SCCC와 LDPC 코드 복호화를 위한 Optimal과 Suboptimal 반복검파 알고리즘

정회원 허 준*

Constrained Iterative Decoding on the Serially Concatenated Convolution Code and Low-Density-Parity-Check Code

Jun Heo* *Regular Member*

요 약

Iterative decoding 알고리즘의 optimality 에 대한 고찰을 바탕으로, 표준화 되어있는 sub-optimal decoding 알고리즘을 변화시켜서, 실현 가능한 complexity하에서 optimal 성능에 가까워질 수 있는 방안을 제안하였습니다. 제안된 새로운 형태(Constrained)의 iterative decoding 알고리즘을 SCCC (Serial Concatenated Convolution Code) 와 LDPC(Low-Density Parity-Check) code에 적용시켜서, 실제로 향상되는 성능과 complexity의 상관관계를 나타내었으며, Iterative decoding의 해석방법으로 최근에 각광 받는 Density Evolution 기법을 자세히 소개하고, 이를 적용하여 CID(Constrained Iterative Decoding)이 보여주는 성능의 향상을 설명하였습니다.

ABSTRACT

A modification to the standard iterative decoding (message passing) algorithm that yields improved performance at the cost of higher complexity is introduced. This modification is to run multiple iterative decoders, each with a different constraint on a system variable (e.g., input value, state value, etc.). This Constrained Iterative Decoding (CID) implements optimal MAP decoding for systems represented by single-cycle graphs (e.g., tail-biting convolutional codes). For more complex graphical models, the CID is suboptimal, but outperforms the standard decoding algorithm because it negates the effects of some cycles in the model. In this paper, it is shown that the CID outperforms the standard ID for a Serially Concatenated Convolution Code (SCCC) system and Low-Density-Parity-Check (LDPC) code system, especially when the interleaver size is small. Density evolution analysis is used to show how CID improves the convergence relative to that of standard ID by showing that the threshold of the CID is lower than that of the standard ID.

Ⅰ. Introduction

Since turbo codes were introduced, Iterative Decoding (ID) has been researched to develop effective algorithms which yield either better performance or low complexity. These ID algorithms were understood as Belief Propagation (BP) algorithms which were known to be optimal for a graph without cycles [1],[2],[3]. However, the ID algorithms are suboptimal, because they are developed ignoring the cycles of codes (e.g., turbo codes, LDPC codes, SCCC etc). There have been several attempts to find an optimal decoding algorithm for a code with cycles. As a special case, the optimal decoding of a tail-biting code, which has a single cycle, was investigated in

* 하이닉스 반도체(주) (junh0522@hotmail.com))

논문번호 : 020475-1030, 접수일자 : 2002년 10월 30일

[4],[5].

When all cycles are removed from a code, the BP algorithm is optimal on the modified system. As a method of removing the cycles, it is proposed to put constraint on nodes (i.e., system variables) of cycles. In other words, each constraint of those nodes is represented by a set of values and the BP algorithm is repeatedly applied based on each constraint. This modified ID algorithm is called as Constraint Iterative Decoding (CID). The tail-biting code is a good example of a single cycle code which is easily cut by putting a constraint on the node (the initial and final state variable). Therefore, CID is optimal for the tail-biting code. As a example with multiple cycles which are easily cut as well, a 4-cycle tail-biting code is presented. This example simply shows that an optimal decoding is still possible when a code has more than one cycle.

This CID algorithm can be applied to decode more complex codes with many cycles, for example turbo codes, LDPC codes, and SCCC etc. Generally, optimal CID algorithms are not possible for these codes because too many constraints are required to cut all the cycles. Although the CID is not optimal for these codes, it outperforms the standard ID algorithm because it negates the effects of some cycles in the graphical representation of the codes.

Along with the development of ID algorithms, analysis has been successful to unveil the characteristics of ID. Particularly, recently developed density evolution techniques have achieved the asymptotic capacity of Turbo codes, when the interleaver size and the number of iteration go to infinity [6],[7]. This asymptotic capacity was empirically obtained based on input-output SNR evolution curves of each iterative decoder. It was shown that the threshold (capacity) is well matched to the region where the bit error curves start to fall down [7]. Density evolution was used to explain many mysteries of turbo codes and SCCC in [8], for example, the importance of systematic bits and recursive constituent codes etc. The method of SNR evolution is based on the fact that the exchanged soft information is well approximated by a Gaussian random variable. In this paper, the CID algorithms with multiple constraint marginalization options are presented. A tail-biting code and 4-cycle tail-biting code are given as examples where CID is optimal. The performance of CID is compared to that of standard tail-biting ID [9]. For more complex codes like SCCC and LDPC codes, it is shown that the suboptimal CID yields better performance at the cost of larger complexity. Because of the parallel structure of CID, it could potentially achieve the standard ID performance with less latency (less interleaver size). In addition, the density evolution technique is used to show how CID improves the convergence compared to that of standard ID. Specifically, the SNR evolution curves of CID shows lower threshold than that of standard ID.

The rest of this paper is organized as follows. In Section II, the CID algorithm is developed as either optimal or suboptimal decoding algorithm depending on codes. Multiple constraint marginalization methods are suggested as well. The performance of CID for the optimal and suboptimal cases are shown in Section III. The simulation results show a tradeoff between the complexity and the performance in CID. Some concluding remarks are given in Section IV.

Ⅱ. CID Algorithms

1. Optimal CID Algorithms

When only a small number of nodes are shared by all cycles of a code and each node has only a small number of configurations, CID can be optimal and feasible. For example, each node can take two values and there are M nodes common to all cycles, then the number of possible constraints is 2^M, with each constraint can be represented by an $M\times 1$ vector. One simple example is a tail-biting code which has only one common node (start and end states: $M=1$) of a single cycle and the number of condition values for the state variable is the number of configurations for that node [4]. In Fig. 1, the concept of state constrained decoding is illustrated for the 4-state non-recursive convolution code.

There are two ways to perform constrained marginalization:

Sum-marginalization[4]:

$$p(b_k, z) = \sum p(b_k, z \mid c_i) p(c_i) \equiv \sum p(b_k, z \mid c_i)$$

Max-marginalization:

$$p(b_k, z) = \max p(b_k, z \mid c_i) p(c_i) \equiv \max p(b_k, z \mid c_i)$$

where b and z represent the information sequence and noisy observation sequence, respectively, and $c_i \in C$ is a set of all possible constraints. Given a constraint c_i, $p(b_k, z \mid c_i)$ can be obtained by either a sum-product algorithm or a min-sum algorithm as:

sum-product:

$$p(b_k, z \mid c_i) = \sum p(z \mid b_k, c_i) p(b_k \mid c_i)$$

min-sum:

$$p(b_k, z \mid c_i) = \max p(z \mid b_k, c_i) p(b_k \mid c_i)$$

As an another example where CID is optimal, we present a 4-cycle tail-biting code. The 4-cycle tail-biting code consists of 4 data segments and the first segment is exactly same as a standard tail-biting code. After the first segment, the tail-biting bits of the first segment are appended

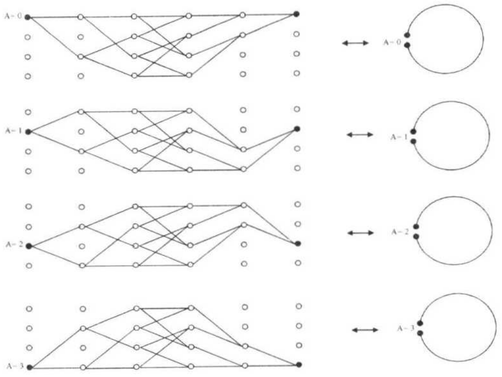


Fig. 1 The state constrained decoding on tail-biting code with 4-state non-recursive convolution code

at the end of each data segment to have the same starting and ending state.

This example shows that optimal decoding is possible when a code has more than one cycle and those cycles can be easily cut by constraints. Fig.2 represents the data and tail bit arrangement of both tail-biting and 4-cycle tail-biting codes. In Fig.2, the shaded bits represent the tail-biting bits repeatedly used in multiple data segments and the capital letters A to B and A' to E' represent the starting and ending state of each segment.

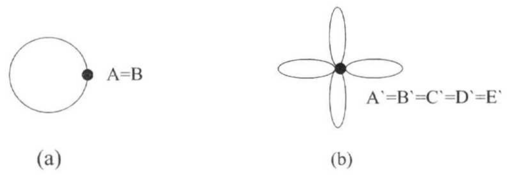


Fig. 2 Data and tail bit arrangement with graph (a)tail-biting code (b)4-cycle tail-biting code

2. Suboptimal CID Algorithms

When it is not possible to consider all constraint values of all common nodes, a small number of nodes can be selected and, for these selected nodes, CID may be applied as a suboptimal decoding algorithm. In other words, a few of the many cycles are removed from the graphical model of the code. For this suboptimal CID algorithm, the constrained marginalization is based on the selected constraints C' instead of all possible constraints C ($C' \subset C$). In addition to the marginalization options (*Sum*, *Max*), two ad hoc marginalization options for min-sum algorithm are proposed:

Reliability-marginalization:

$$c_i^* = \arg\max \mid p(b_k = 0, z \mid c_i) - p(b_k = 1, z \mid c_i) \mid$$

$$b_k^* = \arg\max p(b_k, z \mid c_i^*)$$

Sequence-marginalization:

$$c_i^* = \arg\max \mid p(z \mid b, c_i) p(b \mid c_i)$$

$$b_k^* = \arg\max p(b_k, z \mid c_i^*)$$

The Reliability-marginalization method takes the constraint which gives higher reliability at time index k, while the Sequence-marginalization method takes the constraint which gives highest reliability for the whole sequence. In the following section, the performance of two ad-hoc methods are compared to that of the previous two methods when CID is suboptimal.

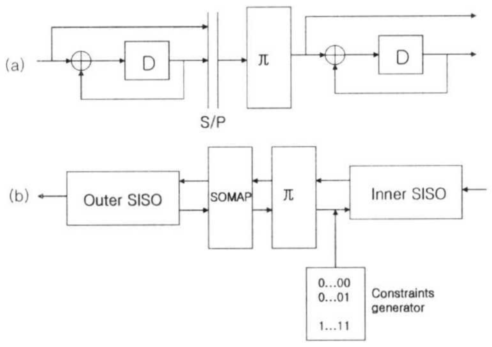


Fig. 3 Block diagrams of (a) the SCCC encoder, (b) the associated CID decoder

The CID algorithm is applied to a Serially Concatenated Convolutional Code (SCCC) and a regular (3,6) LDPC code, which have many cycles and nodes involved. At first, a small number of nodes M' out of all common nodes M are selected. For the binary case, the number of possible constraints is $2^{M'}$. Secondly, standard ID is executed for each of the $2^{M'}$ constraints. It should be noted that the constraints are explicitly applied only for the first iteration, however the soft information at subsequent iterations still evolves based on the constraints. After a fixed number of iterations (e.g, 10), the soft information from each constrained decoding is combined, and then marginalized to obtain the decoded sequence. Fig. 3 shows the SCCC system considered and the CID decoding blocks. Similar CID algorithm was applied for (3,6) regular LDPC code. The constraints were imposed on the message from variable nodes to check nodes.

Ⅲ. Simulation and SNR evolution

In this section, the performance of CID is shown, when it is optimal and suboptimal. For the optimal tail-biting decoding, Fig. 4 shows the performance of CID with the constraint on the starting and ending state. The performance of iterative tail-biting decoding was also shown with two wraps (iterations) [9]. For comparison, the performance of tail-bit decoding using Viterbi Algorithm (VA) was also presented. All decoding algorithms were based on min-sum algorithm with the block size $N=10$.

Because the starting and ending state of the tail-biting decoder can be determined correctly with high probability at high SNR and the tail-biting code saves the energy loss of tail-bits, both tail-biting decoders outperformed the tail-bit decoder at high SNR (i.e., the Eb/No reported includes the penalty for tail-bits). However, at low SNR, the probability of correct decision of the starting and ending state is very low. Therefore, the tail-bit decoder shows better performance despite tail-bit energy loss. The optimal CID performs slightly better than the iterative tail-biting decoding at all SNRs.

Fig. 5 shows the performance of the 4-cycle tail-biting CID algorithm. The CID shows slightly better performance than the tail-bit code at all SNR. The 4-cycle structure increases the probability of correct decision of starting and ending state because of longer observation. At the same time, the tail-bit energy loss is decreased. This results in smaller performance difference.

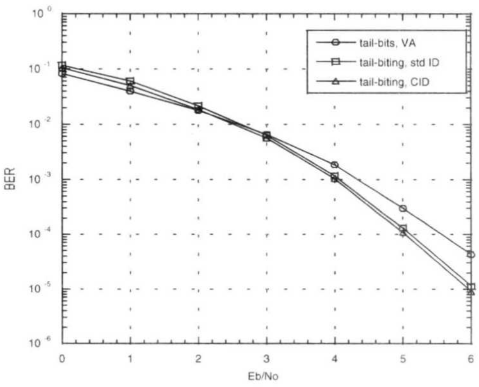


Fig. 4 Performance of tail-biting and tail-bit convolutional codes based on the min-sum algorithm with the block size N=10

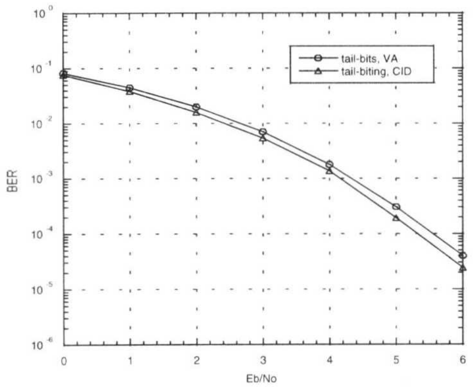


Fig. 5 Performance of the 4-cycle tail-biting codes based on the min-sum algorithm with the block size N =40

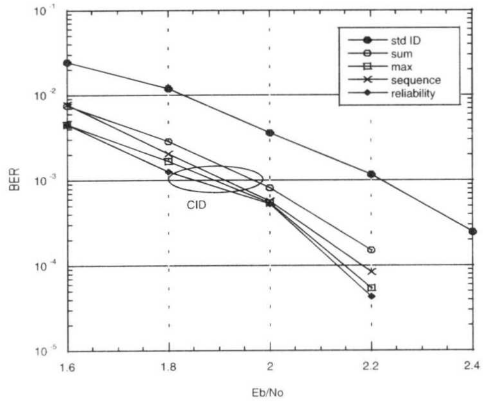


Fig. 6 Performance for SCCC based on min-sum algorithm with the interleaver size 256. M'=4 nodes were constrained.

TABLE I. The CID options of the SCCC decoder

SISO with constraints	M'	marginalization
Inner	4	Sum
Inner	4	Max
Inner	4	Sequence
Inner	4	Reliability

TABLEII. The CID options of the LDPC decoder

options	nodes with constraints	M'	marginalization
CID1	Check	2	Reliability
CID2	Check	4	Reliability

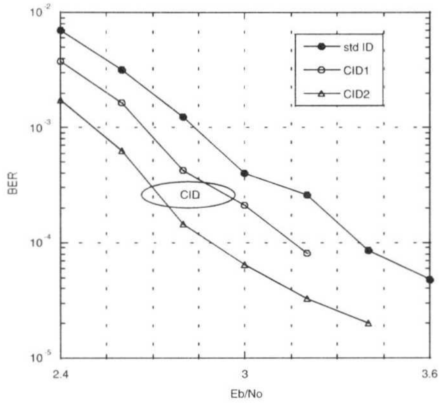


Fig. 7 CID performance for SCCC2 (2-state inner code, 16-state outer code) based on MSM algorithm and N=256, (a)CID1: constrained at 2 nodes, (b)CID2: constrained at 4 nodes. Both CID1 and CID2 used the Reliability-marginalization.

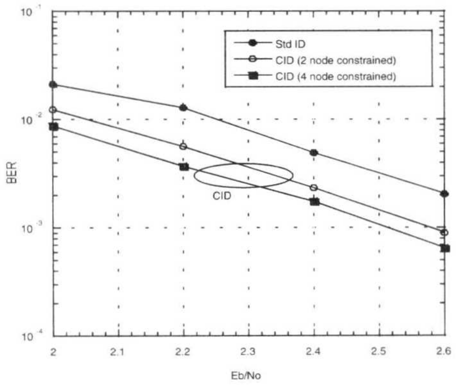


Fig. 8 CID performance of regular (3,6) LDPC code based on MSM algorithm with interleaver size 500. (a)CID1: constrained at 2 nodes, (b)CID2: constrained at 4 nodes. Both CID1 and CID2 were constrained at check nodes with the Reliability-marginalization.

Fig. 6 represents the performance of the CID on the SCCC system in Fig. 3 with a small interleaver size $N=256$. Table I shows four different CID options simulated. The constrains are posed at the inner Soft-In-Soft-Out (SISO) decoder. Because only some of the nodes were constrained,

CID was suboptimal and iteration improves performance. Both standard ID and CID were simulated up to 10 iterations. With approximate 2^4 time higher complexity, CID shows about 0.4 dB gain at the 10^{-3} BER. Among the CID options, the reliability-marginalization showed the best performance. A tradeoff between complexity and performance of CID was observed. As the number of nodes constrained increases, the performance is better. Fig.7 shows the tradeoff between complexity and performance of CID for different SCCC system which has 2-state inner code and 16-state outer code.

The complexity and performance tradeoff in CID was shown for regular (3,6) LDPC code in Fig. 8. With the Reliability-marginalization, the CID showed about 0.3 dB gain compared to that of standard ID.

Ⅳ. Convergence analysis of CID

The density evolution techniques in the literature can be classified into two categories. One is the analytic density evolution, which is mainly applicable to LDPC code [10]. The other is the simulation based density evolution (referred to "SNR evolution" hereafter), which is usually applicable to turbo codes and SCCC [6],[7],[8]. The SNR evolution in the literature can also be divided into two categories. One is actual SNR evolution and the other is gaussian SNR evolution.

In Fig. 9 and Fig. 10, the block diagram of these two SNR evolution technique are shown respectively.

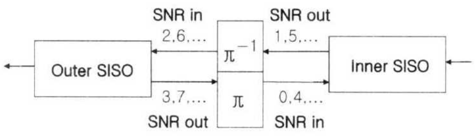


Fig. 9 Block diagram of actual SNR evolution

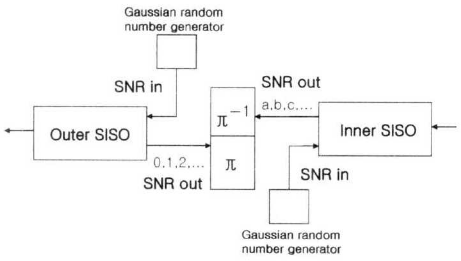
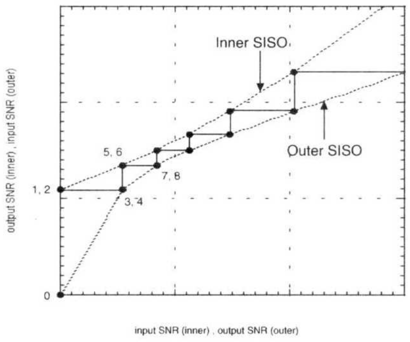
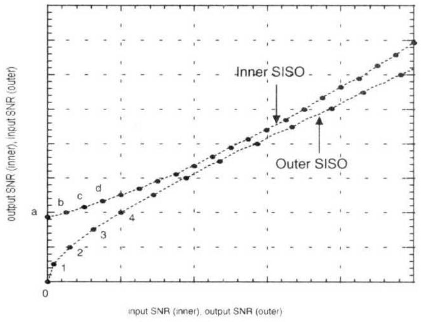


Fig. 10 Block diagram of gaussian SNR evolution

TABLE Ⅲ. Sample SNR evolution curves: actual(upper), gaussian(lower)





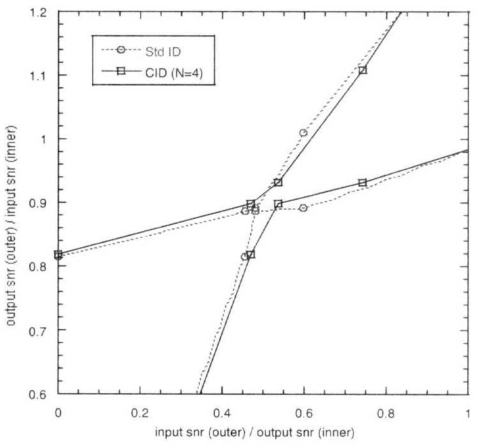


Fig. 11 SNR evolution curves of standard ID and CID on SCCC at 0.53dB.

The actual SNR evolution is calculated based on the collected LLRs at each iterative decoding step. From the collected LLRs, the mean μ for the exchanged soft information is computed. The SNR of a gaussian random variable with mean μ is approximated as $\frac{\mu}{2}$ based on the symmetry condition [11],[12]. In Fig. 9, the numbers indicate the order of actual SNR calculation. Meanwhile, the gaussian SNR evolution uses a generated soft information (LLRs) from a gaussian random number generator, which is fed into each constituent decoder. This idea is based on the well-known gaussian nature of the extrinsic soft information. As we can control the mean value of generated LLRs with the gaussian SNR evolution, the SNR curves and threshold can be calculated with more precisely compared to those of the actual SNR evolution. In Fig. 10, the numbers and alphabets are shown to indicates the SNRs of inner and outer SISO(Soft-In Soft-Out) are independent. Table III shows sample SNR evolution curves for both actual(upper) and gaussian(lower) cases.

For the convergence analysis of the CID algorithm, the mean μ was obtained from a sufficient soft information block. Based on the symmetry property [11], the variance of soft information is 2μ. With this mean and variance, the SNR evolution curves were plotted. The SNR evolution curves have a wider iteration tunnel as the noise level decreases (i.e., as Eb/No increases) A threshold of a certain decoding algorithm can be obtained when the SNR curves start to touch each other. Fig. 11 shows the SNR evolution curves of the standard ID and CID algorithms. At the threshold (Eb/No=0.53dB) of standard ID, the CID SNR curves have an open iteration tunnel, while the standard ID SNR curves are closed. In other words, the threshold of CID is lower than that of standard ID.

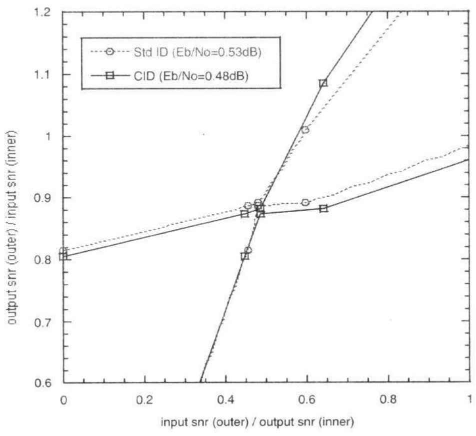


Fig. 12 SNR evolution curves of standard ID and CID on SCCC at 0.53dB and 0.48dB respectively.

Fig. 12 shows that the standard ID and the CID have their thresholds at different noise level, Eb/No=0.48dB and Eb/No=0.53dB respectively. The difference (0.5dB) between two thresholds agrees with the difference of simulation performance (0.4dB at BER 10^{-3}).

V. Conclusion

A modification of the standard iterative decoding algorithm, CID, was introduced in this paper. CID is the generalization of the idea that optimal decoding of a tail-biting convolutional code can be achieved by breaking the cycle using a state variable constraint. The optimal and suboptimal CID algorithm were presented and the performance was compared that of the standard ID. The suboptimal CID was applied to the SCCC and LDPC codes, which have many cycles, and it showed about 0.3-0.5 dB performance gain with higher complexity compared to that of standard ID. Convergence analysis using density evolution techniques showed how the CID improved the performance compared to that of standard ID. Future work is to find a good, big codes with a few cut-points, which is optimally decoded by CID instead of standard ID

REFERENCES

[1] F. Kschischang and B. Frey, "Iterative decoding of compound codes by probability propagation in graphical modes," *IEEE J. Select. Areas Commun.*, pp. 219-231, February 1998

[2] R. J. McEliece, D. J. C. MacKay, and J. J. Cheng, "Turbo decoding as an instance of Pearl's belief progagation algorithm," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 140-152, February 1998

[3] S. Aji and R. McEliece, "The generalized

distributive law," *IEEE Trans. Inform. Theory*, vol. 46, pp. 325-343, March 2000

[4] Y. Wang, R. Ramesh, A. Hassan, and H. Koorapaty, "On map decoding for tail-biting convolutional codes," in *Proc. IEEE symposium on Information Theory*, p.225, June 1997

[5] S. Aji, G. Horn, and R. McEliece, "Iterative decoding on graphs with a single cycle," in *Proc. IEEE symposium on Information Theory*, p.276, August 1998

[6] S. ten Brink, "Convergence of iterative decoding," *Electronics Letters*, vol. 35, pp. 806-808, May 1999

[7] H. E. Gamal and J. A. R. Hammons, "Analyzing the turbo decoder using the gaussian approximation," IEEE Trans. Inform. Theory, vol. 47, pp. 671-686, February 2001

[8] D. Divsalar, S. Dolinar, and F. Pollara, "Iterative turbo decoder analysis based on density evolution," *TMO Progress Reps., Jet Propulsion Lab*, February 2001

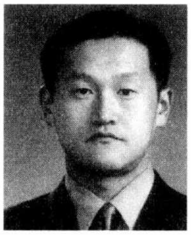
[9] J. B. Anderson and S. M. Hladik, "Tailbiting map decoders," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 297-302, February 1998

[10] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599-618, February 2001

[11] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619-637, February 2001

[12] S. Y. Chung, T. J. Richardson, and R. L. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, pp. 657-670, February 2001

허 준(Jun Heo) 정회원



1989년 2월 : 서울대학교 전자공학과 졸업
1991년 2월 : 서울대학교 전자공학과 석사
2002년 8월 : 미국 USC대학 전기공학과 공학박사
2002년 9월~현재 : 하이닉스 반도체(주) 책임연구원
<주관심 분야> 통신공학, 부호이론