Column-filling Scheme for Luby-Transform Codes

Coding & Crypto Lab.
YONSEI UNIVERSITY
남미영, 송홍엽
{my.nam, hysong}@yonsei.ac.kr
FOUNTAIN CODES
SYSTEM MODEL

- **NOTATION**
  - \( k \): number of information symbols
  - \( n \): number of output symbols
  - \( \varepsilon = \frac{n}{k} - 1 \): overhead
  - \( H, |H| \): binary encoding matrix and the number of 1's in \( H \)
  - Complexity: number of the edges of the Tanner graph of LT code

- **Binary Erasure Channel**

  ![Binary Erasure Channel Diagram]

  GOAL:
  - Universality - minimize the overhead
  - Efficiency - minimize the complexity
LT CODES

- Invented by M. Luby in 1998

- Output symbols should be generated by simple distribution – Robust Soliton Distribution

- First class of universal and almost efficient Fountain Codes

- Encoding and decoding are very simple
LT CODES

- WER performance of an LT code with $k=200$ using Maximum Likelihood Decoding Algorithm

![Graph showing WER performance with MLDA and lower bound for different overhead $\gamma$.](image)
Encoding of LT Codes

**Algorithm 1** A general LT encoding algorithm

1: repeat
2: choose a degree $d$ from degree distribution $\rho(d)$.
3: choose uniformly at random $d$ input symbol blocks $m_{i_1}, \ldots, m_{i_d}$.
4: send $m_{i_1} \oplus m_{i_2} \oplus \cdots \oplus m_{i_d}$.
5: until enough output symbols are received.
The output symbol distribution follows RSD

The input symbol should be selected uniformly at the same time
- The number of selection of each input symbol should be equal.

GOOD PRNGs are needed
- Guarantee perfect uniformity
- Allow flexible selection range
The negative influence of non-uniform column weight distribution (k=200, using MLDA)
LT CODES

**WHY?**

- Null column effect – sometimes some input symbols never be chosen
  - These are never recovered

<table>
<thead>
<tr>
<th>overheads</th>
<th>Null columns / Frame</th>
<th>overheads</th>
<th>Null columns / Frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.0796460177</td>
<td>0.04</td>
<td>0.0466507177</td>
</tr>
<tr>
<td>0.01</td>
<td>0.0821256039</td>
<td>0.05</td>
<td>0.0439461883</td>
</tr>
<tr>
<td>0.02</td>
<td>0.0714285714</td>
<td>0.06</td>
<td>0.0351380423</td>
</tr>
<tr>
<td>0.03</td>
<td>0.0501138952</td>
<td>0.07</td>
<td>0.0368344274</td>
</tr>
</tbody>
</table>
**LT CODES**

- decrease the full-rank probability
  
  ✓ Example for $|H| = 10$, $k = 5$, $n = 6$
  
  row-weights = [2,2,1,3,1,1]
  
  column-weights = [2,2,2,2,2] vs. [4,3,1,1,1]

```
(a) Column indices ➔ 1 2 3 4 5
1 2 2 2 2
2 1 2 2 2
3 1 1 1 1 1
4 1 1 1 1 1
5 0 1 1 1 1
6 0 1 1 1 1

(b) Column indices ➔ 1 2 3 4 5
1 4 2 3 2
2 3 2 2 2
3 2 1 1 1
4 1 1 1 1
5 0 1 1 1
6 0 1 1 1
```

```
(a) Row indices ➔ 1 2 3 4 5
1 1 1
2 0 1
3 0 0 0
4 1 1 1
5 0 1 1
6 0 1 1

(b) Row indices ➔ 1 2 3 4 5
1 1 1
2 1 1 1
3 1 1 1
4 1 1 1
5 0 1 1
6 0 1 1
```
Proposed algorithm

- Just counting and control the number of selection of input symbols
- No need for the PRNG guarantees very very uniform selection

Algorithm 2

An LT encoding algorithm together with the column-filling

1: repeat
2:   choose a degree $d$ from degree distribution $\mu(d)$.
3:   for $j = 1$ to $j = d$
4:     repeat
5:       $count = count + 1$
6:       choose an input symbol block $m_{ij}$ at random.
7:       if $cw[i_j] \leq C_i$
8:         $cw[i_j] = cw[i_j] + 1$
9:         break
10:     end if
11:     until $count < K$
12:   end for
13: send $m_{i_1} \oplus m_{i_2} \oplus \cdots \oplus m_{i_d}$.
14: until enough output symbols are received.
LT CODES

- Performance comparison

![Diagram showing performance comparison of LT codes with different overheads and thresholds. The x-axis represents overhead γ, and the y-axis represents WER. The graph illustrates how performance improves with lower overhead and higher thresholds.]
## LT CODES

- **Fraction of null columns**

<table>
<thead>
<tr>
<th>$C_i$</th>
<th>$\mathcal{E}$</th>
<th>0.00</th>
<th>0.01</th>
<th>0.02</th>
<th>0.03</th>
<th>0.04</th>
<th>0.05</th>
<th>0.06</th>
<th>0.07</th>
</tr>
</thead>
<tbody>
<tr>
<td>no</td>
<td></td>
<td>0.0796</td>
<td>0.0821</td>
<td>0.0714</td>
<td>0.0501</td>
<td>0.0466</td>
<td>0.0439</td>
<td>0.0351</td>
<td>0.0368</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>0.0248</td>
<td>0.0171</td>
<td>0.0168</td>
<td>0.0134</td>
<td>0.0080</td>
<td>0.0063</td>
<td>0.0047</td>
<td>0.0037</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>0.0082</td>
<td>0.0037</td>
<td>0.0010</td>
<td>0.0002</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
</tbody>
</table>
LT CODES

- Complexity comparison
  - The case of $C_t = 6, k = 200, \varepsilon = 0$
CONCLUSION

- The uniform column weight distribution of the encoding matrix for LT Codes is the best for the performance.
- The proposed column-filling scheme is simple way to guarantee the uniform selection of input symbols.
- However there exists some latency for applying the column-filling scheme.
- More efficient way making the uniform distribution should be investigated.
THANK YOU!

{my.nam, hysong} @ yonsei.ac.kr