논문 2011-6-8

GE 삼각화를 이용한 효율적인 LT 복호 기법 연구

A Study on the Efficient LT Decoding Scheme using GE Triangularization

정호영*

Ho-Young Cheong

요 약 본 논문에서는 GE 삼각 화를 이용해 LT 부호의 복호 과정을 수행함으로써 복잡도와 오버헤드 성능을 모두 개선한 효율적인 복호 방식을 제안하였다. BP 복호 방식은 간단하고 빠르기는 하나 부호 블록이 짧을수록 복호하는데 큰 오버헤드가 필요하다는 단점이 있고, OFG 알고리즘은 오버헤드는 작으나 연산 양이 많다. 시뮬레이션 결과 제안한 복호 방식은 OFG 알고리즘에 비해 연산 양이 5배 이상 감소되었으며 오버헤드는 1～5%의 적은 양을 보였다.

Abstract In this paper an efficient LT decoding scheme using GE triangularization is proposed. The proposed algorithm has the desirable performance in terms of both overhead and computational complexity. Belief propagation algorithm is a fast and simple decoding scheme for LT codes. However, for a small code block length k, it requires a large overhead to decode, and OFG which has a small overhead has a large computational complexity. Simulation results show that the proposed algorithm noticeably reduces the computational complexity by more than 1/5 with respect to that of OFG and also its overhead has a small value about 1～5%.

Key Words : Triangularization, On-the-fly Gaussian Elimination, Belief Propagation, Overhead, Computational complexity.

Ⅰ. 서 론

최근 erasure 채널을 통해 오류 없이 데이터를 전송하기 위한 FEC 부호로서 LT 부호와 Raptor 부호가 많은 관심을 받고 있다[1]. BEC(binary erasure channel)는 erasure 채널의 대표적인 형태로 패킷 스위칭을 사용하는 유선 인터넷 망이 대표적인 예이다[2]. 인터넷 망을 통해 패킷이 전송되는 경우 패킷 내에 오류가 발생하기 보다는 전송되는 과정에서 패킷이 유실되어(packet loss) 없어지는 경우가 대부분이므로 유선 인터넷 망은 BEC로 모델링하는 것이 적합하다고 할 수 있다. 이러한 BEC를 위해 설계된 부호가 Fountain 부호인데 Fountain 부호의 일종인 LT 부호는 BEC에서 채널 용량에 근접하는 성능을 보이는 것으로 보고가 되고 있다[1].

LT 부호는 전송해야 할 k 개의 패킷들로 구성된 부호 블록에서 부호 블록 내의 패킷들을 조합하여 끊임없이 부호 패킷들을 발생시켜 전송하게 되며 수신 단에서는 패킷의 전송 순서에 상관없이 원래의 메시지 패킷이 완전히 복원될 때까지 부호화된 패킷을 랜덤하게 수신하여 복호하게 된다.

LT 부호의 복호는 BP(belief propagation) 알고리즘이 대표적인데 다른 복호 방식에 비해 복호 복잡도가 가장 낮은 것으로 알려져 있다. 그러나 전송하고자 하는 부호 블록의 길이 k가 크지 않을 경우 오버 헤드(overhead)가 지나치게 증가하는 단점을 가지고 있다. 반면 GE (Gaussian elimination) 복호 방식은 복호 속도가 BP에 비해 느리고 연산 량이 많기는 하지만 오버헤드 비중이

*정회원, 남서울대학교 정보통신공학과(교신저자)
접수일자 2011.10.8, 수정완료 2011.11.16
게재확정일자 2011.12.16

거의 없다는 장점을 가지고 있다. 따라서 부호 블록의 길이 k가 크지 않을 경우 오버 헤드 비중을 고려하면 BP 복호 기법 보다 GE 복호 기법을 사용하는 것이 바람직할 경우가 많다[3].

BP와 GE 복호 방식은 두 방식 모두 패킷의 전송 과정 끝 부분에서 대부분의 복호 연산이 수행되므로 복호 처리 연산이 마지막 패킷들이 도착하는 시간 이후에 집중되는 특성을 갖는다. 최근 V. Bioglio 등에 의해 제안된 OFG(on the fly Gaussian elimination) 복호 방식은 GE 복호 방식의 일종으로 볼 수 있는데, GE의 복호 연산과정을 패킷들이 도착하는 모든 시간대로 분산시켜 복호를 수행함으로서 마지막 패킷이 도착한 후 짧은 시간 내에 복호 과정을 종료할 수 있기 때문에 전체적인 복호 시간을 보면 BP 복호 알고리즘이나 GE 복호 알고리즘 보다 더 짧은 경우가 많은 것으로 보고하고 있다[2]. OFG 복호 방식은 GE 복호 방식에 기반을 두고 있기 때문에 BP에 비해 오버 헤드 비중이 훨씬 작은 한편 복호 복잡도는 GE 방식과 거의 동일한 특징을 갖는다. OFG 복호 방식에서는 복호 연산 양을 줄이기 위해 xoring 연산과 swap heuristic 과정을 두고 있는데 이는 삼각 화(triangularization)가 진행되는 동안 행렬의 행 차수(row degree)를 줄여나가는 과정이라고 할 수 있으며 이를 통해 상측 삼각 행렬(upper triangle)에서 1의 밀도를 작게 하여 연산 양을 줄이는 효과를 이용하고 있다. 특별히 행 차수가 감소할수록 행의 차수가 1인 경우가 발생할 확률이 높아지며 본 논문에서는 이러한 특성을 이용해 BP 복호 과정에 적용하여 오버헤드와 연산 양 모두를 줄이고자 하였다.

본 논문에서는 오버헤드 비중이 GE 복호 방식과 거의 비슷하면서도 복호 연산 양을 크게 줄인 LT 부호의 복호 기법을 제안한다. OFG 복호 방식의 삼각 화 과정에서 적용되는 xoring 연산을 통해 발생되는 차수-1의 행을 BP 복호 과정에 이용함으로서 BP 복호 기법이 가지고 있는 적은 연산 양과 GE의 삼각 화가 가지고 있는 적은 오버헤드 특징을 모두 가질 수 있음을 보인다.

본 논문의 구성은 다음과 같다. 제 2 장에서는 LT 부호의 복호 방식 중 BP 알고리즘의 단점인 오버 헤드 발생 원인을 살펴보고, 제 3 장에서는 본 논문에서 제안한 xoring 연산을 이용한 GE 삼각 화 BP 복호 알고리즘을 자세히 기술한다. 제 4 장에서 이들에 대해 시뮬레이션을 통해 오버헤드 비중과 복호 연산 양을 중심으로 한 실험 결과를 비교·분석하고 제 5 장에서 결론을 맺는다.

II. BP 복호 방식의 오버헤드 발생 특성

1. LT 부호의 부/복호 과정

LT 부호는 임의의 k 개 메시지 소스 패킷 $m = [m_0, m_1, \ldots, m_{k-1}]$의 원소들을 조합하여 부호 패킷 $y = [y_1, y_2, \ldots]$을 끊임없이 만든다. k 개의 메시지 패킷들을 이용해 수행하는 LT 부호의 부호화 과정은 다음과 같다.

(제 1 단계) d 개의 랜덤 정수 $r_0, r_1, \ldots, r_{d-1}$을 RSD(robust soliton distribution) 분포를 이용해 구간 $[0, k-1]$에서 발생시킨다.

(제 2 단계) k 개의 메시지 패킷 중에서 임의로 d 개를 선택한 후 xor 연산을 수행하여 부호화된 패킷 y_i를 식 (1)과 같이 얻는다.

$$y_i = m_{r_0} \oplus m_{r_1} \oplus \ldots \oplus m_{r_{d-1}} \quad (1)$$

여기에서 m_{r_j}는 xor 연산에 사용된 r_j 번째 소스 패킷을 의미한다.

(제 3 단계) y_i의 xor 연산에 사용된 소스 패킷 정보를 담고 있는 정보 벡터 $b_i = [b_{i1}, b_{i2}, \ldots, b_{ik}]$와 함께 y_i를 전송하게 된다. 여기에서 m_j가 y_i를 생성하기 위해 xor 연산에 사용되었을 경우 $b_{ij} = 1$ 값을 갖고 사용되지 않았을 경우 $b_{ij} = 0$의 값을 갖게 된다.

수신단의 복호기가 n 개의 부호화된 패킷 $\boldsymbol{y} = [y_1, y_2, \ldots, y_n]^T$과 해당 위치 정보 벡터 $\boldsymbol{B} = [b_1, b_2, \ldots, b_n]^T$ 를 수신하게 되면 시스템 $\boldsymbol{Bm} = \boldsymbol{y}$를 풀어 소스 패킷 $\boldsymbol{m}$을 복호하게 된다. 여기에서 시스템 $\boldsymbol{Bm} = \boldsymbol{y}$로부터 잉여 등식(linearly dependent equation)들을 제거하면 $\boldsymbol{B}$와 $\boldsymbol{y}$로부터 각 각 $(k \times k)$ 행렬 $\boldsymbol{G}$와 $(k \times 1)$ 벡터 $\boldsymbol{Y}$를 얻을 수 있고 $\boldsymbol{Gm} = \boldsymbol{Y}$를 풀어 m을 복호할 수 있다.

기본적으로 $\boldsymbol{Gm} = \boldsymbol{Y}$의 해(solution)는 GE (Gaussian elimination)을 통해 구할 수 있으나 GE를 이용할 경우 k 값에 따라 연산양이 급격히 증가하여 k 값이 클 경우 적용하기는 어렵다[4]. LT 부호는 벡터 b_i의 차수가 RSD 특성을 갖도록 적절히 조절하여 수신 단에

서 GE 복호 알고리즘 대신 BP 복호 알고리즘이 적용될 수 있도록 한 부호이다. BP 복호 알고리즘은 가장 간단하면서도 빠른 복호 방식이다.

2. BP(belief propagation) 복호의 오버헤드 특성

$\boldsymbol{B}$ 행렬의 i 번째 행을 $\boldsymbol{B}[i]$라고 표시하기로 하자. BP 복호 알고리즘에서는 차수(degree)가 1인 행 $\boldsymbol{B}[i]$에서 '1'의 열(column) 위치가 j라고 하면 $m_j = y_i$로 복호하고 j 번째 열에 있는 모든 '1'들을 '0'으로 바꿈과 동시에 '0'으로 바뀌는 행에 해당하는 부호 패킷 y_l을 y_i와 xor 연산을 하게 된다. 다시 차수가 1인 행을 찾아 위와 같은 복호 과정을 $\boldsymbol{B}$ 행렬의 원소가 모두 0이 될 때까지 반복하는데 $\boldsymbol{B}$ 행렬의 원소가 모두 0이 되면 복호가 성공한 것이고 모두 0이 되지 않았는데도 불구하고 차수가 1인 행이 없으면 복호 과정이 중단되어 실패하게 된다. 차수가 1인 행이 없어 복호 과정이 중단되면 새로운 패킷을 수신한 후 복호된 위치의 '1'을 모두 '0'으로 변환하는 과정을 먼저 수행하고 B 행렬에 추가하여 행렬의 모든 원소가 '0'이 될 때까지 복호과정을 계속한다.

일반적으로 BP 복호 과정에서 k 개의 부호 패킷들이 도착하자마자 복호 과정이 성공하는 경우는 거의 없으며 새로운 패킷들을 수신하면서 차수-1의 패킷을 탐색하게 된다. 차수-1의 패킷이 수신되면 BP 복호 과정이 반복되며 다시 차수-1의 패킷이 존재하지 않으면 끊임없이 새로운 패킷을 수신하여 복호를 시도하게 된다. 그림 1은 $c=0.1$, $k=1000$, $\delta=0.01$인 LT 부호에 대해 BEC에서 BP 복호를 이용해 복호할 경우 복호 종료 후 사용되지 않은 차수-1의 부호 패킷 수를 나타낸 것이다. 그림에서 BP 복호가 종료된 후 사용되지 않고 남아 있는 수신 부호 패킷은 296개인데 복호된 패킷 수의 약 30%에 해당하는 큰 값으로 오버헤드 측면에서 보았을 때 대단히 비효율 적임을 알 수 있다.

그림 2는 BP 복호가 종료된 후에 남아 있는 차수-1의 미사용 수신 부호 패킷 수를 k 값에 따라 나타낸 것이다. 각 경우에 대해 복호를 500회 이상 수행하였으며 이를 통해 얻은 값들에 대해 평균을 취하여 얻은 결과이다. 그림 2에서 $c=0.1$일 때 효율이 크게 저하됨을 알 수 있으며 k 값이 증가할수록 사용되지 않고 남아 있는 패킷 수가 크게 증가함을 알 수 있다. 이것은 중간에 BP 복호 과정이 실패했을 경우 차수가 1인 새로운 패킷이 도착할 때까지 계속하여 패킷을 수신하여야 하는 BP 복호의 특성에 기인한다.

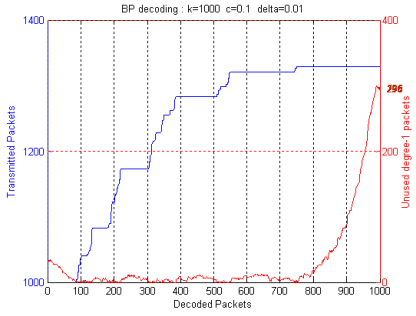

그림 1. BP 복호 종료 후 사용되지 않은 차수-1의 부호 패킷 수 ($c=0.1$, $\delta=0.01$)

Fig. 1. The number of unused packets of degree 1 after decoding($c=0.1$, $\delta=0.01$)

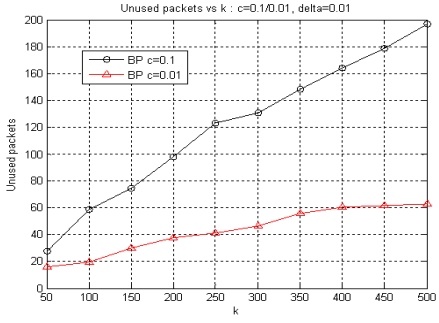

그림 2. k 값에 따른 BP 복호의 차수-1의 미사용 수신 패킷 수 ($c=0.1$, $c=0.01$)

Fig. 2. Unused packets of degree 1 versus k ($c=0.1$, $c=0.01$)

Ⅲ. GE 삼각 화의 xoring 연산 효과를 이용한 BP 복호

[2]에서 제안된 OFG 알고리즘은 GE 과정에 기반을 두고 있지만 k 개의 부호 패킷이 도착할 때까지 기다리지 않고, 처음 도착하는 패킷부터 시작하여 매 번 패킷들이 도착할 때 마다 xoring 연산을 이용하여 행렬 $\boldsymbol{G}$의 삼각화(triangularization)를 한다. 또한 삼각 행렬에서 '1'의 밀도가 작아지도록 하는 swap heuristic 과정을 수행하여 복호 연산 양을 줄일 수 있는 것으로 보고하고 있다.

BP 복호에서 복호 과정이 중간에 실패하는 경우 차수-1의 새로운 부호 패킷을 기다리게 되는데, 삼각 화를 위해 xoring 연산을 하는 과정에서 차수-1의 행(row)이 발생할 확률이 높으므로 이러한 성질을 BP 복호에 적용하면 새로운 패킷을 기다리지 않아도 BP 복호를 연장하여 복호할 수 있다.

그림 3은 위의 xoring 연산 효과를 BP 복호에 적용할 경우 차수-1의 패킷 수의 변화를 보인 것으로 복호 종료 후 남는 차수-1의 수신 패킷은 없음을 알 수 있다. 이로 인해 오버헤드는 크게 감소하게 되며 xoring 연산에 의한 차수-1의 행 발생 효과를 확실하게 볼 수 있다.

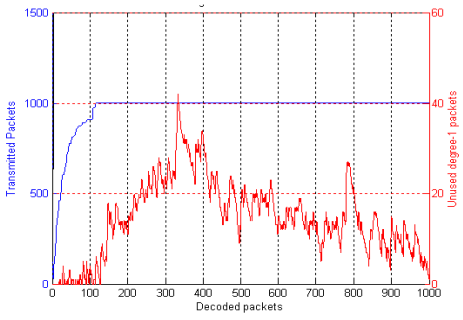

그림 3. 삼각 화를 이용한 복호에서 미사용 패킷 수의 변화 ($c = 0.1,\ k = 1000$)

Fig. 3. The number of unused packets of degree 1 as decoding packets are incresed ($c = 0.1,\ k = 1000$)

본 논문에서 제안하는 삼각 화 과정의 xoring 연산을 적용한 BP 복호 과정은 다음과 같이 정리할 수 있다.

(제 1 단계) 식 b_i를 포함하는 부호 패킷 y_i가 수신되면 복호기는 b_i의 차수를 검사하고 식 b_i를 행렬 $\boldsymbol{G}$에 다음과 같은 방법으로 위치시킨다. b_i의 원소 중에서 가장 좌측에 있는 1의 위치가 s_i라고 하면 b_i를 $\boldsymbol{G}[s_i]$에 넣는다. 만일 $\boldsymbol{G}[s_i]$에 이미 다른 식이 위치하고 있을 경우 [2]의 xoring 연산을 이용해 $\boldsymbol{G}$에 넣는다.

(제 2 단계) 식 b_i의 차수가 1일 경우 메시지 $m_i = y_i$로 복호되며 해당 열(column) 상에 존재하는 모든 1을 0으로 바꾼다. 이때 $\boldsymbol{G}[l]$의 i 번째 열에 1이 존재할 경우 1을 0으로 바꾸는 동시에 $y_l = y_l \oplus y_i$ 연산을 통해 y_l를 갱신한다.

(제 3 단계) (제 2 단계)를 수행하는 과정에서 j_0번째 열에 1이 있는 차수-1의 행 $\boldsymbol{G}[i_0]$가 발견되면 $m_{i_0} = y_{i_0}$로 복호하고 (제 2 단계)의 갱신 과정을 반복한다.

(제 4 단계) 모든 메시지 패킷 m_i가 복호되거나(복호 성공) 차수-1의 패킷이 더 이상 존재하지 않을 때까지 (제 2~3 단계)를 반복한다. 만일 차수-1의 패킷이 존재하지 않아 복호가 중간에 실패한 경우 새로운 패킷을 수신하여 (제 1 단계)부터 반복하여 복호를 연장한다.

OFG 복호 기법의 가장 큰 특징은 복호 연산을 k 개의 패킷들이 모두 도착할 때까지 기다리지 않고 첫 번째 수신 패킷부터 복호 과정을 시작하여 마지막 패킷이 도착하면 즉시 복호화 과정을 종료할 수 있다는 것이다. 본 논문에서 제안한 복호 방식도 첫 번째 수신 패킷부터 복호 과정을 수행하므로 위의 특성을 그대로 가지고 있으므로 전체 복호 시간이 짧으며, 행렬 $\boldsymbol{G}$에 대한 복호 과정은 BP 과정을 이용하므로 연산 양이 GE 방식(후방 대입 과정이 포함됨)에 비해 크게 감소하게 된다.

Ⅳ. 시뮬레이션 결과 및 분석

본 논문에서 제시한 삼각 화 xoring 연산을 이용한 복호 알고리즘의 성능을 고찰하기 위해 다음과 같이 시뮬레이션 실험을 수행하였다. 우선 복호 알고리즘은 LT 부호의 전형적인 복호 방식인 BP 알고리즘과 GE (Gaussian elimination) 기법을 기반으로 하는 OFG 알고리즘 및 본 논문에서 제시한 복호 알고리즘에 대해 각 성능 파라미터 별로 시뮬레이션 실험을 통해 결과를 도출하였다. 공정한 비교를 위해 OFG 알고리즘의 후방 대입(backward substitution) 연산 과정도 포함시켰다. $k = 50 \sim 900$일 경우 1,000 회 이상의 시뮬레이션을 통해 통계치를 얻었으며 $k \geq 1000$일 경우에는 최소 500회 이상의 시뮬레이션 실험을 수행하였다. δ 값은 0.01로 하고 리플의 크기에 영향을 주는 c 값은 0.1로 하여 세 가지 복호 알고리즘에 대해 각 각의 경우를 실험하였다.

표 1. k에 따른 오버헤드(c=0.1, $\delta = 0.01$)
Table 1. overhead versus k(c=0.1, $\delta = 0.01$)

DA / k	BP	OFG	Proposed
100	0.657240	0.052080	0.049820
200	0.548430	0.026120	0.025890
300	0.479947	0.015687	0.016380
400	0.447185	0.013610	0.014545
500	0.427604	0.010496	0.013492
600	0.391673	0.010157	0.009087
700	0.381020	0.007217	0.006414
800	0.364150	0.007385	0.010302
900	0.353740	0.005604	0.005460
1000	0.342586	0.005592	0.005328
1500	0.307916	0.004704	0.004543

표 1은 $c = 0.1$이고 $\delta = 0.01$일 때 k 값에 따른 각 복호 알고리즘 별 오버헤드 비중을 나타낸 것이다. 총 수신된 패킷 수를 n이라고 할 때 오버헤드 비중은 $\epsilon = n/k - 1$으로 표현할 수 있다. $k \leq 500$일 경우 BP 복호기는 약 50% 이상의 오버헤드가 있어야 복호를 성공적으로 종료할 수 있지만 OFG 알고리즘의 경우 약 1~5% 정도의 오버헤드이면 성공적으로 복호하는데 충분함을 알 수 있다. 특히 제안된 알고리즘의 오버헤드 또한 OFG와 거의 동일한 오버헤드 비중을 가지고 있음을 알 수 있다. k 값이 커질수록 BP 알고리즘의 오버헤드 비중은 점차 감소하지만 OFG와 제안된 알고리즘은 거의 오버헤드가 발생하지 않는다고 볼 수 있다. BP 알고리즘의 오버헤드 비중은 $k \geq 1000$일 경우에도 약 30% 이상의 큰 오버헤드를 필요로 한다.

표 2는 $c = 0.1$이고 $\delta = 0.01$일 때 k 값에 따른 각 복호 알고리즘 별 복호 복잡도(complexity)를 나타낸 것이다. 여기에서 제안된 알고리즘은 삼각 화가 완성되는 직후 복호 과정이 종료되는 반면 OFG 알고리즘은 삼각화 완료 후에 후방 대입 연산은 추가로 수행해야 복호가 종료되므로 OFG와 제안된 알고리즘의 전체 복호 시간은 k 값이 클수록 더 크게 차이가 난다고 볼 수 있다. 표 2에서 $k \leq 300$일 경우 BP와 제안된 알고리즘의 복잡도는 거의 동일함을 알 수 있으며 OFG 알고리즘은 약 5 배 이상의 연산 양이 필요함을 알 수 있다. 특히 $k = 500$, $c = 0.1$일 경우 제안된 알고리즘은 OFG 알고리즘에 비해 약 63% 의 연산 양 감소를 보이고 있으며, $k \geq 500$일 경우 k 값이 증가할수록 OFG와 제안된 알고리즘 사이의 연산 양 차이는 급격히 벌어짐을 알 수 있다.

표 2. k 에 따른 복호 복잡도 (c=0.1, $\delta = 0.01$)
Fig. 2. Decoding complexity versus k (c=0.1, $\delta = 0.01$)

DA / k	BP	OFG	Proposed
100	1371.48	6082.384	1239.282
200	2603.16	23481.152	4091.220
300	4199.36	52183.184	8567.530
400	5694.50	91856.274	14152.792
500	7360.952	142903.602	21519.284
600	8778.874	204553.084	29309.554
700	10305.786	277546.512	39035.794
800	10965.366	361333.354	49881.402
900	13836.662	456434.644	61988.632
1000	16640.720	561953.042	74588.932
1500	25630.758	1253954.42	155587.22

V. 결 론

본 논문에서는 오버헤드 비중이 GE 복호 방식과 거의 비슷하면서도 복호 연산 양을 크게 줄인 LT 부호의 복호 기법을 제안하였다. OFG 복호 방식의 삼각 화 과정에서 적용되는 xoring 연산을 통해 발생되는 차수-1의 행을 BP 복호 과정에 이용함으로서 BP 복호 기법이 가지고 있는 적은 연산 양과 GE의 삼각 화가 가지고 있는 적은 오버헤드 특징을 모두 가질 수 있음을 보였다.

OFG 알고리즘의 경우 BP 알고리즘에 비해 오버헤드가 거의 발생하지 않으므로 k 개의 부호 패킷이 전송되자마자 복호를 종료할 수 있는 장점을 가지고 있지만 제안된 알고리즘은 삼각 화 후 후방 대입 연산 과정이 없으므로 OFG 알고리즘보다도 더 빨리 복호를 종료할 수 있어 전체적인 복호 시간이 단축된다.

시뮬레이션 결과 $k \leq 500$일 경우 BP 복호기는 약 50% 이상의 오버헤드가 있어야 복호를 성공적으로 종료할 수 있지만 OFG 알고리즘과 제안된 알고리즘의 경우 약 1~5% 정도의 오버헤드이면 성공적으로 복호하는데

충분함을 알 수 있었다.

복잡도에 대한 시뮬레이션 결과 $k \leq 300$일 경우 BP와 제안된 알고리즘의 복잡도는 거의 동일함을 알 수 있으며 OFG 알고리즘은 약 5 배 이상의 연산 양이 필요함을 알 수 있었다. 특히 $k=500, c=0.1$일 경우 제안된 알고리즘은 OFG 알고리즘에 비해 약 63% 의 연산 양 감소를 보이고 있었으며, $k \geq 500$일 경우 k 값이 증가할수록 OFG와 제안된 알고리즘 사이의 연산 양 차이는 급격히 벌어짐을 알 수 있었다.

향후 제안된 알고리즘을 또 다른 Fountain 부호인 Raptor 부호[5]의 복호에 적용하여 각 파라미터 별 성능 분석을 하는 것이 필요하며, 네트워크 코딩의 복호 과정에 적용하는 연구도 필요하다.

참 고 문 헌

[1] M. Luby, "LT codes," Proceedings 43rd Annual IEEE Symposium on Foundations of Computer Science, pp. 271~280, Vancouver, Canada, Nov. 2002.

[2] Valerio Bilglio, Macro Grangetto, Rossano Gaeta, Matteo Sereno, "On the fly Gaussian Elimination for LT codes," IEEE Communication Letters, Vol. 13, No. 12, pp. 953~955, Dec. 2009.

[3] Saejoon Kim, Karam Ko, and Sae-Young Chung, "Incremental Gaussian Elimination Decoding of Raptor Codes over BEC," IEEE Communication Letters, Vol. 12, No. 14, pp. 307~309, April 2008.

[4] A. Shokrollahi, "Raptor codes," IEEE Transaction on Information Theory, Vol. 52, No. 6, pp. 2551~2567, June 2006.

[5] 심동희, 조동형,"멀티미디어 스트리밍을 위한 모바일 Ad-Hoc 네트워크의 구현,"한국정보기술학회 논문지, 제 5 권 제 2 호, 2007년 6월.

[6] 김한종, "채널 상태 정보를 이용하는 블록터보 부호화된 OFDM 시스템의 성능 분석," 한국산학기술학회논문지, v.12, no. 2, pp. 872-877, 2011.

[7] Dung Vu Anh, ChoonHee Kim, YoungWook Cha, "Design and Implementation of USN Management Protocol with Message Aggregation and Piggyback Mechanisms",한국정보기술학회 논문지, 제 8 권 제 8 호, 2010년 8월.

[8] Le Van Long, Bongsue Suh,"Design And Implementation of Multi Functional Sensor Node,"한국정보기술학회 논문지, 제 9 권 제 1 호, pp. 137-145, 2011년 1월.

[9] 나상혁, 박홍규, 이원석,"분산 데이터 스트림 환경에서의 슬라이딩 윈도우집계 계산 기법,"한국정보기술학회 논문지, 제 8 권 제 11 호, 2010년 11월.

[10] 홍대기, "대역폭 효율적인 다중 부호 변조 방식," 한국산학기술학회논문지, v.10, no. 7, pp. 1601-1607, 2009.

저자 소개

정 호 영(Ho-Young Cheong)

- 1989년 7월 : 연세대학교 본 대학원 전자공학과 (공학석사)
- 1994년 2월 : 연세대학교 본 대학원 전자공학과 (공학박사)
- 1995년 4월 ~ 현재: 남서울대학교 정보통신공학과 교수
- 2004년 3월 ~ 현재 : 남서울대학교 정보통신연구소 상임연구원

<관심분야 : 오류정정부호, 이동 통신, Mobile IPTV 등>

※ 본 연구는 2010학년도 남서울대학교 학술연구비 지원에 의하여 연구되었음