LDPC 코드의 빠른 복원을 위한 1단으로 구성된 적응적인 오프셋 MS 알고리즘

논 문
59P-1-9

Single-Step Adaptive Offset Min-Sum Algorithm for Decoding LDPC Codes

林 小 菊* · 康 秀 蘭* · 이 해 기** · 김 성 수†
(Xiaoju Lin · Gansuren Baasantseren · Hae-Kee Lee · Sung-Soo Kim)

Abstract - Low-density parity-check (LDPC) codes with belief-propagation (BP) algorithm achieve a remarkable performance close to the Shannon limit at reasonable decoding complexity. Conventionally, each iteration in decoding process contains two steps, the horizontal step and the vertical step. In this paper, an efficient implementation of the adaptive offset min-sum (AOMS) algorithm for decoding LDPC codes using the single-step method is proposed. Furthermore, the performances of the AOMS algorithm compared with belief-propagation (BP) algorithm are investigated. The algorithms using the single-step method reduce the implementation complexity, speed up the decoding process and have better efficiency in terms of memory requirements.

Key Words : LDPC Codes, Single-Step, BP Algorithm, MS Algorithm, NMS Algorithm, OMS Algorithm, AOMS Algorithm

1. Introduction

Low-density parity-check (LDPC) codes belong to the family of linear block codes with sparse parity-check matrix H, that is, a small value of row and column weight compared to the code block length. They were first discovered by Gallager [1] in 1962 and rediscovered by Mackay and Neal [2], [3] in 1996. Moreover, an LDPC code can be represented by a parity-check matrix H, which is often described as a Tanner graph [4]. The degree d of a node on the graph is referred to the d edges connected to it. The d_c bit nodes connected to the m^{th} check node in the graph correspond to the d_c 1's on the m^{th} row of H. The object of decoding is to search for the most likely codeword c which satisfies $Hc^T = 0$.

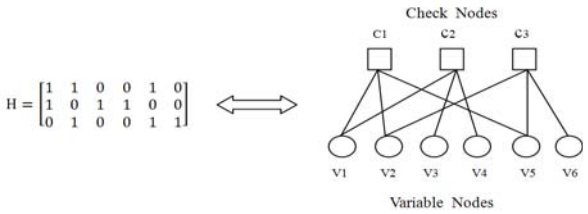

Fig. 1 The parity check matrix and its corresponding Tanner Graph.

* 정 회 원 : 충북대학교 전기공학부 석사과정
** 정 회 원 : 충청대학 전기공학부 교수 · 공박
† 교신저자, 정회원 : 충북대학교 전기공학부 교수 · 공박
E-mail : sungkim@chungbuk.ac.kr
접수일자 : 2010년 1월 19일
최종완료 : 2010년 2월 5일

Figure 1 is an example of the parity check matrix H of (6, 3) LDPC code and its corresponding Tanner graph.

Furthermore, LDPC codes deliver very good performance for long code lengths when decoded with the belief-propagation (BP) algorithm [2]. As LDPC codes are considered for the use in a wide range of applications, for example in the second Satellite Digital Video Broadcasting normalization (DVB-S2), the search for efficient implementation of decoding algorithms has been pursued intensively. The implementation of the BP algorithm in log-likelihood is difficult because a lot of logarithmic and multiplicative operations are involved in the check-node update. Therefore the BP algorithm can be simplified to min-sum (MS) or BP- based algorithm [5], which greatly reduces the implementation complexity but degrades decoding performance. This has led to the development of different variants of MS algorithm which are the normalized min-sum (NMS) algorithm [6], the offset min-sum (OMS) algorithm [7], and the adaptive offset min-sum (AOMS) algorithm [8].

Exactly, many works have been devoted to reduce the complexity of check node update process. However, the update process of the variable node also costs excessive storage and computation resources. Therefore, the purpose of our work is to find the efficient implementation method using the single-step for decoding LDPC codes, which increases the decoding speed and reduces the requirement on memory usage in our low complexity software implementation (in C language).

The remainder of the paper is organized as follows. Section II briefly reviews the standard BP algorithm while the single-step method is explored in section III.

Section IV investigates the MS algorithm and its different variants using the single-step method. Some simulation results of the performance are shown in Section V. Finally, the conclusion of this work is given in Section VI.

2. Standard BP Algorithm

LDPC codes are generally decoded in iterative fashion where the soft messages are updated at variable nodes and check nodes. These messages are iteratively propagated along the Tanner graph. Despite sacrificing optimality, iterative BP algorithm offers a good performance-complexity tradeoff for decoding LDPC codes, especially at large block lengths.

In the following, we assume BPSK modulation which maps a codeword $c=(c_1, c_2, ..., c_N)$, with c_n = 0 or 1, into a transmitted sequence $x=(x_1, x_2, ..., x_N)$, according to $x_n=2c_n-1$. Then x is transmitted over an additive white Gaussian noise (AWGN). The received value corresponding to x_n after the demodulator is $y_n=x_n+v_n$, where v_n is a random variable with zero mean and variance $\sigma^2=N_0/2$. We assume all the messages passing between variable and check nodes are in the form of the log-likelihood ratios (LLR's). So the initial LLR's of bit n is defined by

$$Z_n^{(0)}=\ln\frac{p(x_n=-1/y_n)}{p(x_n=1/y_n)}=2y_n/\sigma^2$$

Let $N(m)=\{v_n|H_{nm}=1\}$ be the set of variable nodes $v_n$that participates in check node c_m, ie, the positions of 1's in the row of the parity check matrix H and N(m)\n represents the set N(m), with excluded variable node v_n. Similarly, $M(n)=\{c_m|H_{nm}=1\}$ denotes the set of check nodes c_m in which the variable nodes v_n participate and M(n)\m represents the set M(n), with excluded check node c_m.

For the iteration, let $Z_{mn}^{(k)}$ and $L_{mn}^{(k)}$ denote the message sent from variable node to check node and the message sent from check node to variable node, respectively [2]. Then each iteration of the standard BP algorithm in LLR domain is given as follows:

- ***Initialization:***

For each n and each $m\in M(n)$,

$$Z_{mn}^{(0)}=Z_n^{(0)}$$

- k^{th} ***Iteration:***

1) Horizontal Step (update check node):

For each m and each $n\in N(m)$,

$$L_{mn}^{(k)}=2\tanh^{-1}\left(\prod_{n'\in N(m)\setminus n}\tanh\left(\frac{Z_{mn'}^{(k-1)}}{2}\right)\right) \quad (1)$$

2) Vertical Step (update variable node):

For each n and each $m\in M(n)$,

$$Z_{mn}^{(k)}=Z_n^{(0)}+\sum_{m'\in M(n)\setminus m}L_{m'n}^{(k)} \quad (2)$$

3) The k^{th} output of variable node:

$$Z_n^{(k)}=Z_n^{(0)}+\sum_{m\in M(n)}L_{mn}^{(k)} \quad (3)$$

4) Hard decision:

For $n\in[1,N]$,detect the transmitted codeword such that:

$$\hat{x}_n^{(k)}=\begin{cases}0, & \text{if } Z_n^{(k)}>0\\ 1, & otherwise\end{cases}$$

If $H(\hat{x}_n^{(k)})^T=0$ or the number of iterations exceeds a given maximum number of iterations, the process stops the iteration and yields the output $\hat{x}_n^{(k)}$ as the decoded codeword; otherwise the decoding repeats from step 1).

3. The Single-step Method

Conventionally, each iteration of standard BP decoding contains two steps. Each check node receives messages from all variable nodes connected to it, called the horizontal step. Then it sends messages back to these variable nodes, called the vertical step.

It can be simplified the processing in either variable nodes or check nodes, or both, to obtain different versions of the BP algorithm. In this paper, we omit the update processing of variable nodes, such that the variable nodes message can be obtained from the update processing of check nodes.

In the standard BP algorithm, compared (2) with (3), it is very straightforward to rewrite the vertical step (2) as

$$Z_{mn}^{(k)}=Z_n^{(k)}-L_{mn}^{(k)} \quad (4)$$

The efficient message computation is derived by substituting (4) into (1), given the following updating formula for the incoming and outgoing messages from a check node c_m. Hence, it merges the horizontal step and the vertical step into a single horizontal step. In summary, the single-step BP algorithm is simply given as follows, as MS algorithm has been presented in [9]:

- ***Initialization:*** *For* $n\in[1,N]$,

$$L_{mn}^{(0)}=0$$

- ***Horizontal Step (update check node):***

$$Z_n^{(k)} = Z_n^{(0)}$$

$$L_{mn,BP}^{(k)} = 2\tanh^{-1}\left(\prod_{n' \in N(m) \setminus n} \tanh\left(\frac{Z_n^{(k-1)} - L_{mn'}^{(k-1)}}{2}\right)\right) \qquad (5)$$

$$Z_n^{(k)} += L_{mn}^{(k)}$$

- ***Hard decision:***

Estimate the original codeword $\hat{x}_n^{(k)}$ as

$$\hat{x}_n^{(k)} = \begin{cases} 0, & \text{if } Z_n^{(k)} > 0; \\ 1, & otherwise; \end{cases} \quad \text{for } n \in [1, N]$$

Compared with the standard double-step BP algorithm, the single-step method is not only faster but also improves memory efficiency reducing the implementation complexity. It saves memory by storing $Z_n^{(k)}s$, which are of N items, instead of $Z_{mn}^{(k)}$, which are of $N \bullet d_v$ (average variable node degree) items. For software implementations, the single-step method can be efficiently implemented with a significant reduction of the complexity and the memory due to the storing need only for the addressing from check nodes to variable nodes. Therefore it cuts down the amount of memory used for addressing by half.

4. Simplifications Of BP Algorithm

Many simplifications of the BP algorithm, that just simplified the check nodes update processing, for decoding LDPC codes have been investigated. Now the different variants of MS algorithm using the single-step method, that simplified not only the check nodes updated processing but also the variable nodes updated processing, are discussed as follow.

4.1 Min-Sum Algorithm

Taking advantage of the odd symmetry property of the function tanh(x), MS algorithm simplifies the updating rule in check nodes by modifying (5) into two parts:

$$L_{mn,MS}^{(k)} = \prod_{n' \in N(m) \setminus n} \alpha_{mn'} \times \min_{n' \in N(m) \setminus n} \beta_{mn'} \qquad (6)$$

where $\alpha_{mn} = sign\left(Z_n^{(k-1)} - L_{mn}^{(k-1)}\right)$,

$$\beta_{mn} = \left|Z_n^{(k-1)} - L_{mn}^{(k-1)}\right|$$

The MS algorithm makes some computation reduction on the check node update and the initialization of the input LLR $Z_n^{(0)} = y_n$. The knowledge of the noise power σ^2 is unnecessary in MS algorithm.

4.2 Normalized/Offset Min-Sum Algorithm

It is clear that the MS algorithm reduces the complexity of check node update, which is made at the expense of a substantial loss in performance. Therefore, there are two basic methods which make MS algorithm more efficient. They are the normalized min-sum (NMS) algorithm and the offset min-sum (OMS) algorithm.

$$L_{mn,NMS}^{(k)} = \alpha \bullet L_{mn,MS}^{(k)} \qquad (7)$$

and

$$L_{mn,OMS}^{(k)} = \max\left\{\left|L_{mn,MS}^{(k)}\right| - \beta, 0\right\} \qquad (8)$$

where α is a normalized factor smaller than 1 while β is a offset factor. The values are to be found theoretically by density evolution (DE) [6]. These two algorithms are both useful techniques which make good tradeoffs between performance and decoding complexity, especially for the short and medium length regular LDPC codes. And they successfully outperform the MS algorithm with only increase a little complexity.

4.3 Adaptive Offset Min-Sum Algorithm

To achieve the optimal performance, α and β should vary with different SNR and different iterations. In [7], an adaptive offset min-sum (AOMS) algorithm is presented, which lets the offset factor adaptively change for the better performance. The adaptive offset factor is defined as

$$\beta_t = \frac{1-\alpha}{2} \bullet \lceil 2 \bullet \min_{n' \in N(m) \setminus n} \beta_{mn'} \rceil \qquad (9)$$

Although the section of ranges β_t from 0 to infinite, the actual offset factors are able to be reduced to 8 parts.

$$\beta_t = \frac{1-\alpha}{2} \bullet t = \begin{cases} 0.1t, & t \in [1, 7] \\ 0.8, & t \in [8, \infty] \end{cases} \qquad (10)$$

where the fix normalized factor α is set to be 0.8 by the DE [6] and $t = \lceil 2 \bullet \min_{n' \in N(m) \setminus n} \beta_{mn'} \rceil$.

This means that there are 8 different adaptive offset

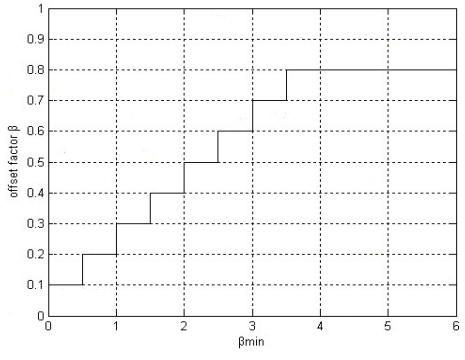

Fig. 2 The adaptive offset function of β_t.

factors range from 0.1 to 0.8. The adaptive offset factor function of $\beta_{\min}$ is showed as Figure 2. It looks like a step function where different input has a different output β_t. The advantage of this method is that it combines normalized and offset schemes together. And it only increases a little computation load and needs a memory place for one look-up table of length 8.

5. Simulation Results

In Figure 3, we present the bit error rate (BER) performance of AOMS algorithm comparing with the BP, MS, NMS and OMS algorithms for an (N, K) = (2016, 1008) irregular LDPC code, with maximum 20 iterations. In this case, we set α = 0.7 and β = 0.2 based on density evolution results. From Figure 3, we show that the MS algorithm has about 0.4 dB performance gap compared to the BP algorithm at the bit error rate of 10e-4. Most of the gap between BP and MS algorithm can be bridged by both the NMS and OMS algorithms. However we can observe that the AOMS algorithm gets closer to the BP algorithm, only about 0.05 dB away at the bit error rate of 10e-4. As the SNR increases, the BP and AOMS algorithms indeed get the very remarkable performance, even reaches the bit error rate of 10e-7.

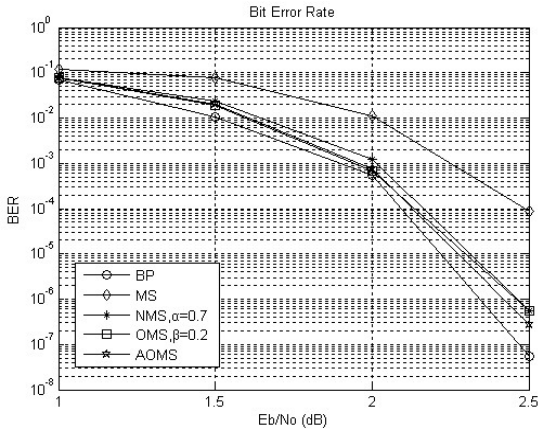

Fig. 3 Performance of the (2016, 1008) LDPC code with BP, MS, NMS, OMS and AOMS algorithm for 20 decoding iterations.

Figure 4 shows the BER performance of the BP, MS, OMS and AOMS algorithms for (4032, 2016) LDPC code with 80 iterations. The AOMS algorithm also gets closer to the BP and achieves a 0.05dB coding gain over the OMS algorithm. Moreover, it performs even slightly better than BP decoding at the high SNRs region since the BP decoding is not optimum for LDPC codes of finite length.

At last, we show some simulation results for LDPC codes with different decoding iterations. Figure 5 depicts the BER performance of 10 and 100 maximum decoding iterations for (1008, 504) LDPC code, with the BP, MS, OMS and AOMS algorithms. The solid line and dashed line denote the 10 and 100 maximum decoding iterations, respectively. It shows that 100 decoding iteration gets better BER than the small 10 decoding iteration at the same SNRs.Importantly, these results suggest that little additional improvements could be achieve the good performance close to the BP algorithm, and they are faster decoding using single-step method in our software implementation with low complexity.

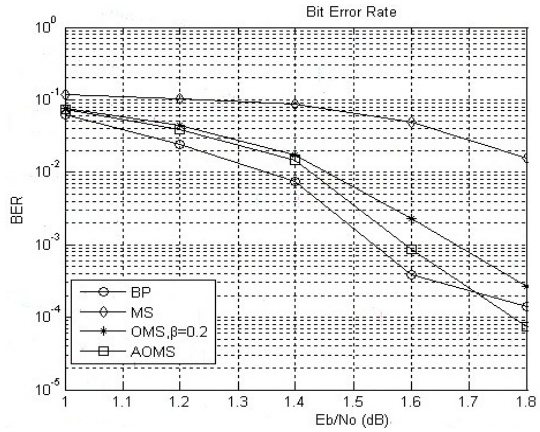

Fig. 4 Performance of the (4032, 2016) LDPC code with BP, MS, NMS, OMS and AOMS algorithm for 80 decoding iterations.

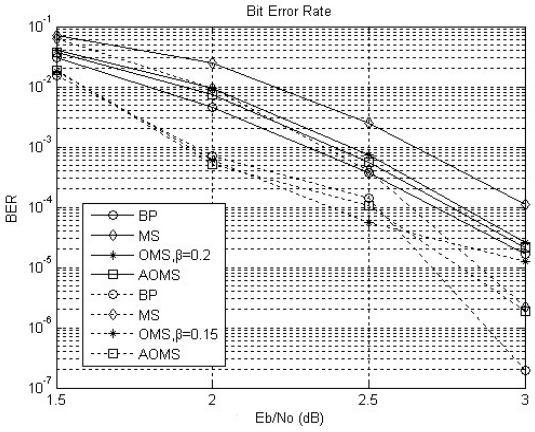

Fig. 5 Performance of the (1008, 504) LDPC code with BP, MS, OMS and AOMS algorithm for 10 (solid) and 100 (dashed) decoding iterations.

6. Conclusions

Efficient implementations of the BP, MS, NMS, OMS and AOMS algorithms using the single-step method for decoding LDPC codes have been investigated. The software implementations have shown that the AOMS algorithm using single-step method results faster

decoding speed of the standard double-step algorithms and better efficiency in terms of memory requirements with a significant reduction in implementation complexity. And they have also shown that it is possible to attain the performance of the BP extremely closely with significant reduced-complexity variants of the MS, NMS, OMS, AOMS algorithms, especially the AOMS algorithm performs more close to the BP algorithm.

감사의 글

이 논문은 2008년도 충북대학교 학술연구지원사업의 연구비 지원에 의하여 연구되었음(This work was supported by the research grant of the Chungbuk National University in 2008)

References

[1] R. G. Gallager, "Low-Density Parity-Check Codes," PhD thesis, MIT, Jully 1963.

[2] D. J. C. Mackay, "Good Error-Correcting Codes based on Very Sparce Matrices," IEEE Transactions on Information Theory, vol. 45, pp. 399-431, March 1999.

[3] D. J. C. MacKay and R. M. Neal, "Near shannonlimitperformanceoflow-densityparity-checkcodes,"Electron.Lett.,vol.32,no.18,pp. 1645 - 1646, Aug. 1996.

[4] R. M. Tanner, "A recursive approach to low complexity codes," IEEE Trans. Inform. Theory, vol. IT-27, no. 5, pp. 399 - 431, Sept. 1981.

[5] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low density parity check codes based on belief propagation," IEEE Trans. Commun., vol. 47, no. 5, pp. 673 - 680, May 1999.

[6] J. Chen and M. P. C. Fossorier, "Near-optimum universal belief propagation based decoding of low-density parity check codes," IEEE Trans. Commun., vol. 50, no. 3, pp. 406 - 414, Mar. 2002.

[7] J. Chen and M. P. C. Fossorier, "Density evolution for two improved BP-based decoding algorithms of LDPC codes," IEEE Commun. Letters , vol. 6, pp. 208-210, May 2002.

[8] M. Jiang, C. Zhao, L. Zhang, and E. Xu, "Adaptive Offset Min-Sum Algorithm for Low-Density Parity Check Codes," IEEE Commun. Letters, vol. 10, no. 6, June 2006.

[9] X. Huang, "Single-Scan Min-Sum Algorithms for Fast Decoding of LDPC Codes," IEEE Information Theory Workshop, Chengdu,China,Sep2006.

저 자 소 개

林 小 菊 (Xiaoju Lin)

2008년 7월 Shenzhen University of China, 전자공학과 (공학사). 2008년 9월~현재 충북대학교 전기공학과 석사과정

<관심분야> LDPC Communication system

康 秀 蘭 (Gansuren Baasantseren)

2008년 2월 Mongolian Science and Technology University, 전기공학과 (공학사). 2008년 9월~현재 충북대학교 전기공학과 석사과정

<관심분야> Ulltra-wideband Communication system

이 해 기 (李 海 基)

1981년 충북대학교 공업교육 (공학사). 1985년 성균관대학교 전기공학 (공학석사). 1990년 성균관대학교 전기공학 (공학박사). 1991년~현재 충청대학 전지전자학부 교수

<관심분야> 신호처리, 전력제어, 씨퀀스제어

김 성 수 (金 聖 洙)

1997년 Univ. of Central Florida(공학박사). 1999년 ~ 2001년 우석대학교 전기공학과 조교수. 2001년 9월 ~ 현재 충북대학교 전자정보대학 전기공학부 교수

<관심분야> 디지털통신, 인공지능, 신호처리