information

MDPI

Article

Dual Threshold Self-Corrected Minimum Sum Algorithm for 5G LDPC Decoders

Rong Chen [1,2] and Lan Chen [1,*]

[1] Institute of Microelectronics of Chinese Academy of Sciences, Beijing 100029, China; chenrong@ime.ac.cn
[2] School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China
* Correspondence: chenlan@ime.ac.cn; Tel.: +86-010-8299-5745

Received: 13 May 2020; Accepted: 4 July 2020; Published: 7 July 2020

Abstract: Fifth generation (5G) is a new generation mobile communication system developed for the growing demand for mobile communication. Channel coding is an indispensable part of most modern digital communication systems, for it can improve the transmission reliability and anti-interference. In order to meet the requirements of 5G communication, a dual threshold self-corrected minimum sum (DT-SCMS) algorithm for low-density parity-check (LDPC) decoders is proposed in this paper. Besides, an architecture of LDPC decoders is designed. By setting thresholds to judge the reliability of messages, the DT-SCMS algorithm erases unreliable messages, improving the decoding performance and efficiency. Simulation results show that the performance of DT-SCMS is better than that of SCMS. When the code rate is 1/3, the performance of DT-SCMS has been improved by 0.2 dB at the bit error rate of 10^{-4} compared with SCMS. In terms of the convergence, when the code rate is 2/3, the number of iterations of DT-SCMS can be reduced by up to 20.46% compared with SCMS, and the average proportion of reduction is 18.68%.

Keywords: 5G; low-density parity-check codes; belief propagation; minimum sum algorithm; self-corrected minimum sum algorithm

1. Introduction

There are three major application scenarios for fifth generation (5G): enhanced mobile broadband (eMBB), ultra-reliable low latency communication (uRLLC), and massive machine type communication (mMTC), as defined by the International Telecommunication Union (ITU) [1]. These scenarios have the characteristics of ultra-high traffic density, ultra-high connection number density and ultra-high mobility. Faced with diverse scenarios and extremely differentiated performance requirements in each scenario, it is difficult for 5G to form a solution suitable for all scenarios, using only a single technology as the previous communication system. Channel coding is a key technology to achieve the requirements and targets of 5G. In 5G new radio (NR), the new channel coding solution needs to support incremental-redundancy hybrid automatic repeat request (IR-HARQ), as well as various block lengths and code rates [2]. In order to meet these demands and achieve a high-rate transmission, the NR standard selects low-density parity-check (LDPC) codes as the long code coding scheme for the data channel [3,4].

The 5G NR LDPC coding scheme uses two base graph (BG) check matrices to adapt to different code rates and block lengths. BG1 is suitable for code rates 1/3~8/9, with the maximum block length of 8448 bits; BG2 is suitable for code rates 1/5~2/3, with the maximum block length of 3840 bits. [2] According to the saying above, various block lengths and code rates are required for LDPC codes in the 5G NR. Much more than this, in the coding scheme, the first several columns of the code block are selected for puncturing. Experiments show that the introduction of relatively high puncturing variable

nodes can produce improved performance with lower complexity [3]. However, the introduction of puncturing variable nodes will also bring some performance loss when decoding, so it is necessary to improve the existing decoding algorithms.

The LDPC code is a type of error correction coding with strong error correction capability, which was first proposed by MIT's Gallager in the 1960s [5]. Since LDPC codes have the advantages of simple description, low decoding complexity, parallel implementation, flexible use, and low error floor, they are widely used in practical communication and data storage systems [6–9]. The LDPC code did not attract people's attention when it was proposed by Gallager due to the hardware and computer conditions at that time. In the 1990s, MacKay and Neal proved that LDPC codes had the performance of approximating the Shannon limit under the combination of iterative decoding with the belief propagation (BP) algorithm, since then LDPC codes have gradually become a hot topic in academic research [10].

The disadvantage of the BP algorithm is that the computational complexity is high and the performance is seriously degraded, especially when the message quantization bits are low. In 1999, Fossorier proposed the minimum sum (MS) algorithm to simplify the calculation of check node messages [11], and then MS quickly became the most widely used decoding algorithm. However, duo to the simplification, the decoding performance has a large loss, as well as some hard-decision algorithms [12–15]. Therefore, many new improved algorithms appeared [16], such as the performance improvement algorithms modified by MS [17–20], the approximate simplified algorithms for the decoding function [21–24], and the scheduling algorithms with improved decoding efficiency [25–30]. Among them, the offset minimum sum (OMS) algorithm and the normalized minimum sum (NMS) algorithm proposed by Chen et al. improved the performance limitedly. Although the function approximation algorithms have improved performance, they still increase the computational complexity. The scheduling algorithms can reduce the computational complexity, but will lose some performance. Besides, the self-corrected minimum sum (SCMS) algorithm proposed by Savin et al. improves the decoding performance from the perspective of variable node messages and accelerates the convergence of decoding [31–33]. However, the SCMS algorithm does not perform well when the channel state is good, and cannot meet the requirements of high reliability and efficiency-oriented communication systems.

In this paper, a dual threshold self-corrected minimum sum (DT-SCMS) algorithm based on the SCMS algorithm is proposed for the design requirements of LDPC codes in 5G communication. In the process of iterative decoding, the new algorithm determines the reliability of the messages by setting thresholds and erasing the unreliable messages, which improves the decoding performance and efficiency. Simulation results show that the DT-SCMS algorithm is superior to the SCMS algorithm in terms of error characteristics and convergence characteristics. When the code rate is 2/3, the DT-SCMS algorithm can reduce the number of iterations by up to 20.46% compared with the SCMS algorithm, and the average proportion of reduction is 18.68%.

The remainder of this paper is organized as follows. Section 2 introduces the basic decoding algorithm of LDPC codes. Section 3 proposes a new decoding algorithm for LDPC codes. Section 4 exhibits the performance analyses and comparisons. An architecture of the LDPC decoder is designed in Section 5. Finally, Section 6 concludes the paper.

2. Basic Decoding Algorithm of LDPC Codes

2.1. BP Algorithm

Among the decoding algorithms of LDPC codes, the BP algorithm has the most outstanding decoding performance, but the complexity is very high. For binary LDPC codes, by converting the probability operation in the algorithm to the log domain, the LLR (log-likelihood ratio) BP algorithm can be obtained. The LLR BP algorithm uses simple addition operations instead of complex multiplication

operations, which greatly reduces the complexity of the decoding operations in the BP algorithm. For the convenience of description, the definitions of relevant variable symbols are shown in Table 1.

Table 1. Definitions of variable symbols.

Symbol	Definition
H	The parity check matrix
i	A variable node in H, $i \in \{1, 2, \cdots N\}$
j	A check node in H, $j \in \{1, 2, \cdots M\}$
$C(i)$	A collection of check nodes connected to variable node i
$R(j)$	A collection of variable nodes connected to check node j
$C(i) \backslash j$	A set of check nodes connected to variable node i except j
$R(j) \backslash i$	A set of variable nodes connected to check node j except i
$L(P_i)$	Channel initial probability likelihood ratio message
$L(r_{ji})$	The check node message (extrinsic probability likelihood ratio message from variable node i to check node j)
$L(q_{ij})$	The variable node message (extrinsic probability likelihood ratio message from check node j to variable node i)
$L(q_i)$	All messages collected by variable node i
α	The correction factor in the NMS algorithm
$\hat{c}$	Decoding sequence obtained by decoding decision

The steps of the LLR BP algorithm can be summarized as follows:

(1) Initialization

Set the maximum number of iterations (I_{max}), for every variable node calculate the initial probability likelihood ratio message $L(P_i)$·$(L(P_i) = log\left(\frac{Prob(i=0)}{Prob(i=1)}\right))$ that the channel passes to the variable node, set the initial message of the check nodes to 0, and then for each variable node i and its adjacent check node $j \in C(i)$, set the initial message that the variable node sends to the check node as:

$$L^{(0)}\left(q_{ij}\right) = L(P_i) \tag{1}$$

(2) Iterative processing

Step 1: Check Node Processing (CNP)

For each check node j and its adjacent check nodes $i \in R(j)$, at the l-th iteration the check node message is calculated as [22]:

$$L^{(l)}\left(r_{ji}\right) = 2\tanh^{-1}\left(\prod_{i' \in R_j \backslash i} \tanh\left(\frac{1}{2} L^{(l-1)}\left(q_{i'j}\right)\right)\right) \tag{2}$$

Step 2: Variable Node Processing (VNP)

For each variable node i and its adjacent check nodes $j \in C(i)$, at the l-th iteration the variable node message is calculated as [22]:

$$L^{(l)}\left(q_{ij}\right) = L(P_i) + \sum_{j' \in C_i \backslash j} L^{(l)}\left(r_{j'i}\right) \tag{3}$$

Step 3: Decision

For every variable node, the decision message is calculated as [22]:

$$L^{(l)}(q_i) = L(P_i) + \sum_{j \in C_i} L^{(l)}\left(r_{ji}\right) \tag{4}$$

Following this, a hard decision is performed. If $L^{(l)}(q_i) > 0$, then $\hat{c}_i = 0$, otherwise $\hat{c}_i = 1$.

(3) Stop

If $H\hat{c}^T = 0$, or the maximum number of iterations is reached, the iteration ends, otherwise the iteration is continued from step 1.

2.2. MS Algorithm

The MS algorithm is an approximation of the LLR BP algorithm. In the update of the check node, the minimum and second minimum values are used instead of the confidence information to be transmitted. In the entire implementation process, there are only "sum" and "comparison" operations.

The CNP of the MS algorithm is described as follows.

For each check node j and its adjacent check nodes $i \in R(j)$, at the l-th iteration the check node message is calculated as [22]:

$$L\left(r_{ji}\right) = \prod_{i' \in R_j \backslash i} \operatorname{sgn}\left(L\left(q_{i'j}\right)\right) \cdot \min_{i' \in R_j \backslash i}\left(\left|L\left(q_{i'j}\right)\right|\right) \tag{5}$$

The MS algorithm greatly reduces the hardware complexity, but the approximate substitution sacrifices a part of the performance. To make up for the performance loss, the NMS algorithm and OMS algorithm are the typical improved algorithms.

NMS algorithm: reduce the magnitude of message confidence by dividing by a value, the check message is expressed as [22]:

$$L\left(r_{ji}\right) = \prod_{i' \in R_j \backslash i} \operatorname{sgn}\left(L\left(q_{i'j}\right)\right) \cdot \min_{i' \in R_j \backslash i}\left(\left|L\left(q_{i'j}\right)\right|\right) \cdot \frac{1}{\alpha} \tag{6}$$

where α is called the correction factor.

OMS algorithm: reduce the magnitude of message confidence by subtracting a value, the check message is expressed as [22]:

$$L\left(r_{ji}\right) = \prod_{i' \in R_j \backslash i} \operatorname{sgn}\left(L\left(q_{i'j}\right)\right) \cdot \max\left(\min_{i' \in R_j \backslash i}\left(\left|L\left(q_{i'j}\right)\right|\right) - \beta, 0\right) \tag{7}$$

where β is called the offset factor.

3. Proposed DT-SCMS Algorithm

The corrections of the NMS algorithm and OMS algorithm are made from the processing of check node messages, and there is another type of improvement that makes corrections from variable node messages, namely the SCMS algorithm. During the variable node message processing, according to the variable fluctuation of the variable node message, the "untrusted" message is selectively erased, which can speed up the convergence and restore the performance loss of the MS algorithm to a certain extent.

3.1. SCMS Algorithm

The VNP of the SCMS algorithm is described as follows:

For each variable node i and its adjacent check nodes $j \in C(i)$, at the l-th iteration the variable node message is calculated as [31]:

$$L^{(l)}\left(q_{ij}\right) = L^{(l-1)}(q_i) - L^{(l-1)}\left(r_{ji}\right) \tag{8}$$

$$e_{ij}{}^{(l)} = \left(\sim e_{ij}{}^{(l-1)}\right) \cdot \left(s_{ij}{}^{(l)} \oplus s_{ij}{}^{(l-1)}\right) \tag{9}$$

$$L^{(l)}\left(q_{ij}^{\mathrm{SC}}\right) = \left(e_{ij}^{(l)} = 1\right) ? 0 : L^{(l)}\left(q_{ij}\right) \tag{10}$$

where "⊕" means the Exclusive-OR (XOR) operation in binary; ${e_{ij}}^{(l)}$ is used to indicate the position of $L^{(l)}\left(q_{ij}\right)$ that is erased in this iteration, to prevent erasing the message of the same position in two consecutive iterations; ${s_{ij}}^{(l)}$ denotes the sign of $L^{(l)}\left(q_{ij}\right)$ in this iteration; $L^{(l)}\left(q_{ij}^{\mathrm{SC}}\right)$ is the new variable node message obtained according to the erasure rule.

3.2. DT-SCMS Algorithm

Based on the SCMS algorithm, the proposed DT-SCMS algorithm adds two thresholds when judging the reliability of the variable node messages, and dynamically adjusts the range of the erasure messages so that it can better adapt to different environments. In addition, the new algorithm uses the NMS algorithm in the check node update process to ensure the performance of the decoding.

In the variable node processing, if the variable node message changes between two set thresholds in the opposite sign direction, it will be set to zero in the present update. This can be understood by means of Figure 1.

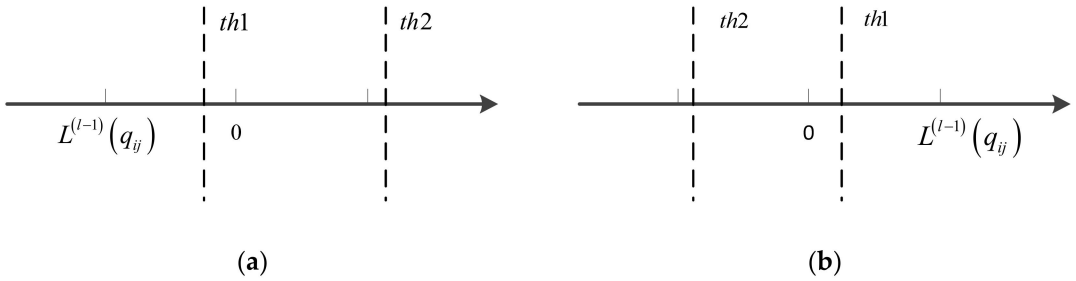


Figure 1. Schematic diagram of reliable message determination for the dual threshold self-corrected minimum sum (DT-SCMS) algorithm.

In Figure 1, $L^{(l-1)}\left(q_{ij}\right)$ is the variable node message in the previous iteration. $th1$ and $th2$ are the set thresholds. If $L^{(l-1)}\left(q_{ij}\right)$ is a negative value, then $L^{(l)}\left(q_{ij}\right)$ will be erased when it is between $th1$ and $th2$ in Figure 1a. If $L^{(l-1)}\left(q_{ij}\right)$ is a positive value, then $L^{(l)}\left(q_{ij}\right)$ will be erased if it falls between $th1$ and $th2$ in Figure 1b.

3.3. Thresholds Setting

If $th1$ deviates far from $L^{(l-1)}\left(q_{ij}\right)$, only a small number of node messages are modified so that the performance improvement is limited. Conversely, if $th1$ deviates near from $L^{(l-1)}\left(q_{ij}\right)$, a large number of node messages are modified, which expands the range of unreliable messages and causes degradation in the decoding performance. The purpose of adding $th2$ is to identify $L^{(l-1)}\left(q_{ij}\right)$ as a reliable message when it deviates very far from $L^{(l-1)}\left(q_{ij}\right)$. Here, the thresholds $th1$ and $th2$ are defined as:

$$th1 = \theta 1 \cdot L^{(l-1)}\left(q_{ij}\right) \tag{11}$$

$$th2 = \theta 2 \cdot L^{(l-1)}\left(q_{ij}\right) \tag{12}$$

where $\theta 1$ and $\theta 2$ are adjustment factors, $\theta 1$ ranges from $-0.5 \le \theta 1 \le 0.5$, and $\theta 2$ ranges from $-1.5 \le \theta 2 \le -0.5$, so that the value of $th1$ does not deviate too far or too near from $L^{(l-1)}\left(q_{ij}\right)$, and $th2$ can filter out certain reliable message nodes. At the same time, both $th1$ and $th2$ are the relative values of $L^{(l-1)}\left(q_{ij}\right)$. As $L^{(l-1)}\left(q_{ij}\right)$ is changed in each iteration, $th1$ and $th2$ will also change as iterative updates, making sure that the criteria for judging reliability are adaptively updated.

In different LDPC codes, different thresholds can be set variously according to the requirements of the applications. The method of density evolution can be used to choose the optimal values of $\theta 1$ and $\theta 2$, as the NMS algorithm uses [17].

3.4. Iterative Process

The flow chart of the DT-SCMS algorithm is shown in Figure 2.

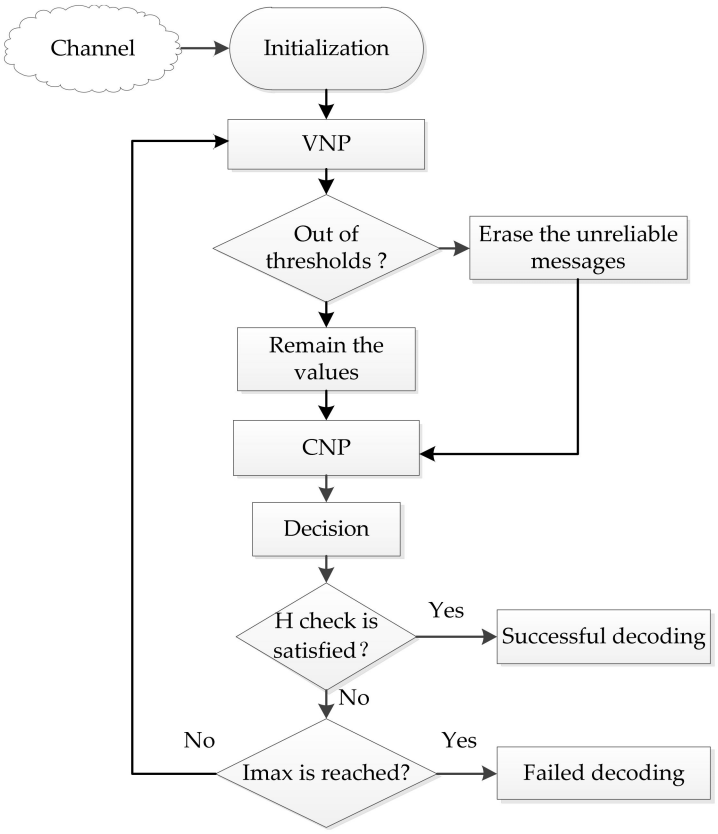


Figure 2. The flow chart of the DT-SCMS algorithm.

In the check node processing (CNP), NMS is adopted to ensure the decoding performance, and α is the correction factor. The iterative process of the DT-SCMS algorithm is described as follows:

Step 1: VNP

For each variable node i and its adjacent check nodes $j \in C(i)$, at the l-th iteration the variable node message is calculated as:

$$L^{(l)}\left(q_{ij}\right) = L^{(l-1)}(q_i) - L^{(l-1)}\left(r_{ji}\right) \tag{13}$$

$$e_{ij}^{(l)} = \left(\sim e_{ij}^{(l-1)}\right) \cdot \left(\left(s_{ij}^{(l-1)} \cdot \left(L^{(l)}\left(q_{ij}\right) - th1\right) > 0\right) \& \left(s_{ij}^{(l-1)} \cdot \left(L^{(l)}\left(q_{ij}\right) - th2\right) < 0\right)\right) \tag{14}$$

$$L^{(l)}\left(q_{ij}^{\mathrm{DTSC}}\right) = \left(e_{ij}^{(l)} = 1\right) ?0 : L^{(l)}\left(q_{ij}\right) \tag{15}$$

Step 2: CNP

For each variable node i and its adjacent check nodes $j \in C(i)$, at the l-th iteration the variable node message is calculated as:

$$L^{(l)}\left(r_{ji}\right) = \prod_{i' \in R(j) \backslash i} s_{i'j}^{(l)} \cdot \min_{i' \in R(j) \backslash i} \left(\left|L^{(l)}\left(q_{i'j}^{\mathrm{DTSC}}\right)\right|\right) \cdot \frac{1}{\alpha} \tag{16}$$

$e_{ij}^{(l)}$ is used to indicate the position of $L^{(l)}\left(q_{ij}\right)$ that is erased in this iteration, to prevent erasing the message of the same position in two consecutive iterations, $s_{ij}^{(l-1)}$ denotes the sign of $L^{(l-1)}\left(q_{ij}\right)$ in this iteration, $L^{(l)}\left(q_{ij}^{\mathrm{DTSC}}\right)$ is the new variable node message obtained according to the erasure rule.

Step 3: Decision

For every variable node, the decision message is calculated as:

$$L^{(l)}(q_i) = L(P_i) + \sum_{j \in C_i} L^{(l)}\left(r_{ji}\right) \tag{17}$$

And then a hard decision is performed. If $L^{(l)}(q_i) > 0$, then $\hat{c}_i = 0$, otherwise $\hat{c}_i = 1$.

SCMS sets zero as a fixed threshold to judge the unreliable messages, ignoring the changes of the variable node messages. In the two adjacent iterations, the change range of the variable node message is unknown. According to the dynamic adjustment rules of the thresholds, Equation (14) gives the positions of the unreliable messages which will be erased in the iteration. Therefore, both small and large changes will be considered. If the new value ($L^{(l)}\left(q_{ij}\right)$) deviates too near from the last value ($L^{(l-1)}\left(q_{ij}\right)$), we can treat it as a reliable message (the role of $th1$); if $L^{(l)}\left(q_{ij}\right)$ deviates too far from $L^{(l-1)}\left(q_{ij}\right)$, we can also treat it as a reliable message (the role of $th2$); besides, if $L^{(l)}\left(q_{ij}\right)$ falls between $th1$ and $th2$, it will be eased. In this way, the proposed algorithm effectively suppresses the transmission of unreliable messages, and retains reliable messages, thereby improving the reliability of decoding and accelerating the convergence speed of decoding.

4. Performance Analysis and Comparisons

4.1. Experiments and Simulations

Some experiments are conducted at the code rates of 1/3 and 2/3 in the 5G LDPC coding scheme. The coded bits are transmitted by Binary Phase Shift Keying (BPSK) modulation with the Additive White Gaussian Noise (AWGN) channel on the Matlab platform. The algorithms of BP, MS, NMS, SCMS, DT-SCMS are simulated with the parameter settings in Table 2.

Table 2. Simulation parameters.

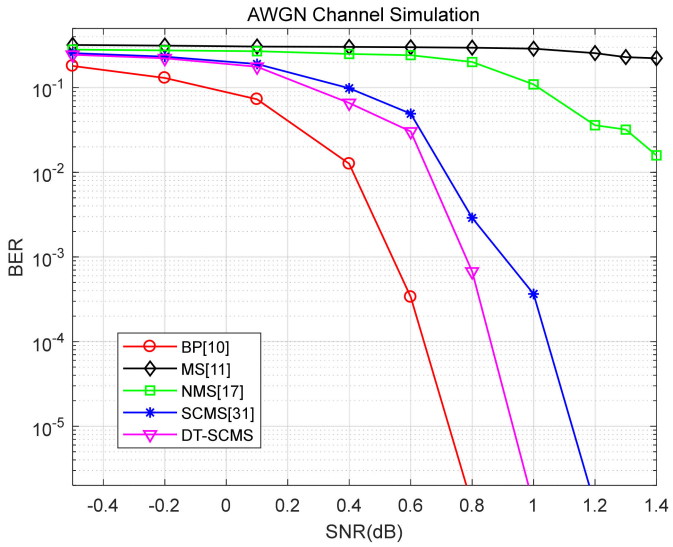
Parameters	R = 1/3	R = 2/3
Code length (bits)	4224	1584
SNR (dB)	[−0.5:1.4]	[1.6:3.4]
Maximum number of iterations	30	30
Correction factor α	1.2	1.2
Adjustment factor $\theta 1$	0.1	0.125
Adjustment factor $\theta 2$	−1.2	−1.125

The simulation results are shown in Figures 3 and 4.

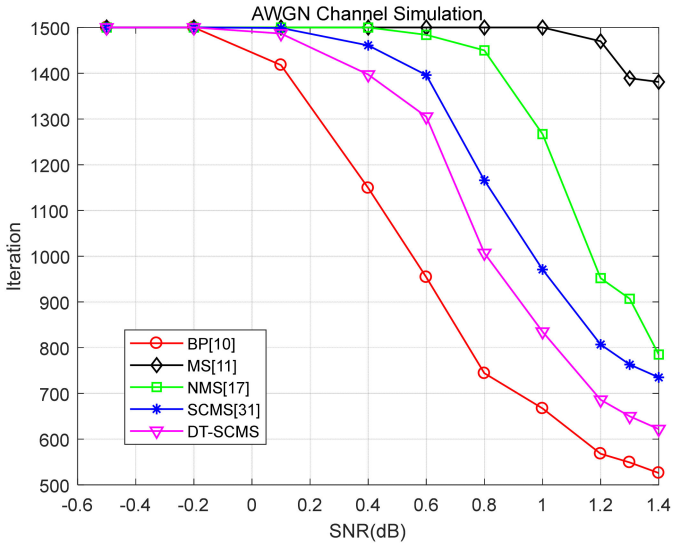
As can be seen from Figures 3 and 4, the DT-SCMS algorithm is superior to the SCMS algorithm both in error characteristics and convergence characteristics. Especially in the areas of high signal-to-noise ratio (SNR), the DT-SCMS algorithm performs more prominently than the SCMS algorithm in terms of convergence. The number of iterations of the SCMS algorithm and the DT-SCMS algorithm in different SNRs in Figure 4b are listed in Table 3.

Table 3. Iterations of SCMS and DT-SCMS under different SNRs (R = 2/3).

SNR (dB)	2.8	3	3.1	3.2	3.3	3.4
SCMS [31]	3311	2811	2607	2467	2246	2214
DT-SCMS	2748	2311	2124	2016	1797	1761



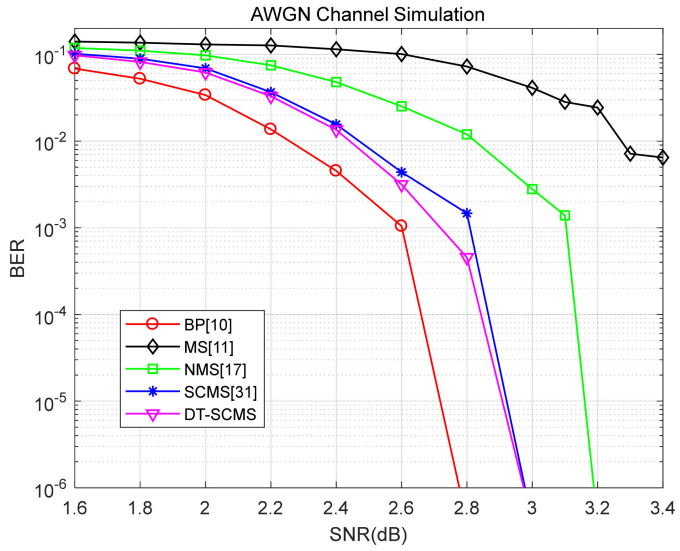
(**a**) Error characteristics



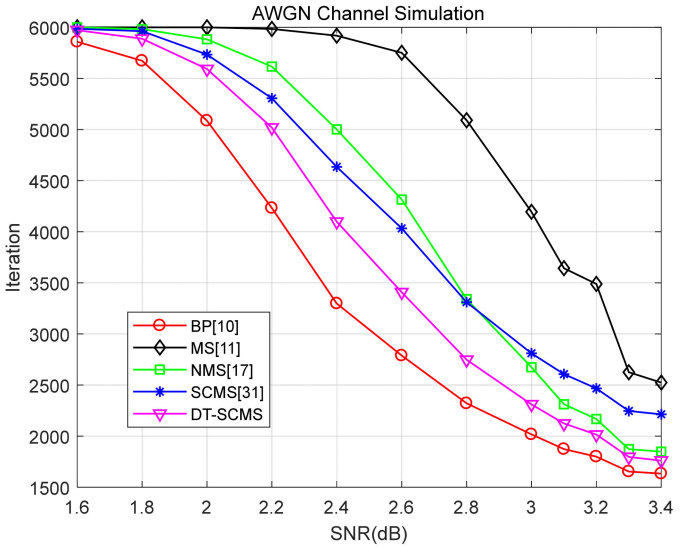
(**b**) Convergence characteristics

Figure 3. Simulations of belief propagation (BP), minimum sum (MS), normalized minimum sum (NMS), self-corrected minimum sum (SCMS), DT-SCMS, when R=1/3.

From Figure 4b and Table 3, it can be seen that, as the channel SNR increases, the number of iterations of LDPC decoding is gradually reduced. Obviously, the convergence characteristics of the DT-SCMS algorithm are better than those of the SCMS algorithm, especially when the SNR is high. This is because the dynamic judgment of the DT-SCMS algorithm on the erased messages is more effective in suppressing the unreliable messages passing, thus speeding up the convergence of decoding. According to Table 3, by calculation, when the code rate is 2/3, the DT-SCMS algorithm can reduce the number of iterations by up to 20.46% compared with the SCMS algorithm, and the average reduction proportion is 18.68%.



(**a**) Error characteristics



(**b**) Convergence characteristics

Figure 4. Simulations of BP, MS, NMS, SCMS, DT-SCMS, when R=2/3.

In the experiment with R = 2/3, several intermediate values of $L^{(l-1)}\left(q_{ij}\right)$ are taken out from the process of iteration, and the values of $th1$ and $th2$ are calculated, as shown in Table 4.

Table 4. Some values of $th1$ and $th2$.

$L^{(l-1)}\left(q_{ij}\right)$	**−1.1247**	**5.2952**	**−3.9980**	**8.1010**	**3.2369**
$th1$	−0.1406	0.6619	−0.4998	1.0126	0.4046
$th2$	1.2653	−5.9571	4.4977	−9.1136	−3.6415

From Table 4, we can see that $th1$ and $th2$ change dynamically with the value of $L^{(l-1)}\left(q_{ij}\right)$. In addition, experiments show that this dynamic adjustability improves the decoding performance of the proposed algorithm.

4.2. Complexity Analyses

Table 5 compares the computational complexity of BP, MS, NMS, SCMS, and DT-SCMS algorithms in a single iteration where N represents the encoding length, K represents the effective information length, and W is the column weight of the check matrix.

Table 5. Computational complexity of decoding algorithms.

Algorithms	Multiplication	Division	Addition
BP [10]	$11NW - 6(N + K)$	$N(W + 1)$	$N(3W + 1)$
MS [11]	0	0	$N(4W - 1) + K(\log W - 2)$
NMS [17]	0	NW	$N(4W - 1) + K(\log W - 2)$
SCMS [31]	0	0	$N(4W - 1) + K(\log W - 2)$
DT-SCMS	$2NW$	NW	$N(6W - 1) + K(\log W - 2)$

It can be seen that the improved algorithm based on the MS algorithm, on the basis of the BP algorithm, reduces a large number of multiplication and division operations, which will greatly reduce the calculation complexity in the implementation. The DT-SCMS algorithm proposed in this paper adds a small amount of multiplication and division operations to the variable node message processing. During implementation, the adjustment factors of thresholds can be set to 1/2, 1/4, 1/8, etc., which can be converted into simple shift and addition operations to obtain corresponding results. In general, the DT-SCMS algorithm improves the performance of the decoding algorithm and reduces the number of iterations with a small amount of computational complexity.

5. Design Architecture and Implementation

The Vivado High-Level Synthesis (HLS) platform launched by Xilinx has implemented a rapid design with IP as the core, greatly improved productivity, and has been widely used in many designs [34,35]. Vivado HLS is a high-level programming language that helps to achieve Register Transfer Level (RTL) hardware functions, improving the level of abstraction of the system design.

An architecture of the LDPC decoder is designed in Figure 5, and mainly includes three modules: initialization (readData), information update (updateInfo), and decision (Check). The information update module consists of CNP (rowUpdate) and VNP (colUpdate). In the Vivado HLS platform, the LDPC decoder with a code length of 1584 bits and a code rate of R = 2/3 is simulated and synthesized, using the chip "xc7k420tffg901-2" from Xilinx's device library.

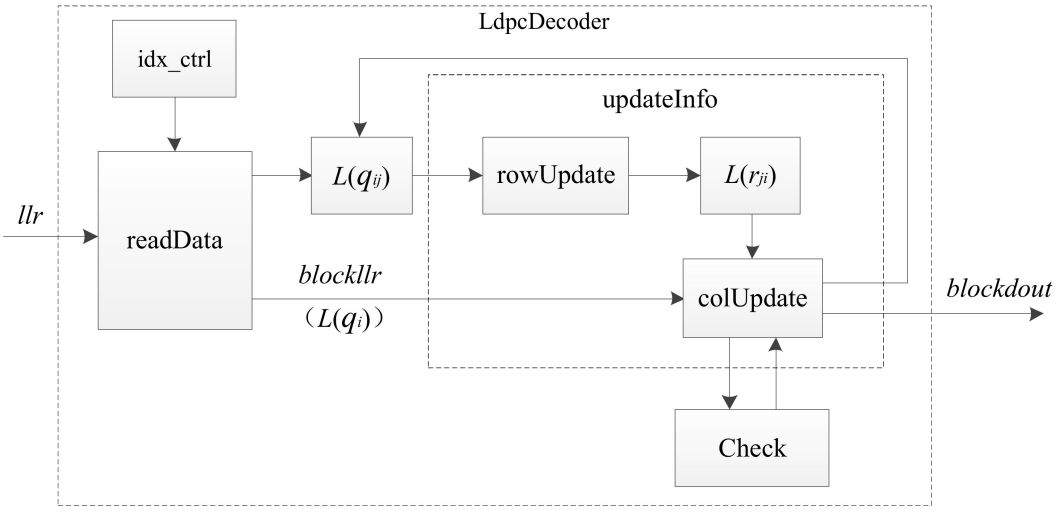


Figure 5. Design of low-density parity-check (LDPC) decoders based on Vivado High-Level Synthesis (HLS).

In Figure 5, *llr* indicates the channel input information, and *blockdout* indicates the output of decoding results. In Vivado HLS, each module is implemented by C ++ functions. After compilation, it can be simulated and synthesized, and verified by the test platform. The initialization module mainly completes reading the initial value from the input channel information (*llr*) and storing it in the variable node message ($L(q_{ij})$) and the judgment information ($L(q_i)$). The CNP module is also called "the row update module", which mainly calculates the check node message ($L(r_{ji})$). The VNP module is also called "the column update module". It first calculates the judgment message ($L(q_i)$), and then calculates the variable node message ($L(q_{ij})$). The performance evaluation of the decoder after synthesis in Vivado HLS is shown in Figure 6.

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
42336	42336	42337	42337	none

Detail

Instance

		Latency		Interval		
Instance	Module	min	max	min	max	Type
grp_updateMinfo_fu_1269	updateMinfo	1802	1802	1802	1802	none
grp_readData_fu_1466	readData	1586	1586	1586	1586	none

Figure 6. Comprehensive performance evaluation.

It can be seen that a total of 42,336 clock cycles are required to complete one decoding process (22 iterations), of which approximately 1586 cycles are required to read the channel soft information, and 1802 cycles are required for decoding iterations. As described in Section 4.1, compared with the SCMS algorithm, the proposed DT-SCMS algorithm can greatly reduce the number of decoding iterations. Therefore, the total latency of decoding can be correspondingly reduced when the average number of iterations required for the decoding is reduced.

6. Conclusions

In this paper, we propose a dual threshold self-corrected minimum sum algorithm based on the SCMS algorithm, and an architecture of LDPC decoders is designed. According to the variable node message, two thresholds are set to judge whether the message is reliable, then the unreliable message is erased, waiting for the next iteration update. The thresholds are set with the relative values of the variable node message, which change with the iteration update, so the criterion for determining the reliability is adaptively updated. Simulation results indicate that the DT-SCMS algorithm shows better performance in a certain code rate range with a small amount of computational complexity, which can effectively improve the error characteristics and convergence characteristics of the decoding system.

Author Contributions: Conceptualization, R.C. and L.C.; methodology, validation, review and editing, R.C.; funding acquisition, L.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Science and Technology Major Project (No.2018ZX03001006-002).

Acknowledgments: The first author, R.C., hereby acknowledges the Institute of Microelectronics of the Chinese Academy of Sciences (IMECAS) and the EDA Center.

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

References

1. Huo, Y.; Dong, X.; Xu, W.; Yuen, M. Enabling Multi-Functional 5G and Beyond User Equipment: A Survey and Tutorial. *IEEE Access* **2019**, *7*, 116975–117008. [CrossRef]
2. 3GPP. NR; Multiplexing and Channel Coding (TS 38.212 Release 15). 3GPP TS 38.212 V15.0.0. Available online: https://www.3gpp.org/specifications (accessed on 2 November 2019).
3. Richardson, T.; Kudekar, S. Design of Low-Density Parity Check Codes for 5G New Radio. *IEEE Commun. Mag.* **2018**, *56*, 28–34. [CrossRef]
4. Nguyen, T.T.B.; Nguyen Tan, T.; Lee, H. Efficient QC-LDPC Encoder for 5G New Radio. *Electronics* **2019**, *8*, 668. [CrossRef]
5. Gallager, R. Low-density parity-check code. *Ire Trans. Inf. Theory* **1962**, *8*, 21–28. [CrossRef]
6. Jung, Y.M.; Jung, Y.H.; Lee, S.J.; Kim, J. Low-complexity multi-way and reconfigurable cyclic shift network of QC-LDPC decoder for Wi-Fi/WIMAX applications. *IEEE Trans. Consum. Electron.* **2013**, *59*, 467–475. [CrossRef]
7. Liu, Y.F.; Olmos, P.M.; Mitchell, G.M. Generalized LDPC codes for ultra reliable low latency communication in 5G and beyond. *IEEE Access* **2018**, *6*, 72002–72014. [CrossRef]
8. Zhang, Y.S.; Zhang, C.; Yan, Z.Y.; Chen, S.; Jiang, H.J. A high-throughput multi-rate LDPC decoder for error correction of solid-state drives. In Proceedings of the 2015 IEEE Workshop on Signal Processing Systems (SiPS), Hangzhou, China, 14–16 October 2015; pp. 1–6. [CrossRef]
9. Chen, Z.J.; Jin, S.; Zhang, C.; Yan, F. A low complexity LDPC-BCH concatenated decoder for NAND flash memory. *IEICE Electron. Express* **2018**, *15*, 20180103. [CrossRef]
10. MacKay, D.J.C.; Neal, R.M. Near Shannon limit performance of low density parity check codes. *Electron. Lett.* **1996**, *32*, 1645–1646. [CrossRef]
11. Fossorier, M.P.C.; Mihaljevic, M.; Imai, H. Reduced complexity iterative decoding of low-density parity check codes based on belief propagation. *IEEE Trans. Commun.* **1999**, *47*, 673–680. [CrossRef]
12. Jiang, M.; Zhao, C.M.; Shi, Z.H.; Chen, Y. An Improvement on the Modified Weighted Bit Flipping Decoding Algorithm for LDPC Codes. *IEEE Commun. Lett.* **2005**, *9*, 814–816. [CrossRef]
13. Lee, C.H.; Wolf, W. Implementation-efficient reliability ratio based weighted bit-flipping decoding for LDPC codes. *Electron. Lett.* **2005**, *41*, 755–757. [CrossRef]
14. Balasuriya, N.; Wavegedara, C.B. Improved symbol value selection for symbol flipping-based non-binary LDPC decoding. *Eurasip J. Wirel. Commun. Netw.* **2017**, *2017*, 1–7. [CrossRef]
15. Ren, D.; Sha, J. Improved gradient descent bit flipping decoder for LDPC codes on BSC channel. *IEICE Electron. Express* **2018**, *15*, 20180195. [CrossRef]
16. Cho, K.; Chung, K.S. Conditional termination check min-sum algorithm for efficient LDPC decoders. *IEICE Electron. Express* **2015**, *12*, 20150738. [CrossRef]
17. Chen, J.; Fossorier, M.P.C. Density evolution for two improved BP-based decoding algorithms of LDPC codes. *IEEE Commun. Lett.* **2002**, *6*, 208–210. [CrossRef]
18. Chen, J.; Fossorier, M.P.C. Near optimum universal belief propagation based decoding of low-density parity check codes. *IEEE Trans. Commun.* **2002**, *50*, 406–414. [CrossRef]
19. Wu, X.F.; Song, Y.; Jiang, M.; Zhao, C.M. Adaptive-normalized/offset min-sum algorithm. *IEEE Commun. Lett.* **2010**, *14*, 667–669. [CrossRef]
20. Myung, S.; Park, S.I.; Kim, K.J.; Lee, J.Y.; Kwon, S.; Kim, J. Offset and normalized min-sum algorithms for ATSC 3.0 LDPC decoder. *IEEE Trans. Broadcasting* **2017**, *63*, 734–739. [CrossRef]
21. Hu, X.Y.; Eleftheriou, E.; Arnold, D.M.; Dholakia, A. Efficient implementations of the sum-product algorithm for decoding LDPC codes. In Proceedings of the GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No.01CH37270), San Antonio, TX, USA, 25–29 November 2001. [CrossRef]
22. Chen, J.; Dholakia, A.; Eleftheriou, E.; Fossorier, M.P.C.; Hu, X.Y. Reduced-complexity decoding of LDPC codes. *IEEE Trans. Commun.* **2005**, *53*, 1288–1299. [CrossRef]
23. Papaharalabos, S.; Mathiopoulos, P.T. Simplified sum-product algorithm for decoding LDPC codes with optimal performance. *Electron. Lett.* **2009**, *45*, 116–117. [CrossRef]
24. Wang, Z.; Wu, B.; Ye, T.C. An improved implementation of sum-product algorithm for LDPC decoder. *IEICE Electron. Express* **2019**, *16*, 20180828. [CrossRef]

25. Fan, J.; Yang, H. A new forced convergence decoding scheme for LDPC codes. In Proceedings of the 2009 IEEE International Conference on Communications Technology and Applications, Beijing, China, 16–18 October 2009; pp. 576–580. [CrossRef]
26. Dai, L.Y.; Yang, H.W.; Fan, J.X.; Rao, W.Y. Forced-convergence decoding for LDPC-coded modulation. *Sci. China Inf. Sci.* **2013**, *56*, 1–11. [CrossRef]
27. Shin, D.; Heo, K.; Oh, S.; Ha, J. A Stopping Criterion for Low-Density Parity-Check Codes. In Proceedings of the IEEE 65th Vehicular Technology Conference, VTC Spring 2007, Dublin, Ireland, 22–25 April 2007; pp. 1529–1533. [CrossRef]
28. Marchand, C.; Boutillon, E. Before convergence early stopping criterion for inner LDPC code in DVB standards. *Electron. Lett.* **2014**, *51*, 114–116. [CrossRef]
29. Zhao, M.; Zhang, X.L.; Zhao, L.; Chen, L. Design of a high-throughput QC-LDPC decoder with TDMP scheduling. *IEEE Trans. Circuits Syst. II Express Briefs* **2014**, *62*, 56–60. [CrossRef]
30. Declercq, D.; Savin, V.; Boncalo, O.; Ghaffari, F. An imprecise stopping criterion based on in-between layers partial syndromes. *IEEE Commun. Lett.* **2018**, *22*, 13–16. [CrossRef]
31. Savin, V. Self-corrected min-sum decoding of ldpc codes. In Proceedings of the 2008 IEEE International Symposium on Information Theory, Toronto, ON, Canada, 6–11 July 2008; pp. 146–150. [CrossRef]
32. Boncalo, O.; Alexandria, A.; Savin, V. Memory efficient implementation of self-corrected min-sum LDPC decoder. In Proceedings of the 21st IEEE International Conference on Electronics, Circuits and Systems (ICECS), Marseille, France, 7–10 December 2014; pp. 295–298. [CrossRef]
33. Boncalo, O.; Amaricai, A.; Mihancea, P.F.; Savin, V. Memory trade-offs in layered self-corrected min-sum LDPC decoders. *Analog Integr. Circuits Signal Process.* **2016**, *87*, 169–180. [CrossRef]
34. Diamantopoulos, D.; Xydis, S.; Siozios, K.; Soudris, D. Dynamic memory management in vivado-hls for scalable many-accelerator architectures. In Proceedings of the 11th International Symposium on Applied Reconfigurable Computing, Bochum, Germany, 14–17 April 2015; pp. 117–128. [CrossRef]
35. Salaskar, A.; Chandrachoodan, N. FFT/IFFT implementation using Vivado™ HLS. In Proceedings of the 20th IEEE International Symposium on VLSI Design and Test (VDAT), Guwahati, India, 24–27 May 2016; pp. 1–2. [CrossRef]