Introduction

- **LDPC codes with finite block length**
  - Parameter related to performance:
    - girth (the shortest cycle)
    - connectivity of graph
  - Construction algorithms:
    - ACE algorithm: connectivity of graph
    - PEG algorithm: local girth
    - **Improved PEG algorithm (IPEG) algorithm**: PEG + ACE
      - One of the best known methods for constructing LDPC codes.

- **IPEG Algorithm**
  - A lot of random selection for new edges
    - About 40~50% of total edges.

  ➔ **IPEG algorithm needs more specific criterion** for selection!!
Connectivity of variable nodes (1)

- Stopping sets and EMD (extrinsic message degree)

**EMD = 0**

**EMD = 2**

**Stopping set**

\[ A = \{v_3, v_5, v_6\} \]

**Not stopping set**

\[ B = \{v_1, v_2, v_3\} \]

**Extrinsic check node**

\[ \text{The number of 1's} = 0 \]

\[ \text{The number of 1's} = 2 \]

**EMD**

\[ \text{EMD} = 0 \]

\[ \text{EMD} = 2 \]
Connectivity of variable nodes (3)

- **ACE (Approximate cycle EMD)**
  - ACE of variable node: (degree -2), ACE of check node: 0
  - \( \text{ACE} = \sum \text{(ACE of each node)} \)

- **EMD and ACE**

  * ACE = 2 = (3-2)+(2-2)+(2-2)+(3-2)
  * EMD = 2
  * ACE = EMD

(a) No subcycle

(b) Subcycles

* ACE = 2 = (3-2)+(2-2)+(2-2)+(3-2)
* EMD = 1
* ACE ≠ EMD
PEG algorithm and IPEG algorithm

- Given parameter
  - the number of variable node \((n)\), the number of check node \((m)\)
  - variable node degree sequence : density evolution.

- Basic idea
  - make local girth as large as possible whenever placing a new edge.

<table>
<thead>
<tr>
<th></th>
<th>PEG</th>
<th>IPEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Construction method</td>
<td>Tree spreading</td>
<td>Tree spreading</td>
</tr>
<tr>
<td>Criterion for selection - 1st</td>
<td>Highest depth</td>
<td>Highest depth</td>
</tr>
<tr>
<td>Criterion for selection - 2nd</td>
<td>Lowest degree</td>
<td>Lowest degree</td>
</tr>
<tr>
<td>Criterion for selection - 3th</td>
<td>Random</td>
<td>ACE</td>
</tr>
<tr>
<td>Criterion for selection - 4th</td>
<td></td>
<td>Random</td>
</tr>
</tbody>
</table>
Modified IPEG algorithm (1)

- Proposed algorithm

  - $\Phi$: the candidate check node classified by IPEG algorithm

  - Select a check node in $\Phi$ by calculating the EMD of subgraph.
    
    ✓ Extract subgraph for each check node in $\Phi$
    
    ▪ By back-tracking from the check node to root (variable) node.

    ✓ Calculate the EMD of variable nodes in each subgraph
    
    ▪ Counting the number of 1’s in the column sum over the variable nodes in subgraphs

    ✓ Select a check node that has the maximum EMD

  - If there are more than two check nodes that have the same EMD, select a check node that has more variable node in subgraph.
Modified IPEG algorithm (2)

Example

Degree=2
ACE =3

Degree=2
ACE =3

EMD = 6

EMD = 5
Simulation results (1)

\[ \lambda(x) = 0.21991x + 0.23328x^2 + 0.02058x^3 + 0.08543x^4 + 0.06540x^5 + 0.04767x^7 + 0.01912x^8 + 0.08064x^{18} + 0.22798x^{19} \]

Block length = 2000
Maximum variable node degree = 20
Coderate 1/2

<table>
<thead>
<tr>
<th>Total edges (except first)</th>
<th>6326</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random selection after PEG</td>
<td>5327 (0.84208)</td>
</tr>
<tr>
<td>Random selection after IPEG</td>
<td>2476 (0.43408)</td>
</tr>
<tr>
<td>Random selection after proposed</td>
<td>77 (0.01217)</td>
</tr>
</tbody>
</table>
Simulation results (2)

Block length = 3000
Maximum variable node degree = 30
Coderate 1/2

$$\lambda(x) = 0.19606x + 0.24039x^2 + 0.00228x^5 + 0.05516x^6 + 0.16602x^7 + 0.04088x^8 + 0.01064x^9 + 0.002218x^{27} + 0.28636x^{29}$$

<table>
<thead>
<tr>
<th>Total edges (except first)</th>
<th>10496</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random selection after PEG</td>
<td>8997 (0.85718)</td>
</tr>
<tr>
<td>Random selection after IPEG</td>
<td>5260 (0.50114)</td>
</tr>
<tr>
<td>Random selection after proposed</td>
<td>148 (0.01410)</td>
</tr>
</tbody>
</table>