순환 행렬과 eIRA 부호를 이용한 효율적인 LDPC 부호화기 설계

준회원 배 슬 기*, 김 준 성*, 종신회원 송 홍 엽*

Efficient design of LDPC code Using circulant matrix and eIRA code

Seul-Ki Bae*, Joon-Sung Kim* *Associate Members*, Hong-Yeop Song* *Lifelong Member*

요 약

랜덤하게 생성된 LDPC 부호의 경우 부호화기의 복잡도가 크기 때문에 효과적인 부호화를 위하여 구조적인 설계를 필요로 한다. 본 논문에서는 효율적인 부호화기를 위해 기존에 제안된 eIRA 부호에 순환 행렬의 구조를 적용한 부호화기 구조를 제안하였다. 제안된 순환 행렬 구조는 쉬프트 레지스터를 사용하여 부호화기를 구성할 수 있으며, 순환 행렬의 사용으로 인한 성능 저하를 방지하기 위해 치환 행렬 구조에 해당하는 인터리버를 사용하였다. 제안된 부호는 LDPC 부호화기의 복잡도는 낮추면서도 기존의 부호화기의 성능과 유사한 성능을 보인다.

Key Words : LDPC, efficient encoding, circulant matrix, eIRA codes, permutation matrix

ABSTRACT

In this paper, we concentrate on reducing the complexity for efficient encoder. We design structural LDPC code using circulant matrix and permutation matrix and eIRA code. It is possible to design low complex encoder by using shift register and differential encoder and interleaver than general LDPC encoder that use matrix multiplication operation. The code designed by this structure shows similar performance as random code. And the proposed codes can considerably reduce a number of XOR gates.

Ⅰ. 서 론

LDPC 부호(Low Density Parity Check codes)는 패리티 검사 행렬의 원소가 대부분 0인 선형 블록 부호(linear block code)로서 Shannon의 채널 용량의 한계에 근접하는 우수한 부호이다. 1962년 Gallager에 의해 처음 제안되었지만 당시의 기술력으로는 구현이 불가능한 정도의 복잡도로 인해 오랜 기간동안 사용하지 않았으나 1995년 Mackay와 Neal의 재발견 이후 LDPC 부호에 대한 활발한 연구가 이루어지고 있다.

LDPC 부호의 우수한 성능은 간단한 패리티 검사식에 대하여 확률적인 반복 복호 방법에 기인한다. 하지만 우수한 성능에도 불구하고 또 다른 Shannon의 채널 용량에 근접하는 부호 중 하나인 터보부호(Turbo Codes)에 비해 부호화 복잡도가 너무 크다는 단점이 있다. LDPC 부호화 과정은 행렬 곱셈에 의해 이루어지는데 이 때 생성 행렬의 1의 수가 패리티 검사 행렬에 비하여 상당히 많으므로 부호화기의 연산량은 더욱 늘어나게 된다. 그림 1에서와 같이 패리티 검사 행렬을 가우스 소거법으로 제거해 나가면 한쪽은 항등 행렬이 나오고 다른 쪽은 1이 밀집한 행렬이 나오게 된다. 1이 밀집된 행렬이 생성 행렬을 이루는 부분이므로 부호화를 하

※ 본 연구는 삼성 종합 기술원의 "4G wireless system의 연구 개발" 과제의 지원에 의해 이루어졌음
* 연세대학교 전기·전자공학과 부호 및 정보이론 연구실 (sk.bae@coding.yonsei.ac.kr)
논문번호 : KICS2004-12-326, 접수일자 : 2004년 12월 22일

기 위해서 상당한 양의 계산이 필요함을 알 수 있다. 따라서 현재까지도 효율적인 부호화기 구조에 대한 연구는 지속되고 있다. [1-7]

본 논문에서는 효율적인 부호화기 구조의 설계를 위해 eIRA(extended Irregular Repeat-Accumulate) 부호 구조와 순환 행렬(Circulant Matrix)를 이용하여 부호화기의 복잡도를 줄이는 부호를 제안한다. 그리고 $n=10^3$의 길이를 가지고, 높은 부호율을 가지는 LDPC 부호에 초점을 맞추어 일반적으로 우수하다고 알려진 랜덤 LDPC 부호와 비슷한 성능을 가지는 부호를 제안한다.

Ⅱ. 기존의 부호화기 구조

2.1 일반적인 부호화 방법

원 메시지를 m, 생성 행렬을 G라고 할 때, 부호화는 $c=mG$와 같이 행렬 곱으로써 이루어진다. 부호화된 메시지는 $c=[m|p]$와 같이 메시지 m와 패리티 p로 연접하여 구성할 수 있다. 패리티 검사 행렬을 H라고 할 때, $GH^T=0$을 만족해야하며 $cH^T=0$이다. 이렇게 행렬 곱을 사용하여 부호화하는 방법의 단점은 생성 행렬 G가 그림 1과 같이 0의 개수보다 1의 개수가 많은 고밀도 행렬이므로 행렬 곱과 합에 따른 연산량이 많아지게 되므로 효율성이 떨어진다.

2.2 eIRA부호의 패리티 검사 행렬 구조

n은 부호의 길이이고, k는 메시지의 길이라고 할 때 패리티 검사 행렬 H의 크기는 $(n-k)\times n$이다. [1]에서 제안된 패리티 검사 행렬은 아래와 같이 나타낼 수 있다.

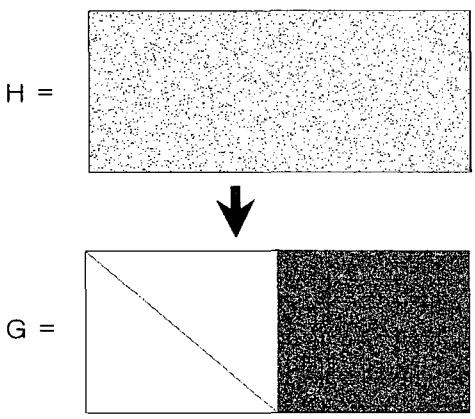

그림 1. LDPC의 패리티 검사 행렬과 생성 행렬

$$H=[H_1 \mid H_2] \tag{1}$$

H_1은 $(n-k)\times k$ 행렬이며 density evolution에 의하여 최적화된 무게 분포에 따라 비균일하게 구성할 수 있다[8, 9]. H_2는 $(n-k)\times(n-k)$ 행렬이며 다음과 같은 구조를 갖도록 구성한다. [1, 3]

$$H_2=\begin{bmatrix} 1 & & & & & & \\ 1 & 1 & & & & & \\ & 1 & 1 & & & & \\ & & 1 & \ddots & & & \\ & & & \ddots & 1 & 1 & \\ & & & & 1 & 1 & \\ & & & & & 1 & 1 \end{bmatrix} \tag{2}$$

H_2는 full-rank를 가지며 역행렬이 존재한다.

2.3 eIRA부호의 생성 행렬 구조

식 (1)과 같이 패리티 검사 행렬이 주어졌을 때 거기에 상응하는 생성 행렬은

$$G=[I \mid H_1^T H_2^{-T}] \tag{3}$$

와 같이 구성된다. 이 때 I는 $k\times k$ 크기의 항등 행렬이며, $GH^T=0$을 만족한다. 그리고 식 (2)의 H_2^{-T}는 다음과 같다.

$$H_2^{-T}=\begin{bmatrix} 1 & 1 & 1 & \cdots & & 1 & 1 \\ & 1 & 1 & \cdots & & \vdots & 1 \\ & & 1 & \cdots & & \vdots & 1 \\ & & & & & & 1 \\ & & & \ddots & \ddots & \vdots & 1 \\ & & & & & & 1 \\ & & & & & 1 & 1 \\ & & & & & & 1 \end{bmatrix} \tag{4}$$

여기서 H_2^{-T} 행렬은 정확히 전달 함수가 $\frac{1}{1\oplus D}$에 해당하는 차등 부호화기(Differential encoder)에 해당하는 변환 행렬이다[1]. 따라서 패리티 비트를 구하기 위해 행렬의 곱셈을 이용하지 않고 차등 부호화를 하므로 LDPC 부호화기의 복잡도를 크게 줄일 수 있다. 식 (3)에 해당하는 LDPC 부호화기는 그림 2와 같다. 항등 행렬에 해당하는 systematic한 부호화 블록과 $H_1^T H_2^{-T}$에 해당하는 패리티 비트를 구하는 블록으로 구성된다. 기존의 LDPC 부호화기에 비하여 매우 간단한 구조를 가지고 있지만 여전히 H_1^T 행렬에 대한 행렬 곱 연산을 필요로 한다는 단점이 있다.

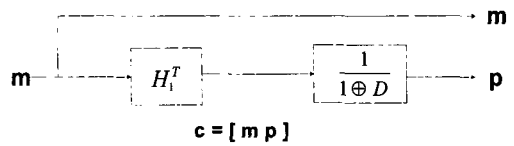

그림 2. eIRA 부호화기의 블록 다이어그램

Ⅲ. 효율적인 부호화기 구조 설계

3.1 제안된 패리티 검사 행렬 구조

본 논문에서는 좀더 효율적인 부호화를 구현하기 위해서 H_1에 순환 행렬을 사용함을 제안한다. H_1 행렬은 다음과 같이 각각의 작은 순환 부행렬 (sub-matrix)[5-7]로 구성할 수 있다.

$$H_1 = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1q} \\ h_{21} & h_{22} & & h_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ h_{p1} & h_{p2} & \cdots & h_{pq} \end{bmatrix} \tag{5}$$

이 때 h_{ij}행렬$(1 \le i \le p,\ 1 \le j \le q)$은 모두 $l \times l$의 같은 크기를 가지며 예를 들면 다음과 같은 순환 행렬 구조를 가지고 있다.

$$h_{ij} = \begin{bmatrix} & 1 & & & 1 \\ 1 & & & 1 & \\ & 1 & & & \ddots \\ & & 1 & & & 1 \\ & & & \ddots & & & 1 \\ 1 & & & 1 & \\ & 1 & & & 1 & \\ & & 1 & & & 1 \end{bmatrix}$$

일반적으로 큰 사이클을 가지며 랜덤하게 생성된 H행렬은 좋은 성능을 가진다. 그러나 제안된 H_1 행렬은 순환 행렬이기 때문에 랜덤한 특성이 사라지며 성능 저하가 생기게 된다. 따라서 H_1 행렬에 랜덤한 특성을 부여하기 위해 행 치환(Permutation)을 사용하였다.

치환 행렬 P는 $(n-k)\times(n-k)$행렬이며 각각의 행과 열에 1이 오직 한개만 존재하는 행렬이다. 따라서 수정된 H_1'행렬은

$$H_1' = PH_1 \tag{6}$$

이 되며, H_1와 같은 크기이며, 치환 행렬에 의해 행의 순서가 바뀐 행렬이다. 이제 부호화 및 복호화를 위해서 H_1 행렬대신 H_1'을 사용하면 수정된 패리티 검사 행렬은 다음과 같다.

$$\begin{aligned} H' &= [H_1' \mid H_2] \\ &= [PH_1 \mid H_2] \end{aligned} \tag{7}$$

3.2 제안된 생성 행렬 구조

앞과 같은 방법으로, 식 (7)에 대응되는 생성행렬은

$$G' = [I \mid H_1'^T H_2^{-T}] \tag{8}$$

로 주어지며, H_1은 순환 행렬로 구성하였으므로 $H_1'^T$ 역시 순환 행렬이다. 여기서 $H_1'^T = H_1^T P^T$로 나타낼 수 있다.

3.3 제안된 부호화기

부호화 과정은 제안된 생성 행렬 G'를 사용하여

$$\begin{aligned} c &= mG' \\ &= m \cdot [I \mid H_1'^T H_2^{-T}] \\ &= [m \mid mH_1'^T H_2^{-T}] \\ &= [m \mid m(H_1^T P^T) H_2^{-T}] \\ &= [m \mid (mH_1^T) P^T H_2^{-T}] \end{aligned} \tag{9}$$

와 같이 이루어진다. 이 때 P^T행렬은 부호화된 mH_1^T 위치를 바꾸는 역할을 한다. P^T는 동일한 역할을 수행하는 인터리버를 이용하여 구성할 수 있다. 예를 들어

$$P^T = \begin{pmatrix} & & 1 & \\ & 1 & & \\ 1 & & & \\ & & 1 & \\ & & & 1 \end{pmatrix}$$

와 같이 주어졌을 때, P^T행렬과 같은 역할을 수행하는 인터리버는 그림 3과 같다.

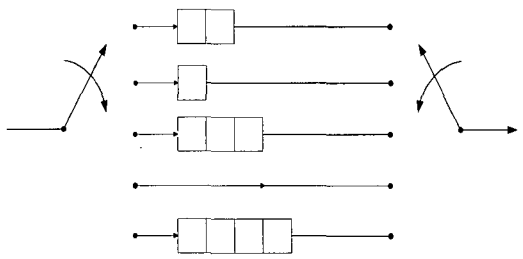

그림 3. 인터리버의 내부 구조의 예

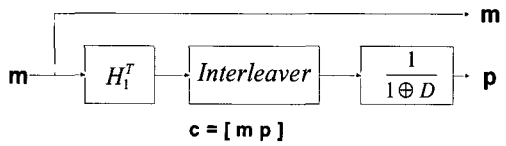

그림 4. 제안된 효율적인 부호화기 블록 다이어그램

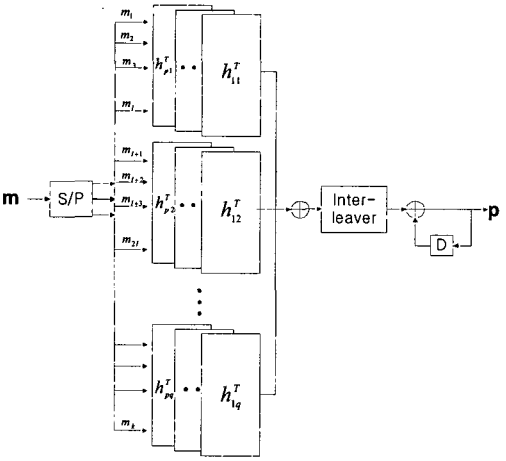

그림 5. 부호화기 내부 구조

이제 변형된 H_1'을 이용해 부호화기를 구현하면 그림 4과 같다. 그림 4은 그림 2에서 제시된 구조에 인터리버가 추가되었다. H_1^T와 인터리버를 통과한 부호는 $mH_1'^T$에 해당한다. 부호화기의 상세 내부 구조는 그림 5와 같다. m은 전송하고자 하는 메시지이고 길이를 k라고 할 때,

$$m = [m_1 m_2 m_3 \cdots m_k] \tag{10}$$

이고, 각각의 메시지는 길이 l만큼 쉬프트 레지스터에 차례대로 할당되어 부호화된다. 예를 들어, 메시지의 길이가 10이고, $H_1 = [h_{11} h_{12}]$이고, h_{11}, h_{12}의 크기가 모두 5이며 순환 행렬의 생성다항식이 각각

$$g_1 = 1 + x^3$$
$$g_2 = x + x^3$$

일 때 쉬프트 레지스터의 구조는 그림 6과 같다. 입력된 메시지는 쉬프트 레지스터의 길이 l만큼 순환되며 부호화된다. 쉬프트 레지스터의 연결선은 순환행렬 h_{11}, h_{12}의 생성 다항식에 의해 결정된다. 쉬프트 레지스터의 각각의 순환 단계에서 이 연결선을 통해 XOR 연산자에 의해 값이 갱신되며, 이렇게 XOR 된 데이터는 차등 부호화되어 전송된다.

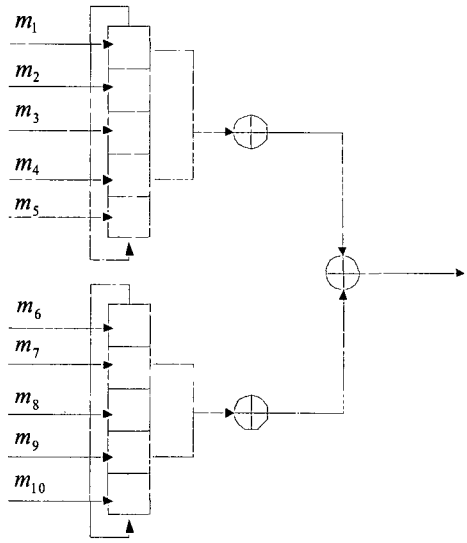

그림 6. 쉬프트 레지스터 구조의 예

3.4 제안한 부호화기의 복잡도

부호화 복잡도의 비교는 표 1에 나타냈었다. 여기서 δ는 H_1 행렬의 1의 밀도이며, δ'은 그림 1에서 생성행렬 G의 1이 밀집한 부분의 밀도이다. k는 메시지의 길이이고, n는 부호의 길이이고, p는 식 (5)에서 행(row)으로 사용된 submatrix의 개수이다. 본 논문에서 제안하는 방식은 H_1의 submatrix의 행의 수를 어떻게 구성하는지에 따라 복잡도가 달라진다.

표 2에서는 표 1에 의한 구체적인 값의 비교를 보였다. 일반적인 행렬 곱 연산으로 검사비트를 구하는 방식의 계산량을 1이라고 했을 때 Yang의 방식과 본 논문에서 제안하는 방식은 1/50 만큼의 적은 계산량을 요구한다. 그리고 본 논문에서 제안하는 방식은 XOR의 수를 줄이기 위해 순환 행렬을 이용해서 구조적인 형태를 이루었기 때문에 동일 정보량을 이용함으로써 XOR의 개수를 줄일 수 있었다. Yang방식의 XOR의 필요 개수를 1이라고 할 때, 본 논문에서 제안하는 부호는 1/256의 비율로 XOR의 필요 개수를 줄일 수 있다.

표 1. 부호화 복잡도 비교

	δ	총계산량	총XOR수	총메모리수
행렬곱	$\delta' \approx 0.5$	δ'k(n-k)	δ'k(n-k)	n-k
Yang	≈ 0.01	(δk+1)(n-k)	δk(n-k)	n-k
FSR	≈ 0.01	(δk+1)(n-k)	δkp	n-k
FSR, 인터리버	≈ 0.01	(δk+1)(n-k)	δkp	$\leq$2(n-k)

표 2. 부호화 복잡도 비교의 예시

	총 계산량 비교	총 XOR 수 비교 n=1024, k=768, p=1
행렬곱	1	•
Yang	1/50	1
FSR	1/50	1/256
FSR, 인터리버	1/50	1/256

Ⅳ. 모의 실험 결과 및 고찰

4.1 실험 환경

실험 환경은 가우시안 채널(AWGN)에서 Sum-Product 복호화 알고리듬을 적용하였다. 부호어의 길이는 1024, 512와 256이며 부호율은 0.75 이고, 사용한 패리티 검사 행렬 H는 다음과 같이 3개의 작은 부행렬을 연접하여 사용하였다.

$$H=[H_1 \mid H_2]=[\, h_{11}\ h_{12}\ h_{13} \mid H_2]$$

각각의 부행렬은 순환 행렬을 이루는 다항식의 무게를 다르게 구성하였다. 실험에서는 h_{11}은 행 무게 3, h_{12}은 행 무게 4, h_{13}은 행 무게 7로 각각 다르게 구성하였다. 위와 같은 조건에서 주어진 H_2 행렬에 대하여 H_1 행렬의 구조를 각각 변경하였을 때 성능 비교 결과를 그림 7, 8, 9에 나타내었다.

4.2 실험 결과 및 고찰

그림 7의 그래프에서 위로부터 첫 번째는 H_1에 일반적으로 사용하는 MacKay의 방식을 따른 랜덤 행렬을 사용하였을 경우이며, 두 번째는 H_1, H_2 두 행렬 모두 MacKay 방식의 랜덤 행렬로 구성한 경우이다. 세 번째는 H_1에 순환 행렬과 인터리버를 사용하였을 경우이며, 네 번째는 H_1에 순환 행렬만으로 구성하였을 경우이다. 본 논문에서 제시한 두 가지 방식은 모두 MacKay의 랜덤 LDPC 부호보다 우수한 성능을 보였다. H_1의 구조가 순환 행렬로만 구성되었을 경우보다 행 치환을 하여 H_1'행렬을 사용한 경우가 더욱 우수한 성능을 보였다. 그러나 첫 번째 제시한 방식은 10^{-4}, 두 번째 제시한 방식은 10^{-5} 이후 오류 마루 현상을 보여 높은 SNR에서는 성능이 점점 열화 되었다.

그림 8과 그림 9는 부호어의 길이가 각각 512와 256인 경우이며 길이 1024와 같은 경향을 보인다. 그리고 본 논문에서 제안한 부호가 10^{-6}까지도 성능은 우수하거나 비슷한 성능을 보였으며 오류마루 현상도 거의 보이지 않았다.

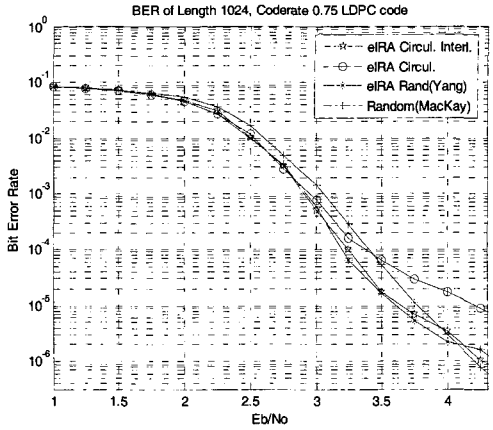

그림 7. 길이 1024, 부호율 0.75인 LDPC 부호의 성능

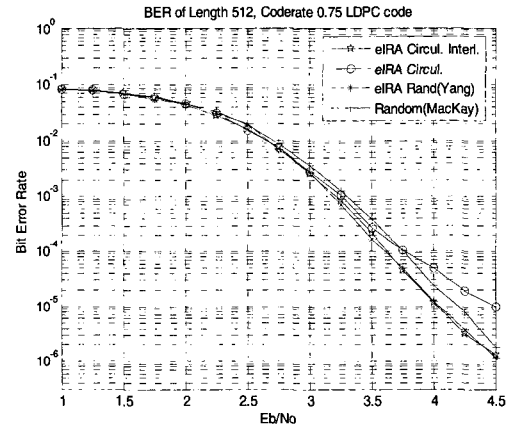

그림 8. 길이 512, 부호율 0.75인 LDPC 부호의 성능

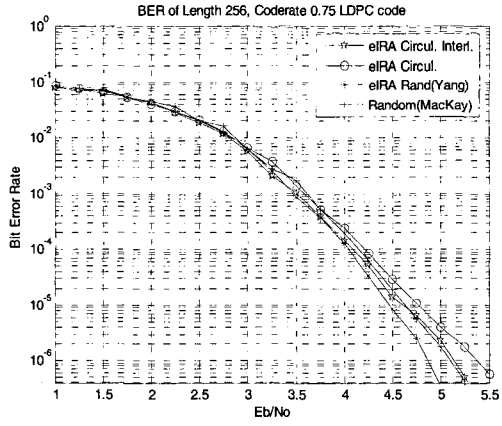

그림 9. 길이 256, 부호율 0.75인 LDPC 부호의 성능

오류 마루 현상이 발생되는 이유는 순환 행렬 특성상 짧은 주기(short cycle)를 피하기 힘들기 때문에 성능 열화가 일어나는 것으로 생각된다. 행 치환하여 랜덤 특성을 이용함으로써 성능저하를 극복할 수 있었다. H_2로 인해 저하된 성능은 잘 설계된 H_1으로써 극복할 수 있다. 짧은 주기의 제거로 큰 girth를 가질 수 있도록 설계하거나 불규칙적인 패리티 검사 행렬을 이용하여 성능을 보다 향상시킬 수 있다.

V. 결 론

본 논문에서는 효율적인 부호화기의 설계를 위하여 패리티 행렬을 순환 행렬을 이용하여 구조적으로 설계하였다. 순환 행렬과 eIRA 부호, 그리고 인터리버를 사용함으로써 행렬 곱을 사용하는 일반적인 LDPC 부호화기보다 복잡도가 작은 부호화기의 설계가 가능하였다. 이 구조로 설계된 부호는 10^{-5} 혹은 10^{-6} 까지는 랜덤 행렬보다 좋은 성능을 내었지만 보다 높은 SNR에서는 오류 마루 현상이 생겨 성능 열화가 생겼다. 이런 오류 마루 현상을 해결하기 위해서는 짧은 주기가 없는 잘 설계된 H_1 행렬에 대한 연구가 필요하다

참 고 문 헌

[1] Michael Yang, "Design of Efficiently Encodable Moderate-Length High-Rate Irregular LDPC Codes," *IEEE Trans. comm*, Vol. 52. pp. 564-571, April 2004.

[2] T. J. Richardson and R. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 638-656, Feb. 2001.

[3] H. Jin, A. Khandekar, and R. McEliece, "Irregular Repeat-Accumulate Codes," in Proc. 2nd. Int. Symp. Turbo Codes and Related Topics, Brest, France, Sept. 2000, pp. 1-8.

[4] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: A rediscovery and new results," *IEEE Trans. Inform. Theory*, vol. 47, pp. 2711-2736, Nov. 2001.

[5] Shu Lin, L. Chen, J. Xu, I. Djurdjevic, "Near Shannon Limit Quasi-Cyclic Low-Density Parity-Check Codes," *IEEE Trans. Inform. Theory*, vol. 52, pp. 1038-1042, July. 2004.

[6] B. Ammar, B. Honary, and Shu Lin, "Construction of Low-Density Parity-Check Codes Based on Balanced Incomplete Block Designs," *IEEE Trans. Inform. Theory*, vol. 50, pp. 1257-1268, June. 2004.

[7] H. Tang, Shu Lin, "On Algebraic Construction of Gallager and Circulant Low- Density Parity-Check Codes," *IEEE Trans. Inform. Theory*, vol. 50, pp. 1269-1279, June. 2004.

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

[9] D. Divsalar, S. Dolinar, and F. Pollara, "Iterative turbo decoder analysis based on density evolution," *IEEE j. Select. Areas Comm.* vol. 19, pp. 891-907, May. 2001.

배 슬 기 (Seul-Ki Bae) 준회원

2003년 2월 연세대학교 전자공학과 학사

2005년 2월 연세대학교 전기·전자공학과 석사

2005년 3월~현재 삼성전자 정보통신 총괄

<관심분야> Error Correcting Codes, LDPC Codes

김 준 성 (Joon-Sung Kim) 준회원

2001년 2월 연세대학교 전파공학과 졸업

2003년 2월 연세대학교 전기전자공학과 석사

2003년 3월~현재 연세대학교 전기전자공학과 박사과정

<관심분야> Error Correcting Codes, LDPC Codes

송 홍 엽(Hong-Yeop Song) 종신회원

1984년 2월 연세대학교 전자공학과 졸업(공학사)
1986년 5월 USC 대학원 전자공학과 졸업(공학석사)
1991년 12월 USC 대학원 전자공학과 졸업(공학박사)
1992년~1993년 Post Doc., USC 전자공학과
1994년~1995년 8월 Qualcomm Inc., 선임연구원
2002년 3월~2003년 2월 University of Waterloo, Canada, 방문연구교수
1995년 9월~현재 연세대학교 전기전자공학부 교수
<관심분야> PN Sequences, Error Correcting Codes, Spread Spectrum Communication Systems, Steam Cipher Systems