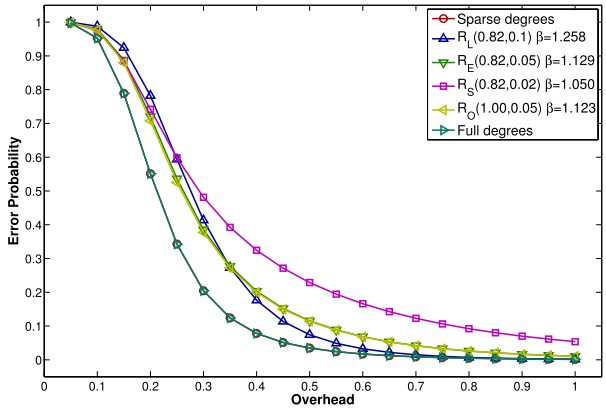


(a) $k = 100$

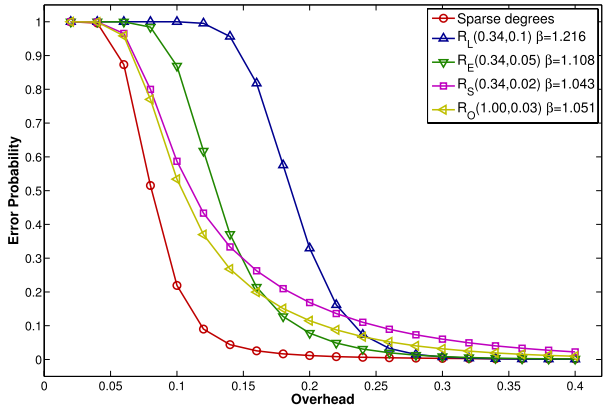


(b) $k = 1000$

Fig. 8 Error probability of LT codes with the assumption that an average reception overhead is received.



(a) $k = 100$



(b) $k = 1000$

Fig. 9 Error probability of LT codes with the assumption that an exact reception overhead is received.

Figure 8 shows the performance curves of the optimized distributions for overhead $\varepsilon = 0.1$. The optimized sparse degree distributions clearly outperform all the robust soliton distributions with different δ and c in both $k = 100$ and $k = 1000$. We can find that the performance curves of robust soliton distributions do not match their parameter settings that the decoding error probability δ should be achieved when $\varepsilon = \beta - 1$. It can be understood that the theoretical analysis on robust soliton distributions is based on the assumption that $k = \infty$. For this reason, there exists

a gap between theory and practice for finite input symbols, and the gap becomes obvious when k decreases. In addition, the optimized full degree distribution joins the comparison for $k = 100$, and a very similar result to that of sparse degree distributions was obtained. Such an evidence supports the same result in [22] that the functionality of full degree distributions can be approximated well by sparse ones for the use of LT codes.

From the viewpoint of the receiver side, to consider the error probability of LT codes when an exact number of encoding symbols is received is intuitive. Another evaluation method [10] based on a different assumption is employed to investigate the degree distributions. The results are presented in Fig. 9, and sparse degree distributions still have better performance than robust soliton distributions do. Even though the results of R_O are comparable at a lower reception overhead, R_O is unable to provide a certain level of reliability to ensure a successful decoding process. Finally, the comparison for $k = 100$ again confirms that sparse and full degree distributions have very similar performance.

7. Conclusions

In this work, an optimization framework employing evolutionary algorithms as an optimization engine was proposed to search for good degree distributions to work with LT codes. Firstly, several experiments were conducted to verify the robustness of the proposed framework, and then, the optimized degree distributions were fully presented. These degree distributions were compared with control groups of different robust soliton distributions. It has been shown that the optimized degree distributions outperform robust soliton distributions by evaluating them with two different methods. In summary, the two major contributions are (1) to propose a flexible optimization framework for the degree distribution of LT codes and (2) by presenting examples, to demonstrate the existence of the degree distributions of which the performance is better than that of robust soliton distributions.

There is no doubt that LT codes are a very important implementation of digital fountain codes, and LT codes have been widely used in many areas. However, there exists a significant amount of applications that are limited to adopting such a technique due to the data size. Good degree distributions are necessary to boost the performance of LT codes and to support LT codes to be used in these practical applications in which the number of input symbols is smaller than ten thousands. An effective and flexible solution for the issue has been proposed in the present work. The improvement makes it possible to utilize LT codes in various scenarios in the future. Moreover, the adopted evaluation function is computed with dynamic programming. It is easy to compute the error probability in the same way for recovery 80%, 90%, or 95% of the input symbols. Thus, the proposed framework can also assist Raptor codes designers to customize the weakened LT code for the use of different precodes.

Acknowledgments

The work was supported in part by the National Science Council of Taiwan under Grant NSC 99-2221-E-009-123-MY2. The authors are grateful to the National Center for High-performance Computing for computer time and facilities.

References

[1] M. Luby, “LT codes,” Proc. 43rd Annual IEEE Symposium on Foundations of Computer Science, pp.271–280, 2002.
[2] E. Hyytiä, T. Tirronen, and J. Virtamo, “Optimal degree distribution for LT codes with small message length,” Proc. 26th IEEE International Conference on Computer Communications (INFOCOM 2007), pp.2576–2580, 2007.
[3] E.A. Bodine and M.K. Cheng, “Characterization of Luby Transform codes with small message size for low-latency decoding,” Proc. IEEE International Conference on Communications (ICC 2008), pp.1195–1199, 2008.
[4] V. Bioglio, M. Grangetto, R. Gaeta, and M. Sereno, “On the fly gaussian elimination for LT codes,” IEEE Commun. Lett., vol.13, no.12, pp.953–955, 2009.
[5] M. Rossi, N. Bui, G. Zanca, L. Stabellini, R. Crepaldi, and M. Zorzi, “Synapse++: Code dissemination in wireless sensor networks using fountain codes,” IEEE Trans. Mobile Comput., vol.9, no.12, pp.1749–1765, 2010.
[6] E. Hyytiä, T. Tirronen, and J. Virtamo, “Optimizing the degree distribution of LT codes with an importance sampling approach,” Proc. 6th InternationalWorkshop on Rare Event Simulation (RESIM 2006), pp.64–73, 2006.
[7] C.M. Chen, Y.P. Chen, T.C. Shen, and J.K. Zao, “On the optimization of degree distributions in LT code with covariance matrix adaptation evolution strategy,” Proc. IEEE Congress on Evolutionary Computation (CEC 2010), pp.3531–3538, 2010.
[8] C.M. Chen, Y.P. Chen, T.C. Shen, and J.K. Zao, “Optimizing degree distributions in LT codes by using the multiobjective evolutionary algorithm based on decomposition,” Proc. IEEE Congress on Evolutionary Computation (CEC 2010), pp.3635–3642, 2010.
[9] A. Talari and N. Rahnavard, “Rateless codes with optimum intermediate performance,” Proc. 28th Global Telecommunications Conference (GLOBECOM 2009), pp.4197–4202, 2009.
[10] R. Karp, M. Luby, and A. Shokrollahi, “Finite length analysis of LT codes,” Proc. IEEE International Symposium on Information Theory 2004 (ISIT 2004), p.39, 2004.
[11] E. Maneva and A. Shokrollahi, “New model for rigorous analysis of LT-codes,” Proc. IEEE International Symposium on Information Theory (ISIT 2006), pp.2677–2679, 2006.
[12] A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing, SpringerVerlag, 2003.
[13] T. Bäck, D.B. Fogel, and Z. Michalewicz, Handbook of Evolutionary Computation, 1st ed., Institute of Physics Publishing, Bristol, UK, 1997.
[14] N. Hansen and A. Ostermeier, “Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation,” Proc. IEEE International Conference on Evolutionary Computation (ICEC’96), pp.312–317, 1996.
[15] I. Rechenberg, Evolutionsstrategie; Optimierung technischer Systeme nach Prinzipien der biologischen Evolution, Frommann-Holzboog, Stuttgart, 1973.
[16] H.G. Beyer and H.P. Schwefel, “Evolution strategies — A comprehensive introduction,” Natural Computing, vol.1, no.1, pp.3–52, 2002.
[17] A. Auger and N. Hansen, “Performance evaluation of an advanced

local search evolutionary algorithm," Proc. IEEE Congress on Evolutionary Computation (CEC 2005), pp.1777–1784, 2005.
[18] A. Auger and N. Hansen, "A restart CMA evolution strategy with increasing population size," Proc. IEEE Congress on Evolutionary Computation (CEC 2005), pp.1769–1776, 2005.
[19] R. Ros and N. Hansen, "A simple modification in CMA-ES achieving linear time and space complexity," Proc. 10th International Conference on Parallel Problem Solving from Nature: PPSN X, pp.296–305, Springer-Verlag, 2008.
[20] N. Hansen, "CMA-ES source code." http://www.lri.fr/~hansen/cmaes_inmatlab.html accessed May 9, 2013.
[21] A. Shokrollahi, "Raptor codes," IEEE Trans. Inf. Theory, vol.52, no.6, pp.2551–2567, 2006.
[22] P.C. Tsai, C.M. Chen, and Y.P. Chen, "Sparse degrees analysis for LT codes optimization," Proc. IEEE Congress on Evolutionary Computation (CEC 2012), pp.2463–2468, 2012.

Chih-Ming Chen received the B.S. degree in 2006 and currently studies for the Ph.D. degree in Computer Science at National Chiao Tung University, Taiwan.

Ying-ping Chen is currently an Associate Professor in the Department of Computer Science, National Chiao Tung University, Taiwan. His research interests include data grid and MapReduce technologies in distributed computation as well as theories, working principles, and dimensional/facet-wise models in genetic and evolutionary computation. He received the B.S. degree and the M.S. degree in Computer Science and Information Engineering from National Taiwan University, Taiwan, in 1995 and 1997, respectively, and the Ph.D. degree in 2004 from the Department of Computer Science, University of Illinois at Urbana-Champaign, Illinois, USA.

Tzu-Ching Shen received the B.S. degree and the M.S. degree in Computer Science and Information Engineering from National Chiao Tung University, Taiwan, in 2008 and 2010.

John K. Zao is currently an Associate Professor of the Computer Science Department of the Chiao Tung University in Hsinchu, Taiwan. Before returning to Asia, he has served as a Senior Member of Technical Staff (1994–1999) and then a Principal Member of Technical Staff (1999–2002) in the Information Security Department of BBN Technologies, the research company that pioneered the packet-switching ARPANet. During his tenure at BBN, he served as the principal investigator of several US-DARPA funded research projects including: Security Architecture for Global Mobile IP Support (MoIPS), Policy Based Security Management (PBSM), Policy Management for Defense Information Assurance Networks (Pledge) and Formal Information Assurance System Modeling (AVDA). His current research interests include: H.264 SVC Heterogeneous Multicasting, Sensor/Actuator Infrastructure for Assistive Living and Multi-Tier Fog Computing Infrastructure for Mobile Computing. Dr. Zao was born in Hong Kong. He received B.A.Sc. degree in Engineering Science and M.A.Sc. degree in Electrical Engineering from University of Toronto and later S.M. and Ph.D. degrees in Computer Science from Harvard University. Dr. Zao was elected an IEEE Senior Member in 2001.