Efficient Design of Structured LDPC codes

2005. 7.1

연세대학교 부호 및 정보이론 연구실
정 비 웅

CONTENTS

- INTRODUCTION
- STRUCTURE OF H-MATRIX
- ALGORITHM
- CONSTRUCTION
- SIMULATION RESULTS
- CONCLUSION

INTRODUCTION

- Structured LDPC codes
  - decoder implementation complexity, structured decoding
  - low complexity encoding
  - shortening, puncturing, scaling method
  - memory efficiency

- Efficient design algorithm
  - block type
  - regular and irregular LDPC structure
  - Goal
    - Construction of structured LDPC codes with good performance

STRUCTURE OF H-MATRIX

- H-Matrix
  - $m \times n$ sub-blocks defined as a shifted identity matrix
  - sub-block size : $N_s \times N_s$
  - $X^{s_{i,j}}$ is the circulant matrix of I to the right by $(s_{i,j} \bmod N_s)$.
  - code rate (if H has full-rank)

$$R = \frac{N_s n - N_s m}{N_s n} = \frac{n-m}{n} = 1 - \frac{m}{n}$$

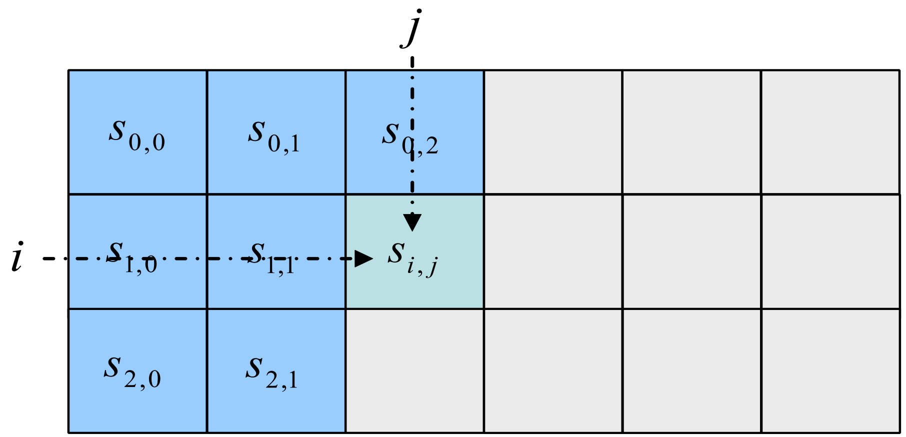
$$H = \begin{bmatrix} X^{s_{0,0}} & X^{s_{0,1}} & X^{s_{0,2}} & \cdots & X^{s_{0,(n-2)}} & X^{s_{0,(n-1)}} \\ X^{s_{1,0}} & X^{s_{1,1}} & X^{s_{1,2}} & \cdots & X^{s_{1,(n-2)}} & X^{s_{1,(n-1)}} \\ X^{s_{2,0}} & X^{s_{2,1}} & X^{s_{2,2}} & \cdots & X^{s_{2,(n-2)}} & X^{s_{2,(n-1)}} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ X^{s_{(m-1),0}} & X^{s_{(m-1),1}} & X^{s_{(m-1),2}} & \cdots & X^{s_{(m-1),(n-2)}} & X^{s_{(m-1),(n-1)}} \end{bmatrix}$$

ALGORITHM (1)

- Proposed Algorithm
  - for given m, n, N_s, weight distribution
  - use local girth condition, improved PEG algorithm

- Regular

```
for j = 0 to n-1
begin
    for i = 0 to m-1
    begin
        Decide s_{i,j}
        Insert (i, j, s_{i,j})
    end
end
```



ALGORITHM (2)

- Decide $s_{i,j}$

```
Decide s_{i,j}
begin
if i = 0
    s_{i,j} = 0
else
    temp ← 0
    Insert (i, j, temp)
    g1 ← local girth (j × N_s)
    Delete (i, j)
    for k = 1 to N_s − 1
    begin
        Insert (i, j, k)
        g2 ← local girth (j × N_s)
        if g2 > g1
            g1 ← g2
            temp ← k
        else if g2 = g1
            gsum1 ← Σ_{l=0}^{j−1} local girth (l × N_s)
            Delete (i, j)
            Insert (i, j, temp)
            gsum2 ← Σ_{l=0}^{j−1} local girth (l × N_s)
            if gsum1 > gsum2
                temp ← k
        Delete (i, j)
    end
    s_{i,j} ← temp
end
```

ALGORITHM (3)

- Insert $(i, j, s_{i,j})$: insert $X^{s_{i,j}}$ to (i, j)th block
- Delete (i, j) : delete (i, j)th block from tanner graph
- local girth (x) : check cycle xth bit node under the current graph

Irregular

- optimum weight distribution by density evolution
- additional work : determine position which sub-block are insert
  - ✓ Improved PEG algorithm
  - ✓ balance between bit node and check node distribution
- Decide $s_{i,j}$ is same.

CONSTRUCTION (1)

- Structured Regular LDPC codes
  - $m = 3$, $n = 6$, $N_s = 167, 334$
  - $R = 0.5$, $N = 1002, 2004$

- Structured Irregular LDPC codes
  - $m = 24$, $n = 48$, $N_s = 24, 32, 48$
  - $R = 0.5$, $N = 1152, 1728, 2304$

length	code	4-cycle	6-cycle	8-cycle	10-cycle
1002	Mackay	0	544	457	1
	PEG	0	0	78	904
	Proposed	0	0	0	1002
2004	Mackay	0	666	1313	25
	PEG	0	0	0	2004
	Proposed	0	0	0	2004

Table 1.Cycle distribution of rate-0.5 regular LDPC codes

CONSTRUCTION (2)

- bit node distribution

  $\lambda(x) = 0.2528x + 0.2967x^2 + 0.4615x^{11}$
- Parity part is pre-determined for simple encoding
  - ✓ Richardson's encoding method

length	code	4-cycle	6-cycle	8-cycle
1152	imPEG	0	1086	66
	B-LDPC	0	864	288
	Proposed	0	720	432
1728	imPEG	0	1414	314
	B-LDPC	0	1512	216
	Proposed	0	612	1116
2304	imPEG	0	1475	829
	B-LDPC	0	1296	1008
	Proposed	0	576	1728

Table 2.Cycle distribution of rate-0.5 irregular LDPC codes

SIMULATION RESULTS (1)

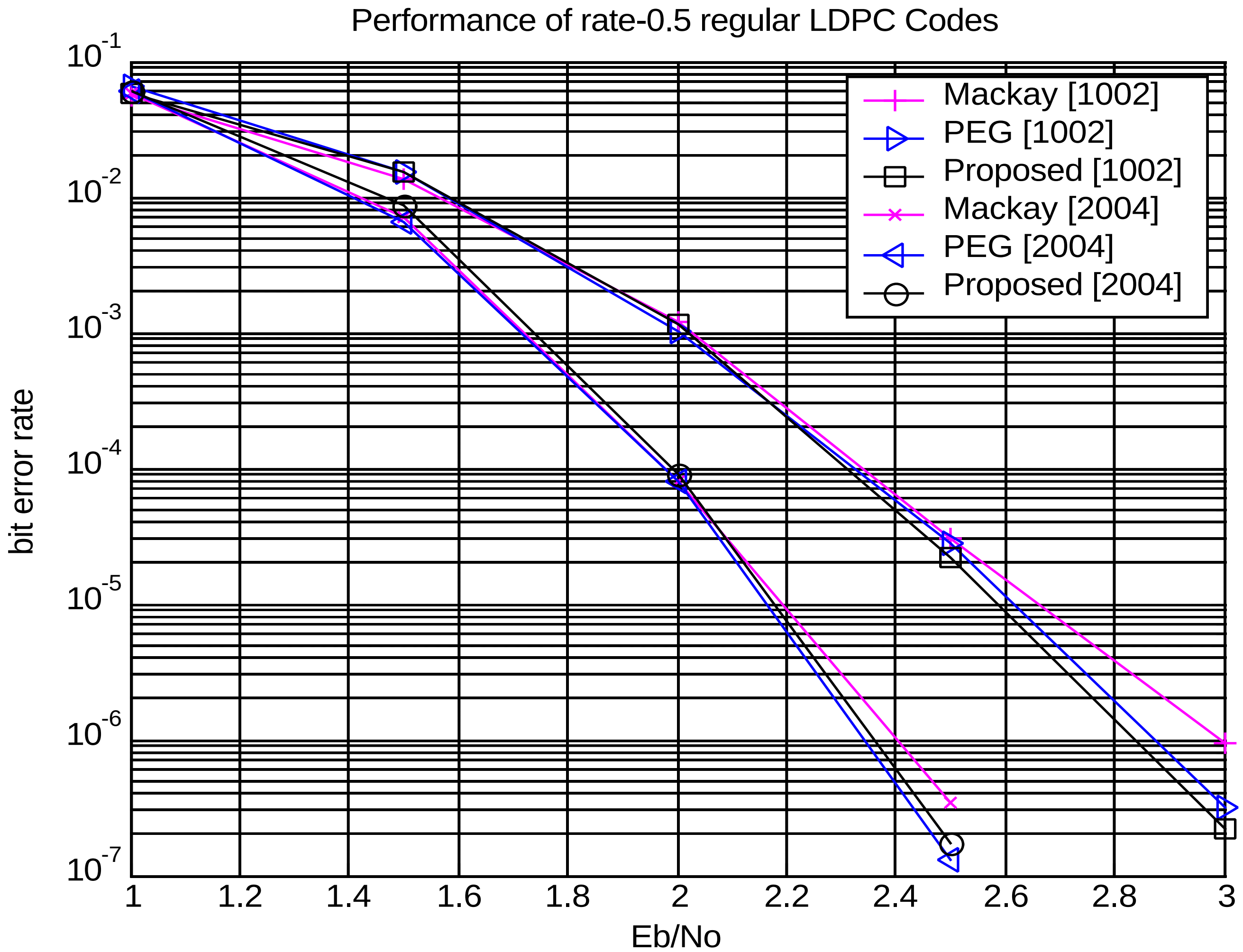


Fig 1. Performance of rate-0.5 regular LDPC Codes

SIMULATION RESULTS (2)

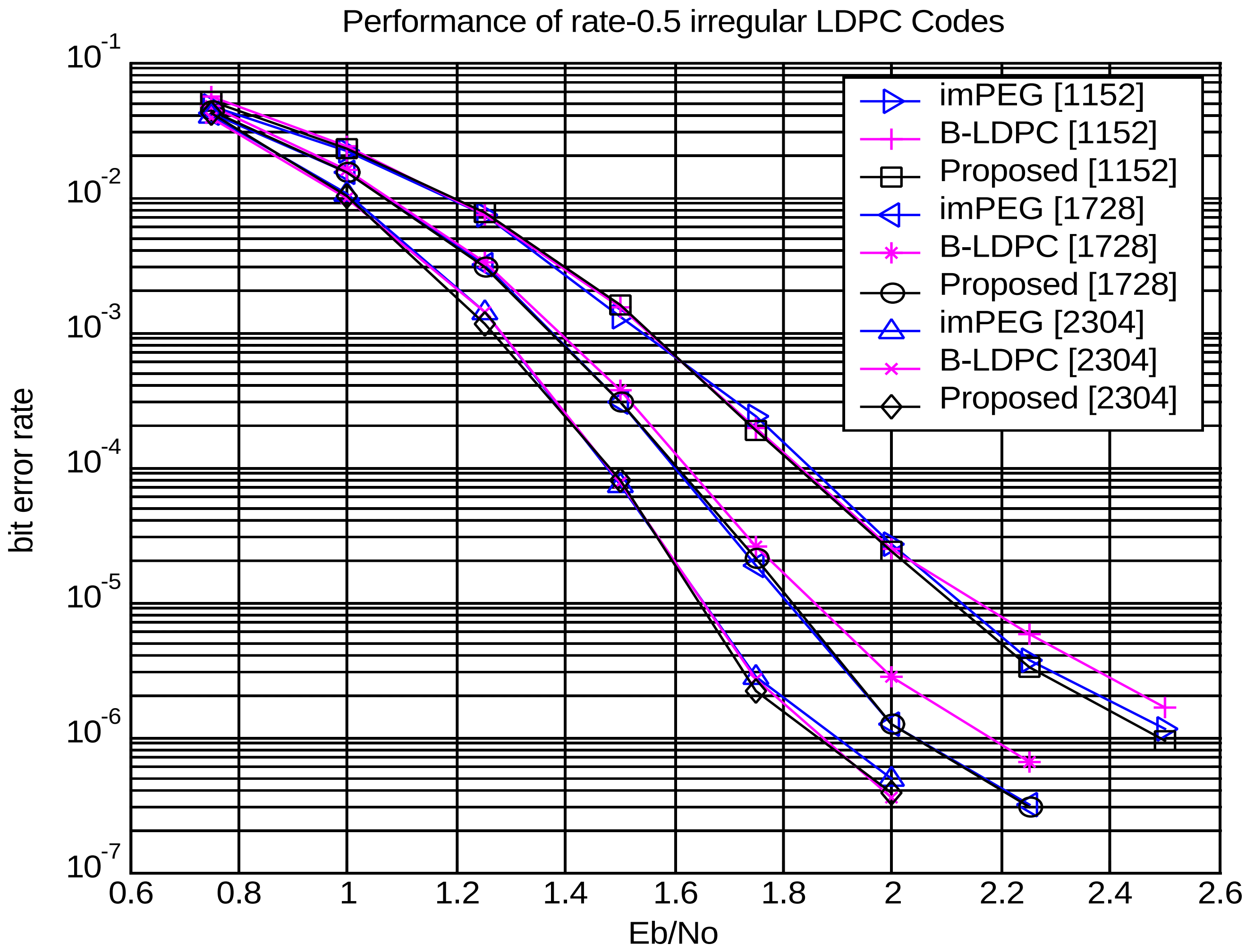


Fig 2. Performance of rate-0.5 irregular LDPC Codes

CONCLUSION

- Proposed Algorithm
  - short construction time
  - various parameter
  - high average local girth
  - good performance
  - small error floor

- Future work
  - more criterion (stopping set)
  - high rate code ($R \geq 0.9$)