LDPC Code Length Reduction

R. Borkowski, R. Bonk, A. de Lind van Wijngaarden, L. Schmalen – Nokia Bell Labs
B. Powell – Nokia Fixed Networks CTO Group

IEEE P802.3ca 100G-EPON Task Force Meeting, Orlando, FL, November 2017
Introduction

- During the last several meetings, several LDPC and RS codes have been proposed. The preferred length is between 2 kB and 4 kB. Current proposals focus on length-18493 LDPC codes, e.g., in May 2017, an [11×74×256] LDPC code was proposed, which was shortened to an (18493,15677) LDPC code (see [laubach_3ca_1_0517]).
- In previous meetings, a preference was expressed for using the same FEC code for upstream and downstream (for symmetry, and to simplify implementation and testing).
- The burst quantization unit, neglecting any overhead, is limited to 18493/25 Gbit/s = 739.7 ns.
- One contribution [laubach_3ca_1_0317], considered the aggregate throughput in the upstream as a function of the burst size (200-20,000 bytes payload), and concluded that 20Gb/s throughput could be achieved for bursts of 12,000 bytes or longer by shortening the (last) LDPC codeword.
- Further analysis is needed to determine the code performance and rate for shorter burst lengths:
  - Low-latency service requirements
  - Minimum Ethernet packet size
  - Efficient method of sending US ONT queue reports
- A rate of 0.848 is needed to support 2×10G (net rate) channels within a 25G channel.
- FEC input BER 10^{-2}, post-FEC error floor <10^{-12} [laubach_3ca_3_0317]
Shortening and Puncturing

- A code can be simultaneously shortened and punctured to maintain the same code rate while reducing transmitted codeword length
  - Shortening inserts 0’s in place of some data bits, and these bits are not transmitted
  - Puncturing omits sending some of the codeword bits

- In a binary symmetric channel (BSC):
  - LDPC shortening improves performance because the log-likelihood ratio (LLR) of removed bits is set to a high value at the decoder, and they can be forced to remain 0’s (i.e., shortened bits are certain)
  - LDPC puncturing degrades performance because the LLR of removed bits is set to a low value at the decoder (i.e., punctured bits are treated as erasures)
Shortening and Puncturing

Example - Proposed Broadcom LDPC matrix [11×74×256]

Natural codeword
- 2816 parity bits
- 16128 information bits

Simultaneous shortening and puncturing. 100% codeword: $s = 451$, $p = 0$ [aubach_3ca_1_0517] (only shortening). Further length reduction obtained by increasing $s$ and $p$ at constant rate.

- $\rho$ punctures
- $2816-\rho$ parity bits
- 16128- $s$ user bits
- $s$ shortened bits

Enc: discarded
Dec: erasures (LLR=0)

Transmitted codeword
Enc: zeros
Dec: zeros (LLR=$\infty$)
Shortening and Puncturing

Simulation Details

- Shortening from the rightmost matrix side by zeros \(\rightarrow\) dense matrix part improves performance of the shortened code as certainty of shortened bits is shared across multiple equations.
- Puncturing from the leftmost matrix side \(\rightarrow\) matrix already permuted so that consecutive erasures from the left will be distributed across multiple equations.
- Number of iterations: 5, 10 and 15
- BER calculated over data bits only
- Length reduced from 100\%=739.7 ns (original code from laubach_3ca_1_0517) in steps of 1/10 while keeping constant rate of 0.848

<table>
<thead>
<tr>
<th>%original length</th>
<th>100%</th>
<th>90%</th>
<th>80%</th>
<th>70%</th>
<th>60%</th>
<th>50%</th>
<th>40%</th>
<th>30%</th>
<th>20%</th>
<th>10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Burst duration</td>
<td>(t,\text{ ns})</td>
<td>739.7</td>
<td>665.8</td>
<td>591.8</td>
<td>517.8</td>
<td>443.8</td>
<td>369.9</td>
<td>295.9</td>
<td>221.9</td>
<td>148.0</td>
</tr>
<tr>
<td>Transmitted bits</td>
<td>(n')</td>
<td>18493</td>
<td>16644</td>
<td>14795</td>
<td>12946</td>
<td>11096</td>
<td>9247</td>
<td>7398</td>
<td>5548</td>
<td>3699</td>
</tr>
<tr>
<td>Information bits</td>
<td>(k')</td>
<td>15677</td>
<td>14110</td>
<td>12543</td>
<td>10975</td>
<td>9407</td>
<td>7839</td>
<td>6272</td>
<td>4704</td>
<td>3136</td>
</tr>
<tr>
<td>Parity bits</td>
<td>(r')</td>
<td>2816</td>
<td>2534</td>
<td>2252</td>
<td>1971</td>
<td>1689</td>
<td>1408</td>
<td>1126</td>
<td>844</td>
<td>563</td>
</tr>
<tr>
<td>Rate = (k'/n')</td>
<td>(\text{rate})</td>
<td>0.848</td>
<td>0.848</td>
<td>0.848</td>
<td>0.848</td>
<td>0.848</td>
<td>0.848</td>
<td>0.848</td>
<td>0.848</td>
<td>0.848</td>
</tr>
<tr>
<td>Shortened bits</td>
<td>(s)</td>
<td>451</td>
<td>2018</td>
<td>3585</td>
<td>5153</td>
<td>6721</td>
<td>8289</td>
<td>9856</td>
<td>11424</td>
<td>12992</td>
</tr>
<tr>
<td>Punctured bits</td>
<td>(p)</td>
<td>0</td>
<td>282</td>
<td>564</td>
<td>845</td>
<td>1127</td>
<td>1408</td>
<td>1690</td>
<td>1972</td>
<td>2253</td>
</tr>
</tbody>
</table>
Simulation Results

- Computer simulations are currently limited to BER $10^{-8}$–$10^{-9}$
- Error floors appear for codewords shorter than or equal to 70% (517.8 ns) of the original size
- At 80% and 90% of the original length, the codeword floors below $10^{-8}$ but it is uncertain whether the net coding gain is sufficient
Conclusions

- Puncturing is non-trivial; we were not able to shorten the LDPC(18493,15677) code below 80% of the original length without loss of performance.

- If one solely relies on shortening, the error correction performance is maintained, but the code rate and achievable throughput become very low, not accounting for additional factors like laser switch on/off time and sync time. Interleaving complicates shortening.

- If one avoids short bursts by waiting for more data, this significantly increases latency and it may introduce additional jitter (data may need to wait in the buffer before a sufficient amount of user data for one codeword is available).

- It is believed that RS codes are more well behaved when shortened, as it is easier to adjust the number of parity symbols for a given number of information symbols; one can thus operate at a higher code rate and ensure the avoidance of an error floor for shorter code lengths.