Implementation of LDPC Decoders

Mohammad M. Mansour

ECE Department
American University of Beirut
Beirut, Lebanon

Email: mmansour@aub.edu.lb

Invited Paper

IEEE Communication Theory Workshop
Park City, UTAH
June 12-15, 2005
Outline

- Preliminaries
- Implementation challenges of current LDPC decoders
- New design methodology for LDPC codes and decoders
  - Code design: Architecture-aware (AA) LDPC code design
  - Algorithms: Turbo-decoding of LDPC codes, simple message computation algorithm
  - Architecture: Advanced scalable/programmable/tunable LDPC decoding platform
  - Physical design: Core-generator for LDPC decoders
- LDPC decoder chip implementation
- Performance results
- Generalizations to Repeat-Accumulate Codecs
- Conclusions
LDPC Codes

- Introduced by Gallager in ’63, “rediscovered” at least 3 times after turbo codes
- LDPC codes are linear block codes defined as the null-space of a sparse parity-check matrix $H_{m \times n}$.
  - Regular $(c, r)$-LDPC: $H$ has exactly $c$ 1’s per column, $r$ 1’s per row
  - Irregular $(C, R)$-LDPC: Distribution of 1’s is drawn from $C$ and $R$
- Encoding complexity: quadratic in code length
- Decoding: Efficient sub-optimal iterative two-phase message-passing (TPMP) algorithm
- Reappeared as irregular repeat-accumulate (IRA) codes [1]

\[
H = \begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
\end{bmatrix}
\]
Gallager’s Two-Phase Message-Passing (TPMP) Algorithm

- TPMP algorithm is an iterative decoding algorithm consisting of two phases of computations that improve the log-likelihood estimate $\ln \left( \frac{P_i(1)}{P_i(0)} \right)$ of each bit:

  - **Phase 1**: Bit-to-check messages ($Q$) proportional to the prob. that a bit is 1 given that it satisfies the other checks.

  - **Phase 2**: Check-to-bit messages ($R$) corresponding to the prob. with which the bit satisfies a check given the prob. of the remaining bits being 1.
Parallel LDPC Decoders: Implementation Challenges

- Parallel/Crossbar Architectures: Mimic topology of Tanner graph
  - Allocate a Bit Function Unit (BFU) for each bit-node, a Check Function Unit (CFU) for each check-node, connect through interconnection network

![](image)

- Advantages: High-throughput

- Disadvantages:
  - Complex interconnect limiting scalability and clock speed
  - Prohibitive number of function units
  - Power consumption: Uses registers to store messages
Implementation Challenges (cont’d)

- Interconnect bottleneck restricts efficient implementation.
- Ring placement topology has average interconnect length $\propto \mathcal{O}(\sqrt{n})$
  - 52.5 mm$^2$ area for only a 1K code in 0.18 μm CMOS!
  - 50% utilization, 3 mm-interconnect, routing congestion restricts clock speed.
Interconnect Bottleneck of Parallel Architectures (cont’d)

- Based on post-layout data for a regular (3,6)-LDPC code in 0.18 μm, 1.8 V CMOS technology

![Graph showing average interconnect length (mm) and decoder area (mm²) vs n (x 1024)]
Serial LDPC Decoder Architectures

- Fold bit-message and check-message computations onto a small set of BFUs and CFUs.
- Store all messages in centralized memory, and communicate through read/write networks.

Disadvantages:
- Significant memory overhead
- Each FU needs separate read/write networks
- Complex network control
- Low throughput
Proposed Design Methodology for LDPC Codecs

I- Architecture-Aware LDPC codes
- Reduced interconnect complexity
- Graph properties matched to the requirements of iterative decoding

IIa- Turbo-decoding algorithm
- Fast convergence, memory efficient

IIb- Optimized message processing unit
- Shorter bit representation
- Lookup table free, short critical path
- Small area

III- Code-programmable rate-tunable platform
- Distributed message memory
- Parallel processing using multiple MPU’s
- Efficient, scalable, dynamic routing networks

IV- Parameterized decoder core
- Low-level transistor sizing
- Power-rail scaling
- Geometric parameterization
- Power-delay-area optimization

Programmable AA-LDPC decoder core

© 2005 M. Mansour, American University of Beirut
Area Savings Achieved by Proposed Methodology

\[ A(n) \text{ (mm}^2\) ]

\begin{align*}
\text{State-of-the-art} & \quad \sim 369 \text{ mm}^2 \\
\text{Proposed} & \quad \sim 56 \text{ mm}^2 \\
& \quad \sim 92 \text{ mm}^2 \\
& \quad \sim 14 \text{ mm}^2
\end{align*}

© 2005 M. Mansour, American University of Beirut
Transforming the Decoding Problem

- Gallager’s LDPC decoder:

- Proposed architecture-aware LDPC decoder:
The parity-check matrix $H$ of an AA-LDPC code is defined as:

\[
H_{D \times B} = \begin{bmatrix}
P & P & P & P & P \\
P & P & P & P & P \\
P & P & P & P & P \\
P & P & P & P & P
\end{bmatrix}
\]

- The sub-matrices are all-zero or permutation matrices of size $S (\approx \mathcal{O}(\sqrt{n}))$.
- The positions and structure of the sub-matrices are chosen to obtain large girth.
  - Examples: Combinatorial designs, cyclotomic cosets, Cayley/Ramanujan Graphs
- Architectural implications: Scale down interconnect complexity by a factor of $S$.
- Algorithmic implications: Transform problem into a turbo-decoding problem!
Ramanujan Graphs


  - Cayley graph $X_{RG}(q, p)$ of $\text{PGL}_2(q)$ generated by the generator set $S_p$
  - Both $q$ and $p$ are prime, and $p$ is a QNR mod $q$

- **Problem**: Not much flexibility in code parameters when $q$ is prime since $n \sim q^3$

- **Solution**: Generalize the LPS method into the case $q$ is a composite integer $m$.

- When $m$ is not prime, the generator set $S_p$ does not generate the whole group $\text{PGL}_2(m)$ (invertible $2 \times 2$ coset matrices).

- $S_p$ generates a subgroup $L_m$ of $\text{PGL}_2(m)$ which contains $\text{PSL}_2(m)$ as a subgroup of index 2 depending on the prime factorization of $m$.

- When $m$ is prime, $L_m$ coincides with $\text{PGL}_2(m)$.
Ramanujan Graphs: Standard LPS Method

Generator $S_p$

\[
\begin{bmatrix}
a & b \\
c & d \\
\end{bmatrix}
\]

\[a^2 + b^2 + c^2 + d^2 = p\]

$\varepsilon \begin{bmatrix} a & b \\ c & d \end{bmatrix}$

\[\varepsilon (ad - bc) \equiv \text{unit (mod } q)\]

$PGL_2(q)$

$PGL_2(q) = P \cup Q$

If $p$ is QNR mod $q$, then $S_p \subset Q$

$\forall A \in Q, s \in S, \Rightarrow A \cdot s \in P$

$\forall B \in P, s \in S, \Rightarrow B \cdot s \in Q$

$P = \text{subgroup of matrices with det. QR's mod } p$

$Q = \text{coset of matrices with det. QNR's mod } p$

Cayley graph of $PGL_2(q)$ over $S_p$ is Ramanujan.

- order $q^3 - q$
- degree $p + 1$

[Margulis’82]
[Rosenthal-Vontobel’00]
Generalized LPS Method

**Theorem 1** Let \( m = \prod_{j=1}^{h_1} q_j^{e_j} \cdot \prod_{j=h_1+1}^{h_1+h_2} q_j^{e_j} \) denote the prime factorization of \( m \) with \( h_1 \geq 1 \), such that \( p \) is a QNR modulo the left factors and a quadratic residue (QR) modulo the right factors. Define the map

\[
\tau_m : \text{PGL}_2(\mathbb{Z}_m) \to \{ \pm 1 \}^{h_1} \times \{ \pm 1 \}^{h_2}
\]

\[
A \mapsto \left[ \left( \frac{\text{det}(A)}{q_1} \right), \ldots, \left( \frac{\text{det}(A)}{q_{h_1}} \right) \right| \left( \frac{\text{det}(A)}{q_{h_1+1}} \right), \ldots, \left( \frac{\text{det}(A)}{q_{h_1+h_2}} \right) \right]
\]

Then

- \( L_m = \tau_m^{-1}([\pm 1, \ldots, \pm 1 | 1, \ldots, 1]) \),
- The Cayley graph \( X(L_m, S_p) \) is bipartite, and the bipartitions are \( P_m = \text{PSL}_2(m) = \tau_m^{-1}([1, \ldots, 1 | 1, \ldots, 1]) \) and \( Q_m = \tau_m^{-1}([-1, \ldots, -1 | 1, \ldots, 1]) \),
- \( |L_m| = \frac{m^3}{2^{h_1+h_2-\nu-1}} \prod_{i=1}^{h_1+h_2} (1 - \frac{1}{q_i^2}) \) where \( \nu = 1 \) if some \( q_i = 2 \), else \( \nu = 0 \), and
- \( X(L_m, S_p) \) is Ramanujan, and \( g(X) \geq 4 \log_p (m) - 4 \log_p (2) \approx \frac{4}{3} \log_p (|L_m|) \).
Girth and Degree of Some Ramanujan Graphs

| $m$ | $p$ | $|L_m|$ | Girth(g) / Degree(d) |
|-----|-----|--------|-----------------------|
| 5   | 3   | 120    | $g = 6, d = 4$        |
| 7   | 5   | 336    | $g = 6, d = 6$        |
| 9   | 5   | 648    | $g = 6, d = 6$        |
| 15  | 7   | 1440   | $g = 6, d = 8$        |
| 14  | 5   | 2016   | $g = 6, d = 6$        |
| 17  | 5   | 4896   | $g = 6, d = 6$        |
| 18  | 5   | 3888   | $g = 6, d = 6$        |
| 19  | 13  | 6840   | $g = 6, d = 6$        |
| 21  | 5   | 4032   | $g = 6, d = 6$        |
| 21  | 11  | 4032   | $g = 4, d = 6$        |
| 22  | 7   | 7920   | $g = 8, d = 6$        |
| 25  | 7   | 15000  | $g = 8, d = 8$        |
| 27  | 5   | 17496  | $g = 10, d = 6$       |
| 33  | 5   | 15840  | $g = 8, d = 6$        |
| 35  | 13  | 20160  | $g = 8, d = 6$        |
| 35  | 3   | 20160  | $g = 14, d = 4$       |
| 39  | 5   | 26208  | $g = 10, d = 6$       |
| 39  | 7   | 26208  | $g = 8, d = 6$        |
| 39  | 7   | 26208  | $g = 8, d = 8$        |

© 2005 M. Mansour, American University of Beirut
Theorem 2 (\(\text{PGL}_2(\mathbb{Z}_m)\)) Let \(m\) be as defined in Theorem 1, and \(A\) be any element of \(\text{PGL}_2(\mathbb{Z}_m)\). Then \(A\) can be expressed as

\[
A = \sum_{i=1}^{h} \left( u_i A_i \cdot \prod_{j=1\atop j \neq i}^{h} q_{ij}^{e_j} \right),
\]

where \(A_i \in \text{PGL}_2(\mathbb{Z}_{q_i^{e_i}})\) has the form \((\begin{smallmatrix} 1 & b_i \\ c_i & d_i \end{smallmatrix})\) with \(d_i \neq b_i c_i\), or \((\begin{smallmatrix} a_i q_i & 1 \\ c_i & d_i \end{smallmatrix})\) with \(a_i d_i q_i \neq c_i\), for arbitrary integers \(a_i, b_i, c_i, d_i\), and appropriately determined integers \(u_1, \ldots, u_h\). Moreover, the order of \(\text{PGL}_2(\mathbb{Z}_m)\) is given by

\[
|\text{PGL}_2(\mathbb{Z}_m)| = \prod_{i=1}^{h} q_i^{3e_i} (1 - \frac{1}{q_i^2}) = m^3 \prod_{i=1}^{h} \left(1 - \frac{1}{q_i^2} \right).
\]
Theorem 3 (AA-structure) The reduced adjacency matrix $A_{RG}(m, p)$ of a Ramanujan graph $X_{RG}(m, p)$ can be partitioned into a $D \times D$ array of square matrices $W_{S \times S}$ where

$$D = m \prod_{i=1}^{h} \left(1 + \frac{1}{q_i}\right) \text{ and } S = \frac{m^2}{2^{h-\nu}} \prod_{i=1}^{h} \left(1 - \frac{1}{q_i}\right).$$

Each $W$ is either $0_{S \times S}$ or the sum of one or more non-overlapping permutation matrices.
**Example**

**Example 1** For $m = 9$ and $p = 5$, $X_{RG}(9, 5)$ has 648 vertices and valency 6. Its adjacency matrix $A_{RG}(9, 5)$ has size 340, and can be partitioned into a $12 \times 12$ array of submatrices of size 27 as follows:

$$A_{RG} = \begin{bmatrix}
\end{bmatrix}.$$
Vontobel-Loeliger [6] introduced a procedure for replacing the nodes of a regular graph with nodes of a factor graph.

We extend this procedure to Ramanujan-based LDPC codes obtained via the GLPS method in such a way as to preserve the AA-structure.
## Transformations

<table>
<thead>
<tr>
<th>$\mathbf{A_{RG}} / X_{RG}$</th>
<th>Node splitting</th>
<th>Edge splitting</th>
<th>Node replacement + Edge splitting</th>
<th>Edge merging</th>
<th>Edge merging + Node replacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$x_1 x_2$</td>
<td>$n_1 n_2 n_3 n_4 n_5 n_6$</td>
<td>$n_1 n_2 n_3 d e f$</td>
<td>$n_1 n_2 n_3$</td>
<td>$n_1 d e f$</td>
</tr>
<tr>
<td>$a$</td>
<td>$P_1$</td>
<td>$P_1$</td>
<td>$P_1$</td>
<td>$a$</td>
<td>$P_1$</td>
</tr>
<tr>
<td>$b$</td>
<td>$P_2$</td>
<td>$P_2$</td>
<td>$P_2$</td>
<td>$b$</td>
<td>$P_2$</td>
</tr>
<tr>
<td>$c$</td>
<td>$P_3$</td>
<td>$P_3$</td>
<td>$P_3$</td>
<td>$c$</td>
<td>$P_3$</td>
</tr>
<tr>
<td>$d$</td>
<td>$P_4$</td>
<td>$P_4$</td>
<td>$P_4$</td>
<td>$d$</td>
<td>$P_4$</td>
</tr>
<tr>
<td>$e$</td>
<td>$P_5$</td>
<td>$P_5$</td>
<td>$P_5$</td>
<td>$e$</td>
<td>$P_5$</td>
</tr>
<tr>
<td>$f$</td>
<td>$P_6$</td>
<td>$P_6$</td>
<td>$P_6$</td>
<td>$f$</td>
<td>$P_6$</td>
</tr>
<tr>
<td>$x_1$</td>
<td>$x_1$</td>
<td>$x_1$</td>
<td>$x_1$</td>
<td>$x_1$</td>
<td>$x_1$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$x_2$</td>
<td>$x_2$</td>
<td>$x_2$</td>
<td>$x_2$</td>
<td>$x_2$</td>
</tr>
<tr>
<td>$n_1$</td>
<td>$n_1$</td>
<td>$n_1$</td>
<td>$n_1$</td>
<td>$n_1$</td>
<td>$n_1$</td>
</tr>
<tr>
<td>$n_2$</td>
<td>$n_2$</td>
<td>$n_2$</td>
<td>$n_2$</td>
<td>$n_2$</td>
<td>$n_2$</td>
</tr>
<tr>
<td>$n_3$</td>
<td>$n_3$</td>
<td>$n_3$</td>
<td>$n_3$</td>
<td>$n_3$</td>
<td>$n_3$</td>
</tr>
<tr>
<td>$n_4$</td>
<td>$n_4$</td>
<td>$n_4$</td>
<td>$n_4$</td>
<td>$n_4$</td>
<td>$n_4$</td>
</tr>
<tr>
<td>$n_5$</td>
<td>$n_5$</td>
<td>$n_5$</td>
<td>$n_5$</td>
<td>$n_5$</td>
<td>$n_5$</td>
</tr>
<tr>
<td>$n_6$</td>
<td>$n_6$</td>
<td>$n_6$</td>
<td>$n_6$</td>
<td>$n_6$</td>
<td>$n_6$</td>
</tr>
</tbody>
</table>

© 2005 M. Mansour, American University of Beirut
Performance of Some AA-LDPC Codes

Rate 0.33
Rate 0.5
128 iterations
20,000 frames

© 2005 M. Mansour, American University of Beirut
II-Turbo-Decoding of AA-LDPC Codes

- An AA-LDPC code can be viewed as a concatenation of super-codes, each of which is a direct-sum of single-parity-check codes.

- The turbo-decoding algorithm can be applied to decode LDPC codes instead of Gallager’s TPMP algorithm.

- Advantages: Process one type of messages, faster convergence, 75% memory savings, interleavers easy to implement.

Parameters: $S = 4$, $D = 3$, $B = 4$, $m = 12$, $n = 16$
Simpler Flow of Messages in Factor Graph

Turbo-code decomposition

Convolutional code

\( C^1 \)

SISO

Interleaver \( \pi_1 \)

\( \Lambda \)

\( C^2 \)

Interleaver \( \pi_2 \)

\( \Lambda \)

\( C^3 \)

Interleaver \( \pi_3 \)

(channel values)

Architecture-aware decomposition

(sub-code)

\( \pi_{11} \)

\( \pi_{12} \)

\( \pi_{13} \)

\( \pi_{14} \)

\( \pi_{21} \)

\( \pi_{22} \)

\( \pi_{23} \)

\( \pi_{24} \)

\( \pi_{31} \)

\( \pi_{32} \)

\( \pi_{33} \)

\( \pi_{34} \)
Dual Extrinsic Principle: Reduced Memory Requirements

- Outer extrinsics
- Inner extrinsics

![Diagram](attachment:image.png)
Convergence Rate of the TDMP Algorithm

- Length 4200, rate 0.5, regular (3,6)-AA-LDPC code
- Use the BCJR algorithm in differential form on the syndrome trellis of an even parity-check code [ISLPED’02].

- Kernel equation: Max-quartet approximation with NO lookup tables

\[
Q(x, y) = \max(x, y) - \max(x + y, 0) + \max\left(\frac{5}{8} - \frac{|x - y|}{4}, 0\right) - \max\left(\frac{5}{8} - \frac{|x + y|}{4}, 0\right)
\]

- Key update equations:

\[
\Delta \alpha' = Q(\Delta \alpha, \lambda), \quad \Delta \beta' = Q(\Delta \beta, \lambda), \\
\Lambda = Q(\Delta \alpha, \Delta \beta)
\]
Simple and Accurate Approximation

\[ \delta(u) = \max\left(\frac{5}{8} - \frac{|u|}{4}, 0\right) \]

\[ Q(x, y) \]

\[ E_b/N_0 \text{ [dB]} \]

\[ \text{BER} \]

Copyright © 2005 M. Mansour, American University of Beirut
SISO Message Processing Units

- **Parallel implementation:**

- **Serial implementation:**
III-Architecture: Programmable and Tunable Decoding Platform

- Scalable decoder architecture for the ensemble of $[D, B, S]$-AA-LDPC codes
Core-based IC design methodologies: Tradeoff between the high quality of full-custom designs and the short design cycle time of synthesis-based designs.

Communication systems:
- Applications, standards, and process technologies change and evolve towards optimum system energy and throughput efficiencies.
- Desirable: portability across technology generations and predictability of design quality

Approach: Build a core-generator based on parameterized layout libraries
- Easy to upgrade
- Instantiate cells with virtually any size

Stack a parameterized macro-cell (PMC) library on top of parameterized leaf-cell (PLC) containing layout of all building blocks of a TDMP AA-LDPC decoder.
- Novelty: Optimize blocks by performing low-level transistor sizing, power-rail scaling, and other geometric modifications with minimal effort.
Parameterized Layout Cell Libraries

- Includes cells ranging from inverters to full adders (with saturation arithmetic), SRAM cells, and switch cells
- Cells parameterized by scaling vectors, power-line sizing, and other geometric attributes
- Current model developed for delay, power, reliability characterization

PMC library

- Logic Units
  - Q function
  - Serial SISO MPU
  - Parallel SISO MPU
  - Periphery blocks
- Networks
  - Omega network
  - Beneš network
  - Other topologies
- Memory
  - Single-port $\lambda$-memory
  - Single-port $H$-memory
  - Dual-port $\pi$-memory
  - Dual-port $\Gamma$-memory

PLC library

Technology file
Examples of Parameterized Macro Cells

- MPU (top), $\Omega$-Network (left), $\Gamma$-Memory (right)
# LDPC Decoder Chip Implementation

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datapath</td>
<td>4-bit</td>
</tr>
<tr>
<td>Code length</td>
<td>2048</td>
</tr>
<tr>
<td>Code rate</td>
<td>0.5-0.875</td>
</tr>
<tr>
<td>Frequency</td>
<td>125 MHz</td>
</tr>
<tr>
<td>Throughput</td>
<td>6.4 Gbps</td>
</tr>
<tr>
<td>IO Pins</td>
<td>76</td>
</tr>
<tr>
<td>Area</td>
<td>14.3 mm²</td>
</tr>
<tr>
<td>Power</td>
<td>787 mW</td>
</tr>
<tr>
<td>Technology</td>
<td>0.18 μm, 1.8 V TSMC CMOS</td>
</tr>
<tr>
<td>Foundry</td>
<td>MOSIS</td>
</tr>
</tbody>
</table>

© 2005 M. Mansour, American University of Beirut
Die Micrograph of the Decoder Chip
- Regular (3,6)-LDPC code, length 2048, rate 0.5
- 0.18 μm, 1.8 V TSMC CMOS technology
Generalizations to Repeat-Accumulate Codes

- Same ideas can be applied to Repeat-Accumulate codes introduced by Divsalar-Jin-McEliece (1998)

(c) IRA Encoder

(d) IRA Tanner graph

(e) Parity-check matrix
Generalizations to Repeat-Accumulate Codes (cont’d)

- Architecture-aware parity-check matrix
  - Base systematic part obtained from combinatorial designs, cage graphs, Ramanujan graphs

\[
H_{N \times (N+K)} = \begin{bmatrix}
I_{\pi_1} & I_{\pi_2} & I_{\pi_3} & I_{\pi_4} & I & I \\
I_{\pi_5} & I_{\pi_6} & I_{\pi_7} & I_{\pi_8} & I & I \\
I_{\pi_9} & I_{\pi_{10}} & I_{\pi_{11}} & I_{\pi_{12}} & I & I \\
\end{bmatrix}
\]

\(\pi_1, \pi_2, \ldots, \pi_{12}\) are indices for the information parity matrix.

© 2005 M. Mansour, American University of Beirut
Conclusions

- Decoders for LDPC codes require an order of magnitude more memory than turbo codes, and their randomness creates an on-chip interconnect bottleneck.
- Existing techniques address these issues at the architectural level, result in limited effects.
- We addressed the problems of memory overhead and interconnect bottleneck at four levels of abstraction:
  - Code design: Architecture-aware LDPC codes
  - Decoding algorithm:
    * Proposed a new turbo-decoding algorithm that reduces memory requirements by > 75%, and improves decoding throughput.
    * Proposed a reduced-complexity message computation mechanism
  - Architecture: Proposed a programmable and scalable TDMP decoder architecture
  - Layout: Core-based design using PMC layout library
- LDPC decoder chip implementation
References


