On Bit Error Rate Performance of Polar Codes in Finite Regime

A. Eslami and H. Pishro-Nik

Abstract*— Polar codes have been recently proposed as the first low complexity class of codes that can provably achieve the capacity of symmetric binary-input memoryless channels. Here, we study the bit error rate performance of finite-length polar codes under Belief Propagation (BP) decoding. We analyze the stopping sets of polar codes and the size of the minimal stopping set, called "stopping distance". Stopping sets, as they contribute to the decoding failure, play an important role in bit error rate and error floor performance of the code. We show that the stopping distance for binary polar codes, if carefully designed, grows as $O(\sqrt{N})$ where N is the code-length. We provide bit error rate (BER) simulations for polar codes over binary erasure and gaussian channels, showing no sign of error floor down to the BERs of 10^{-11}. Our simulations asserts that while finite-length polar codes do not perform as good as LDPC codes in terms of bit error rate, they show superior error floor performance. Motivated by good error floor performance, we introduce a modified version of BP decoding employing a guessing algorithm to improve the BER performance of polar codes. Our simulations for this guessing algorithm show two orders of magnitude improvement over simple BP decoding for the binary erasure channel (BEC), and up to 0.3 dB improvement for the gaussian channel at BERs of 10^{-6}.

***Index Terms*— Polar Codes, Belief Propagation, Error Floor, Binary Erasure Channel.**

I. Introduction

Polar codes, recently proposed by Arikan are the first provably capacity achieving class of codes for symmetric binary-input discrete memoryless channels (BDMC) with low encoding and decoding complexity [1]. Since Arikan's paper, there have been many papers studying the performance characteristics of polar codes as well as various potential applications of these codes [2]–[17]. However, polar codes show some drawbacks when compared against well-known modern coding techniques such as LDPC codes. For instance, polar codes hold an encoding and decoding complexity of $O(N \log N)$ compared to linear complexity of LDPC codes. Furthermore, polar codes show worse BER performance than LDPC codes for finite lengths [3], [4]. Thus, an interesting problem is to find better approaches to encoding and decoding. In this direction, [3] and [4] showed that the BER can be improved by using Belief Propagation (BP), instead of Successive Cancelation (SC), to decode polar codes. Along the same line, another interesting issue is to study the structure and minimum size of the stopping sets as one of the most common causes in the failure of BP decoding. Due to the significant contribution of stopping sets to the bit error rate and error floor, such an analysis can reveal many advantages or disadvantages of polar codes over other coding schemes. In this paper, we find the structure of minimal stopping set and show that its size can grow as large as $O(\sqrt{N})$ which can be considered as an advantage over the LDPC codes. The effect of such a large stopping distance on the error floor behavior of polar codes is depicted in our simulation results where no sign of error floor can be seen down to the BERs of 10^{-11} for the binary erasure and gaussian channels.

In addition to the above, since BP is rather well-studied in the context of LDPC codes, there are many approaches to modify BP in order to obtain better BER performance; examining such schemes in the context of polar codes is another interesting issue. Therefore, we consider a modified version of BP employing a guessing algorithm in the second part of the paper. The algorithm was studied in [18] and [19] and was shown to be considerably helpful in the case of LDPC codes with good error floor performance. Here we show that by applying this algorithm to the polar codes we can achieve significant improvements in BER. The rest of the paper is organized as follows. We first explain the notations and provide a short background on the belief propagation. In Section III, we study the error floor performance of polar codes, where we first provide some analysis on the minimal stopping set of the code. We will then study the improvement made in BER by employing a modified version of BP using guessing techniques.

II. Belief Propagation Decoding for Polar Codes

In this section, we explain some notations and preliminary concepts we will be using in our analysis. The construction of polar codes is based on the following observation: Let $F = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right]$. Apply the transform $F^{\otimes n}$ (where $\otimes n$ denotes the nth Kronecker power) to a block of $N = 2^n$ bits and transmit the output through independent copies of a symmetric BDMC, call it W. As n grows large, the channels seen by individual bits (suitably defined in [1]) start polarizing: they approach either a noiseless channel or a pure-noise channel, where the fraction of channels becoming noiseless is close to the capacity $I(W)$. The channel polarization phenomenon suggests to use the noiseless channels for transmitting information while fixing the symbols transmitted through the noisy ones to a value known both to the sender as well as the receiver. For symmetric channels we can assume without loss of generality that the fixed positions are set to 0. Since the fraction of channels becoming noiseless tends to $I(W)$, this scheme achieves the capacity of the channel. In the following, let $\bar{u}$ denote the

This work was supported by the National Science Foundation under grants CCF-0830614 and ECCS-0636569.

The authors are with the Electrical and Computer Engineering Department, University of Massachusetts, Amherst, MA, USA (email:eslami, pishro@ecs.umass.edu).

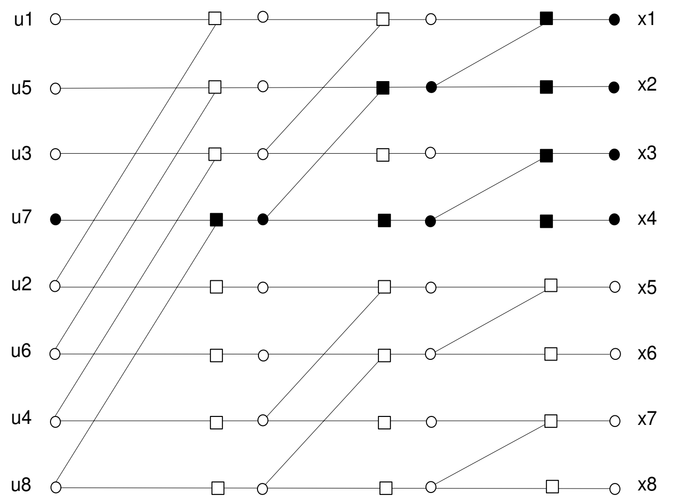


Fig. 1. Normal realization of the encoding graph for $N = 8$. An example of the stopping set is shown with white variable and check nodes.

vector $(u_1, ..., u_N)$ of input bits and $\bar{x}$ denote the vector $(x_1, ..., x_N)$ of coded bits. We also represent the channel output vector by $\bar{y} = (y_1, ..., y_N)$.

The code is defined through its generator matrix as follows. Compute the Kronecker product $F^{\otimes n}$. This gives a $2^n \times 2^n$ matrix. The generator matrix of polar codes is a sub-matrix of $F^{\otimes n}$ in which only a fraction of rows of $F^{\otimes n}$ are present. An equivalent way of expressing this is to say that the codewords are of the form $\bar{x} = \bar{u}F^{\otimes n}$, where the components u_i of $\bar{u}$ corresponding to the "frozen bits" are fixed to 0 and the remaining components contain the "information". Information bits, according to [1], are chosen to be the ones corresponding to the best (least noisy) polarized channels.

A Successive Cancelation (SC) decoding scheme is employed in [1] to prove the capacity-achieving property of the code. However, [3] and [4] later proposed using belief propagation decoding to obtain better BER performance while keeping the decoding complexity at $O(N \log N)$. Belief propagation can be run on the normal representation of the encoding graph [3]. Such a representation is easily obtained by adding check nodes to the encoding graph of polar codes, as it is shown in Fig. 1 for a code of length 8. We sometimes refer to this graph as the "tanner graph" of the code. Note that the graph is formed of columns of variable nodes and check nodes. In the tanner graph, each variable node v has a set of check nodes as its neighbors, denoted by $\mathcal{C}_v$. The check nodes in $\mathcal{C}_v$ also have a set of variable nodes $\mathcal{V}_v$ as their neighbors from which we exclude v. We refer to $\mathcal{V}_v$ as "depth-2 neighborhood" of v and we call the members of $\mathcal{V}_v$ "depth-2 neighbors" of v.

In this paper, we are particularly interested in the analysis of "stopping sets" in the tanner graph of the polar code, as an important cause of the decoding failure and error floor [20]. A stopping set is a set of variable nodes such that every neighboring check node of the set is connected to at least two variable nodes in the set. Fig. 1 shows an example of the stopping set in the polar codes'graph. It is easy to see that if all the variable nodes in a stopping set are erased, then non of them can be decoded in belief propagation. On the other hand, if any of the variable nodes in a stopping set is decoded then all of them can be decoded [20]. As a result, the probability of decoding error is closely related to the probability of having a stopping set with erased variable nodes. Since the stopping set with the minimum number of variable nodes, called "minimal stopping set", is more probable to be erased than larger stopping sets, it plays the dominant role in the error probability. Therefore, in code design, codes with large minimal stopping sets are desired. Here, we study the minimal stopping set of polar codes and its size, called "stopping distance".

III. Stopping Set Analysis of Polar Codes

We first consider the problem of minimal stopping set given the set of information bits. The results of this analysis may also help finding the optimal rule of choosing information bits (rows of the generator matrix) to minimize the BER in belief propagation. One of the eminent stopping sets are the trees rooted at the information bits (on the left side of graph) with leaves at the code bits (on the right side of graph). An example of such stopping sets is shown in Fig. 1 with black variable and check nodes. We refer to such a tree as the "stopping tree" of an information bit. A stopping tree with the minimum number of variable nodes on it is called a minimal stopping tree. We refer to the set of leaf nodes of a stopping tree as the "leaf set" of the tree. We now find the relation between the minimal stopping tree and the minimal stopping set. First we state the following lemma.

Lemma 1: Any stopping set in the tanner graph of a polar code includes variable nodes and check nodes from all columns of the graph. In particular, any stoping set includes at least one information bit and one code bit.

Proof: First, note that we only have degree 2 and 3 check nodes in the graph. Also note that the graph is formed of Z-shaped parts, as it can be seen in Fig. 1. In every Z-shaped part there are two check nodes, one at the top and one at the bottom. The top check node is always of degree 3 and the bottom one is always of degree 2. We show that if a stopping set includes either of these check nodes in a column of the graph, then it must include check nodes and variable nodes from both right side and left side columns of that specific column. Therefore, any stopping set includes variable nodes and check nodes from all the columns of the graph including both ends.

We consider two cases. If the bottom check node is in the stopping set, then both of its neighboring variable nodes must be in the set since each check node in the set is connected to at least two variable nodes. Since all the check nodes connected to a variable node in the stopping set are also in the set, this means that some of the check nodes in the left and right columns of the considered Z-shaped part are in the set as well. In the second case, if the upper check node (of degree 3) is in the stopping set, then its neighbors in the stopping set are either a variable node at its right and one at its left, or two variable nodes at its left, one at the top and one at the bottom of the Z. In the former case, the set clearly includes nodes from the left and right columns. In the later case, the bottom variable node has the bottom check node as its neighbor in the set, leading to the same situation as we discussed above. ■

A key observation in the tanner graph of a polar code of length 2^{m+1} is that it is formed of two tanner graphs for $N = 2^m$ as its upper and lower halves, connected together via an additional column of check and variable nodes in the left. The following lemma employs this observation to find the structure of the stopping sets.

Lemma 2: Any stopping set in the tanner graph of a polar code of length 2^{m+1} is formed of at least one stopping set in the upper or lower half subgraphs of $N = 2^m$, and some variable and check nodes in the leftmost column.

Proof: Lemma 1 implies that any stopping set for $N = 2^{m+1}$ includes at least one information bit. Assume such a stopping set and remove the information bits and their neighboring check nodes from the graph. This leaves us with two disconnected sets of nodes in the upper and lower subgraphs, each of them corresponding to a code of length $N = 2^m$. We denote these two sets by S^{up} and S^{low}. However, the removed information nodes in the left were connected to the rest of the graph only through the set of check nodes in the leftmost column, which are now removed. Thus, if the variable nodes and check nodes in S^{up} and S^{low} were satisfying the conditions of a stopping set before, they are still satisfying those conditions. Therefore, for any stopping set in the graph of $N = 2^{m+1}$, the induced subsets in the upper and lower halves are also stopping sets for the graph of $N = 2^m$. ■

Theorem 1: The minimal stopping tree in the tanner graph of a polar code is a minimal stopping set.

Proof: We provide a constructive proof in which we inductively find the minimal stopping set for a code of length 2^{m+1} based on the minimal stoping set of a code of length 2^m. First consider the tanner graph for $N = 2$. The minimal stopping set for this code is clearly a stopping tree. The minimal stopping set for $N = 4$ can also be obtained easily as a stopping tree.

Now we use the induction. Assuming that given the information bits the minimal stopping set for $N = 2^m$ is a stopping tree, we show that the same argument is true for $N = 2^{m+1}$. Based on Lemma 1, we know that the minimal stopping set includes at least one information bit. In our graph, we will probably have information bits in both halves. Note that information bits in the upper and lower halves are connected to one and two check nodes, respectively. Since each check node has only one variable node in its right neighborhood, every information bit in the upper and lower halves has, respectively, one or two variable nodes in its neighborhood of depth 2. Therefore, for every information bit in the upper half we can find a stopping tree formed of the information bit itself, its neighboring check node, and the stopping tree for its depth-2 neighboring variable node. Among all such stopping trees, we denote the one with minimum number of variable nodes by T^{up}_{min}. For the lower half information bits, we can do the same thing, this time by including both depth-2 neighboring variable nodes and their stopping trees. Among all such stopping trees, we denote the one with minimum number of variable nodes by T^{low}_{min}.

We find the smaller tree between T^{up}_{min} and T^{low}_{min} and call it T_{min}. We show that T_{min} is the minimal stopping set for $N = 2^{m+1}$. First note that the minimality of our choice for the stopping trees of depth-2 variable nodes is justified by Lemma 2 and the induction hypothesis. Now, assume that T_{min} is not minimal and there is another subgraph T'_{min} smaller than T_{min}. T'_{min} includes either one or more than one information bits. In the former case, clearly T'_{min} cannot have better variable nodes than T_{min} in the second column; here from better we mean variable nodes with smaller stopping trees. In the case of having more than one information bits, for T'_{min} to be smaller, it needs to have fewer variable nodes than T_{min} in the rest of the graph, i.e. from the second column to the rightmost column. This cannot happen because T'_{min} includes either less or more than two variable nodes in the second column. In the former case, those variable nodes cannot lead to smaller stopping sets in their half compared to our choice in T_{min}. In the later case, those variables nodes lead to more than one stopping tree or non-tree stopping sets in their half, which is not optimal based on the induction hypothesis. Thus, T_{min} is the minimal stopping set. ■

Now, we need to find the information bit with the minimal corresponding tree. Below, we show that the size of the stopping tree for any information bit can be obtained using the corresponding row in the generator matrix. We denote the number of variable nodes in the stopping tree of an information bit u by $N_{var}(u)$. We also denote the row in the generator matrix assigned to an information bit u by $\bar{c_u} = [c^u_1\ c^u_2\ \dots\ c^u_N]$. Note that we have $\log N + 1$ columns of variable nodes in any stopping tree. Let us show the number of variable nodes in column j of such a tree by v^u_j. Then, based on the structure of the tanner graph and its relation to the generator matrix, it can be verified that

$$v^u_j = \sum_{\{i=1,\dots,N \mid \mathrm{mod}(i-1,\frac{N}{2^j})=0\}} c^u_i, \tag{1}$$

where $\mod(i-1, \frac{N}{2^j})$ is the reminder of the division of $i-1$ by $\frac{N}{2^j}$.

Theorem 2: The size of the minimal stopping set is

$$\min_{\text{all information bits}} N_{var}(u) = \min_{\text{all information bits}} \sum_{j=0}^{n} \sum_{\{i=1,\dots,N \mid \mathrm{mod}(i-1,\frac{N}{2^j})=0\}} c^u_i. \tag{2}$$

Proof: Using (1), the number of variable nodes in the corresponding tree of any information bit u can be obtained from the corresponding row in the generator matrix as

$$N_{var}(u) = \sum_{j=0}^{n} \sum_{i:\mathrm{mod}(i-1,\frac{N}{2^j})=0} c^u_i. \tag{3}$$

The minimal stopping set is the tree corresponding to the information bit with minimum $N_{var}(u)$ as in (3). ■

We now show the relation between the stopping distance and the minimum distance of polar codes.

Theorem 3: The stopping distance of a polar code defined on a normal realization graph such as the one in Fig. 1 is at least equal to the minimum distance of the code, d_{min}.

Proof: The size of the leaf set for any stopping tree is in fact equal to the weight of the corresponding row in the generator matrix. According to [4], the minimum distance of the code is equal to the minimum weight of the rows in the generator matrix. Thus, the size of the minimal leaf set among all the stopping trees is equal to d_{min}. This implies that the size of leaf set in the minimal stopping tree is at least equal to d_{min}. Since such a tree has some other variable nodes in addition to the ones in its leaf set, the stopping distance will be at least as large as the minimum distance of code. In fact, since the number of variable nodes in each column of the tree either remains the same or gets doubled in the next column, the number of variable nodes in the minimal stopping tree is at least $d_{min} + \frac{d_{min}}{2} + ... + \frac{d_{min}}{2^{\lfloor \log_2 \sqrt{N} \rfloor}} = \Omega(d_{min})$. ■

Based on the Theorem 3, the number of erased code bits in the minimal stopping set grows asymptotically as fast as the minimum distance. Although the code graph here includes $N \log N$ variable nodes instead of N variable nodes, Theorem 3 can be interpreted as a positive result, particularly compared to the capacity approaching LDPC codes for which the stopping distance is fairly small in comparison to the block length [21]–[23].

The above analysis was based on the assumption of knowing the set of information bits. However, it is not always easy to find the information set in polar codes especially with large code lengths. Thus, we are interested in finding the stopping distance regardless of a specific choice of the information set. The following theorem asserts that with a slight modification of a polar code in hand, we can obtain a new code with a stopping distance larger than $\sqrt{N}$, regardless of the selection of information set.

Theorem 4: In the tanner graph of a polar code of length N, the number of input bits u_i for which $N_{var}(u_i)$ is less than N^{ϵ}, $0 < \epsilon < \frac{1}{2}$, is less than $N^{H(\epsilon)}$. Thus, they make a vanishing fraction of total input bits as $N \rightarrow \infty$.

Proof: In the matrix $F^{\otimes n}$, there are $\binom{n}{i}$ rows with weight 2^i [4]. This means that in the tanner graph of a polar code, there are $\binom{n}{i}$ stopping trees with a leaf set of size 2^i. Thus the corresponding tree of these input bits is at least of size 2^i. As a result, the number of input bits with less than $2^{\epsilon n} = N^{\epsilon}$ variable nodes in their tree is less than $\sum_{i=0}^{\epsilon n} \binom{n}{i}$, which is itself upper-bounded by $2^{H(\epsilon)n} = N^{H(\epsilon)}$ for $0 < \epsilon < \frac{1}{2}$. Since $0 < \epsilon < \frac{1}{2}$, we have $H(\epsilon) < 1$ and $\frac{N^{H(\epsilon)}}{N}$ would be vanishing as $N \rightarrow \infty$. ■

The above theorem implies that regardless of the choice of information bits, we can always replace a vanishing fraction of information bits with some fixed bits that have larger trees, and obtain a code with a stopping distance larger than $\sqrt{N}$. The next theorem asserts that such a replacement does not effectively change the BER performance of polar codes under BP.

Theorem 5: In the tanner graph of a polar code, if we replace a vanishing number of information bits by some fixed bits, the overall error probability under BP does not change asymptotically.

Proof: First note that the tanner graph of polar codes is a sparse graph where the maximum degree of a variable node is 3. Suppose that we replace a fraction $\alpha(N)$ of the information bits with fixed bits, where $\alpha(N)$ is vanishing by N, i.e. $\alpha(N) = o(N)$. We first find out how many of the remaining information bits will be affected by this replacement. We consider the expanded version of the polar code's graph up to a neighborhood of depth l, where l is actually the number of iterations in BP and is finite. The number of information bits in this neighborhood is at most $3^l \alpha(N)$ which is itself vanishing with N. The overall BER is the average of BER over the affected and unaffected information bits as follows

$$\begin{aligned} P_e &= \frac{\sum_{\text{affected info bits}} P_e(u_i) + \sum_{\text{unaffected info bits}} P_e(u_i)}{R \times N} \\ &= \frac{o(N)P_e^a + [RN - o(N)]P_e^u}{R \times N} \\ &= o(1)P_e^a + [1 - o(1)]P_e^u \rightarrow P_e^u, \end{aligned}$$

where P_e^a and P_e^u are, respectively, the average BER of affected and unaffected information bits. ■

The above theorem along with Theorem 4 implies that if we replace the unwanted information bits in the information set (which are of a vanishing number with N) by information bits with larger trees, we can expect to still achieve about the same BER performance. This is not generally true for SC where replacing only a few bits can deteriorate the BER in the finite lengths. Fig. 2 shows the effect of such replacement in both SC and BP. The figure presents the simulation result for a code of rate $\frac{1}{2}$ and length 2^{13}, for which 2^9 information bits are randomly selected and replaced by randomly selected fixed bits. However, for a fair comparison, the same set of information and fixed bits are chosen for both decoding schemes. As it can be seen, BP is less sensitive to the exact choice of the information bits which can be actually difficult for polar codes in the case of BSC, gaussian, and many other practical channels.

A. Simulation Results for Error Floor

Here, we show the simulation results obtained for error floor performance of polar codes over binary erasure and gaussian channels. Fig. 3(a) shows the simulation results for a code of length 2^{15} and rate 1/2 over BEC. As it can be seen, no error floor emerges down to BERs of 10^{-11}. This is consistent with the relatively large stopping distance of polar codes which we conjecture to grow as $O(\sqrt{N})$. Fig. 3(b) also shows the simulation results for a rate $\frac{1}{2}$ polar code of length 2^{13}. The figure shows no sign of error floor down to the BERs of 10^{-9}.

B. Girth of Polar Codes

Here, we discuss the girth of the tanner graph for polar codes. The *girth* of a graph is the length of shortest cycle contained in the graph. When decoded by belief propagation, the external information at every variable node remains uncorrelated until the iteration reaches its girth. We are often interested in constructing large girth codes that can achieve

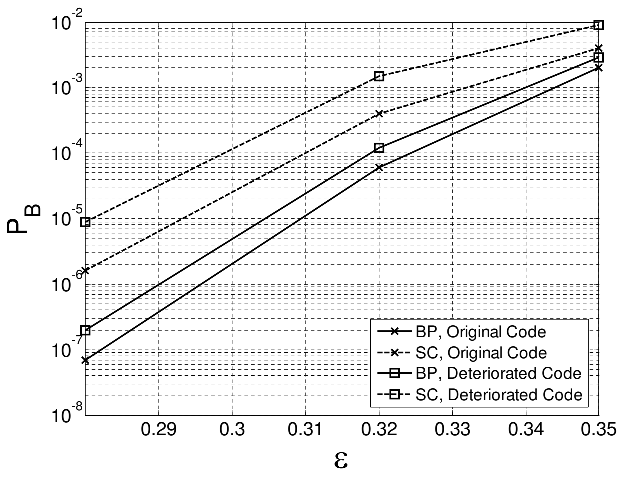
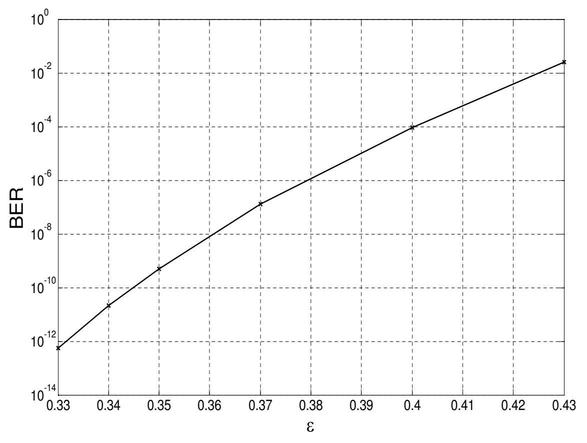
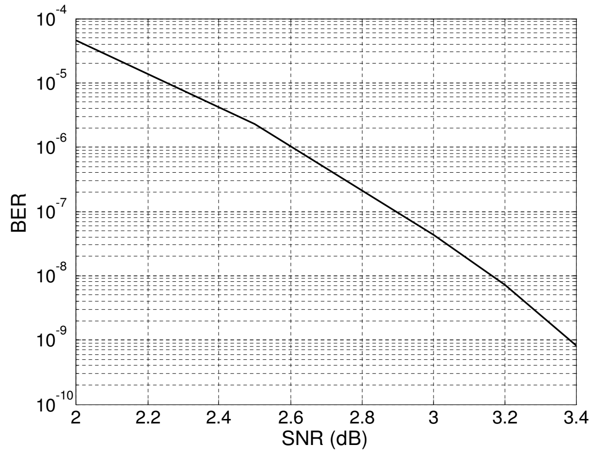


Fig. 2. Performance comparison of BP and SC when 2^9 information bits are replaced by fixed bits in a code of length 2^{13} and rate $\frac{1}{2}$.



(a) BER for BP decoding over BEC. The code-length and code-rate are 2^{15} and $1/2$, respectively.



(b) BER for BP decoding over gaussian channel. The code-length and code-rate are 2^{13} and $1/2$, respectively.

Fig. 3. BER performance of polar codes over the binary erasure and gaussian channels.

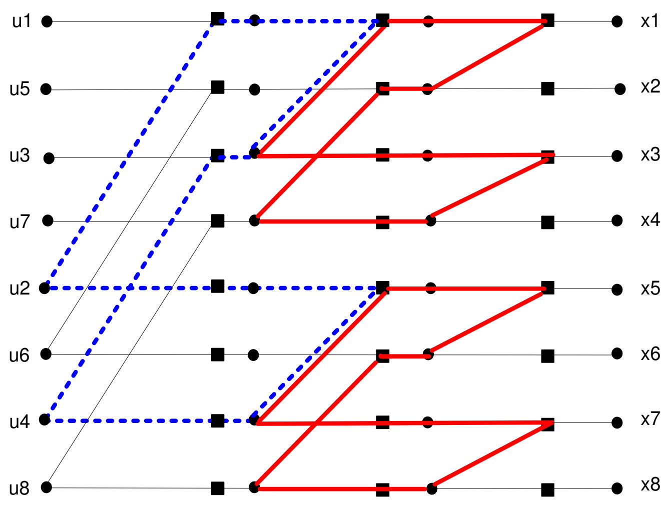


Fig. 4. Different types of cycles in the tanner graph of polar codes for $N = 8$. Thick solid and dashed lines show the first and second types of cycles, respectively.

high performance under belief propagation decoding. As it can be seen in the tanner graph shown in Figure 4, there exist two types of cycles: first, the cycles including nodes only from one of the top or bottom part of the graph (shown by thick solid lines), and second, the cycles including nodes from both top and bottom parts of our symmetric graph (shown by thick dashed lines). The first type of cycles have the same shape in both upper and lower halves of the graph. The interesting fact about the cycles is that because the graph for a code of length 2^m is contained in the graph of a code of length 2^{m+1}, all the cycles of the shorter code are also present in the graph of the longer code. The shortest cycle appears in the graph of a length-4 polar code, a cycle of size 12, including 6 variable nodes and 6 check nodes as it is shown in Fig. 4. On the other hand, it can be checked that the second type of cycles mentioned above actually follows the same skeleton as the code length, and so the size of graph, becomes larger. The minimum size of this type of cycles is also 12 and is related to the length-8 code (dotted lines in Fig. 4). Thus, based on the above, the girth of a polar code is 12. This can be seen as a natural advantage of polar codes over LDPC codes. In fact, a girth of 12 or more is considered so desirable for LDPC codes that many techniques have been proposed in the literature to guarantee such girths (see for example [24] and references therein).

IV. Improved Decoding Using Guessing Algorithm

Fig. 5 provides a comparison between the bit error rate performance of BP and maximum likelihood (ML) decoding for polar codes over a BEC. As it can be seen, ML decoding lead to error rates as large as four orders of magnitude better than the BP. This, along with relatively poor error rate performance of finite-length polar codes compared to LDPC codes, motivates us to find modifications to BP in order to improve its performance. Since LDPC uses belief propagation decoding, there have been various methods proposed to

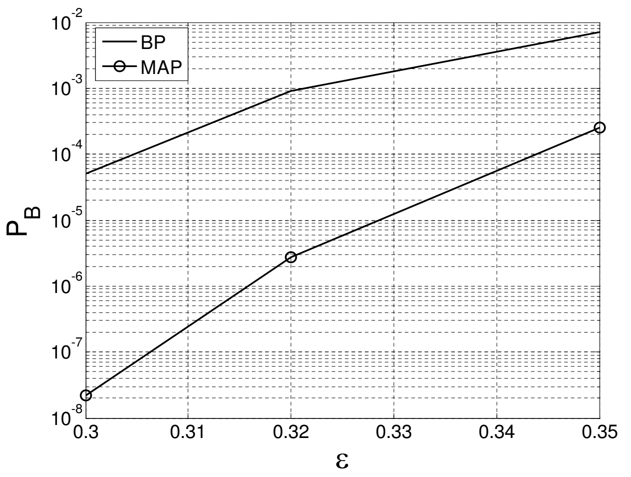


Fig. 5. BER comparison of BP and MAP for a polar code of length 2^{10} and rate 1/2 over the binary erasure channel.

improve the BER performance of belief propagation in LDPC codes. Many of those ideas can be used for polar codes with a slight modification. However, as we have seen in the previous section, polar codes do not show error floor, benefiting from a large stopping distance and a relatively large girth. One of the schemes proved to be helpful for codes with such characteristics is to use *guessing algorithms* alongside BP [18]. The key idea is the following observation. Consider a BEC with an erasure probability ϵ and a polar code of finite length N that has a small enough error probability. If the message-passing decoder fails to decode a received word completely, then there exists a few (usually less than or equal to 3 bits) undecoded bits such that if their values are exposed to the decoder, then the decoder can finish the decoding successfully. Note that this is true only when the BER is small enough (for example, less than 10^{-2}). Simulations and intuitive arguments strongly confirm the above statement.

In message passing algorithm basically, if the values of all but one of the variable nodes connected to a check node are known, the missing variable bit is set to the XOR of the other variable nodes and the check node is labeled "finished". Message passing continues this procedure until all check nodes are labeled as finished or the decoding cannot continue further. Let us call this "algorithm A". We now explain a modified message passing algorithm which we call "algorithm B". This algorithm continues the decoding when algorithm A fails to decode the received codeword. It chooses one of the unknown variable nodes, say w_1, and guesses its value (for example, by setting its value to zero). Intuitively, an appropriate scheme is to choose w_j that guessing its value frees as many as unknown variable nodes. In polar codes, since all variable nodes are degree 2 or 3, we choose variable nodes of degree 3 to guess their values. Then the algorithm continues to label the check nodes as in algorithm A with one more option. If all the variable nodes connected to the check node are known, then if the check node is satisfied it labels that check node "finished," otherwise the check node is labeled "contradicted." The procedure is done sequentially and the algorithm continues to run until either all check nodes are labeled or the decoding cannot continue further.

Once the above procedure is finished, if all of the check nodes are labeled and none of them is labeled "contradicted," the decoder outputs the resulting word as the decoded word. If all of the check nodes are labeled but some of them are labeled "contradicted," then it changes the value of w_1, the guessed variable node, and repeats the decoding from there. This time the decoding finishes successfully because we have found the actual value of w_1. But if the decoding stops again (i.e. some of the check nodes are not labeled) we have to choose another unknown variable node w_2 and guess its value to continue the decoding. Again, if some check nodes are labeled as "contradicted," we have to go back and try other values for w_1 and w_2. Obviously, Algorithm B is efficient only if the number of guesses is very small.

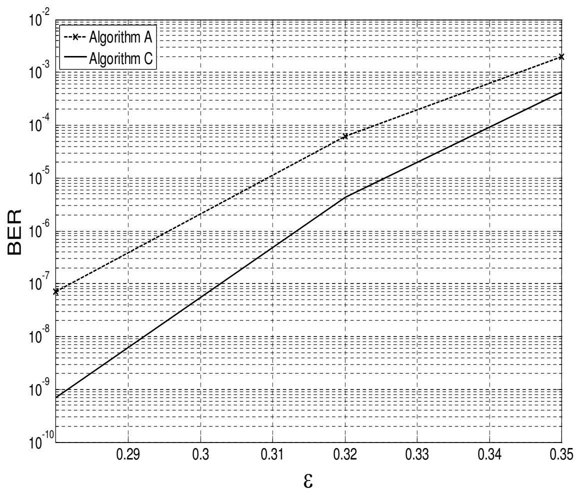
Algorithm B has a complexity that grows exponentially with the number of guesses. An improved algorithm called algorithm C was proposed in [18] to address this problem. Here we explain the basic idea of this algorithm. Let $w_1, w_2, ..., w_g$ be the variable nodes that we guess and $x_1, x_2, ..., x_g$ be their values. In general, any variable node that is determined after the first guess can be represented as $a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus ... \oplus a_gx_g$, where $a_j \in \{0, +1\}$. Algorithm C uses this parametric representation of the variable nodes to solve the set of equations obtained at the satisfied check nodes. This way, it finds the values of $x_1, x_2, ..., x_g$ and hence, the unknown variable nodes. It can be shown that this algorithm has complexity $O(g_{max}^2 N)$ where g_{max} is the maximum number of guesses [18]. We refer the reader to [18] for more details on this algorithm. Algorithm B can also be modified slightly to be used for the gaussian channel [19]. Since the basic idea is the same as the erasure channel, we omit the detailed discussion of this case here. We will show the simulation results for both cases in the next section.

A. Simulation Results

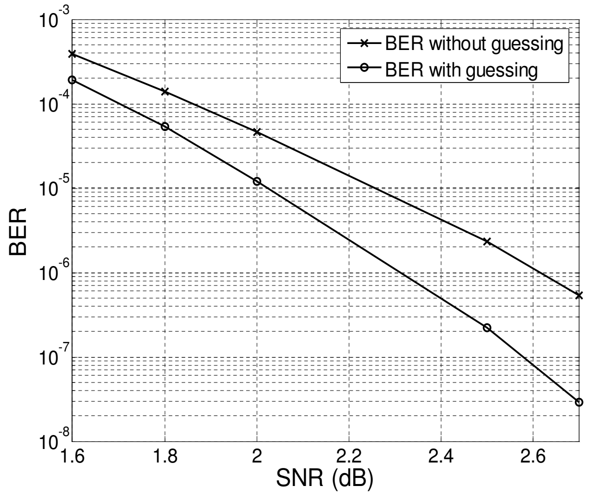
Fig. 6(a) shows the simulation results for BER over BEC, where Algorithm C is compared to algorithm A. Note that Algorithms B and C show almost the same BERs. We have run our simulations for a rate 1/2 polar code of length 2^{13} while we set g_{max} to 6. As it can be seen in the figure, Algorithm C shows two orders of magnitude improvement in BER over Algorithm A. We also observed that the average running time of Algorithm C was about 1.04 times of Algorithm A. The average number of guesses is 3.07 when $\epsilon = 0.32$. Fig. 6(b) shows the simulation results for employing the guessing algorithm in the gaussian channel. The code we are using is of length 2^{13} and has a rate of $\frac{1}{2}$. The maximum number of guesses g_{max} is set to 6. As it can be seen, there is about 0.3 dB improvement in the BER of 2×10^{-6}.

V. Conclusion

We studied the BER performance of polar codes under belief propagation decoding. We analyzed the stopping sets in the tanner graph of polar codes as one of the main contributors to the decoding failure and error floor. We obtained the



(a) BER performance of BP with guessing algorithm for decoding over BEC. Code length is 2^{13} and code rate is 1/2. g_{max} is set to 6.



(b) BER performance of BP with guessing algorithm for decoding over the gaussian channel. Code length is 2^{13} and code rate is 1/2. g_{max} is set to 6.

Fig. 6. BER performance of BP with guessing algorithm over the binary erasure and gaussian channels.

minimal stopping set and its size. We then investigated the error floor performance of polar codes through simulations where no sign of error floor was observed down to BERs of 10^{-11}. Motivated by the good error floor performance, we investigated the application of a guessing algorithm to improve the performance of BP decoding. Our simulations showed that using such a modified version of BP decoding can result in up to 2 orders of magnitude improvement in BER.

REFERENCES

[1] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions o Information Theory*, vol. 55, pp. 3051–3073, July 2009.

[2] E. Arikan, "Source polarization," in *IEEE International Symposium on Information Theory (ISIT)*, June 2010.

[3] E. Arikan, "A performance comparison of polar codes and reed-muller codes," *IEEE Communications Letters*, vol. 12, no. 6, pp. 447 – 449, 2008.

[4] N. Hussami, S. Korada, and R. Urbanke, "Performance of polar codes for channel and source coding," in *IEEE International Sympousiom on Information Theory (ISIT)*, 2009.

[5] R. Mori and T. Tanaka, "Performance and construction of polar codes on symmetric binary-input memoryless channels," in *IEEE International Sympousiom on Information Theory (ISIT)*, pp. 1496 – 1500, 2009.

[6] S. Hassani, S. Korada, and R. Urbanke, "The compound capacity of polar codes," in *47th Annual Allerton Conference on Communication, Control, and Computing*, pp. 16–21, 2009.

[7] S. Korada, E. Sasoglu, and R. Urbanke, "Polar codes: Characterization of exponent, bounds, and constructions," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 1483 – 1487, 2009.

[8] S. Korada and R. Urbanke, "Polar codes are optimal for lossy source coding," *IEEE Transactions on Information Theory*, vol. 56, no. 4, pp. 1751 – 1768, 2010.

[9] E. Sasoglu, E. Telatar, and E. Arikan, "Polarization for arbitrary discrete memoryless channels," in *IEEE Information Theory Workshop (ITW)*, pp. 144 – 148, 2009.

[10] S. Hassani and R. Urbanke, "On the scaling of polar codes: The behavior of polarized channels," in *IEEE International Symposium on Information Theory (ISIT)*, June 2010.

[11] S. Korada, A. Montanari, E. Telatar, and R. Urbanke, "An emprical scaling law for polar codes," in *IEEE International Symposium on Information Theory (ISIT)*, June 2010.

[12] T. Tanaka and R. Mori, "Refined rate of channel polarization," in *IEEE International Symposium on Information Theory (ISIT)*, June 2010.

[13] R. Muri and T. Tanaka, "Channel polarization on q-ary discrete memoryless channels by atbitrary kernels," in *IEEE International Symposium on Information Theory (ISIT)*, June 2010.

[14] H. Cornie and S. Korada, "Lossless source coding with polar codes," in *IEEE International Symposium on Information Theory (ISIT)*, June 2010.

[15] M. Karzand and E. Telatar, "Polar codes for q-ary source coding," in *IEEE International Symposium on Information Theory (ISIT)*, June 2010.

[16] H. Mahdavifar and A. Vardy, "Achieving the secrecy capaity of wiretap channels using polar codes," in *IEEE International Symposium on Information Theory (ISIT)*, June 2010.

[17] M. Bakshi, S. Jaggi, and M. Effros, "Concatenated polar codes," in *IEEE International Symposium on Information Theory (ISIT)*, June 2010.

[18] H. Pishro-Nik and F. Fekri, "On decoding of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inform. Theory*, vol. 50, pp. 439–454, 2004.

[19] H. Pishro-Nik and F. Fekri, "Results on punctured low-density parity-check codes and improved iterative decoding techniques," *IEEE Trans. on Inform. Theory*, vol. 53, pp. 599–614, February 2007.

[20] C. Di, D. Proietti, I. E. Telatar, T. Richardson, and R. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inform. Theory*, vol. 48, pp. 1570 –1579, 2002.

[21] C. Di, T. J. Richardson, and R. L. Urbanke, "Weight distribution of low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 4839 – 4855, 2006.

[22] A. Orlitsky, K. Viswanathan, and J. Zhang, "Stopping set distribution of ldpc code ensembles," *IEEE Transactions on Information Theory*, vol. 51, no. 3, pp. 929 – 953, 2005.

[23] A. Orlitsky, R. Urbanket, K.Viswanathan, and J. Zhang, "Stopping sets and the girth of tanner graphs," in *IEEE International Symposium on Information Theory (ISIT)*, June 2002.

[24] M. Esmaeili and M. Gholami, "Geometrically-structured maximum-girth ldpc block and convolutional codes," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 6, pp. 831–845, 2009.