


# Enhanced Loop-weakened Belief Propagation Algorithm for Performance Enhanced Polar Code Decoders

Arvid B. van den Brink \*  
Email: a.b.vandenbrink@utwente.nl

Marco J.G. Bekooij\*<sup>†</sup>  
Email: marco.bekooij@nxp.com

\*Department of Computer Architectures  
for Embedded Systems  
University of Twente, Enschede, The Netherlands

<sup>†</sup>Department of Embedded Software  
and Signal Processing  
NXP Semiconductors, Eindhoven, The Netherlands

**Abstract**—A polar code decoder based on the belief propagation algorithm is desirable because of the potentially low latency and its suitability for parallel execution on multicore and SIMD processors. However, current state-of-the-art algorithms require many iterations to achieve comparable bit and frame error rate compared to successive cancellation algorithms. Also, the current state-of-the-art belief propagation algorithms have a high computational complexity compared to successive cancellation.

In this paper we present an enhanced belief propagation algorithm, in which parts of the computations are altered to reduce the negative effect of the short cycles in the polar code factor graph.

Our proposed algorithm has a gain of  $\approx +0.4$ dB in both frame and bit error rate compared to successive cancellation and a gain of  $\approx +0.16$ dB and  $\approx +0.13$ dB at a frame and bit error rate of  $10^{-3}$  respectively, compared to belief propagation. Also, the maximum number of iterations of our algorithm is reduced to  $0.6 \cdot I_{max}$ . As a result, the latency is up to  $\approx 11$  times lower compared to successive cancellation and up to  $\approx 1.8$  times lower compared to the current state-of-the-art belief propagation polar code algorithm. Furthermore, the reduction in the maximum iteration count results in a lower power consumption after implementation.

**Index Terms**—Performance Enhanced, Belief Propagation, Polar Code, Algorithm

## I. INTRODUCTION

Polar codes are the first error-correcting code to achieve the symmetric capacity of binary-input discrete memoryless channel (B-DMC) with a low complexity implementation using the successive cancellation (SC) algorithm as proposed by Arikan [1] as the block length reaches infinity. However, the error correcting performance of the polar code SC algorithm degrades if the code length is degraded, since the polarization effect is not fully utilized for these code lengths. Also the serial nature of the SC algorithm results in a low decoding throughput of the SC algorithm [2] compared to, e.g. the low-density parity check (LDPC) or convolutional Turbo code (CTC) algorithms.

Multiple possible solutions have recently been proposed to improve both performance and throughput. The performance can be improved by using the successive cancellation list

(SCL) algorithm or the CRC-aided successive cancellation list (CA-SCL) [3] algorithm. However, these approaches still suffer from a low decoding throughput, because these algorithms are based on the SC algorithm [2].

Among the throughput improving solutions, the simplified successive cancellation (SSC) [4] and fast successive cancellation (FSC) [5] algorithms give the highest improvement over the original SC algorithm. Polar codes of length  $N$  are formed from two constituent polar codes of length  $N/2$ . Using this recursive nature, the decoding of the constituent polar codes can be performed directly and without recursion, which increases the throughput of the decoder.

In this work we focus on improving the performance, in both error correction performance and latency, of polar code decoders by enhancing the belief propagation (BP) algorithm. By adapting the computations of some parts of the BP algorithm, this results in that the decoder converges faster, hence less iterations are required. We present a polar code algorithm that, for a  $\mathcal{P}(1024, 512)$  polar code decoder at  $10^{-3}$ , has an signal to noise ratio (SNR) gain in frame error rate (FER) of  $\approx +0.4$ dB, and an SNR gain in bit error rate (BER) of  $\approx +0.15$ dB, with respect to both the SC and BP algorithm respectively. Additionally, the latency of the proposed algorithm is, due to the lower required number of iterations, approximately 7.7 to 11 times lower compared to the SC algorithm [1] and approximately 0.95 to 2.2 times lower compared to the BP [6] algorithm. The proposed algorithm is not bounded by code length or code rate.

The remainder of this paper is organized as follows. In Section II we present related work. In Section III we give a review of polar codes, and the SC, and BP algorithms. Our proposed enhanced algorithm is presented in Section V, including a more formal explanation of why our modification of the BP algorithm improves its performance. In Section VI we describe the used methodology and the simulation results. Finally, in Section VII we state the conclusions.

## II. RELATED WORK

As a capacity achieving code for any binary-input discrete memoryless channel (B-DMC) [1], polar codes are introduced. This type of codes has recently raised a lot of interest in the research community due to this property [3], [7], [8]. Although, for infinite code lengths in combination with successive cancellation (SC) decoding, this capacity achieving property has been proven, the error correcting performance for finite code length is not guaranteed to be optimal. Another hurdle to take is the inherently sequential decoding structure of the SC algorithm, resulting in a decreasing throughput if the code length increases. To reduce the degradation of the error correcting performance of the SC algorithm for shorter code lengths, the successive cancellation list (SCL) algorithm [3] was proposed. This algorithm explores multiple paths of the decoding tree and the best candidate is chosen as the decoder output, hence increasing the error correcting performance of the decoder, but at the cost of an increased computational complexity of the decoder.

Research has been performed on the belief propagation (BP) algorithm [6] in order to increase the performance in terms of error correcting capabilities, throughput, or latency of the polar code BP decoder. Unlike the SC algorithm, the BP algorithm can be executed in parallel to a high degree. To improve the error correction performance a list variant of the BP decoder was proposed in [9]. A latency efficient BP decoder was proposed in [10], by reducing the critical path delay. Several early stopping criteria have been proposed in [11]–[13], which also reduces the decoding latency. Architectural optimizations have been proposed in [14], to increase the throughput of the BP decoder. Other proposed methodes to improve the throughput of the polar code decoder are the vectorized version of the BP decoder in [15] and a stage combined BP decoder in [16].

## III. PRELIMINARY

### A. Polar codes

We assume a polar code, as in [1], to be defined by a parameter vector  $(N, K, \mathcal{A}, u_{\mathcal{A}^c})$ , where  $N = 2^n$ ,  $\forall n > 1$  is the length of the codeword,  $K$  the number of information bits,  $\mathcal{A}$  the set of indices of the information bits, and  $u_{\mathcal{A}^c}$  representing the frozen bits. The encoding and decoding process is represented as  $x = uF^{\otimes n}$ , where  $u$  is the input vector and  $F^{\otimes n}$  is the  $n$ -th Kronecker power of  $F = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ , where  $n = \log_2 N$ .

### B. Successive cancellation polar decoding

Fig. 1a shows the tree representation of a  $\mathcal{P}(8, 4)$  polar code with the root node initialized with the vector  $\alpha_n^{0 \rightarrow N}$ , denoting the channels log likelihood ratios (LLRs), defined as  $\alpha_n^j = \ln(\Pr[y_j | x_j = 0] / \Pr[y_j | x_j = 1])$ . Each node will compute their respective  $\alpha_s^j$  and  $\beta_s^j$ , using the min-sum approximation according to [2]:

$$\alpha_s^j \approx \varphi(\alpha_{s+1}^j \cdot \alpha_{s+1}^{j+2^s}) \min(|\alpha_{s+1}^j|, |\alpha_{s+1}^{j+2^s}|), \quad (1)$$

with  $\varphi$  is the sign function, or

$$\alpha_s^{j+2^s} = \begin{cases} \alpha_{s+1}^{j+2^s} + \alpha_{s+1}^j, & \text{if } \beta_s^j \geq 0; \\ \alpha_{s+1}^{j+2^s} - \alpha_{s+1}^j, & \text{otherwise,} \end{cases} \quad (2)$$

where  $\beta_s^j$  is the modulo-2 sum of the decoded bits. Equation (1) is known as the  $f$ -function and (2) as the  $g$ -function.

### C. Belief propagation polar decoding

Based on [6], a belief propagation decoder for polar codes is constructed using the same factor graph representation as used for the SC algorithm. Fig. 1b shows an example of the factor graph for the case of an  $N = 4$  BP polar decoder, where each computational unit (CU) has four terminals, as shown in Fig. 2a, which map to:

$$L_{s,i}^{(m+1)} = f(L_{s+1,i}^{(m)}, L_{s+1,i+2^{n-s}}^{(m)} + R_{s,i+2^{n-s}}^{(m)}) \quad (3)$$

$$L_{s,i+2^{n-s}}^{(m+1)} = f(R_{s,i}^{(m)}, L_{s+1,i}^{(m)}) + L_{s+1,i+2^{n-s}}^{(m)} \quad (4)$$

$$R_{s+1,i}^{(m+1)} = f(R_{s,i}^{(m)}, L_{s+1,i+2^{n-s}}^{(m)} + R_{s,i+2^{n-s}}^{(m)}) \quad (5)$$

$$R_{s+1,i+2^{n-s}}^{(m+1)} = f(R_{s,i}^{(m)}, L_{s+1,i}^{(m)}) + R_{s,i+2^{n-s}}^{(m)} \quad (6)$$

The function  $f(x, y)$  is computed using (1). Each message is initially assigned an LLR depending on the side at which the node is located. The left most  $R$  messages are initialized with  $R_{0,i} = +\infty$  if  $i \in \mathcal{A}^c$ , where  $\mathcal{A}^c$  is the set of frozen bits, otherwise  $R_{0,i} = 0$ . The right most  $L$  messages are initialized with the channel LLR. All other, intermediate messages in the factor graph are initially set to zero (0) and updated iteratively using (3)–(6). After  $M$  iterations,  $\hat{u}_1^N$  can be decided using threshold detection at the left most terminals.

## IV. BASIC IDEA

With a careful comparison of the left bound LLR calculations of both the equations (1) and (3) of the SC and BP algorithm, respectively, a difference in the form of the addition of an extra variable can be observed. This extra variable in (3) has a large effect on the behaviour of the BP algorithm. To outperform the SC algorithm a large number of iterations are required, with a much higher computational complexity as a result. In our proposal we replace (3) of the BP algorithm with a more general equation, which is applicable for the BP algorithm as well as the SC algorithm. By adapting a parameter in this equation, the performance measured in error rate as in latency can be improved.

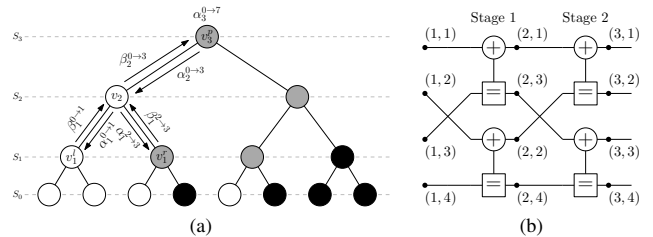


Fig. 1. (a) Tree representation of a  $\mathcal{P}(8, 4)$  polar code decoder; (b) Polar code factor graph for code length  $N = 4$ .

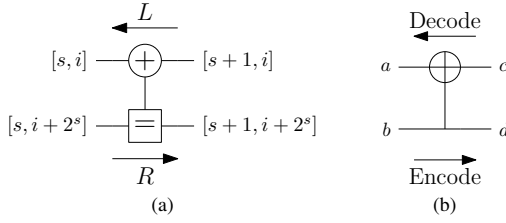


Fig. 2. (a) Belief propagation computational unit; (b) Basic polar code channel transformation.

## V. PROPOSED ENHANCED LOOP-WEAKENED BELIEF PROPAGATION ALGORITHM FOR POLAR CODES

For our proposal we will first give Lemma 1 on how the upper node input likelihood ratio (LR)  $a$  can be estimated using the other LRs  $b$ ,  $c$  and  $d$  of the basic polar code channel transformation as defined in Fig. 2b. These estimations are indicated as  $\hat{a}$ ,  $\hat{b}$ ,  $\hat{c}$  and  $\hat{d}$ .

**Lemma 1.** *For the estimation  $\hat{a}$  of the upper input node  $a$  of the basic polar code channel transformation, as shown in Fig. 2b, the estimation  $\hat{b}$  of the lower input bound node  $b$  is not required. Therefore  $LLR(\hat{a}) = f(LLR(\hat{c}), LLR(\hat{d}))$ , where  $f(x, y)$  is computed using (1).*

*Proof:* Using the basic polar code channel transformation, as shown in Fig. 2b, to decode the channel, it is trivial that the relationship between the estimated output values  $\hat{a}$  and  $\hat{b}$  and the estimated input values  $\hat{c}$  and  $\hat{d}$  are derived as follows:  $\hat{a} = \hat{c} \oplus \hat{d}$  and  $\hat{b} = \hat{d}$ . Similar to the SC polar code algorithm, the probabilities of  $\hat{c}$  and  $\hat{d}$  are the inputs of the basic channel transformation, Fig. 2b, to compute the probabilities of  $\hat{a}$  and  $\hat{b}$ .

Consider that  $\Pr(\hat{a} = s) \triangleq \Pr(\hat{a} = s, \hat{u}_1^{i-1} | y)$  where  $s \in \{0, 1\}$ . In the case that  $\hat{a} = 0$ , there are two possible combinations of  $\hat{c}$  and  $\hat{d}$ :  $\hat{c} = \hat{d} = 0$  or  $\hat{c} = \hat{d} = 1$ . Therefore,  $\Pr(\hat{a} = 0) = \Pr(\hat{c} = 0) \Pr(\hat{d} = 0) + \Pr(\hat{c} = 1) \Pr(\hat{d} = 1)$ . In the case that  $\hat{a} = 1$ , the two possible combinations of  $\hat{c}$  and  $\hat{d}$  are:  $\hat{c} = 0$  and  $\hat{d} = 1$  or  $\hat{c} = 1$  and  $\hat{d} = 0$ . Therefore,  $\Pr(\hat{a} = 1) = \Pr(\hat{c} = 0) \Pr(\hat{d} = 1) + \Pr(\hat{c} = 1) \Pr(\hat{d} = 0)$ .

The LR of  $\hat{a}$  is defined as  $LR(\hat{a}) = \Pr(\hat{a} = 0) / \Pr(\hat{a} = 1)$ , hence

$$\begin{aligned} LR(\hat{a}) &= \frac{\Pr(\hat{a} = 0)}{\Pr(\hat{a} = 1)} \\ &= \frac{\Pr(\hat{c} = 0) \Pr(\hat{d} = 0) + \Pr(\hat{c} = 1) \Pr(\hat{d} = 1)}{\Pr(\hat{c} = 0) \Pr(\hat{d} = 1) + \Pr(\hat{c} = 1) \Pr(\hat{d} = 0)} \\ &= \frac{\Pr(\hat{c} = 0) \Pr(\hat{d} = 0)}{\Pr(\hat{c} = 0) \Pr(\hat{d} = 1) + \Pr(\hat{c} = 1) \Pr(\hat{d} = 0)} \\ &\quad + \frac{\Pr(\hat{c} = 1) \Pr(\hat{d} = 1)}{\Pr(\hat{c} = 0) \Pr(\hat{d} = 1) + \Pr(\hat{c} = 1) \Pr(\hat{d} = 0)} \\ &= \frac{\Pr(\hat{c} = 1) \Pr(\hat{d} = 1) \left( \frac{\Pr(\hat{c} = 0) \Pr(\hat{d} = 0)}{\Pr(\hat{c} = 1) \Pr(\hat{d} = 1)} + 1 \right)}{\Pr(\hat{c} = 1) \Pr(\hat{d} = 1) \left( \frac{\Pr(\hat{c} = 0)}{\Pr(\hat{c} = 1)} + \frac{\Pr(\hat{d} = 0)}{\Pr(\hat{d} = 1)} \right)} \end{aligned}$$

$$\begin{aligned} &+ \frac{\Pr(\hat{c} = 1) \Pr(\hat{d} = 1)}{\Pr(\hat{c} = 1) \Pr(\hat{d} = 1) \left( \frac{\Pr(\hat{c} = 0)}{\Pr(\hat{c} = 1)} + \frac{\Pr(\hat{d} = 0)}{\Pr(\hat{d} = 1)} \right)} \\ &= \frac{\frac{\Pr(\hat{c} = 0)}{\Pr(\hat{c} = 1)} \cdot \frac{\Pr(\hat{d} = 0)}{\Pr(\hat{d} = 1)}}{\frac{\Pr(\hat{c} = 0)}{\Pr(\hat{c} = 1)} + \frac{\Pr(\hat{d} = 0)}{\Pr(\hat{d} = 1)}} + \frac{1}{\frac{\Pr(\hat{c} = 0)}{\Pr(\hat{c} = 1)} + \frac{\Pr(\hat{d} = 0)}{\Pr(\hat{d} = 1)}} \\ &= \frac{\frac{\Pr(\hat{c} = 0)}{\Pr(\hat{c} = 1)} \cdot \frac{\Pr(\hat{d} = 0)}{\Pr(\hat{d} = 1)} + 1}{\frac{\Pr(\hat{c} = 0)}{\Pr(\hat{c} = 1)} + \frac{\Pr(\hat{d} = 0)}{\Pr(\hat{d} = 1)}} = \frac{LR(\hat{c})LR(\hat{d}) + 1}{LR(\hat{c}) + LR(\hat{d})}. \end{aligned} \quad (7)$$

The LLR of  $\hat{a}$  equals to:

$$\begin{aligned} LLR(\hat{a}) &= \log \left( LR(\hat{a}) \right) = \log \left( \frac{LR(\hat{c})LR(\hat{d}) + 1}{LR(\hat{c}) + LR(\hat{d})} \right) \\ &\approx \varphi(LLR(\hat{c}) \cdot LLR(\hat{d})) \min(|LLR(\hat{c})|, |LLR(\hat{d})|) \\ &= f(LLR(\hat{c}), LLR(\hat{d})), \end{aligned} \quad (8)$$

which concludes the proof of Lemma 1.  $\blacksquare$

We look at the BP polar code algorithm, which is based on the loopy BP, referring to the Pearl polytree algorithm [17] for Bayesian networks with loops. This algorithm is an exact inference algorithm for singly connected networks, but will not give the correct beliefs for multiply connected networks.

When short cycles exists in the graph, as is the case in the polar code BP algorithm, after several iterations the beliefs from a variable node are propagated back to the same node. The correlations between the nodes in the graph result in a degradation of the algorithms decoding performance. For the estimation of  $\hat{a}$ , using the scaled version of (5), the following equation is used:

$$\hat{a} \approx s \cdot \varphi(\hat{c} \cdot (\hat{b} + \hat{d})) \min(|\hat{c}|, |\hat{b} + \hat{d}|). \quad (9)$$

It is trivial that the value of  $\hat{b}$  is introduced by the short cycles in the polar code construction. To reduce the effects of the short cycle, and therefore to reduce the possibility for the algorithm to get stuck in a local minimum, we come with the following proposition.

**Proposition 1.** *For the estimation  $\hat{a}$ , as shown in Fig. 2b, when using the BP algorithm for polar codes, a scaled version of the feedback of node  $b$  is required to weaken the influence of the short cycle.*

Following Proposition 1, we can rewriting (9):

$$\hat{a} \approx s_1 \cdot \varphi(\hat{c} \cdot (s_2 \cdot \hat{b} + \hat{d})) \min(|\hat{c}|, |s_2 \cdot \hat{b} + \hat{d}|). \quad (10)$$

In Lemma 2 we show in which interval the scaling factor  $s_2$  for the feedback node  $\hat{b}$  is located.

**Lemma 2.** *For scaling the feedback node  $\hat{b}$ , the factor  $s_2$  is in between 0 and 1, that is  $s_2 \in [0, 1]$*

*Proof:* For the lower bound of the interval we compare (8) to (10). Setting the two equations equal to each other we get the lower bound value for  $s_2$ .

$$\begin{aligned} s_1 \cdot \varphi(\hat{c} \cdot \hat{d}) \min(|\hat{c}|, |\hat{d}|) &= \\ s_1 \cdot \varphi(\hat{c} \cdot (s_2 \cdot \hat{b} + \hat{d})) \min(|\hat{c}|, |s_2 \cdot \hat{b} + \hat{d}|) \end{aligned} \quad (11)$$

It is trivial from (11) that there is equality if and only if  $s_2$  is equal to 0. Any value of  $s_2 < 0$  will change the sign of feedback node  $\hat{b}$ , introducing a bit-flip on this node, and therefore an error in the computation of  $\hat{a}$ , hence (11) is the lower bound.

For the upper bound of the interval we compare (9) to (10). Similar to the lower bound, setting the two equations equal to each other we get the upper bound value for  $s_2$ .

$$\begin{aligned} s_1 \cdot \varphi(\hat{c} \cdot (\hat{b} + \hat{d})) \min(|\hat{c}|, |\hat{b} + \hat{d}|) = \\ s_1 \cdot \varphi(\hat{c}) \cdot (s_2 \cdot \hat{b} + \hat{d}) \min(|\hat{c}|, |s_2 \cdot \hat{b} + \hat{d}|) \end{aligned} \quad (12)$$

It is trivial from (12) that there is equality if and only if  $s_2$  is equal to 1. Any value of  $s_2 > 1$  will increase the value of the feedback node  $\hat{b}$ , and therefore the influence of the feedback node is not weakened, hence (12) is the upper bound. Which concludes the proof of Lemma 2. ■

## VI. PERFORMANCE ANALYSIS AND COMPARISON

### A. Methodology

The proposed algorithm has been validated against a baseline implementation of both the original SC [1] and BP [18] algorithms. For a more elaborate comparison the proposed algorithm is also compared to the FSC [5] and SCL algorithms.

A Monte Carlo simulation is performed to evaluate the performance of the baseline and proposed algorithms. A polar code of different block lengths ( $n \in \{8, 10, 11\}$ ), using a systematic encoder, as described in [1], are used. The codeword is then modulated using binary phase-shift keying (BPSK) and send over an additive white Gaussian noise (AWGN) channel. The received signal is then demodulated, the LLRs are computed and fed to 3 variants of the proposed decoder, the baseline BP decoder, the baseline SC decoder, and the FSC and SCL decoders. For the baseline BP and proposed decoders, the maximum number of iterations  $I_{max} = 60$ . Decoding results in a BER and FER for the simulated SNR range. At the same time several other statistics are gathered, like average iteration count and average latency.

In the BP decoding algorithm the computational complexity of the nodes is equal for all nodes, see (3)-(6). The complexity of the upper node in the SC algorithm is slightly less compared to the BP algorithm and the lower node is even less complex, but still requires an addition and a multiplication if the scaled algorithm is used. For ease of use we assume that all nodes can be computed within 1 time unit, regardless of the number of internal operation, like additions and multiplications. We also assume that the highest possible parallelization is used. Therefore, for all the nodes in the algorithm that can be computed simultaneous, the latency is increased by 1.

### B. Proposed algorithm versus baseline algorithms

Before comparing the proposed enhanced belief propagation (eBP) algorithm to the original algorithms, a simulation with different scaling factors for the proposed algorithm are executed in order to find the scaling parameters which will be used for the rest of the comparisons. These simulations showed that

TABLE I  
SIMULATION PARAMETERS FOR THE SC, BP,  
AND THE PROPOSED ALGORITHMS

Algorithm	$s(s_1)$	$s_2$	$I_{max}$
SC [1]	0.9	—	—
SCL (List = 8)	0.9375	—	60
BP <sup>-4</sup> [18]	0.984375	0.25	60
proposed (eBP)	0.984375	0.25	60

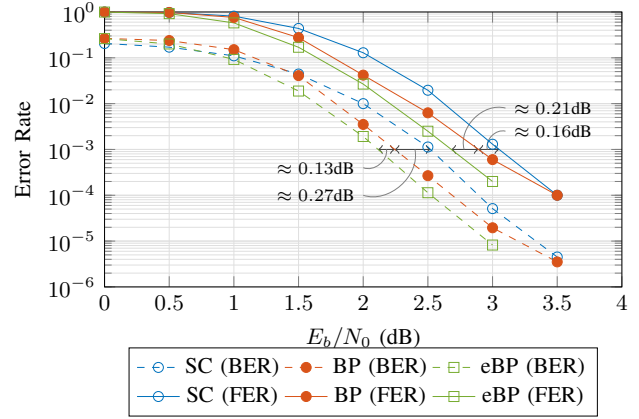


Fig. 3. BER and FER results for a  $\mathcal{P}(1024, 512)$  polar code using the SC, BP, and the proposed (eBP) algorithms. Dashed plots show the BER and solid plots show the FER.

the proposed algorithm performs well using a scaling factor of  $s_1 = 0.984375 = 1 - 2^{-6}$  and  $s_2 = 0.375 = 2^{-2} + 2^{-3}$ . For realization purposes, we will use  $s_2 = 0.25 = 2^{-2}$  instead, because this scaling can be realized with only a shift operation. The parameters in Table I will be used for the comparison of all algorithms in this paper, unless stated otherwise.

The main performance parameter of the proposed algorithm are the BER and FER. As shown in Fig. 3, the proposed algorithm outperforms both the SC and BP algorithms. At an error rate of  $10^{-3}$ , the proposed algorithm has an SNR gain of  $\approx +0.4$ dB and  $\approx +0.13$ dB (FER), and  $\approx +0.37$ dB and  $\approx +0.16$ dB (BER), with respect to the SC and BP respectively.

As explained in Section VI-A we will assume that the latency for all the computations are equal to 1 time unit. We also assume that the highest possible parallelization is used. Given these assumptions, the latency of our decoder is approximately 1.8 to 10.9 times lower compared to the SC algorithm and approximately 0.8 to 1.8 times lower compared to the BP algorithm.

Further simulations show that a reduction of 60%, i.e.  $I_{max} = 24$ , will still provide a decoding performance that is near equal to the BP algorithm (see Fig. 4), but inherently with a major reduction of the latency (see Fig. 5). A further reduction of  $\approx 60\%$  to  $I_{max} = 10$  will provide a decoding performance that is near equal to the SC algorithm. This extra reduction of iterations will reduce the expected latency even more. The latency of the proposed eBP<sub>60</sub> decoder is approximately 1.8 to 11.0 times lower compared to the SC

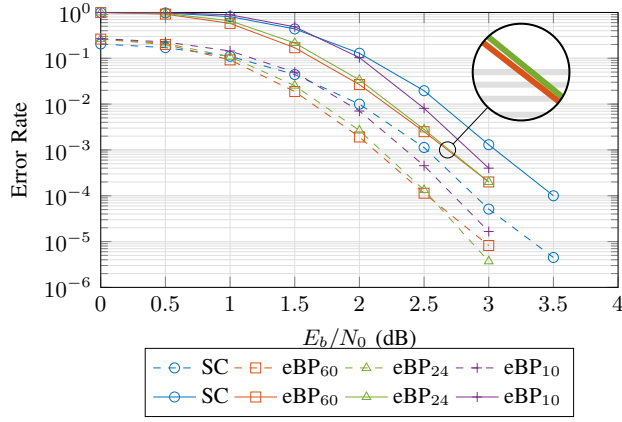


Fig. 4. BER and FER results for a  $\mathcal{P}(1024, 512)$  polar code using the SC, BP, and the proposed (eBP) algorithms, with  $I_{max}$  as a subscript. Dashed plots show the BER and solid plots show the FER.

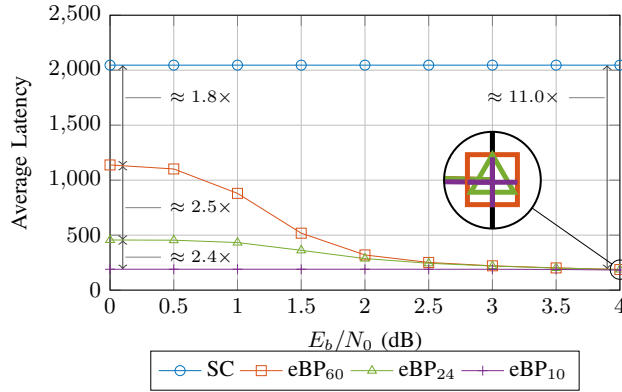


Fig. 5. Latency results for a  $\mathcal{P}(1024, 512)$  polar code using the SC, and the proposed (eBP) algorithms, with  $I_{max}$  as a subscript.

algorithm. The latency of the proposed  $eBP_{24}$  decoder is upto  $\approx 2.5$  times lower compared to the proposed  $eBP_{60}$  algorithm. Finally the latency of the proposed  $eBP_{10}$  decoder is approximately 2.4 to 0.1 times lower compared to the proposed  $eBP_{24}$  algorithm, as shown in Fig. 5.

In Fig. 6 the results are shown for the smaller and larger sized polar codes,  $\mathcal{P}(256, 128)$  and  $\mathcal{P}(2048, 1024)$  respectively. It is clear that the influence of the short cycles become larger as the code size is increased. At an error rate of  $10^{-3}$ , the proposed algorithm has an SNR gain of  $\approx +0.37$ dB and  $\approx +0.17$ dB (FER), and  $\approx +0.35$ dB and  $\approx +0.17$ dB (BER), with respect to the SC and BP respectively, for the polar code of block size  $N = 2048$ . For the smaller sized polar code, the error correcting performance of the BP and the proposed algorithm are nearly equal.

Finally, the error correcting performance and the latency of the proposed algorithm are compared to a simulation of the SCL and FSC algorithms. As shown in Fig. 7, and Fig. 8, the proposed algorithm outperforms the FSC algorithm, but still has a performance gap compared to the SCL algorithm,

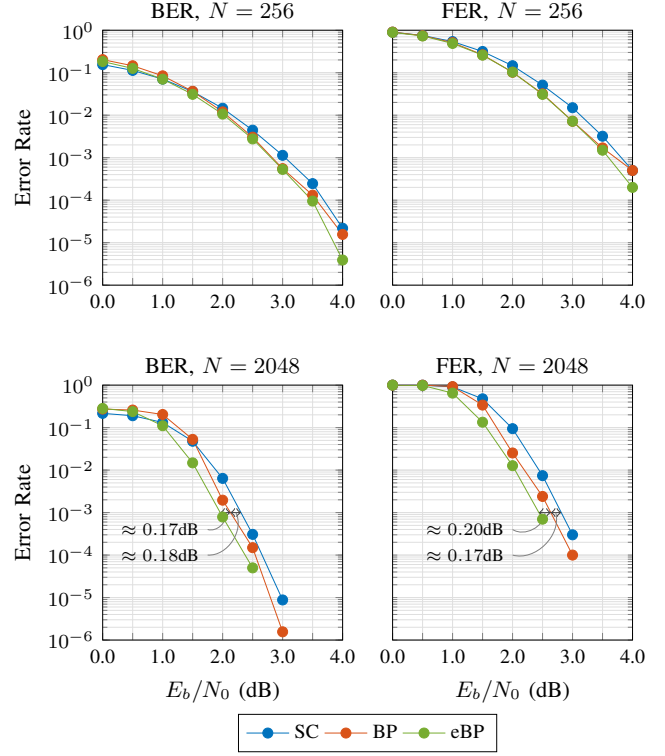


Fig. 6. BER and FER results for different size polar codes ( $\mathcal{P}(256, 128)$ , and  $\mathcal{P}(2048, 1024)$ ) using the proposed (eBP) algorithm, and the original SC, BP algorithms.

in terms of error correcting performance. At an error rate of  $10^{-3}$ , the proposed algorithm has an SNR loss of  $\approx +0.3$ dB and a gain of  $\approx +0.37$ dB (FER), and a loss of  $\approx +0.4$ dB and a gain of  $\approx +0.4$ dB (BER), with respect to the SCL and FSC algorithms respectively.

The latency of the proposed eBP decoder is approximately 1.8 to 11.0 times lower compared to the SCL algorithm, due

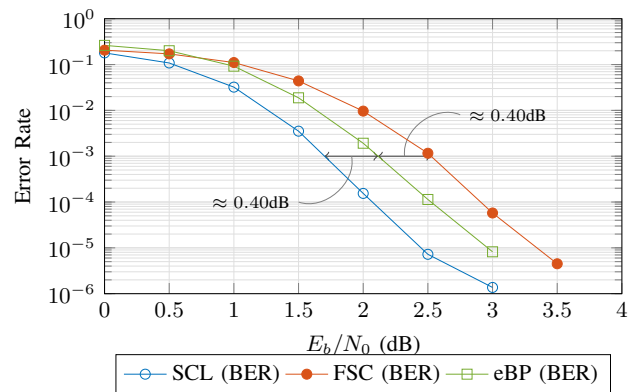


Fig. 7. BER results for a  $\mathcal{P}(1024, 512)$  polar code using the SCL, FSC, and the proposed (eBP) algorithms.

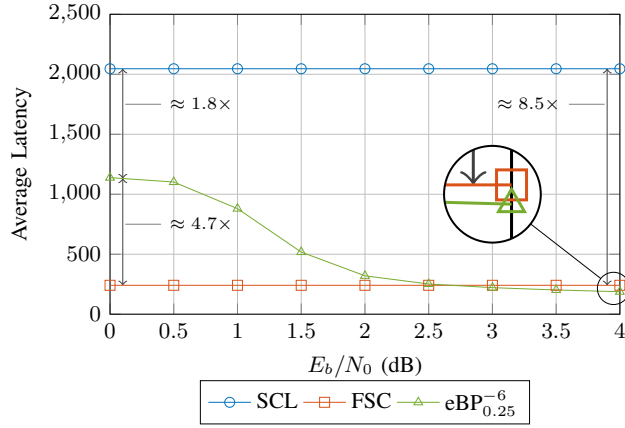


Fig. 9. Latency results for a  $\mathcal{P}(1024, 512)$  polar code using the SCL, FSC, and the proposed (eBP) algorithms, with  $I_{max}$  as a subscript.

to the equal latency of the SCL algorithm compared to the SC algorithm (see Fig. 5). The latency of the proposed eBP decoder is  $\approx 4.7$  upto  $\approx -1.29$  times lower compared to the proposed FSC algorithm, as shown in Fig. 9.

## VII. CONCLUSION

In this paper we have introduced an enhanced belief propagation algorithm for the use in a polar code decoder. The enhanced algorithm uses a different scaling factor, but also scales one of the input parameters in order to reduce the effect of short cycles in the factor graph, which improves the bit error rate and frame error rate compared to both the successive cancellation, fast successive cancellation, and current state-of-the-art belief propagation algorithms. The proposed algorithm also converges faster resulting in 60% less iterations required for the decoding of the received codeword then the belief propagation algorithm. This lower maximum number of iterations results in a lower latency compared to the successive cancellation [1], successive cancellation list, and the belief propagation [6] algorithms. Depending on the current signal to noise ratio, the proposed algorithm is  $\approx 4.7$  times slower upto  $\approx 1.29$  time faster then the state-of-the-art fast successive cancellation algorithm. Finally, a reduction of almost 84% in the maximum number of iterations results in a near identical decoding performance (BER and FER) compared to the SC algorithm, and will subsequently reduce the latency and the required number of operations even further.

## ACKNOWLEDGMENT

This work is part of the research program Perspectief ZERO with project number P15-06 Project 4, which is (partly) financed by the Dutch Research Council (NWO).

## REFERENCES

[1] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

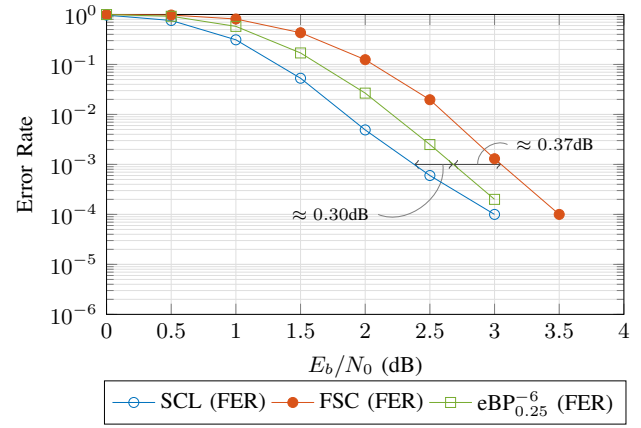


Fig. 8. FER results for a  $\mathcal{P}(1024, 512)$  polar code using the SCL, FSC, and the proposed (eBP) algorithms.

[2] C. Leroux, A. J. Raymond, G. Sarkis, and W. J. Gross, "A Semi-parallel successive-cancellation decoder for polar codes," *IEEE Trans. Signal Process.*, vol. 61, no. 2, pp. 289–299, 2013.

[3] I. Tal and A. Vardy, "List Decoding of Polar Codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, may 2015.

[4] A. Alamdar-Yazdi and F. R. Kschischang, "A simplified successive-cancellation decoder for polar codes," *IEEE Commun. Lett.*, vol. 15, no. 12, pp. 1378–1380, 2011.

[5] G. Sarkis, P. Giard, A. Vardy, C. Thibault, and W. J. Gross, "Fast polar decoders: Algorithm and implementation," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 946–957, 2014.

[6] E. Arıkan, "Polar codes : A pipelined implementation," *4th Int. Symp. Broadband Commun. (ISBC 2010)*, pp. 11–14, 2010.

[7] E. Arıkan, "Systematic polar coding," *IEEE Commun. Lett.*, vol. 15, no. 8, pp. 860–862, 2011.

[8] K. Niu, K. Chen, J. Lin, and Q. T. Zhang, "Polar Codes: Primary Concepts and Practical Decoding Algorithms," *IEEE Commun. Mag.*, vol. 52, no. July, pp. 192–203, 2014.

[9] A. Elkelesh, M. Ebada, S. Cammerer, and S. T. Brink, "Belief Propagation List Decoding of Polar Codes," *IEEE Commun. Lett.*, vol. 22, no. 8, pp. 1536–1539, 2018.

[10] M. Xu, S. Jing, J. Lin, W. Qian, Z. Zhang, X. You, and C. Zhang, "Approximate belief propagation decoder for polar codes," in *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2018-April, no. 3, IEEE, apr 2018, pp. 1169–1173.

[11] C. Albayrak, C. Simsek, and K. Turk, "Low-complexity early termination method for rateless soft decoder," *IEEE Commun. Lett.*, vol. 21, no. 11, pp. 2356–2359, 2017.

[12] Y. Ren, C. Zhang, X. Liu, and X. You, "Efficient early termination schemes for belief-propagation decoding of polar codes," *Proc. - 2015 IEEE 11th Int. Conf. ASIC, ASICON 2015*, 2016.

[13] C. Simsek and K. Turk, "Simplified Early Stopping Criterion for Belief-Propagation Polar Code Decoders," *IEEE Commun. Lett.*, vol. 20, no. 8, pp. 1515–1518, 2016.

[14] S. Sun and Z. Zhang, "Architecture and optimization of high-throughput belief propagation decoding of polar codes," in *Proc. - IEEE Int. Symp. Circuits Syst.*, vol. 2016-July, IEEE, may 2016, pp. 165–168.

[15] A. B. Van Den Brink and M. J. Bekooij, "Conflict-Free Vectorized In-order In-place Radix-r Belief Propagation Polar Code Decoder Algorithm," in *ACM Int. Conf. Proceeding Ser.* New York, NY, USA: ACM, apr 2020, pp. 18–23. [Online]. Available: <https://dl.acm.org/doi/10.1145/3390525.3390539>

[16] J. Sha, J. Liu, J. Lin, and Z. Wang, "A Stage-Combined Belief Propagation Decoder for Polar Codes," *J. Signal Process. Syst.*, vol. 90, no. 5, pp. 687–694, 2018.

[17] Judea Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[18] B. Yuan and K. K. Parhi, "Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders," *IEEE Trans. Signal Process.*, vol. 62, no. 24, pp. 6496–6506, 2014.