LETTER

Multi-Stage Decoding Scheme with Post-Processing for LDPC Codes to Lower the Error Floors[**]

Beomkyu SHIN[†*], **Hosung PARK**[†a)], ***Nonmembers***, **Jong-Seon NO**[†], ***Member***, ***and*** **Habong CHUNG**[††], ***Nonmember***

SUMMARY In this letter, we propose a multi-stage decoding scheme with post-processing for low-density parity-check (LDPC) codes, which remedies the rapid performance degradation in the high signal-to-noise ratio (SNR) range known as error floor. In the proposed scheme, the unsuccessfully decoded words of the previous decoding stage are re-decoded by manipulating the received log-likelihood ratios (LLRs) of the properly selected variable nodes. Two effective criteria for selecting the probably erroneous variable nodes are also presented. Numerical results show that the proposed scheme can correct most of the unsuccessfully decoded words of the first stage having oscillatory behavior, which are regarded as a main cause of the error floor.

key words: *error floor, log-likelihood ratio (LLR), low-density parity-check (LDPC) codes, sum-product algorithm*

1. Introduction

Low-density parity-check (LDPC) codes can be considered as one of the strong candidates for the systems requiring capacity-approaching error-correcting performance such as wireless communication systems without feedback channel and data storage. However, in the high signal-to-noise ratio (SNR) region, the performance curve of LDPC codes generally shows error floor which hinders achieving very low error rate.

Many researchers have made efforts to cope with this problem: some proposed specific construction methods of LDPC codes [1], [2], which unfortunately cannot be used for the existing codes, and some tried to modify the decoding algorithms of LDPC codes based on the assumptions that the existing dominant trapping sets [3] could be identified before decoding [4], [5].

In the error floor region of the LDPC codes, error events in the unsuccessfully decoded words can be categorized into three different types [5]:

1. Unstable type: positions of bit errors change randomly at each iteration of decoding;
2. Stable type: positions of bit errors hardly change;
3. Oscillating type: positions of bit errors change periodically.

In most cases, error events of oscillating type are dominant in the error floor region, which is known to be caused by finite precision decoder implementation [6] in conjunction with the trapping sets.

Periodicity of oscillating error patterns in the unsuccessfully decoded words can give rise to useful information for low error rate. Bit error rate can considerably be lowered with few extra computation by using the fact that the numbers of bit errors and check nodes at each iteration are highly correlated [7]. In [8], belief propagation decoding followed by ordered statistic decoding is proposed based on the number of oscillations of log-likelihood ratio (LLR) messages in each variable node, and modified belief propagation decoding is also proposed.

In this letter, two measures based on the oscillation of LLR are used to select the most unreliable variable nodes. And we propose a multi-stage decoding scheme to re-decode, with the same decoder, the unsuccessfully decoded words of the previous decoding stage by manipulating the received LLRs of the selected variable nodes.

2. Multi-Stage Decoding Scheme With Post-Processing

The proposed multi-stage decoding scheme reads as follows:

1. Firstly, decode the received data;
2. If decoding is successful or the maximum number of decoding stages is accomplished, then go to end;
3. Otherwise, do post-processing of the received data;
4. Do additional decoding and go to 2.

Post-processing can be defined as a subsequent process applied to the received data or its modified version when decoding failure is declared in the previous decoding stage. Naturally, these post-processed data are used as the input of the next decoding stage in the proposed multi-stage decoding scheme. Post-processing is very important especially for data storage, because once stored data are corrupted by leakage of electrons in memory or bad sectors in disk drives, additional retrieval cannot make any performance improvement. In the multi-stage decoding schemes with post-processing, one may generally consider two important matters: how to modify the decoding algorithm and

Manuscript received December 9, 2010.
Manuscript revised March 4, 2011.

[†]The authors are with the Department of Electrical Engineering and Computer Science, INMC, Seoul National University, Seoul 151-744, Korea.
[††]The author is with the School of Electronics and Electrical Engineering, Hongik University, Seoul 121-791, Korea.
[*]Presently, with the Samsung Electric, Co. Ltd., Hwasung, Gyeonggi-do 445-701, Korea.

[**]This work was supported by the NRF grant funded by the MEST of Korea (No.2011-0000328) and also supported by the KCC under the R&D program supervised by the KCA of Korea (KCA-2011-08913-04003).

a) E-mail: lovepk98@snu.ac.kr

DOI: 10.1587/transcom.E94.B.2375

how to manipulate the received data from the channel. In this letter, we will focus on the latter case and use the same decoding algorithm at each decoding stage because it has more benefits in implementation.

When the iterative decoder fails with the received data, the adequately modified received data can lead the same decoder to successful decoding in the next decoding stage. In post-processing, we should carefully select the set of variable nodes whose received values seem to be erroneous. However, it is not necessary that all the selected nodes are from trapping sets because the variable nodes outside trapping sets can also contribute to a decoding failure. Thus, our selection schemes, which appear in the next section, can work well with no identification of trapping sets of a given code.

When the previous decoding fails, we can manipulate the received LLR values of the target variable nodes in three different ways for the post-processing:

- *Erasing*: Set the initial LLRs to 0's;
- *Inverting*: Flip the signs of the initial LLRs;
- *Scaling*: Multiply the magnitudes of the initial LLRs by some scaling factors.

In some cases, inverting and scaling methods show good performances but we will consider only erasing method in this letter since those operations are more harmful than erasing method when the wrong nodes are selected as target variable nodes.

3. New Criteria for Selecting Target Variable Nodes

In this section, two measures of each variable node for being selected as a target variable node are defined. Let $I_{\max}$ be the maximum number of iterations and $L^i(j)$ be the LLR of the j-th node at the i-th iteration, where $L^0(j)$ is the initial received LLR of the j-th node.

- Number of *sign changes* of the j-th node, $N_{\mathrm{SC}}(j)$, is counted as;
  $N_{\mathrm{SC}}(j)$ is increased by 1 if $\mathbf{sign}(L^i(j)) \neq \mathbf{sign}(L^{i-1}(j))$, $1 \leq i \leq I_{\max}$.
- Number of *sign differences* of the j-th node, $N_{\mathrm{SD}}(j)$, is counted as;
  $N_{\mathrm{SD}}(j)$ is increased by 1 if $\mathbf{sign}(L^i(j)) \neq \mathbf{sign}(L^0(j))$ and $L^0(j) \neq 0$, $1 \leq i \leq I_{\max}$.

In the error floor region (in other words, relatively high SNR region), most of the received data from the channel are correct and just a few nodes are erroneous. Hence most of the transmitted codewords are correctly decoded. Most of the incorrectly decoded words have oscillating error patterns in the iterative decoding process. Interestingly in this case, it is revealed by the observation on the LLR messages of the sum-product algorithm that the sign of LLR of initially erroneous nodes tends to be changed more frequently than error-free nodes. Since most of the messages passed between variable and check nodes are correct in the error floor region, an initially erroneous node tends to have a different sign of received LLR from the messages sent by neighboring nodes with high probability. On the contrary, an initially error-free node has the same sign of received LLR as the messages sent by neighboring nodes with high probability. Hence, we can say that the larger number of LLR sign changes (differences) a variable node has, the higher probability the variable node can be initially erroneous with. These observations are illustrated in Fig. 1, where 1/2-rate Margulis code is simulated with $I_{\max} = 200$ at $E_b/N_o = 2.5$ dB. Distributions of LLR sign changes (differences) for the initially erroneous variable nodes and the error-free variable nodes are plotted separately. Now we propose the following two criteria:

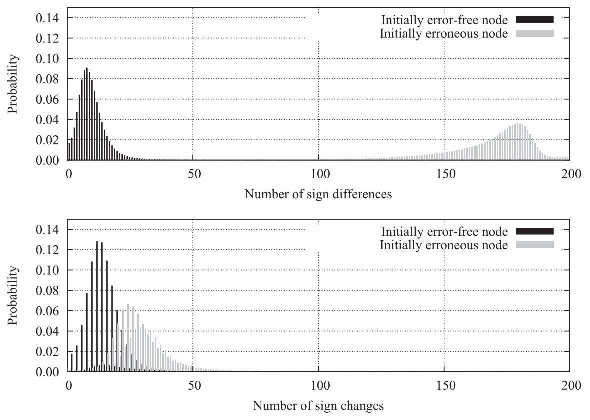

Fig. 1 Probabilities of variable nodes having the specific number of sign changes (differences) given that the variable nodes are initially error-free or erroneous with $I_{\max} = 200$.

1. **Scheme-SC**: Rank the variable nodes according to the number of LLR sign changes in descending order and then select the topmost N nodes as the target nodes in each iterative decoding process.
2. **Scheme-SD**: Rank the variable nodes according to the number of LLR sign differences in descending order and then select the topmost N nodes as the target nodes in each iterative decoding process.

Here N is a pre-determined positive integer. While the **Scheme-SC** is designed to correct the error events of oscillating type, the **Scheme-SD** is designed to correct the error events of oscillating and stable types. Note that it does not need a lot of computations to rank the variable nodes because linear-time sorting algorithms such as counting sort can be used.

4. Simulation Results

The simulation results for the proposed schemes with only one additional decoding stage are presented in this section. Simulations are carried out on three codes: 1/2-rate Margulis code with $n = 2640$ [9] and 1/2- and 3/4-rate LDPC codes with $n = 2304$ in IEEE 802.16e standard [10]. Sum-product decoding algorithm with double precision is considered for all the decoding stages and the numbers of max-

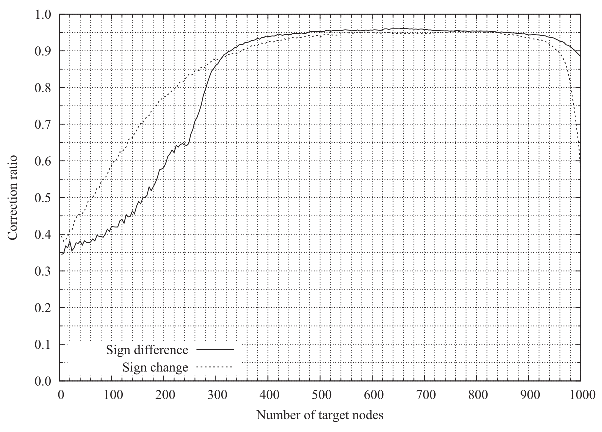

Fig. 2 Correction ratios for the proposed schemes at 2.5 dB for the Margulis code.

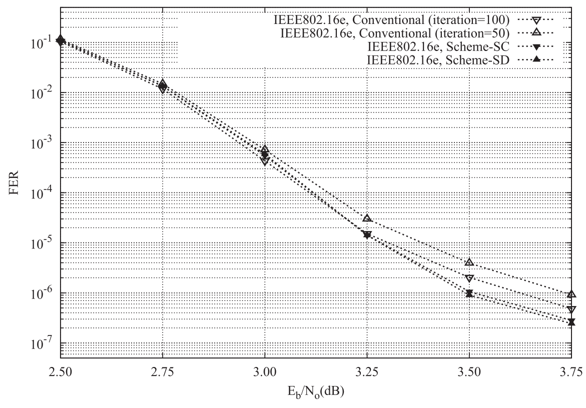

Fig. 4 Performance of the proposed schemes with 3/4-rate 802.16e LDPC code.

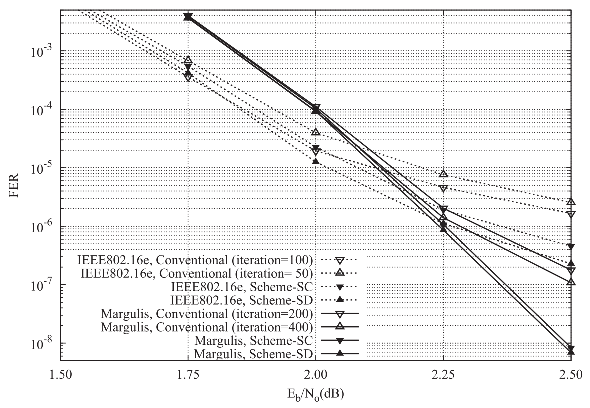

Fig. 3 Performance of the proposed schemes with 1/2-rate 802.16e LDPC code and 1/2-rate Margulis code.

imum iterations for each decoding stage are set to 200 for the Margulis code and 50 for the 802.16e LDPC codes, respectively.

Correction ratio (*CR*) is defined as the ratio of the number of corrected frames after the next stage decoding and the number of error frames after the previous stage decoding. The *CR*s of **Scheme-SC** and **Scheme-SD** for the Margulis code are shown in Fig. 2. As the number of erased variable nodes N increases, *CR*s increase for small N and rapidly decrease for large N. For the wide range of N, *CR*s are almost constant, which implies the robustness of the proposed criteria. FER curves for three codes with the proposed schemes are shown in Fig. 3 and Fig. 4. For comparison, the performances with the same iterations without post-processing are also plotted. We can see that the error floor performances are far improved with the aid of the proposed post-processing schemes.

5. Conclusions

In this letter, a multiple decoding scheme with post-processing for LDPC codes is proposed to improve the performance in the error floor region. Post-processing is implemented by erasing the received LLRs of the selected variable nodes that are highly suspected to be erroneous in the previous decoding stage. Two selection criteria for the post-processing are proposed based on the sign changes and sign differences of LLR messages. With these proposed schemes, most of the oscillatory errors, regarded as main cause of the error floor, are corrected. As a result, we can significantly lower the error floor for the Margulis code and the LDPC codes in IEEE802.16e.

References

[1] S.J. Johnson and S.R. Weller, "Constraining LDPC degree distributions for improved error floor performance," IEEE Commun. Lett., vol.10, no.2, pp.103–105, Feb. 2006.

[2] Z. He, P. Fortier, and S. Roy, "A class of irregular LDPC codes with low error floor and low encoding complexity," IEEE Commun. Lett., vol.10, no.5, pp.372–374, May 2006.

[3] T. Richardson, "Error floors of LDPC codes," Proc. Allerton Conference on Communications, Control, and Computing, pp.1426–1435, Monticello, Oct. 2003.

[4] Y. Han and W.E. Ryan, "LDPC decoder strategies for achieving low error floors," Proc. Information Theory and Applications Workshop, pp.277–286, Jan. 2008.

[5] S. Lander and O. Milenkovic, "Algorithmic and combinatorial analysis of trapping sets in structured LDPC codes," Proc. Int. Conf. on Wireless Networks, Comm. and Mobile Computing, pp.630–635, June 2005.

[6] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. Wainwright, "Investigation of error floors of structured low-density parity-check codes by hardware emulation," Proc. IEEE GLOBECOM, Nov. 2006.

[7] E. Alghonaim, M. Landolsi, and A. El-Maleh, "Improving BER performance of LDPC codes based on the intermediate decoding results," Proc. IEEE ICSPC, pp.1547–1550, Nov. 2007.

[8] S. Gounai and T. Ohtsuki, "Decoding algorithms based on oscillation for low-density parity check codes," IEICE Trans. Fundamentals, vol.E88-A, no.8, pp.2216–2226, Aug. 2005.

[9] D. MacKay and M. Postol, "Weaknesses of Margulis and Ramanujan-Margulis low-density parity-check codes," Electronic Notes in Theoretical Computer Science, vol.74, pp.97–104, 2003.

[10] IEEE Std. 802.16., "IEEE Standard for local and metropolitan area network part 16: Air interface for fixed and mobile broadband access systems," 2004.