On Optimal and Near-Optimal Turbo Decoding
Using Generalized $\max^*$ Operator

Stylianos Papaharalabos, Member, IEEE, P. Takis Mathiopoulos, Senior Member, IEEE,
Guido Masera, Senior Member, IEEE, and Maurizio Martina, Member, IEEE.

Abstract—Motivated by a recently published robust geometric programming approximation, a generalized approach for approximating efficiently the $\max^*$ operator is presented. Using this approach, the $\max^*$ operator is approximated by means of a generic and yet very simple $\max$ operator, instead of using additional correction term as previous approximation methods require. Following that, several turbo decoding algorithms are obtained with optimal and near-optimal bit error rate (BER) performance depending on a single parameter, namely the number of piecewise linear (PWL) approximation terms. It turns out that the known Max-Log-MAP algorithm can be viewed as special case of this new generalized approach. Furthermore, the decoding complexity of the most popular previously published methods is estimated, for the first time, in a unified way by hardware synthesis results, showing the practical implementation advantages of the proposed approximations against these methods.

Index Terms—Turbo codes, iterative decoding.

I. INTRODUCTION

O

VER the past decade or so several algorithmic approaches aiming to simplify the well-known $\max^*$ operator [1] for decoding turbo codes [2] have been proposed and analyzed. These algorithms include: Improved Max-Log-MAP [3], Constant Log-MAP [4], Linear Log-MAP [5], Average Log-MAP [6], and recently the algorithms in [7], [8].

The penalty paid for such approximations is a small bit error rate (BER) performance degradation as compared with the performance achieved by the optimal Log-MAP algorithm [9]. However, although these sub-optimal algorithms are computationally simpler as compared with the optimal solution, an additional correction term is required to be added to the $\max$ operation in order to minimize performance degradation.

Geometric programming is an optimization problem used in various fields, such as information theory, analog/RF circuit design, power control of wireless communication networks and statistics [10]. However, its application to turbo decoding has not been investigated so far. Recently, the authors in [11] have dealt with the convex log-sum-exp (lse) function and considered a robust geometric programming problem as robust linear programming problem. In particular, a constructive algorithm was presented in order to find the best piecewise linear (PWL) approximation terms of the bivariate lse function. Their research has shown that the exact number of PWL terms depends on the approximation error resulting from the original bivariate lse function.

In this letter, we modify and apply for the first time the optimal approximations presented in [11] to decode turbo codes noting that the bivariate lse function is equivalent to the $\max^*$ operator. Consequently, a generalized approximation for the $\max^*$ operator is obtained using: (i) a small number of efficient PWL terms that ease a hardware implementation; and (ii) the very simple $\max$ operator. The main feature of this approximation is that there is no need to use any correction term, in contrast to all previously published methods [4]–[8]. Furthermore, it turns out that the known Max-Log-MAP algorithm can be viewed as special case of this new approach, when two PWL terms are considered. Our research has shown that by considering four PWL approximation terms with the additional use of scaling as in [3], the resulting turbo code BER performance is essentially identical to the performance of the optimal Log-MAP algorithm. Hardware synthesis results have shown that by considering three PWL approximation terms, the resulting algorithm outperforms most of the previously published methods, i.e. [5]–[8], in terms of occupied area savings, and achieves near Log-MAP performance.

II. REDUCED COMPLEXITY TURBO DECODING

In this section the most important reduced complexity turbo decoding algorithms are reviewed. Consider an information sequence of $N$ bits denoted with $\bar{u} = [u_1, u_2, \ldots, u_N]$. This sequence is turbo encoded, then each coded bit is binary phase-shift keying (BPSK) modulated, taking values from the alphabet $\{\pm 1\}$ with equal probabilities, and transmitted with bit energy $E_b$ over an additive white Gaussian noise (AWGN) channel with one-sided power spectral density $N_0$. At the receiver the turbo decoder estimates the transmitted sequence of bits.

The $\max^*$ operation, i.e. Jacobian logarithm, used in turbo decoding is defined as [1]

$$\max^*(x_1, x_2) \triangleq \log\{\exp(x_1) + \exp(x_2)\} = \max(x_1, x_2)$$

$$+ \ln\{1 + \exp(-|x_1 - x_2|)\} = \max(x_1, x_2) + f_c(|x_1 - x_2|)$$

(1)

where $f_c(|x_1 - x_2|)$ is a non-linear function referred to as ‘correction term’ [9] and $|.|$ denotes absolute value. For more
than two arguments, the Jacobian logarithm is applied recursively. For example, considering three arguments, it yields
\[
\max^*(x_1, x_2, x_3) = \max^* \{ \max^*(x_1, x_2), x_3 \}.
\] (2)

For the Log-MAP algorithm, a look-up table (LUT) substitutes \( f_\ell(x_1 - x_2) \), which is usually implemented with eight values [9]. If LUT is omitted, then the Log-MAP simplifies to the Max-Log-MAP algorithm. In the past, several reduced complexity decoding algorithms have been obtained by approximating \( f_\ell(|x_1 - x_2|) \) with different methods, e.g. see [4]–[8], a summary of which is presented in Table I.

Having evaluated the performance of reduced complexity decoding algorithms given in Table I and as shown in Fig. 1 (for the turbo code simulation parameters see the next section), it is concluded that at BER of 10\(^{-4}\); (i) The Max-Log-MAP is the worst performing algorithm with approximately 0.4 dB degradation as compared with Log-MAP; (ii) The maximum performance degradation of reduced complexity algorithms as compared with Log-MAP is approximately 0.1 dB; and (iii) the Linear Log-MAP and also the algorithm of [7] achieve the best, i.e. near Log-MAP performance. A thorough complexity estimation of these algorithms is reported in the next section.

### III. Optimal \( \max^* \) Approximations and Their Application to Turbo Decoding

From pure mathematical curiosity, the authors in [11] have, instead of approximating \( f_\ell(|x_1 - x_2|) \), approximated (1) as a whole, i.e. the \( \max^* \) operator directly. Hence, (1) becomes
\[
\max^* (x_1, x_2) = \max(\kappa_1 + x_1 + \lambda_1 x_2 + \mu_1, \ldots, \kappa_i x_1 + \lambda_i x_2 + \mu_i)
\] (3)
where \( \kappa_i, \lambda_i, \) and \( \mu_i \) are real positive values and \( i \geq 2 \). The best PWL approximations of the \( \max^* \) operator with different number of terms are shown in Table II. The approximation error reduces in the order of \( \sqrt{2}/r^2 \) and for practical applications \( 5 \leq r \leq 10 \) has been considered [11]. It is underlined that in case of turbo decoding, the \( r = 2 \) approximation is identical to the Max-Log-MAP algorithm.

Performance evaluation results for various values of \( r \), with the additional use of scaling as in [3], have shown that at BER of 10\(^{-5}\); (i) Both \( r = 5 \) and \( r = 4 \) approximations achieve essentially identical to the Log-MAP performance; and (ii) The \( r = 3 \) approximation has performance degradation of less than 0.03 dB against the Log-MAP algorithm. In order to ease a hardware implementation, the \( r = 3 \) and \( r = 4 \) approximations have been modified, respectively as
\[
\max^* (x_1, x_2) = \max (x_1, 0.5 (x_1 + x_2 + 1), x_2)
\] (4)
\[
\max^* (x_1, x_2) = \max (x_1, 0.25 (x_1 + 0.75 x_2 + 0.5), 0.75 x_1 + 0.25 x_2 + 0.5, x_2).
\] (5)

In terms of implementation, synthesizable VHDL descriptions have been produced for the \( \max^* \) approximations of (4) and (5) as well as for Log-MAP with LUT, Max-Log-MAP and the algorithms shown in Table I. In order to derive fair comparisons, the same area optimization effort of the synthesis tool must be guaranteed for all cases. To this purpose, although all considered implementations of the \( \max^* \) operation are pure combinational architectures, registers have been placed at the architecture inputs and output. This allows setting a unique clock frequency constraint for all considered cases, \( f_{\text{CK}} = 200 \text{ MHz} \). Synthesis results obtained in terms of area occupied by the combinational part on a 130 nm standard cell CMOS technology are given in Table III for precision metrics represented with 8, 10 and 12 bits, respectively. These results show, on the one hand, that the proposed \( r = 3 \) approximation outperforms most of the previously published methods, i.e. [5]–[8]. In particular, its occupied area is 35% smaller than that required by the Log-MAP algorithm and it is only inferior to the Constant Log-MAP algorithm by 8%. On the other hand, the proposed \( r = 4 \) approximation has comparable complexity with the method in [7] and outperforms [5], [6], [8].

Performance evaluation results have been obtained for the most efficient method, i.e. Constant Log-MAP, the \( r = 3 \) and \( r = 4 \) proposed approximations, and these are illustrated in Fig. 2. A 16-states turbo code is considered with coding rate equal to 1/2 and generator polynomials \( (1, 33/23)_{o} \) in octal form representing the feed-forward and backward polynomials, respectively. Furthermore, an information sequence of \( N = 10^5 \) bits is assumed, whereas the total number of transmitted frames is \( 10^5 \). A pseudo-random turbo interleaver is considered and at the receiver a maximum of 10 decoding iterations are performed. In order to reduce computer simulation time, a genie stopping rule is assumed at the turbo decoder. In computer-based simulations the scaling factor, denoted with \( s \), was constant when varying the number of decoding iterations, having the following values: (i) 0.65 for Max-Log-MAP; (ii) 0.9 for Log-MAP; (iii) 0.85 for Constant Log-MAP; and (iv) 0.75 for the rest of the algorithms.

As shown in Fig. 2, the best performance is achieved by the Constant Log-MAP algorithm and the proposed \( r = 4 \) approximation followed by the \( r = 3 \) approximation. In more detail, at BER of 10\(^{-4}\) and with respect to the Log-MAP algorithm: (i) The Constant Log-MAP algorithm has performance degradation of approximately 0.03 dB; (ii) The \( r = 4 \) approximation has comparable performance with the Constant Log-MAP algorithm; and (iii) The \( r = 3 \) approximation has performance degradation of 0.05 dB. However, both \( r = 3 \) and \( r = 4 \) approximations offer practical implementation advantages with respect to other methods.

### IV. Conclusion

It has been shown that the \( \max^* \) operator used in turbo decoding can be simplified into different number of PWL approximation terms in an efficient way using \( \max \) only operation. The proposed \( r = 3 \) approximation is 0.05 dB inferior to the Log-MAP algorithm but 35% much simpler. Further, it outperforms, in terms of occupied area savings, most of the previously published methods, such as Linear Log-MAP [5], Average Log-MAP [6], and the algorithms from [7], [8]. The proposed \( r = 4 \) approximation is only 0.03 dB inferior to the Log-MAP algorithm and it is less complex than the methods presented in [5], [6], [8].
TABLE I
THE MOST IMPORTANT REDUCED COMPLEXITY TURBO DECODING ALGORITHMS USING DIFFERENT APPROXIMATION IN \( f_e(\lvert x_1 - x_2 \rvert) \) OF (1).

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( f_e(\lvert x_1 - x_2 \rvert) )</td>
<td>( \text{max}(\ln 2 - 0.25 \ast \lvert x_1 - x_2 \rvert, 0) )</td>
<td>( \text{max}(0, \ln 2 - 0.5 \ast \lvert x_1 - x_2 \rvert) )</td>
</tr>
<tr>
<td></td>
<td>( 3/8, \text{if } \lvert x_1 - x_2 \rvert &lt; 2 ) ( , 0, \text{otherwise} )</td>
<td>( \text{max}(\ln 2 - 0.25 \ast \lvert x_1 - x_2 \rvert, 0) )</td>
<td>( \text{max}(0, \ln 2 - 0.5 \ast \lvert x_1 - x_2 \rvert) )</td>
</tr>
<tr>
<td>( f_e(\lvert x_1 - x_2 \rvert) )</td>
<td>( \ln 2 \ast 2^{-</td>
<td>x_1-x_2</td>
<td>} )</td>
</tr>
</tbody>
</table>

TABLE II

<table>
<thead>
<tr>
<th>No. of Terms (( r ))</th>
<th>Resulting max* Approximation</th>
<th>No. of max Ops.</th>
<th>Approx. Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>( \text{max}(x_1,x_2) )</td>
<td>1</td>
<td>0.093</td>
</tr>
<tr>
<td>3</td>
<td>( 0.5 \ast (x_1 + x_2) + 0.693, x_2) )</td>
<td>2</td>
<td>0.223</td>
</tr>
<tr>
<td>4</td>
<td>( \text{max}(x_1, \text{max}(x_1 + x_2 + 0.584, x_2) , x_2) )</td>
<td>3</td>
<td>0.109</td>
</tr>
<tr>
<td>5</td>
<td>( \text{max}(x_1, \text{max}(x_1 + 0.833 \ast x_2 + 0.45, x_2), x_2) )</td>
<td>4</td>
<td>0.065</td>
</tr>
</tbody>
</table>

TABLE III
OCCUPIED AREA COMPARISON (SQUARE \( \mu \text{m} \)) OF DIFFERENT max* APPROXIMATIONS ON A 130 NM STANDARD CELL TECHNOLOGY AND VARIOUS PRECISION METRICS REPRESENTATIONS.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>8 bits</th>
<th>10 bits</th>
<th>12 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max-Log-MAP</td>
<td>250.133</td>
<td>312.666</td>
<td>373.182</td>
</tr>
<tr>
<td>Log-MAP</td>
<td>1022.72</td>
<td>1276.888</td>
<td>1613.786</td>
</tr>
<tr>
<td>( r = 4 ) of (4)</td>
<td>635.575</td>
<td>831.886</td>
<td>1066.385</td>
</tr>
<tr>
<td>( r = 4 ) of (5)</td>
<td>883.534</td>
<td>1149.804</td>
<td>1426.160</td>
</tr>
<tr>
<td>Constant Log-MAP [4]</td>
<td>599.108</td>
<td>764.519</td>
<td>931.946</td>
</tr>
<tr>
<td>Linear Log-MAP [5]</td>
<td>978.342</td>
<td>1264.784</td>
<td>1559.290</td>
</tr>
<tr>
<td>Average Log-MAP [6]</td>
<td>1069.116</td>
<td>1363.627</td>
<td>1555.261</td>
</tr>
<tr>
<td>Ref. [7]</td>
<td>891.602</td>
<td>1135.684</td>
<td>1377.748</td>
</tr>
<tr>
<td>Ref. [8]</td>
<td>1137.701</td>
<td>1456.418</td>
<td>1758.988</td>
</tr>
</tbody>
</table>

REFERENCES