EURASIP Journal on Wireless Communications and Networking

RESEARCH

Open Access

Flexible soft-output decoding of polar codes

Sunghoon Lee[1], Jooyoun Park[1], Il-Min Kim[2] and Jun Heo[1*]

*Correspondence: junheo@korea.ac.kr
[1] The School of Electrical Engineering, Korea University, 145, Anam-ro, Seongbuk-gu, Seoul 02841, Republic of Korea
Full list of author information is available at the end of the article

Abstract

In this research, we study soft-output decoding of polar codes. Two representative soft-output decoding algorithms are belief propagation (BP) and soft cancellation (SCAN). The BP algorithm has low latency but suffers from high computational complexity. On the other hand, the SCAN algorithm, which is proposed for reduced complexity of soft-output decoding, achieves good decoding performance but suffers from long latency. These two algorithms are suitable only for two extreme cases that need very low latency (but with high complexity) or very low complexity (but with high latency). However, many practical systems may need to work for the moderate cases (i.e., not too high latency and not too high complexity) rather than two extremes. To adapt to the various needs of the systems, we propose a very flexible soft-output decoding framework of polar codes. Depending on which system requirement is most crucial, the proposed scheme can adapt to the systems by controlling the level of parallelism. Numerical results demonstrate that the proposed scheme can effectively adapt to various system requirements by changing the level of parallelism.

Keywords: Polar codes, Soft-output decoding, Parallelization

1 Introduction

The development of polar codes by E. Arikan [1, 2] was a breakthrough in coding theory. Polar codes have been proven to achieve the capacity of symmetric binary-input discrete memoryless channels with an explicit construction. A length-N polar code can be efficiently decoded by a hard-output successive cancellation (SC) decoder. The successive cancellation list (SCL) decoder [3] was later introduced achieving the maximum likelihood bound for a sufficiently large list size L at the expense of increased complexity due to the nature of the list decoding. Furthermore, concatenating a high rate outer code such as cyclic redundancy check (CRC) [3–5] or parity-check [6] considerably improved the performance of polar codes. However, the sequential nature of the SC decoders limits the throughput of implementations. To increase the throughput of SC decoders, estimating simultaneously certain redundant decoding steps in SC decoding, called simplified successive cancellation [7], and several works based on that approach [8, 9] were proposed.

Another interesting decoding method of polar codes is a belief propagation (BP) decoding which was originally introduced in [1]. Although the BP decoder has insufficient performance comparing to the hard-output decoder and high computational complexity [10], it has been widely researched thanks to the high potential of parallel implementation [11–14]. However, a number of redundant computations are required in the BP decoding, which results in high computational complexity.

To address high complexity of the BP algorithm, many methods have been studied to reduce computational complexity [15], and the soft cancellation (SCAN) decoding [16] is one of them. Following the serial message update schedule of the SC decoding, the SCAN algorithm has much lower computational complexity compared with the BP algorithm. However, at the expense of decreased complexity, the SCAN algorithm suffers from long decoding latency and low throughput.

In soft-output decoding of polar codes, the SCAN and BP algorithms can be considered as two extreme cases. With these two algorithms, the system can work well *only* for two cases: (i) when very low latency is demanded at the expense of very high complexity, or (ii) when very low complexity is demanded at the expense of very high latency. In practice, however, some soft-output decoders may require moderate latency and moderate computational complexity. Furthermore, such requirements might be time varying, depending on the time-varying demands of the system. It is *not* possible to dynamically cope with such change of system requirements with the SCAN or BP algorithms, because each of the two algorithms is fixedly (and permanently) tailored for each of two extremes.

The soft-output decoding algorithm *adaptable* to system requirements has not been studied yet to the best of our knowledge. In this work, we propose a decoding algorithm that can work effectively for any scenarios of system requirements including two extreme cases and any moderate (in-between) cases. The system requirements of the proposed algorithm can be flexibly adjusted by controlling the level of parallelism.

The remainder of this paper is organized as follows. We first present the soft-output decoding and perform the extrinsic information transit (EXIT) analysis for measuring the convergence latency. We construct the proposed decoding scheme by representing the polar code as the concatenated codes in which the outer codes are processed by the SCAN decoding in parallel, and the inner code is processed by the BP decoding. The convergence behavior with the PEXIT analysis shows how latency of the proposed scheme varies with the level of parallelism. It is also shown that the expense of the decreased latency is the increased complexity.

2 Background

With the construction method in [1], a polar code is defined by trio: codeword length $N = 2^n$, message length K, and an information set $\mathcal{A} \subset [N]$ of cardinality K. Let $u_0^{N-1} = (u_0, \dots, u_{N-1})$ and $x_0^{N-1} = (x_0, \dots, x_{N-1})$ denote the data and codeword, respectively. The ith component of u_0^{N-1} is set to zero for all $i \in [N] \setminus \mathcal{A}$. Then, $x_0^{N-1} = u_0^{N-1}G$, where G is the generator matrix of a polar codes (defined in [1]). Bit u_i is referred to as an information bit if $i \in \mathcal{A}$ or a frozen bit if $i \notin \mathcal{A}$, and the frozen bits are set to 0. The received vector is denoted by $y_0^{N-1} = (y_0, \dots, y_{N-1})$. The codeword x_0^{N-1} is transmitted through the channel and the received vector y_0^{N-1} is processed by the decoder.

2.1 Fundamentals of soft-output decoding

The soft-output decoding is performed over the factor graph, which is a graphical representation of the generator matrix interconnecting variable nodes (VNs) and check nodes (CNs). The factor graph is constructed from a protograph which serves as a blueprint. Figure 1 shows an instance of the factor graph for a rate-1/2 polar code with $N = 8$ and the protograph. In Fig. 1a, the rightmost gray-boxed (resp. white-boxed) VNs represent frozen bits (resp. information bits).

The factor graph is composed of $N(n+1)$ distinguishable nodes, divided into $n+1$ layers indexed by $\lambda \in \{0, \ldots, n\}$. Each layer is composed of 2^{λ} groups indexed by $\phi \in \{0, \ldots, 2^{\lambda}-1\}$, and each group is composed of $2^{n-\lambda}$ nodes indexed by $\omega \in \{0, \ldots, 2^{n-\lambda}-1\}$. Therefore, all the nodes in the factor graph can be identified by the trio (λ, ϕ, ω). Node (λ, ϕ, ω) has two associated logarithmic likelihood ratio (LLR) messages $L_{\lambda}(\phi, \omega)$ and $B_{\lambda}(\phi, \omega)$ which are passed to the right and the left, respectively. LLR updates on the protograph shown in Fig. 1**b** can be calculated as follows:

$$L_{\lambda}(\phi,\omega) = L_{\lambda-1}(\psi,2\omega) \boxplus [L_{\lambda-1}(\psi,2\omega+1) + B_{\lambda}(\phi,\omega+1)], \tag{1}$$

$$L_{\lambda}(\phi+1,\omega) = L_{\lambda-1}(\psi,2\omega+1) + [L_{\lambda-1}(\psi,2\omega) \boxplus B_{\lambda}(\phi,\omega)], \tag{2}$$

$$B_{\lambda}(\psi,2\omega) = B_{\lambda}(\psi,\omega) \boxplus [B_{\lambda}(\phi+1,\omega) + L_{\lambda-1}(\psi,2\omega+1)], \tag{3}$$

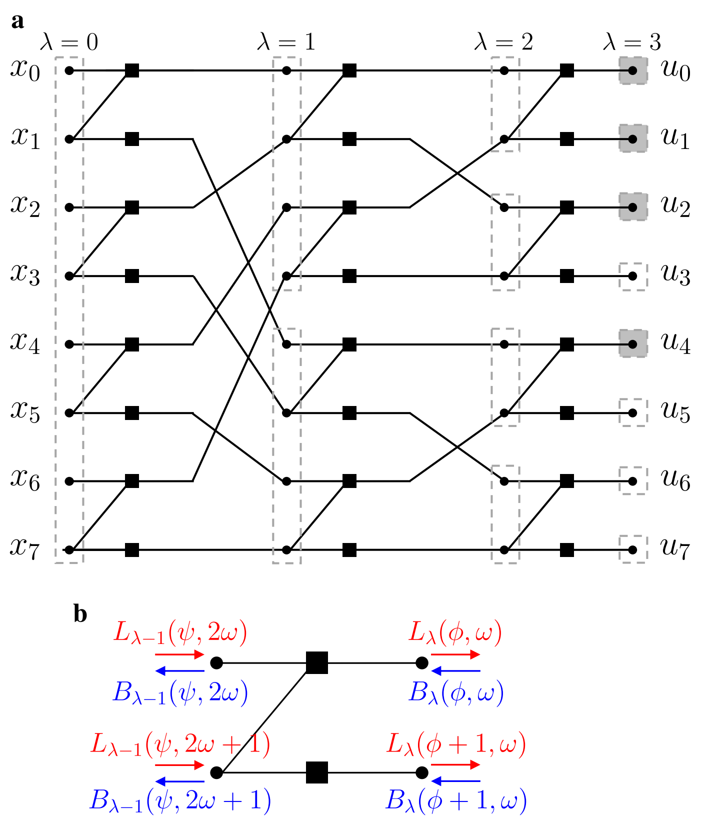
a
$\lambda = 0$
$\lambda = 1$
$\lambda = 2$
$\lambda = 3$
x_0
x_1
x_2
x_3
x_4
x_5
x_6
x_7
u_0
u_1
u_2
u_3
u_4
u_5
u_6
u_7
b
$L_{\lambda-1}(\psi,2\omega)$
$B_{\lambda-1}(\psi,2\omega)$
$L_{\lambda}(\phi,\omega)$
$B_{\lambda}(\phi,\omega)$
$L_{\lambda-1}(\psi,2\omega+1)$
$B_{\lambda-1}(\psi,2\omega+1)$
$L_{\lambda}(\phi+1,\omega)$
$B_{\lambda}(\phi+1,\omega)$

Fig. 1 **a** Factor graph for a rate-1/2 polar code with $N = 8$. **b** Protograph of a polar code where $\psi = \lfloor \phi/2 \rfloor$

$$B_\lambda(\psi+1,2\omega+1) = B_\lambda(\phi+1,\omega) + [B_\lambda(\phi,\omega) \boxplus L_{\lambda-1}(\psi,2\omega)], \tag{4}$$

where $\boxplus$ is defined as

$$\alpha \boxplus \beta \triangleq 2\tanh^{-1}\left[\tanh\left(\frac{\alpha}{2}\right)\tanh\left(\frac{\beta}{2}\right)\right]. \tag{5}$$

2.2 Existing scheduling for soft-output decoding

There are many types of scheduling for soft-output decoding. First, we discuss the BP-based scheduling. Messages on the flooding BP algorithm [13] are updated in parallel, from $\lambda = n$ to $\lambda = 0$. For each layer, L-messages and B-messages of each layer are updated simultaneously. Another scheduling is the round-trip BP [10, 17] which separately computes L-messages and B-messages. The updates in each layer are separated to two phases. In the first phase, B-messages are updated from $\lambda = n-1$ to $\lambda = 0$. In the second phase, L-messages are updated from $\lambda = 1$ to $\lambda = n$. In the BP algorithm, the latency is very low; however, the computational complexity is high.

The other well-known scheduling is the SCAN algorithm where messages are updated by serial message updating schedule used in the SC decoding. The SCAN algorithm is implemented iteratively by omitting hard decisions on the data sequence. In the SCAN algorithm, the computational complexity is low; however, the latency is high.

2.3 EXIT analysis

For the convergence analysis of iterative decoders, the EXIT chart was introduced as a novel tool because of their simplicity and accuracy [18, 19]. The EXIT chart can also be used for code design [20]. The EXIT chart analyzes exchanges of the average extrinsic mutual information of VNs and CNs and tell when the decoding converges. The protograph-based EXIT (PEXIT) analysis [21] is a modified version of the EXIT analysis. In contrast to the EXIT analysis which only treats average values, every mutual information of node is considered in the PEXIT analysis.

Let $J(\sigma)$ denote the mutual information between a binary random variable X with $\Pr(X = +\mu) = 1/2$ and $\Pr(X = -\mu) = 1/2$, and a continuous Gaussian random variable Y with mean X and variance $\sigma^2 = 2\mu$. $J(\sigma)$ is given by [18]

$$J(\sigma) = 1 - \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(\xi-\sigma^2/2)^2}{2\sigma^2}} \cdot \log_2\left(1+e^{-\xi}\right) d\xi. \tag{6}$$

Consider a VN with degree d_v and a CN with degree d_c. Let $I_{Ev|g}$ (resp. $I_{Ec|g}$) be the mutual information between the g-th output message of the VN (resp. CN) and the associated codeword bit. For an additive white Gaussian noise (AWGN) channel, the EXIT function of the PEXIT analysis for the g-th message is given by [21]

$$I_{Ev|g} = J\left(\sqrt{\sum_{k=1,k\neq g}^{d_v}\left[J^{-1}(I_{Av|k})\right]^2 + \left[J^{-1}(I_{ch})\right]^2}\right), \tag{7}$$

$$I_{Ec|g} \simeq 1 - J\left(\sqrt{\sum_{k=1,k\neq g}^{d_c}\left[J^{-1}(1 - I_{Ac|k})\right]^2}\right), \tag{8}$$

where $I_{Av|k}$ (resp. $I_{Ac|k}$) is *a priori* mutual information related to the message received by the VN (resp. CN) on its k-th edge and I_{ch} is the channel mutual information. The convergence is declared if each $I_{\text{APP}}(j)$, which is the mutual information between the *a posteriori* LLR evaluated by a VN and an associated codeword bit x_j, reaches 1 as the iteration number increases.

3 Proposed scheme

The recursive structure makes polar codes be considered as generalized concatenated codes and the SC decoding can be interpreted as an instance of multistage decoding [22]. We consider a length-N polar code as a concatenated code of S length-(N/S) outer codes with a length-N inner code where $S = 2^s$ denotes the number of outer codes. Figure 2 shows two different concatenated codes for a length-8 polar code. Each outer code is processed by the SCAN decoder in parallel, and an inner code is decoded in the reverse order of the round-trip BP.

The proposed decoding consists of three phases. The first phase is to update L-messages of an inner code based on the inputs of channel LLRs. In the second phase, the messages in outer codes are updated according to the SCAN schedule using the L-messages updated in the first phase as inputs. Each outer code updates the messages in parallel. The last phase is updating B-messages of an inner code using the output LLRs of outer codes.

In the framework, the SCAN algorithm is viewed as the proposed scheme with $s = 0$, which is the serialized extreme, and the round-trip BP algorithm is viewed as the proposed scheme with $s = n - 1$, which is the parallelized extreme. Thus, the framework provides the explicit scheduling of soft-output polar decoding containing both cases as two extremes. In the proposed framework, we can gradually change the updating schedule from the most serial way to the most parallel way. By controlling s, we can determine how much the decoding is parallelized. Therefore, we call s the level of parallelism. The proposed scheme is described in Algorithm 1.

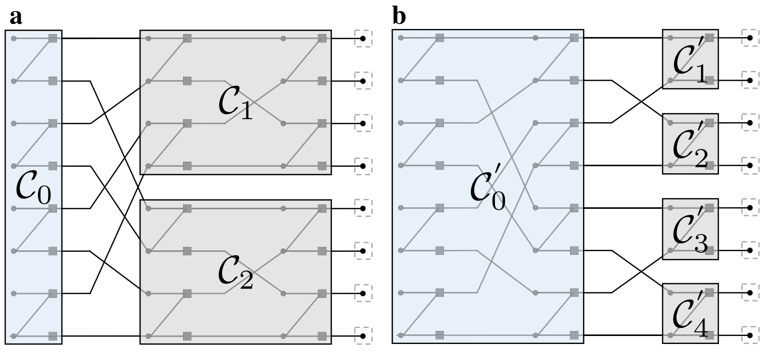

Fig. 2 Polar code with $N = 8$ can be considered as two concatenated codes: **a** Two length-4 outer codes $\mathcal{C}_1$ and $\mathcal{C}_2$ with the inner code $\mathcal{C}_0$. **b** Four length-2 outer codes $\mathcal{C}_1'$, $\mathcal{C}_2'$, $\mathcal{C}_3'$, and $\mathcal{C}_4'$ with the inner code $\mathcal{C}_0'$

```
Algorithm 1: Proposed scheme
Input: the level of parallelism $s$, channel LLRs
Output: $\{L_n(i,0)\}_{i=0}^{(N-1)}$ and $\{B_0(j,0)\}_{j=0}^{(N-1)}$
1  $L_0(0,\omega) \leftarrow$ channel LLRs, $\forall\omega \in \{0,\dots,N-1\}$
2  $B_n(\phi,0) \leftarrow \infty, \forall\phi \in \mathcal{A}$
3  Set all the uninitialized LLRs to 0
4  for $l = 1,\dots,l_{max}$ do
5      for $\lambda = 0,\dots,s-1$ do
6          Update $L_{\lambda+1}(\phi,\omega)$ using (1) and (2)
7      for $\mathcal{C}_i, \forall i \in \{1,\dots,s\}$ do in parallel
8          Update $L_{\lambda+1}(\phi,\omega)$ and $B_{\lambda+1}(\phi,\omega)$ in each inner code using (1)–(4) based on the
           serial message updating schedule
9      for $\lambda = s-1,\dots,0$ do
10         Update $B_{\lambda+1}(\phi,\omega)$ using (3) and (4)
11 for $i = 0,\dots,N-1$ do
12     if $B_n(i,0) + L_n(i,0) \geq 0$ then $\hat{u}_i \leftarrow 0$
13     else $\hat{u}_i \leftarrow 1$
```

For example, in Fig. 2a, the SCAN algorithm makes one to update the output LLRs of $\mathcal{C}_0$ as the inputs LLRs of $\mathcal{C}_1$ first. After the decoding of $\mathcal{C}_1$, a decoder of $\mathcal{C}_0$ uses the output LLRs of $\mathcal{C}_1$ as input. The same goes on $\mathcal{C}_2$. Finally, LLRs for deciding codeword bits are updated in $\mathcal{C}_0$ and one iteration is concluded. Figure 2b shows a different concatenation of a length-8 polar code. The input LLRs of the decoder for outer codes, $\{L_s(\phi,\omega)\}$, are updated in lines 5 and 6 of Algorithm 1. The decoding of outer codes is performed in lines 7 and 8 in parallel. The decoding of outer codes is the same as the process of the SCAN decoding. The output LLRs of the outer code is updated in lines 9 and 10.

We also study the convergence behavior obtained based on the PEXIT analysis to measure latency. The convergence is declared if each $I_{\text{APP}}(j)$, which is the mutual information between the *a posteriori* LLR evaluated by a VN and an associated codeword bit x_j, reaches 1 as the iteration number increases. To analyze the convergence behavior, we track the $\min_{\forall j\in\mathcal{A}} I_{\text{APP}}(j)$. We can evaluate $I_{\text{APP}}(j)$ as:

$$I_{\text{APP}}(j) = J\left(\sqrt{J^{-1}\left(I_n^L(0,j)\right)^2 + J^{-1}\left(I_n^B(0,j)\right)^2}\right). \tag{9}$$

We analyze the mutual information passed to the right and the left, which are denoted by $I_\lambda^L(\phi,\omega)$ and $I_\lambda^B(\phi,\omega)$, respectively. Mutual information, paired with LLR messages denoted by the same (λ,ϕ,ω) on the protograph in Fig. 1b, can be calculated based on EXIT functions [21]. Calculations of mutual information are as follows:

$$\begin{aligned} I_\lambda^L(\phi,\omega) =& 1 - J\Bigg(\sqrt{J^{-1}\left(1 - I_{\lambda-1}^L(\psi,2\omega)\right)^2 +} \\ & \overline{J^{-1}\left(1 - J\left(\sqrt{J^{-1}\left(I_{\lambda-1}^L(\psi,2\omega+1)\right)^2 + J^{-1}\left(I_\lambda^B(\phi+1,\omega)\right)^2}\right)\right)^2}\Bigg), \end{aligned} \tag{10}$$

$$\begin{aligned} I_\lambda^L(\phi+1,\omega) =& J\Bigg(\sqrt{J^{-1}\big(I_{\lambda-1}^L(\psi,2\omega+1)\big)^2+}\\ &\overline{J^{-1}\Bigg(1-J\bigg(\sqrt{J^{-1}\big(I_{\lambda-1}^L(\psi,2\omega)\big)^2+J^{-1}\big(I_\lambda^B(\phi,\omega)\big)^2}\bigg)\Bigg)^2}\Bigg), \end{aligned} \tag{11}$$

$$\begin{aligned} I_{\lambda-1}^B(\psi,2\omega) =& 1-J\Bigg(\sqrt{J^{-1}\big(1-I_\lambda^B(\phi,\omega)\big)^2+}\\ &\overline{J^{-1}\Bigg(1-J\bigg(\sqrt{J^{-1}\big(I_\lambda^B(\phi+1,\omega)\big)^2+J^{-1}\big(I_{\lambda-1}^L(\psi,2\omega+1)\big)^2}\bigg)\Bigg)^2}\Bigg), \end{aligned} \tag{12}$$

$$\begin{aligned} I_{\lambda-1}^B(\psi,2\omega+1) =& J\Bigg(\sqrt{J^{-1}\big(I_\lambda^B(\phi+1,\omega)\big)^2+}\\ &\overline{J^{-1}\Bigg(1-J\bigg(\sqrt{J^{-1}\big(I_\lambda^B(\phi,\omega)\big)^2+J^{-1}\big(I_{\lambda-1}^L(\psi,2\omega)\big)^2}\bigg)\Bigg)^2}\Bigg). \end{aligned} \tag{13}$$

The process of tracking the minimum $I_{\mathrm{APP}}(j)$ is described in Algorithm 2. The process of Algorithm 2 is very similar to Algorithm 1. First, initialize each parameter in lines 1 to 3. Update input mutual information of the decoders for outer codes, $I_{\lambda+1}^L$, in lines 5 and 6 of Algorithm 2. The decoding of outer codes is performed in lines 7 and 8 in parallel. The output mutual information of the outer code is updated in lines 9 and 10. Then, calculate $I_{APP}(j)$ using (1). If the value of $I_{APP}(j)$ is 1 for all js, then it is considered the time when the decoding ends and the number l is set as the convergence iteration number l_s. We can obtain l_s which is the number of iteration when the proposed scheme with s converges.

Algorithm 2: Convergence of the proposed scheme

```
Input: the level of parallelism $s$
Output: the convergence iteration number $l_s$
1  $I_0^L(0,\omega) \leftarrow J\left(\sqrt{8RE_b/N_0}\right), \forall\omega \in \{0,\dots,N-1\}$
2  $I_n^B(\phi,0) \leftarrow 1, \forall\phi \in \mathcal{A}$
3  Set all the uninitialized mutual information to 0
4  while $l < l_{max}$ do
5      for $\lambda = 0,\dots,s-1$ do
6          Update $I_{\lambda+1}^L(\phi,\omega)$ using (10) and (11)
7      for $\mathcal{C}_i, \forall i \in \{1,\dots,s\}$ do in parallel
8          Update $I_\lambda^L(\phi,\omega)$ and $I_\lambda^B(\phi,\omega)$ in each inner code using (10)–(13) based on the serial
             message updating schedule
9      for $\lambda = s-1,\dots,0$ do
10         Update $I_{\lambda+1}^B(\phi,\omega)$ using (12) and (13)
11     for $j = 0,\dots,N-1$ do
12         Calculate $I_{\mathrm{APP}}(j)$ using (9)
13     if $\min_{\forall j\in\mathcal{A}} I_{APP}(j) == 1$ then
14         $l_s \leftarrow l$, $l \leftarrow l_{max}$
15     else
16         $l \leftarrow l+1$
```

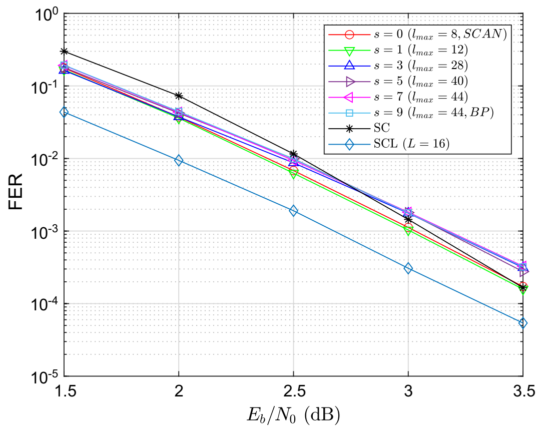

Fig. 3 FER comparison between the proposed algorithm, SCAN, BP, and SC for $N = 1024$ and $R = 1/2$. The polar code is constructed using Gaussian approximation [22] optimized for $E_b/N_0 = 2.0$ dB

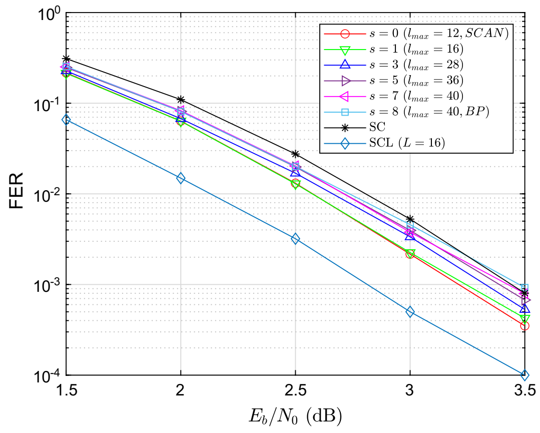

Fig. 4 FER comparison between the proposed algorithm, SCAN, BP, and SC for $N = 512$ and $R = 1/2$. The polar code is constructed using Gaussian approximation [22] optimized for $E_b/N_0 = 2.0$ dB

4 Methods

Simulations were performed over an AWGN channel with binary phase shift keying(BPSK) modulation. The polar codes with codeword length $N = 512, 1024$, and $32, 768$ and code rate $R = 1/2$ which are constructed using Gaussian approximation [22] optimized for $E_b/N_0 = 2.0$ dB for $N = 512$ and 1024 and $E_b/N_0 = 1.0$ dB for $N = 32, 768$ were used to evaluate the performance.

5 Results and discussion

5.1 Error correction performance

In Figs. 3 and 4, we have plotted the frame error rate (FER) curves of the proposed scheme for six different values of s. The proposed scheme is also compared to two

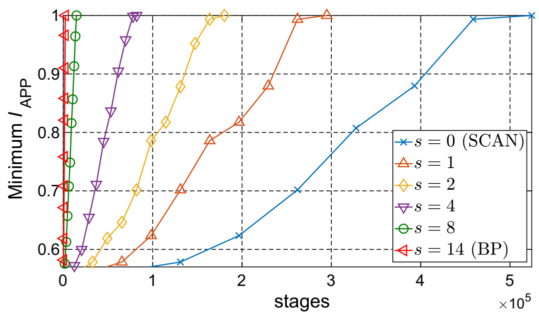

Fig. 5 Convergence behavior of the minimum I_{APP} using different decoding scheduling for $N = 32,768$, $R = 1/2$, and $E_b/N_0 = 1.0$ dB

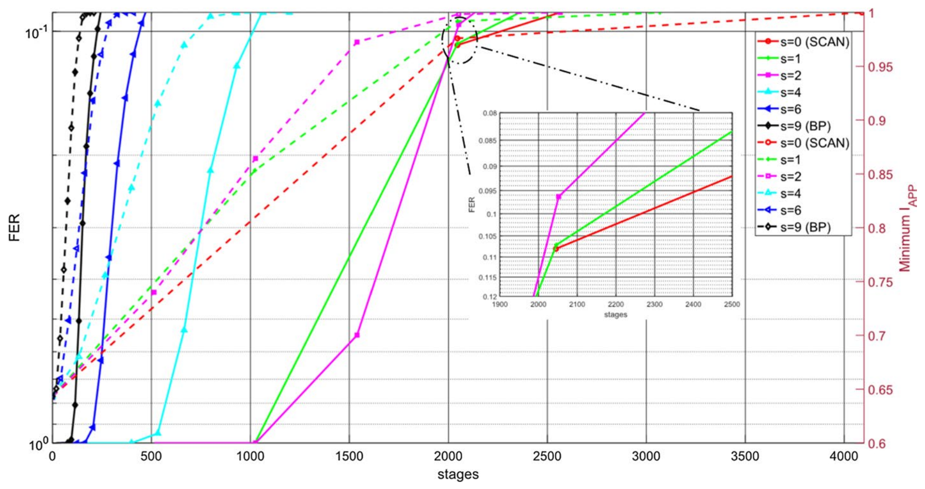

Fig. 6 FER and convergence behavior of the minimum I_{APP} using different decoding scheduling for $N = 1024$ with code rate $R = 1/2$. The polar code is constructed using Gaussian approximation optimized for $E_b/N_0 = 2.0$ dB (FER : solid line, Minimum I_{APP} : dashed line)

extreme cases of the SCAN and BP algorithms with 10^6 frames. The number of iterations l_{max} for each simulation is chosen to be $4l_s$.

The proposed scheme with $s = 1$ and $s = 3$ performs similar to the SCAN (i.e., $s = 0$) and BP (i.e., $s = 9$) decoding, respectively. The proposed scheme with $s = 5$ and $s = 7$ performs almost the same as the BP decoding. Generally speaking, the performance of the proposed algorithm spans approximately from that of the SCAN decoding to that of the BP decoding.

5.2 Latency

We assume that decoders for length-N polar codes have N processing units with each capable of implementing (10)–(13) in one stage, where the stage denotes the required number of serial message updates.

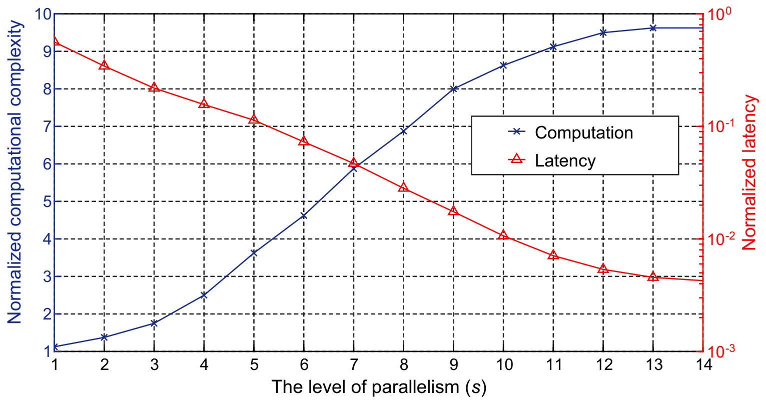

Fig. 7 Normalized latency and computational complexity for $N = 32,768$ when the proposed decoding converges. (The values are respectively normalized by their corresponding values of the SCAN algorithm.)

The SCAN decoder requires $(2N - 3)$ stages to terminate the message updates at each iteration.

The stages for the inner code of the proposed scheme are twice the number of the layers in the inner code. The stages for the outer codes are the same as the SCAN decoding of an N/S-length polar code. Thus, the total number of stages of the proposed scheme, T_s, is given by

$$T_s = \left(2s + \frac{2N}{S} - 3\right) l_s. \tag{14}$$

The convergence behavior is obtained over the same environment except that codeword length is now 32,768. The convergence trajectories for I_{APP} are plotted in Fig. 5. Furthermore, in Fig. 6, mutual information is tracked by PEXIT, and FER is tracked by Monte Carlo simulation. Two curves are not exactly matched but the order of latency (based on stage as we defined) with different value of s is the same for both PEXIT and simulation results. In this curve, minimum $I_{APP} = 1$ and FER=10^{-1} at $E_b/N_0 = 2$dB when $N = 1024$ were used for the standards for successful decoding, respectively. From the convergence behavior, we confirm that the proposed scheme with the higher level of parallelism has the lower latency.

5.3 Complexity

Each iteration of the proposed scheme and the SCAN algorithm has the same computational complexity because each message is updated only once. Therefore, the computational complexity of the proposed decoding is only dominated by the number of iterations.

Figure 7 shows the values of latency and computational complexity of the proposed scheme with $N = 32,768$ that are respectively normalized by their corresponding values of the SCAN algorithm (i.e., $s = 0$). Each point shows a normalized value when the decoding converges as the level of parallelism s increases. The proposed decoding flexibly decreases latency compared to the SCAN algorithm at the cost of the computational complexity.

6 Conclusion

In this work, we propose the framework of soft-output polar decoding adaptable to system requirements. In the proposed framework, decoding of a polar code is divided into parallel SCAN decoding of outer codes and round-trip BP decoding of one inner code. By intelligently controlling the level of parallelism, various system requirements can be adaptively satisfied by the proposed algorithm. Thus, the proposed algorithm can work effectively for any scenarios of system requirements including two extreme cases and any moderate (in-between) cases. Numerical results show that the proposed scheme can change latency and computational complexity as a trade-off. This flexibility renders the proposed scheme adaptable to various needs of the systems.

Abbreviations
AWGN: Additive white Gaussian noise; BP: Belief propagation; CN: Check node; CRC: Cyclic redundancy check; EXIT: Extrinsic information transit; PEXIT: Protograph-based extrinsic information transit; SC: Successive cancellation; SCAN: Soft cancellation; SCL: Successive cancellation list; VN: Variable node.

Authors' contributions
SL designed the main of the algorithm, analyzed the data, and wrote this paper. JP made important revisions to the manuscript. I-MK co-designed methods and gave important advice. JH gave valuable suggestions on the idea of this paper. All authors read and approved the final manuscript.

Funding
This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2020-2015-0-00385) supervised by the IITP(Institute for Information & communications Technology Promotion). This work was supported by the ICT R&D program of MSIT/IITP. [2021-0-01810, Development of elemental technologies for Ultra-secure Quantum Internet.

Availability of data and materials
All data generated or analyzed during this study are included in this paper.

Declarations

Competing interests
The authors declare that they have no competing interests.

Author details
[1]The School of Electrical Engineering, Korea University, 145, Anam-ro, Seongbuk-gu, Seoul 02841, Republic of Korea.
[2]Department of Electrical and Computer Engineering, Queen's University, 99 University Avenue, Kingston K7L 3N6, Canada.

Received: 2 May 2019 Accepted: 11 August 2021
Published online: 24 August 2021

References

1. E. Arikan, Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. IEEE Trans. Inf. Theory **55**(7), 3051–3073 (2009)
2. E. Arikan, Polar codes: a pipelined implementation, in *International Symposium on Broadband Communications (ISBC)* (2010)
3. I. Tal, A. Vardy, List decoding of polar codes. IEEE Trans. Inf. Theory **61**(5), 2213–2226 (2015)
4. K. Niu, K. Chen, CRC-aided decoding of polar codes. IEEE Commun. Lett. **16**(10), 1668–1671 (2012)
5. B. Li, H. Shen, D. Tse, An adaptive successive cancellation list decoder for polar codes with cyclic redundancy check. IEEE Commun. Lett. **16**(12), 2044–2047 (2012)
6. T. Wang, D. Qu, T. Jiang, Parity-check-concatenated polar codes. IEEE Commun. Lett. **20**(12), 2342–2345 (2016)
7. A. Alamdar-Yazdi, F.R. Kschischang, A simplified successive-cancellation decoder for polar codes. IEEE Commun. Lett. **15**(12), 1378–1380 (2011)
8. H. Yoo, I.-C. Park, Efficient pruning for successive-cancellation decoding of polar codes. IEEE Commun. Lett. **20**(12), 2362–2365 (2016)

9. G. Sarkis, P. Giard, A. Vardy, C. Thibeault, W.J. Gross, Fast list decoders for polar codes. IEEE J. Sel. Areas Commun. **34**(2), 318–328 (2016)
10. Y.S. Park, Y. Tao, S. Sun, Z. Zhang, A 4.68 gb/s belief propagation polar decoder with bit-splitting register file, in: *2014 Symposium On VLSI Circuits Digest of Technical Papers*. (IEEE, 2014), pp. 1–2
11. E. Arikan, A performance comparison of polar codes and reed-muller codes. IEEE Commun. Lett. **12**(6), 447–449 (2008)
12. A. Eslami, H. Pishro-Nik, On finite-length performance of polar codes: stopping sets, error floor, and concatenated design. IEEE Trans. Commun. **61**(3), 919–929 (2013)
13. N. Hussami, S. B. Korada, R. Urbanke, Performance of polar codes for channel and source coding, in *IEEE International Symposium On Information Theory, 2009. ISIT 2009*. (IEEE, 2009), pp. 1488–1492
14. Y. Zhang, A. Liu, X. Pan, Z. Ye, C. Gong, A modified belief propagation polar decoder. IEEE Commun. Lett. **18**(7), 1091–1094 (2014)
15. A. Elkelesh, M. Ebada, S. Cammerer, S. ten Brink, Improving Belief Propagation decoding of polar codes using scattered EXIT charts, in *2016 IEEE Information Theory Workshop (ITW)* (2016)
16. U.U. Fayyaz, J.R. Barry, Low-complexity soft-output decoding of polar codes. IEEE J. Sel. Areas Commun. **32**(5), 958–966 (2014)
17. J. Xu, T. Che, G. Choi, XJ-BP: Express journey belief propagation decoding for polar codes, in *Global Communications Conference (GLOBECOM), 2015 IEEE*. (IEEE, 2015), pp. 1–6
18. S. Ten Brink, Convergence behavior of iteratively decoded parallel concatenated codes. IEEE Trans. Commun. **49**(10), 1727–1737 (2001)
19. S. Ten Brink, G. Kramer, A. Ashikhmin, Design of low-density parity-check codes for modulation and detection. IEEE Trans. Commun. **52**(4), 670–678 (2004)
20. J.H. Shin, K. Noh, W. Sung, J. Heo, Simple and accurate design of low-density parity-check codes for multi-input multi-output systems. Wirel. Pers. Commun. **62**(4), 923–936 (2012)
21. G. Liva, M. Chiani, Protograph ldpc codes design based on exit analysis, in *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*. (IEEE, 2007), pp. 3250–3254
22. P. Trifonov, Efficient design and decoding of polar codes. IEEE Trans. Commun. **60**(11), 3221–3227 (2012)

Publisher's Note