Random Permutation을 이용한 LDPC 부호의 성능 분석

한국통신학회 2012년도 추계종합학술발표회

고려대학교

2012 / 11 / 24

김영태, 전석민, 송민규, 김인선, 박진수, 송홍엽

연세대학교

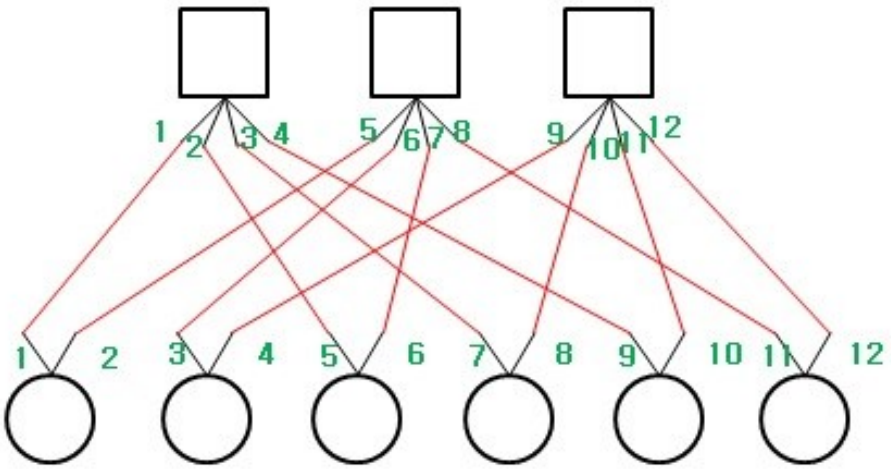
I. 서론

- LDPC 부호의 디자인은 degree distribution의 결정과 주어진 분포에서의 node 연결로 이루어짐.
- Node간의 연결상태를 결정하기 위해 PEG, ACE, Random 생성법 등이 있음.
- Degree distribution이 주어지면, 충분한 길이에서 random하게 node를 연결해도 좋은 성능을 얻을 수 있다는 사실을 보이고자 함.
- 5가지의 pseudo-random permutation으로 연결을 생성함.
- 성능의 비교를 위하여 비슷한 길이의 802.11n 표준의 irregular LDPC 코드와 PEG로 생성한 LDPC 부호를 사용.
- Random 생성한 parity-check 행렬은 표준과 동일한 degree distribution을 가지는 irregular 부호와 PEG와 동일한 code rate를 가지는 regular 부호임.

II. 본론 (1)

- 주어진 permutation으로부터 parity check 행렬을 생성하는 법
  - 예시

    -(2,4) regular LDPC code
    - permutation={1,5,7,9,2,3,6,11,4,8,10,12}



$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

그림1. Permutation과 node 연결의 관계

  - Edge 개수의 길이를 가지는 permutation이 필요함.

II. 본론 (2)

- 하나의 variable 노드, check 노드가 2개의 edge로 연결될 때

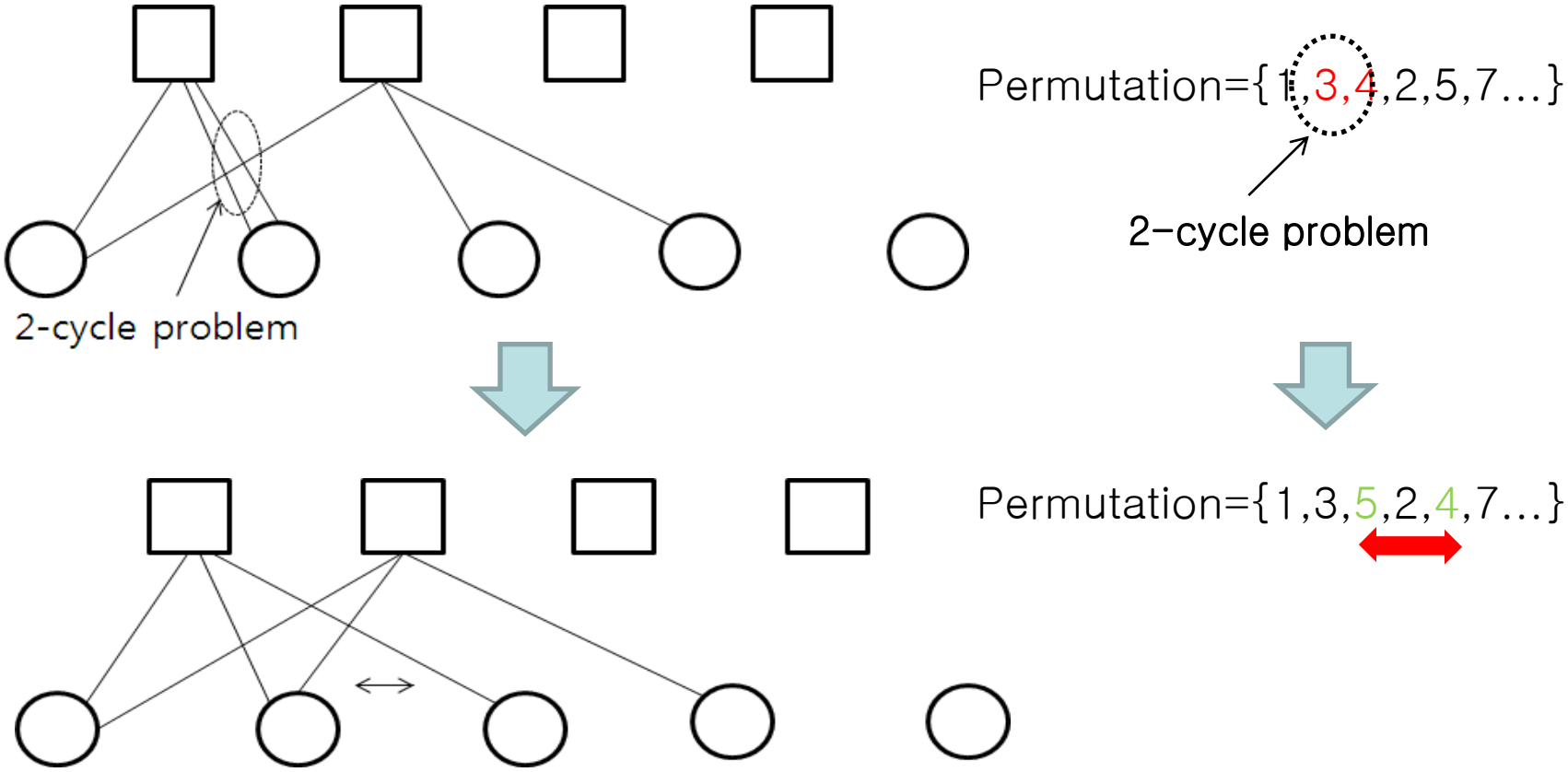


그림 2. 이웃 Edge와의 교환

II. 본론 (3)

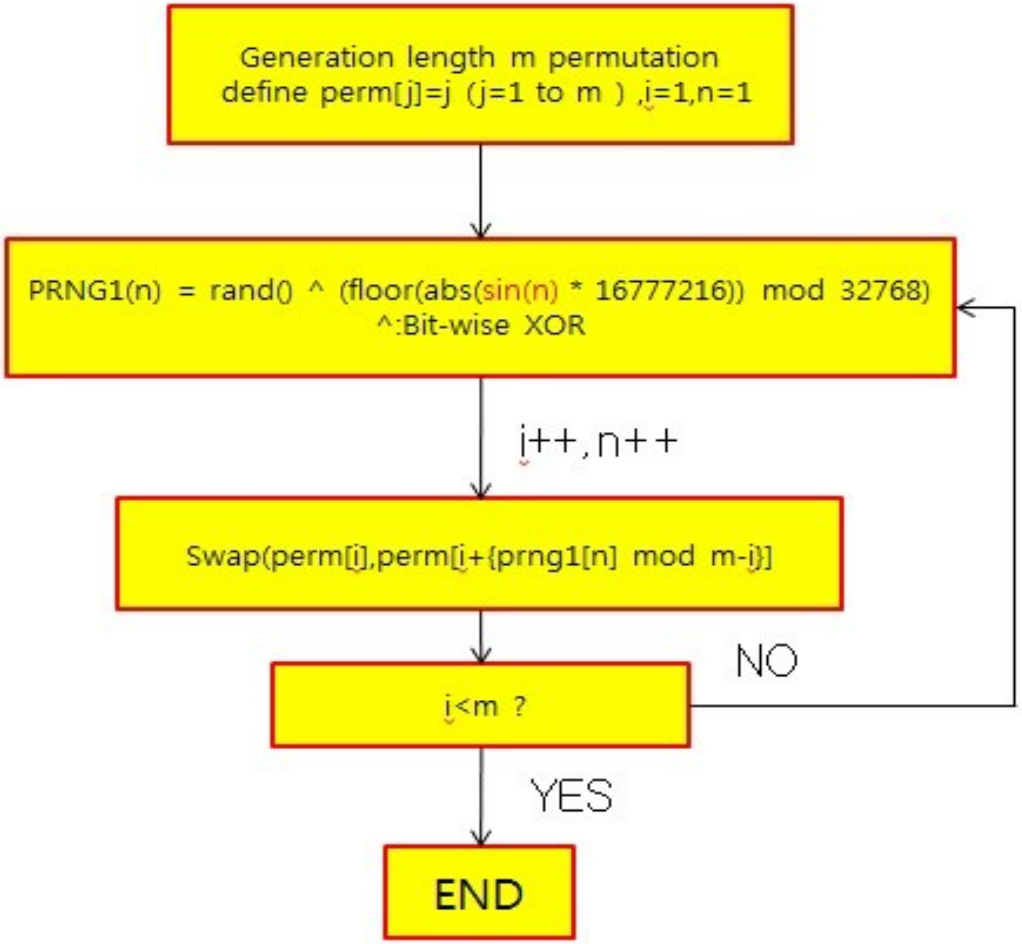
- Random permutation의 생성법

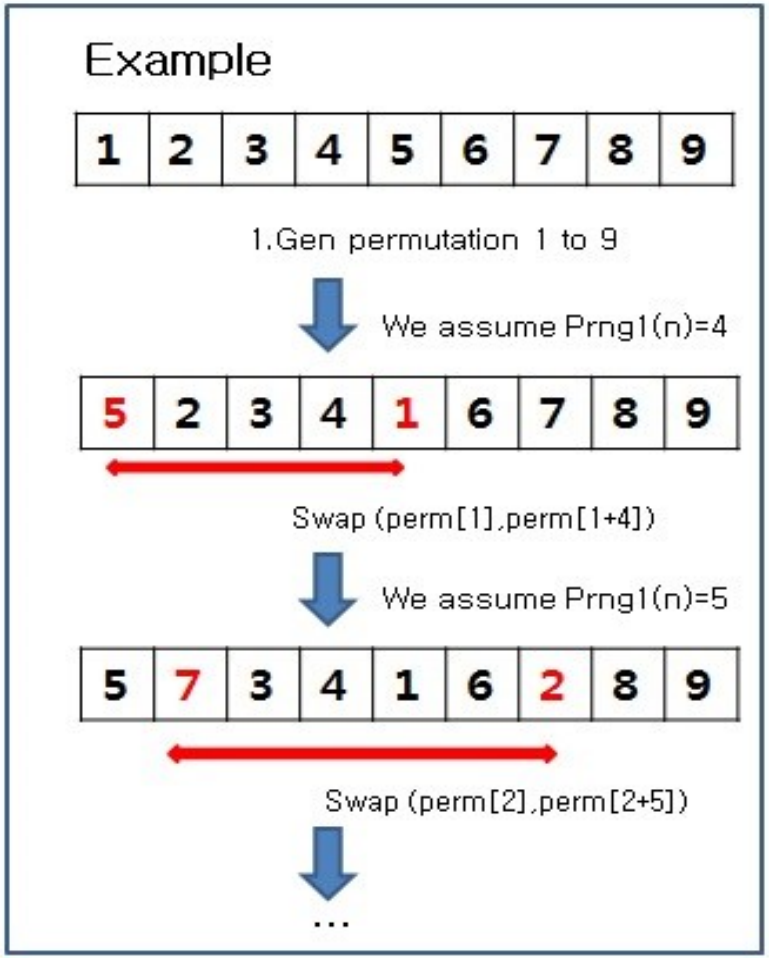
PRNG 1	PRNG(n) = rand() ^ (floor(abs(sin(n) * 16777216)) mod 32768)
PRNG 2	$prem[\,i\,] = floor(M \times \sin[n + x])$
Quadratic hash	x(i+1)=[x(i)+2∗i+1] (mod m) x(i+1)=[x(i)+t] (mod m)
Primitive root	the primitive root is defined by p_1 $p_1{}^{i} = n(mod\ m)$
LFSR	Span-n property of m-sequence

표 1. 각 permutation 생성법의 정의

II. 본론 (4)

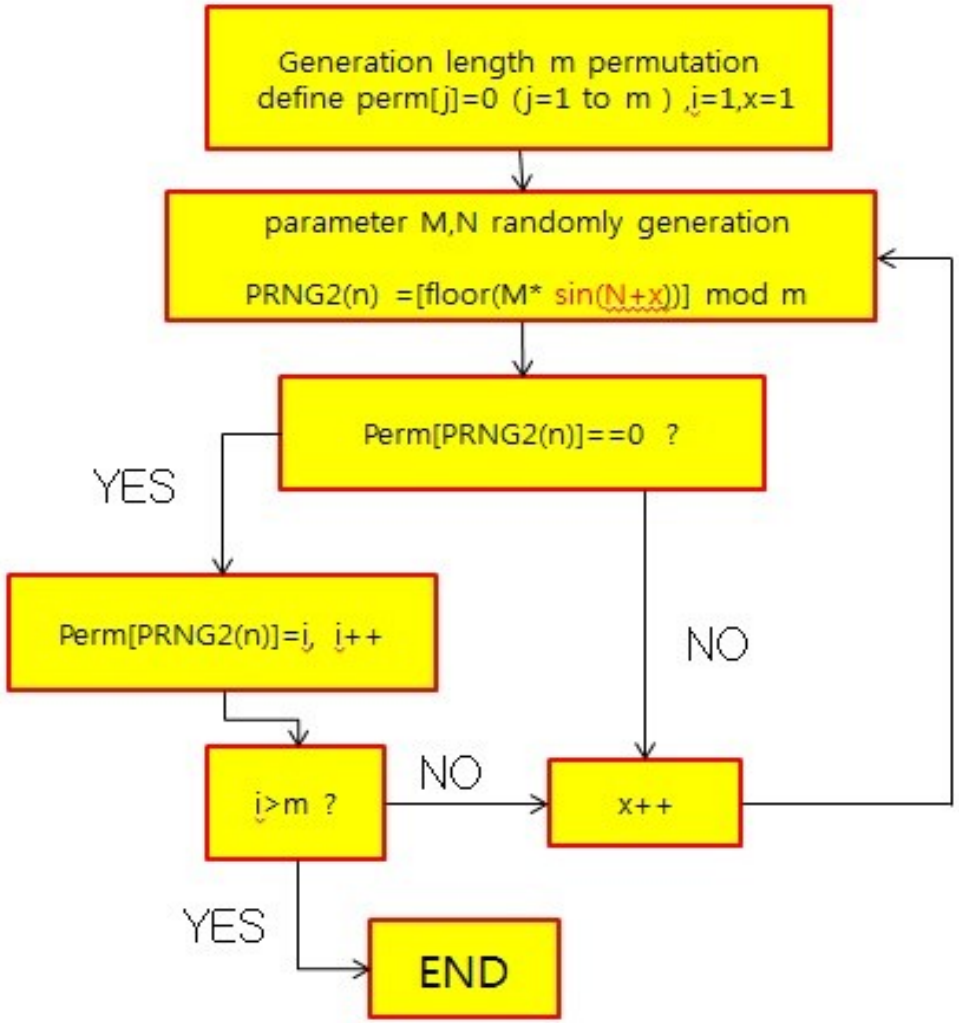
1. PRNG 1

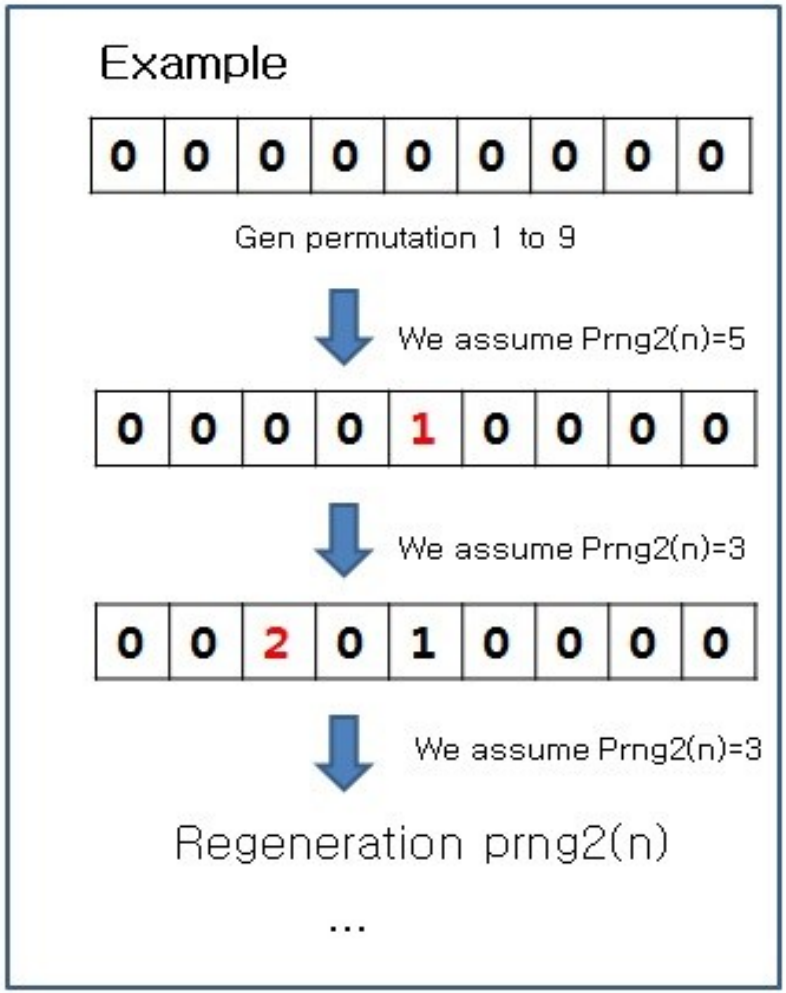




II. 본론 (5)

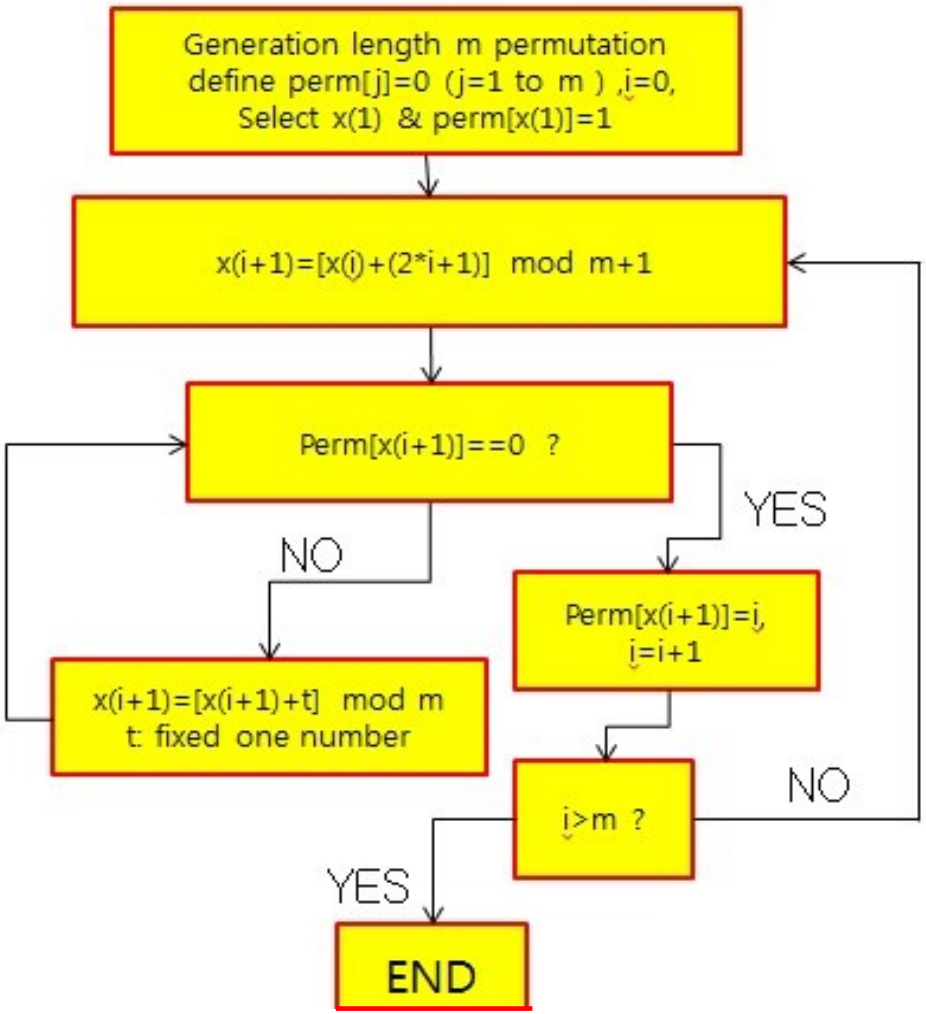
2. PRNG 2

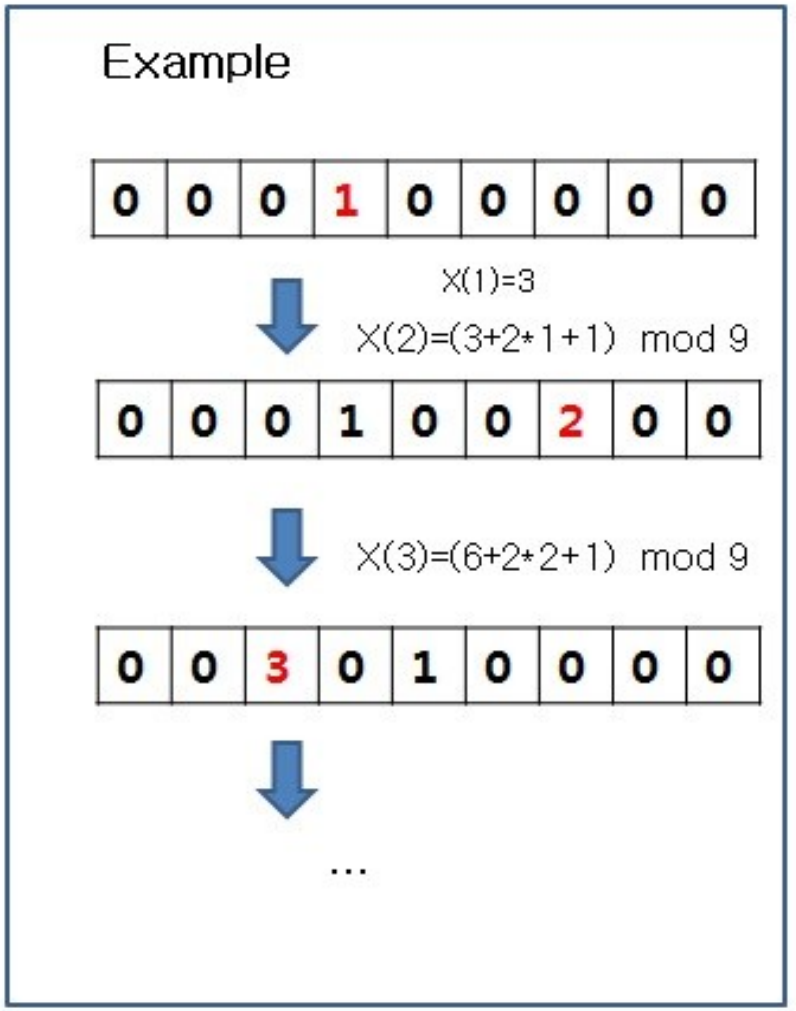




II. 본론 (6)

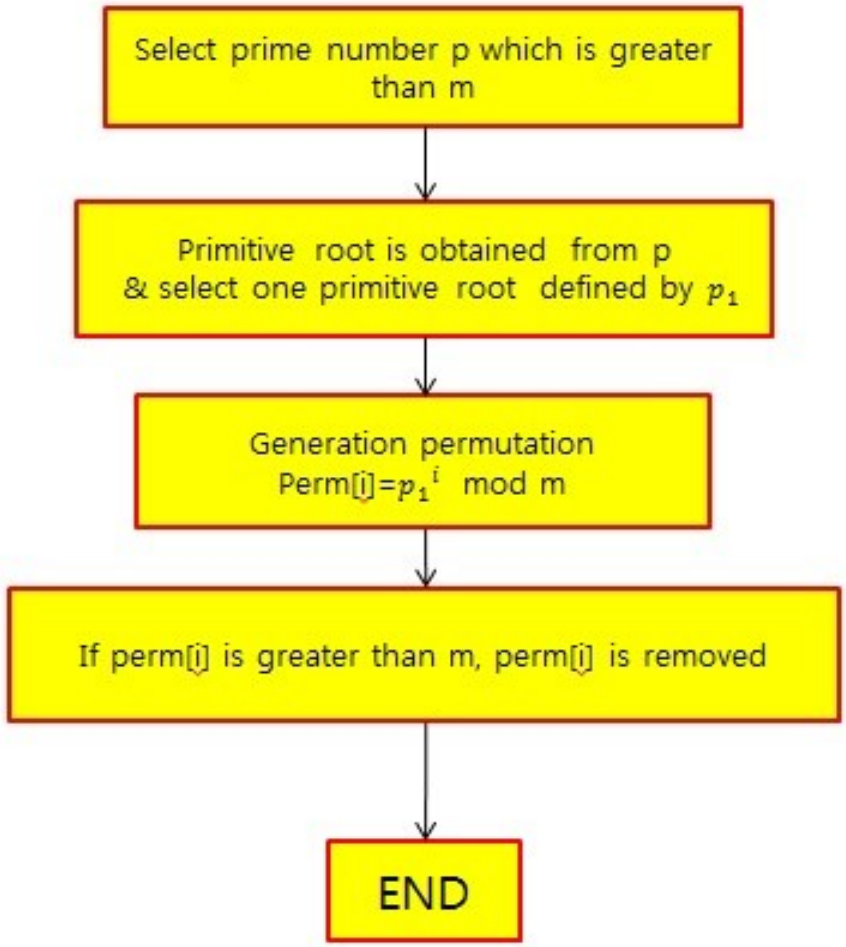
3. Quadratic hash permutation





II. 본론 (7)

4. primitive root permutation



Example

Select number 3 which is primitive of 5

3 → 3

4 → 4=(3*3) mod 5

2 → 2=(4*3) mod 5

1 → 1=(2*3) mod 5

Select number 2 which is primitive of 5

2 → 2

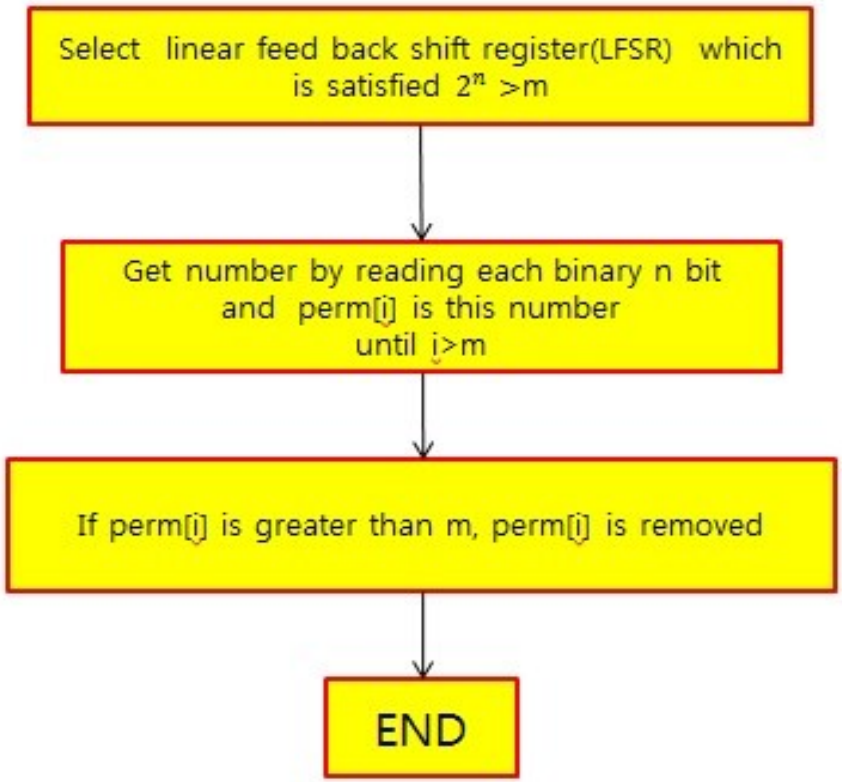
4 → 4=(2*2) mod 5

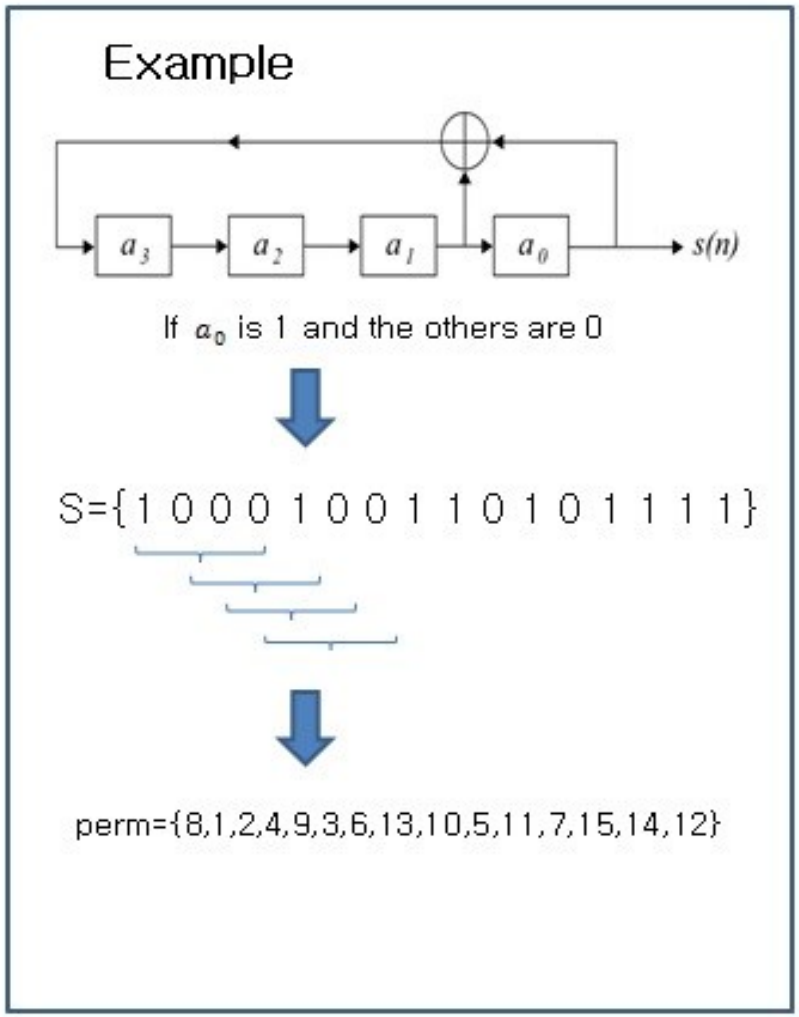
3 → 3=(4*2) mod 5

1 → 1=(3*2) mod 5

II. 본론 (8)

5. LFSR


Select linear feed back shift register(LFSR) which is satisfied $2^n > m$
Get number by reading each binary n bit and perm[i] is this number until i>m
If perm[i] is greater than m, perm[i] is removed
END


Example
a_3
a_2
a_1
a_0
s(n)
If a_0 is 1 and the others are 0
S={1 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1}
perm={8,1,2,4,9,3,6,13,10,5,11,7,15,14,12}

II. 본론 (9)

- PEG를 사용한 regular LDPC 부호
  - Code rate ½ 를 가지는 부호 길이 500, 1000 의 (3,6) regular 부호

- 802.11n 표준상의 irregular LDPC 부호
  - 부호 길이 1296, code rate ½ 의 irregular code
  - 총 4644개의 edge
  - Degree distribution

$$\lambda(x) = 0.2558x + 0.3140x^2 + 0.0465x^3 + 0.3837x^{10}$$

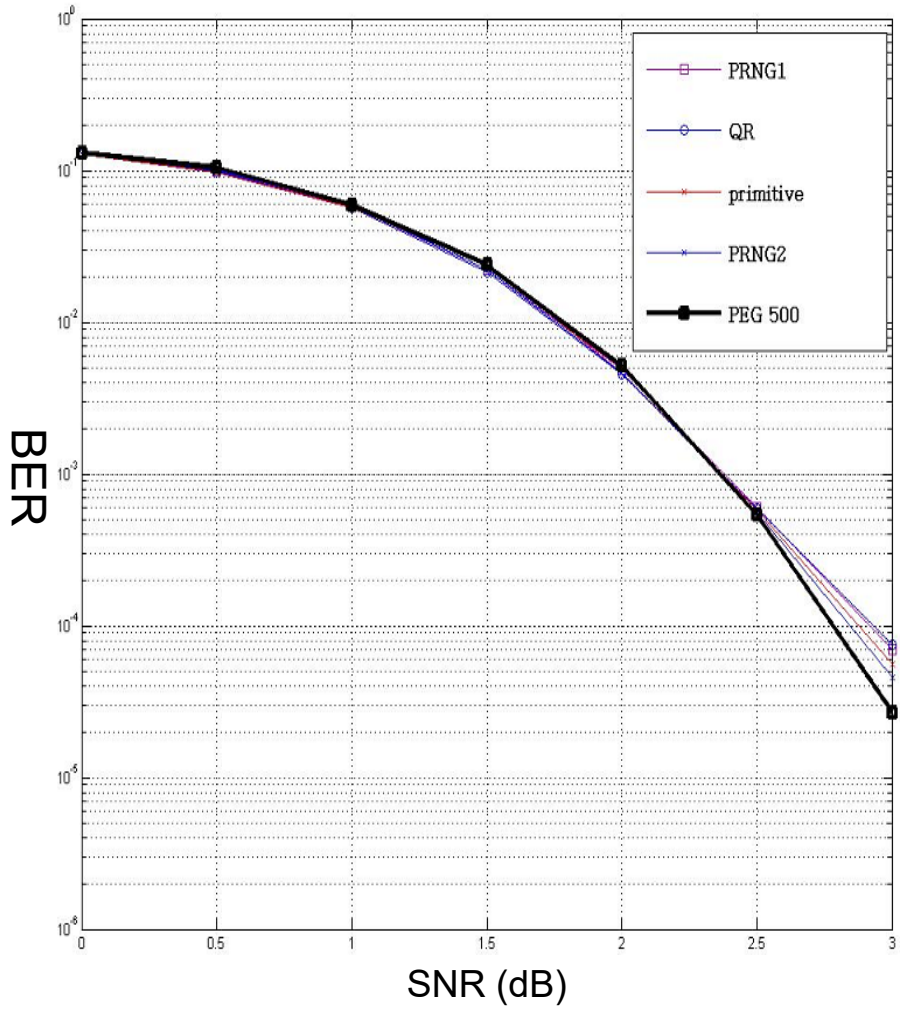
$$\rho(x) = 0.8140x^6 + 0.1860x^7$$

Variable node				Check node	
# degree 2	# degree 3	# degree 4	# degree 11	# degree 7	# degree 8
594	486	54	162	540	108

표 2. 802.11n 표준의 node 분포

III.결론 (1)

Regular code : 길이 500



Regular code : 길이 1000

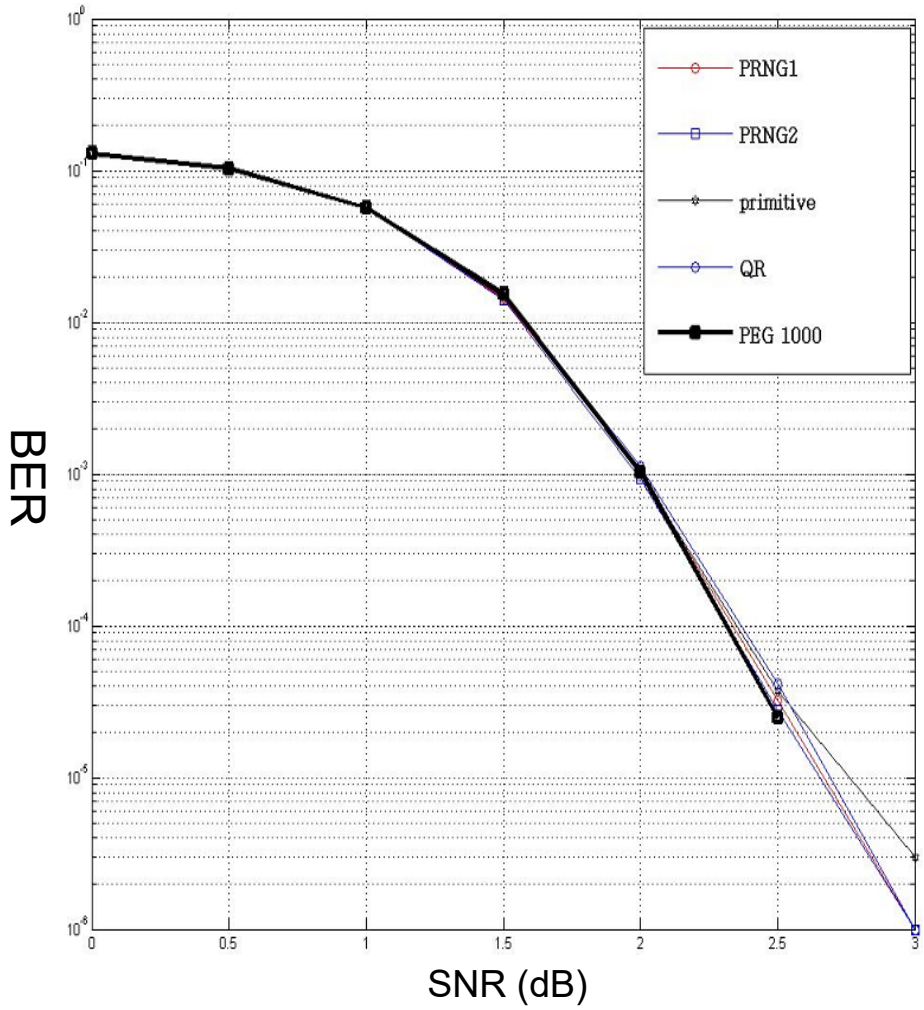


그림 3. 길이 500,1000의 PEG와 랜덤 regular LDPC 부호의 성능비교

CCL

YONSEI UNIVERSITY

III.결론 (2)

Irregular code : 길이 1296

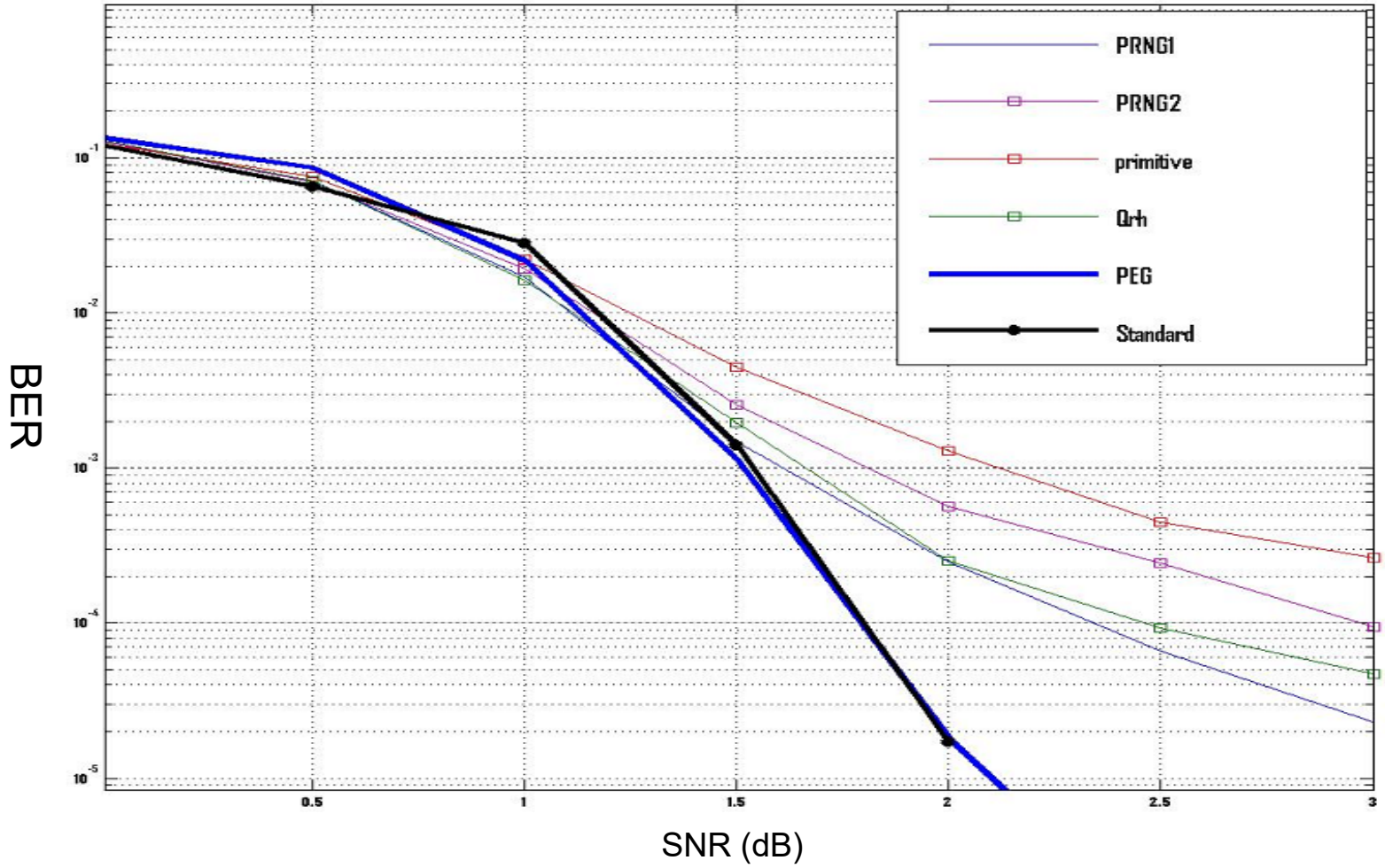


그림 4. 802.11n 표준과 랜덤 irregular LDPC 부호의 성능비교

III.결론 (3)

- Regular Case
  - LFSR을 제외한 4가지 방법은 길이 500, 1000 모두 PEG와 거의 차이가 없을 정도로 훌륭한 성능을 보여준다.
- Irregular Case
  - PRNG 1 방식이 가장 좋음.
  - LFSR 방식의 성능이 가장 떨어짐.
  - 최적화를 하지 않았기 때문에, Low SNR영역에서 좋은 성능을 보였으나 high SNR에서 급격히 성능이 떨어짐.
  - 이러한 성능저하가 특정 degree distribution에 의한 것인지, 최적화의 문제인지 알 수 없음.
- 결과 분석
  - LFSR을 제외한 4가지 방법은 최적화 과정 없이 매우 쉽게 생성할 수 있고, LDPC 부호로는 짧은 1000 내외의 길이만으로도 좋은 성능을 보여줌. 특히 regular 부호에서의 성능이 매우 좋음.