Error Patterns in Belief Propagation Decoding of Polar Codes and Their Mitigation Methods

Shuanghong Sun, Sung-Gun Cho, and Zhengya Zhang
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, MI, 48109-2122

***Abstract*—Belief propagation (BP) is a high-throughput decoding algorithm for polar codes, but it is known to underperform successive cancellation (SC) decoding and list decoding in error-correcting performance. In this work, we study the error patterns of BP decoding of polar codes to uncover the error mechanisms, as well as the influence of channel condition, code design, and decoder implementation. Based on the insights, we design new ways to detect, prevent and overcome BP decoding errors.**

I. Introduction

Polar codes are the first provably capacity-achieving error-correcting codes for any binary-input discrete memoryless channels (B-DMC) [1], and the error-correcting capability of polar code holds high promise. The two main decoding algorithms are successive cancellation (SC) [1] and belief propagation (BP) [2]. SC exhibits a better error-correcting performance than BP [3], and list decoding [4], viewed as an enhanced SC, further improves the performance but with an increased complexity. BP on the other hand, provides a higher throughput and a reduced latency, but it sacrifices error-correcting performance.

BP is an iterative message passing algorithm operating on a factor graph. The same BP decoding algorithm has been widely used in decoding low-density parity-check (LDPC) codes. Despite the impressive performance seen in decoding LDPC codes, BP has shown a weakness known as the error floor phenomenon [5]. Error floors occur at moderate to high SNR levels, preventing the waterfall-like improvement in error rate with increasing SNR. In the error floor region, decoding errors are dominated by a small number of fixed patterns known as trapping sets [5].

In this work, we analyze the error patterns in the BP decoding of polar codes. The decoding errors are classified and the factors affecting decoding, including channel SNR, code design, decoding algorithm, and implementation, are analyzed. Based on the insights, we provide preliminary ideas of how the decoding errors can be mitigated.

II. Background

A polar code of block length $N = 2^n$ has a generator matrix G that is the n-th Kronecker power of matrix $F = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right]$, i.e., $G_N = F^{\otimes n}$ [1]. Fig. 1 is the factor graph of an $N = 8$ polar code, where the plus sign represents XOR, and the equal sign represents pass-through. To encode, a message u is placed on the left side of the factor graph, and the codeword $x = uG$ is obtained on the right side. The subchannel seen by each bit

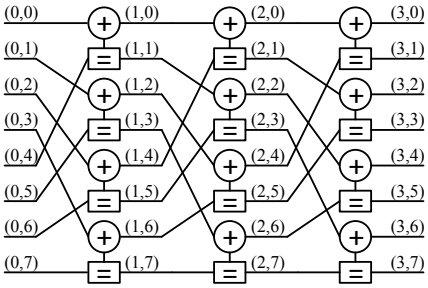

Fig. 1. Factor graph of N=8 polar code.

exhibits a polarization phenomenon when the code length is sufficiently long and decoded using the SC algorithm. Reliable bits are used to transmit information, and the unreliable ones are fixed to known values, normally 0. The frozen set $\mathcal{A}$ denotes the set of unreliable bits [1].

In SC decoding, the message bits $\hat{u_0}$ to $u_{\hat{N-1}}$ are obtained one bit after another based on the channel output y and the previously decoded bits. If $i \in \mathcal{A}$, $\hat{u_i} = 0$; otherwise $\hat{u_i}$ is decoded by the maximum likelihood decision rule:

$$\hat{u_i} = \begin{cases} 0 & \text{if } \frac{P(y,\hat{u}_0^{i-1}|u_i=0)}{P(y,\hat{u}_0^{i-1}|u_i=1)} > 1 \\ 1 & \text{otherwise} \end{cases}$$

where $P(y, \hat{u}_0^{i-1}|u_i = b)$ refers to the probability that the received vector is y and the previously decoded bits being $\hat{u}_0$ through $\hat{u}_{i-1}$, given the current bit being b, where $b \in \{0, 1\}$ [6]. The SC decoding latency is $\mathcal{O}(N)$.

BP decoding operates message passing on the factor graph. Frozen set information is propagated from left to right on the factor graph, and channel output y is propagated from right to left. The estimated message $\hat{u}$ is obtained on the left side. The decoding accuracy can be improved by running more iterations. The BP decoding latency is $\mathcal{O}(\log N)$.

III. Error Classification

To understand how decoding fails, we obtain hard decisions $\hat{u}$ at the end of each BP decoding iteration. Due to the lack of a definitive convergence check in polar decoding, we use hard decisions from consecutive iterations to decide whether decoding has converged. If hard decisions over consecutive iterations agree, we consider it has converged.

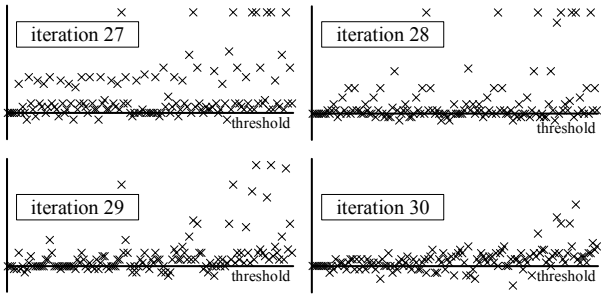

Fig. 2. Soft decisions of an unconverged error in BP decoding of a (256, 128) code.

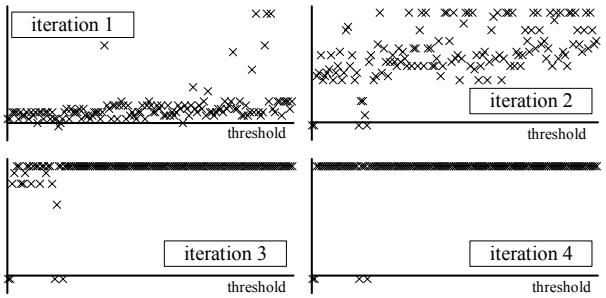

Fig. 3. Soft decisions of a converged error in BP decoding of a (256, 128) code.

A. Unconverged Error

If hard decisions fail to agree within a maximum allowed iteration limit, and there is no defined pattern of error, we call it an unconverged error. Unconverged errors are most common at a low SNR level where the channel is noisy and the decoder is unable to resolve the errors.

Assume an all-zero codeword is transmitted using binary phase-shift keying (BPSK) modulation; and a bit u_i is decoded correctly if the soft decision of $\hat{u}_i > 0$ and incorrectly otherwise. A plot of the soft decisions of $\hat{u}$ is shown in Fig. 2, illustrating an unconverged error for a (256, 128) code. The decision threshold is 0. The soft decisions keep flipping across the decision threshold, and incorrect soft decisions hover around the decision threshold. There is no obvious pattern in the decisions, and more iterations do not help to find correct convergence.

B. Converged Error

As SNR increases, unconverged errors start to disappear, and errors of systematic patterns start to emerge. The majority of systematic error patterns we found are attributed to converging to wrong codewords, or falling to local minima of BP decoding operating on loopy factor graphs. The factor graphs of polar codes contain loops, so a flooding BP decoder is not immune to local minima problems.

If hard decisions are stable and agree over consecutive iterations, but the decoded message is incorrect, i.e., $\hat{u} \neq u$, we call it a converged error. Converged errors are most common at moderate to high SNR, and it usually takes only a small number of iterations to reach a steady state, as illustrated in Fig. 3 for a converged error in the decoding of a (256, 128) code. In this example, within two or three iterations, the soft decisions of a small number of bits are found to be trapped in wrong decisions, and they cannot be recovered using more iterations.

The particular case illustrated in Fig. 3 represents a local minimum state in BP decoding. The few incorrect bits reinforce the wrong decisions among themselves through loops in the factor graph, making it impossible to make any progress towards convergence, which is similar to a trapping set found in the BP decoding of LDPC codes.

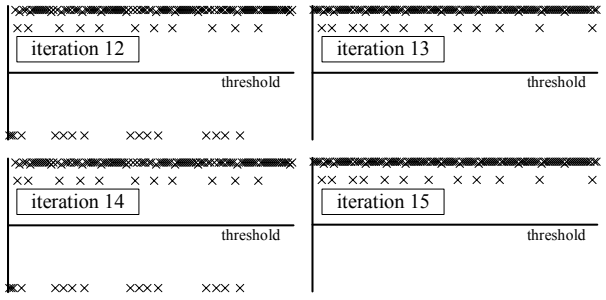

Fig. 4. Soft decisions of an oscillation error in BP decoding of a (256, 128) code.

C. Oscillation Error

Loops in the factor graph allow the propagation of incorrect messages through BP decoding, causing oscillations. As incorrect messages travel around a loop, the decisions also go through a round of changes.

If hard decisions are unstable and change periodically over iterations, we call it an oscillation error. Although an oscillation error is also an unconverged error, an oscillation error features a pronounced pattern of periodic changes. An example of the oscillation error is shown in Fig. 4. The illustrated error has an oscillation period of 2 iterations. A group of bits are incorrect in iteration 12; the incorrect bits all turn correct in iteration 13, but they turn incorrect again in iteration 14. If decoding is terminated in iteration 13, decoding would be done correctly. However, there is no way for the decoder to decide when to terminate in the absence of a definitive convergence detector in polar codes. Relying on checking hard decisions over consecutive iterations does not help terminate an oscillation error in the right iteration.

D. Error Distribution

In Fig. 5, we show the statistical breakdown of errors at each SNR point for the BP decoding of a (256, 128) code. At a low SNR level, unconverged errors dominate; as SNR increases, converged errors and oscillation errors become dominant. The error breakdown demonstrates the importance of fixing the loopy behavior of BP decoding and of designing polar codes with a large minimum distance to improve the error-correcting performance.

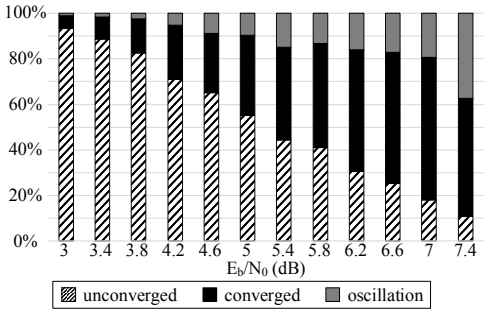

Fig. 5. Error distribution for BP decoding of a (256, 128) code.

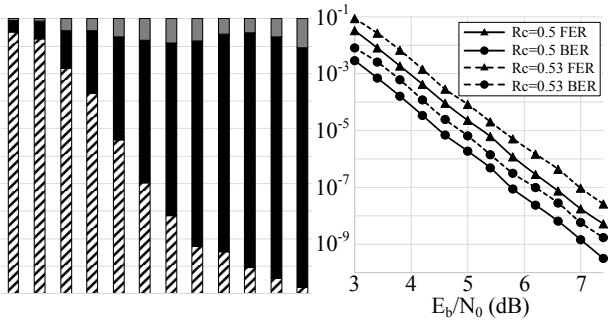

Fig. 6. Error distribution and error rate for BP decoding of a (256, 136) code.

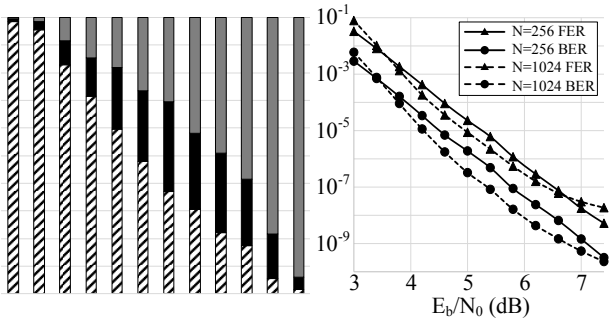

Fig. 7. Error distribution and error rate for BP decoding of a (1024, 512) code.

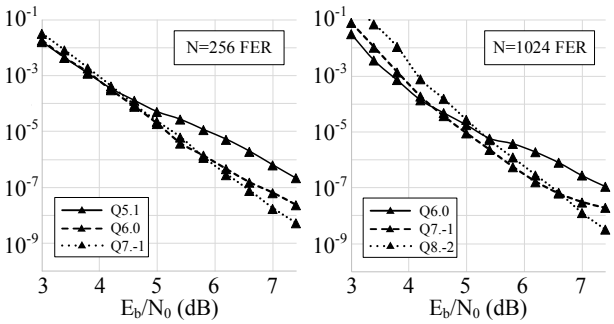

Fig. 8. Error rate for BP decoding of a (256, 128) code and a (1024, 512) code that are implemented in different quantization schemes.

IV. Factors Affecting Decoding

To gain an insight into decoding errors, we adjust code and decoder design parameters and analyze the corresponding changes in error rate and error breakdown.

A. Code Design

Take the rate-0.5 (256, 128) code in Fig. 5 as reference. We increase the rate of the 256-bit code from 0.5 to 0.53 and plot the error distribution and error rate of the rate-0.53 code in Fig. 6. The axes and markers of the distribution in Fig. 6 are identical to those in Fig. 5, so they are omitted in Fig. 6 for simplicity. All the later plots follow the same convention, unless they are explicitly marked.

The error rate of a higher rate code is worse as expected. The number of converged errors at a high SNR level is noticeably higher. More converged errors can be explained by more free bits in a higher rate code resulting in more codewords, or a more crowded codeword space, making it more likely to converge to a wrong codeword.

As we increase the block length from 256 to 1024 while keeping the code rate of 0.5, the error distribution and error rate of the (1024, 512) code are shown in Fig. 7. The error rate of the (1024, 512) code improves over the (256, 128) code, but it suffers from an error floor at FER below 10^{-7}. Compared to the (256, 128) code, the (1024, 512) code has fewer converged errors but more oscillation errors. The (1024, 512) code has a larger minimum distance and a sparser codeword space, so it is expected to outperform the (256,128) code. However, with two more stages in the factor graph, the factor graph of the (1024, 512) code contains more loop configurations than the (256, 128) code, resulting in more oscillation errors.

We also note that with more processing stages in a larger factor graph, numerical saturation occurs more easily. When reliable bits are saturated, they are less pronounced and cannot effectively prevent incorrect bits from propagating. Allocating more bits to cover a larger numerical range is expected to alleviate the problem.

B. Decoder Implementation

The choice of fixed-point quantization affects the decoding performance [7]. In the above simulations, the Q7.-1 (6 bits covering the range of -64 to 62 with a resolution of 2) fixed-point quantization was used. Keeping the same 6-bit word length, the Q6.0 quantization covers the range of -32 to 31 with a resolution of 1; and the Q5.1 quantization covers the range of -16 to 15.5 with a resolution of 0.5. The quantizations used in Fig. 8, Q5.1, Q6.0, Q7.-1, Q8.-2, all share the same 6-bit wordlength, but they result in different error-correcting performance.

At a low SNR level, a quantization with a finer resolution improves numerical accuracy; and at a high SNR level, a quantization with a larger range prevents clipping and yields better performance. The comparison between the two codes in Fig. 8 shows that a longer code requires a higher range to obtain the expected performance, and a larger range also alleviates the error floor problem.

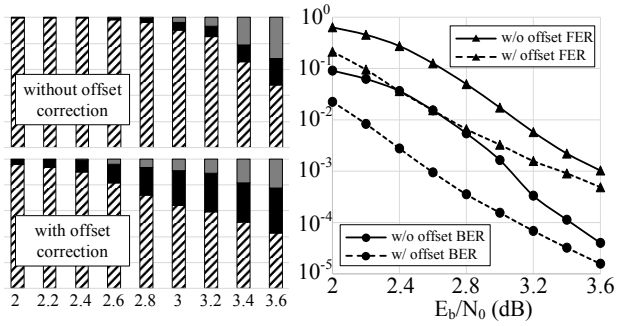

Fig. 9. Error distribution and error rate for BP decoding of a (1024, 512) code at low SNR with and without offset correction.

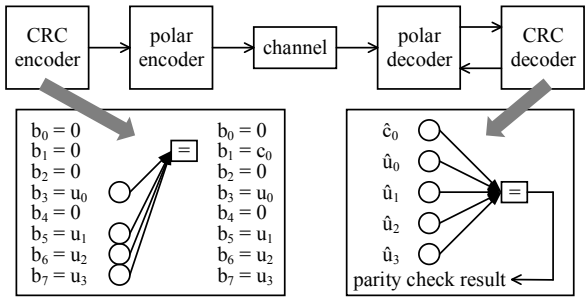

Fig. 10. Illustration of concatenation of polar code with CRC.

Min-sum approximation is often applied to simplifying the log-likelihood ratio calculations. The associated min-sum approximation error can be compensated by offset correction. Offset correction is especially effective at a low SNR level, as illustrated in Fig. 9. The number of unconverged errors is reduced, as offset correction reduces approximation errors and improves the decoding performance.

V. Error Detection

As discussed above, a BP polar decoder is unaware of whether decoding has converged. An iteration-by-iteration hard decision check detects unconverged errors, but it fails to detect converged errors, which account for 20% to 90% of the errors. An iteration-by-iteration hard decision check detects oscillation errors, but it fails to find the right iteration to terminate decoding and stop oscillations. Therefore, we add a low-cost error detection scheme to catch the majority of the undetected errors.

The error detection scheme is based on concatenating polar code with cyclic redundancy check (CRC) that consumes only a small number of parity bits but provides a good detection capability. A CRC codec is added outside the polar codec, illustrated in Fig. 10. The CRC encoder generates parity bits for an input message. The message bits along with the parity bits are remapped to the free bits of the polar code. On the decoder side, the CRC decoder checks if the hard decisions obtained by the polar decoder is a valid CRC codeword at the end of each decoding iteration.

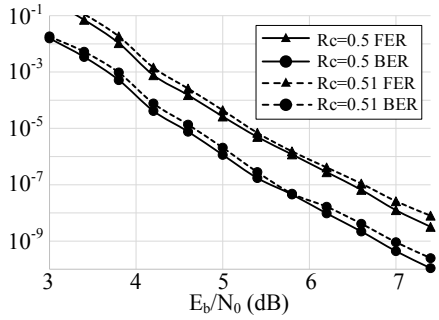

Fig. 11. Error rate for BP decoding of a (1024, 512) code before and after CRC concatenation.

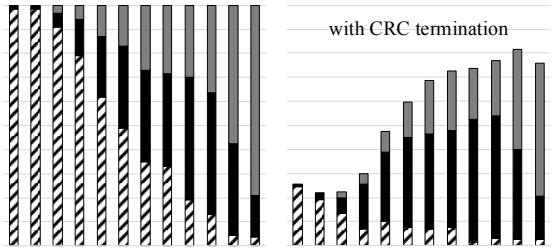

Fig. 12. Error distribution for BP decoding of a (1024, 522) code without and with CRC-based termination.

A CRC-n code generates n parity bits, which covers up to $2^n - 1 - n$ message bits. We employed CRC-8 in the rate-0.5 256-bit polar code and CRC-10 in the rate-0.5 1024-bit polar code. Our simulation shows that more than 99% of the previously undetected errors are detected by CRC. Note that CRC concatenation increases the code rate. If the overall code rate is kept the same, CRC concatenation results in a slight performance loss.

In the 256-bit code and the 1024-bit code, CRC concatenation increases the code rates from 0.5 to 0.53 and 0.5 to 0.51, respectively. The performances are compared in Fig. 6 and Fig. 11. The coding gain of the 256-bit code is reduced by approximately 0.3 to 0.5 dB, but the loss is much smaller in the 1024-bit code due to the neglible number of parity bits relative to the block length.

VI. Error Mitigation

In this study, we make use of CRC to reliably determine when to terminate decoding, i.e., if CRC passes, the iterative decoding is terminated. The effect of CRC-based termination in the BP decoding of a (1024, 522) polar code is shown in Fig. 12, where the white space above the bars represents the percentage of errors being resolved with the proper termination based on CRC. We observe that BP decoding is able to produce error-free messages in some iterations even if the decoding itself is not stable, as in the cases of unconverged errors and oscillation errors. The CRC-based termination helps to lock in the correct codeword before decoding diverges.

CRC-based termination helps resolve most of the unconverged errors and a portion of oscillation errors. The remain-

ing errors are dominated by converged errors and oscillation errors. One approach to fix the remaining errors is via post-processing by perturbation. Prior work in LDPC post-processing points out that perturbation is especially beneficial when a BP decoder is trapped in a local minimum [8]. From a cost standpoint, post-processing can be implemented as part of BP decoding with biased messages. We are in the process of completing a full experimentation and analysis of post-processing methods.

VII. Conclusion

In this work, we classify BP decoding errors into three categories: unconverged errors, converged errors, and oscillation errors. We analyze the important factors affecting decoding, including code design and decoder implementation. While unconverged errors and oscillation errors are detectable by checking the hard decisions in each iteration for consistency, converged errors are undetected. Concatenation of polar codes with CRC enables the detection of undetected errors and the proper termination of BP decoding. CRC-based termination helps to remove a large portion of unconverged errors and oscillation errors.

References

[1] E. Arikan, "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, July 2009.

[2] A. Pamuk, "An FPGA implementation architecture for decoding of polar codes," in *Int. Symp. Wireless Commun. Syst.*, Nov 2011, pp. 437–441.

[3] B. Yuan and K. K. Parhi, "Architecture optimizations for BP polar decoders," in *IEEE Int. Conf. Acoustics, Speech and Signal Process.*, May 2013, pp. 2654–2658.

[4] I. Tal and A. Vardy, "List decoding of polar codes," in *IEEE Int. Symp. Inf. Theory*, July 2011, pp. 1–5.

[5] T. Richardson, "Error floors of LDPC codes," in *Proc. Annu. Allerton Conf. Commun. Control and Computing*, vol. 41, no. 3, 2003, pp. 1426–1435.

[6] 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, Jan 2013.

[7] S. Sun and Z. Zhang, "Architecture and optimization of high-throughput belief propagation decoding of polar codes," in *2016 IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2016, pp. 165–168.

[8] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. J. Wainwright, "Lowering LDPC error floors by postprocessing," in *2008 IEEE Global Telecommunications Conf.*, Nov 2008, pp. 165–168.