Soft Bit Decoding of Low Density Parity Check Codes

Sheryl L. Howard, Vincent C. Gaudet, and Christian Schlegel
Department of Electrical and Computer Engineering
ECERF Building, University of Alberta, Edmonton, Alberta T6G 2V4
{sheryl,vgaudet,schlegel}@ece.ualberta.ca
May 3, 2004

Abstract

A novel soft-bit-domain decoding algorithm for low-density parity-check codes is derived as an alternative to belief propagation and min-sum algorithms. A simple approximation analogous to the min-sum approximation is also presented. Simulation results demonstrating the functionality of the algorithm are given both for floating and fixed point representations.

Keywords: Low-Density Parity-Check Codes, Iterative Decoding Algorithms, Soft Output Decoding, Belief Propagation
I Introduction

Turbo Codes [1] and Low-Density Parity-Check (LDPC) Codes [2], both iteratively decoded near-capacity-performing classes of error-correcting codes, are used in a number of recent digital communications standards, including W-CDMA, CDMA2000, IEEE 802.16, 802.20, DVB-RCT and DVB S-2. LDPC codes are typically decoded with belief propagation (BP) [3] or sub-optimal min-sum [4] algorithms.

BP algorithms operate in either the probability or log-likelihood ratio (LLR) domains. Min-sum algorithms operate in the LLR domain. Both BP and min-sum algorithms have been used in recent state-of-the-art LDPC decoder implementations [5], [6], [7]. An alternate form of the BP decoding algorithm is derived in this paper, with metrics represented in the hyperbolic tangent domain. This domain is sometimes called the soft bit domain, as metrics are in the range (-1,+1).

This paper is organized as follows. Section II derives the soft-bit-domain BP algorithm. Section III discusses an approximation to the algorithm of Section II, analogous to the min-sum algorithm. Section IV presents simulation results, including both floating point and fixed point representations. Finally, Section V concludes this paper.

II The Soft Bit Decoding Algorithm

LDPC decoding is viewed graphically via a bipartite graph, or factor graph. The factor graph has both variable nodes, representing the codeword bits, and parity check nodes, representing the parity check equations of the code’s parity check matrix $H$. Edges connect variable nodes to check nodes according to the check equations. The factor graph is thus a visual representation of $H$. Decoding consists of sending messages along the connecting edges, from variable nodes, initialized with received channel metrics, to parity-check nodes and back. Each node type performs different computations, utilizing the extrinsic principle, i.e., an outgoing message on an edge excludes information received along the same edge.

In the log-domain version of BP, each check node $j$ performs the following computation for its outgoing LLR message to variable node $i$, assuming check node degree (number of edge connections) $d_c$:

$$\lambda_{j \rightarrow i} = 2 \tanh^{-1} \left( \prod_{k=1}^{d_c} \tanh \left( \frac{\lambda_{k \rightarrow j}}{2} \right) \right).$$

(1)
Equation (1) can be rewritten as
\[
\tanh \left( \frac{\lambda_{j \rightarrow i}}{2} \right) = \prod_{k=1}^{d_{v}} \tanh \left( \frac{\lambda_{k \rightarrow j}}{2} \right).
\] (2)

Introducing the soft bit \( T = \tanh(\lambda/2) \), equation (2) may be rewritten as
\[
T_{j \rightarrow i} = \prod_{k=1 \atop (k \neq i)}^{d_{v}} T_{k \rightarrow j}.
\] (3)

This soft bit has fixed range (-1,+1). Linear quantization of the tanh metric corresponds to nonlinear LLR quantization at larger values, perhaps a more accurate model of actual message evolution over iterations.

Each variable node \( i \), in the log-domain version of BP, performs the following computation for its outgoing LLR message to parity check node \( j \), assuming variable node degree \( d_{v} \):
\[
\lambda_{i \rightarrow j} = \lambda_{\text{ch},i} + \sum_{l=1 \atop (l \neq j)}^{d_{v}} \lambda_{l \rightarrow i},
\] (4)

where \( \lambda_{\text{ch},i} \) is the channel LLR available from bit \( i \). Moving into the soft bit domain, equation (4) becomes
\[
T_{i \rightarrow j} = \tanh \left( \frac{\lambda_{i \rightarrow j}}{2} \right) = \tanh \left( \frac{\lambda_{\text{ch},i}}{2} + \sum_{l=1 \atop (l \neq j)}^{d_{v}} \frac{\lambda_{l \rightarrow i}}{2} \right).
\] (5)

Using \( \tanh(x + y) = (\tanh(x) + \tanh(y))/(1 + \tanh(x) \tanh(y)) \), a degree 2 variable node message becomes
\[
T_{i \rightarrow j} = \tanh \left( \frac{\lambda_{\text{ch},i}}{2} + \frac{\lambda_{l_1 \rightarrow i}}{2} \right) = \frac{T_{\text{ch},i} + T_{l_1 \rightarrow i}}{1 + T_{\text{ch},i} T_{l_1 \rightarrow i}}, \quad l \neq j.
\] (6)

Similarly for a degree 3 variable node, using \( \tanh(x + y + z) = \tanh((x + y) + z) \), we obtain
\[
T_{i \rightarrow j} = \tanh \left( \frac{\lambda_{\text{ch},i}}{2} + \frac{\lambda_{l_1 \rightarrow i}}{2} + \frac{\lambda_{l_2 \rightarrow i}}{2} \right) = \frac{T_{\text{ch},i} + T_{l_1 \rightarrow i} + T_{l_2 \rightarrow i} + T_{\text{ch},i} T_{l_1 \rightarrow i} T_{l_2 \rightarrow i}}{1 + T_{\text{ch},i} T_{l_1 \rightarrow i} + T_{\text{ch},i} T_{l_2 \rightarrow i} + T_{l_1 \rightarrow i} T_{l_2 \rightarrow i}}, \quad l_1, l_2 \neq j.
\] (7)

In summary, a soft-bit-domain BP algorithm is implemented by performing products at check nodes and equation (5)/equation (7) at variable nodes. The complexity has shifted from the check nodes for
the LLR-domain BP algorithm to the variable nodes. A simplification of the variable node equation (7), suitable for VLSI implementation, is presented in the next Section.

III Simplification of the Variable Node Equation

In a degree 3 variable node with incoming soft bit messages near 1, equation (7) approaches 1. For small-valued soft bit messages, equation (7) reduces to approximately the sum of the incoming messages, neglecting higher order terms. Equation (7) for a degree 3 variable node is thus approximated as

\[
T_{i \rightarrow j} = \begin{cases} 
T_{ch,i} + T_{l_1 \rightarrow i} + T_{l_2 \rightarrow i}, & \text{if } |T_{ch,i} + T_{l_1 \rightarrow i} + T_{l_2 \rightarrow i}| < 1; \\
\text{sign}(T_{ch,i} + T_{l_1 \rightarrow i} + T_{l_2 \rightarrow i}), & \text{if } |T_{ch,i} + T_{l_1 \rightarrow i} + T_{l_2 \rightarrow i}| \geq 1; 
\end{cases}
\]

(8)

IV Simulation Results

Decoding using both exact equation (7) and approximation (8) was performed for a [3,6] LDPC code of codeword length \(N=1008\) and 504 parity check equations [8], which is slightly irregular in parity node degree, over an AWGN (additive white Gaussian noise) channel. Simulation results are presented, with each SNR point frame error rate (FER) value calculated from 50+ frame errors. Both floating and fixed point precision were examined. The bit precision given includes one sign bit. Quantization to \(m\) bits results in \(2^m\) bins. Bin edges are placed at \(\pm i/2^m - 1, \forall i = 0, 1, \ldots, 2^m - 1\), with the quantized value set to the bin center value at \(\pm(2i + 1)/2^m, \forall i = 0, \ldots, 2^m - 1\).

Figure 1 displays exact equation (7) results for both floating and fixed point precision. Convergence to floating point results requires 7 bits of precision; 6 bits of precision results in approximately 0.1 dB loss. Floating point FER results are consistent with BP decoding of the same code with LLR-domain messages.

Figure 2 displays results using the approximation of equation (8) for both floating and fixed point precision. Floating point FER results for the approximation show a 0.5 dB loss compared to the exact equation. Convergence to floating point results for the approximation requires 6 bits of precision; 5 bits of precision results in roughly 0.1 dB loss.
V Conclusions

A novel decoding algorithm for LDPC codes which operates in the soft-bit domain rather than the LLR domain was derived and simulated. Use of soft-bit messages, which have a fixed range from (-1,1) rather than (-inf,+inf) as with LLRs, allows for linear quantization of a nonlinear parameter, the tanh(LLR) messages. This is equivalent to a nonlinear quantization of the LLR messages, which may be desirable in order to more accurately reflect the LLR distribution, but is difficult to implement in the LLR domain.

A simple approximation to the more complex variable node processing in the soft-bit-domain algorithm was presented. Simulation results showed a 0.5 dB loss over exact variable node processing (similar to the loss incurred by the min-sum approximation in LLR-domain BP decoding), but achieved floating point results with lower bit precision. This algorithm is a soft-bit counterpart to the min-sum algorithm and could be used in hardware implementations of decoders.
References


Figure 1. Signal-to-noise ratio (SNR) vs frame error rate (FER) for soft-bit message decoding of $N=1008$ [3,6] LDPC code, with exact equation (7) at variable nodes.

Figure 2. Signal-to-noise ratio (SNR) vs frame error rate (FER) for soft-bit message decoding of $N=1008$ [3,6] LDPC code, with sum approximation (8) at variable nodes.
FER vs SNR: $N=1008$ $M=512$ [3,6] LDPC, tanh msgs

FER = frame error rate, 50+ frame errors

SNR = $E_b/N_0$

Exact tanh eqn at var: floating pt
- Exact: bit prec=4 bits
- Exact: bit prec=5 bits
- Exact: bit prec=6 bits
- Exact: bit prec=7 bits

SUM approx at var: floating pt
- SUM approx: bit prec=4 bits
- SUM approx: bit prec=6 bits
- SUM approx: bit prec=5 bits