Design and Construction of LDPC Codes

Prof. Francis C.M. Lau

Department of Electronic and Information Engineering
Hong Kong Polytechnic University

Content

- Introduction
  - Properties of LDPC Codes
  - Design Factors/Requirements
- Block-Structured LDPC Codes
- Girth
- Construction Methods
Introduction

Properties of the LDPC codes

- \( H \) is sparse (very few non-zero elements in \( H \))
- regular code
  - constant column weight and constant row weight
- irregular code
  - unequal column weights and/or unequal row weights
- optimal weight distributions
  - column weights are non-uniform
  - row weights are nearly-uniform with only two or three consecutive weight values
Factors/Requirements in Designing a Channel Code

- Error detection and/or error correction
- Channel type
- Error performance
- Transmission power
- Bandwidth
- Information rate
- Code rate
- Coding gain
- Data structure
- Development cost
- Latency
- Hardware complexity

Design of the LDPC codes

- error performance
- complexity of the encoder and the decoder

\[ \begin{align*}
H & \xrightarrow{\text{Gaussian Elimination}} H_i \\
& \xrightarrow{\text{transposed}} \text{Q} \\
& \xrightarrow{\#} G = I \cdot Q
\end{align*} \]

- \( H_{\text{in}} \): randomly constructed
- \( H_i \): sparse
- \( Q \): not sparse
- \( \text{Q} \): may not have \( m \) independent rows

\[ G = \begin{bmatrix}
1 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 1
\end{bmatrix} \]

Prof. Francis C.M. Lau, EIE
Parity portion $H_2$ in a lower-triangular (LT) form

$$(H) = [(H_1) \times (H_2)]$$

allow low complexity encoding

WHY ??

Block-Structured LDPC Codes
Block-structured LDPC codes

A block-structured LDPC code is represented by a “base parity-check matrix” $H_b$, in which each element is a square sub-block matrix of size $z$ by $z$.

**Advantages:**
- much less memory required for storage due to the much smaller size of $H_b$ (reduced by $z$ times $z$)
- easily and flexibly expanded to a parity-check matrix $H$ for different code lengths by using sub-block matrices with appropriate sizes

---

**Block structure of $H$**

- $(H_b)_{4 \times 5}$
- $(H)_{16 \times 20}$

**expansion factor** $z = 4$
Lower-triangular structure

$0, I$ and $P$ represent a zero matrix, an identity matrix and a cyclic-right-shifted identity matrix, respectively. (Also called quasi-cyclic (QC) LDPC code)

Dual-diagonal structure presented in IEEE 802.16e standard

Reduction in short cycles during the construction

Deterministic
IEEE802.16e code

- Code length $n = 2304$
- Expansion factor $z = 96$
- Base parity-check matrix contains 12 rows and 24 columns for a rate 1/2 code
Another possible structure

P₁ to P₇ are identity matrices I or cyclic-right-shifted identity matrices. At least two out of P₁, P₂, and P₃ are the same. P₄=P₅ and P₆=0; or P₄=P₆ and P₅=0.

The design becomes more flexible.

Particular form of the structure

P₁ is a cyclic-right-shifted identity matrix P.
Encoding

\[ H = \begin{bmatrix}
    h_{0,0} & h_{0,1} & \cdots & h_{0,L-1} & 0 & I & 0 & \cdots & 0 & 0 & 0 \\
    h_{1,0} & h_{1,1} & \cdots & h_{1,L-1} & 0 & h_{L,L+1} & I & \cdots & 0 & 0 & 0 \\
    h_{2,0} & h_{2,1} & \cdots & h_{2,L-1} & 0 & h_{L+1,L+2} & h_{L+1,L+3} & \cdots & 0 & 0 & 0 \\
    \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
    h_{m-4,0} & h_{m-4,1} & \cdots & h_{m-4,L-1} & 0 & h_{L-1,L+1} & h_{L-1,L+2} & \cdots & 1 & 0 & 0 \\
    h_{m-3,0} & h_{m-3,1} & \cdots & h_{m-3,L-1} & 0 & h_{L-1,L+1} & h_{L-1,L+2} & \cdots & h_{L-1,L+3} & I & 0 \\
    h_{m-2,0} & h_{m-2,1} & \cdots & h_{m-2,L-1} & 0 & h_{L-1,L+1} & h_{L-1,L+2} & \cdots & h_{L-1,L+3} & I & I \\
    \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
    h_{m-1,0} & h_{m-1,1} & \cdots & h_{m-1,L-1} & 0 & h_{L-1,L+1} & h_{L-1,L+2} & \cdots & h_{L-1,L+3} & 0 & I \\
\end{bmatrix} \]

The codeword \( w = [d_0, d_1, \ldots, d_{L-1}, p_0, p_1, \ldots, p_{m-1}] \) is known to be found where \( d_i \) and \( p_i \) are sub-vectors of length \( e \), which are given by \( d_i = [d_{i,0}, d_{i,1}, \ldots, d_{i,e-1}] \) and \( p_i = [p_{i,0}, p_{i,1}, \ldots, p_{i,m-1}] \) respectively.

THE HONG KONG POLYTECHNIC UNIVERSITY  
Prof. Francis C.M. Lau, EIE
"A code" constructed by the structure

row number  
column number

row weights of matrices $H_{b2}$

row weights of matrices $H_{b1}$

row weights of matrices $H$

column weights

THE HONG KONG
POLYTECHNIC UNIVERSITY

Prof. Francis C. M. Lau, EIE

The base parity-check matrix $H_b$ of a code following our proposed structure. Code rate is 0.5. The empty boxes represent zero sub-block matrices $0$. A non-negative integer $S = 0, 1, \ldots, z - 1$ denotes a circulant permutation matrix $P$ which is obtained by cyclic-right shifting the columns of an identity matrix $I$ $S$ times. $S = 0$ represents an identity matrix $I$. 

THE HONG KONG
POLYTECHNIC UNIVERSITY

Prof. Francis C. M. Lau, EIE

19

prof. francis c.m. lau, eie

20

10
Comparison of Column weights

<table>
<thead>
<tr>
<th>Column number</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
</tr>
</thead>
<tbody>
<tr>
<td>A code</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>B code</td>
<td>3</td>
<td>3</td>
<td>8</td>
<td>3</td>
<td>8</td>
<td>3</td>
<td>8</td>
<td>3</td>
<td>8</td>
<td>3</td>
<td>8</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>IEEE 802.16e</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Column weights of the base parity-check matrix $H_3$ used to construct the A code and the B code, and those for the rate 1/2 code used in the IEEE 802.16e standard.

---

Comparison of row weights

Row weights of matrices $H_{A1}$, $H_{A2}$ and $H_{B}$ for the constructed rate-1/2 A code and the IEEE 802.16e code:

<table>
<thead>
<tr>
<th>Row number</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_{A1}$ of A code</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$H_{A2}$ of A code</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>$H_{B}$ of A code</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

non-uniform row weights

<table>
<thead>
<tr>
<th>Row number</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_{A1}$ of 802.16e code</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>$H_{A2}$ 802.16e code</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$H_{B}$ 802.16e code</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

nearly-uniform row weights
Comparison of row weights

### B Code

**Row weights of matrices $H_{b1}$, $H_{b2}$ and $H_b$ for the constructed code of rate 1/2.**

<table>
<thead>
<tr>
<th>Row number</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_{b1}$</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>$H_{b2}$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>$H_b$</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>

- Non-uniform row weights
- Nearly-uniform row weights

**Row weights of matrices $H_{b1}$, $H_{b2}$ and $H_b$ for rate-1/2 IEEE 802.16e code.**

<table>
<thead>
<tr>
<th>Row number</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_{b1}$</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>$H_{b2}$</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$H_b$</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

- Nearly-uniform row weights

---

Error performance

- BLER
- BER
- A code
- B code

**Error rate vs. $E_b/N_0 (dB)$**

**802.16e code**

**A code**

**B code**
Factors/Requirements in Designing a Channel Code

- Error detection and/or error correction
- Channel type
- Error performance
- Transmission power
- Bandwidth
- Information rate
- Code rate
- Coding gain
- Data structure
- Development cost
- Latency
- Hardware complexity
Belief Propagation (BP) Iterative Decoding Algorithm

- Messages passing back and forth between variable nodes and check nodes
- Codes with large girth (smallest cycle length) are more desirable

Block structure of $\mathbf{H}$

$\mathbf{H}_{16 \times 20}$

expansion factor $z = 4$

What is the girth of this code?
The codes in order from highest to lowest error floor are LDPC codes of length 2640, 2048, 2640, and 8096.

Error Floor

Construction Methods
Construction Methods

- Progressive Edge-Growth Method
- Bit-Filling Method
- Guess-and-Test Method
- Hill-Climbing Method
- Array-Code Based Method
- Our Method

Progressive Edge-Growth Method

- The PEG algorithm constructs an LDPC code edge-by-edge
- In the creation of a new edge for a symbol node, the distances between the concerning symbol node and the candidate check nodes are evaluated to ensure that the largest local girth for the symbol node can be achieved

Subgraph spreading from symbol $s_j$ node
Bit-Filling Method

- The main idea is to add the new columns to the parity-check matrix such that cycles of length \((g-2)\) or smaller will not be created.

---

Bit-Filling Method

- Problem 1 (fixed girth, maximize code rate): Given the parameters for the parity-check matrix: column weight \(r\), maximum row weight \(q\), required girth \(g\) and the number of rows \(m\); construct a parity-check matrix \(H\) with the largest possible code rate (which is the largest possible number of columns \(n\)).
- Problem 2 (fixed code rate, maximize girth): Given the parameters for the parity-check matrix: column weight \(r\), maximum row weight \(q\), the number of the columns \(n\) and the number of rows \(m\); construct a parity-check matrix \(H\) with the largest possible girth.
Guess-and-Test Method

Consider a \((r, q)\)-regular QC-LDPC code, the parity-check matrix of which is given by

\[
H = \begin{bmatrix}
I & I & I & \ldots & I \\
I & p_{p,1} & p_{p,2} & \ldots & p_{p,q-1} \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
I & p_{r-1,1} & p_{r-1,2} & \ldots & p_{r-1,q-1}
\end{bmatrix}
\]

where \(I\) is an \(p \times p\) identity matrix and \(p_{p,2}\) represents an \(p \times p\) circulant permutation matrix.

**Theorem:** A necessary and sufficient condition for the Tanner graph representation of the matrix \(H\) to have girth at least \(2(L + 1)\) is

\[
\sum_{k=0}^{M-1} (p_{i_k,j_k} - p_{i_{k+1},j_{k+1}}) \neq 0 \mod p
\]

for all \(M, 2 \leq M \leq L\), all \(i_k, 0 \leq i_k \leq r - 1\), all \(i_{k+1}\),

\(0 \leq i_{k+1} \leq r - 1\), and all \(j_k, 0 \leq j_k \leq q - 1\), with

\(i_0 = i_m, i_k \neq i_{k+1}\), and \(j_k \neq j_{k+1}\).
Guess-and-Test Method

- **Problem (fixed code rate and girth, minimize p):**
  
  Given the parameters for the basic parity-check matrix: row weight $q$, column weight $r$, and desired girth $g$; construct an $rp \times qp$ parity-check matrix $H$ with the smallest possible $p$ value (which is the smallest possible number of columns $n$ and the smallest possible number of rows $m$) such that $H$ has girth $g$.

Guess-and-Test Method

- **Random Construction**
  
  - Given row weight $q$, column weight $r$ and desired girth $g$, randomly generate $(r-1)(q-1)$ integers $p_{ij}$ for the matrix $H$ until $g$ is achieved according to the Theorem.
  - Then find the smallest values of $p$ by computer search
Hill-Climbing Method

- Main idea: Start with a random basic parity-check matrix $H$, then iteratively change the value $p_{i,j}$ for a single element such that the weighted sum of the number of cycles with length less than the desired girth is minimized.

<table>
<thead>
<tr>
<th>$L$</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guess-and-test</td>
<td>9</td>
<td>14</td>
<td>18</td>
<td>21</td>
<td>26</td>
<td>33</td>
<td>39</td>
<td>46</td>
<td>54</td>
</tr>
<tr>
<td>Hill-climbing</td>
<td>9</td>
<td>13</td>
<td>18</td>
<td>21</td>
<td>25</td>
<td>30</td>
<td>35</td>
<td>41</td>
<td>47</td>
</tr>
</tbody>
</table>

UPPER BOUNDS ON $p_{\text{min}}$ FOR $g = 8$ AND $r = 3$ QC-LDPC CODES

Array-Code Based Method
Array-Code Based Method

- A typical array code can be represented by
  \[ H = \begin{bmatrix}
  I & I & I & \ldots & I \\
  I & P & P^2 & \ldots & P^{v-1} \\
  I & P^2 & P^4 & \ldots & P^{2(q-1)} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  I & P^{r-1} & P^{2(r-1)} & \ldots & P^{(r-1)(q-1)}
  \end{bmatrix} \]
  where \( I \) is an identity matrix, and \( P^v \) is a circulant permutation matrix obtained by cyclically shifting the rows of the identity matrix to the right by the shift size \( v \).

Array-Code Based Method

- General form of the array code
  \[ H = \begin{bmatrix}
  p_{a_0}^0 & p_{a_0}^1 & p_{a_0}^2 & \ldots & p_{a_0}^{q-1} \\
  p_{a_1}^0 & p_{a_1}^1 & p_{a_1}^2 & \ldots & p_{a_1}^{q-1} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  p_{a_{r-1}}^0 & p_{a_{r-1}}^1 & p_{a_{r-1}}^2 & \ldots & p_{a_{r-1}}^{q-1}
  \end{bmatrix} \]

- Proper array code (PAC): the sequence \( a_0, a_1, \ldots, a_{r-1} \) forms an arithmetic progression (A.P.), which is to say that there exists an integer \( a \neq 0 \) such that \( a_{i+1} - a_i = a \) for \( i = 0, 1, 2, \ldots, r-2 \).

- Improper array code (IAC): the sequence \( a_0, a_1, \ldots, a_{r-1} \), does not form an A.P.
Array-Code Based Method

- **A closed path of length 2k** in any parity-check matrix of the general form is a sequence of block-row and block-column index pairs 
  \((i_1, j_1), (i_1, j_2), (i_2, j_2), \ldots, (i_k, j_k), (i_k, j_1)\) with \(i_l \neq i_{l+1}\) and \(j_l \neq j_{l+1}\) for \(l=1, 2, \ldots, k-1\); and \(i_k \neq i_1\), and \(j_k \neq j_1\).

**Theorem:**

A cycle of length 2k exists in the Tanner graph of an array code with parity-check matrix \(H\) and block-row labels \(a_0, a_1, \ldots, a_{r-1}\) if and only if there exists a closed path 
\((i_1, j_1), (i_1, j_2), (i_2, j_2), \ldots, (i_k, j_k), (i_k, j_1)\)
in \(H\) such that
\[
\prod_{l=1}^{k} h_{i_l, j_l} (P^{a_{i_{l-1}}j_{l-1}})^{-1} P^{a_{i_l}j_{l-1}} (P^{a_{i_{l-1}}j_{l-1}})^{-1} \ldots P^{a_{i_{k-1}}j_{k-1}} (P^{a_{i_k}j_{k-1}})^{-1} = I
\]
evaluates to the identity matrix \(I\), which is equivalent to
\[
a_i (j_i - j_{i+1}) + a_{i+1} (j_{i+1} - j_i) + \ldots + a_k (j_k - j_1) = 0 \mod p i_i
\]
or
\[
\dot j_i (a_i - a_{i+1}) + \dot j_{i+1} (a_{i+1} - a_i) + \ldots + \dot j_k (a_k - a_{i-1}) = 0 \mod p
\]
Array-Code Based Method

Girth 6 (Free of Cycle of length 4)
- Based on the Theorem, it is easily found that array codes are free of cycles of length 4. This is because a cycle of length 4 exists if and only if there exist indices $i_1$, $i_2$, $j_1$, $j_2$, $i_1 \neq i_2$, $j_1 \neq j_2$ such that

$$\ (a_{i_1} - a_{i_2})(j_1 - j_2) \equiv 0 \mod p$$

- which is clearly impossible since $i_1 \neq i_2$ and $j_1 \neq j_2$

Cycles of length 6
- A closed path of length 6 in the parity-check matrix of an array code must pass through three different block rows and three different block columns.

- A PAC has cycle of length 6 if and only if there exist three different block rows, indexed by $r_1$, $r_2$, $r_3$, and three different block columns, indexed by $i$, $j$, $k$ such that

$$i(r_1 - r_3) + j(r_2 - r_1) + k(r_3 - r_2) \equiv 0 \mod p$$
Array-Code Based Method

Some cycles of length 6, and their governing equations

\[
(\eta - r_3)i + (r_2 - \eta)j + (r_3 - r_2)k = 0
\]
\[-2i + j + k = 0
\]

\[
(\eta - r_2)i + (2r_2 - \eta - r_3)j + (r_3 - r_2)k = 0
\]
\[-i + k = 0
\]
Array-Code Based Method

- Cycles of length 8
  - A PAC contains a cycle of length 8 if and only if its parity-check matrix contains a closed path of the form 
    \[(r_1, i), (r_1, j), (r_2, j), (r_2, k), (r_3, k), (r_3, l), (r_4, l), (r_4, i)\]
    such that
    \[i(\eta - r_4) + j(r_2 - \eta) + k(r_3 - r_2) + l(r_4 - r_3) \equiv 0 \mod p\]

- Note that closed paths of length 8 may pass through two, three, or four different block-columns of the parity-check matrix

---

Array-Code Based Method

Cycles of length 8
Array-Code Based Method

Cycles of length 8
(passing through 2 different columns)

In the case when \( k = i \) and \( l = j \), a closed path passes through two different block columns. Thus

\[ i(\eta - r_4) + j(r_2 - \eta) + k(r_3 - r_2) + l(r_4 - r_3) \equiv 0 \quad \text{mod} \quad p \]

becomes

\[ (i - j)(r_1 + r_3 - r_2 - r_4) \equiv 0 \quad \text{mod} \quad p \]

Since \( i \neq j \), it yields,

\[ \eta + r_3 - r_2 - r_4 \equiv 0 \quad \text{mod} \quad p \]

The above equation is always satisfied by taking \( r_1 = 0, r_2 = 1, r_3 = 2, r_4 = 1 \).
Array-Code Based Method

- Girth 8 (Free Cycle of length 4 and 6)
  - Theorem: Let C be a PAC with modulus p whose parity-check matrix, H, has column weight r. If r = 4, then C contains a cycle of length 6 if and only if there exists three distinct block columns in H whose labels \( i, j, k \) satisfy at least one of the following two congruences
    \[
    -2i + j + k = 0 \quad \text{mod } p \\
    -3i + j + 2k = 0 \quad \text{mod } p
    \]
  - If r = 3, then C contains a cycle of length six if and only if there exist three distinct block columns whose labels \( i, j, k \) satisfy the first of the two equalities.

Array-Code Based Method

- Cycle-governing equations over \( \mathbb{Z}_p \) for PACs with modulus \( p \) and column-weight \( r = 3 \), and greedy sequences avoiding solutions over \( \mathbb{Z}_{1213} \) to them

<table>
<thead>
<tr>
<th>Six-cycle equation</th>
<th>Greedy sequences avoiding the six-cycle equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 2i - j - k = 0 )</td>
<td>0, 1, 3, 4, 9, 10, 12, 13, 27, 28, 30, 38, ...</td>
</tr>
<tr>
<td></td>
<td>0, 2, 3, 5, 9, 11, 12, 14, 27, 29, 30, 39, ...</td>
</tr>
<tr>
<td></td>
<td>0, 3, 4, 7, 9, 12, 13, 16, 27, 30, 35, 36, ...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Eight-cycle equations</th>
<th>Greedy sequences avoiding all six- and eight-cycle equations</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 2i + j - k - 2l = 0 )</td>
<td>0, 1, 4, 11, 27, 39, 48, 84, 134, 163, 223, 284, 333, ...</td>
</tr>
<tr>
<td>( i + j - k - l = 0 )</td>
<td>0, 2, 5, 13, 20, 37, 58, 91, 135, 160, 220, 292, 354, ...</td>
</tr>
<tr>
<td>( 3i - j - 2k = 0 )</td>
<td>0, 3, 4, 13, 25, 32, 65, 92, 139, 174, 225, 318, 341, ...</td>
</tr>
<tr>
<td>( 2i - j - k = 0 )</td>
<td>0, 1, 4, 11, 27, 39, 48, 84, 134, 163, 223, 284, 333, ...</td>
</tr>
</tbody>
</table>
Array-Code Based Method

- Cycle-governing equations over $\mathbb{Z}_p$ for PACs with modulus $p$ and column-weight $r = 4$, and greedy sequences avoiding solutions over $\mathbb{Z}_{911}$ to them

<table>
<thead>
<tr>
<th>Six-cycle equations</th>
<th>Greedy sequences avoiding all six-cycle equations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2i - j - k = 0$</td>
<td>0, 1, 4, 5, 11, 19, 20, ...</td>
</tr>
<tr>
<td>$3i - j - 2k = 0$</td>
<td>0, 2, 3, 7, 13, 18, 20, ...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Eight-cycle equations</th>
<th>Greedy sequences avoiding all six- and eight-cycle equations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$3i - j - k - l = 0$</td>
<td>0, 1, 5, 18, 25, 62, 95, 148, 207, ...</td>
</tr>
<tr>
<td>$3i - 2j - 2k + t = 0$</td>
<td>0, 2, 7, 20, 45, 68, 123, 160, 216, ...</td>
</tr>
<tr>
<td>$2i - 2j - k + l = 0$</td>
<td>0, 3, 7, 22, 39, 68, 123, 154, 244, ...</td>
</tr>
<tr>
<td>$3i - 3j + k - l = 0$</td>
<td></td>
</tr>
<tr>
<td>$3i - 3j + 2k - 2l = 0$</td>
<td></td>
</tr>
<tr>
<td>$i + j - k - l = 0$</td>
<td></td>
</tr>
<tr>
<td>$2i - j - k = 0$</td>
<td></td>
</tr>
<tr>
<td>$4i - 3j - k = 0$</td>
<td></td>
</tr>
<tr>
<td>$3i - 2j - k = 0$</td>
<td></td>
</tr>
<tr>
<td>$5i - 3j - 2k = 0$</td>
<td></td>
</tr>
</tbody>
</table>

Shortened-Array-Code Based Method
Shortened-Array-Code Based Method

• **Block-column-shortened array code**: A code whose parity-check matrix is obtained by deleting a prescribed set of block-columns from the parity-check matrix of an array code

\[
H = \begin{bmatrix}
    p_{00} & p_{01} & p_{02} & \cdots & p_{0(q-1)} \\
p_{10} & p_{11} & p_{12} & \cdots & p_{1(q-1)} \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
p_{r0} & p_{r1} & p_{r2} & \cdots & p_{r(q-1)}
\end{bmatrix}
\]

Details of cycle-governing equations

**Insevitability of cycle-eight due to block-row indices**: For \(a_{g_1}, a_{g_2}, a_{g_3}, a_{g_4} \in \{a_0, a_1, \ldots, a_{r-1}\}\), cycle-eight must exist if the following equality holds,

\[
a_{g_1} + a_{g_2} - a_{g_3} - a_{g_4} = 0 \pmod{p}
\]

subject to \(a_{g_1} \neq a_{g_2}, a_{g_3} \neq a_{g_4}, a_{g_3} \neq a_{g_2}, a_{g_4} \neq a_{g_1}\)  \(1\)

Once the set of block-row indices \(\{a_{g_1}, a_{g_2}, \ldots, a_{r-1}\}\) has been selected, the block-column indices can be regarded as variables and the cycle-governing equations can be written as

\[
\sum_{i=1}^{k} c_i x_i \equiv 0 \pmod{p}, \quad (2)
\]

where the coefficient \(c_i = a_{g_i} - a_{g_{i+1}}\) if \(i = 1\) and \(c_i = a_{g_{i-1}} - a_{g_i}\) if \(i \neq 1\). Hence, the equation \(\sum_{i=1}^{k} c_i = 0\) is always satisfied.
### Details of cycle-governing equations

**Any solution** \( x = (x_1, x_2, \ldots, x_k) \) to \( \sum_{i=1}^{k} c_i x_i \equiv 0 \pmod{p} \) with \( x_i \in \{0, 1, \ldots, p-1\} \) and \( x_i \neq x_j \) for all \( i \neq j \) is referred to as a **proper solution** over \( Z_p \), where \( Z_p \) is the ring of integers modulo \( p \).

In the construction of shortened array codes, we aim to find a subset of block-column indices \( S(p; \Omega) \subseteq \{0, 1, \ldots, p-1\} \) such that \( S(p; \Omega) \) contains no proper solutions to a system of cycle-governing equations denoted by \( \Omega \).

We further denote by \( s(p; \Omega) \) the maximum number of columns retained in a shortened array code.
Theorems

Theorem 2: (General Lower Bound): Given a system of cycle-governing equations $\Omega$ up to cycle-2$k$, there exists a sequence $s_1, s_2, \ldots, s_n$ with $1 = s_1 < s_2 < \cdots < s_n \leq |\Omega|k(n-1)^{k-1}$ such that $S(p; \Omega) = \{s_1, s_2, \ldots, s_n\}$ does not contain proper solutions to $\Omega$. Then $s(p; \Omega)$ for shortened array codes of girth-2$(k+1)$ is lower-bounded by $s(p; \Omega) \geq p^{\frac{1}{k+1}}(|\Omega|k)^{-\frac{1}{k+1}}$.

Theorem 3: (Upper Bound of $s(p; \Omega)$ for shortened array codes of girth-eight) Let $\Omega$ denote the system of cycle-six governing equations in the form of

$$\Omega: (c_{m,1} + c_{m,2})x_3 \equiv c_{m,1}x_1 + c_{m,2}x_2 (\text{mod } p), m = 1, \ldots, |\Omega|. \quad (1)$$

Then $s(p; \Omega)$ is upper-bounded by $s(p; \Omega) \leq p^{(\log \log p)^{-c(\nu)}}$, where $c(\nu) = 2^{-2^{\nu+10}}$ and $\nu = \min_{m \in \{1, \ldots, |\Omega|\}} \{c_{m,2} + c_{m,1}\}$.

Theorems

Corollary 1: For shortened array codes of girth-ten or higher, $s(p; \Omega)$ is upper-bounded by $s(p; \Omega) \leq \sqrt{6p}$ and $s(p; \Omega) = O(\sqrt{p})$.

Theorem 4: Given a column-weight-three array code, $s(p; \Omega)$ for shortened array codes of girth-ten is lower bounded by $s(p; \Omega) \geq \beta \sqrt{pe^{-\alpha \sqrt{\log p}}}$ for some positive $\beta$ and $\alpha$. 
Constructing high-rate QC-LDPC codes of girth-10 (Proposed algorithm)

Algorithm - Proposed Code Construction Method

Let $V = \max_{m=1,2,\ldots,q} \{c_{m,1} + c_{m,2}\}$. For the largest prime number $q$ with $q < \sqrt{V}$, do the following steps:

1. Construct a set $\mathcal{X} = \{x' + Vqx : 0 \leq x \leq q - 1, x' = x^2 \pmod{q}\}$.

2. Initialize $S(p; \Omega)$ as the largest subset of $\mathcal{X}$ that avoids proper solutions to cycle-governing equations in four variables. ($\mathcal{X}$ is guaranteed to avoid proper solutions to all cycle-governing equations in three variables.)

3. Update $S(p; \Omega)$ by sequentially adding the integers in $[0, p - 1] \setminus S(p; \Omega)$ that would not create proper solutions to $\Omega$.

Constructing high-rate QC-LDPC codes of girth-10 (Based on theory)

$$
H = \begin{bmatrix}
P^{a_0} \cdot x_1 & \ldots & P^{a_0} \cdot x_q \\
P^{a_1} \cdot x_1 & \ldots & P^{a_1} \cdot x_q \\
P^{a_2} \cdot x_1 & \ldots & P^{a_2} \cdot x_q \\
\end{bmatrix}
$$

First $q$ columns found by theory

Other columns found by searching and testing
Constructing high-rate QC-LDPC codes of girth-10 (Greedy algorithm)

\[ H = \begin{bmatrix} P^{a_0} & \cdots & \cdots & \cdots & \cdots \\ P^{a_1} & \cdots & \cdots & \cdots & \cdots \\ P^{a_2} & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \]

All columns found by searching and testing

Table 1: Comparison of the minimum \( p \) required to construct girth-ten shortened array codes for different code rates \( R \) using the greedy code construction (\( \text{P_{greedy}} \)) and our proposed method (\( \text{P_{proposed}} \)). The random construction method involves 500 iterations. The column weight of the codes is \( r = 3 \) and the block-row indices is \( \{0, 1, 3\} \).

<table>
<thead>
<tr>
<th>No. of columns</th>
<th>Code Rate</th>
<th>Minimum ( p ) required</th>
<th>Random Code Construction</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>( R = 0.727 )</td>
<td>( \text{P_{proposed}} = 911 )</td>
<td>( p = \text{P_{proposed}} : R = 0.727 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{P_{greedy}} = 1039 )</td>
<td>( p = \text{P_{greedy}} : R = 0.77 )</td>
</tr>
<tr>
<td>12</td>
<td>( R = 0.75 )</td>
<td>( \text{P_{proposed}} = 1319 )</td>
<td>( p = \text{P_{proposed}} : R = 0.77 )</td>
</tr>
<tr>
<td>13</td>
<td>( R = 0.77 )</td>
<td>( \text{P_{proposed}} = 1723 )</td>
<td>( p = \text{P_{proposed}} : R = 0.786 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{P_{greedy}} = 1669 )</td>
<td>( p = \text{P_{greedy}} : R = 0.77 )</td>
</tr>
<tr>
<td>14</td>
<td>( R = 0.786 )</td>
<td>( \text{P_{proposed}} = 1767 )</td>
<td>( p = \text{P_{proposed}} : R = 0.786 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{P_{greedy}} = 2179 )</td>
<td>( p = \text{P_{greedy}} : R = 0.8 )</td>
</tr>
<tr>
<td>15</td>
<td>( R = 0.8 )</td>
<td>( \text{P_{proposed}} = 2570 )</td>
<td>( p = \text{P_{proposed}} : R = 0.8 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{P_{greedy}} = 2797 )</td>
<td>( p = \text{P_{greedy}} : R = 0.8125 )</td>
</tr>
<tr>
<td>16</td>
<td>( R = 0.8125 )</td>
<td>( \text{P_{proposed}} = 2999 )</td>
<td>( p = \text{P_{proposed}} : R = 0.824 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{P_{greedy}} = 2971 )</td>
<td>( p = \text{P_{greedy}} : R = 0.8125 )</td>
</tr>
<tr>
<td>17</td>
<td>( R = 0.824 )</td>
<td>( \text{P_{proposed}} = 3440 )</td>
<td>( p = \text{P_{proposed}} : R = 0.833 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{P_{greedy}} = 3407 )</td>
<td>( p = \text{P_{greedy}} : R = 0.824 )</td>
</tr>
<tr>
<td>18</td>
<td>( R = 0.833 )</td>
<td>( \text{P_{proposed}} = 3823 )</td>
<td>( p = \text{P_{proposed}} : R = 0.833 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{P_{greedy}} = 4079 )</td>
<td>( p = \text{P_{greedy}} : R = 0.842 )</td>
</tr>
<tr>
<td>19</td>
<td>( R = 0.842 )</td>
<td>( \text{P_{proposed}} = 4493 )</td>
<td>( p = \text{P_{proposed}} : R = 0.842 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \text{P_{greedy}} = 4961 )</td>
<td>( p = \text{P_{greedy}} : R = 0.85 )</td>
</tr>
</tbody>
</table>
What you have learnt?

- Block-structured LDPC codes
  - Low encoder/decoder complexity
  - Data portion
  - Parity portion
  - Quasi-cyclic LDPC codes

- Code-Construction Methods

Reading