electronics

MDPI

Article

Design of A Parallel Decoding Method for LDPC Code Generated via Primitive Polynomial

Zhe Zhang, Liang Zhou * and Zhi Heng Zhou

National Key Lab on Communication, University of Electronic Science and Technology of China, Chengdu 611731, China; zzhang@std.uestc.edu.cn (Z.Z.); zhzhou@uestc.edu.cn (Z.H.Z.)
* Correspondence: lzhou@uestc.edu.cn

Abstract: An effective way of improving decoding performance of an LDPC code is to extend the single-decoder decoding method to a parallel decoding method with multiple sub-decoders. To this end, this paper proposes a parallel decoding method for the LDPC codes constructed by m-sequence. In this method, the sub-decoders have two types. The first one contains only one decoding module using the original parity-check constraints to implement a belief propagation (BP) algorithm. The second one consists of a pre-decode module and a decoding module. The parity-check matrices for pre-decode modules are generated by the parity-check constraints of the sub-sequences sampled from an m-sequence. Then, the number of iterations of the BP process in each pre-decode module is set as half of the girth of the parity-check matrix, resulting in the elimination of the impact of short cycles. Using maximum a posterior (MAP), the least metric selector (LMS) finally picks out a codeword from the outputs of sub-decoders. Our simulation results show that the performance gain of the proposed parallel decoding method with five sub-decoders is about 0.4 dB, compared to the single-decoder decoding method at the bit error rate (BER) of 10^{-5}.

Keywords: LDPC; parallel decoding; belief propagation; short cycles

Citation: Zhang, Z.; Zhou, L.; Zhou, Z.H. Design of A Parallel Decoding Method for LDPC Code Generated via Primitive Polynomial. *Electronics* **2021**, *10*, 425. https://doi.org/10.3390/electronics10040425

Academic Editor: Francisco Garcia-Herrero
Received: 14 January 2021
Accepted: 5 February 2021
Published: 9 February 2021

1. Introduction

The m-sequence code, which generates via primitive polynomial, is an LDPC code [1]. The generator of the code is a linear feedback shift register (LFSR), which is suitable for scenarios with limited resources for encoders. Moreover, the m-sequence code is designed with extremely low code rate because it is supposed to work in scenarios with extremely low signal to noise ratio (SNR), such as deep-space communication [2] and wireless sensor network [3]. And It has been proven [1] that the m-sequence code with moderate code length has a better BER performance than the LDPC code constructed by progressive edge growth (PEG) algorithm [4]. Therefore, in such scenarios, improving the bit error ratio (BER) performance of m-sequence code is valuable. And developing decoding method is an effective approach to improve the BER performance, as decoding processes can be done at resource-rich nodes.

The Parallel decoding methods, such as multiple-based belief propagation (MBBP) algorithm [5–7] and modified random redundant decoding (mRRD) algorithm [8,9], have been proven that they can achieve better performance than the single-decoder system for decoding block codes. In this paper, we then apply the parallel decoding method to m-sequence codes.

In MBBP algorithm, candidate codewords are outputted by BP decoding module of multiple sub-decoders with different parity-check matrices. Then, they are selected by a least metric selector (LMS) through maximum a posterior (MAP) rule. Therefore, the construction of parity-check matrices is essential for MBBP. Literature [7] had proposed a method that combining the cycles of parity-check matrix of LDPC codes constructed by progressive edge-growth [4] to generate new parity-check matrices. But for m-sequence code, there is no discussion about the construction of parity-check matrices for sub-decoders.

In mRRD, an automorphic group of code is employed to construct parity-check matrix of sub-decoder. Each sub-decoder uses RRD algorithm to output the candidate codeword. Therefore, the essential point of applying mRRD is finding the automorphic group of an LDPC code. Literature [10] proposed an LDPC code constructed by a cyclic code and the automorphic group of this code is easy to be found. This code had achieved great performance improvement by employing mRRD algorithm. However, it is hard to find automorphic group for m-sequence code. The existing parallel decoding methods cannot be employed by m-sequence code.

To solve this problem, we first propose a method to construct parity-check matrix for m-sequence code. Then, we design a new parallel decoding method for m-sequence code. To construct parity-check matrices, the first step is to generate an m-sequence by an order-k primitive polynomial. Then, the sampling sequences are obtained by sampling the m-sequence with multiple sampling intervals which are relatively prime with. The second step is to find new primitive polynomials which can generate the sampling sequences. The final step is to use the parity-check constraints of these new primitive polynomials to construct parity-check matrix by cyclic shift. The constructed parity-check matrices have many short cycles due to the cyclic shift procedure. Therefore, they cannot be employed by the sub-decoders of MBBP algorithm.

To tackle this problem, we propose a new parallel decoding method which can eliminate the affection of short cycles in parity-check matrices. In our method, the sub-decoders are designed to two types. The first type contains only one decoding module, which uses the basic parity-check matrix constructed in [1] to perform BP algorithm and output candidate codeword. The second type is constructed by a pre-decode module and a decoding module. The pre-decode module use a parity-check matrix to perform the BP process and outputs extrinsic information to the decoding module. At this point, the number of iterations in pre-decode module is set as half of girth of parity-check matrix. Therefore, the outputted extrinsic information will not be affected by short cycles. Then the cascaded decoding module output candidate codeword with the help of extrinsic information. At last, all outputted codewords of sub-decoders are sent into the same LMS and the final codeword is selected by MAP rule.

In this paper, the m sequence codes with code length 3000, 1100 are constructed by primitive polynomial $f(x) = x^{89} + x^{38} + 1$ and $f(x) = x^{33} + x^{13} + 1$, respectively. Then the bit error rate simulation experiments of these codes are carried out. In the experiment, 6 and 4 parity-check matrices are constructed by the proposed construction method, respectively. In addition, when the numbers of sub-decoders are 5 and 4, the bit error rate (BER) curves have converged. Simulation results show that the proposed parallel decoding method is about 0.4 dB better than the single decoder method in [1].

This paper is organized as follows. Section 2 gives a brief introduction of LDPC code generated via primitive polynomial and presents the problem statement. The construction method of parity-check matrices and new parallel decoding method are specified in Section 3. Section 4 shows and analyzes the numerical results. Finally, some conclusions are drawn in Section 5.

2. Motivation

For an m-sequence generated by an LFSR which employs a primitive polynomial as connected polynomial, the segment of the m-sequence can be an LDPC codeword. This LDPC code is called m-sequence code [1]. In this section, we briefly introduce the basis of the code and refer readers to [1] for more details. Then, we discuss the problem of designing parallel decoding method for m-sequence codes.

2.1. LDPC Codes Generated via Primitive Polynomial

Consider an LFSR with connected primitive polynomial $f(x) = x^k + x^p + 1$, where k, p are positive integers such that $k > p$. Denote trace function $tr_1^k(\cdot)$ as a mapping from

field $\mathbf{GF}(2^k)$ to field $\mathbf{GF}(2)$ [11]. Let α be a root of $f(x)$ and β be the initial phase of the LFSR. Then, the m-sequence (a_i) generated by the register can be described as

$$a_i = tr_1^k(\beta\alpha^i)\ \alpha, \beta \in \mathbf{GF}(2^k) \tag{1}$$

For the m-sequence, $f(x)$ further gives the following parity-check constraints

$$a_i + a_{i+p} + a_{i+k} = 0, i = 0, 1, \ldots. \tag{2}$$

With (2), it has been proven that given integer $n(> k)$, we can construct an LDPC matrix for all segments $(a_i)_{i=0}^{n-1}$ truncated from (a_i) [1]. That is, $(a_i)_{i=0}^{n-1}$ is a codeword of an LDPC code.

The parity-check matrix of an m-sequence code has two types of check rows. One is obtained from parity-check constraints in (2). The other is derived from the conjugate primitive elements $\alpha^{2^m},\ m = 1,\ 2, \cdots, M$ of α, where M is the maximum integer such that $2^M k < n$. Consider the m-sequence $(a_i^{(m)})$ generated by conjugate primitive element α^{2^m}, it can be written as

$$a_i^{(m)} = tr_1^k(\beta(\alpha^{2^m})^i) = tr_1^k(\beta\alpha^{2^m i}) = a_{2^m i}. \tag{3}$$

Note that α^{2^m} is the root of $f(x)$ [11]. As a result, sequence $(a_i^{(m)})$ can be also checked by (2), i.e.,

$$a_i^{(m)} + a_{i+p}^{(m)} + a_{i+k}^{(m)} = 0. \tag{4}$$

Combining (3) and (4), the following parity-check constraints for codeword $(a_i)_{i=0}^{n-1}$ are derived

$$a_i + a_{i+2^m p} + a_{i+2^m k} = 0,\ i = 0, 1, \cdots, n - 2^m k - 1. \tag{5}$$

Finally, the parity-check matrix can be 4-cycle free if the factors k, p satisfy the conditions derived in [1], i.e.,

$$\begin{cases} k \neq (1 + 2^m)p \\ k \neq 2^m p \\ k \neq (1 - 2^{-m})p \\ k \neq (1 + 2^{-m})p \end{cases} \tag{6}$$

In this paper, the matrix constructed by (2) and (5) is called the basic parity-check matrix.

2.2. Problem Statement

The code rates of the m-sequence codes are extremely low, as they are supposed to work in extremely low SNR cases. In such cases, it is desirable to design a decoding algorithm with better BER performance.

In [5–7], authors show that the performance of a parallel decoding method with multiple sub-decoders can overcome the single-decoder decoding method. For instance, Figure 1 shows the structure of MBBP, in which each sub-decoder $D_i,\ 0 \leq i \leq N$ uses a different parity-check matrix H_i [5].

LLR $\mathbf{L}_{ch}$ of channel observation value $\mathbf{y}$ is simultaneously entered into the $N + 1$ sub-decoders, resulting in $N + 1$ candidate codewords $\hat{\mathbf{c}}_0, \hat{\mathbf{c}}_1, \cdots, \hat{\mathbf{c}}_N$. The value of each element of the codewords is "1" or "−1". The decoding method is BP decoding algorithm. These candidate codewords are then fed into the LMS, in which the final output codeword is selected by means of the MAP rule or the equivalent minimum distance rule, i.e.,

$$\begin{aligned} \hat{\mathbf{c}} &= \arg\max_{i \in \{0,1,\cdots,N\}} Pr\{\mathbf{y}|\mathbf{c} = \hat{\mathbf{c}}_i\} \\ &= \arg\min_{i \in \{0,1,\cdots,N\}} d(\mathbf{y}, \hat{\mathbf{c}}_i) \end{aligned} \tag{7}$$

where $d(\mathbf{a}, \mathbf{b})$ is the Euclidean distance between vectors $\mathbf{a}$ and $\mathbf{b}$.

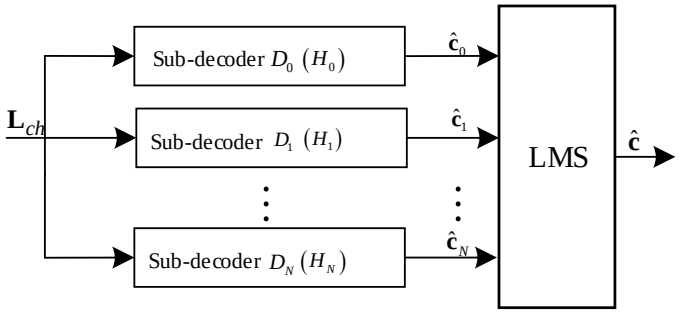

Figure 1. The structure of multiple-based belief propagation (MBBP) algorithm.

For the parallel decoding architecture shown in Figure 1, it requires for sub-decoders that parity-check matrices contain different parity-check constraints with each other. It is due to the fact that errors cannot be fixed by one parity-check matrix are uncorrectable for other parity-check matrices being of the same parity-check constraints. In order to apply such method on m-sequence codes, we need to construct parity-check matrices through different parity-check constraints. However, the construction methods in [5–7] are not design for m-sequence code, while [1] did not give the method to construct different parity-check matrices for m-sequence code.

To solve this problem, we investigate the algebraic properties of sampling sequence of m-sequence to obtain different parity-check constraints. Then, these constraints are utilized to build parity-check matrices for sub-decoders.

3. Parallel Decoding Method for m-Sequence Codes

In this section, we first construct parity-check matrices by using the parity-check constraints of sampling sequence of m-sequence. Then, we analyze the deterioration mechanism of decoding performance caused by short cycles and present a way to eliminate the affection of short cycles. At the last of the section, we develop a new parallel decoding method with sub-decoder consisting of a pre-decode module and a decoding module.

3.1. Construction of Parity-Check Matrices

Finite field $\mathbf{GF}(2^k)$ can be represented by the powers of a primitive element α, i.e.,

$$\mathbf{GF}(2^k) = \{\alpha^{-1} = 0,\ \alpha^0 = 1,\ \alpha,\ \alpha^2, \cdots,\ \alpha^{2^k-2}\}. \tag{8}$$

Consequently, $\alpha^q \in \mathbf{GF}(2^k)$ is the primitive elements of this finite field if and only if $gcd(q, 2^k - 1) = 1$ [11], where $gcd(a, b)$ is the greatest common divisor of a and b. The new primitive element α^q can regenerate $\mathbf{GF}(2^k)$ as follows.

$$\mathbf{GF}(2^k) = \{(\alpha^q)^{-1} = 0, (\alpha^q)^0 = 1, \alpha^q, (\alpha^q)^2, \cdots, (\alpha^q)^{2^k-2}\}. \tag{9}$$

Let $f_q(x)$ be the primitive polynomial of which α^q is a root,

$$f_q(x) = x^k + \sum_{i=1}^{k-1} b_i x^i + 1,\ b_i \in \{0, 1\}. \tag{10}$$

Polynomial $f_q(x)$ is then used to generate a new m-sequence $(a_i^{(q)})$ with the same initial phase β. Thus, sequence $(a_i^{(q)})$ meets the following constraints

$$a_i^{(q)} + \sum_{j=1}^{k-1} b_j a_{i+j}^{(q)} + a_{i+k}^{(q)} = 0. \tag{11}$$

In addition, using the trace function, we have

$$a_i^{(q)} = tr_1^k(\beta(\alpha^q)^i) = a_{qi}. \tag{12}$$

Now, it is clearly shown from (12) that the parity-check constraints of $f_q(x)$ can be used to check the codeword $(a_i)_{i=0}^{n-1}$

$$a_i + \sum_{j=1}^{k-1} b_j a_{i+qj} + a_{i+qk} = 0,\ i = 0,\ 1, \cdots,\ n - qk - 1 \tag{13}$$

Based on the above analysis, we design a construction method of parity-check matrices as Algorithm 1 shown, where $BM((a_i))$ is the function that using Berlekamp-Messey (BM) algorithm [12] to get the minimum generating polynomial of (a_i). $Matrix(f_q(x), n)$ is the function to use $f_q(x)$ to construct a parity-check matrix with code length n by cyclic shift as (13).

In Algorithm 1, we first find all the candidate sampling intervals that are relatively prime with $2^k - 1$. Then we generate a segment of m-sequence $(a_i)_{i=0}^{1000k}$ with length $1000k$ to get the sampling segment $(a_i^{(q)})_{i=0}^{\lfloor 1000k/q \rfloor}$ by the sampling interval q. The BM algorithm can be employed to find the maximum generating polynomial $f_q(x)$ of $(a_i^{(q)})_{i=0}^{\lfloor 1000k/q \rfloor}$. Finally, we use $f_q(x)$ to construct a parity-check matrix through the way of (13).

Algorithm 1: Construction of parity-check matrices for m-sequence code

Input: A primitive polynomial $f(x)$;
A code length n.
Output: The parity-check matrices $H_i, i = 1, 2, \cdots, N$.
1: Begin procedure
2: Initialize: Let Q be the set of sampling interval, k is the order of $f(x)$, counter $cnt = 1$.
3: **for** $i := 2;\ i < \lfloor \frac{n}{k} \rfloor;\ i++$ **do**
4: **if** $gcd(i, 2^k - 1) = 1$ **then**
5: $Q = Q \cup \{i\}$.
6: **end if**
7: **end for**
8: Generate an m-sequence $(a_i)_{i=0}^{1000k}$ via $f(x)$.
9: **for** $q \in Q$ **do**
10: Sample $(a_i)_{i=0}^{1000k}$ with sampling interval q to get $(a_i^{(q)})_{i=0}^{\lfloor 1000k/q \rfloor}$.
11: $f_q(x) = BM((a_i^{(q)})_{i=0}^{\lfloor 1000k/q \rfloor})$.
12: $H_{cnt} = Matrix(f_q(x), n)$.
13: $cnt = cnt + 1$.
14: **end for**
15: **return** $H_i, i = 1, 2, \cdots, N$.

The parity-check matrix generated by this method is the result of cyclic shift of one parity-check constraint. Therefore, it has too many short cycles to be equipped into the sub-decoders of the MBBP algorithm. To take full advantage of the parity-check matrices, we thus design a new parallel decoding method in the next subsection.

3.2. Parallel Decoding Method with Multiple Sub-Decoders

Now let us discuss the affection of short cycles on decoding first. In BP algorithm, the information of each node is transferred and iterated according to the connection relation of Tanner graph. If there exist short cycles in the Tanner graph, the information of nodes in the cycles cannot receive enough extrinsic information, resulting in overestimation of node information and errors in decoding results.

Figure 2 shows information flows of short cycles in a Tanner graph, where Figure 2a,b are 4-cycle and 6-cycle cases respectively. The solid line represents the information transfer in the first iteration, while the segment line and dotted line represent the information flow in the second and third iteration, respectively.

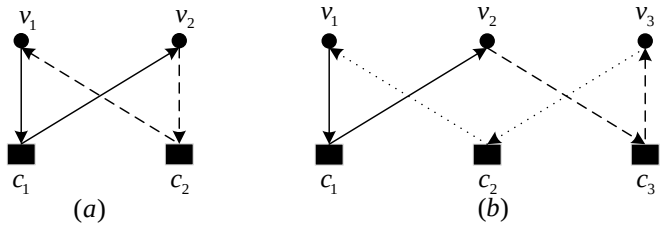

Figure 2. Information flows of short cycles in a Tanner graph. (**a**) 4-cycle. (**b**) 6-cycle.

In Figure 2a, information is first passed to variable node v_2 through the check node c_1. In the second iteration, v_2 passes the information back to v_1 through c_2. This is an information loop. Similarly, the 6-cycle case in Figure 2b completes a loop through 3 iterations.

It can be seen from Figure 2, if the number of iterations is half of the girth, then the information flowing in a short cycle will not back to the original node. As a consequence, the overestimation problem caused by the short cycle will never happen.

However, if the iteration times of BP algorithm is set as half of the girth of parity-check matrix, the BP decoding algorithm is hardly to output a codeword. To tackle this difficult, this paper proposes a new parallel decoding method whose decoding structure is shown in Figure 3.

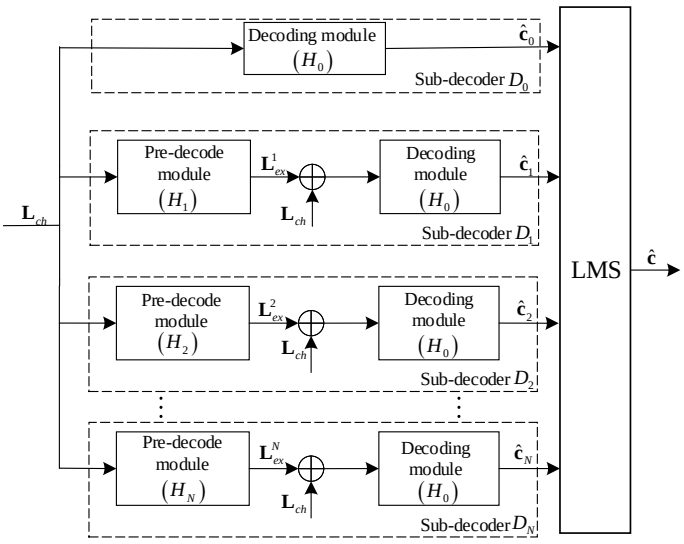

Figure 3. The structure of parallel decoding method.

The sub-decoders of the designed parallel decoding method are of two types. The first type of sub-decoder D_0 only contains the decoding module that directly outputs the codeword $\hat{\mathbf{c}}_0$. The parity-check matrix employed by decoding module is the basic parity-check matrix H_0. The second type of sub-decoders D_i, $i = 1, 2, \cdots, N$ are composed by a pre-decode module and a decoding module. In these sub-decoders, after receiving LLR $\mathbf{L}_{ch}$ of channel received value $\mathbf{y}$, the pre-decode module uses the BP algorithm to output extrinsic information $\mathbf{L}_{ex}^i$. The employed parity-check matrix H_i is generated by Algorithm 1. The number of iterations in the pre-decode module is set as half of the girth of H_i. Therefore, $\mathbf{L}_{ex}^i$ is not affected by the cycle structure of H_i. Then, $\mathbf{L}_{ex}^i$ plus $\mathbf{L}_{ch}$ is sent into the decoding module to get candidate codeword $\hat{\mathbf{c}}_i$ by BP decoding with H_0.

The pre-decode modules employ different parity-check matrices, resulting in different outputted extrinsic information. It enables the decoding modules to output different candidate codewords, which are valuable for the LMS to choose the best final codeword.

The parallel decoding method is summarized in Algorithm 2, where $BP_h(H, \mathbf{L}, t)$ and $BP_s(H, \mathbf{L}, t)$ represents that using BP algorithm with parity-check matrix H to decode the input LLR $\mathbf{L}$. The maximum iteration number is$t$. The output is hard decision sequence for $BP_h(H, \mathbf{L}, t)$ and soft extrinsic information for $BP_s(H, \mathbf{L}, t)$, respectively.

In Algorithm 2, the LLR value $\mathbf{L}_{ch}$ of channel output is simultaneously entered into $N+1$ sub-decoders. Then these sub-decoders D_i, $i = 0, 1, \cdots, N$ output the candidate codewords $\hat{\mathbf{c}}_i$ respectively. Let S be the set of candidates which satisfy $H_0\hat{\mathbf{c}}_i^T = 0$. If $S = \emptyset$, then let $S = \{\hat{\mathbf{c}}_0, \hat{\mathbf{c}}_1, \hat{\mathbf{c}}_2, \cdots, \hat{\mathbf{c}}_N\}$ be the set of all candidates. The decoding result $\hat{\mathbf{c}}$ is selected from S by an LMS with MAP rule.

Algorithm 2: Parallel decoding method

Input: The LLR value $\mathbf{L}_{ch}$ of channel output;
Maximum iteration number t_0;
Basic parity-check matrix H_0;
Parity-check matrices $H_1, H_2, \ldots, H_N$.
Output: The decoding result $\hat{\mathbf{c}}$.

Begin procedure
Initialize: Let S be the set of candidate codewords, $t_i, i = 1, 2, \cdots, N$ is the half of the girth of H_i.
In sub-decoder D_0, $\hat{\mathbf{c}}_0 = BP_h(H_0, \mathbf{L}_{ch}, t_0)$.
if $H_0(\hat{\mathbf{c}}_0)^T = 0$ **then**
  $S := S \cup \{\hat{\mathbf{c}}_0\}$
end if
for $i := 1$; $i < N$; $i++$ **do**
  In sub-decoder D_i, $\mathbf{L}_{ex}^i = BP_s(H_i, \mathbf{L}_{ch}, t_i)$.
  $\hat{\mathbf{c}}_i = BP_h(H_0, \mathbf{L}_{ch} + \mathbf{L}_{ex}^i, t_0)$.
  if $H_0(\hat{\mathbf{c}}_i)^T = 0$ **then**
    $S := S \cup \{\hat{\mathbf{c}}_i\}$.
  end if
end for
if $S = \emptyset$ **then**
  $S := \{\hat{\mathbf{c}}_0, \hat{\mathbf{c}}_1, \hat{\mathbf{c}}_2, \ldots, \hat{\mathbf{c}}_N\}$.
end if
$\hat{\mathbf{c}} = \text{argmin}_{\mathbf{s}\in S} d(\mathbf{L}_{ch}, \mathbf{s})$
return $\hat{\mathbf{c}}$.

4. Numerical Results

In this section, we simulate the BER of m sequence codes generated via primitive polynomials $f(x) = x^{89} + x^{38} + 1$ and $f(x) = x^{33} + x^{13} + 1$. The code rates of these two codes are the same 0.03 and the code lengths are 3000 and 1100, respectively. There are two decoding methods employed in the simulation, one is the proposed parallel decoding method with multiple sub-decoders, the other one is the single-decoder decoding method which employed in [1]. The channel is additive white Gaussian noise (AWGN) channel.

Firstly, we simulate the BER performance of the proposed decoding method with different number of sub-decoders. According to the construction method of parity-check matrix proposed in Section 3.1, we have found the sampling intervals $q = 3, 5, 7, 9, 11, 13$ to sample the m-sequence generated by $f(x) = x^{89} + x^{38} + 1$ and obtain the primitive polynomials of the sampling sequences.

$$\begin{cases} f_3(x) = x^{89} + x^{72} + x^{55} + x^{38} + 1 \\ f_5(x) = x^{89} + x^{61} + x^{38} + x^{33} + 1 \\ f_7(x) = x^{89} + x^{69} + x^{38} + x^{29} + 1 \\ f_9(x) = x^{89} + x^{72} + x^{55} + x^{38} + x^{31} + x^{24} + 1 \\ f_{11}(x) = x^{89} + x^{67} + x^{52} + x^{38} + x^{30} + x^{15} + 1 \\ f_{13}(x) = x^{89} + x^{44} + x^{43} + x^{40} + x^{39} + x^{38} + 1 \end{cases} . \tag{14}$$

For the m-sequence generated by $f(x) = x^{33} + x^{13} + 1$, we also use sampling intervals $q = 3, 5, 7, 9$ to get the sampling sequences and corresponding primitive polynomials.

$$\begin{cases} f_3(x) = x^{33} + x^{29} + x^{17} + x^{13} + 1 \\ f_5(x) = x^{33} + x^{22} + x^{13} + x^{11} + 1 \\ f_7(x) = x^{33} + x^{16} + x^{14} + x^{13} + 1 \\ f_9(x) = x^{33} + x^{22} + x^{16} + x^{13} + x^{11} + x^8 + 1 \end{cases} . \tag{15}$$

Then we have simulated the BER performance of the proposed decoding method with different number of sub-decoders for these two codes, the results are shown in Figure 4.

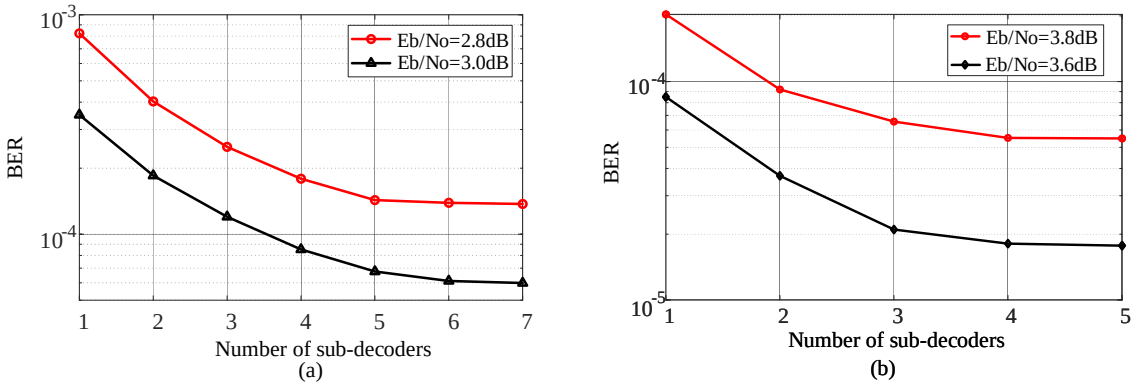

Figure 4. The bit error rate (BER) performance of the proposed decoding method with different number of sub-decoders. (**a**) code length 3000. (**b**) code length 1100.

Figure 4 presents the relation between BER performance and the number of sub-decoders. With the increase of employed sub-decoders, the BER performance is better but the computation complexity is increased. In our method, all the sub-decoders have nearly the same computation complexity. Therefore, the parallel decoding method with N sub-decoders is N times complex than the original single-decoder system. To get the tradeoff of BER performance and complexity, we employ 5 sub-decoders for the code with length 3000 as Figure 4a has shown that the BER curve is converged when employing 5 sub-decoders. Similarly, we employ 4 sub-decoders for the code with length 1100.

In the end, we have simulated the BER performance of these two codes with different decoding methods, the results are shown in Figure 5.

From Figure 5 we can see that, the proposed decoding method outperforms the single-decoder decoding method in terms of BER when the value of E_b/N_0 is fixed. The E_b/N_0 range of the simulation is from 1.4 dB to 4.2 dB while the corresponding SNR range is from -13.82 dB to -11.02 dB. In the BER region of 10^{-5}, our decoding method has a BER gain of 0.3 dB and 0.4 dB compared to the single-decoder decoding method for these two codes, respectively.

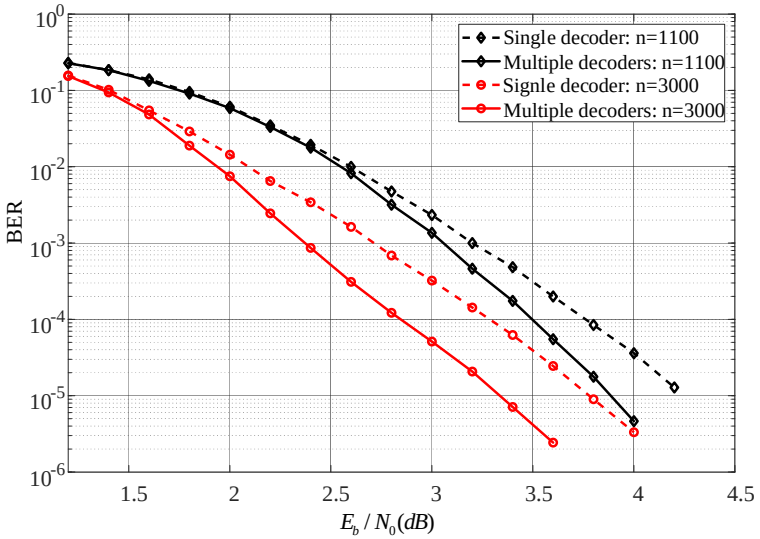

Figure 5. The BER performance of different decoding methods.

5. Conclusions

In this paper, we had improved the BER performance of m-sequences code by applying parallel decoding method. Firstly, to construct parity-check matrices for sub-decoders, we investigated the parity-check constraints from sampling sequence of m-sequence and used them to construct the parity-check matrices by cyclic shift. Then, to overcome the influence of short cycles of parity-check matrices, we had proposed a new parallel decoding method with two types of sub-decoders. Simulation results show that the proposed decoding method improved about 0.4 dB compared with the original single-decoder decoding method at a BER of 10^{-5}.

Author Contributions: Conceptualization, Z.Z. and L.Z.; methodology, Z.Z. and Z.H.Z.; software, Z.Z. and Z.H.Z.; validation, Z.Z., L.Z. and Z.H.Z.; formal analysis, Z.Z. and Z.H.Z.; investigation, Z.Z.; resources, L.Z.; data curation, Z.H.Z.; writing—original draft preparation, Z.Z.; writing—review and editing, Z.Z., L.Z., Z.H.Z.; visualization, Z.Z. and Z.H.Z.; supervision, L.Z.; project administration, L.Z.; funding acquisition, L.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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

Abbreviations

The following abbreviations are used in this manuscript:

LDPC	Low Density Parity Check
BP	Belief Propagation
MAP	Maximum A Posterior
LMS	Least Metric Selector
BER	Bit Error Rate
SNR	signal-to-noise-ratio
LFSR	Linear Feedback Shift Register
MBBP	Multiple-Based Belief Propagation
RRD	Random Redundant Decoding
LLR	Log-Likelihood Ratio
BM	Berlekamp-Messey
AWGN	Additive White Gaussian Noise

References

1. Zhang, Z.; Zhou, L.; Du, J.; Peng, S. An algebraic approach to design low rate low density parity check code. In Proceedings of the 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 11–13 October 2017.
2. De Cola, T.; Paolini, E.; Liva, G.; Calzolari, G.P. Reliability Options for Data Communications in the Future Deep-Space Missions. *Proc. IEEE* **2011**, *99*, 2056–2074. [CrossRef]
3. Abughalieh, N.; Steenhaut, K.; Nowé, A. Low power channel coding for Wireless Sensor Networks. In Proceedings of the 2010 17th IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT2010), Enschede, The Netherlands, 24–25 November 2010.
4. Hu, X.-Y.; Eleftheriou, E.; Arnold, D.M. Regular and irregular progressive edge-growth tanner graphs. *IEEE Trans. Inf. Theory* **2005**, *51*, 386–398 [CrossRef]
5. Hehn, T.; Huber, J.B.; Laendner, S.; Milenkovic, O. Multiple-Bases Belief-Propagation for Decoding of Short Block Codes. In Proceedings of the 2007 IEEE International Symposium on Information Theory, Nice, France, 24–29 June 2007.
6. Hehn, T.; Huber, J.B.; Milenkovic, O.; Laendner, S. Multiple-bases belief-propagation decoding of high-density cyclic codes. *IEEE Trans. Commun.* **2010**, *58*, 1–8. [CrossRef]
7. Hehn, T.; Huber, J.B.; He, P.; Laendner, S. Multiple-Bases Belief-Propagation with Leaking for Decoding of Moderate-Length Block Codes. In Proceedings of the International ITG Conference on Source and Channel Coding, Ulm, Germany, 14–16 January 2008.
8. Halford, T.R.; Chugg, K.M. Transactions Letters—Random Redundant Iterative Soft-in Soft-out Decoding. *IEEE Trans. Commun.* **2008**, *56*, 513–517. [CrossRef]
9. Dimnik, I.; Berery, Y. Improved random redundant iterative HDPC decoding. *IEEE Trans. Commun.* **2009**, *57*, 1982–1985. [CrossRef]
10. Chen, C.; Bai, B.; Yang, X.; Li, L.; Yang, Y. Enhancing Iterative Decoding of Cyclic LDPC Codes Using Their Automorphism Groups. *IEEE Trans. Commun.* **2013**, *61*, 2128–2137. [CrossRef]
11. Zhang, K.Q.T. Channel Coding. In *Wireless Communications: Principles, Theory and Methodology*; Wiley: Hoboken, NJ, USA, 2015; pp. 121–170.
12. Robert Redinbo, G. Correcting DFT Codes with a Modified Berlekamp-Massey Algorithm and Kalman Recursive Syndrome Extension. *IEEE Trans. Comput.* **2014**, *63*, 196–203. [CrossRef]