symmetry

MDPI

Article

Review and Evaluation of Belief Propagation Decoders for Polar Codes

Lingxia Zhou [1,2,†], Meixiang Zhang [1,†], Satya Chan [2,†] and Sooyoung Kim [2,*,†]

[1] College of Information Engineering, Yangzhou University, Yangzhou 225009, China
[2] Division of Electronics Engineering, IT Convergence Research Center, Jeonbuk National University, Jeonju 54896, Republic of Korea
* Correspondence: sookim@jbnu.ac.kr
† These authors contributed equally to this work.

Abstract: The polar code has become one of the most popular and important forward error correction (FEC) coding schemes due to its symmetric characteristics of channel polarization. This paper reviews various decoding schemes for polar codes and discusses their advantages and disadvantages. After reviewing the existing performance-enhancing techniques such as belief propagation decoding with list, a new method is proposed to further improve the performance. In addition, a new complexity reduction technique based on the constituent codes is proposed, and a new scheduling scheme is introduced to reduce the decoding latency. Due to the recent development of neural networks, their applications to decoding schemes are also reviewed and evaluated. Finally, the proposed complexity-reduced technique is integrated with a neural network-based belief propagation decoding, which demonstrates performance enhancement as well as computational complexity reduction.

Keywords: polar codes; belief propagation decoding; factor graph; iterative decoding; neural networks

Citation: Zhou, L.; Zhang, M.; Chan, S.; Kim, S. Review and Evaluation of Belief Propagation Decoders for Polar Codes. *Symmetry* **2022**, *14*, 2633. https://doi.org/10.3390/sym14122633

Academic Editors: Jia Hou, Jun Li and Xueqin Jiang

Received: 29 October 2022
Accepted: 7 December 2022
Published: 13 December 2022

1. Introduction

The polar code was invented by using symmetric channel polarization properties, and it has been widely studied because of its ability to achieve channel capacity under infinite code length [1]. Since it has been specified as one of the forward error correction (FEC) schemes for the 5G system [2], efficient implementation of the polar code with low latency has become a hot issue to achieve a 5G vision of ultra-reliable and low latency communications (URLLC). The successive cancellation decoding (SCD) algorithm was introduced as an effective means of reaching the performance approximating the channel capacity with low complexity [3]. Even though there was an effort to speed up the decoding time of the successive cancellation (SC) decoder [4], the decoding time, i.e., the delay of the SC decoder is relatively high because of its serial processing characteristics, and it was considered to be unsuitable for high-speed computing.

For practical hardware implementation, belief propagation decoding (BPD) for polar codes has been widely studied, due to its high throughput [1]. Even though a BPD method accelerates the decoding time by taking advantage of its parallel processing characteristics, it should sacrifice the performance gain compared to the SCD method. Research on efficient BPD algorithms has been carried out from various aspects since BPD is a realistic option for hardware implementation. This paper reviews various research on BPD methods for polar codes and analyzes their pros and cons by evaluating their performances.

This study first reviews and analyzes a means to improve the performance of BPD, which is called BPD with list (BPDL) [5–13]. The BPDL operates on multiple permuted factor graphs to select the best decoding result, and eventually to improve the decoding performance. After reviewing the basics of BPDL, a simple and compact method is newly proposed to further improve the performance, compared to the existing BPDL.

Although BPDL can improve the performance of BPD, its computational complexity increases linearly with the number of list size. Therefore, the second technique analyzed in this paper is about computational complexity-reducing methods. It was considered that specific parts of the factor graph did not require full computations during the decoding process, and these parts were called constituent codes [14]. These constituent codes could be removed from the factor graph in the BPD process, thereby reducing the decoding complexity. Because a BPD algorithm works iteratively, this removal of constituent codes from the BPD process played an important role in reducing the decoding latency and computational complexity.

As deep learning has become one of the current research hotspots, efforts have been made to combine BPD with deep learning [15–19]. Therefore, the third technique to review is to apply machine learning to BPD. Various network structures were proposed for BPD to improve the error rate performance. This paper details the above-mentioned research on BPD, and newly proposes combining one of the deep learning techniques with complexity-reduced BPD, analyzing the error performance and latency characteristics.

As far as the hardware implementation is concerned, the scheduling scheme is one of the most important techniques to determine convergence speed and decoding latency. Therefore, the fourth technique to review is about the scheduling scheme of BPD. In the early BPD study, a one-way scheduling scheme was used [20]. Later, a scheduling scheme called round-trip scheduling (RTS) was proposed to speed up the convergence speed of decoding [21]. Even though segmented scheduling (SS) methods with symmetric structures were proposed to further accelerate the decoding speed at the expense of computations [22], they degraded the decoding performance, because of the uneven distribution of information. As an effective means of taking advantage of the existing methods, this paper proposes a new hybrid method to combine RTS and SS to improve the decoding speed, as well as not to degrade the performance.

The contribution of this paper is therefore summarized as follows: (1) Providing surveys and reviews on various aspects of BPD for polar codes; (2) Proposing a new BPDL method to improve the performance of the existing BPDL; (3) Proposing the application of a neural network-based BPD for polar codes in combination with a complexity-reduced mechanism; (4) Proposing a hybrid scheduling method; (5) Presenting extensive simulation results on various BPD algorithms, and analysis of the comparison results; (6) Presenting a comparison of computational complexity and computing time for various BPD algorithms.

The remaining part of the paper is structured as follows. Section 2 describes the basics of polar codes, including the concept of the polar code and its symmetric encoding principle. Furthermore, the principles of the SCD and BPD methods are explained. Section 3 investigates various techniques for BP decoders, including BPDL, complexity-reducing methods, application of deep learning to BP decoders, and finally, scheduling methods for implementation. Section 4 provides extensive simulation results of the investigated techniques and analyzes the computational complexities, as well as the latency characteristics. Finally, Section 5 concludes the paper.

2. Polar Codes

2.1. The Concept and Encoding

Symmetric channel polarization is the theoretical basis of the polar code, which is composed of channel union and channel splitting [3]. Figure 1 shows the Tanner graph, also known as the factor graph of a polar code with length, $N = 8$, where u_j, x_j, and y_j are the jth input to the encoder, output of the encoder, and received information after passing through the channel W, respectively. Assuming a binary erasure channel W with an erasure probability of 0.5, the channel capacity of each input path (channel) will be 0.5 before encoding. On the other hand, the channel capacity of each channel is becoming symmetrically polarized after a few stages of unions and splits, as shown in Figure 1, where $I(W^{(8)})$ denotes the capacity of each channel polarized by the polar code with $N = 8$.

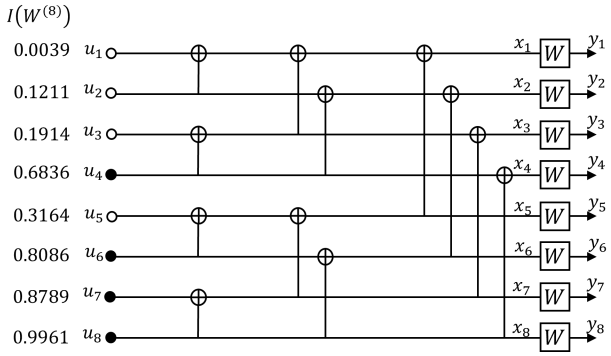


Figure 1. Tanner graph of a polar code with $N = 8, R = 1/2$.

As the number of channels tends to be infinite, the polarization becomes more evident, i.e., all channels are divided into two types: no-noise channels and all-noise channels. The coding principle of the polar code takes advantage of the characteristics of this phenomenon. The information required by the receiving end is transmitted through the reliable channels with higher capacity, and the agreed information or no information is transmitted through the unreliable channels with lower capacity. In other words, we load information onto K input bits with higher capacity, while we freeze the values of the remaining $N - K$ input bits with lower capacity. This makes the code rate, R of K/N. The example in Figure 1 shows the case when $K = 4$ and $R = 1/2$. The encoding is performed by recursive exclusive-or (XOR) operations of the frozen and information bits as shown in Figure 1, where the information and frozen bits are denoted with white and black symbols, respectively.

The XOR operations of the encoding process to produce the codeword $\mathbf{x}$ with the input $\mathbf{u}$ can be generalized using the generator matrix, as follows:

$$\mathbf{x} = \mathbf{u} \cdot \mathbf{G}_N, \tag{1}$$

where $\mathbf{G}_N$ is the $N \times N$ generator matrix, which can be recursively obtained from the Kronecker matrix operations from the kernel matrix $\mathbf{F}$, i.e.,

$$\mathbf{G}_N = \mathbf{F}^{\otimes m}, \ \ \mathbf{F} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \tag{2}$$

and $m = \log_2 N$.

2.2. Decoding Methods

2.2.1. Successive Cancellation Decoding

When decoding the polar codes, different decoding strategies should be adopted, whether it is a frozen or information bit. For the frozen bits, the estimated bit value, $\hat{u}_j$ can be directly set to $\hat{u}_j = u_j$. On the other hand, the bit decision is made with the aid of the soft information propagation and bit decision on the frozen bits [3].

The decoder is first fed with the soft information of $\mathbf{x}$ in the form of log-likelihood ratio (LLR), by the demodulator. Then, the LLR values, $L_{i,j}$s are propagated from right to left, where $L_{i,j}$ denotes the LLR value at the jth node of the ith stage, $1 \leq i \leq m$ and $1 \leq j \leq N$. The initial LLR value provided by the demodulator at the $(m+1)$th stage is expressed as follows:

$$L_{m+1,j} = \ln\left(\frac{P(y_j|0)}{P(y_j|1)}\right), \tag{3}$$

where $P(y_j|0)$ and $P(y_j|0)$ are the probabilities of y_j being 0 and 1, respectively.

Afterwards, the LLR message passing is performed using the following formula:

$$L_{i,j} = \begin{cases} 2\tanh^{-1}\left(\tanh\frac{L_{i+1,j}}{2}\tanh\frac{L_{i+1,j+2^{i-1}}}{2}\right), & \text{if } \lfloor\frac{j-1}{2^{i-1}}\rfloor \mod 2 = 0, \\ \left(1-2s_{i,j-2^{i-1}}\right)L_{i+1,j+2^{i-1}} + L_{i+1,j}, & \text{otherwise,} \end{cases} \tag{4}$$

where $s_{i,j}$ is the hard (binary) information at the jth node of stage i, which is supposed to be propagated from left to right upon receiving the LLR values at the first stage, and making the bit decision on the information. To pass the hard message, the bit decision of the ith bit, $\hat{u}_j$ is required at the leftmost end, and it can be estimated as follows:

$$\hat{u}_j = \begin{cases} u_j, & \text{if } u_j \text{ is frozen bit,} \\ \frac{1-\text{sign}(L_{1,j})}{2}, & \text{otherwise.} \end{cases} \tag{5}$$

The hard message at the first stage $s_{1,j} = \hat{u}_j$, and those of the following stages can be estimated as follows:

$$s_{i+1,j} = \begin{cases} s_{i,j} \oplus s_{i,j+2^{i-1}}, & \text{if } \lfloor\frac{j-1}{2^{i-1}}\rfloor \mod 2 = 0, \\ s_{i,j}, & \text{otherwise.} \end{cases} \tag{6}$$

Using the rate 1/2 code shown in Figure 1, here we detail the process of SCD. Figure 2 shows the propagation of soft and hard messages that proceed back and forth sequentially. First, the decoder receives eight soft inputs from the demodulator, $L_{4,j}, 1 \le j \le 8$. Second, $L_{3,j}, 1 \le j \le 4$ and $L_{2,j}, 1 \le j \le 2$ are estimated at stages three and two, respectively, by using (4). Then, reaching the first stage, the bit decision on u_1 and its hard message $s_{1,1}$ are estimated by using (5) and (6), respectively. Next, with the available value of $s_{1,1}$, $L_{2,1}$ is estimated using (4). After that, hard messages $s_{2,1}$ and $s_{2,2}$ can be estimated using (6). With this procedure, soft and hard messages can be alternatively estimated, until all the bit decisions are made.

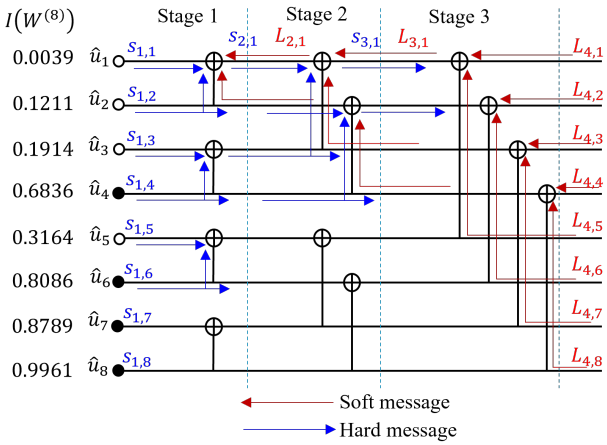


Figure 2. Message flow of SCD for the polar code with $N = 8, R = 1/2$.

2.2.2. Belief Propagation Decoding

The BPD is an algorithm transferring information on a factor graph, and the performance can be enhanced by iteratively exchanging the soft information between input and output nodes. The BPD has been conventionally used to decode low-density parity-check (LDPC) codes, and it was applied to the polar codes [1].

For each node of the factor graph, there are two directions of soft information in the form of LLR. The LLR propagated from right to left is denoted as $L_{i,j}$, while that from left to right is denoted as $R_{i,j}$. The decoder employs the so-called processing element (PE), which

performs the following LLR estimations with the Min-Sum approximation, so that LLRs are traversed through the factor graph.

$$\begin{cases} L_{i,j}^{(t)} = g(L_{i+1,j}^{(t-1)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}), \\ L_{i,j+N/2^i}^{(t)} = g(L_{i+1,j}^{(t-1)}, R_{i,j}^{(t)}) + L_{i+1,j+N/2^i}^{(t-1)}, \\ R_{i+1,j}^{(t)} = g(R_{i,j}^{(t)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}), \\ R_{i+1,j+N/2^i}^{(t)} = g(L_{i+1,j}^{(t-1)}, R_{i,j}^{(t)}) + R_{i,j+N/2^i}^{(t)}, \end{cases} \tag{7}$$

where $g(x,y)$=sign(x)sign(y)min$(|x|,|y|)$. In addition, the superscript of $^{(t)}$ denotes the iteration index, because the BPD works iteratively. The LLR value in the rightmost column of the factor graph, $L_{m+1,j}$ is fed by the soft demodulator as in (3). On the other hand, the LLR value in the leftmost column, $R_{1,j}$ is the pre-decoded LLR of the information sequence $\mathbf{u}$, and ∞ and 0 are assigned to the frozen and the information bits, respectively, at the beginning of decoding.

Figure 3 shows an example of LLR exchange with PE over the factor graph of the same polar code as in the previous example. BPD is performed on the factor graph by operating the PE from left to right, and from right to left to update L and R, respectively. The final decision on $\hat{\mathbf{u}}$ and $\hat{\mathbf{x}}$ are, respectively, estimated after the completion of T iterations, as follows:

$$\hat{u}_j = \begin{cases} 0, & \text{if } L_{1,j}^{(T)} \geq 0, \\ 1, & \text{otherwise,} \end{cases} \tag{8}$$

$$\hat{x}_j = \begin{cases} 0, & \text{if } L_{1+m,j}^{(T)} + R_{1+m,j}^{(T)} \geq 0, \\ 1, & \text{otherwise.} \end{cases} \tag{9}$$

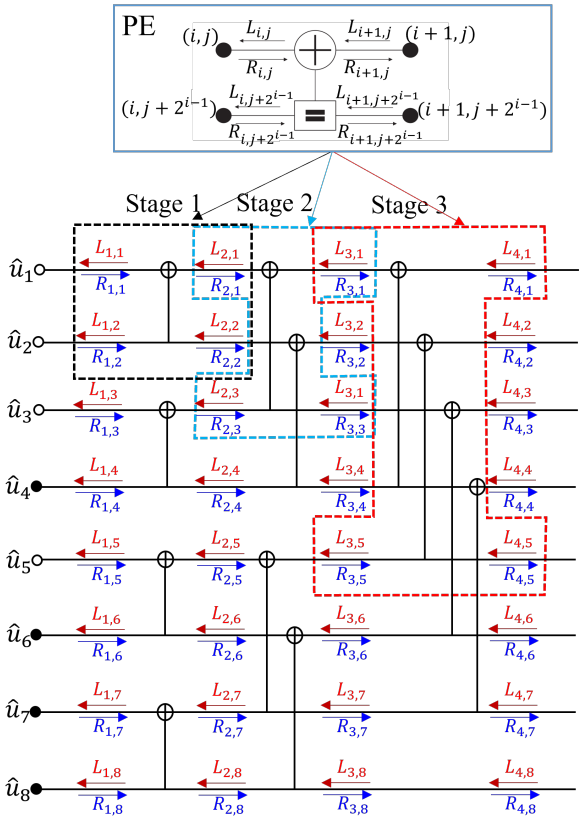


Figure 3. BPD with PE for the polar code with $N = 8, R = 1/2$.

3. Variations of Belief Propagation Decoders and Implementation Issues

3.1. Belief Propagation Decoding with List, BPDL

3.1.1. Review of the Previous Works on BPDL

Since the critical disadvantage of BPD compared to SCD is performance degradation, efforts have been made to improve the performance by employing multiple permuted factor graphs, referred to as BPD with list (BPDL). BPDL could effectively improve the decoding performance at the expense of decoding complexity.

There exist $m!$ different permutations of the factor graph for a polar code of length N. In the BPDL, P permutations of the factor graph, $P << m!$, are used for decoding, and the permutation that satisfies the early stopping criterion is used as the final decoding result. A random permutation method was adopted in [5], where the nodes from stages 1 to m are randomly exchanged. This led to a very large number of permutations and increased the decoding complexity. To solve this problem, cyclic shift permutation methods were proposed [6]. For example, the order of the original stage $\{1, 2, ..., m\}$ in the factor graph is changed to $\{m, 1, ..., m-1\}$ if the cyclic left shift permutation is used. Figure 4 shows an example of a cyclic left shift of permutation of the factor graph for the polar code in the previous example with $N = 8, R = 1/2$. This cyclic shift permutation could efficiently reduce the number of permutations, and eventually, lead to reducing the decoding complexity of BPDL.

In addition, various permutation methods were introduced to find better factor graphs with which the BP decoder converged faster [7–12]. Each factor graph requires the corresponding decoder structure, and thus the complexity and the memory requirement of the BPDL hinder practical implementation. A method to map each permuted factor graph to the permuted codeword was proposed; thereby, a single decoder can be used [13].

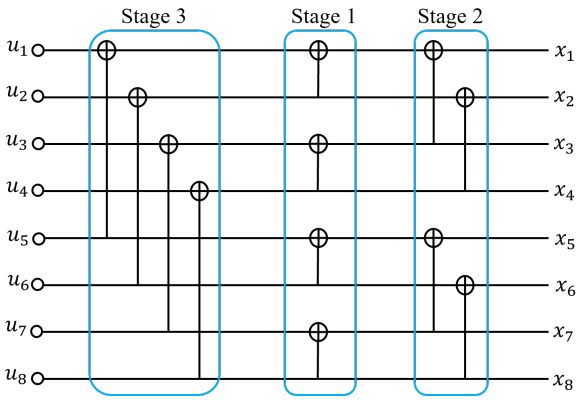


Figure 4. Example of cyclic left shift permutation for the polar code with $N = 8, R = 1/2$.

3.1.2. A New Proposed BPDL

Additionally, we propose a simple new BPDL scheme, which multiplies a scaling coefficient to L and R. We call this method the scalable BPDL (S-BPDL), and the update formula of L and R in the S-BPDL is as follows:

$$\begin{cases} L_{i,j}^{(t)} = \rho_p \cdot g(L_{i+1,j}^{(t-1)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}), \\ L_{i,j+N/2^i}^{(t)} = \rho_p \cdot g(L_{i+1,j}^{(t-1)}, R_{i,j}^{(t)}) + \rho_p \cdot L_{i+1,j+N/2^i}^{(t-1)}, \\ R_{i+1,j}^{(t)} = \rho_p \cdot g(R_{i,j}^{(t)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}), \\ R_{i+1,j+N/2^i}^{(t)} = \rho_p \cdot g(L_{i+1,j}^{(t-1)}, R_{i,j}^{(t)}) + \rho_p \cdot R_{i,j+N/2^i}^{(t)}, \end{cases} \tag{10}$$

where $-0.5 \leq \rho_p \leq 0.5, 1 \leq p \leq P$, is a scaling factor that is applied to the pth BPD. The S-BPDL applies P different scaling factors to the BPD with the same factor graph, instead of utilizing different permutations of the factor graph; thereby, the result satisfying the early stopping criterion is used as the final decoding result. Since this method uses the

same factor graph, it requires much less memory requirement, which is almost the same as the conventional BPD. In addition, this method provides better error rate performance compared to the existing BPDL, which is demonstrated in Section 4.

3.2. Belief Propagation Decoding with Reduced Factor Graph

It was discovered that a part of the factor graph for the entire codeword did not require certain computations during the decoding process, and this part was called the constituent code [14]. The following four constituent codes were found, where computing $R_{i,j}^{(t)}$ can be simplified to the following $R_{i,j}^{\prime(t)}$, due to the removal of the computations for the constituent code. In the following, l is the length of the constituent code, and e is the stage index where the constituent code ends at the factor graph, and can be estimated as $e = \log_2 l$, $0 < e \leq m$.

N^0 **code:** All the leaf nodes of this constituent code are frozen bits, whose LLR, $R_{0,j}$ is set to ∞, this leads to:

$$R_{e,j}^{\prime(t)} = \infty. \tag{11}$$

N^1 **code:** This is the opposite case of N^0; all leaf nodes are information bits, so $R_{0,j}$ is set to 0, and during the decoding process, it is always maintained as 0. This leads to:

$$R_{e,j}^{\prime(t)} = 0. \tag{12}$$

N^{REP} **code:** This constituent code has a single information bit on the last leaf node. Since each node is a duplication of others, they share the belief messages with others in the factor graph. This leads to:

$$R_{e,j}^{\prime(t)} = \sum_{k \neq j} L_{e,k}^{(t)}. \tag{13}$$

N^{SPC} **code:** This has a single frozen bit on the first leaf node. In this case, R update can be simplified as follows:

$$R_{e,j}^{\prime(t)} = \prod_{k \neq j} \operatorname{sign}(L_{e,k}^{(t)}) \cdot \min_{k \neq j} |L_{e,k}^{(t)}|. \tag{14}$$

The above simplifications of LLR computations enhance the computational efficiency without degrading error performance. In particular, this kind of reduction in LLR computation is highly effective for BPD, since it works iteratively. This idea was called express-journey (XJ) BPD [14], and it is demonstrated in Section 4 that XJ-BPD produces exactly the same performance as the conventional BPD, with much lower decoding complexity.

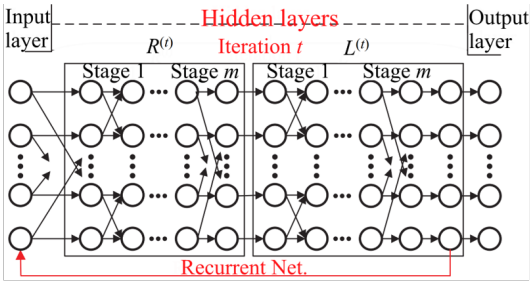
3.3. Belief Propagation Decoding with Neural Network

3.3.1. Review of the Previous Works on BPD with Neural Network

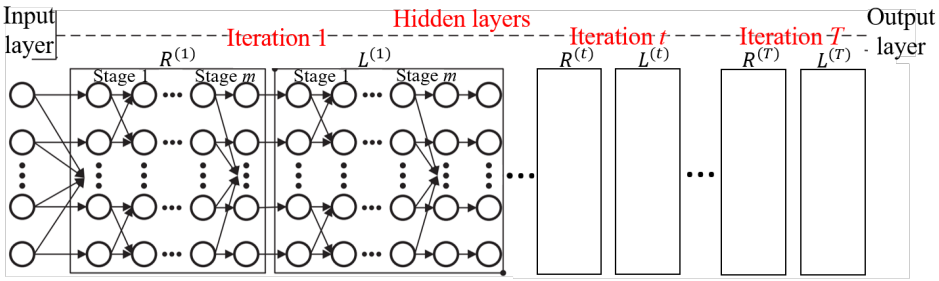
With the widespread use of machine learning in various applications, efforts have been made to apply deep learning to BPD. A deep neural network (DNN) was used for BPD by unfolding the iterative structure into the layered structure of the neural network, resulting in BPD with DNN [15,16]. This DNN-based BPD imitates the performance of the classical BPD with trained weight multiplications across the graphs. Although this BPD with DNN improved the performance by virtue of the well-trained weight multiplications across a large number of hidden layers, the complexity of decoding increases sharply, and a large amount of memory is occupied by the storage of the weights. These problems hinder its practical implementation.

To solve this problem, recurrent neural network (RNN)-based decoders were proposed for polar codes [17–19], which assign the same weights across the iterations, and thus require almost the same memory size as the conventional BPD. Figure 5 compares the structure of the DNN-based BPD to that of the RNN-based BPD. The biggest difference

between the RNN-based BPD compared to the DNN-based BPD is that it converts the feed-forward structure into a recurrent network, which forces the decoder to reuse weights in different iterations, leading to completely different optimization problems. In other words, the RNN-based BPD can be regarded as a constrained version of the DNN-based BPD. The RNN-based BPD can effectively reduce a large number of parameters, without significant performance degradation.



(a) RNN-based BPD



(b) DNN-based BPD

Figure 5. Structures of the RNN and DNN-based BPDs.

For example, a residual neural network-based belief propagation (ResNet-BP) algorithm was proposed by reducing the number of variables in the network, and the computational complexity [17]. The simulation results showed that this could achieve the same performance as the conventional BPD with a lower number of iterations. In addition, a practical implementation method of the RNN-based BPD was presented by adopting a weight quantization mechanism [19]. It was shown that the storage overhead could be further reduced by using the codebook-based weighting method.

3.3.2. The Proposed Complexity Reduced BPD with a Neural Network; RNN XJ-BPD

In this study, we combine the XJ-BPD introduced in Section 3.2, with the RNN-based BPD. In this way, XJ-BPD contributes to reducing the complexity of the decoder, while RNN contributes to enhancing the decoding performance. In the proposed RNN XJ-BPD, the weight factors are trained using the conventional BPD, so that it can best approximate the maximum performance. Next, the XJ-BPD is applied with the trained weights. The performance simulation results in Section 4 demonstrate that the performance can be further improved with the aid of RNN.

3.4. Implementation of Belief Propagation Decoding with Segmented Scheduling

3.4.1. Review of the Previous Works on Scheduling for BPD

For the BP decoder, scheduling refers to the method of information dissemination and update from one end of the factor graph to the other end. This has a significant impact on the convergence speed of BPD. The simplest way is the one-way scheduling scheme that activates the phases from left to right to update the L and R information at the same time, and completes an iteration when the activation reaches the rightmost phase [20], as shown in Figure 6.

In this way, the information on the left of the factor graph completes the traversal of the entire factor graph, while the information on the right is actually only updated at stage m, which leads to uneven information transmission. Therefore, it requires more iterations to reach saturation. On the other hand, round-trip scheduling (RTS) is a method that makes the information on the right traverse to the left, and then the information on the left traverse to the right [14], as shown in Figure 6. Because the message is spread and updated more uniformly, fewer iterations are required for convergence, so the overall delay is smaller.

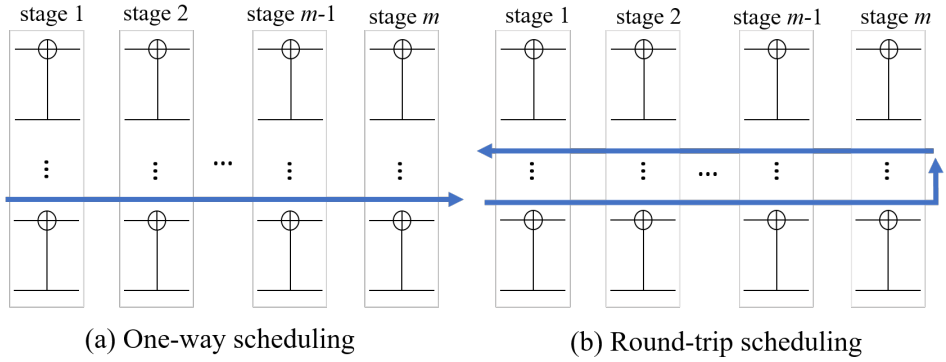


Figure 6. Comparison of one-way and round-trip scheduling.

To accelerate the computing time, new scheduling schemes were proposed by segmenting the factor graph into multiple sub-graphs and performing scheduling in parallel [22]. Figure 7 shows the concept of segmented scheduling (SS) in various forms. For example, the information on the right and left ends of the factor graph is simultaneously propagated to the middle of the factor graph, with halfway scheduling as in (a) of Figure 7. Information exchange occurs in the middle of the factor graph. After that, the newly updated information traverses to the right and left ends of the factor graph, respectively. In other words, the main cycle of information propagation in RTS is divided into two sub-cycles of information flow.

Although this method can reduce the time to traverse the factor graph, the soft information transmission is not as uniform as RTS, resulting in a decrease in performance. This is because the transmission and update of the L and R information are performed at the same time. The factor graph can be further segmented and results in quarter-way scheduling, and finally, a fully segmented structure, as in (b) of Figure 7.

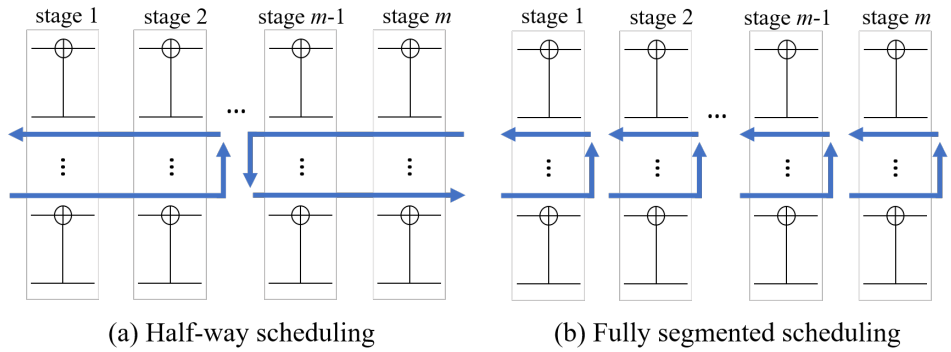


Figure 7. Concept of segmented scheduling.

3.4.2. The Proposed Hybrid Scheduling Scheme

The RTS requires a longer time to update LLR values, and produces LLR values with higher reliability, compared to a segmented scheduling scheme. On the other hand, SS can speed up the decoding time, but it may lead to performance degradation, because of the production of less reliable LLR values. By considering that the reliability of LLR

values will be increased as the number of iterations is increased, a new hybrid scheduling scheme is proposed. This scheme utilizes RTS at the earlier stages of the iteration to prevent performance degradation, while it utilizes a fully segmented scheduling scheme to speed up the decoding time. We compare the performance and decoding time of this idea in Section 4.

4. Performance Evaluation

We conducted BER performance simulations by using the polar codes over an additive white Gaussian noise (AWGN) channel. Since our purpose here is to compare the BER performance and decoding complexity, we utilize simple binary phase shift keying (BPSK) as a modulation scheme. We first analyze the error rate performance of various BPD algorithms, including our new proposals, and then analyze their decoding time and complexity. In addition, we implement RNN and apply it to various existing BPD, as well as our proposed S-BPDL introduced in this paper, and evaluate the performance in comparison with existing methods.

4.1. Error Rate Performance

We first compare the BER performance of various BPD algorithms with that of the SCD. Figure 8 compares the performance of BPDL with SCD, the conventional BPD, and the proposed S-BPDL using the (1024, 512) and (128, 64) polar codes, respectively. In the figure, n_t denotes the number of iterations in the BPD. It is clearly shown that as the codeword length increases, the conventional BPD requires an increasing number of iterations to achieve the same performance as the SCD. The performance of the BPDL is better than that of the traditional BPD, regardless of the codeword length, with list size L. In addition, it is not difficult to see that the decoding performance of the proposed S-BPDL is better than that of the traditional BPDL with the same list size.

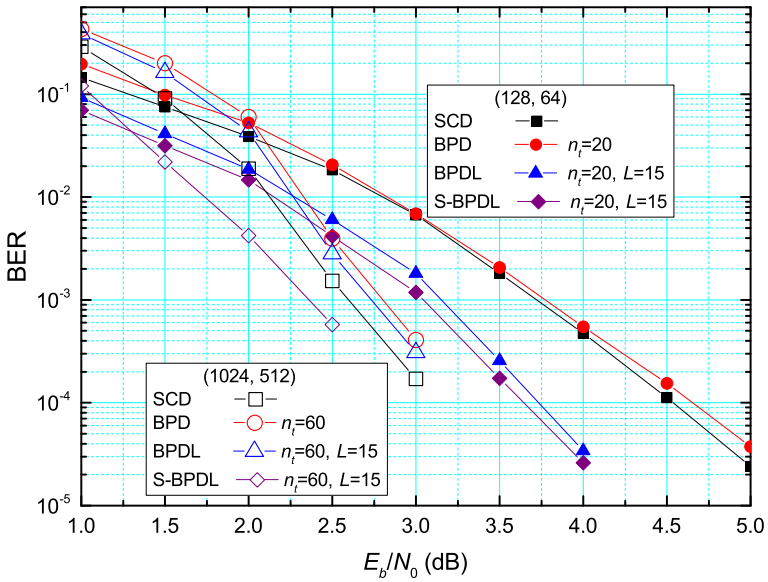


Figure 8. Comparison of the BER performances of BPDL for the (1024, 512) and (128, 64) polar codes.

Next, Figure 9 shows the BER performance comparisons of XJ-BPD according to the number of iterations, n_t, using the (1024, 512) and (128, 64) polar codes. It shows that the performance of XJ-BPD is the same as the conventional BPD, regardless of the number of iterations, proving that it can contribute to reducing the complexity without degrading the performance. Complexity comparison is discussed in the next section.

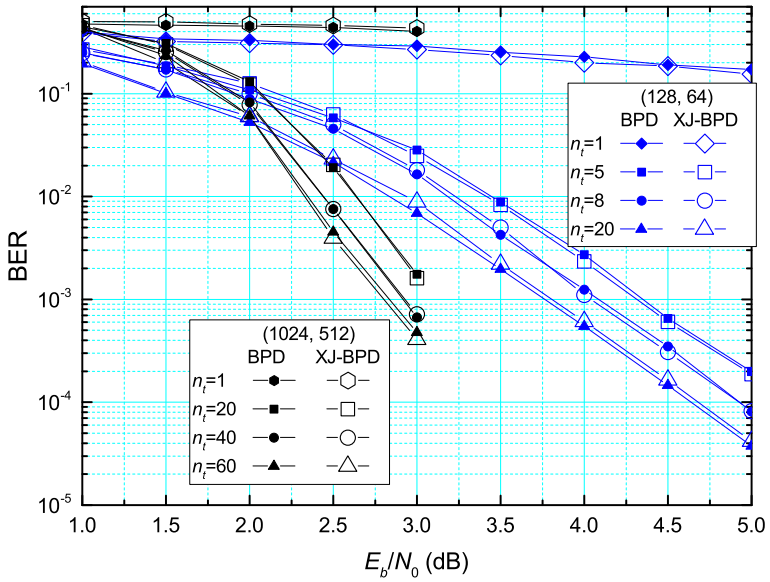


Figure 9. BER performance comparisons of XJ-BPD for the (1024, 512) and (128, 64) polar codes.

We apply RNN to XJ-BPD, and Figure 10 compares the BER performance of the RNN XJ-BPD with those of the conventional BPD, for the (128, 64) polar code. The simulation results show that the RNN-based XJ-BPD does not degrade the performance of the conventional RNN-BPD. Moreover, the RNN-based decoders substantially reduce the number of iterations needed to achieve the saturated decoding performances and outperform the conventional BPD with the same n_t. Specifically, the RNN-BPD only requires eight iterations to achieve the saturated BER performance, which is only 40% of the number of iterations required for the conventional BPD. Although the results in Figure 10 are only for the (128, 64) code, because of the time limitation of the long training process, it is expected that as the codeword length is increased, the gain we can obtain can be further increased.

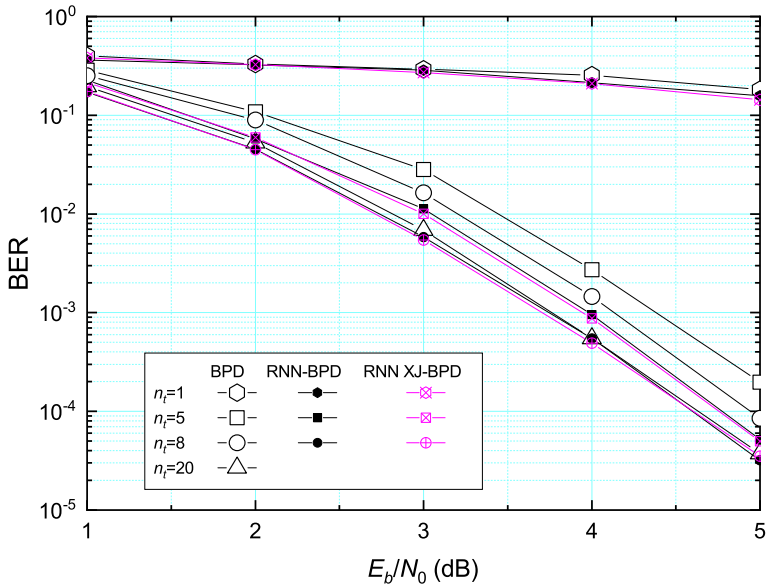


Figure 10. BER performance comparison of RNN-based XJ-BPD for the (128, 64) code.

Finally, Figure 11 compares the BER performance of the three different scheduling methods applied to the (128, 64) polar code. It is evident that the SS shows much worse performance, compared to the RTS. In the proposed hybrid method, the decoder uses RTS at the first two iterations, while it uses SS during the remaining iterations. As can be seen

from the figure, the proposed hybrid method shows much better performance than the SS and it approximates the RTS. Although the number of iterations required by the proposed hybrid method is certainly larger than that by the RTS, the decoding latency of the proposed hybrid method is less than that of the RTS at the same BER performance, as shown in the next section.

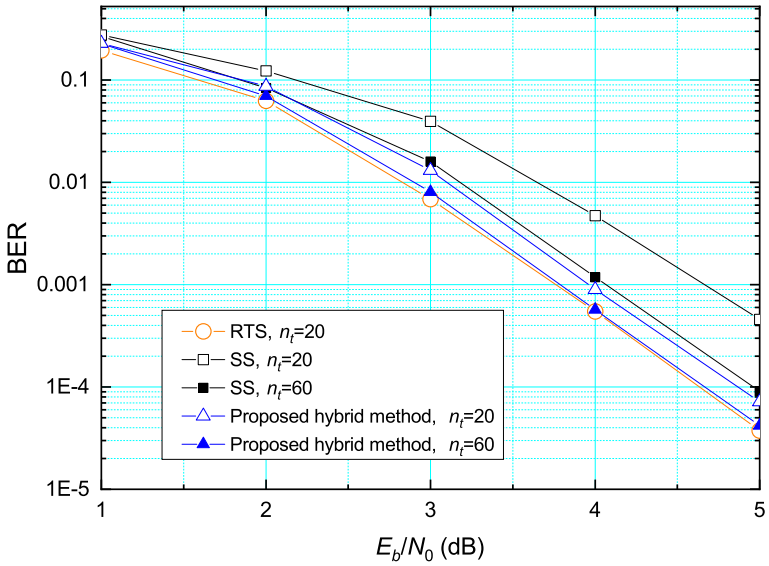


Figure 11. BER performance comparison for the (128, 64) polar code according to the scheduling method.

4.2. Complexity and Latency Performance

We first compare the computational complexities of various decoders in terms of the number of floating point operations (FLOP), n_f, and Table 1 shows the comparison. In the estimation, we assume the Min-Sum algorithms for the BPD and SCD and ignore the sign multiplications. Although the types and numbers of constituent codes are different for each polar code, which makes it impossible for us to specify the exact values, it is evident that the complexities of XJ-BP decoders are much less than those of conventional BP decoders.

Table 1. Complexity comparisons in terms of n_f.

Method	Addition	Multiplication	Comparison
BPD	$2N\log_2 N$	-	$2N\log_2 N$
RNN-BPD	$2N\log_2 N$	$2N\log_2 N$	$2N\log_2 N$
SCD	$(N/2)\log_2 N$	-	$(N/2)\log_2 N$
BPDL	$2N\log_2 N \cdot P$	-	$2N\log_2 N \cdot P$
S-BPDL	$2N\log_2 N \cdot P$	$3N\log_2 N \cdot P$	$2N\log_2 N \cdot P$
XJ-BPD	« $2N\log_2 N$	-	«$2N\log_2 N$
RNN XJ-BPD	«$2N\log_2 N$	«$2N\log_2 N$	«$2N\log_2 N$

Figure 12 shows a comparison of n_f, required at the first iteration and required to achieve the saturated BER performance; i.e., the same performance as the SCD. At the first iteration, the RNN XJ-BPD shows worse computational efficiency compared to the XJ-BPD, because of the additional weight multiplications. It is evident that by effectively eliminating unnecessary computations, the RNN XJ-BPD eventually requires fewer computations than both the conventional BPD and the conventional RNN-BPD.

We compare the decoding latency in terms of the time required to update a node of the information, and we refer to a unit of this time as T_n. Although SCD does not require iterations, its latency is non-negligible, because it operates in a sequential manner. The soft and hard information at the same node cannot be simultaneously updated, and even the

soft information of the upper and the lower branches at the same node cannot be updated in parallel. Therefore, the latency of SCD is $(2N-2)$ times of T_n. On the other hand, a decoder with BPD estimates all N nodes of LLR information in parallel at each stage, and LLR information in two directions, i.e., L and R need to be estimated. Therefore, the required latency for the BPD with the conventional RTS is $(2\log_2 N \cdot n_t)$ times of T_n. When using SS, the decoder can update the information in both directions at the same time, so the latency of the decoder can be reduced to n_t times of T_n. In addition, the latency of the proposed hybrid method is $(2log_2 N \cdot 2 + (n_t - 2))$ times of T_n.

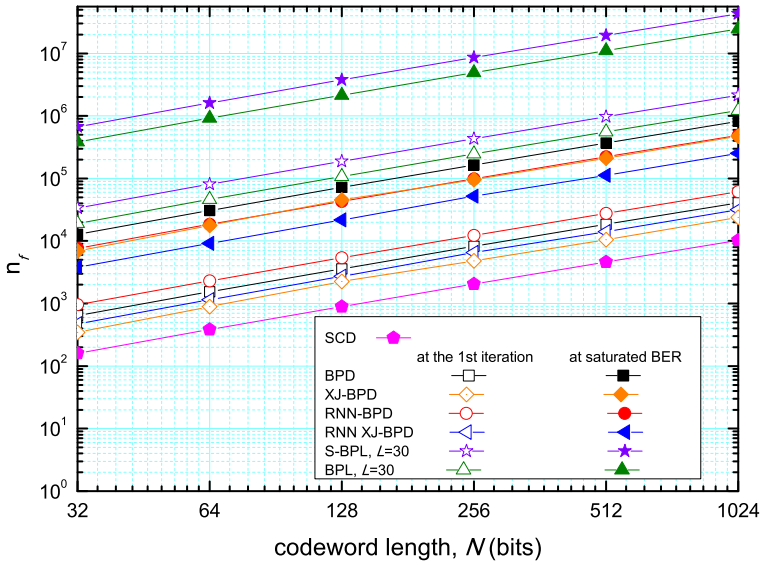


Figure 12. Complexity comparisons in terms of n_f according the codeword length.

Figure 13 shows a comparison of decoding latency according to the scheduling method. The proposed hybrid method offers a faster decoding speed even with higher n_t, and it produces exactly the same performance as the conventional RTS method, as confirmed in Figure 7. As the codeword length increases, SCD requires exponentially increasing decoding latency, while BPD requires almost constant decoding time, showing a strong advantage when implementing it into hardware.

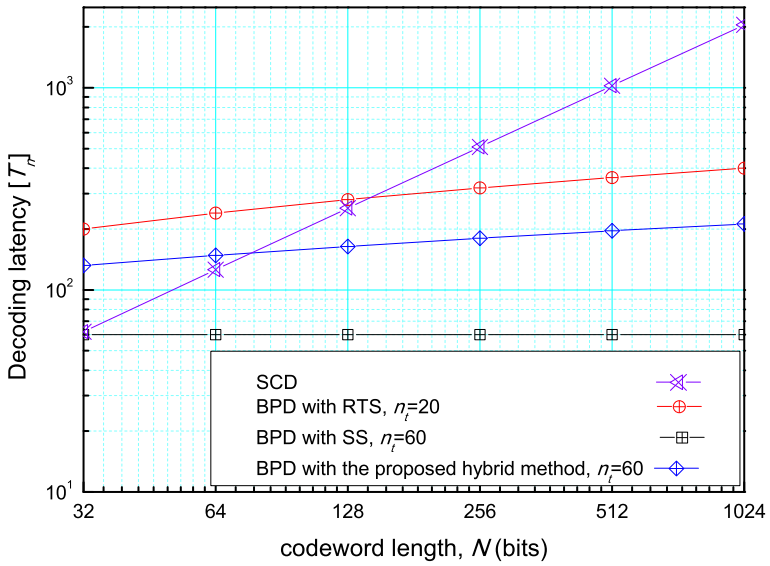


Figure 13. Comparison of decoding latency.

5. Conclusions and Discussion on Future Works

This paper reviewed state-of-the-art decoding methods for polar codes, especially focusing on BPD. We suggested various methods to improve the performance of BPD and speed up the decoding, including efficient BPDL, a complexity reduction method, and neural network-assisted BP decoding algorithms. The performance results found that both BPDL and neural network-assisted BP decoding algorithms could effectively improve the decoding performance. In addition, this paper also investigated the reduction in decoding complexity by removing unnecessary constituent codes, and the use of different scheduling schemes to speed up decoding.

Author Contributions: magentaConceptualization, S.K. and M.Z.; methodology, L.Z.; validation, S.K., M.Z., S.C. and L.Z.; formal analysis, L.Z., S.C.; data curation, L.Z.; writing—original draft preparation, L.Z.; writing—review and editing, S.K., M.Z.; supervision, S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2021R1A2C1003121), and research funds of Jeonbuk National University in 2022.

Data Availability Statement: Not applicable.

Acknowledgments: The authors extend their appreciation to the Yangzhou University Graduate International Academic Exchange Fund Project and the Young Backbone Teachers Project of Yangzhou University.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AWGN	Additive white Gaussian noise
BER	Bit error rate
BPD	Belief propagation decoding
BPDL	Belief propagation decoding with list
BPSK	Binary phase shift keying
DNN	Deep neural network
FEC	Forward error correction
FLOP	Floating point operations
LDPC	Low-density parity-check
LLR	Log-likelihood ratio
PE	Processing element
ResNet-BP	Residual neural network-based belief propagation
RNN	Recurrent neural network
RNN-BPD	RNN-based belief propagation decoding
RNN XJ-BPD	RNN based express-journey belief propagation decoding
RTS	Round-trip scheduling
S-BPDL	Scalable belief propagation decoding with list
SC	Successive cancellation
SCD	Successive cancellation decoding
SS	Segmented scheduling
URLLC	Ultra reliable and low latency communications
XJ-BPD	Express-journey belief propagation decoding
XOR	Exclusive-or

References

1. Arıkan, E. A performance comparison of polar codes and Reed-Muller codes. *IEEE Commun. Lett.* **2008**, *12*, 447–449. [CrossRef]
2. 3GPP. Multiplexing and Channel Coding (Release 10) 3gpp ts 21.101 v10.4.0. 2018. Available online: http://www.3gpp.org/ftp/Specs/2018-09/Rel-10/21-series/21101-a40.zip (accessed on 12 December 2022).

3. Arıkan, E. Channel polarization: A method for constructing capacity achieving codes for symmetric binary-input memoryless channels. *IEEE Trans. Inf. Theory* **2009**, *55*, 3051–3073. [CrossRef]
4. Leroux, C.; Raymond, A.J.; Sarkis, G.; Gross, W.J. A semi-parallel successive-cancellation decoder for polar codes. *IEEE Trans. Signal Process.* 2013, *61*, 289–299. [CrossRef]
5. Elkelesh, A.; Ebada, M.; Cammerer, S.; Ten Brink, S. Belief propagation decoding of polar codes on permuted factor graphs. In Proceedings of the 2018 IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain, 15–18 April 2018; pp. 1–6.
6. Elkelesh, A.; Ebada, M.; Cammerer, S.; Ten Brink, S. Belief propagation list decoding of polar codes. *IEEE Commun. Lett.* **2018**, *22*, 1536–1539. [CrossRef]
7. Li, B.; Bai, B.; Zhu, M.; Zhou, S. Improved Belief Propagation List Decoding for Polar Codes. In Proceedings of the 2020 IEEE International Symposium on Information Theory (ISIT), Los Angeles, CA, USA, 21–26 June 2020; pp. 1–6.
8. Ranasinghe, V.; Rajatheva, N.; Latva-aho, M. Partially Permuted Multi-Trellis Belief Propagation for Polar Codes. In Proceedings of the 2020 IEEE International Conference on Communications (ICC 2020), Dublin, Ireland, 7–11 June 2020; pp. 1–6.
9. Wu, W.; Zhai, Z.; Siegel, P.H. Improved Hybrid RM-Polar Codes and Decoding on Stable Permuted Factor Graphs. In Proceedings of the 2021 11th International Symposium on Topics in Coding (ISTC), Montreal, QC, Canada, 30 August–3 September 2021; pp. 1–5.
10. Dai, L.; Huang, L.; Bai, Y.; Liu, Y.; Liu, Z. CRC-Aided Belief Propagation with Permutated Graphs Decoding of Polar Codes. In Proceedings of the 2020 IEEE 3rd International Conference on Electronic Information and Communication Technology (ICEICT), Shenzhen, China, 13–15 November 2020; pp. 445–448.
11. Li, L.; Liu, L. Belief Propagation with Permutated Graphs of Polar Codes. *IEEE Access* **2020**, *8*, 17632–17641. [CrossRef]
12. Ren, Y.; Shen, Y.; Zhang, Z.; You, X.; Zhang, C. Efficient Belief Propagation Polar Decoder with Loop Simplification Based Factor Graphs. *IEEE Trans. Veh. Technol.* **2020**, *69*, 5657–5660. [CrossRef]
13. Doan, N.; Hashemi, S.A.; Mondelli, M.; Gross, W.J. On the decoding of polar codes on permuted factor graphs. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–6.
14. Xu, J.; Che, T.; Choi, G. XJ-BP: Express Journey Belief Propagation Decoding for Polar Codes. In Proceedings of the 2015 IEEE Global Communications Conference (GLOBECOM), San Diego, CA, USA, 6–10 December 2015; pp. 1–6.
15. Nachmani, E.; Marciano, E.; Lugosch, L.; Gross, W.; Burshtein, D.; Be'ery, Y. Deep Learning Methods for Improved Decoding of Linear Codes. *IEEE J. Sel. Top. Signal Process.* **2018**, *12*, 119–131. [CrossRef]
16. Xu, W.; Wu, Z.; Ueng, Y.-L.; You, X.; Zhang, C. Improved polar decoder based on deep learning. In Proceedings of the IEEE International Workshop on Signal Processing Systems (SiPS), Lorient, France, 3–5 October 2017; pp. 1–6.
17. Huang, Y.; Zhang, M.; Dou, Y. A Low-Complexity Residual Neural Network based BP Decoder for Polar Codes. In Proceedings of the 2020 International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 21–23 October 2020; pp. 889–893.
18. Teng, C.-F.; Chen, C.-H.; Wu, A.-Y. An Ultra-Low Latency 7.8–13.6 pJ/b Reconfigurable Neural Network-Assisted Polar Decoder with Multi-Code Length Support. In Proceedings of the 2020 IEEE Symposium on VLSI Circuits, Honolulu, HI, USA, 16–19 June 2020; pp. 1–2.
19. Teng, C.F.; Wu, C.H.D.; Ho, A.K.S.; Wu, A.Y.A. Low-complexity Recurrent Neural Network-based Polar Decoder with Weight Quantization Mechanism. In Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2019), Brighton, UK, 12–17 May 2019; IEEE: Piscataway, NJ, USA, 2019.
20. Yuan, B.; Parhi, K.K. Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders. *IEEE Trans. Signal Process.* **2014**, *62*, 6496–6506. [CrossRef]
21. Park, Y. S.; Tao, Y.; Sun, S.; Zhang, Z. A 4.68 gb/s belief propagation polar decoder with bit-splitting register file. In Proceedings of the 2014 Symposium on VLSI Circuits Digest of Technical Papers, Honolulu, HI, USA, 10–13 June 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1–2.
22. Abbas, S.M.; Fan, Y.; Chen, J.; Tsui, C.-Y. High-throughput and energy-efficient belief propagation polar code decoder. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2017**, *25*, 1098–1111. [CrossRef]