COMPUTATIONAL COMPLEXITY AND QUANTIZATION EFFECTS OF DECODING ALGORITHMS FOR NON-BINARY LDPC CODES

Henk Wymeersch, Heidi Steendam and Marc Moeneclaey

DIGCOM research group, TELIN Dept.,
Ghent University
Sint-Pietersnieuwstraat 41, 9000 GENT, BELGIUM
E-mail: {hwymeers,hs,mm}@telin.ugent.be

ABSTRACT

This contribution deals with the comparison of the sum-product algorithm (SPA) and its log-domain version (log-SPA) for decoding LDPC codes over general binary extension fields. For both algorithms, we determine their computational complexity based on the number of real-valued operations and investigate their sensitivity to quantization effects. Whereas the log-SPA yields the shorter decoding time in the case of binary LDPC codes, we point out that increasing the field size tends to favor the SPA, especially when a multiplication takes only slightly more time than an addition. Further, we show that log-SPA requires fewer quantization levels and suffers less from a quantization induced error-floor.

1. INTRODUCTION

Binary low density parity check (LDPC) codes were originally introduced by Gallager in 1963 [1]. They have recently received a lot of attention because of their impressive performance on both the binary symmetric channel (BSC) and the AWGN channel [2, 3]. LDPC codes have certain advantages as compared to turbo codes and are considered to be a serious competitor in near-Shannon limit communication. Decoding of LDPC codes is based on Pearl's belief propagation (BP) algorithm [4], also known as the sum-product algorithm (SPA) [5]. In practice, a mathematically equivalent log domain version (log-SPA) is favored, since it requires no normalization step and is less sensitive to quantization effects [6]. In the log-SPA, the multiplications and additions from the SPA are replaced by additions and by the so-called Jacobi logarithm [7], respectively. In a fixed point DSP implementation multiplications may require many more clock cycles than additions. Therefore, decoding of binary LDPC codes is commonly performed in the log-domain. The effects of a finite precision representation of the received data and the extrinsic information was investigated in [8]. In [6] an optimized quantization scheme for log-SPA was proposed, requiring changes to the decoder for non-linear operations. As the statistical properties of the extrinsic information change from one iteration to the next, an adaptive quantization scheme would reduce the number of quantization levels. Such an approach, which would of course increase the overall computational complexity, was studied in [9] in the context of turbo codes.

More recently, LDPC codes over *higher order Galois fields* were considered, resulting in more powerful error-correcting codes [10]. An SPA decoding algorithm was proposed, but an equivalent log-domain approach was not investigated. Consequently, no information about computational complexity or quantization effects is available.

In this paper we compare the computational complexity of SPA and log-SPA for LDPC codes over binary extension fields, $GF(q)$. We show that when multiplications require only few additional clock cycles as compared to additions, log-SPA decoding can be significantly more time-consuming than SPA decoding, especially with increasing field size. On the other hand, assuming fixed point quantization of the decoder inputs and/or extrinsic messages, we study the bit error rate (BER) vs. quantization trade-off. We demonstrate that log-SPA requires fewer quantization bits for the decoder inputs and has a lower error floor than SPA.

2. SYSTEM MODEL

The overall structure, consisting of the coder, mapper, channel, demapper and decoder is depicted in Fig. 1. One can examine the effect of quantization of the received signal (point A), of the decoder inputs (point B) or of the extrinsic messages (point C), or any combination thereof.

This work has been supported by the Interuniversity Attraction Poles Program P5/11- Belgian State - Federal Office for Scientific, Technical and Cultural Affairs.

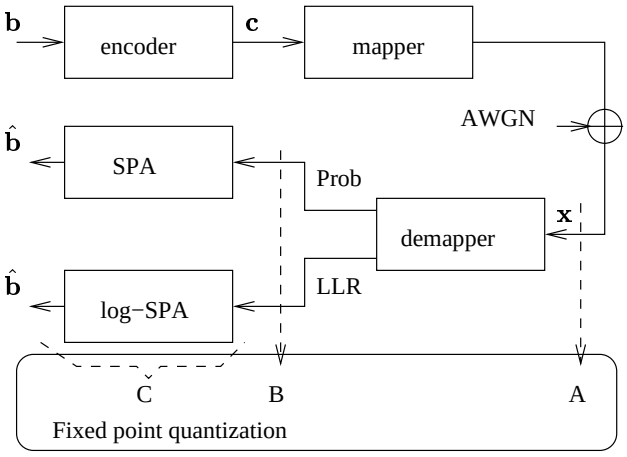

Fig. 1. Transceiver structure and quantization points

Transmitter

Let $\mathbf{H}$ denote the $M \times N$ sparse parity check matrix and $\mathbf{G}^T$ a corresponding $N \times K$ (with $K = N - M$) generator matrix. An information word $\mathbf{b}$ in $GF(q)$ is encoded to a coded word $\mathbf{c} = \mathbf{G}^T\mathbf{b}$, mapped to a constellation and transmitted over an AWGN channel, resulting in a received vector $\mathbf{x}$.

Receiver

The vector $\mathbf{x}$ is converted by the demapper to either probabilities or log likelihood ratios (LLR):

$$\mathbf{x} \rightarrow \left\{ \begin{array}{c} P[c_0 = a\,|\mathbf{x}], \ldots, P[c_N = a\,|\mathbf{x}] \\ LLR[c_0 = a], \ldots, LLR[c_N = a],\ a \neq 0 \end{array} \right.$$

for all $a \in GF(q)$, with

$$LLR[c_k = a] \doteq \ln \frac{P[c_k = a\,|\mathbf{x}]}{P[c_k = 0\,|\mathbf{x}]}.$$

The probabilities or the LLRs are then forwarded to the SPA or the log-SPA decoder, respectively.

The decoding algorithms are based on a Tanner graph representation of the sparse parity check matrix [5]. This bipartite graph, say Γ, consists of N variable nodes, associated with the N coded symbols, and M check nodes, associated with the M checks (i.e., the M rows in $\mathbf{H}$). $\mathbf{H}$ is the edge-labeled reduced adjacency matrix of Γ. When $H_{m,n} \neq 0$, coded symbol n is checked by check m. Hence, in Γ, check node m and variable node n are connected. The corresponding edge has label $H_{m,n}$.

The sum-product algorithm operates by passing messages between variable and check nodes:

1. Initialization step: variable nodes are initialized with the belief of the corresponding coded symbol, based solely on the received vector $\mathbf{x}$.
2. Tentative decoding: variable node n computes, based on the information from the channel vector $\mathbf{x}$ and messages from adjacent check nodes, the most likely value of symbol n, $\hat{c}_n$. If $\mathbf{H}\hat{\mathbf{c}} = \mathbf{0}$ STOP.
3. Horizontal step: a message is passed from variable node n to adjacent check node m, expressing the belief of the n-th symbol, given all the information from *all* connected check nodes, *except* check node m itself.
4. Vertical step: each check node m sends a message to adjacent variable node n, reflecting the belief of the n-th symbol, given all the information from the channel *and* all variable nodes connected to check node m, *except* variable node n itself. GOTO step (2).

A decoding failure is declared if, after a fixed number of iterations, no valid codeword has been produced in step 2. The vague terms "belief" and "information" can be expressed in terms of probabilities or LLRs, resulting in an SPA or a log-SPA decoder, respectively. The core operations in SPA are real-valued additions and multiplications of messages, while for log-SPA these are real-valued additions and max^*-operations, where $max^*(x_1, x_2) \doteq \ln\left(e^{x_1} + e^{x_2}\right)$.

3. IMPLEMENTATION ISSUES

3.1. Quantization

One can investigate the effect of quantization of the signals at the three points (A,B,C) in Fig. 1. The effect of fixed point *input quantization* (point A) for binary LDPC codes was considered in [8], where it is shown that in general 4 bits are sufficient. *Extrinsic message quantization* (C) was investigated in [8] and [6]. However, [6] does not employ fixed point quantization in the strict sense and requires extra look-up tables for non-linear operations. Neither [8] nor [6] take into account the changing statistics of the extrinsic messages by using an adaptive quantization scheme, as proposed for turbo codes [9, 11].

Here, for LDPC codes over $GF(q)$, we make the common assumption of infinite quantization of the input signal (point A). Similar to [11] in a turbo-coded context, we first consider *decoder input message quantization* (point B) to find the minimum required word length that yields an acceptable BER degradation. Using this word-length, (non-adaptive) quantization is applied to the signals at points B and C and the resulting BER degradation is determined. In both cases, we restrict ourselves to fixed-point quantization.

Under a fixed point quantization scheme a real number (say x) is mapped to a binary sequence $\mathbf{a} = [a_0 \ldots a_{b-1}]$. Of these b bits, the last f bits represent the fractional part.

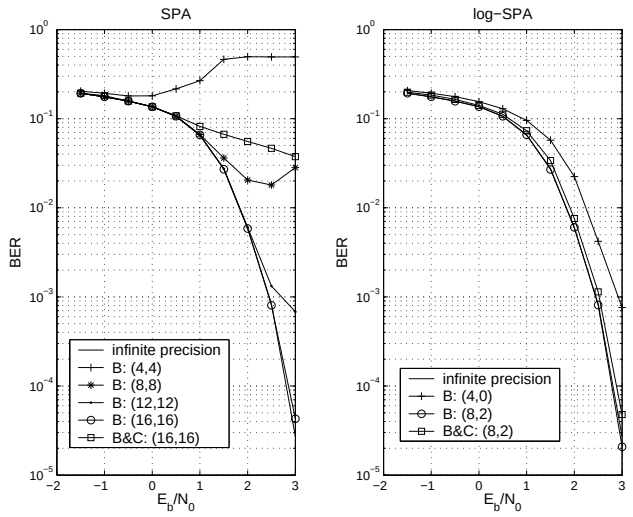

Fig. 2. quantization effects on SPA and log-SPA for an LDPC code over $GF(4)$

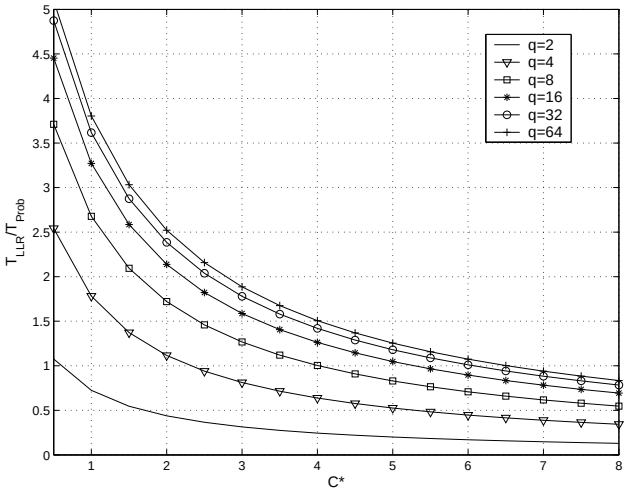

Fig. 3. ratio of decoding time for log-SPA and SPA as a function of $C*$ with the field size as parameter

When both positive and negative numbers need to be represented, the first bit (a_0) is used as a sign bit. In both cases we will denote the quantization scheme by (b, f). The mapping as defined as follows:

$$x \rightarrow \begin{cases} (-1)^{a_0}\left(\sum_{k=1}^{b-1} a_k 2^{b-f-1-k}\right) & \text{(signed)} \\ \sum_{k=0}^{b-1} a_k 2^{b-f-1-k}. & \text{(unsigned)} \end{cases}$$

Observe that the accuracy of the fixed point representation is determined by f, while $b-f$ is related to the dynamic range [12]. For SPA *probabilities* are converted to fixed point. Since these lie within the interval $(0, 1)$ we set $f = b$. In Log-SPA, however, *LLRs* are quantized. In that case, since the domain is now $(-\infty, +\infty)$, we can vary f between 0 and $b-1$ in order to find an optimal trade-off between accuracy and dynamic range. It is important to note that in a fixed point quantization scheme (as opposed to an arbitrary quantization scheme) additions and multiplications can be performed efficiently by the DSP.

Fig. 2 shows simulation results for a rate $1/2$ LDPC code over $GF(4)$ from [13]. We assume BPSK mapping. Decoder input message quantization in SPA leads to a significant degradation at higher SNR when b is too low. The minimum value for b resulting in negligible BER degradation is 16 bits. Keeping this value for b and performing SPA with (B&C) quantization again leads to very high BER degradations. It turns out that this is due to the insufficient accuracy during the normalization of the extrinsic messages (i.e., in order that the probabilities add up to one).

On the right part of Fig. 2 we observe that a fixed point input message quantization scheme with 4 and 8 bits results in BER degradation around 0.5 and 0.01 dB, respectively. Also, there is no error floor visible, even for higher SNR. Keeping $(8, 2)$ quantization and performing log-SPA with (B&C) quantization leads to fairly low degradations (i.e., below 0.1 dB). We remind that in log-SPA no normalization is performed. This may account for the excellent performance of the (B&C) quantization scheme, even without resorting to adaptive quantization. Finally, we note (results not shown) that for log-SPA with (B&C) quantization, no undetected errors occurred.

3.2. Computational complexity

We denote by u (resp. t) the mean row weight[1] (resp. column weight) of the parity check matrix $\mathbf{H}$. It can be shown that SPA has a computational complexity of $\mathcal{O}\left(q^2 M u\right)$ per iteration [10]. In Table 1 we compare the number of additions, multiplications and max^* operations on messages. Clearly, log-SPA has the same order of computational complexity as SPA. We will make the assumption that computations in $GF(q)$ are stored in tabular form and that a max^* operation requires 3 times as many clock cycles as an addition. We denote by $C*$ the ratio of the number of clock cycles per fixed point multiplication to the number of clock cycles per fixed point addition. This allows us to determine the decoding time in clock cycles of SPA and log-SPA, denoted by T_{Prob} and T_{LLR}, respectively. Fig. 3 shows, as a function of $C*$, the ratio of these times for various values of the field size (q) for $t = 3$ and $u = 6$. Note that $C* = 1$ corresponds to current floating point DSP or dedicated hardware fixed point implementations. In that case, log-SPA has a shorter computation time only for $q = 2$. As $C*$ increases, log-SPA becomes more attractive, even for large fields. For example, if a multiplication requires four times the number

[1] the number of non-zero entries

Table 1. operations per iteration of SPA and log-SPA decoding for the horizontal (H) and vertical (V) step

	+	*	max^*
SPA(H)	$(3u-4)\,Mq^2$	$(3u-4)\,Mq^2$	0
SPA(V)	uMq	$2Mtuq$	0
log-SPA(H)	$2\,(3u-4)\,M\,(q-1)^2$	0	$2\,(3u-4)\,M\,(q-1)^2$
log-SPA(V)	$uM\,(2t-1)\,(q-1)$	0	0

of clock cycles of an addition (i.e., $C_* = 4$), log-SPA has a shorter computation time only for fairly small fields of size $q = 2$ and $q = 2^2$.

4. CONCLUSION

We have compared the sum-product decoding algorithm (SPA) with its log-domain version (log-SPA) for LDPC codes over binary extension fields. As far as the BER vs. fixed point quantization trade-off is concerned, log-SPA has a distinct advantage as it requires fewer quantization bits and has a lower error floor. However, for certain DSP architectures (for which a multiplication takes only slightly more time than an addition) SPA has a shorter decoding time when $q > 2$.

5. REFERENCES

[1] R.G. Gallager. "Low density parity-check codes". *IRE Trans. Inform. Theory*, IT-8:pp. 21–29, Jan. 1962.

[2] D.J.C. MacKay. "Good error-correcting codes based on very sparse matrices". *IEEE Trans. Inform. Theory*, 45(2):pp. 399–431, March 1999.

[3] T. Richardson, M. Shokrollahi and R. Urbanke. "Design of capacity approaching low-density parity-check codes". *IEEE Trans. Inform. Theory*, 47(2):619–637, February 2001.

[4] J. Pearl. *"Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference"*. Morgan Kaufmann, San Mateo, 1988.

[5] F. Kschinschang, B. Frey and H.-A. Loeliger. "Factor graphs and the sum-product algorithm". *IEEE Trans. Inform. Theory*, 47(2):pp.498–519, February 2001.

[6] L. Ping and W.K. Leung. "Decoding low density parity check codes with finite quantization bits". *IEEE Comm. Letters*, 4(2):pp.62–64, February 2000.

[7] J. Hagenauer, E. Offer, C. Méasson and M. Mörz. "Decoding and equalization with analog non-linear networks". *European Trans. Comm.*, 10:pp.659–680, November 1999.

[8] T. Zhang, Z. Wang and K.K. Pahri. "On finite precision implementation of low density parity check codes". In *Proc. ISCAS*, Sydney, Australia, May 2001.

[9] R. Hoshyar, A.R.S. Bahai and R. Tafazolli. "Finite precision turbo decoding". In *Proc. 3rd Int. Symposium on Turbo Codes & Related Topics*, pages 483–486, Brest, France, September 2003.

[10] M.C. Davey and D.J.C. MacKay. "Low density parity check codes over GF(q)". *IEEE Comm. Letters*, 2(6):pp.165–167, June 1998.

[11] G. Montorsi and S. Benedetto. "Design of fixed point iterative decoders for concatenated codes with interleavers". *IEEE Journal on Selected Areas in Comm.*, 19(5):871–882, May 2001.

[12] H. Michel, A. Worm and N. Wehn. "Influence of quantization on the bit-error performance of turbo-decoders". In *Proc. VTC Spring*, Tokyo, Japan, May 2000.

[13] D.J.C. MacKay. "Online database of low-density parity check codes". http://www.inference.phy.cam.ac.uk/mackay /codes/data.html.