APPROXIMATE BELIEF PROPAGATION DECODER FOR POLAR CODES

Menghui Xu[1,2,3], *Shusen Jing*[1,2,3], *Jun Lin*[4], *Weikang Qian*[5], *Zaichen Zhang*[2,3], *Xiaohu You*[2], *and Chuan Zhang*[1,2,3,*]

[1]Lab of Efficient Architectures for Digital-communication and Signal-processing (LEADS)
[2]National Mobile Communications Research Laboratory, Southeast University
[3]Quantum Information Center of Southeast University, Nanjing, China
[4]School of Electronic Science and Engineering, Nanjing University, Nanjing, China
[5]University of Michigan-Shanghai Jiao Tong University Joint Institute, Shanghai, China
Email: [2]{mhxu, shs.jing, zczhang, xhyu, chzhang}@seu.edu.cn, [4]jlin@nju.edu.cn, [5]qianwk@sjtu.edu.cn

ABSTRACT

Polar code is increasing its popularity recently for its capacity-achieving property for B-DMCs. However, when designing decoders for polar code, it has always been an inevitable concern for us to balance the decoding performance and the hardware consumption. In this paper, we propose an approximate belief propagation (BP) decoder for polar code for the first time. By introducing the approximate computation schemes, we reduced the critical path delay (CPD) and the hardware consumption of the conventional BP decoders. Simulation results show that the proposed approximate BP decoder achieves nearly the same decoding performance as the conventional one. Advantages of the proposed decoder has been verified by FPGA implementation.

Index Terms— Polar code, belief propagation, approximate computation, hardware consumption.

1. INTRODUCTION

Polar code, proposed by Arıkan's breakthrough paper [1], has received significant attentions from both academia and industry. It has been proved to be the first capacity-achieving code for binary-input discrete memory-less channels (B-DMCs). Two main decoding algorithms for polar code are successive cancellation (SC) decoding (or SC list, SCL decoding) and belief propagation (BP) decoding. The SC algorithm decodes bits in serial schedule with low complexity, but suffers from high decoding latency. On the other hand, BP decoder works much faster than SC decoder due to its inherent high parallelism and is more popular for implementation. However, facing the performance requirements raised by applications such as 5G wireless communication and Internet of Things (IoT), both decoders have to increase either list size L (for SCL decoding) or iteration number I (for BP decoding). Since the complexities of SCL decoder and BP decoder are of $\mathcal{O}(LN\log N)$ [2–5] and $\mathcal{O}(IN\log N)$ [6–9], respectively, balancing the decoding performance and hardware complexity has always been an inevitable concern for all designers.

However, 100% precision in computation is not always required, especially in some error-resilient systems such as image processing, handwritten recognition, and neuromorphic systems. In addition, decoders of forward error correction (FEC) codes can be viewed as an error-resilient system since they work with noisy channel inputs. It has been shown in [10] that BP decoding can be implemented with noisy circuits. So decoders with approximate computation has been considered to balance decoding performance, hardware cost, and energy dissipation. In prior works, approximate computation has been already implemented for LDPC decoders [11–13].

In this paper, we combined polar BP decoding and approximate computation together and proposed an efficient approximate BP decoder. The remainder of this paper is organized as follows. Section 2 briefly reviews polar codes and BP decoding. Section 3 presents the approximate BP decoder. Design flow for the proposed approximate BP decoder is given in Section 4. Section 5 compares the hardware consumption. Conclusions are drawn in Section 6.

2. REVIEW OF POLAR CODES AND BP DECODING

2.A. Polar Codes

Any polar code can be determined by the parameter set of (N, K, A, u_{A^c}). Here, $N = 2^n$ denotes the code length, K is the number of information bits, A is the set of information bits, A^c is the complementary set of A, and u_{A^c} represents the frozen bits' values. For polar encoding, $x_1^N = (x_1, x_2, ..., x_{n-1}, x_n)$ is constructed based on source vector $u_1^N = (u_1, u_2, ...u_{n-1}, u_n)$ as follows:

$$x_1^N = u_1^N G_N = u_1^N B_N F^{\otimes n}, \tag{1}$$

where G_N and B_N are the generator matrix and bit reversal permutation matrix respectively, and $F^{\otimes n}$ is *Kronecker* power of $F = \left[\begin{smallmatrix}1 & 0\\ 1 & 1\end{smallmatrix}\right]$. For decoding, the estimate of u_1^N is obtained based on the channel output y_1^N. The reader is referred to [1] for more details.

2.B. BP Decoding Algorithm

Polar BP decoding can be illustrated by an n-stage factor graph [14] shown in Fig. (1), which is consisted of two types of nodes: F and G. Each node is labeled with coordinate (i, j), where i indicates the stage number and j indicates the node index in the column. Polar BP decoder updates messages iteratively over this factor graph.

For BP decoding, two types of log likelihood ratio (LLR) messages: left-to-right message L and right-to-left message R, are involved. They are initiated by Eqs. (2) and (3), respectively.

$$L_{n+1,j} = \ln \frac{P(y_j|x_j = 0)}{P(y_j|x_j = 1)}, \tag{2}$$

$$R_{1,j} = \begin{cases} 0, & \text{if } j \in A, \\ \infty, & \text{if } j \in A^c. \end{cases} \tag{3}$$

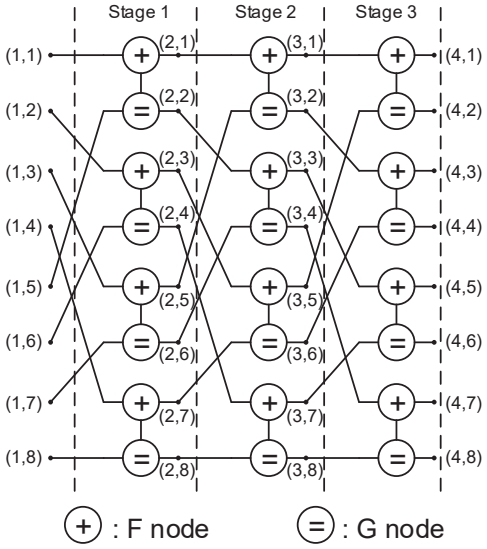

Fig. 1. Factor graph of BP decoding with $N = 8$.

Then messages are passed iteratively from left to right and then from right to left according to Eq. (4).

$$\begin{cases} L_{i,j} = f(L_{i+1,2j-1}, g(L_{i+1,2j}, R_{i,j+N/2})), \\ L_{i,j+N/2} = g(f(R_{i,j}, L_{i+1,2j-1}), L_{i+1,2j}), \\ R_{i+1,2j-1} = f(R_{i,j}, g(L_{i+1,2j}, R_{i,j+N/2})), \\ R_{i+1,2j} = g(f(R_{i,j}, L_{i+1,2j-1}), R_{i,j+N/2}); \end{cases} \tag{4}$$

where

$$f(x, y) = x + y, \tag{5}$$

$$g(x, y) \approx \operatorname{sgn}(x) \operatorname{sgn}(y) \min(|x|, |y|). \tag{6}$$

After I iterations, the j-th bit is estimated according to:

$$\hat{u}_j = \begin{cases} 0 & \text{if } R_{1,j} \geq 0, \\ 1 & \text{else.} \end{cases} \tag{7}$$

3. PROPOSED APPROXIMATE BP POLAR DECODER

3.A. Quantization Schemes

Before giving details of BP polar decoders, we need to decide the quantization scheme first. For $(64, 32)$ polar code, we find that the values of LLR mainly distributed in the interval of $[-35, 35]$, which means 1 sign-bit and 5 integer-bits are required at least for quantization in order to achieve satisfying performance.

The fixed point results for $(64, 32)$ polar code with different quantization schemes are shown in Fig. (2). Here, $(s - k - l)$ denote s sign bits, k integer bits and l fractional bits. In Fig. (2), the (1-5-3) quantization scheme achieves a good trade off between performance and complexity, therefore is employed in following.

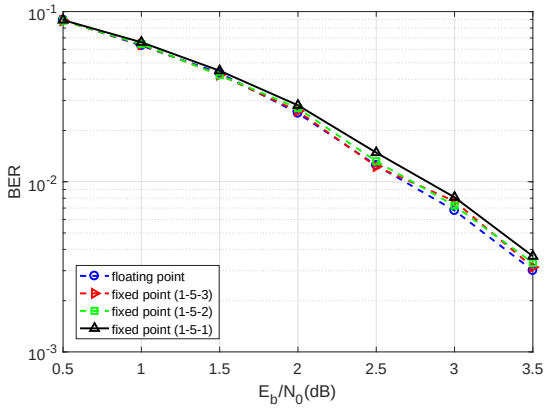

Fig. 2. Performance comparison of different quantization schemes.

3.B. Conventional Architecture for G Node

According to Eq. (6), it is shown that the comparison of absolute value is carried out during BP decoding process. As a result, the LLR messages are usually stored in sign-magnitude form (SMF).

The major computation of G node is magnitude comparison. For the conventional G node, we need to compare the two binary data from MSB to LSB until two different bits are found. However, in some cases, we have to compare almost all the bits to find out which is the bigger one. But in fact, the two input data are about the same, which means whichever we choose may not affect the final result. So the conventional architecture of G node may result in a waste of time and power.

3.C. Proposed Approximate Architecture for G Node

In this section, an approximate architecture for G node is proposed. For approximate G node, we only compare the high-order $n - k$ bits and the rest k bits are ignored. The detail is shown in Fig. (3) with an example of $k = 2$.

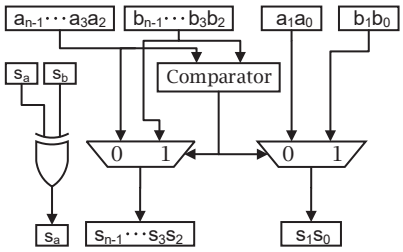

Fig. 3. Proposed approximate architecture for G node.

If $a_{[n-1:k]} \geq b_{[n-1:k]}$, then the approximate G node predicts $s_{[n-1:0]} = b_{[n-1:0]}$. Otherwise we have $s_{[n-1:0]} = a_{[n-1:0]}$. However, when we have $a_{[n-1:k]} = b_{[n-1:k]}$ and the ignored k bits of a is smaller that those of b, we will get a wrong result.

Supposing that a and b are random input numbers with uniform distribution, then the probability of $a_{[n-1:k]} = b_{[n-1:k]}$ and the probability of the $a_{[k-1:0]} < b_{[k-1:0]}$ are derived as follows:

$$\begin{cases} P(a_{[n-1:k]} = b_{[n-1:k]}) = (\frac{1}{2})^{n-k}, \\ P(a_{[k-1:0]} < b_{[k-1:0]}) = \frac{2^k-1}{2^{n+1}}. \end{cases}$$

Hence, the Error Rate (ER) of the proposed approximate G node is considered as follows:

$$ER = (\frac{1}{2})^{n-k} \cdot \frac{2^k-1}{2^{k+1}} = \frac{2^k-1}{2^{k+1}}. \quad (8)$$

According to Eq. (8), for a specific n, a larger k will cause greater performance loss and less hardware consumption. We use half-rate polar codes with code length $N = 64$ for simulation. The quantization scheme is (1-5-3), including 1 sign bit, 5 integer bits, and 3 fractional bits. The result is shown in Fig. (4). With a proper number of ignored bits k, the approximate decoder performs nearly the same as the conventional one with less hardware consumption.

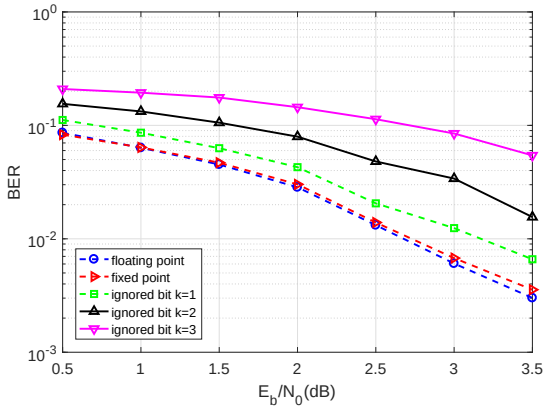

Fig. 4. Results of BP decoders with different ignored bit k.

3.D. Conventional Architecture for F Node

For the updating schedule in F nodes, messages should be transformed into their complement forms before addition or subtraction and then converted back to the SMF after the computation. As shown in Fig. (5), input messages a, b and output message s are stored in the SMF. Let S_a, S_b, S_s represent the sign of a, b and s, respectively. Let M_a, M_b, M_s represent the magnitude of a, b and s, respectively. The F node functions as an adder and subtracter when $S_b = 0$ and $S_b = 1$ respectively. The adding one unit (AOU) adds 1 to its input. The architecture in Fig. (5) suffers from long critical path delay due to data format conversion at the input and output ports.

3.E. Proposed Approximate Architecture for F Node

Notice that the architecture in Fig. (5) suffers from long CPD because of the data format conversion. In addition, the AOU and comparator in the conventional architecture increases the overall hardware consumption and the CPD. So, in this section, an efficient approximate architecture for F node is designed and shown in Fig. (6).

As shown in Fig. (6), similarly, input messages a, b and output message s are also stored in the SMF. Let S_a, S_b, S_s represent the sign of a, b and s, respectively. Let M_a, M_b, M_s represent the magnitude of a, b and s, respectively. For the proposed approximate architecture, subtraction is directly carried out instead of doing data

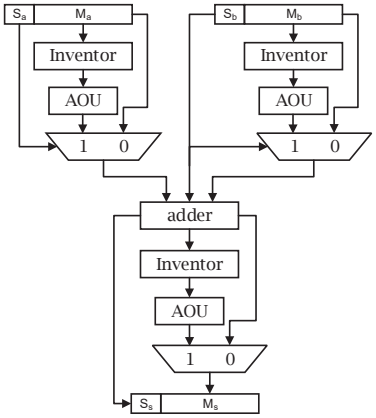

Fig. 5. Conventional architecture for F node.

Table 1. Principles of the proposed approximate F node.

S_a	S_b	Relative Size	S_s	M_s
0	0	-	0	$M_a + M_b$
1	1	-	1	$M_a + M_b$
0	1	$M_a \geq M_b$	0	$M_a - M_b$
0	1	$M_a < M_b$	1	$-(M_a - M_b)$
1	0	$M_a \geq M_b$	1	$M_a - M_b$
1	0	$M_a < M_b$	0	$-(M_a - M_b)$

format conversion before computation. As a result, we only need one data format conversion for F node. $M_a + M_b$ and $M_a - M_b$ are computed at the same time and the magnitude of the final result s is chosen from these two values.

The specific arithmetical operations of F node is illustrated in Table 1. When $S_a \oplus S_b = 0$, the F node implement $M_a + M_b$. Otherwise, it implements $M_a - M_b$. When $M_a \geq M_b$, $S_s = S_a$. Otherwise, $S_s = S_b$.

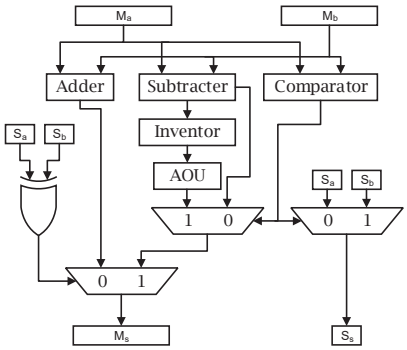

Fig. 6. Proposed approximate architecture for F node.

In the proposed architecture for F node, we also introduce approximate schemes for the add one unit (AOU) since the operation of AOU may not significantly affect the final computing result. We

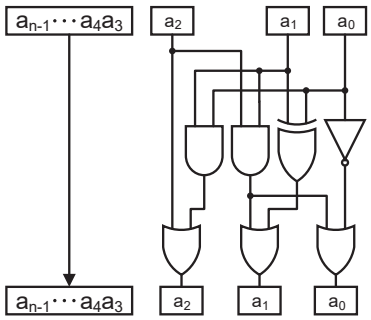

Fig. 7. Proposed approximate AOU with $m = 3$.

only add 1 to the m low-order bits of the input. In addition, the carry out bit of the adding one operation is dropped without being propagated to higher bits. To reduce the overall relative error, these m low-order bits are all set to 1 if the carry out bit is 1. Fig. (7) gives an example of the approximate AOU with $m = 3$.

Similarly, we use half-rate polar codes with code length $N = 64$ for simulation. The quantization scheme is also(1-5-3). The result in Fig. (8) shows that even if when $m = 1$, the proposed approximate BP decoder still achieves almost the same decoding performance as the conventional one, which makes it possible to remove this AOU in the proposed architecture.

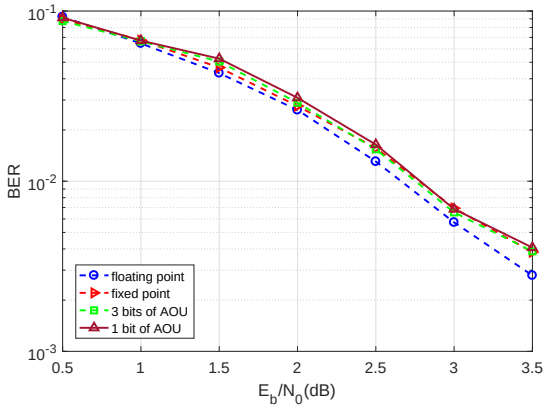

Fig. 8. Simulation results of BP decoders with different bits of AOU.

4. DESIGN FLOW OF APPROXIMATE POLAR BP DECODERS

Although by introducing approximate computation scheme, we reduced the hardware consumption of the conventional BP decoder, however, the decoding performance may suffer from significant degradation without careful design. In this section, we propose a design flow of the approximate polar BP decoder:

Step 1: For a polar BP decoder, we have to find optimal quantization schemes for left-to-right and right-to-left messages. For example, we use (5-3) for half rate polar codes with code length $N = 64$ bits.

Step 2: We introduce approximate computation scheme to the quantized polar BP decoder and redesign the F and G node respectively. By evaluating the simulation result of decoding performance, we make balance between the degree of accuracy and the hardware consumption.

Step 3: We combine all the approximate computation schemes together and adjust the degree of accuracy by the overall decoding performance.

5. HARDWARE IMPLEMENTATION AND COMPARISON

To demonstrate the advantage of the proposed approximate BP decoder, the corresponding implementation results for $(64, 32)$ BP polar decoder are listed in table 2. It is shown that the proposed approximate BP decoder shows good hardware reduction than the accurate one. Compared with the conventional decoder, the arithmetic logic unit (ALUT) reduction of the approximate decoder is 36.3%. Meanwhile, the number of registers reduces 9.0%. However, this kind of hardware reduction has little adverse effect on decoding performance.

Table 2. Different implementations for $(64, 32)$ polar code.

Hardware overheads	Conventional	Approximate	Reduction
ALUT	$97,283$	$61,958$	36.3%
Registers	16214	14751	9.0%

6. CONCLUSION

In this paper, an approximate BP decoder for polar code is proposed. Approximate computation schemes are introduced to alleviate the contradiction between higher throughput and hardware consumption. The flow for designing an approximate polar BP decoder is also proposed. Simulation results show that the proposed approximate computation schemes lead to negligible performance degradation compared with the conventional one, which makes the proposed approximate polar BP decoder highly attractive in the future. Future work will be directed towards incorporate the proposed decoder into our 5G Cloud Platform.

Acknowledgement

This work is supported in part by NSFC under grant 61501116, Jiangsu Provincial NSF under grant BK20140636, Huawei HIRP Flagship under grant YB201504, the Fundamental Research Funds for the Central Universities, the SRTP of Southeast University, State Key Laboratory of ASIC & System under grant 2016KF007, ICRI for MNC, and the Project Sponsored by the SRF for the Returned Overseas Chinese Scholars of MoE.

References

[1] Erdal Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[2] François Leduc-Primeau, Saied Hemati, Warren J Gross, and Shie Mannor, "A relaxed half-stochastic iterative decoder for

LDPC codes," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM)*, 2009, pp. 1–6.

[3] Chuan Zhang, Bo Yuan, and Keshab K Parhi, "Reduced-latency SC polar decoder architectures," in *Proc. IEEE International Conference on Communications (ICC)*, June 2012, pp. 3471–3475.

[4] Chuan Zhang and Keshab Parhi, "Low-latency sequential and overlapped architectures for successive cancellation polar decoder," *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2429–2441, 2013.

[5] Xiao Liang, Chuan Zhang, Menghui Xu, Shunqing Zhang, and Xiaohu You, "Efficient stochastic list successive cancellation decoder for polar codes," in *Proc. IEEE International System-on-Chip Conference (SOCC)*, 2015, pp. 421–426.

[6] Erdal Arıkan, Haesik Kim, Garik Markarian, U Ozgur, and Efecan Poyraz, "Performance of short polar codes under M-L decoding," in *Proc. IEEE ICT-Mobile Summit*, Santander, Spain, Jun. 2009.

[7] Yuanrui Ren, Chuan Zhang, Xing Liu, and Xiaohu You, "Efficient early termination schemes for belief-propagation decoding of polar codes," in *ASIC (ASICON), 2015 IEEE 11th International Conference on*. IEEE, 2015, pp. 1–4.

[8] Junmei Yang, Chuan Zhang, Huayi Zhou, and Xiaohu You, "Pipelined belief propagation polar decoders," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2016, pp. 413–416.

[9] Menghui Xu, Xiao Liang, Chuan Zhang, Zhizhen Wu, and Xiaohu You, "Stochastic bp polar decoding and architecture with efficient re-randomization and directive register," in *Signal Processing Systems (SiPS), 2016 IEEE International Workshop on*. IEEE, 2016, pp. 315–320.

[10] Chu Hsiang Huang, Yao Li, and Lara Dolecek, "Belief propagation algorithms on noisy hardware," *IEEE Transactions on Communications*, vol. 63, no. 1, pp. 11–24, 2015.

[11] Yangcan Zhou, Jun Lin, and Zhongfeng Wang, "Efficient approximate layered LDPC decoder," in *Proc. IEEE International Symposium on Circuits and Systems*, 2017, pp. 1–4.

[12] Pascal Giard, Gabi Sarkis, Claude Thibeault, and Warren J Gross, "A 237 Gbps unrolled hardware polar decoder," *Electronics Letters*, vol. 51, no. 10, 2014.

[13] Pascal Giard, Alexios Balatsoukas-Stimming, Thomas Christoph Mller, Andrea Bonetti, Claude Thibeault, Warren J. Gross, Philippe Flatresse, and Andreas Burg, "PolarBear: A 28 nm FD-SOI ASIC for decoding of polar codes," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. PP, no. 99, pp. 1–1, 2017.

[14] Jin Sha, Xing Liu, Zhongfeng Wang, and Xiaoyang Zeng, "A memory efficient belief propagation decoder for polar codes," *Communications, China*, vol. 12, no. 5, pp. 34–41, 2015.