변형된 윈도우 복호를 이용한 SC-LDPC와 QC-LDPC 코드의 성능 비교

하유진, 김강산, 채상원, 송홍엽

연세대학교

2024년도 한국통신학회 하계종합학술발표회

본 논문은 SC-LDPC 코드의 윈도우 복호에서 메시지 함수를 사용했을 때의 성능 변화를 비교하고 이에 대한 알고리즘을 제시한다. 개선된 윈도우 복호를 적용한 SC-LDPC 코드와 BP flooding 방식으로 복호된 QC-LDPC 코드의 성능을 비교한다.

서론

- SC-LDPC 코드는 구조적인 특성을 활용한 윈도우 복호를 통해 낮은 복잡도로 복호를 수행하지만 스케줄링 방식에 따라 성능의 차이가 발생함.
- 본 논문은 메시지 함수에 따른 윈도우 복호 방식의 성능 변화를 비교하고, 가장 좋은 성능을 보인 윈도우 복호 방식을 이용하여 QC-LDPC와 성능을 비교함.

메시지 함수를 통한 피드백

현재 윈도우에서 패리티 검사 결과에 따라 다음 윈도우에서 사용될 메시지에 피드백을 주는 실험을 진행함.

$$\text{Rectifier: } g(x) = \begin{cases} x & (H\underline{v}^T = 0) \\ 0 & (H\underline{v}^T \neq 0) \end{cases} \quad (1)$$

$$\text{Scaling: } g(x) = \begin{cases} 2x & (H\underline{v}^T = 0) \\ x & (H\underline{v}^T \neq 0) \end{cases} \quad (2)$$

$$\text{Cube: } g(x) = \begin{cases} x^3 & (H\underline{v}^T = 0) \\ x & (H\underline{v}^T \neq 0) \end{cases} \quad (3)$$

알고리즘

Algorithm 1 Window Decoding

Inputs:

L	Coupling length
W	Window size
R_i	Received LLR at ith variable node over AWGN channel
I_{max}	Maximum iteration

// LLR of received signals
Every variable nodes ← R_i
// Window Decoding start
current position ← 0
while current position ≤ $L - W + w$ **do**
  // Initialization phase
  Calculate window arguments
  Load window variable nodes, check nodes, edge messages
  // Iterative decoding
  while $I < I_{max}$ or $H\underline{v}^T = \underline{0}$ **do**
    Variable nodes to check nodes:
    $M_{i,j} = \sum_{j' \notin A,\, j' \neq j} E_{j',i} + R_i$
    Check nodes to variable nodes:
    $E_{j,i} = \log \frac{1 + \prod_{i' \notin B_j,\, i' \neq i} \tanh\left(M_{j,i'}/2\right)}{1 - \prod_{i' \notin B_j,\, i' \neq i} \tanh\left(M_{j,i'}/2\right)}$
    Hard decision:
    $L_i = \sum_{j \in A} E_{j,i} + R_i$
    $v_i = \begin{cases} 0, & (L_i \leq 0) \\ 1, & (L_i > 0) \end{cases}$
    Test $H\underline{v}^T = \underline{0}$
  end while
  // Keep decoding results for the next position
  Apply $g(x)$ to edge messages according to $H\underline{v}^T$ results
  Update target variable node LLRs L_i and hard decision values v_i
  current position ← current position + 1
end while

성능 비교

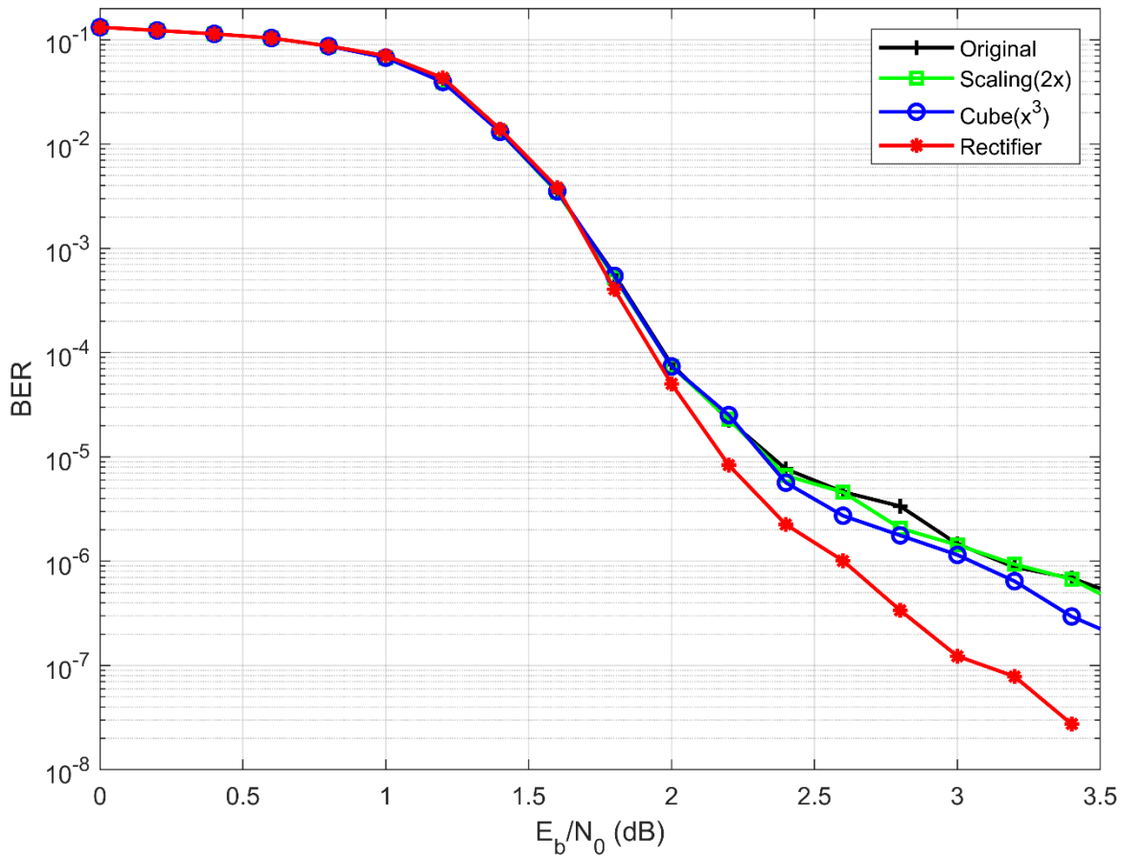

그림 1. $g(x)$에 따른 BER 성능

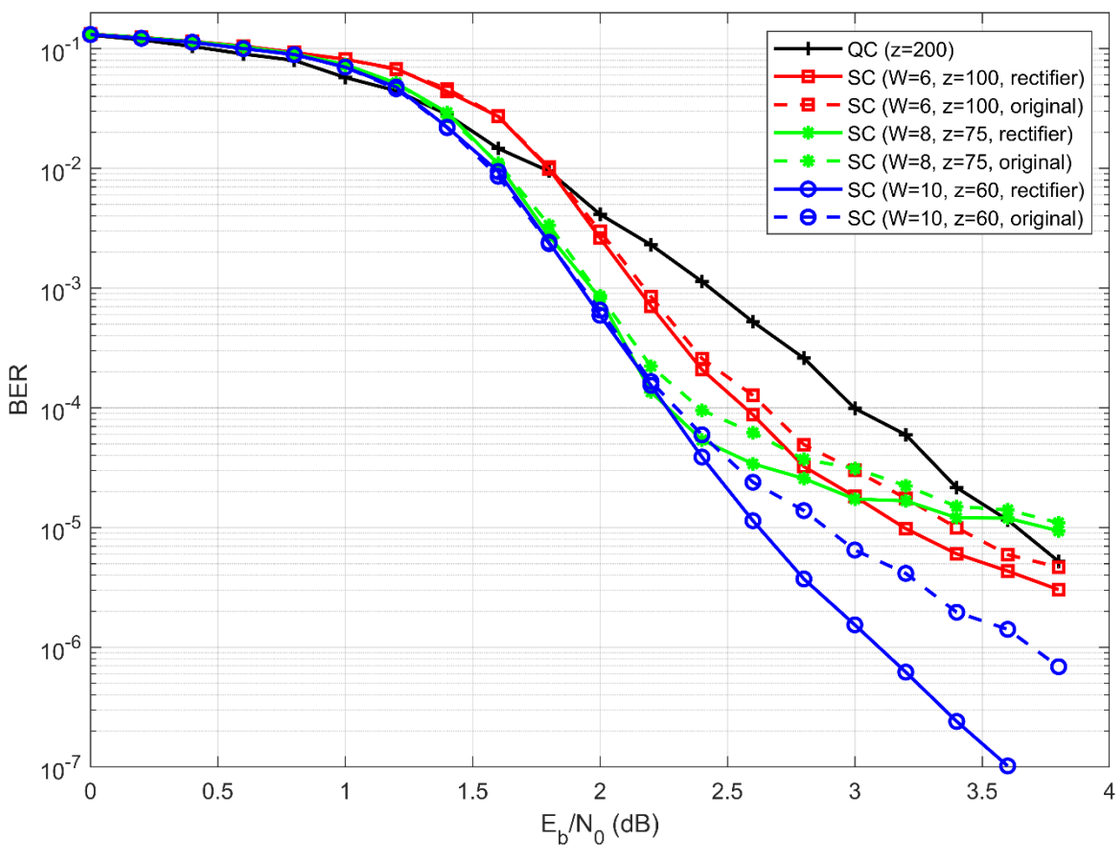

그림 2. 동일한 지연시간에서 SC-LDPC와 QC-LDPC의 성능

결론

- Rectifier 메시지 함수를 사용한 윈도우 복호에서 오류 마루 영역의 큰 성능 개선을 보였음.
- QC-LDPC와의 성능 비교에서 동일한 지연시간에 대해 SC-LDPC의 성능이 대체로 좋게 나타났음.
- 메시지 함수의 오류 마루 개선 효과는 윈도우 크기과 리프팅 계수에 따라 다르게 나타남.

REFERENCES

[1] I. Ali, J. -H. Kim, S. -H. Kim, H. Kwak, and J. -S. No, "Improving Windowed Decoding of SC LDPC Codes by Effective Decoding Termination, Message Reuse, and Amplification," *IEEE Access*, vol. 6, pp. 9336-9346, 2018.

[2] M. Lentmaier, M. M. Prenda, and G. P. Fettweis, "Efficient message passing scheduling for terminated LDPC convolutional codes," *2011 IEEE International Symposium on Information Theory Proceedings*, St. Petersburg, Russia, 2011, pp. 1826-1830.

[3] W. -J. Kim, H. -W. Cho, H. -Y. Song, and M. -K. Song, "Some Variations of Tanner's Construction for Short Length QC-LDPC Codes," *IET Electronics Letters*, vol. 60, no. 3, pp. e13088, January 2024.