Distributed LT Codes on Multi-hop Networks

Yan Zhang[1,a], Jianhua Chen[2], Meng Tang[2], Xin Jin[1]

[1]School of Software, Yunnan University, Kunming 650500, China

[2]School of Information, Yunnan University, Kunming 650500, China

[a]zy_helen@ynu.edu.cn

Keywords: distributed LT codes, binary erasure channels, multi-hop networks, And-Or Tree, asymptotic performance, linear optimization.

Abstract. Distributed LT codes on binary erasure channels (BECs) have been widely investigated, but the number of sources was usually limited within ten. In the practical cases we often need to transmit information blocks from scores of sources to a common destination through one or more relays, which encourages us to reclaim the scenarios in which up to one hundred sources transmit information blocks to the destination. Obviously, in these scenarios, sources need to be re-organized to improve the system performance. Our propositions are Multi Distributed LT codes (MDLT) and Backtracking Distributed LT codes (BDLT). We investigate the performances of these codes by density evolution, and present the simulating results, compared with those Conventional Distributed codes (CDLT).

1. Introduction

Fountain codes [1][2][3] are a novel forward error correction scheme for information transmission over Binary Erasure Channels (BEC). Unlike traditional ARQ schemes, fountain codes are able to adapt their rates on-the-fly, so they are also called rateless codes. In practical circumstances such as wireless networks, when the characteristics of the channels are unknown or time-varying, fountain codes can obtain much better performance than ARQ schemes because they need not repeat and request every erased packet. LT codes[2] are the first realization of fountain codes. Discarding Gaussian elimination decoding whose computational cost is O(k3) (where k is the number of packets in an original source block), LT codes utilize Belief Propagation decoding which can cut the computational cost down to O(k logk). But LT codes suffer from a relatively high error floor. To eliminate this problem Shokrollahi proposed a compound coding structure named Raptor codes [3], usually including a high-rate outer LDPC code and an inner LT code. Raptor codes could not only abate the error floor, but also reduced the computational cost to O(k).

LT codes and Raptor codes are all suit well for point-to-point transmissions, but in the multicasting networks, we need multi sources transmitting to one destination, i.e. distributed LT codes. Distributed LT codes is the engagement of LT codes and Network Coding [4][5]. Since the structure of Network Coding would destroy the degree distribution of LT codes, people must take some measures to overcome it. [6] firstly explore distributed LT codes by deconvolving the degree distribution of LT codes, named Robust Soliton Distribution (RSD), in order to obtain an approximate RSD at the destination. Considering the complexity of deconvolution, this approach supports only 2-sources networks and 4-sources networks. And-Or Tree provides a unifying, intuitive, and powerful framework for carrying out the analysis of several random including random loss-resilient codes, random k-SAT formulae using the pure literal rule, the greedy algorithm for matchings in random graphs, etc [7], thereafter Sejdinovic utilize it to analysis the asymptotic performance of Distributed LT codes [8]. And-Or Tree analysis provides a pre-view of the proposed codes before they are simulated with real numerical parameters. In [10] the author proposed a Binary soliton-like rateless coding for the two-sources-one-relay network, and [11] proposed an improved edition with fewer size of buffers and low-complexity. In these two contributions information blocks are coded with the Robust Soliton Distribution (RSD) but re-coded through a special procedure in the relay to produce a soliton-like degree distribution at the destination. To increase the maximum degree of the distribution of the relay, [12] proposed buffer-based distributed LT codes, which can raise the maximum degree up to 10 in a 4-sources network. And to maximize the minimum-distance of variable-node degree, [13] proposed the Rescheduled distributed LT Codes, which can abate the error floor remarkably. [14] proposed a Hybrid Soliton Distribution Coding which engage the Robust Soliton Distribution (RSD) and the deconvolved Robust Soliton Distribution (DSD) [6], but it can only deal with the 2n-sources networks.

In these previous contributions, the number of sources is always not so large – no more than 10. We explore the scenarios that scores of sources transmitting to one destination, and find out that Multi Distributed LT Codes (MDLT) work quite excellently. We analysis the asymptotic performance of MDLT, and demonstrate the numerical simulating results compared to the conventional Distributed LT Codes (CDLT). We also offer a Backtracking Distributed LT codes (BDLT) for some special scenes which MSDLT cannot be configured but erasure ratios need to be decreased.

2. LT and DLT Codes

2.1 LT Codes

Unlike traditional fixed-rate block codes, an LT encoder generates an arbitrary number of coded symbols from a finite set of K information symbols. The number n of coded symbols is usually slightly larger than K, and n is expressed as K(1+ε), where ε is a relatively little number and is named "the transmitting overhead". Normally, larger ε leads to lower erasure rate, but brings greater transmitting overload.

The coding of an LT code is a linear mapping form $\mathbb{F}_2^k$ to $\mathbb{F}_2^n$. Let a binary information block X=(x1, x2, ... , xk) consist of k bits, each bit represents a variable node in the coding and decoding graph . Through a generator matrix G=(g1, g2, ... , gn) the variable nodes are mapped to the check nodes Y=(y1, y2, ... , yn):

X· G =X· (g1, g2, ... , gn) = (Xg1, Xg2, ... , Xgn) = (y1, y2, ... , yn) = Y (1)

Obviously, the generator matrix G is the heart of the LT code. Considering Hamming weights of each column vectors of G, a key concept is raised as the node-perspective check-node degree distribution $\Omega(x)=\sum_{d=1}^{d_{\Omega,\max}}\Omega_d x^d$. When generating a column vector gi, we ensure gi has a Hamming weight d with the probability Ωd. d is called the degree of this newly generated check node yi. In [2] Luby proposed a well-performed distribution for Ωd , Robust Soliton Distribution (RSD). If the positions of ones in the binary vector gi are uniformly-at-random selected in the whole column, this LT code is an Equal Erasure Protection (EEP) code, otherwise it is an Unequal Erasure Protection (EEP) code [9].

The check-node degree distribution Ω(x) describes how much variable nodes are linear combined to obtain a check node, thus in a contrary way, how much a variable node are selected to produce check nodes? Considering the generator matrix G, the Hamming weights of each row vector would present the pattern. We term the node-perspective variable-node degree distribution as $\Lambda(x)=\sum_{d=1}^{d_{\Lambda,\max}}\Lambda_d x^d$. Obviously as the information bits are selected uniformly at random by the check nodes, the variable-node degrees are binomially distributed $\Lambda_d=\binom{n}{d}\left(\frac{\mu}{k}\right)^d\left(1-\frac{\mu}{k}\right)^{n-d}$, where $\mu=\Omega'(1)$ is the average degree of the check nodes. As k grows large the binomial distribution can be accurately approximated by a Poisson distribution $\Lambda_d\approx e^{-\alpha}\alpha^d/d!$, where $\alpha=n\cdot\frac{\mu}{k}=(1+\varepsilon)\Omega'(1)=\Lambda'(1)$ is the average degree of variable nodes. So the variable-node degree distribution is approximated by

$$\Lambda(x)\approx e^{\alpha(x-1)} \tag{2}$$

We term Pl as the erasure probability of a variable node after l iterating decoding loops, following the Lemmas in [7], the asymptotic performance of a LT code is:

$$\begin{aligned} &P_0=1,\\ &P_l=\lambda\left(1-\omega(1-P_{l-1})\right) \end{aligned} \tag{3}$$

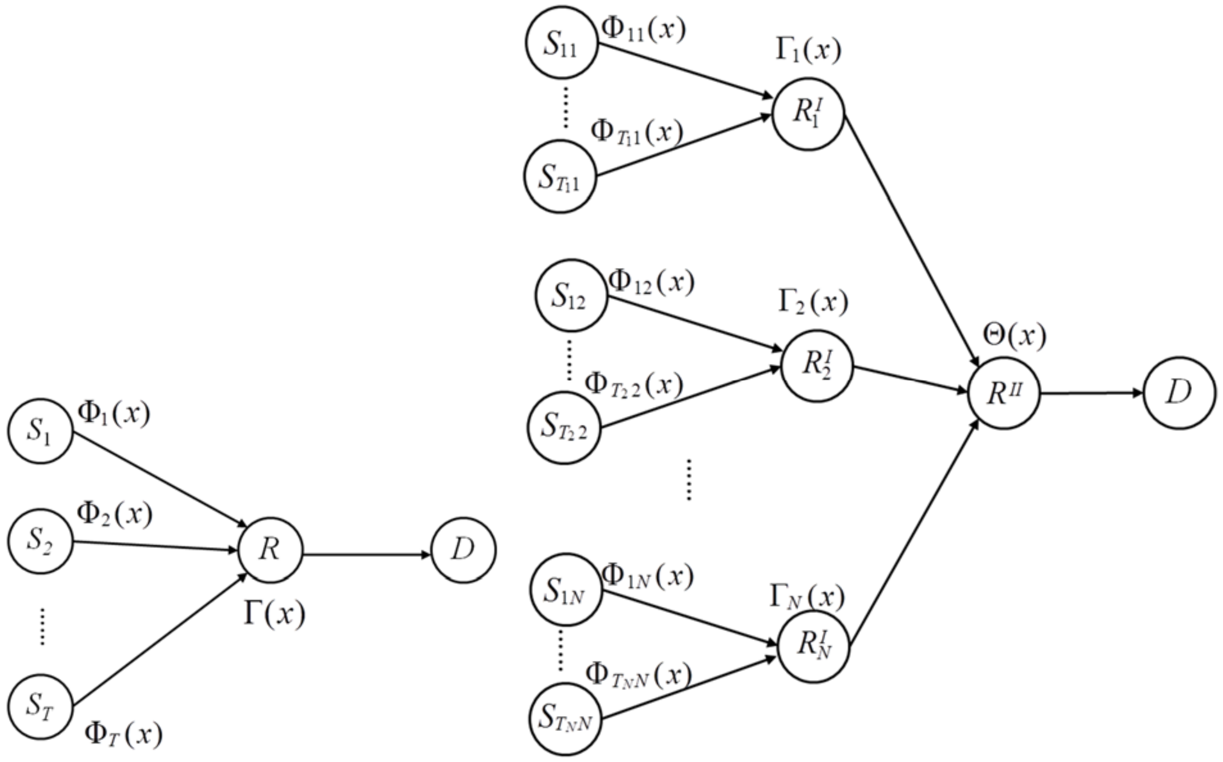
where $\lambda(x)=\Lambda'(x)/\Lambda'(1)$ and $\omega(x)=\Omega'(x)/\Omega'(1)$ are the edge-perspective variable-node degree distribution and edge-perspective check-node degree distribution , respectly. From formula (2) we can easily get $\lambda(x)=e^{\alpha(x-1)}$, so formula (3) can be expressed as

$$\begin{aligned} &P_0=1,\\ &P_l=\exp\left(-\alpha\cdot\omega(1-P_{l-1})\right) \end{aligned} \tag{4}$$

Note that α can be expressed in terms of the overhead ε as $\alpha=(1+\varepsilon)\Omega'(1)$, so formula (3) can also be expressed as the function of overhead ε:

$$\begin{aligned} &P_0=1,\\ &P_l=\exp\left(-(1+\varepsilon)\cdot\Omega'(1-P_{l-1})\right) \end{aligned} \tag{5}$$

2.2 DLT Codes



(a) (b)

Fig. 1(a) System model of DLT codes; (b) System Model of MDLT

The system model of the DLT codes is shown in Fig. 1(a). Information bits are coded at the corresponding source through degree distributions Φ1(x), Φ2(x), ... , ΦT(x), where T is the number of the sources. After source-coded bits are transmitted to the relay, the relay selects some sources through a degree distribution termed as Γ(x), re-coded the bits transmitted from them into one bit, and forward it to the destination. Let the binary information blocks are termed as X1, X2, ... , XT, each block has a sequence of information bits (x1, x2, ... , xk). Then the corresponding coding and decoding graph is 1, 2, ... , T. Expressed in terms of generator matrices, the coding procedure is

$$X\cdot G=(X_1,X_2,\cdots,X_T)\cdot\begin{pmatrix}G_1\\G_2\\\vdots\\G_T\end{pmatrix}=Y \tag{6}$$

Considering the overall generator matrix G, regard it as a collection of column vectors, so each vector gi is a generator vector, and each generator vector will generate a coded bit. gi is combined by T portions, and the j-th portion is delivered from the source j. Not each portion will be selected to assemble gi, in fact, we choose d portions with probability Γd, and the selection procedure is uniformly-at-random, as we did in the source-coding phase.

$\Gamma(x)=\sum_{d=1}^{d_{\Gamma,\max}}\Gamma_d x^d$ is called the relay degree distribution, and obviously, the maximum degree Γmax is no larger than the number of the sources T. Abide by the conclusions in [8], if Φ(x) = Φ1(x)= Φ2(x)= ... = ΦT(x), the overall degree distribution of multi-coded bits at the destination is the nested function Ω(x)=Γ(Φ(x)). And the degree distribution of variable nodes remain the Poisson distribution $\Lambda(x)=e^{\bar{\alpha}(x-1)}$, where $\bar{\alpha}=(1+\varepsilon)\cdot\Gamma'(1)\cdot\Phi'(1)$ is the average degree of variable nodes. The edge-perspective variable-node degree distribution $\omega(x)=\Omega'(x)/\Omega'(1)$, substituting Ω(x) with Γ(Φ(x)), we can get ω(x)=φ(x)·γ(Φ(x)), where $\varphi(x)=\Phi'(x)/\Phi'(1)$ and $\gamma(x)=\Gamma'(x)/\Gamma'(1)$. So the asymptotic performance of a DLT code is given by:

$$P_0=1$$
$$P_l=\exp\left[-\bar{\alpha}\varphi(1-y_{l-1})\gamma\left(\Phi(1-y_{l-1})\right)\right] \tag{7}$$

Importing $\bar{\alpha}=(1+\varepsilon)\cdot\Gamma'(1)\cdot\Phi'(1)$, (6) can also be expressed as

$$P_0=1$$
$$P_l=\exp\left[-(1+\varepsilon)\Phi'(1-y_{l-1})\Gamma'\left(\Phi(1-y_{l-1})\right)\right] \tag{8}$$

Given Φ(x), (7) and (8) can be easily transformed into a linear optimization procedure for asymptotically good degree distributions Γ(x). In all linear programs below, 0=x1<x2<…<xm=1-δ are m equidistant points on [0, 1-δ], δ is the desired erasure ratio, and dmax is the maximum degree of the degree which is being optimized.

Fix the overhead ε and minimize the average degree of the distribution,

$$\text{LP 1: S.t. } \min \sum_{d=1}^{d_{\Gamma,\max}} d\Gamma_d$$

$$\sum_{d=1}^{d_{\Gamma,\max}}\Gamma_d\left(\Phi(x_i)\right)^{d-1}\geq\frac{-\ln(1-x_i)}{(1+\varepsilon)\Phi'(x_i)},\ i=1,2,\cdots,m \tag{9}$$

Fix the input average degree $\bar{\alpha}$ and minimize the overhead ε ($1+\varepsilon=\frac{\bar{\alpha}}{\Phi'(1)}\sum_{d=1}^{d_{\gamma,\max}}\frac{\gamma_d}{d}$),

$$\text{LP 2: S.t. } \min \frac{\bar{\alpha}}{\Phi'(1)}\sum_{d=1}^{d_{\gamma,\max}}\frac{\gamma_d}{d}$$

$$\sum_{d=1}^{d_{\gamma,\max}}\gamma_d\left(\Phi(x_i)\right)^{d-1}\geq\frac{-\ln(1-x_i)}{\bar{\alpha}\cdot\phi(x_i)},\ i=1,2,\cdots,m \tag{10}$$

[8] takes a Φ(x) = 0.05x + 0.5x2 + 0.4x 3+ 0.05x4, and obtain a Γ(x) = 0.7741x + 0.0025x 2+ 0.1490x3 + 0.026x4 + 0.0718x10. This couple of distributions can achieve an error floor down to 10-3 when the overhead ε is raised to 0.3.

Theoretically, while the number of sources N increases, the maximum degree Γmax been optimized abide by (9) or (10) would increase correspondingly. But inexplicably, when we raise N up to 100 or more, the maximum value of optimized degree distributions remain not larger than 10. This phenomenon constraints the development of the scores-sources networks. So we propose the Multi Distributed LT Codes (MDLT) to overcome this problem.

3. MDLT and BDLT

3.1 Multi Distributed LT Codes (MDLT)

From (7) and (8) we can realize that given the same overhead, larger input average degree can lead to lower error floor. But through the linear optimization described by (9) and (10), we cannot get an enough large Γmax for scores-sources networks. So the Multi Distributed LT Codes are raised. The system model is shown in Fig. 1(b). MDLT configures two-hop relays in the networks, RI and RII. If necessarily, more hops of relays can be configured as RI, RII, RIII, RVI… etc. Each hop-I relay receives coded bits from its corresponding set of sources, re-code them through its degree distribution

Γi(x), and transmitted re-coded bits to the hop-II relay. The hop-II relay receives re-coded bits from all the hop-I relays, re-re-code them through Θ(x) and transmit it to the destination. $\Theta(x)=\sum_{d=1}^{d_{\Theta,\max}}\Theta_d x^d$ is the degree distribution of the hop-II relay. To be simplified, we assume T1=T2=...=TN, Φij(x)= Φ(x) for i=1,2,...,TN and j=1,2,...,N, and Γi(x)= Γ(x). Then the overall degree distribution at the destination is $\Omega(x)=\Theta\left(\Gamma\left(\Phi(x)\right)\right)$, and the asymptotic performance of a MDLT code is given by:

$$P_0=1$$
$$P_l=\exp\left[-\bar{\alpha}\varphi(1-y_{l-1})\gamma\left(\Phi(1-y_{l-1})\right)\theta\left(\Gamma\left(\Phi(x)\right)\right)\right] \tag{11}$$

where $\bar{\alpha}=(1+\varepsilon)\cdot\Theta'(1)\cdot\Gamma'(1)\cdot\Phi'(1)$, and $\theta(x)=\Theta'(x)/\Theta'(1)$. Or in term of overhead ε,

$$P_0=1$$
$$P_l=\exp\left[-(1+\varepsilon)\Phi'(1-y_{l-1})\Gamma'\left(\Phi(1-y_{l-1})\right)\Theta'\left(\Gamma\left(\Phi(1-y_{l-1})\right)\right)\right] \tag{12}$$

Giving Φ(x) and Γ(x), Θ(x) can be optimized by the below linear procedures:
Fix the overhead ε and minimize the average degree of the distribution,

$$\text{LP 3: S.t. } \min \sum_{d=1}^{d_{\Gamma,\max}} d\Theta_d$$

$$\sum_{d=1}^{d_{\Theta,\max}}\Theta_d\left(\Gamma\left(\Phi(x_i)\right)\right)^{d-1}\geq\frac{-\ln(1-x_i)}{(1+\varepsilon)\Phi'(x_i)\Gamma'\left(\Phi(x_i)\right)},\ i=1,2,\cdots,m \tag{13}$$

Fix the input average degree $\bar{\alpha}$ and minimize the overhead ε,

$$\text{LP 4: S.t. } \min \frac{\bar{\alpha}}{\Phi'(1)\Gamma'(1)}\sum_{d=1}^{d_{\theta,\max}}\frac{\theta_d}{d}$$

$$\sum_{d=1}^{d_{\theta,\max}}\theta_d\left(\Gamma\left(\Phi(x_i)\right)\right)^{d}\geq\frac{-\ln(1-x_i)}{\bar{\alpha}\cdot\varphi(x_i)\cdot\gamma\left(\Phi(x_i)\right)},\ i=1,2,\cdots,m \tag{14}$$

Taking Φ(x) = 0.05x + 0.5x2 + 0.4x3 + 0.05x4, we firstly optimize Γ(x) by (9) and obtain Γ(x) = 0.7190x + 0.1004x2 + 0.0123x3 + 0.1338x7 + 0.0345x8. (We used a little different δ and ε compared to [8].) According to Φ(x) and Γ(x), we subsequently optimize Θ(x) and obtain Θ(x) = 0.9064x + 0.0245x2 + 0.0692x10. The results of And-Or Tree analysis of MDLT are shown in Fig. 2, compared with conventional DLT (CDLT) proposed in [8]. It can be easily seen that at an acceptable cost of ε, MDLT win a remarkable improvement in the error floor region compared to CDLT.

3.2 Backtracking Distributed LT Codes (BDLT)

Fig. 1(b) shows the system model of the two-hop-relay networks, but in some practical scenarios only one-hop relays are provided, then the system model in Fig. 1 should be configured. In order to depress the error floor of CDLT, we proposed the Backing Distributed LT codes. Taking the optimized results of Γ(x) and Θ(x) in section 3.1, let $\Gamma^*(x)=\Theta\left(\Gamma(x)\right)$, then the asymptotic performance of this code would be the same as (11) and (12), theoretically. Because we optimize Γ(x) and Θ(x) successively, and then backtrack them to obtain $\Gamma^*(x)=\Theta\left(\Gamma(x)\right)$, so these codes are names Backtracking Distributed LT Codes.

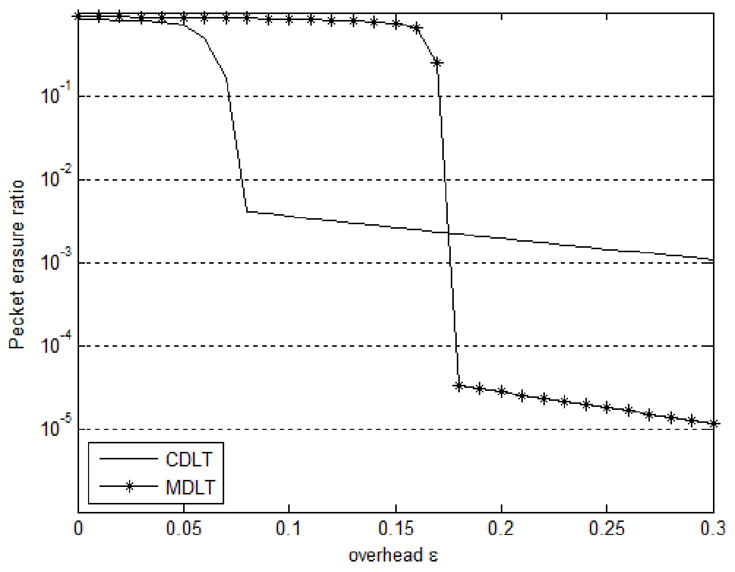


Fig. 2 Asymptotic performances of MDLT and CDLT

4. Numerical Results

In the simulation of MDLT, we take T=10 and N=10, i.e. 100 sources transmit information bits through 10 hop-I relays and one hop-II relay to a destination. To be compared, we also present the And-Or analysis and simulation of a CDLT code in which 100 sources transmit information through one relay to the destination. The results are shown in Fig. 3. The And-Or Tree analysis and simulating result of CDLT are almost the same as those were demonstrated in [8], but the simulating result of MDLT is even quite better than the corresponding And-Or Tree analysis. The reason is that the optimized degree distributions Γ(x) and Θ(x) can exploit almost all the input information bits, so when the overhead ε rises to a key value, the erasure ratios are depressed down to zero.

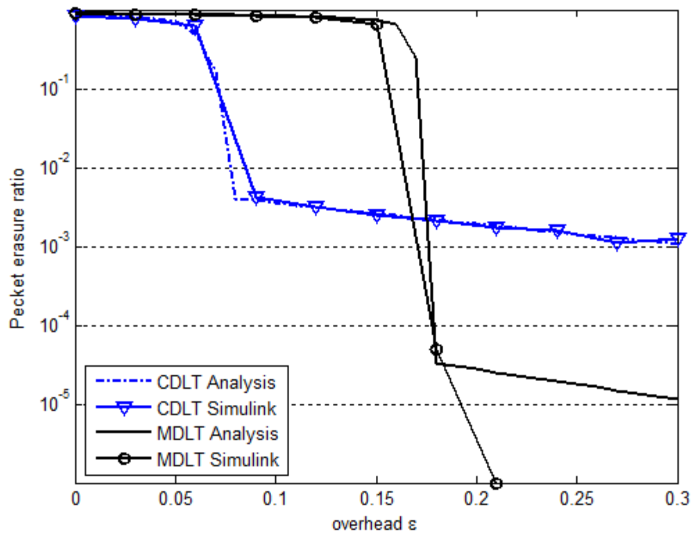


Fig. 3 Simulation Results of MDLT and CDLT at a 100-sources Network

Performances of BDLT are also been demonstrated and compared to CDLT in Fig. 4. Although sharing the same asymptotic performance with the MDLT, the simulating results appear an unsteady tendency when ε rises to 0.21. The reason is that, crudely nesting Γ(x) and Θ(x) to obtain $\Gamma^*(x)=\Theta(\Gamma(x))$ would produce a lot of nearly-zero values of Γ_d^*, such as $\Gamma_{76}^*=7.85\times10^{-12}$, $\Gamma_{77}^*=1.15\times10^{-12}$, and $\Gamma_{78}^*=1.12\times10^{-13}$. It can be easily seen that if the relay try to select 76 sources with the probability 7.85×10-12 for 100,000 (k·N) times, it cannot always succeed. In spite of the instability, BDLT achieve obviously improvement compared to CDLT when the number of sources rises to 100.

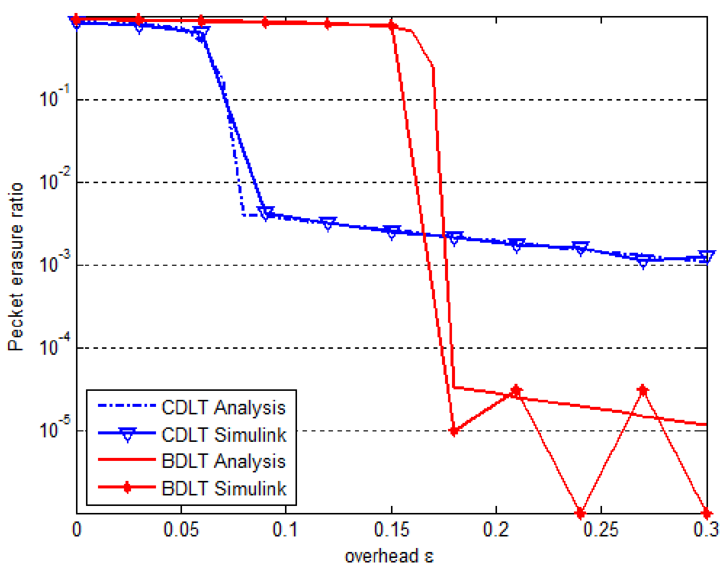


Fig. 4 Simulation Results of MDLT and BDLT at a 100-sources Network

5. Conclusion

From LTs to DLTs, the average degrees of input nodes pay a key part in affecting the error floors. We propose two modified DLT codes, MDLT and BDLT, which out-performance the conventional DLT codes in the scores-sources-networks. Although the analysis and simulations of these two codes are carried out over lossless channels, they can be easily extended to lossy channels. And following with the rising of the number of sources, the hops of relays can be easily added, too.

References

[1]. J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A Digital Fountain Approach to Reliable Distribution of Bulk Data," Proc. of the ACM SIGCOMM 98, pp. 56-67, Vancouver, Canada, Sept. 1998.

[2]. M. Luby, "LT Codes," in Proc. IEEE Symp. Found. Comp. Sci., Vancouver, BC, Canada, Nov.2002, pp.271-280.

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

[4]. R. Ahlswede, N. Cai, S Y R Li, and R W Yeung, "Network Information Flow," IEEE Transactions on Information Theory, 2000,46(4), pp.1204-1216 .

[5]. S Y R Li, R W Yeung, and N. Cai, "Linear Network Coding", IEEE Transaction on Information Theory, 2003, 49(2): 371-381 .

[6]. S. Puducheri, J. Kliewer and T. Fuja, "The Design and Performance of Distributed LT Codes", in IEEE Trans. Info. Theory, vol.53, no.10, pp.3740-3754, Oct.2007.

[7]. M. Luby, M. Mitzenmacherand and A. Shokrollahi, "Analysis of Random Processes via And-Or Tree Evaluation," Proc.of the 9th Annual SIAM Symp. On Discrete Algorithms (SODA), pp.364-373, SanFrancisco, USA, Jan.1998.

[8]. D. Sejdinovic, R. Piechocki, and A. Doufexi, "And-Or tree analysis of distributed LT codes," in Proc. IEEE Inf. Theory Workshop, Taormina, Italy, Oct.2009, pp.261-265

[9]. N. Rahnavard, B. Vellambi, and F. Fekri, "Rateless codes with unequal error protection property," IEEE Trans. Inf. Theory, vol. 53, no. 4, pp. 1521–1532, Apr. 2007.

[10]. A. Liau, S. Yousefi, and I.-M.Kim, "Binary soliton-like rateless coding for the Y-network," IEEE Trans. Commun., vol.59, no.12, pp. 3217-3222, Dec. 2011.

[11]. A. Liau, I. Kim, and S. Yousefi, "Improved low-complexity soliton-like network coding for a resource-limited relay," IEEE Trans. Commun., vol.61, no.8, pp.3327-3335, Aug. 2013.

[12]. I. Hussain, M. Xiao, and L. K. Rasmussen, “Buffer-based distributed LT codes,” IEEE Trans. Commun., vol. 62, no. 11, pp. 3725-3739, Nov. 2014.
[13]. I. Hussain, M. Xiao, and L. K. Rasmussen, “Erasure Floor Analysis of Distributed LT Codes”, IEEE Transactions on Communications, Vol. 63, No. 8, pp. 2788-2796, August 2015.
[14]. Haitao Yang, Ming Jiang, Hong Shen, and Chunming Zhao, “A Distributed LT Code Design for Multiple-Access Relay Networks Subject to Erasures”, IEEE Communications Letters, Vol. 19, No. 4, pp. 509-512, April 2015.