Construction of parity-check-concatenated polar codes based on minimum Hamming weight codewords

J. Park, I. Kim and H.-Y. Song✉

In this Letter, a new parity-check-concatenated (PCC) polar code construction that considers the number of minimum Hamming weight (MHW) codewords is proposed. The parity bits to reduce the number of MHW codewords as much as possible is successively constructed. The proposed construction can reduce the number of MHW codewords more than other codes. The results show that the proposed codes can outperform the other codes at a high signal-to-noise region.

Introduction: After Arikan [1] proposed the polar codes and successive cancellation (SC) decoding, Tal and Vardy [2] improved the performance using a SC list (SCL) decoding and cyclic-redundancy-check (CRC) aided SCL (CA-SCL) decoding. Li *et al.* [3] found that the CRC codes effectively eliminate the minimum Hamming weight (MHW) codewords of polar codes. Recently, an optimal CRC construction for polar codes was proposed in [4]. The authors optimised CRC codes to minimise the number of MHW codewords (NMHC). In [5], polar codes of a new type, parity-check-concatenated (PCC) polar codes, were proposed. The authors concatenated single-parity-check codes instead of CRC codes. They constructed the PCC polar codes with a (burst-error-based) heuristic technique and showed that the PCC polar codes could outperform (standard) CRC-concatenated polar codes (CRC polar codes). Because the CRC codes are a subclass of parity-check codes, the PCC polar codes have more optimisation potential than the CRC polar codes [5].

In this Letter, we successively construct the parity-check bits to reduce NMHC as much as possible. To the best of the authors' knowledge, this is the first construction of PCC polar codes that considers MHW codewords. Simulation results show that the proposed PCC polar codes can further reduce the number of MHW codewords, and they outperform the other polar codes.

CRC polar codes: We briefly review the CRC polar codes [1, 2, 4]. We denote $\boldsymbol{u} = (u_1, \ldots, u_N)$ as an N bit input vector of the encoder. The input vector $\boldsymbol{u}$ is composed of M message bits, P CRC parities, and N-M-P zeros. A CRC encoder appends P parities to the end of message vector $\boldsymbol{m} = (m_1, \ldots, m_M)$, and then a CRC codeword $\boldsymbol{w} = (w_1, \ldots, w_K)$ is produced, where $K = M + P$. The bits of $(w_1, \ldots, w_K)$ and N-K zeros are placed into $\boldsymbol{u}$ according to the index set F and F^c, where F and F^c are the index sets of the frozen and unfrozen bits, respectively. The function $\boldsymbol{u} = \pi(\boldsymbol{w})$ represents the bit mapping according to F and F^c. A codeword is generated as $\boldsymbol{x} = (x_1, \ldots, x_N) = \boldsymbol{uG}$, where $\boldsymbol{G}$ is the generator matrix of the polar code without a bit-reversal operation.

The CA-SCL decoding is performed using the received vector $\boldsymbol{y} = (y_1, \ldots, y_N)$ and the number of list L. We denote $\hat{u}_i(l)$ and $\hat{u}_1^i(l)$ as an estimate of the ith input bit of the lth list and the estimated bit sequence (path) from $\hat{u}_1(l)$ to $\hat{u}_i(l)$, respectively. The decoder successively calculates the path metric $\mathrm{PM}_i(l)$ of $\hat{u}_1^i(l)$. Using $\mathrm{PM}_{i-1}(l)$ and $\hat{u}_1^{i-1}(l), l = 1, \ldots, L$, the decoder calculates the path metrics for $2L$ candidates of $\hat{u}_i(l) = 0$ or 1. It should be noted that if $i \in F$, then only $\hat{u}_i(l) = 0$, as one candidate. For each l, the two path metrics of the candidates are obtained by

$$\begin{cases} \mathrm{PM}_{i-1}(l) & \text{if } \hat{u}_i(l) = h\big(\alpha_i(l)\big) \\ \mathrm{PM}_{i-1}(l) - |\alpha_i(l)| & \text{otherwise,} \end{cases} \tag{1}$$

where

$$\alpha_i(l) = \log \frac{\Pr[\boldsymbol{y}, \hat{u}_1^{i-1}(l) | u_i = 0]}{\Pr[\boldsymbol{y}, \hat{u}_1^{i-1}(l) | u_i = 1]} \quad \text{and} \quad h(\alpha_i) = \begin{cases} 0 & \text{if } \alpha_i > 0 \text{ or } i \in F \\ 1 & \text{otherwise} \end{cases}$$

We take only the L most reliable paths among the $2L$ candidates and update $\hat{u}_1^i(l)$ and $\mathrm{PM}_i(l)$ using these L paths. After $\hat{u}_1^N(l)$ and $\mathrm{PM}_N(l)$ are obtained, we find the most reliable path that satisfies the CRC test. If there is no such a satisfied path, then decoding is failed.

Zhang *et al.* [4] optimised the CRC codes to reduce NMHW. They showed that the optimised CRC codes could improve the performance at a high signal-to-noise ratio (SNR) region. Even though the performance could be improved by the optimised CRC codes, it could be further improved by relaxing the CRC constraint. This will be discussed in a later section.

PCC polar codes: In this section, we introduce the PCC polar codes in [5]. The PCC polar codes exploit P parity-check codes instead of CRC codes. For $j = 1, \ldots, P$, we denote c_j and $p_j \in F^c$ as the jth parity bit and its bit index in $\boldsymbol{u}$, respectively. A set T_j is a subset of F^c, and all of its elements are smaller than p_j. Each parity bit c_j is obtained by $c_j = \sum_{i \in T_j} u_i \mathrm{mod}\, 2$, and u_{p_j} becomes c_j. Fig. 1 depicts an example of $\boldsymbol{u}$ containing one parity-check bit, where the grey and white squares represent frozen and unfrozen bits, respectively. In this example, the parity bit c_1 is obtained using message bits of $T_1 = \{4, 8\}$, and it is placed at u_{11}, where $p_1 = 11$. We note that T_1 is a subset of F^c, and all of the elements in T_1 are smaller than $p_1 = 11$.

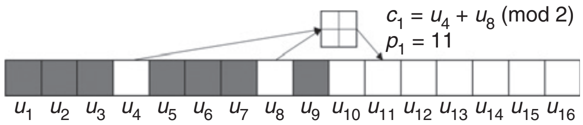

Fig. 1 *Example of one parity-check bit* c_1

The decoding of the PCC polar codes is almost the same as the CA-SCL decoding. The only difference is that the decoder checks P parity-check bits in the middle of the decoding, instead of using the CRC check at the end of the decoding. When the decoder updates the path metric of one parity bit, i.e. $i = p_j$, the decoder decides that $\hat{c}_j(l) = \sum_{i' \in T_j} \hat{u}_{i'}(l) \mathrm{mod}\, 2$, and $\hat{u}_i(l)$ is fixed as $\hat{c}_j(l)$. Then, the path metric is updated by (1), where $h(\alpha_i(l))$ is replaced by $\hat{c}_j(l)$. It should be noted that this is almost the same as the path metric update of the frozen bit, except the bit is decided by $\hat{c}_j(l)$.

Wang *et al.* [5], also proposed a heuristic construction technique. First, they investigated the bitwise error probability of F^c, and found P burst-error blocks. Then, they constructed the parity bits p_j as follows. *Principle 1:* Each burst-error block has one parity bit. *Principle 2:* Every element in T_j should be selected from a different burst-error block. *Principle 3:* An unfrozen bit channel with high error probability has a higher priority to be selected as an element of T_j. Moreover, there are no duplicated elements in every T_j, where $j = 1, \ldots, P$.

The results in [5] show that the PCC polar codes can outperform the (standard) CRC polar codes. In the simulation results section, we will show that the burst-error-based construction can reduce NMHC more than the (standard) CRC polar codes. We note that the construction does not consider the MHW, and hence there is room for improvement in terms of the number of MHW codewords.

Proposed construction of PCC polar codes: In this section, we propose a construction of PCC polar codes based on the MHW codewords. The main objective of this construction is reducing NMHC as much as possible. Because finding a set of optimal single-parity-check codes without any constraint requires huge complexity, we construct the parity-check codes one by one in each of the blocks.

Let us denote b_j, $j = 1, \ldots, P$, as the last index of P blocks in $\boldsymbol{w} = (w_1, \ldots, w_K)$. Each block $(w_1, \ldots, w_{b_j})$ is defined by $b_{j+1} - b_j = \tau$ for $j = 1, \ldots, P-1$, where $b_0 = 0$, $b_P = K$, and $\tau = \lfloor K/P \rfloor$. It should be noted that every block starts from w_1. Let us assume that an all-zero codeword is transmitted. We denote $A_d = \{\boldsymbol{x} | \boldsymbol{x} \in \mathbb{C}, \mathrm{wt}(\boldsymbol{x}) = d\}$ as a set of codewords with d Hamming weight, where $\mathbb{C}$ is the codebook of the (PCC) polar code and the function $\mathrm{wt}(\boldsymbol{x})$ returns the Hamming weight of the vector $\boldsymbol{x}$. Its corresponding message vectors that produce the codewords of A_d are denoted by $V_d = \{\boldsymbol{w} | \boldsymbol{u} = \pi(\boldsymbol{w}), \boldsymbol{x} = \boldsymbol{uG}, \boldsymbol{x} \in A_d\}$. Let us define two functions as $f(\boldsymbol{w}, \boldsymbol{a}, j) = |\{i | w_i = a_i = 1, i \leq b_j\}|$ and $\mathrm{wt}(\boldsymbol{w}, j) = |\{i | w_i = 1, i \leq b_j\}|$. We denote $d_{\min}$ as the non-zero MHW. The pseudo code of the proposed construction algorithm is described in the following.

Input: $\mathbf{G}$, $\mathbf{G}$, F, $b_1, \ldots, b_p$
for $j = 1$: P **do**
  Generate a PCC polar code using $\boldsymbol{G}$, $p_{j'}$, and $T_{j'}$, for $j' < j$
  Find $d_{\min}$, construct $A_{d_{\min}}$ and $V_{d_{\min}}$
  $\boldsymbol{a} = (a_1, \ldots, a_K) \leftarrow \mathbf{0}_K$ and $\boldsymbol{s} = (s_1, \ldots, s_K) \leftarrow \mathbf{0}_K$
  continue ← true
  while continue = true **do**

8: **if** $\boldsymbol{a} = \mathbf{0}_K$ **then**
9: $W = \{\boldsymbol{w} | \boldsymbol{w} \in V_{d_{\min}}, \mathrm{wt}(\boldsymbol{w}, j) - \mathrm{wt}(\boldsymbol{w}, j-1) > 0\}$
10: **else**
11: $W = \{\boldsymbol{w} | \boldsymbol{w} \in V_{d_{\min}}, f(\boldsymbol{w}, \boldsymbol{a}, j) = 0, \mathrm{wt}(\boldsymbol{w}, j) > 0\}$
12: **end if**
13: **if** $W = \emptyset$ and $\boldsymbol{a} \neq \mathbf{0}_K$ **then**
14: continue $\leftarrow$ false
15: **else if** $W = \emptyset$ and $\boldsymbol{a} = \mathbf{0}_K$ **then**
16: $s_i \leftarrow 1$ with probability α for $i \leq b_j$
17: $\boldsymbol{a} \leftarrow \boldsymbol{s}$
18: continue $\leftarrow$ false
19: **else**
20: $\boldsymbol{v} \leftarrow \arg\min_{\boldsymbol{w} \in W} \mathrm{wt}(\boldsymbol{w}, j)$
21: $i \leftarrow$ last non-zero index of $(v_1, \ldots, v_{b_j})$
22: $s_i \leftarrow 1$
23: $\boldsymbol{a} \leftarrow \boldsymbol{a} + \boldsymbol{v}$
24: **end if**
25: **end while**
26: **if** $\mathrm{wt}(\boldsymbol{s}) = 1$ and $\mathrm{wt}(\boldsymbol{a})$ is an odd number larger than 2 **then**
27: $\boldsymbol{s} \leftarrow \boldsymbol{a}$
28: **else if** $\mathrm{wt}(\boldsymbol{s}) = 1$ **then**
29: $s_i \leftarrow 1$ with probability α for $i \leq b_j$, if $a_i = 0$
30: **end if**
31: $p_j \leftarrow$ last non-zero index of $\boldsymbol{s}$
32: $T_j \leftarrow \{i | s_i = 1, i \leq b_j, i \neq p_j\}$
33: **end for**

The key idea of the proposed algorithm is that if there is no vector with 1 at the same position in a vector set, all of the non-zero vectors in the set can be removed from the codebook of one single-parity-check code. In other words, if every vector in a subset of $V_{d_{\min}}$ has disjoint supports, then the subset can be removed from $V_{d_{\min}}$ by one parity bit, where the support means an index set of non-zero elements in a given vector. Let $\mathcal{W}$ be a set of vectors $\boldsymbol{v}$ composing $\boldsymbol{a}$ in line 20. Because we only place one 1 from each vector in $\mathcal{W}$ into $\boldsymbol{s}$, it is always true that $\boldsymbol{v} \cdot \boldsymbol{s} = 1$ for all $\boldsymbol{v} \in \mathcal{W}$. Then, every vector in $\mathcal{W}$ can be removed from $V_{d_{\min}}$ (and the corresponding vectors in $A_{d_{\min}}$) by the parity bit constructed by $\boldsymbol{s}$. To make the size of $\mathcal{W}$ as large as possible, we add the lowest weight vectors first. It should be noted that we have applied $\mathrm{wt}(\boldsymbol{w}, j) - \mathrm{wt}(\boldsymbol{w}, j-1) > 0$ in line 9 to uniformly distribute the parity bits in $b_{j-1} < p_j \leq b_j, j = 1, \ldots, P$. If $\boldsymbol{a}$ is the only single element in $\mathcal{W}$ and it has an odd weight >2, then $\boldsymbol{a}$ becomes $\mathbf{s}$, because $\boldsymbol{a} \cdot \boldsymbol{a} = 1$. For the other cases, we construct $\mathbf{s}$ randomly by inserting 1 with probability α. For $\boldsymbol{a} \cdot \boldsymbol{s} = 1$, we have considered the positions i of $a_i = 0$ in line 29. Throughout this Letter, $\alpha = 0.5$ is applied.

Simulation results: For the simulations, $M/N = 1/2$, $N = 2^7, \ldots, 2^{10}$, and binary phase shift keying over additive white Gaussian channel are considered. The sets F and F^c are obtained using the Gaussian approximation method with $E_b/N_0 = 1.5$ dB [6]. We exploit the standard CRC codes (S.CRC) in [4]. We optimise the CRC codes (O.CRC) using the method in [4]. The burst-error-based PCC polar codes (B.PCC) are constructed using the heuristic construction in [5]. The bitwise error probabilities are obtained from the simulation results of the SCL decoding with $L = 32$, at $E_b/N_0 = 1.5$ dB. The proposed MHW-based PCC polar codes (H.PCC) are constructed with $L = 10{,}000$. Every A_d is obtained using the method in [3].

The investigated $|A_d|$ values are given in Table 1. We omit some results of different numbers of parity bits P, because they did not have the best performance in our simulations. The table shows that the B.PCC and O.CRC have fewer MHW codewords than the S.CRC. The table also shows that the H.PCC have further reduced NMWC. The effect of the reduced $|A_d|$ can be found in Fig. 2, where $L = 32$ is applied. In the figure, the proposed codes show the best frame error rate (FER) performance for every N. For $N = 128$, the H.PCC outperforms the other codes in the entire SNR region. For the other lengths, the performance gain is increased at a higher SNR. This result is agreed with the reduced NMHC in Table 1. Because we have reduced NMHC, we can obtain a performance gain at a higher SNR. We note that the PCC polar codes require more parity bits P than O.CRC, and hence the PCC polar codes should overcome a rate loss of inner polar code. Even though the B.PCC has less NMHC than O.CRC, they cannot outperform the O.PCC in our simulations. It suggests that we should carefully construct the PCC polar codes to obtain the best performance.

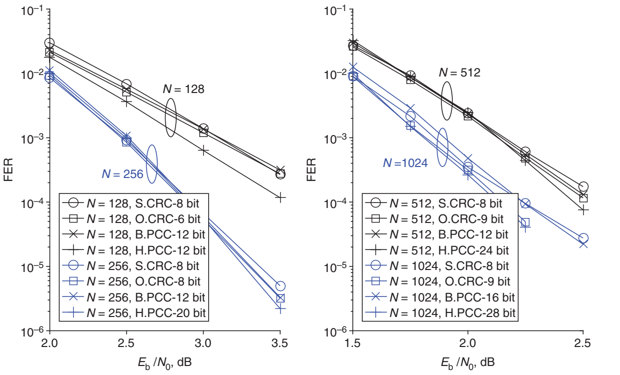

Fig. 2 *FER performances of S.CRC, O.CRC, B.PCC, and H.PCC polar codes*

Table 1: Number of codewords $|A_d|$ for $d = 8$, 12, 16

N	Type (P)	L	$\lvert A_8 \rvert$	$\lvert A_{12} \rvert$	$\lvert A_{16} \rvert$
128	S.CRC (8)	40,000	131	92	88
	O.CRC (6)	40,000	105	0	211
	B.PCC (12)	40,000	19	0	654
	H.PCC (12)	40,000	0	0	2256
256	S.CRC (8)	40,000	84	0	675
	O.CRC (8)	40,000	75	0	761
	B.PCC (12)	40,000	0	0	231
	H.PCC (20)	40,000	0	0	227
512	S.CRC (8)	40,000	20	0	1329
	O.CRC (9)	40,000	20	0	1305
	B.PCC (12)	40,000	0	0	213
	H.PCC (24)	40,000	0	0	56
1024	S.CRC (8)	30,000	0	0	1939
	O.CRC (9)	30,000	0	0	1812
	B.PCC (16)	30,000	0	0	135
	H.PCC (28)	30,000	0	0	54

Concluding remarks: In this Letter, we propose a MHW-based construction of PCC polar codes. It constructs the parity bits to reduce NMHC as much as possible. The results show that the proposed construction reduces NMHC. Consequently, the FER performance is improved at a high SNR.

Finally, we comment that the proposed algorithm is also heuristic and does not guarantee the optimal performance. Finding an optimal construction algorithm will be an important future research topic.

Acknowledgment: This work was supported by Institute for Information and Communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (No. B0717-16-0024, Development on the core technologies of transmission, modulation and coding with low-power and low-complexity for massive connectivity in the IoT environment).

Submitted: *21 March 2017* E-first: *26 May 2017*
doi: 10.1049/el.2017.1037
One or more of the Figures in this Letter are available in colour online.

J. Park, I. Kim and H.-Y. Song (*Yonsei University, Seoul, South Korea*)

✉ E-mail: hysong@yonsei.ac.kr

References

1 Arkan, E.: 'Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels', *IEEE Trans. Inf. Theory*, 2009, **55**, (7), pp. 3051–3073
2 Tal, I., and Vardy, A.: 'List decoding of polar codes', *IEEE Trans. Inf. Theory*, 2015, **61**, (5), pp. 2213–2226
3 Li, B., Shen, H., and Tse, D.: 'An adaptive successive cancellation list decoder for polar codes with cyclic redundancy check', *IEEE Commun. Lett.*, 2012, **16**, (12), pp. 2044–2047
4 Zhang, Q., Liu, A., Pan, X., and Pan, K.: 'CRC code design for list decoding of polar codes', *IEEE Commun. Lett.*, 2017, accepted for publication, doi: 10.1109/LCOMM.2017.2672539
5 Wang, T., Qu, D., and Jiang, T.: 'Parity-check-concatenated polar codes', *IEEE Commun. Lett.*, 2016, **20**, (12), pp. 2342–2345
6 Trifonov, P.: 'Efficient design and decoding of polar codes', *IEEE Trans. Commun.*, 2012, **60**, (11), pp. 3221–3227