Node Monitoring 알고리듬과 NP 방법을 사용한 효율적인 LDPC 복호방법

서희종*

Node Monitoring Algorithm with Piecewise Linear Function Approximation for Efficient LDPC Decoding

Hee-jong Suh*

요 약

본 논문에서는 NM(node monitoring) 알고리듬과 NP(Piecewise Linear Function Approximation)를 사용해서 LDPC 코드 복호의 복잡도를 감소시키기 위한 효율적인 알고리듬을 제안한다. 이 NM 알고리듬은 새로운 node-threshold 방법과 message passing 알고리듬에 근거해서 제안되었는데, 이에 NP 방법을 사용해서 알고리듬의 복잡도를 더 줄일 수 있었다. 이 알고리듬의 효율성을 입증하기 위해서 모의 실험을 하였다. 모의실험 결과, 기존에 잘 알려진 방법에 비해서 20% 정도 더 효율적이었다.

ABSTRACT

In this paper, we propose an efficient algorithm for reducing the complexity of LDPC code decoding by using node monitoring (NM) and Piecewise Linear Function Approximation (NP). This NM algorithm is based on a new node-threshold method, and the message passing algorithm. Piecewise linear function approximation is used to reduce the complexity for more. This algorithm was simulated in order to verify its efficiency. Simulation results show that the complexity of our NM algorithm is reduced to about 20%, compared with thoes of well-known method.

키워드

LDPC codes, node monitoring, node-threshold, piecewise linear function approx

I. INTRODUCTION

Low-density parity-check (LDPC) codes were first proposed by Gallager in his doctoral dissertation [1] and were forgotten for several decades. The study of LDPC codes was resurrected in the mid-1990s because of its good performance and lower decoding complexity. LDPC codes are linear block codes which are falling only 0.04dB short of the Shannon limit [2].

LDPC codes algorithm has two parts, which are encoding and decoding. Its encoding is very easy, but its decoding is more complicated than the encoding. Decoding process of the LDPC code takes much time with the existing algorithms [1, 2, 3].

There are three approaches to reduce the

* 전남대학교 전자통신공학과(hjsuh@jnu.ac.kr)
접수일자 : 2010. 12. 12 심사(수정)일자 : 2011. 01. 17 게재확정일자 : 2011. 02. 09

complexity of LDPC decoding, which are to simplify the computation of the decoder, reduce the number of iterations of the decoder and diminish the messages of the iterations. In this approaches, there are many algorithms, such as the massage passing algorithm (MPA) [2, 8], the min-sum algorithm [4], the various scheduling techniques [5], the forced convergence method [3, 6] and the bit-level stopping method [7]. The MPA algorithm is known as one of the most effective methods among the well-known methods [6].

In this paper, we propose a node monitoring (NM) algorithm. This method is a new one on reducing the decoding complexity, which is different from the other methods because it uses two monitoring vectors to monitor all the variable nodes and check nodes. For initialization the two vectors are filled with zeros. Some nodes are steady enough in case that these nodes achieve some node-threshold, at this case the vector for these nodes will stop receiving and updating process of these nodes. Our algorithm is a new method because our NM algorithm only monitors the check nodes instead of monitoring both the check and variable nodes. Although it does not monitor the variable nodes, it has the same performance in monitoring the messages of the check and variable nodes. But, this will finally reduce the complexity of the decoding process. And the piecewise linear function approximation [9] is a good way to reduce the function that is used in the program of the node monitoring algorithm.

To verify the efficiency of our algorithm, we do simulations. Then, because the MPA algorithm is known as most effective methods among the well-known methods, we did comparing our NP algorithm only with the MPA algorithm. With this comparison, we could get the conclusion that NP algorithm is more efficient method than the method. Therefore, we knew our algorithm is most efficient one than the well known methods. Its efficiency was shown to be improved by about 20% improvement.

This paper is organized as follows. Section II gives the background of the LDPC code. And Section III introduces a new decoding algorithm of the LDPC codes and the piecewise linear function approximation. In Section IV there are the results of the simulation of the algorithm. Conclusion is Section V. References are next.

II. Related Algorithm and Problem

There are two types LDPC codes, regular LDPC codes and irregular LDPC codes, which can be represented by a Tanner graph with N variable nodes on the left (representing the bits of the code word) and M check nodes on the right (representing the parity checks constraints).

Fig 1 is a Tanner graph of a block length 8 (3, 6). Nodes on the left hand side in Fig. 1 represent the code bits; nodes on the right hand side represent the parity check constraints. Throughout the decoding process, the nodes exchange messages $V_{n,m}$ and $U_{m,n}$ over the edges of the graph.

A. Standard brief propagation (BP) algorithm for iterative decoding of LDPC codes

We introduce the message passing algorithm (MPA), the most popular decoding algorithm [2], to make a our algorithm for the LDPC code decoding.

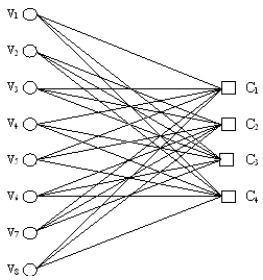

Fig. 1 Tanner graph of a block length 8 (3,6) regular LDPC code

We suppose a regular binary (N, K) (d_v, d_c) LDPC code C is used for error control over an AWGN channel with a mean of zero and power spectral density $N_0/2$, and assume BPSK signals with unit energy, which maps a code word $w=(w_1, w_2, \cdots, w_n)$ into a transmitted sequence $q=(q_1, q_2, \cdots, q_n)$, according to $q_n=1-2w_n$, for $n=1, 2, \cdots, N$ and $w_n=1$ or $w_n=0$. If $\overline{w}=[w_n]$ is a code word in C, and $\overline{q}=[q_n]$ is the corresponding transmitted sequence, then the received sequence is $\overline{q}+\overline{g}=\overline{y}=[y_n]$, with $y_n=q_n+g_n$, where for $1 \leqq n \leqq N$, g_n is Gaussian random variables with a mean of zero and variance $N_0/2$. Let $\overline{H}=[H_{mn}]$ be the parity check matrix which defines an LDPC code. We denote the set of bits that participate in check m by $N(m)=\{n : H_{mn}=1\}$ and the set of checks in which bit n participates as $M(n)=\{m : H_{mn}=1\}$. And we denote $N(m)/n$ as the set $N(m)$ out of bit n, and $M(n)/m$ as the set $M(n)$ out of check m. In order to explain the iterative decoding, we define the following notations with the ith iteration:

- $U_{ch,n}$: The log-likelihood ratio (LLR) of bit n which is from the channel output y_n. In belief propagation decoding, we initially set $U_{ch,n}=(4/N_0)y_n$.
- $U_{mn}^{(i)}$: The LLR of bit n that check node m sends to bit node n.
- $V_{nm}^{(i)}$: The LLR of bit n that bit node n sends to check node m.
- $V_n^{(i)}$: The posteriori LLR of bit n.

The standard belief propagation algorithm is carried out as follows [5]:

Initialization: Set i = 0, and the maximum number of iteration to I_{Max}. For all m, n, set $V(0)_{nm}=U_{ch,n}$, $U(0)_{mn}=0$.

Step 1: (i) check-node update: for $1 \leqq m \leqq M$ and each $n \in N(m)$:

$$U_{mn}^{(i)}=2\tanh^{-1}\prod_{n' \in N(m)/n}\tanh\frac{V_{n'm}^{(i-1)}}{2} \quad (1)$$

(ii) bit-node update: for $1 \leqq n \leqq N$ and each $m \in M(n)$:

$$V_{nm}^{(i)}=U_{ch,n}+\sum_{m' \in M(n)/m}U_{m'n}^{(i)} \quad (2)$$

$$V_n^{(i)}=U_{ch,n}+\sum_{m \in M(n)}U_{mn}^{(i)} \quad (3)$$

Step 2: Hard decision and stopping criterion test:

(i) Create $\overline{w}^{(i)}=[\overline{w}_{]n}^{(i)}$ in which $\overline{w}_{=1n}^{(i)}$ if $V_n^{(i)}<0$, and $\overline{w}_{=0n}^{(i)}$ if $V_n^{(i)} \geqq 0$.

(ii) If $H \cdot \overline{w}_{=0n}^{(i)}$ or $i=I_{Max}$, stop the decoding iteration and go to Step 3. Otherwise set $i=i+1$ and go to Step 1.

Step 3: Output $\overline{w}^{(i)}$ as the decoded code word.

B. The complexity reducing problem

The standard belief propagation algorithm for iterative decoding of LDPC codes has about two problems. First, for both the check-to-bit messages and bit-to-check messages, the more independent information is used to update the messages, the more reliable they become. Iteration i of the standard two steps implementation of the belief propagation algorithm uses all values $V_{n'm}^{(i-1)}$ computed at the previous iteration in (1). However certain values $V_{n'm}^{(i)}$ could already be computed based on a partial computation of the values $U_{mn}^{(i)}$ obtained from (2), and then be used instead of $V_{n'm}^{(i-1)}$ in (1) to compute the remaining values $U_{mn}^{(i)}$. So, if we use certain values $V_{n'm}^{(i)}$ to compute a partial values $U_{mn}^{(i)}$, then we can reduce the complexity. Second, during the iterative

decoding, the fact is that a large number of variable nodes converge to a strong belief after very few iterations, i.e., these bits have already been reliably decoded and we can skip updating their messages in subsequent iterations. If we have some ways to decide whether a node should update massage at a given iteration, then we also can reduce the complexity of the decoding process.

III. NODE MONITORING ALGORITHM AND PIECEWISE LINEAR FUNCTION APPROXIMATION

For the two problems we described above, we introduce a decoding method based on the node monitoring algorithm. Most of the variable nodes achieve a stable state after very few iterations, barely to change the bit that it represents. So decoder could skip updating their messages in subsequent iterations. Node monitoring algorithm uses this phenomenon to reduce the complexity of the decoding. It updates messages only that instable nodes send at some iteration. In order to describe the degree of some nodes that have achieved a certain stable state, we define the "aggregate messages" B_i for each variable node as follows:

$$B_i = \left| V_n^{(i)} \right| = \left| U_{ch,n} + \sum_{m \in M(n)} U_{mn}^{(i)} \right| \quad (4)$$

Checking B_i against the node-thresholds t_v will find the nodes that are stable. B_i is the confidence of the variable node to be in state 0 or 1, and the bigger B_i is the more stable variable node is. Now, we introduce the node monitoring algorithm in detail. Suppose a decoding system dealing with LDPC codes. First, we define two vectors to store the state of check nodes and variable nodes, respectively. They are deactivated-v and deactivated-c. For initialization, deactivated vectors are filled with zeros. Then the variable nodes get the information bits, and compute the value $V(0)_{nm} = U_{ch,n}$. And they send these values as messages to their neighbor check nodes. When their neighbor check nodes receive these messages, the check nodes begin to compute their messages $U_{mn}^{(i)}$ and send them to variable nodes. Now, two kinds of nodes have finished their first message sending process. Variable nodes begin a deciding program to decide whether continue the next iteration or not. At this time, the variable nodes also compute B_i to compare with the node-threshold t_v. If it is bigger than the t_v, the element in deactivated-v to this node will change to 1 from 0. It is a flag that represents a stable node at that iteration. Then decoder checks each check nodes whether its neighbor variable nodes are all stable. If they are all in stable state, the element in deactivated-c to this node will change to 1 from 0. And this check node will not send the message to its neighbors, because its neighbors are all in stable state. But even if one of its neighbors is instable, the element in deactivated-c will still be 0, and all its neighbor variable nodes will be reactivated by resuming their elements to 0 again in deactivated-v. Then begin the new iteration just as above except that check the deactivate vector before sending the messages. If the element for a node is 1, then skip this node's message sending.

Piecewise linear function approximation is used to reduce the function of $\tanh(x)$. The function of $\tanh(x)$ is not a linear function, so it will cost much time to compute its value. First we can write the equation (1) in another way: separate Piecewise linear function approximation is used to reduce the function of $\tanh(x)$. The function of $\tanh(x)$ is not a linear function, so it will cost much time to compute its value. First we can write the equation (1) in another way: separate $V_{nm}^{(i-1)}$ to

$$V_{nm}^{(i-1)} = \alpha_{nm}\beta_{nm} \quad (5)$$
$$\alpha_{nm} = sign[V_{nm}^{(i-1)}]$$
$$\beta_{nm} = |V_{nm}^{(i-1)}|$$

we then have

$$U_{nm}^{(i)} = \prod_{n' \in N(m)/n} \alpha_{n'm} \cdot \Phi\left(\sum_{n' \in N(m)/n} \Phi(\beta_{n'm})\right) \quad (6)$$

Where we have

$$\Phi(x) = -\log[\tanh(\frac{x}{2})] = \log(\frac{e^x+1}{e^x-1}) \quad (7)$$

The function of $\Phi(x)$ is fairly well behaved, and $.\Phi^{-1}(x) = \Phi(x)$ But, $\Phi(x)$ is not a linear function. So we use Piecewise linear function $\Phi 1(x)$ to get an approximation of $\Phi(x)$ and the complexity can thus be reduced for more. It is showed in Fig. 2.

What we have introduced above is the node monitoring algorithm. It reduces the complexity of the decoding process, but barely brings out degradation of the bit error rate (BER) and the frame error rate (FER) performance.

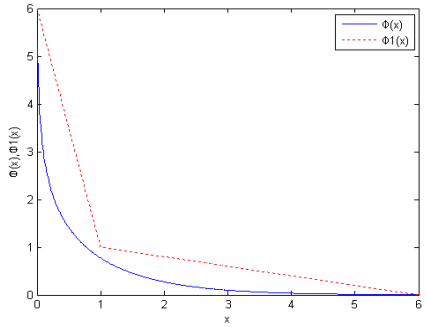

Fig. 2 The transfer function $\Phi(x)$ used in check node calculations and the piecewise linear approximation of it.

IV. SIMULATION RESULTS

We simulated with a (3, 6) regular LDPC code with a block-length of 2000 bits and rate R=0.5.

Fig. 3 shows the bit error performance of three kinds of decoding algorithm for comparison. It is easy to see that the performance of the Combination of Node Monitoring Algorithm and Piecewise Linear Function Approximation algorithm (NM-PW) is much better.

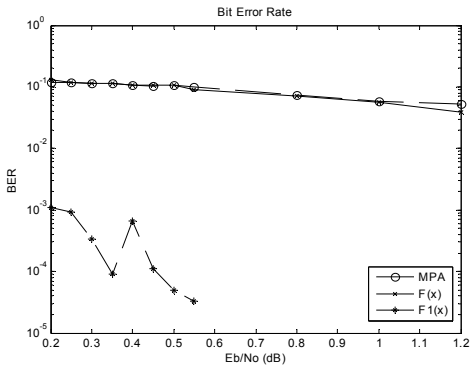

Fig. 3 Bit error performance of the LDPC code with a block-length of 2000 bits.

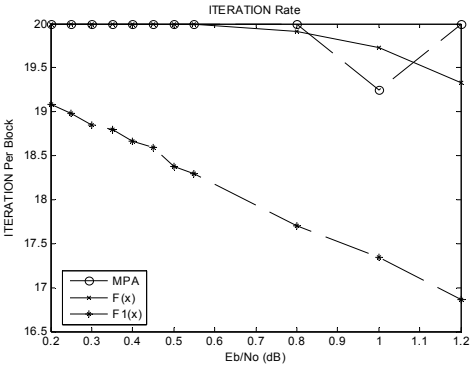

Fig. 4 Frame error performance of the LDPC code with a block-length of 2000 bits.

Fig. 4 shows the frame error performance of three kinds of decoding algorithm for comparison. It

is easy to see that the performance of the Combination of Node Monitoring Algorithm and Piecewise Linear

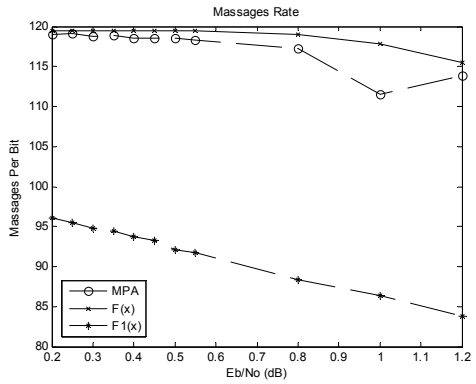

Fig. 5 Messages per bit of the LDPC code with a block-length of 2000 bits.

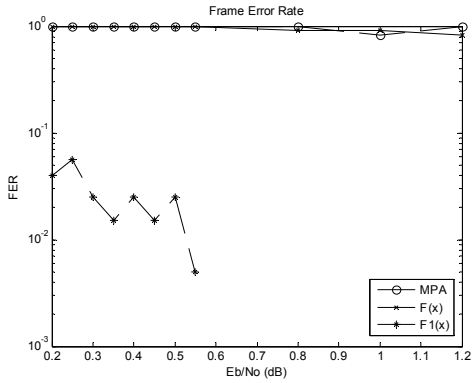

Fig. 6 Iterations per block of the LDPC code with a block-length of 2000 bits.

Function Approximation algorithm (NM-PW) is much better too. Fig. 5 shows the Messages per bit of three kinds of decoding algorithm for comparison. It is easy to see that the complexity of the Combination of Node Monitoring Algorithm and Piecewise Linear Function Approximation algorithm (NM-PW) is about 80% of the other two. Fig. 6 shows the Iterations per block of three kinds of decoding algorithm for comparison. It is easy to see that the iterations of the Combination of the Node Monitoring Algorithm and Piecewise Linear Function Approximation algorithm (NM-PW) is less than the other two.

V. CONCLUSIONS

We have proposed a Combination of Node Monitoring Algorithm and Piecewise Linear Function Approximation algorithm (NP) to reduce the complexity of the LDPC decoding. With the simulation, we could see its better performance than existing well-known methods.

So, we can conclude that this method is the best way to reduce efficiently the complexity of the decoder. But we must endeavor in order to achieve a more practical node monitoring decoder for LDPC codes. And we must try to make more improvement to get better performance.

REFERENCES

[1] R. G. Gallager, Low-Density Parity-Check Codes. MIT Press, 1963.

[2] John R. Barry, "Low-Density Parity-Check Codes," Georgia Institute of Technology, 2001.

[3] G. Fettweis, E. Zimmermann, and W. Rave, "Forced convergence decoding of LDPC codes: EXIT chart analysis and combination with node complexity reduction techniques," in Proc. 11th European Wireless Conference 2005.

[4] J. Zhang, M. Fossorier, D. Gu, and J. Zhang, "Two-dimensional correction for min-sum decoding of irregular LDPC codes," IEEE Communication Lett., vol. 10, pp. 180-182, 2006.

[5] Y. Wang, J. Zhang, M. Fossorier, and J. Yedidia, "Reduced latency iterative decoding of LDPC codes," IEEE Global Telecommunication Conf. 2005.

[6] D. Levin, E. Sharon, and S. Litsyn, "Lazy Scheduling for LDPC Decoding," IEEE Communication Lett., vol. 11, pp. 70-72, Jan.

2007.
[7] D. H. Kim and S. W. Kim, "Bit-level stopping of turbo decoding," IEEE Communication Lett., vol. 10, pp. 183 – 185, 2006.
[8] T. J. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message passing decoding," IEEE Trans. Inf. Theory, vol. 47, pp. 599 – 618, 2001.
[9] X. Y. Hu, E. Eleftheriou, D.-M. Arnold, and A. Dholakia. Efficient Implementations of the Sum-Product Algorithm for Decoding LDPC Codes. Proceedings of the IEEE Globe Com 2001, pp. 1036 – 1036E, 2001.

저자 소개

서희종(Hee-jong Suh)

1975년 한국항공대학교 항공통신공학과 졸업(공학사)
1984년 중앙대학교 대학원 전자공학과 졸업(공학석사)
1996년 중앙대학교 대학원 전자공학과 졸업(공학박사)
1980년～2006년 여수대학교 전자통신공학과 교수
2006년～현재 전남대학교 전자통신공학과 교수
※ 관심분야 : 컴퓨터 네트워크, 인터넷통신, 위성통신