Progress in Binary and Non-Binary Low Density Parity Check Codes
Philipp Schläfer from University of Kaiserslautern
Sponsored by NASA-GSFC
Introducing myself

**Qualification:**
2011: Diploma in electrical engineering
Since 2011: Ph.D. student in a joint project of the university of Kaiserslautern at the chair of professor Wehn and Creonic GmbH.

**Fields of research:**
Very high throughput binary LDPC decoders
Efficient architectures for Non-Binary LDPC decoders
Outline

Ultra High Speed Binary Low Density Parity Check Decoders

Progress in Non-Binary Low Density Parity Check Decoding

Maximum Likelihood and Soft-Reed-Solomon Decoding
Ultra High Speed Binary Low Density Parity Check Decoders
Low Density Parity Check Decoding

Low Density Parity Check (LDPC) codes are linear block codes. They are defined by a sparse parity check matrix $H$. Valid codewords have to satisfy $H \cdot x^T = 0$.

Hardware architectures use Belief-Propagation (BP) algorithms for the decoding. Soft messages are iteratively exchanged between Check Nodes (CN) and Variable Nodes (VN) until a valid codeword is detected.

Node complexity is rather low. E.g. CN of degree 5 needs to find the first two minima from 4 soft messages.
LDPC Min-Sum Decoding Algorithm

Set of CNs connected to VN n: $\mathcal{M}(n) = \{m | m \in \{0, ..., M - 1\} \land H_{mn} \neq 0\}$

Set of VNs connected to CN m: $\mathcal{N}(m) = \{n | n \in \{0, ..., N - 1\} \land H_{mn} \neq 0\}$

**CN processing:**

$$\text{sgn} \left( \epsilon_{m \rightarrow n}^{(i)} \right) = \prod_{n' \in \mathcal{N}(m) \setminus n} \text{sgn} \left( z_{n' \rightarrow m}^{(i)} \right)$$

$$|\epsilon_{m \rightarrow n}^{(i)}| = \gamma \times \min_{n' \in \mathcal{N}(m) \setminus n} \left( |z_{n' \rightarrow m}^{(i)}| \right)$$

**VN processing:**

$$\Lambda_{n}^{(i)} = \lambda_{n}^{ch} + \sum_{m' \in \mathcal{M}(n)} \epsilon_{m' \rightarrow n}^{(i)}$$

$$z_{n \rightarrow m}^{(i)} = \lambda_{n}^{ch} + \sum_{m' \in \mathcal{N}(n) \setminus m} \epsilon_{m' \rightarrow n}^{(i-1)} = \Lambda_{n}^{(i-1)} - \epsilon_{m \rightarrow n}^{(i-1)}$$
How to Increase Throughput?

**Use multiple slow decoders**

- **PRO**
  - Easy to implement
- **CON**
  - Low efficiency
  - Large memory
  - Large latency

**Use monolithic high speed decoder**

- **PRO**
  - Higher efficiency
  - Lower latency
- **CON**
  - Challenging architecture
Increasing the Parallelism

**Fully parallel architecture:**
High throughput, e.g. 10 GBASE-T standard.

No flexibility in means of block length or code rate.

Routing congestion problems (>50% area) due to two overlapping networks.

Pipelining limited due to delay penalty in iterative loop.

Throughput limited by iterative data exchange and routing congestion.
Unrolled LDPC decoder:
Unrolling the decoding iterations for increased throughput.

Deep pipelining possible for high clock frequency without penalty.

Largely reduced routing complexity as only one network between VNs and CNs is required.

High implementation efficiency (throughput / area) and very high throughput.
Multi-Gigabit LDPC Decoder Implementation

Unrolled LDPC decoder features: [1]
IEEE 802.11ad standard
672 bit code word length
Code rate 13/16
One code word per clock cycle
9 iterations with two-phase schedule
4 bit quantization
105 ns latency
12 mm$^2$ core area (in 65nm technology)
160 Gbit/s throughput

Progress in Non-Binary Low Density Parity Check Decoding
Non-Binary LDPC Codes

Non-Binary (NB) LDPC codes are processing Galois Field (GF(q)) symbols instead of single bits.

The basic concepts of binary LDPC stay the same for decoding of NB LDPC codes.

BP decoding can be applied but instead of one soft message a vector of \( q \) soft messages is exchanged between the nodes.

The decoding complexity of VNs and CNs increases significantly. E.g. CN of degree 5 and GF(64) must find the first 64 minima from \( 64^4 = 16777216 \) possibilities.
NB LDPC codes show an enhanced communications performance.

Compared to state-of-the-art binary LDPC codes, the communications performance can be enhanced by 0.5 up to several dB by NB LDPC codes [1][2][3].

Especially for short blocks the gain is significant which makes NB LDPC codes interesting for latency critical applications.

Benefits of Non-Binary LDPC (2)

Modulation symbols represent a number of bits. By mapping modulation symbols to independent bits for binary decoding, the information concerning their correlation is lost.

NB-LDPC decoding avoids bit interpolation by direct processing of symbols. Dependent on the modulation scheme additional performance gains up to several dB have been observed [1][2][3].

At the same time the complexity of the demapper is reduced [2].

Decoding Algorithms for NB LDPC

**Exact:**
- Sum Product Algorithm (SPA) [1]
- FFT-Belief Propagation [2]
- Integer Linear Programming [3]

**Approximate:**
- Extended Min-Sum (EMS) [4]
- Min-Max [5]
- Symbol Flipping [6]
- Stochastic Decoding [7]

The EMS Decoding Algorithm

At high level EMS is like binary Min-Sum.

Messages are iteratively passed between VNs and CNs until a valid codeword is found.

Instead of binary messages, a GF(q) vector is transferred on each edge. EMS reduces the size of the GF(q) vector from q messages to nm messages with negligible loss in communications performance. E.g. only the most reliable messages are exchanged between VNs and CNs.

The most complex task in EMS decoding is the Check Node processing.

CN calculation in GF(q):

Symbol:
\[ \forall i,j,k,l \in q: out_i = in^j_i \oplus in^k_i \oplus in^l_i = i \]

LLR:
\[ \forall i,j,k,l \in valid(q): out_i = \min \{ in^j_i + in^k_i + in^l_i \} \]
Forward Backward (FWBW) processing is the state-of-the-art check node algorithm.

With a divide and conquer approach using Elementary Check Nodes (ECN) the most reliable messages for each outgoing edge are computed. Each ECN considers two GF(q) vectors. The intermediate results are combined in a smart way to generate the output vectors.

The FWBW scheme allows for small hardware implementations but suffers from low throughput and high latency.
The ECN transforms two input GF(q) vectors U and V into one output GF(q) vector consisting of the most reliable symbol combinations. For EMS the vector size is reduced from q elements to only nm < q elements.

The search space (nm² possibilities) can be represented by a matrix showing all possible combinations of the input symbols. Each field in the matrix can be computed as the sum of the according input reliabilities and GF values.

Under the assumption that the input vectors are sorted according to their reliability (e.g. U₀ and V₀ have the highest reliability), the most reliable values in the matrix are located left from and above the red line. Bubble Check [1] is an efficient implementation for the ECN. It executes a smart exploration of the matrix.

Several new directions are investigated to achieve more efficient architectures:

**Symbol parallel FWBW check node**
Same divide and conquer approach like FWBW.
Process N symbols per clock cycle, decoding speedup of factor N.
Minimal hardware increase in CN and VN to support approximately sorted messages.

**Multi edge check node**
Direct processing of multiple input vectors instead of FWBW approach.
Apply the Bubble Check algorithm in more than two dimensions.
Significantly reduced CN delay but high hardware cost for high CN degrees.

**Truncated T-EMS check node**
Efficient algorithmic transformation of the EMS algorithm for high CN degrees. The original algorithm requires full GF(q) vectors. We are currently investigating what is necessary to reduce the vector size.
Maximum Likelihood and Soft-Reed Solomon Decoding
Integer Programming Decoding [1]

Decoding as optimization problem:

Objective function:

\[ \min \sum_{i=0}^{N} \lambda_i x_i \quad x_i \in \{0,1\} \]

Constraints on \( x \):
ensures that bitvector \( x \)
is a code word

\[ s.t. \quad H x - 2z = 0 \]

Variables \( x \) are bits -> integer program, IP (hard to solve!)
Real ML Decoder, not a heuristic
ML Monte Carlo Simulation of channel codes possible

Efficient Maximum Likelihood Decoding

Efficient Linear Programming Decoding

Integer Program (IP Formulation)

Our optimized solver algorithm:
10x faster than CPLEX

Solver uses: branch & bound

Database for Simulation Results:
www.uni-kl.de/channel-codes

16.03.2014
Algorithm: Information Set Decoding

Variant: Ordered-Statistics Decoding

We use: Low Complexity Version [1]

Hard Decision Dec. added for better FER

1) **Sorting:**
   determine the $M$ least reliable bit positions (LRP)

2) **Gaussian Elimination:**
   diagonalize $\mathbf{H}$ at the LRP to obtain $\hat{\mathbf{H}}$

3) calculate the syndrome $\hat{s} = \hat{\mathbf{H}}\hat{y}^T$
   and its binary weight $wgt(\hat{s})$

4) If $wgt(\hat{s}) > \Theta$: /* MRPE errors */
   flip the received bit at position
   $j = \arg\min_{i=0,\ldots,N-1} wgt(\hat{s} \oplus \hat{h}_i)$
   update the syndrome $\hat{s} = \hat{s} \oplus \hat{h}_j$ and $wgt(\hat{s})$, goto 5

5) If $wgt(\hat{s}) \leq \Theta$: /* only LRP errors remaining */
   For all $\hat{s}_i = 1$, flip the LRP 1, for which $\hat{h}_{il} = 1$
   output OSD result, terminate
   else
   perform HDD on $\hat{y}$ and output HDD result, terminate

Soft Decision Decoding for Reed-Solomon Codes (2)

New Hardware Architecture

Information Set Decoding

Outstanding gain in FER over HDD
- RS(255,239): 0.75 dB gain
- RS(63,55): 1.4 dB gain

Hardware complexity lower than comparable state-of-the-art soft decoders

16.03.2014
FPGA implementations

State-of-the-art soft decoder RS(255,239)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>Adaptive BP</td>
<td>Stochastic Chase Information Set</td>
<td></td>
</tr>
<tr>
<td>Communications Gain</td>
<td>0.75 dB</td>
<td>0.7 dB</td>
<td>0.75 dB</td>
</tr>
<tr>
<td>FPGA</td>
<td>Stratix II</td>
<td>Virtex 5</td>
<td>Virtex 5</td>
</tr>
<tr>
<td>Size in FPGA LUT</td>
<td>43,700</td>
<td>117,000</td>
<td>13,700</td>
</tr>
<tr>
<td>Throughput</td>
<td>4 Mbit</td>
<td>50 Mbit/s</td>
<td>805 Mbit/s</td>
</tr>
</tbody>
</table>

Currently developed: Improved Architecture with a gain of 1.3 dB

[1]: Kan et al. “Hardware implementation of soft-decision decoding for Reed-Solomon code”, IntTurbo Codes and Related Topics Symp 2008
Backup
The area and energy efficiency (throughput / area and energy/bit) of state-of-the-art NB LDPC decoders is still orders of magnitude less than their binary counterparts.

Ongoing research is significantly reducing this difference.

The gain in communications performance is up to several dB.

<table>
<thead>
<tr>
<th></th>
<th>[1]</th>
<th>Our work</th>
</tr>
</thead>
<tbody>
<tr>
<td>GF(q)</td>
<td>64</td>
<td>2(binary)</td>
</tr>
<tr>
<td>Algorithm</td>
<td>EMS</td>
<td>Min-Sum</td>
</tr>
<tr>
<td>Throughput</td>
<td>1150 Mbit/s</td>
<td>1560 Mbit/s</td>
</tr>
<tr>
<td>Area</td>
<td>7 mm$^2$</td>
<td>1.8 mm$^2$</td>
</tr>
</tbody>
</table>