병렬 구조를 이용한 Turbo Product Code 성능 분석

정회원 이 태 길*, 정 지 원*

Performance Analysis of Turbo Product Code Using Parallel Structure

Tae-Gil Lee*, Ji-Won Jung* ***Regular Members***

요 약

최근 터보 부호에 비해서 구현시 복잡하지 않고, 높은 부호화율에서 거의 샤논 이론에 접근하는 Turbo Product Code에 대해 관심이 고조되고 있다. 본 논문에서는 기존의 Turbo Product code 복호기에서 row과 column을 직렬로 복호를 하지 않고 복호 구조가 병렬로 동작하는 Turbo Product code 복호기를 제안한다. 모의 실험을 한 결과 기존의 방식에 비해 복호 지연이 줄어들고 성능면에서 직렬 방식과 거의 비슷한 성능이 나타난다.

Key Words : Turbo Product Code; Shannon limit; parallel decoding.

ABSTRACT

Recently, there has been intensive focus on Turbo Product Codes(TPCs) which have low decoding complexity and achieve near-optimum performances at high code-rate. This paper present a parallel algorithm of turbo product codes enable simultaneous decoding of row and column. The row and column decoders operate in parallel and update each other after row and column has been decoded. simulation results show that the performance of proposed parallel turbo code is almost the same as that conventional scheme for several turbo product codes.

Ⅰ. 서 론

최근의 무선 통신 시스템은 무선 멀티미디어 전송에 기반을 두고 있기 때문에, 고속 데이터 전송에 효율적이고 성능이 우수한 복호기 개발이 필수적이다. 1993년에 Berrou등에 의해 발표된 Turbo 부호[1]는 Shannon's Limit에 근접한 성능을 나타내지만 많은 데이터양과 연산작용이 매우 복잡해 저속 서비스에만적용된다. 최근의 오류정정분야의 또 다른 연구는 LDPC(Low Density Parity Check) 부호에 관심이 집중되고 있다. 그러나 LDPC는 복호화가 간단한 반면에 부호화 부분의 높은 복잡도가 LDPC 부호의 최대의 단점이다. 1998년 Pyndiah에 의해 소개된 TPC(Turbo Product Code)[2]는 기존의 LDPC 부호의 단점인 부호화 시 구성 어려움, 그리고 성능 향상을 위한 많은 블록 크기를 요구한다는 것과 Turbo 부호의 많은 계산량과 고속 복호기 구성의 어려움 등의 단점을 보완한 작은 블록 크기를 가로 세로로 product 시킨 후 같은 복잡도로서 많은 블록 크기의 효과를 얻을 수 있고 복호기가 간단하여 고속 구현이 가능하며, 높은 부호화율에서 Shannon Limit에 근접하는 새로운 차세대 오류정정 부호화 방식으로 무선 멀티미디어 통신을 요구하는 최근의 무선 통신시스템에 오류정정방식으로 적합하다. 그러나 TPC 복호기의 가장 큰 문제점은 두개

* 한국해양대학교 전파공학과 위성통신 연구실
※ 본 연구는 한국과학재단의 지역대학 우수과학자 지원 연구사업(R05-2003-000-10701-0)의 지원으로 수행되었습니다.
논문번호 : 030310-0722, 접수일자 : 2003년 7월 22일

복호기가 직렬로 연결된 구조이기 때문에 두개의 복호기가 직렬로 연산 반복하는 과정에서 지연으로 인한 속도 저하를 가져올 수 있다. 따라서 본 논문에서는 병렬 구조로 구성된 TPC 복호기의 구조를 제안한다.

Ⅱ. 기존의 Turbo Product Code

TPC 부호는 두개 혹은 그 이상의 짧은 길이의 블록부호 (C_1, C_2)를 이용하여 긴 블록 부호 ($P= C_1 \times C_2$)를 만드는 것이 가장 효율적이고 간단한 부호화 알고리즘이다. 그림 1은 TPC 부호화기 구성도를 나타낸다[2].

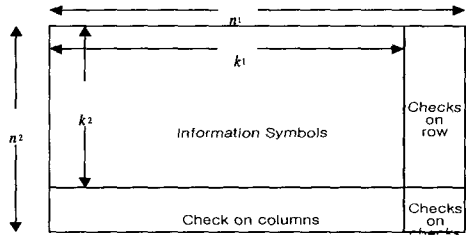


그림 1. TPC 부호화기 구성도 ($P= C_1 \times C_2$)

그림 1과 같이 두 개의 블록 부호를 적용할 경우, k_1(또는 k_2)개의 정보 비트를 가로(또는 세로)로 배치한 후, 가로는(n_1, k_1, δ_1)을 가지는 블록 코드 C_1으로 부호화 시키고, 세로는 (n_2, k_2, δ_2)를 가지는 C_2로 부호화시켜 전송한다. 따라서 TPC부호 $P= C_1 \times C_2$ 이므로, (n, k, δ)를 가진다. 여기서 $n= n_1 \times n_2$, $k= k_1 \times k_2$, $\delta= \delta_1 \times \delta_2$ 이고 부호화율은 $R= R_1 \times R_2$ ($R_1= k_1/ n_1$, $R_2= k_2/ n_2$) 이다. 따라서 두 부호어를 product 함으로써, 높은 부호화율에서 minimum hamming distance의 증가에 의해서 오류 정정 능력은 향상된다. 그리고 오류를 산발시키는 효과가 있는 인터리버가 필요치 않으며, 복호시 가로 부분을 먼저 복호 한 후 이를 extrinsic 정보로 이용하여 세로 부분을 복호하면서 반복 복호를 한다. TPC에 적용되는 블록 부호 C_1, C_2는 해밍부호, BCH부호, RS부호등 다양한 블록 부호를 적용시킬 수 있다. Pyndiah가 제안한 TPC복호기 구조는 아래 그림 2와 같다.

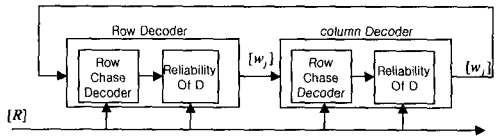


그림 2. TPC 복호기 구조

원 신호 그리고 가우시안 잡음 신호 ..에 의해 수신신호 벡터 다음식과 같이 나타낼 수 있다.

$$R= E+N$$
$$R= (r_1, \cdots, r_l, \cdots r_n)$$
$$E= (e_1, \cdots, e_l, \cdots e_n).$$
$$N= (n_1, \cdots, n_l, \cdots n_n) \text{ 이다.}$$

최적 결정 비트 $D= (d_1, \cdots, d_l, \cdots d_n)$는 식 (1)과 같이 maximum likelihood 방식에 의해 결정 된다.

$$D= C^i \text{ if } \Pr\{E= C^i|R\} > \Pr\{E= C^j|R\} \forall j \neq i$$
$$D= C^i \text{ if } |R- C^i|^2 > |R- C^j|^2 \forall j \neq i$$
$$|R- C^i|^2 = \sum_{l=1}^{n} (r_i - c_i^l)^2 \qquad (1)$$

$C^i= (c_1^i, c_2^i, \cdots, c_l^i, \cdots c_n^i)$는 가능한 모든 부호어의 집합 C의 i번째 부호어 이다. 이 경우 값이 크면 계산량이 매우 많고 오래 걸리며 거의 불가능하다. 따라서 1972년 low complexity 알고리즘을 제안하였으며, 이는 높은 SNR에서 D는 Y의 중심점에서 반경이 $\delta-1$의 구안에 포함 될 확률이 높다. 여기서 $Y= (y_1, \cdots, y_l, \cdots y_n)$이며,

$y_i= 0.5(1+ sgn(r_i))$이다. 후보 가능한 부호어 C를 찾는 chase 알고리즘은 다음과 같다[3].

1단계: $p= \lfloor \delta/2 \rfloor$ 개의 신뢰성 없는 Y의 비트 위치를 수신 벡터 R을 이용해서 결정한다. 신뢰성 없는 비트의 위치는 수신되는

$$\Lambda(y_j)=\ln\left(\frac{\Pr(e_j=+1/r_j)}{\Pr(e_j=+1/r_j)}\right)$$

$$=\left(-\frac{2}{\delta^2}\right)r_j=|r_j| \quad \text{이다.}$$

2단계: q개의 test pattern T^q를 생성한다. T^q의 생성 방법은 개의 비트 위치 중 $\Lambda(y_j)$가 가장 적은 값에 해당하는 위치 .에 "1"을 위치시키고 나머지 비트위치는 "0"을 삽입하고, $\Lambda(y_j)$가 가장 적은 두개의 비트 위치에 "1"을 위치시키고 나머지 비트는 "0"으로 배치한다. 같은 방법 계속해서 $\Lambda(y_j)$가 가장 작은 개의 비트 위치에 "1"을 위치시키고 나머지 비트는 "0"을 삽입한다.

3단계 : q개의 T^q를 생성하고 난 뒤에, $Z^q=Y\oplus T^q$하여 오류 위치를 정정한 Z^q를 생성한다.

4단계: Z^q를 블록 복호하여 C^q를 생성한다.

4단계가 완료되면 검토 되어지는 $C^q=(C^1, C^2, C^3,\cdots, C^n)$벡터가 생성 되며, 그림 2의 부호기에서 연판정을 출력하기 위해 D의 j 번째 비트 신뢰도 $(LLR)_j$ 는 아래 식 (2)와 같다.

$$(LLR)_j=\ln\left(\frac{\Pr(e_j=+1/R)}{\Pr(e_j=+1/R)}\right) \qquad (2)$$

$$\Pr(e_j=+1/R)=\sum_{i\in S_j^{+1}}\Pr(E=C^i/R)$$

이며, S_j^{+1}는 $c_j^i=+1$을 가지는 codeword의 인덱스 $\{C^j\}$의 집합이다. $\Pr(e_j=-1/R)=\sum_{i\in S_j^{-1}}\Pr(E=C^i/R)$이며 S_j^{-1}는 $c_j^i=-1$을 가지는 codeword의 인덱스 $\{C^j\}$의 집합이다. Bayes 이론을 적용시키고 서로 다른 부호어들이 균등하게 분포되어 있다고 가정하면 d_j에 관한 LLR은 식 (3)과 같다.

$$(LLR)_j=\ln\left(\frac{\sum_{i\in S_j^{+1}}\exp-\frac{|R-C^j|^2}{2\delta^2}}{\sum_{i\in S_j^{-1}}\exp-\frac{|R-C^j|^2}{2\delta^2}}\right) \qquad (3)$$

식 (3)은 높은 부호율을 가지는 부호어에 대해서는 거의 계산 불가능하고 복잡하므로 Pyndiah 에 의해 식 (4)와 같이 간략하게 최적화 하였다.

$$(LLR)_j\approx\frac{2}{\delta^2}\left(r_j+\sum_{l=1,l\neq j}^{n}r_l\,c_l^{\min(+1)}\,P_l\right)$$

$$P_l=\begin{cases}0 & if\ \ c_l^{\min(+1)}=c_l^{\min(-1)}\\ 1 & if\ \ c_l^{\min(+1)}\neq c_l^{\min(-1)}\end{cases} \qquad (4)$$

$c_l^{\min(+1)}$과 $c_l^{\min(-1)}$은 . 번째 비트가 +1, -1을 가지고 부호어 중 수신신호와 해밍거리가 가장 작은 부호어의 . 번째 비트를 나타낸다. 따라서 식(4)는 식(5)와 같이 나타낼 수 있다.

$$r_j'=r_j+w_j$$

$$w_j=\sum_{l=1,l\neq j}^{n}r_l\,c_l^{\min(+1)}\,P_l \qquad (5)$$

r_j : soft input data ,

ω_j : extrinsic information

이는 과 r_j $(l\neq j)$의 최소 유클리언 거리의 함수 이다. 식 (5)에서 반복 시 생성되는 r_j의 soft decision 값 r_j'는 입력 수신벡터의 r_j와 extrinsic 정보의 합으로서 표시될 수 있다. extrinsic 정보 ω_j는 자기 신호 .번째를 제외한 r_j와 선택된 부호어 C_j의 곱의 합으로 표시 될 수 있다. 즉 r_j'는 chase 알고리즘에 의해 복호된 의 soft decision 값이다. r_j'는 $r_j'=\beta d^j$ $(\beta\geq 0)$ 와 같이 나타낼 수 있으며, β는 신뢰도 factor 이다. 처음 반복 시에는 신뢰도가 낮으므로 낮은 값으로 반영하면서 반복횟수가 증가 할수록 높게 설정한다.

product code를 가로, 세로 복호 시 TPC의 복호 블록도는 아래 그림 3과 같다.

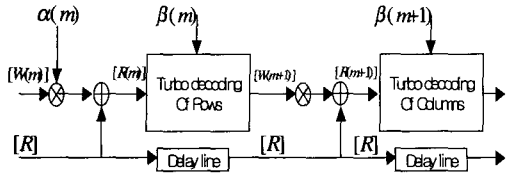


그림 3. Pyndiah가 제안한 TPC 복호기 구조

$$R[2]=[R]+\alpha[2]\cdot[W[2]]$$
$$W[2]=R[2]-R[1]$$

α는 스케일링(scaling) factor 이며, 이는 수신신호 R과 W에 있는 샘플들의 표준편차를 고려한 것이다. 따라서 대부분의 논문에서도 $\alpha[m]=[0, 0.2, 0.3, 0.5, 0.7, 0.9, 1.0, 1.0]$, $\beta[m]=[0.2, 0.4, 0.6, 0.8, 1.0, 1.0, 1.0, 1.0]$ 으로 할당한다. 다음 그림4는 위와 같은 방식으로 BCH(63.57.3)code 두 개를 사용하여 TPC를 구성하여 각 반복 횟수 따른 성능을 나타낸 것이다.

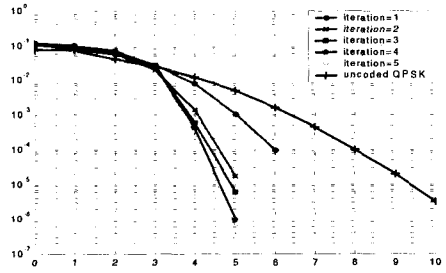


그림 4. BCH(63.57.3)TPC의 반복횟수에 따른 성능

Ⅲ. 본 논문에서 제안하는 병렬 TPC구조

앞에서 살펴본 그림 3과 같이 Pyndiah에 의해 제안된 복호기는 세로(또는 가로)를 다음 복호하기 전에 반드시 가로(또는 세로)를 복호 하여야 하는데 반해, 지연을 감소 시키기 위한 본 논문에서 제안된 병렬 TPC복호기 구조는 그림 5와 같다. 그림 5에서와 같이 가로 세로 복호기에 대한 복호는 병렬로 이루어져지며, 복호 지연은 기존의 방식에 비해 절반 효과를 갖기 때문에 복호 속도는 2배 개선 시킨다. 메트릭 $[W^{row}]$와 $[W^{col}]$은 가로와 세로 부호어에 대한 수신 신호의 extrinsic 정보 이며, 각 블록에 대한 복호를 마치고 block-by-block으로 같은 시간에 extrinsic 정보의 교환으로 복호가 수행되어 진다. 첫 번째 반복 시, 즉 m=0, 일 시 $[R^{row}(0)]=[R^{col}(0)]=[R]$ 이라 두고, 가로 세로의 블록 길이는 이라 가정한다.

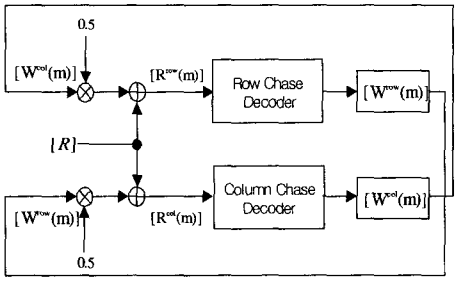


그림 5. 병렬 복호기 구조 및 복호 과정

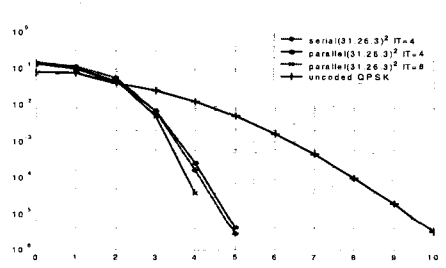
가로 세로에 대한 복호가 마친후 업데이터된 신호 벡터 $[R^{row}(m+1)]$, $[R^{col}(m+1)]$은 다음 반복을 위한 입력으로 전달 된다. 병렬적으로 업데이터 된 신호 벡터는 아래 식 (6)과 같다.

$$[R^{row}(m+1)]=[R]+\alpha(m)\cdot[W^{col}(m)]$$
$$[R^{col}(m+1)]=[R]+\alpha(m)\cdot[W^{row}(m)] \quad (6)$$

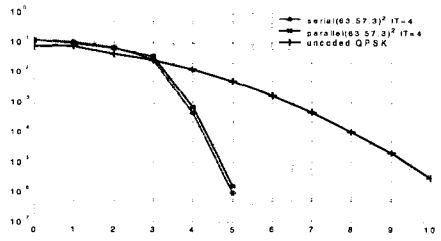
Ⅳ. 모의 실험 결과

본 논문에서 모의 실험 시 BCH 부호를 사용하여 구성 하였고 부호화율이 같은 두개의 BCH부호를 사용하였다. Chase 알고리즘을 위해서 가장 신뢰성이 없는 비트를 선택시 $p=4$로 하였다. 그리고 반복 복호를 위한 scaling factor로서 $\alpha=0.5$, $\beta=1$로 고정하였다. β는 모든 반복시 1로 고정하였기 때문에 Chase 복호기의 출력 decoder의 연판정 값을 구하기 위한 β의 곱셈이 필요치 않으며 α는 0.5로 고정했기 때문에 가로 세로에 대한 복호기의 extrinsic 정보의 절반에 해당하는 bit 만큼 우측으로 shift 하면 되기 때문에 H/W구현이 간단해 진다. 이러한 조건을 중심으로 반복 횟수를 4로

하였을 때 결과는 그림6과 같다. BER=10-4을 기준으로 Pyndiah가 제안한 방식과 비교하였을 때 본 논문에서 제안한 병렬 복호 방식은 0.2dB 이내의 성능감소가 발생된다. 반복횟수가 동일 할 때 기존의 방식과 비교하여 성능의 감소가 일어 나는 것은 첫 번째 반복시 기존의 방식은 row 복호이후 발생된 extrinsic 정보가 column 복호에 제공되지만 본 논문에서 제안한 방식은 column 복호에 제공되는 extrinsic 정보가 0이기 때문이다. 반복 횟수 8번을 같는 parallel(31.26. 3) 에서 반복횟수 4를 같는 serial (31.26.3)와 비교시 각각은 같은 지연을 가지지만 병렬방식이 성능이 0.4dB정도 좋아짐을 알 수 있다.



(a)BCH(31,26,3)TPC



(b)BCH(63,57,3)TPC

그림 6. 기존방식과 병렬 복호방식의 성능비교

V. 결 론

본 논문에서는 고속 통신을 실현 하기 위해서 병렬 TPC 복호기를 제안하였다. 기존의 TPC복호기는 row복호 후 column 복호를 하기 때문에 지연으로 인한 복호 속도 저하가 생긴다. 그러나 병렬로 동작하는 복호기는 row와 column이 부분에서 동시에 복호하고 동시에 extrinsic 정보를 넘겨주기 때문에 기존의 방식보다 지연 시간이 작다. BER=10^{-4} 을 중심으로 살펴 볼 때 기존의 방식에서 비교 할 때 0.2dB 이내의 성능감소가 일어남을 알 수 있다. 구현 상의 메모리의 요구량은 늘어나지만 기존 방식의 한번 반복을 기준으로 볼 때 같은 수의 메모리가 사용되고 성능이 0.4dB정도 향상됨을 알 수 있다.

참 고 문 헌

[1] C. Berrou, A. Glavieux, and P.Thitimajshima, "Near Shannon Limit Error-Correcting Code and Decoding : Turbo Codes", in Proc. Of ICC'93, 1993.

[2] R.M. Pyndiah, "Near-optimum decoding of product codes: Block turbo codes," IEEE Trans. Commun., vol. 46, pp1003-1010, Aug. 1998.

[3] D.Chase, "A class of algorithms for decoding block codes with channel measurement information," IEEE Trans.Inform. Theory, vol. IT-18,pp.170-182, Jan.1972

[4] Cenk Argon, "A parallel decoder for low latency decoding of turbo product codes," IEEE commun, LETTER, vol. 6, No 2, Feb, 2002.

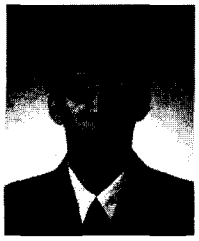
이 태 길 (Tae-Gil Lee) 정회원

2002년 2월 : 한국해양대학교 전파공학과 학사

2002년 3월~ 현재 : 한국해양대학교 전파공학과 석사

<관심분야> 변.복조기술, 채널코딩, FPGA 기술 등

정 지 원 (Ji-Won Jung) 정회원



1989년 2월 : 성균관대학교 전자공학과(공학사)
1991년 2월 : 성균관대학교 전자공학과(공학석사)
1995년 2월 : 성균관대학교 정보공학과(공학박사)
1991년 1월~ 1992년 2월 : LG 정보통신연구소 연구원
1995년 9월~ 1996년 8월 : 한국통신 위성통신연구실 선임연구원
1997년 3월~ 1998년12월 : 한국전자통신연구원 초빙 연구원
1996년 9월~현재 : 한국해양대학교 전파공학과 부교수
2001.8 ~ 2002.8 : 캐나다 NSERC Fellowship (Communication Research Center 근무)
<관심분야> 위성통신, 이동통신, 변.복조기술, 채널코딩, FPGA 기술 등