LETTER

Parity-Check Matrix Extension to Lower the Error Floors of Irregular LDPC Codes

Jianjun MU[†a)], **Xiaopeng JIAO**[†], **Jianguang LIU**[†], ***Nonmembers***, ***and*** **Rong SUN**[††], ***Member***

SUMMARY Trapping sets have been identified as one of the main factors causing error floors of low-density parity-check (LDPC) codes at high SNR values. By adding several new rows to the original parity-check matrix, a novel method is proposed to eliminate small trapping sets in the LDPC code's Tanner graph. Based on this parity-check matrix extension, we design new codes with low error floors from the original irregular LDPC codes. Simulation results show that the proposed method can lower the error floors of irregular LDPC codes significantly at high SNR values over AWGN channels.

key words: *error floor, low-density parity-check (LDPC) codes, trapping sets*

1. Introduction

Although low-density parity-check (LDPC) codes have been intensively researched due to their near Shannon limit performance and low decoding complexity, the error floor phenomenon [1] is still one of the significant impediments to the use of LDPC codes in optical communication and data storage systems. Therefore, lowering error floors of LDPC codes and predicting the iterative decoding performance in the error-floor regions are active research areas [2], [3]. Trapping sets are considered the primary factor for degrading the performance of low-density parity-check (LDPC) codes in the error-floor region [2]. Searching small trapping sets in a code's Tanner graph is proved to be an NP-hard problem in general [4]. Since trapping sets are largely related to small cycles in a code's Tanner graph, the authors in [5] proposed an efficient method to find small trapping sets in regular LDPC codes by cycle enumeration. However, this method is not applicable to irregular codes. In [6], the authors use the ACE metric to distinguish different cycles and then apply this method to search for small trapping sets in irregular LDPC codes. Generally, irregular LDPC codes have higher error floors than regular LDPC codes.

Changing the graph representation of an existing code, e.g. by adding new parity-check nodes that eliminate small stopping sets [7], [8], has been suggested for reducing the error floors of LDPC codes under iterative decoding. In this letter, we propose a novel method to eliminate small trapping sets in irregular LDPC codes. Firstly, we use the cycle enumeration method plus ACE metric to identify the small trapping sets in irregular LDPC codes. Secondly, by adding several new check nodes and having their edges connected to variable nodes of small trapping sets in the original code, a novel method is proposed to eliminate small trapping sets in the LDPC code's Tanner graph. Finally, based on this parity-check matrix extension, we design new codes with low error floors under iterative decoding from the original irregular LDPC codes.

The remainder of this letter is organized as follows. In Sect. 2, the identification method of Trapping sets are reviewed. Section 3 proposes a novel method to eliminate small trapping sets in the LDPC code's Tanner graph and design new codes with low error floors from the original irregular LDPC codes. In Sect. 4, simulation results are provided to show the efficiency of the proposed method. Finally, Sect. 5 concludes the letter.

2. Identification of Trapping Sets

Generally, an (a, b) trapping set is a set of a variable nodes, for which the induced subgraph of these a variable nodes and their neighborhood check nodes contain exactly b odd-degree check nodes [1]. If $b = 0$, the trapping set also represents a codeword. The detrimental trapping sets, which largely contribute to the iterative decoding in the error floor region, have small values of both a and b. Thus, eliminating small trapping sets can significantly improve the performance of LDPC codes under iterative decoding in the error floor region. For a given LDPC code that needs to be improved, we must first find as many small trapping sets as possible. Here we use the method proposed in [6] to search detrimental trapping sets in irregular LDPC codes. The search procedure can be summarized as follows.

(1) Find all small cycles with size less than λ and ACE values less than χ through the breadth-first search in a code's Tanner graph.

(2) Apply some unnatural noises, called error impulses with amplitude ε, to all the bit positions in cycles found in step (1), and scale other bits in the codeword by a relatively small parameter α. The noised codeword is then transmitted over an AWGN channel and decoded by the iterative decoder. Assume all-zero codeword is transmitted and BPSK modulation is used, then the noised codeword can be represented by

Manuscript received October 26, 2010.
Manuscript revised January 27, 2011.

[†]The authors are with the School of Computer Science and Technology, Xidian University, Xi'an, 710071, P.R. China. They are also with the State Key Laboratory of Integrated Service Networks, Xidian University.

[††]The authors is with the State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, 710071, P.R. China.

a) E-mail: jjmu@xidian.edu.cn

DOI: 10.1587/transcom.E94.B.1725

$$x_{noise} = \{-1+\varepsilon, -1+\varepsilon, \cdots, -1+\varepsilon, -\alpha, -\alpha, \cdots, -\alpha\}.$$

Here, we assume that the first few bits of the codeword are involved in the cycle.

(3) Run the iterative decoder with the deterministic input x_{noise}. When the decoder fails to find a valid codeword with the predefined iteration number, we do some further iterations (say 20) and identify the trapping set as the union of all bits that do not be decoded correctly during those 20 iterations. If the trapping set found here does not exist in our list, add it to the list.

3. Elimination of Trapping Sets

When small trapping sets of LDPC codes are identified, we try to eliminate them by adding several new check nodes to the code's Tanner graph. The method we used here resembles our previous work to eliminate small stopping sets [7]. Generally, for a given (a, b) trapping set T, add a new check node and have its edge connected to one of the variable nodes in T, then T is transformed to an $(a, b+1)$ trapping set. For example, Fig. 1(a) shows that an $(a, 1)$ trapping set is transformed to an $(a, 2)$ trapping set by adding a new check node c_a and having its edge connected to v_a. The degree of the newly added check node should be close to the maximum check-node degree in the original Tanner graph of the given LDPC code in order to reduce the number of the new check nodes, which can also make the degree distribution of the new code keep close to that of the original one. Moreover, any two edges of c_a cannot be added to the same trapping set since this leads to the trapping set unremoved. Note that adding new check nodes may cause code rate loss. To reduce the code rate loss, the number of the newly added check nodes should be kept as small as possible. We observe from simulations that many trapping sets have common variable nodes. Thus one can eliminate several small trapping sets by adding only one edge. Figure 1(b) illustrates that three trapping sets with a common variable node v can be eliminated simultaneously by adding only one edge between c_a and v. Based on the above description, the trapping set eliminating procedure can be summarized as follows.

(1) Select a list L of trapping sets to be removed according to the trapping set identification procedure.

(2) Pick a variable node v with the highest frequency from L and connect it to the newly added check node c_a.

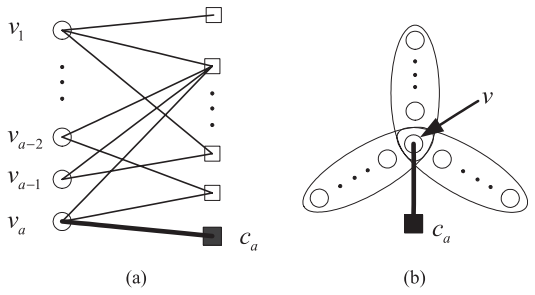

Fig. 1 Eliminating small trapping sets by adding a new check node c_a.

(3) Select a variable node v' with the highest frequency from the unremoved trapping sets obtained by step (2) such that v' does not appear in the trapping sets which have been removed by step (2). Connect v' to c_a, if the connected edges exceed the predefined check-node degree, then add a new check node.

(4) Repeat step 3 until all dominant trapping sets are removed.

4. Simulation Results

Two LDPC codes C_1 and C_2 constructed by PEG algorithm [9] are used to simulate our method. The variable node degree distributions we used are as follows:

$$C_1\text{: } \lambda_1(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^{10}$$

and

$$C_2 : \lambda_2(x) = 0.23802x + 0.20997x^2 + 0.03492x^3 + 0.12015x^4 + 0.01587x^6 + 0.00480x^{13} + 0.37627x^{14}.$$

C1 is a code with a block length of 100 and rate of 0.5. Using the trapping set identification procedure (where $\lambda = 8$, $\chi = 3$, $\varepsilon = 1.7$, and $\alpha = 0.5$), we find and select the following three kinds of trapping sets to be removed: 17 trapping sets with $b = 0$ and $a < 10$ (also codewords), 21 trapping sets with $b = 1$ and $a < 9$, 23 trapping sets with $b = 2$ and $a = 4$. According to the trapping set elimination procedure, C_3 is obtained by adding two check nodes of degree 8 to C_1. All these 61 trapping sets are removed and after the low-weight codewords are transformed to $(a, 1)$ trapping sets, then we continually to transform some of them to $(a, 2)$ trapping sets. The rate of C_3 is 0.48 and its loss is only 0.02 when compared to C_1. Figure 2 shows the FER performance of C_1 and C_3 under iterative BP decoding over AWGN channel. BPSK modulation is used and the maximum number of iterations is set to 50. It can be seen that C_3 has significant

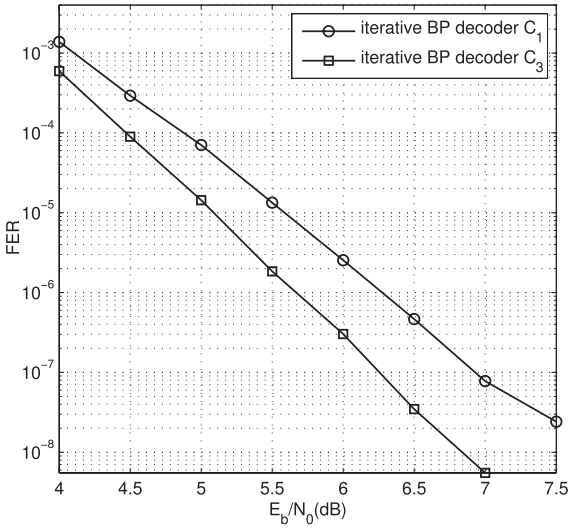

Fig. 2 FER performance of C_1 and C_3 under BP decoder over AWGN channel.

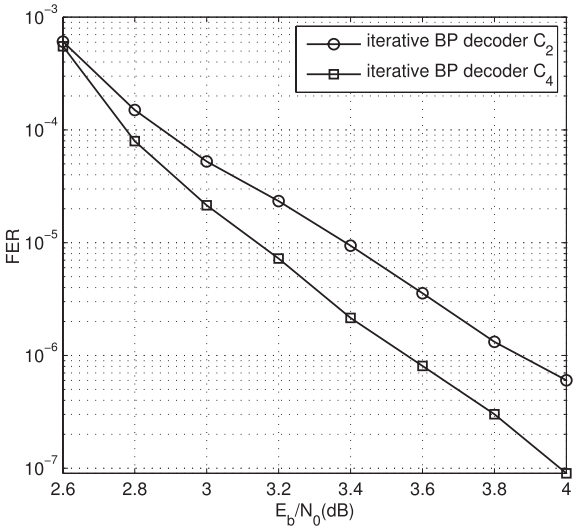

Fig. 3 FER performance of C_2 and C_4 under BP decoder over AWGN channel.

performance improvement over C_1 in the error floor region. For example, C_3 outperforms C_1 by about 0.8 dB at the FER of 2×10^{-8}. C_2 is a code with a block length of 500 and rate of 0.5. C_4 is obtained by adding two new check nodes to C_2. Similar performance improvement is observed through Fig. 3.

5. Conclusions

Based on parity-check matrix extension for eliminating small trapping sets in the Tanner graphs of irregular LDPC codes, we propose a new method to design new codes with low error floors under iterative decoding from the original irregular LDPC codes. The advantages of this method are a) it is universal as it can be applied to most codes and all decoding algorithms and b) it improves performance significantly at the expense of the less code rate loss only. Simulations show that our proposed method is efficient and the error floors of irregular LDPC codes under iterative decoding are lowered significantly.

Acknowledgments

This research is supported by the National Nature Science Foundation of China (Grant Nos. 61001132 and 61001131), the open research fund of State Key Laboratory of Integrated Service Networks, Xidian University (No. ISN10-04), and the National Basic Research Program (973) of China (Grant No. 2010CB328300).

References

[1] T.J. Richardson, "Error floors of LDPC codes," Proc. 41st Annu. Allerton Conf. Commun., Contr. Comput., 2003.
[2] Y. Han and W.E. Ryan, "Low-floor decoders for LDPC codes," IEEE Trans. Commun., vol.57, no.6, pp.1663–1673, June 2009.
[3] L. Dolecek, P. Lee, Z. Zhang, V. Anantharam, B. Nikolic, and W. Wainwright, "Predicting error floors of structured LDPC codes: Deterministic bounds and estimates," IEEE J. Sel. Areas Commun., vol.27, no.6, pp.908–917, Aug. 2009.
[4] C.C. Wang, S.R. Kulkarni, and H.V. Poor, "Finding all small error-prone substructures in LDPC codes," IEEE Trans. Inf. Theory, vol.55, no.5, pp.1976–1999, May 2009.
[5] E. Cavus, C.L. Haymes, and B. Daneshrad, "Low BER performance estimation of LDPC codes via application of importance sampling to trapping sets," IEEE Trans. Commun., vol.57, no.7, pp.1886–1888, July 2009.
[6] X. Zheng, F.C.M. Lau, C.K. Tse, Y. He, and M.Z. Wang, "Evaluation of the extremely low block error rate of irregular LDPC codes," Proc. IEEE Int. Conf. Commun., pp.1939–1943, Dresden, Germany, 2009.
[7] X. Jiao, J. Mu, J. Song, and L. Zhou, "Eliminating small stopping sets in irregular low-density parity-check codes," IEEE Commun. Lett., vol.13, no.6, pp.435–437, June 2009.
[8] T. Wadayama, "Average stopping set weight distribution of redundant random ensembles," IEEE Trans. Inf. Theory, vol.54, no.11, pp.4991–5004, Nov. 2008.
[9] X. Hu, E. Eleftheriou, and D.M. Arnold, "Regular and irregular progressive edge-growth Tanner graphs," IEEE Trans. Inf. Theory, vol.51, no.1, pp.386–398, Jan. 2005.