Faulty Stochastic LDPC Decoders Over the Binary Symmetric Channel

C. Kameni Ngassa $^{1,2}$, V. Savin $^2$ & D. Declercq $^1$

1 ETIS / ENSEA - Université de Cergy-Pontoise - CNRS UMR 8051
95000 Cergy-Pontoise Cedex, France

2 CEA-LETI
Minatec Campus, 38054 Grenoble cedex 9, France

August 19, 2014
Contents

1 Introduction

2 LDPC Codes and Stochastic Decoding

3 Faulty Stochastic Decoder

4 Monte-Carlo Simulation

5 Conclusion
**Introduction**

**Motivations**
- Future generation of communication systems will need to deal with hardware failure
- Increasing interest on LDPC decoders in presence of unreliable components:
  - we previously studied faulty Min-Sum decoders [Kameni’13]

**Our previous results**
- The *robustness* to hardware of the Min-Sum decoder is significantly improved when the *sign bit* of exchanged messages is protected.
- However it might be difficult to protect only the most significant bit of messages

**Intuition**
Stochastic decoder *might be inherently robust* to noise from unreliable component because
- messages exchanged in the decoder are random;
- information is represented using bits of similar significance;
- timing errors can be tolerated in stochastic decoder [Perez’13].
LDPC Codes and Stochastic Decoding

LDPC Codes and Belief Propagation algorithm
- Capacity approaching linear block codes
- Sparse parity check matrix
- Bipartite graph representation and message-passing algorithm
- Belief Propagation decoding
  - Drawback: computationally complex

Stochastic Decoding
- Stochastic implementation of the probability-domain BP
- Convert probability beliefs into streams of random bits
- Advantages:
  - simple logic gates to perform complex arithmetic operations
  - only one bit exchanged at each iteration
  - less the number of wires in the circuit [Sharifi’08]
Probability to Stochastic Stream Conversion

**Figure**: Stochastic Stream Generator

- $P$ probability to convert (quantized on $q$ bits)
- $R$ random number uniformly generated (quantized on $q$ bits)
- Output = 1 if $R < P$
  - $P$ = the frequency of 1 in the stochastic stream
- representation *not unique*

\[
\begin{align*}
0 & 1 1 0 0 0 0 1 0 0 \\
0 & 0 0 1 0 0 1 0 0 1 \\
0 & 0 0 0 1 1 0 1 0 0
\end{align*}
\]

\[\rightarrow 0.3\]

- one error occurring on any bit has a limited impact on the probability value.
From Belief Propagation to Stochastic Decoding Algorithm

Check-to-variable node processing

**Probability domain BP**

\[ \beta_{m,n} = \frac{1}{2} - \frac{1}{2} \prod_{n' \in H(m) \setminus n} (1 - 2\alpha_{m,n'}) \quad (1) \]

**Stochastic Decoding Algorithm**

\[ b_{m,n} = \text{XOR}_{n' \in H(m) \setminus n} (a_{m,n'}) \quad (2) \]
From Belief Propagation to Stochastic Decoding

Variable-to-check node processing

Probability domain BP

\[ \alpha_{m,n} = \frac{\gamma_n \prod_{m' \in H(n) \setminus m} \beta_{m',n}}{\left( \gamma_n \prod_{m' \in H(n) \setminus m} \beta_{m',n} \right) + \left( 1 - \gamma_n \prod_{m' \in H(n) \setminus m} (1 - \beta_{m',n}) \right)} \]
From Belief Propagation to Stochastic Decoding

Variable-to-check node processing

Stochastic Decoding Algorithm

Division in stochastic computing: JK flip-flop

\[ J^{(\ell)} \quad K^{(\ell)} \quad Q^{(\ell)} \]

<table>
<thead>
<tr>
<th>( j^{(\ell)} )</th>
<th>( k^{(\ell)} )</th>
<th>( q^{(\ell)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>( q^{(\ell - 1)} )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>( q^{(\ell - 1)} )</td>
</tr>
</tbody>
</table>

**Table:** JK flip-flop truth table

\[ P_q = \frac{P_j}{P_j + P_k} \] (4)

\[ a^{(\ell)}_{m,n} = \begin{cases} 
  c^{(\ell)}_n, & \text{if } c^{(\ell)}_n = b^{(\ell)}_{m',n}, \forall m' \in \mathcal{H}(n) \setminus m \\
  a^{(\ell - 1)}_{m,n}, & \text{otherwise}
\end{cases} \] (5)
From Belief Propagation to Stochastic Decoding

A Posteriori Update

**Probability domain BP**

$$\tilde{\gamma}_n = \frac{\gamma_n \prod_{m \in H(n)} \beta_{m,n}}{\left(\gamma_n \prod_{m \in H(n)} \beta_{m,n}\right) + \left((1 - \gamma_n) \prod_{m \in H(n)} (1 - \beta_{m,n})\right)} \quad (6)$$

**Stochastic Decoding Algorithm : up/down counter**

$$\theta_n = \theta_n + \sum_{m \in H(n)} (2a_{m,n} - 1) \quad (7)$$

not equivalent to the BP a posteriori update

**Hard decision**

**Probability domain BP**

$$\hat{x}_n = \begin{cases} 1, & \text{if } \tilde{\gamma}_n > 0.5 \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

**Stochastic Decoding Algorithm**

$$\hat{x}_n = \begin{cases} 1, & \text{if } \theta_n > 0 \\ 0, & \text{otherwise} \end{cases} \quad (9)$$
Improving the Stochastic Decoder Performance

Latching Problem
- cycles in the graph $\rightarrow$ correlation in stochastic streams
- $a_{m,n}^{(\ell)} = a_{m,n}^{(\ell-1)} \rightarrow$ hold state and low level of switching activity

Consequence: groups of nodes locked into fixed states which prevents proper decoding and leads to poor performance.

Noise-Dependent Scaling

Goal: ensure similar switching activity for different channel noise level
Achieved by replacing $\epsilon$ by a constant $\mu$ [Huang’13] as follows.

$$
Pr(c_n = 1) = \begin{cases} 
1 - \epsilon, & \text{if } y_n = 1 \\
\epsilon, & \text{if } y_n = 0 
\end{cases} \quad \rightarrow \quad Pr(c_n = 1) = \begin{cases} 
1 - \mu, & \text{if } y_n = 1 \\
\mu, & \text{if } y_n = 0 
\end{cases}
$$

(10)

Edge-Memories

Goal: decorrelate bits in stochastic streams
- $S$-bit shift register assigned to one edge of the decoder
Improving the Stochastic Decoder Performance

\[ b_{m_1,n}, b_{m_2,n}, b_{m(dv-1),n}, c_n \]

**Figure:** Variable node of stochastic decoder with edge-memories

**VN Processing with Edge-Memories**

At cycle \( \ell \), for \( \forall \ n \in \{1, \ldots, N\} \) and \( m \in H(n) \) if \( \forall m' \in H(n)\setminus m \), \( b_{m',n}^{(\ell)} = c_n \)

\[
\begin{align*}
a_m^{(\ell),n} & = c_n & \text{(the new value of } a_{m,n} \text{ is stored in the memory)} \\
c_n & \rightarrow \text{EM} & \\
\text{otherwise, } a_m^{(\ell),n} & = \text{EM}(i) & \text{(a bit is randomly chosen from the memory)}
\end{align*}
\]

\[ i = \text{random position in the edge-memory.} \]
Faulty Stochastic Decoder (1)

Stochastic Decoders with Noisy Edge-Memories

- \( \text{EM}(i) \) = bit read at address \( i \) from the noiseless EM
- \( \text{EM}_{pr}(i) \) = bit read at address \( i \) from the noisy EM

\[
\text{EM}_{pr}(i) = \begin{cases} 
\text{EM}(i), & \text{with probability } 1 - p_{em} \\
\text{EM}(i), & \text{with probability } p_{em}
\end{cases} \tag{11}
\]

VN Processing with Edge-Memories

At cycle \( \ell \), for all \( n \in \{1, \ldots, N\} \) and \( m \in \mathcal{H}(n) \) if \( \forall m' \in H(n) \setminus m \), \( b_{m',n}^{(\ell)} = c_n \)

\[
a_{m,n}^{(\ell)} = c_n \quad \text{(the new value of } a_{m,n} \text{ is stored in the memory with no error)}
\]

\[
c_n \rightarrow \text{EM}
\]

otherwise, \( a_{m,n}^{(\ell)} = \text{EM}_{pr}(i) \) (a bit is randomly chosen from the faulty memory)

\( i = \text{random position in the edge-memory.} \)
Faulty Stochastic Decoder (2)

Full Noisy Stochastic Decoder

Output of the Noisy Stochastic Stream Generator:

\[ \pi_{pr}(p) = \begin{cases} \pi(p), & \text{with probability } 1 - p_{\tau} \\ \frac{\pi(p)}{\pi(p)}, & \text{with probability } p_{\tau} \end{cases} \]

\( \pi_{pr}(p) \) behaves like the noiseless \( \pi(p(1 - p_{\tau}) + (1 - p)p_{\tau}) \).

Output of the Noisy Check Node Processing:

\[ C_{pr} = \begin{cases} C, & \text{with probability } 1 - p_c \\ \frac{C}{C}, & \text{with probability } p_c \end{cases} \]

Output of the Noisy Variable Node Processing:

\[ V_{pr} = \begin{cases} V, & \text{with probability } 1 - p_v \\ \frac{V}{V}, & \text{with probability } p_v \end{cases} \]
Robustness Assessment of Noisy Stochastic Decoders

No density evolution analysis
because variable-to-check node messages are computed as functions of dependent random variables.

- $a^{(\ell)}_{m,n}$ is a function of $c_n$, $(b^{(\ell)}_{m',n})_{m' \in H(n) \setminus m}$, and $a^{(\ell-1)}_{m,n}$
- but $(b^{(\ell)}_{m',n})_{m' \in H(n) \setminus m}$ and $a^{(\ell-1)}_{m,n}$ are dependent.

Monte-Carlo Simulation

- $(3, 6)$-regular LDPC code with length $N = 1008$ bits
- channel input probabilities are quantized on $q = 8$ bits,
- the noise dependent scale factor is $\mu = 0.12$,
- all decoders use 48-bit edge-memories,
- the maximum number of decoding cycles is set to 1000.

- For comparison: floating-point BP decoding with maximum number of iterations equals to 100
Numerical Results for the Noisy EM Stochastic Decoder

**Figure:** BER performance of the stochastic decoder with noisy edge-memories
Numerical Results for the Full Noisy Stochastic Decoder

**Figure:** BER performance of the full noisy stochastic decoder
Numerical Results with Only Noisy Variable Node Units

**Figure:** BER performance with only noisy Variable Node Units
Conclusion and Perspectives

Conclusion

- We investigated faulty Stochastic decoders
- Two error models introduced
- Stochastic decoders managed to operate with high level of errors due to different components
  - Edge-Memories can be unreliable
  - Results can be used to \textit{reduce the energy consumption} of Edge-Memories through \textit{voltage scaling}

Perspectives

- Investigate faulty stochastic decoders with \textit{technology dependent} error models under aggressive voltage scaling
- Study other noisy re-randomization units (besides edge-memories)
References


C. L. Kameni Ngassa, V. Savin, and D. Declercq, “Min-sum-based decoders running on noisy hardware,” in *proc. of IEEE Global Communications Conference (GLOBECOM)*, 2013.


Thank You for Your Attention!