A NOVEL HARDWARE-FRIENDLY SELF-ADJUSTABLE OFFSET MIN-SUM ALGORITHM FOR ISDB-S2 LDPC DECODER

Wen Ji†, Makoto Hamaminato†, Hiroshi Nakayama†, and Satoshi Goto‡

†Fujitsu Laboratories LTD.
1-1, Kamikodanaka 4-chome, Nakahara-ku, Kawasaki, 211-8588, Japan
Email: {ji.wen, hamamy, h.nakayama}@jp.fujitsu.com
‡Graduate School of Information, Production and Systems, Waseda University
2-7 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka, 808-0135, Japan
Email: goto@waseda.jp

ABSTRACT

In this paper, a novel self-adjustable offset min-sum LDPC decoding algorithm is proposed for ISDB-S2 (Integrated Services Digital Broadcasting via Satellite - Second Generation) application. We present for the first time a uniform approximation of the check node operation through mathematical induction on Jacobian logarithm. The approximation theoretically shows that the offset value is mainly dependent on the difference between the two most unreliable inputs from the bit nodes and the algorithm proposed can adjust the offset value according to the inputs during the iterative decoding procedure. Simulation results for all 11 code rates of ISDB-S2 demonstrate that the proposed method can achieve an average of 0.15dB gain under the same Bit Error Rate (BER) performance, compared to the Min-sum based algorithms, and consumes only 1.21% computation complexity compared to BP-based algorithms in the best case.

1. INTRODUCTION

Low Density Parity Check (LDPC) code is an error correcting code first discovered in 1963 by Gallager [1], and rediscovered by Mackay and Neal in 1996 [2]. In Japan, a next generation satellite broadcasting system named "Integrated Services Digital Broadcasting via Satellite - Second Generation (ISDB-S2)" was proposed by NHK (Japan Broadcasting Corporation), and is currently under the examination of Association of Radio Industries and Businesses (ARIB) [3]. To ensure the transmission quality and high error correction capability, LDPC code is selected as the error correction code for ISDB-S2 and is expected to achieve a Bit Error Rate (BER) of 10^{-11}.

LDPC code can be efficiently decoded through messages exchange between check nodes and bit nodes by performing check node and bit node operations iteratively. Among decoding algorithms, Belief Propagation (BP) algorithm, also known as Sum Product algorithm, is well known for its good error correcting performance. However it is not hardware-friendly due to the necessity of implementing Hyperbolic functions [1]. Min-sum (MS) algorithm approximates BP algorithm with easy hardware implementation but greatly degrades the error correcting performance [4].

Recently, many approaches have been proposed to trade off between the BER performance and hardware complexity. These approaches can be categorized as two kinds of schemes: MS-based schemes and BP-based schemes. The MS-based schemes aim at improving the error correcting performance of the MS algorithm by introducing a multiplied or additive factor, *i.e.*, Normalized Min-sum (NMS) algorithm and Offset Min-sum (OMS) algorithm [5]. Later on, some further derivatives of OMS algorithm appear, such as the Degree-Matched Min-sum (DMMS) algorithm [6] which associates the offset with the degree of the check node, and the Adaptive Offset Min-sum (AOMS) algorithm [7] which adapts the offset according to the most unreliable information sent from the bit nodes. BP-based schemes, on the other hand, approximate the BP algorithm by calculating the Hyperbolic function term by term using Jacobian logarithm, such as Modified Min-sum (MMS) algorithm and Delta Min (DM) algorithm [8, 9].

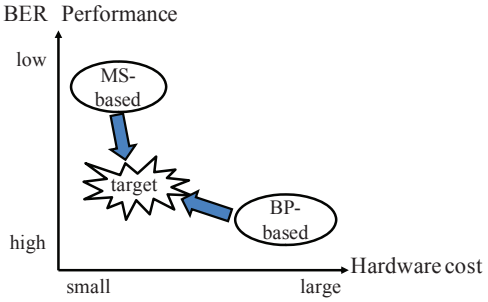

Figure 1: Requirement for ISDB-S2 LDPC decoder

Generally, BP-based algorithms outperform MS-based algorithms in BER performance, but they require larger hardware cost due to the iterative term-based implementation, as shown in Figure 1. Specifically, for the LDPC codes in ISDB-S2, a maximum of 90 times of computation complexity is introduced compared to MS algorithm, which directly increases the hardware overhead and power consumption. As far as the high BER performance requirement of practical ISDB-S2 application is concerned, MS-based algorithms are not competent enough. On the other hand, the hardware and power overhead of BP-based algorithms also limit their practical usage for the highly parallel implementation of ISDB-S2 LDPC decoder. Therefore, a decoding scheme which can achieve a similar BER performance as BP-based algorithms while maintaining the low hardware cost, will become the trend of future LDPC decoder design for next generation satellite applications.

Motivated by this challenging design task, we proposed a hybrid decoding scheme as an initial attempt for both high BER performance and low hardware cost design. The algorithm improves the OMS algorithm by a uniform approximation to the check node computation while the approximation is derived through mathematical induction on Jacobian logarithm, adopted widely by BP-based algorithms. It utilizes a self-adjustable offset based on the difference of the two most unreliable input values from the bit nodes. The simulation results further demonstrate that the proposed method can not only improve the BER performance compared to the MS-based schemes with nearly no overhead in hardware cost, but also consumes far less hardware than the BP-based schemes.

The rest of the paper is organized as follows. Section 2 introduces the LDPC codes used in ISDB-S2. Section 3 describes LDPC decoding algorithms in detail. Section 4 discusses the proposed algorithm, its simulation result and hardware cost analysis, and finally Section 5 concludes.

2. ISDB-S2 LDPC CODES

LDPC codes can be defined by a parity check matrix H_{MN}, where M and N are the number of rows and columns respectively. Defined by ISDB-S2, the parity check matrices targeted in this work are 11 different ones with code rate ranging from $\frac{1}{4}$ to $\frac{9}{10}$. The numbers of columns (N) for all 11 codes are fixed as 44,880 and the numbers of

 ISSN 2076-1465

rows (M) are related to the code rate. And all the LDPC codes are structured QC LDPC codes, with a sub-block size of 374×374 [3].

3. LDPC DECODING ALGORITHMS

Generally, LDPC decoding can be performed by the Two Phase Message Passing (TPMP) scheduling [2] or the layered scheduling (shuffled scheduling) [10]. The latter one was proved to be efficient for structured QC LDPC codes and converges two times faster than TPMP schedule. We utilize this layered schedule to meet the high performance requirement of ISDB-S2 application.

Layered LDPC decoding algorithm is mainly composed of three operations, *i.e.*, bit node operation, check node operation and A Posteriori Probability (APP) update operation. Let α_{mn} be the message sent from check node m to bit node n, β_{mn} be the message sent from bit node n to check node m, and sum_n be the APP message of the bit node n of the codeword and be initialized as λ_n, which is the Log-Likelihood Ratios (LLR) of the received codeword from the channel. Then, the bit node operation, check node operation and APP update operation of the BP algorithm can be expressed as Equation (1), Equation (2) and Equation (3), respectively. Note that $A(m)$ is defined as $A(m) = \{n | H_{mn} = 1\}$.

$$\beta_{mn} = sum_n - \alpha_{mn} \tag{1}$$

$$\alpha_{mn} = 2\tanh^{-1}\Big(\prod_{n' \in A(m)\backslash n} \tanh\big(\frac{\beta_{mn'}}{2}\big)\Big) \tag{2}$$

$$sum_n = \beta_{mn} + \alpha_{mn} \tag{3}$$

Since the check node function of BP algorithm is not hardware friendly, varies researches have been done to approximate the BP algorithm for better hardware implementation.

3.1 MS-based approximation

A simple approximation to Equation (2) is called Min-Sum algorithm which uses the minimum magnitude of input β as a replacement of the Hyperbolic functions, as shown in Equation (4).

$$\alpha_{mn} = \prod_{n' \in A(m)\backslash n} sign(\beta_{mn'}) \times \min_{n' \in A(m)\backslash n} |\beta_{mn'}| \tag{4}$$

Although MS algorithm can be easily implemented in hardware, it suffers a large performance degrading which encourages further researches to find better approximation based on the MS algorithm. For instance, a normalization factor or offset factor is applied to the MS algorithm, which forms the well-known Normalized MS algorithm and Offset MS algorithm, as shown in Equation (5) and (6) [5].

$$\alpha_{mn} = \gamma \prod_{n' \in A(m)\backslash n} sign(\beta_{mn'}) \times \min_{n' \in A(m)\backslash n} |\beta_{mn'}| \tag{5}$$

$$\alpha_{mn} = \prod_{n' \in A(m)\backslash n} sign(\beta_{mn'}) \times \max\Big(\min_{n' \in A(m)\backslash n} |\beta_{mn'}| - \varepsilon, 0\Big) \tag{6}$$

Note that the normalization factor γ and offset factor ε is not subject to change during the decoding procedure. Some recent progress claims that techniques to adjust the offset factor according to either the degree of the check node (DMMS algorithm [6]) or the minimum output data from the check node (AOMS algorithm [7]) can achieve better performance. However, DMMS requires significant computation power to determine the offset factor while the AOMS lacks sufficient theoretical evidence to support its approximation.

3.2 BP-based approximation

We first denote a basic computation in the check node operation of BP algorithm (Equation (2)) as function $\otimes$:

$$2\tanh^{-1}(\tanh\frac{\beta_1}{2} \times \tanh\frac{\beta_2}{2}) = \beta_1 \otimes \beta_2 \tag{7}$$

Therefore, Equation (2) can be simplified as Equation (8).

$$2\tanh^{-1}\Big(\prod_{n' \in A(m)\backslash n} \tanh\big(\frac{\beta_{mn'}}{2}\big)\Big) = \underbrace{\beta_{m1} \otimes \beta_{m2} \otimes \ldots \otimes \beta_{mn'}}_{|A(m)\backslash n|} \tag{8}$$

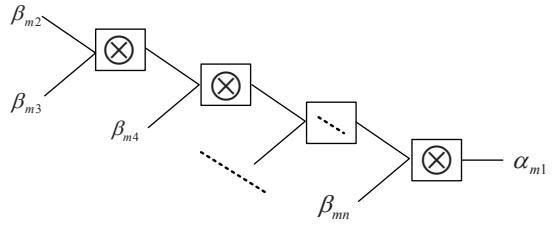

Figure 2: Iterative calculation for row operation using BP-based scheme

Equation (7), the primitive form of Equation (8), can be expanded using Jacobian Logarithm ($\ln(e^a + e^b) = \max(a,b) + \ln(1 + e^{-|a-b|})$) twice as follows [8]:

$$\begin{aligned} &\beta_1 \otimes \beta_2 \\ =&sign(\beta_1)sign(\beta_2)\big((\min(|\beta_1|,|\beta_2|) \\ &+ f(|\beta_1| + |\beta_2|) - f(|\beta_1| - |\beta_2|)\big) \end{aligned} \tag{9}$$

where function f(x) is defined as $f(x) = ln(1 + e^{-|x|})$.

Since f(x) is not hardware friendly, several works focus on the approximation of Equation (9). An MMS algorithm is proposed in [8] with Equation (10) as a substitution of Equation (9). Similarly, a DM algorithm is proposed in [9] using Equation (11) to calculate the parameter D in Equation (10).

$$\beta_1 \otimes \beta_2 = sign(\beta_1)sign(\beta_2)\big(\max(\min(|\beta_1|,|\beta_2|) - D, 0)\big)$$

$$where\ D = \begin{cases} 0.5 & |\beta_1 + \beta_2| \le 1\ \&\ |\beta_1 - \beta_2| > 1 \\ -0.5 & |\beta_1 - \beta_2| \le 1\ \&\ |\beta_1 + \beta_2| > 1 \\ 0 & \text{else} \end{cases} \tag{10}$$

$$D = \max\big((0.9 - \frac{\delta}{2}), 0\big)\ where\ \delta = \big||\beta_1| - |\beta_2|\big| \tag{11}$$

Equation (10) and Equation (11) are then applied iteratively for the check node operation (Equation (8)). Figure 2 demonstrates this iterative computation process for message α_{m1}. In each iteration, the $\otimes$ function of the intermediate result and a β message is calculated. Therefore, for each α value, a total of $(|A(m)| - 2)$ $\otimes$ computations are required. Since altogether there are $|A(m)|$ α values to be calculated in one row, the computation complexity of the check node operation is proportional to $|A(m)| \times (|A(m)| - 2)$, which is relatively large for some codes in ISDB-S2.

4. PROPOSED ALGORITHM

In this section, a novel self-adjustable offset min-sum algorithm is proposed, in which a uniform approximation for the check node operation of the BP algorithm is developed through mathematical induction on Jacobian logarithm. The effectiveness of the proposed approximation is demonstrated by the simulation results of all the 11 parity check matrices in ISDB-S2, showing a better BER performance than MS-based schemes. The computation complexity and area cost are also analyzed to further exhibit that the proposed algorithm has much smaller hardware cost than the BP-based schemes.

4.1 Proposed approximation of BP algorithm

In order to reduce the computation complexity of check node operation, we first consider a general case as shown in Equation (12). Note that the general case is targeted here by considering $n' \in A(m)$ rather than $n' \in A(m)\backslash n$ in Equation (2). The exact calculation of α_{mn} will be explained after the uniform approximation is derived.

$$2\tanh^{-1}\Big(\prod_{n' \in A(m)} \tanh\big(\frac{\beta_{mn'}}{2}\big)\Big) = \underbrace{\beta_{m1} \otimes \beta_{m2} \otimes \ldots \otimes \beta_{mn'}}_{|A(m)|} \tag{12}$$

Since function $\otimes$ holds commutative law, we can fairly assume that $|\beta_{m1}| < |\beta_{m2}| < \ldots < |\beta_{mn'}|$. Under this assumption, Equation

(12) can be further expanded as Equation (13) through a mathematical induction based on Equation (9).

$$\begin{aligned}
&2\tanh^{-1}\big(\tanh(\frac{\beta_{m1}}{2})\ldots\tanh(\frac{\beta_{mn'}}{2})\big)\\
\approx&sign(\beta_{m1})\ldots sign(\beta_{mn'})\big(\min(|\beta_{m1}|,\ldots,|\beta_{mn'}|)\\
&-f(|\beta_{m2}|-|\beta_{m1}|)-f(|\beta_{m3}|-|\beta_{m1}|)-\ldots-f(|\beta_{mn'}|-|\beta_{m1}|)\\
&+f(|\beta_{m2}|+|\beta_{m1}|)+f(|\beta_{m3}|+|\beta_{m1}|)+\ldots+f(|\beta_{mn'}|+|\beta_{m1}|)\big)
\end{aligned}\tag{13}$$

The detailed proof of Equation (13) is listed below.

---

(1)The condition of $n'=2$ is already proved in Section 3.2.

(2)Suppose $n'=k$ is correct, consider the situation of $n'=k+1$

$$\tanh(\frac{\beta_{m1}}{2})\ldots\tanh(\frac{\beta_{mk}}{2})=X$$

$$2\tanh^{-1}\big(\tanh(\frac{\beta_{m1}}{2})\ldots\tanh(\frac{\beta_{mk}}{2})\big)=Y$$

$$\Rightarrow X=\tanh(\frac{Y}{2})$$

$$\begin{aligned}
&2\tanh^{-1}\big(\tanh(\frac{\beta_{m1}}{2})\ldots\tanh(\frac{\beta_{mk}}{2})\tanh(\frac{\beta_{m(k+1)}}{2})\big)\\
=&2\tanh^{-1}\big(\tanh(\frac{Y}{2})\tanh(\frac{\beta_{m(k+1)}}{2})\big)\\
=&sign(Y)sign(\beta_{m(k+1)})\big(\min(|Y|,|\beta_{m(k+1)}|)\\
&+f(|Y|+|\beta_{m(k+1)}|)-f(|\beta_{m(k+1)}|-|Y||)\big)\\
=&sign(\beta_{m1})\ldots sign(\beta_{mk})sign(\beta_{m(k+1)})\\
&\big(\min(\min(|\beta_{m1}|,\ldots,|\beta_{mk}|)\\
&-f(|\beta_{m2}|-|\beta_{m1}|)-\ldots-f(|\beta_{mk}|-|\beta_{m1}|)\\
&+f(|\beta_{m2}|+|\beta_{m1}|)+\ldots+f(|\beta_{mk}|+|\beta_{m1}|),\\
&|\beta_{m(k+1)}|)-f(|\beta_{m(k+1)}|-|Y|)+f(|\beta_{m(k+1)}|+|Y|)\big)
\end{aligned}$$

$$\begin{aligned}
&\because\min(|\beta_{m1}|,\ldots,|\beta_{mk}|)\\
&=\min(|\beta_{m1}|,\ldots,|\beta_{mk}|,|\beta_{m(k+1)}|)=|\beta_{m1}|\\
&|Y|\approx\min(|\beta_{m1}|,\ldots,|\beta_{mk}|)=|\beta_{m1}|\\
\therefore&2\tanh^{-1}\big(\tanh(\frac{\beta_{m1}}{2})\ldots\tanh(\frac{\beta_{mk}}{2})\tanh(\frac{\beta_{m(k+1)}}{2})\big)\\
\approx&sign(\beta_{m1})\ldots sign(\beta_{m(k+1)})\big(\min(|\beta_{m1}|,\ldots,|\beta_{mk}|,|\beta_{m(k+1)}|)\\
&-f(|\beta_{m2}|-|\beta_{m1}|)-f(|\beta_{m3}|-|\beta_{m1}|)-\ldots-f(|\beta_{m(k+1)}|-|\beta_{m1}|)\\
&+f(|\beta_{m2}|+|\beta_{m1}|)+f(|\beta_{m3}|+|\beta_{m1}|)+\ldots+f(|\beta_{m(k+1)}|+|\beta_{m1}|)\big)
\end{aligned}$$

Since $|A(m)|$ is usually a large number for ISDB-S2, the implementation of Equation (13) requires a large amount of hardware resources. Hence an efficient approximation to the equation to reduce hardware cost is a necessity. Based on the characteristics of function f(x), we find out that f(x) is a monotonically decreasing function with $f(x)\doteq 0$ when $x>2.5$. Because of the relationships among $|\beta_{m1}|,\ldots,|\beta_{mn}|$, we can derive that $|\beta_{m2}|-|\beta_{m1}|$ is the smallest one among all the arguments of $f(x)$ in the equation, thus $-f(|\beta_{m2}|-|\beta_{m1}|)$ becomes the dominant term of all the function $f(x)$ terms. We can easily figure out, through the above derivation, the offset term is mainly dependent on the two most unreliable inputs from the bit nodes which are denoted as β_{min1} and β_{min2} from now on. However, simply keeping the dominant term and ignoring all the other ones degrades the precision of computation. Therefore, we further approximate all the other ones by multiplying a normalization factor or adding an offset factor to the dominant term $-f(\beta_{min2}-\beta_{min1})$. In this work, we use the normalization factor γ' and obtain Equation (14) as an approximation to Equation (13).

$$\begin{aligned}
&2\tanh^{-1}\big(\tanh(\frac{\beta_{m1}}{2})\ldots\tanh(\frac{\beta_{mn'}}{2})\big)\\
\approx&sign(\beta_{m1})\ldots sign(\beta_{mn'})\big(min(|\beta_{m1}|,|\beta_{m2}|,\ldots,|\beta_{mn'}|)\\
&-\gamma' f(|\beta_{min2}|-|\beta_{min1}|)\big)
\end{aligned}\tag{14}$$

As can be seen from Equation (2), the computation of α_{mn} is based on $\beta_{mn'}$ values with $n'\in A(m)\backslash n$. However, Equation (14) is derived considering the $\beta_{mn'}$ values with $n'\in A(m)$. In the following parts, we will discuss, in three different cases, how we derive the proposed approximation of Equation (2) from Equation (14).

- Case 1: $|\beta_{mn}|$ is $|\beta_{min1}|$, the smallest one among all absolute β values. In this case, $|\beta_{min1}|$ should not be included in the computation of α_{mn}. Therefore, $|\beta_{min2}|$ and $|\beta_{min3}|$ become the minimum value and second minimum value among all $\beta_{mn'}(n'\in A(m)\backslash n)$. Hence,
$$\alpha_{mn}=\prod_{n'\in A(m)\backslash n}sign(\beta_{mn'})\big(|\beta_{min2}|-\gamma' f(|\beta_{min3}|-|\beta_{min2}|)\big)$$
- Case 2: $|\beta_{mn}|$ is $|\beta_{min2}|$, the second minimum value among all absolute β values. In this case, $|\beta_{min1}|$ and $|\beta_{min3}|$ become the minimum value and second minimum value among all $\beta_{mn'}(n'\in A(m)\backslash n)$. Therefore,
$$\alpha_{mn}=\prod_{n'\in A(m)\backslash n}sign(\beta_{mn'})\big(|\beta_{min1}|-\gamma' f(|\beta_{min3}|-|\beta_{min1}|)\big)$$
- Case 3: $|\beta_{mn}|$ is neither $|\beta_{min1}|$ nor $|\beta_{min2}|$. In this case $|\beta_{min1}|$ and $|\beta_{min2}|$ are still the minimum value and second minimum value among all $\beta_{mn'}(n'\in A(m)\backslash n)$. Therefore,
$$\alpha_{mn}=\prod_{n'\in A(m)\backslash n}sign(\beta_{mn'})\big(|\beta_{min1}|-\gamma' f(|\beta_{min2}|-|\beta_{min1}|)\big)$$

So altogether three cases should be considered to implement Equation (14), which gives rise to additional design overhead. To solve the problem, we further simplify the check node operation. Through simulation, we notice that the computation of $|\beta_{min3}|-|\beta_{min2}|$ in Case 1 can be approximated as $|\beta_{min2}|-|\beta_{min1}|$, and using $-\gamma' f(|\beta_{min2}|-|\beta_{min1}|)$ instead of $-\gamma' f(|\beta_{min3}|-|\beta_{min1}|)$ for Case 2 incurs nearly no performance degrading. Hence, we combine three cases into one uniform expression shown in Equation (15), which greatly reduces the hardware implementation cost.

$$\begin{aligned}
\alpha_{mn}\approx&\prod_{n'\in A(m)\backslash n}sign(\beta_{mn'})\big(\min_{n'\in A(m)\backslash n}|\beta_{mn'}|\\
&-\gamma' f(|\beta_{min2}|-|\beta_{min1}|)\big)
\end{aligned}\tag{15}$$

From Equation (15), we can see that the offset factor is self-adjustable, during the iterative decoding, according to the difference of the two most unreliable inputs from the bit nodes. Such adjustable scheme precisely models the variations of bit node messages, hence enhances the decoding efficiency.

4.2 Simulation results

Software simulation of the proposed decoding algorithm has been conducted for all 11 parity check matrices used in ISDB-S2. The QPSK modulation and AWGN channel is modeled in the simulation. A total of 10,771,200 input bits are used for simulation. The maximum number of iteration is set to 50, and the simulation program terminates when the decoded codeword is a valid one or the maximum iteration times are achieved.

Figure 3 and Figure 4 illustrate the simulation result of the BER performance of BP, NMS, OMS, DMMS, AOMS, MMS, DM and the proposed decoding algorithm using layered scheduling for rate $\frac{3}{5}$ and $\frac{3}{4}$, which will be mainly used in ISDB-S2 service. Except BP algorithm is simulated using floating values, all the intermediate messages of simulations for the other algorithms are coded in 6 bit sign-magnitude format and the APP message is realized in an 8 bit sign-magnitude format to avoid overflow. The parameters of all algorithms are chosen to optimize both the BER performance and hardware implementation as $\gamma=0.875$ for NMS (Equation (5)), $\varepsilon=0.125$ for OMS (Equation (6)), and $\gamma'=0.125$ for the proposed method (Equation (15)). Also, for simple hardware implementation, we use the same Δ function $\Delta(x)=\max(\frac{5}{8}-\frac{|x|}{4},0)$ as [10] for approximation of function $f(x)$ for the proposed algorithm in this work. It can be observed from the figure that the proposed algorithm achieves an average of 0.2dB gain compared to the MS-based algorithms, and sometimes even outperforms BP-based algorithms.

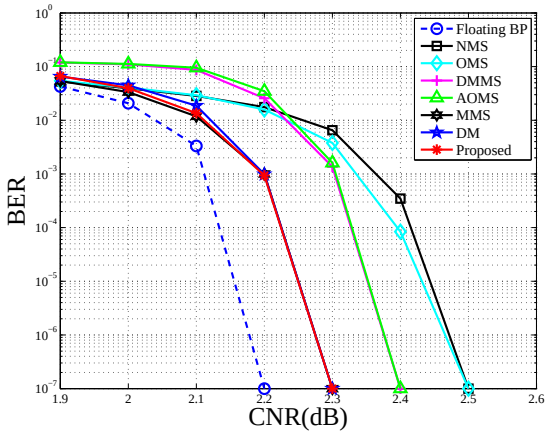

Figure 3: BER performance for rate $\frac{3}{5}$

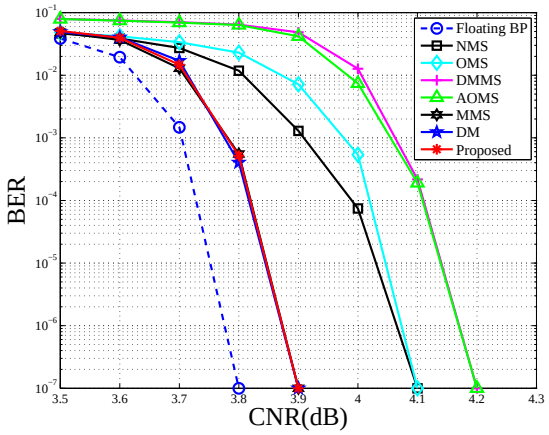

Figure 4: BER performance for rate $\frac{3}{4}$

4.3 Comparison of required CNR

In order to further analyze the efficiency of the proposed algorithm and its suitability to all the LDPC codes in ISDB-S2, we use a metric, called the required CNR. The required CNR is defined as the carrier-to-noise ratio when the BER exceeds 10^{-11} for ISDB-S2 [11]. Because of the error floor free performance of ISDB-S2 code and relatively long computer simulation time to evaluate the BER downto the range of 10^{-11}, in this work, we use the same evaluation method as [11], namely extrapolation, to calculate the required CNR. The simulation uses 10^7 input data, if no error can be found in the simulation, it is fair to say that this point is free of error at $BER = 10^{-7}$. We call this point "BER = 0 Observation Point", as shown in Figure 5. In this figure, P1 and P2 are simulation points obtained from the computer simulation result. P3 is the BER=0 Observation Point and P4 is the point with the required CNR (CNR4). We calculate CNR4 as shown in Equation (16) using the extrapolation technique.

$$CNR4 = 2 \cdot \frac{log(10^{-11}) - log(BER3)}{\frac{log(BER2)-log(BER1)}{CNR2-CNR1} + \frac{log(BER3)-log(BER2)}{CNR3-CNR2}} + CNR3 \tag{16}$$

The results of the required CNR are listed in Table 1 for BP algorithm with TPMP scheduling [11], BP with layered scheduling, and all other algorithms discussed in this paper with layered scheduling. Except that BP algorithms use floating simulations, other algorithms include a 6-bit quantization. The row ΔBP(TPMP) in the table indicates the average differences of required CNR for all the code rates compared to the BP(TPMP) algorithm. As can be seen from Table 1, the proposed algorithm is only 0.147dB away from the standard BP algorithm [11], and is about 0.15dB better than the MS-based algorithms in average.

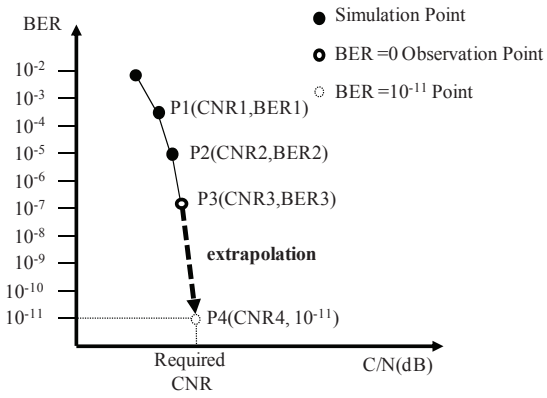

Figure 5: Required CNR calculation using extrapolation

4.4 Comparison of Computation Complexity and Hardware Cost

Although the BER performance of BP-based algorithms outperforms the proposed algorithm, their computation complexity and hardware cost can not be neglected. The comparison of computation complexity and the hardware cost of the check node operation for one row (exclude the sign computation) are listed in Table 2. Under column *computation complexity*, $[comp]$, $[add]$, $[shift]$ indicate the computation complexity of comparison operation, addition or subtraction operation, and shift operation, respectively. For MS-based algorithm, $|A(m)|$ items are compared serially to get the minimum and the second minimum value, so $2 \times |A(m)| \times [comp]$ is needed. After that, normalization factor or offset factor is applied to minimum and second minimum value, so additional calculations for the normalization factor γ and offset factor ε in Equation (5) and Equation (6) are needed. For the proposed algorithm, after the minimum value and the second minimum value are found, according to Equation (15), we require two more subtraction, two more shift operation and 2 subtraction for offset. For BP-based algorithm, Equation (10) or Equation (11) is invoked $(|A(m)| - 2)$ times for each $n(n \in A(m))$ and a total of $|A(m)|$ different n values, thus requiring $|A(m)| \times (|A(m)| - 2)$ times of Equation (10) or Equation (11). For rate $\frac{9}{10}$ with the biggest row weight $|A(m)|$ of 32 among all the parity check matrices in ISDB-S2, the computation complexity relation between NMS, OMS, DMMS, AOMS, MMS, DM and the proposed method is 1.03 : 1.03 : 1.22 : 1.05 : 90 : 75 : 1.09. The computation complexity for the proposed algorithm is similar to MS-based algorithms, and much smaller compared to BP-based algorithms. Figure 6(a) shows the relation of average computation complexity and average required CNR for all the rates in ISDB-S2. From the figure, we can see that the proposed algorithm consumes much less computation complexity compared to the BP-based algorithms but can achieve much better error correcting performance compared to MS-based algorithm with almost the same computation complexity.

We also estimate the hardware cost for one check node operation (exclude the sign operation) using gate counts. The estimation results are listed under column *hardware cost* with $[adder5]$ and $[adder6]$ indicating the cost of an adder or subtractor for 5 bits and 6 bits. Note that we assume a comparator shares a similar cost with an adder, and we neglected the cost for shifter. To keep almost the same clock cycles for one check node operation for all algorithms, MMS and DM require a parallel implementation of comparison, thus making the hardware cost almost $|A(m)|$ times as the other algorithms. In Figure 6(b), we show the relation of area cost and average required CNR for all the rates in ISDB-S2. The adder is estimated as 6 gates per bit and the LUT is estimated as 10 gates per bit. The figure demonstrates a similar trend as Figure 6(a) that the proposed algorithm greatly reduces the area compared to the BP-based algorithms while achieves much better error correcting performance than MS-based algorithms.

Table 1: Comparison of Required CNR (dB)

Rate	floating BP		MS-based				BP-based		proposed
	TPMP [11]	Layered	NMS	OMS	DMMS	AOMS	MMS	DM	
1/4	-2.1	-2.15	-1.56	-1.33	-1.49	-1.52	-1.73	-1.56	-1.55
1/3	-1.0	-1.14	-0.60	-0.45	-0.42	-0.41	-0.94	-0.86	-0.67
2/5	0.0	-0.16	0.31	0.36	0.44	0.54	0.04	0.13	0.34
1/2	1.2	1.05	1.42	1.45	1.45	1.44	1.14	1.24	1.38
3/5	2.5	2.35	2.67	2.67	2.55	2.54	2.46	2.45	2.45
2/3	3.3	3.27	3.47	3.46	3.34	3.34	3.26	3.25	3.25
3/4	4.0	3.95	4.29	4.26	4.36	4.36	4.06	4.05	4.05
4/5	5.0	4.97	5.28	5.37	5.15	5.16	5.08	4.95	5.07
5/6	5.5	5.48	5.79	6.01	5.66	5.66	5.46	5.46	5.45
7/8	5.9	5.97	6.16	6.08	6.15	6.15	6.01	5.96	6.07
9/10	6.8	6.79	7.09	6.89	6.87	6.87	6.78	6.78	6.88
$\Delta BP(TPMP)$	**0**	**-0.065**	**0.293**	**0.335**	**0.269**	**0.275**	**0.047**	**0.068**	**0.147**

Table 2: Comparison of computation complexity and hardware cost

	Computation Complexity	Hardware Cost
NMS	$2\times\lvert A(m)\rvert\times[comp]+2\times[shift]$	$2\times[adder5]$
OMS	$2\times\lvert A(m)\rvert\times[comp]+2\times[add]$	$4\times[adder5]$
DMMS	$(2\times\lvert A(m)\rvert+4)\times[comp]+3\times[shift]+7\times[add]$	$5\times[adder5]+6\times[adder6]$
AOMS	$2\times\lvert A(m)\rvert\times[comp]+1\times[shift]+2\times[add]$	$4\times[adder5]+8word\times 3bit[LUT]$
MMS	$\lvert A(m)\rvert\times(\lvert A(m)\rvert-2)\times(3\times[comp]+3\times[add])$	$\lvert A(m)\rvert\times(4\times[adder6]+2\times[adder5])$
DM	$\lvert A(m)\rvert\times(\lvert A(m)\rvert-2)\times(1\times[comp]+3\times[add]+1\times[shift])$	$\lvert A(m)\rvert\times(4\times[adder5])$
proposed	$2\times\lvert A(m)\rvert\times[comp]+4\times[add]+2\times[shift]$	$6\times[adder5]$

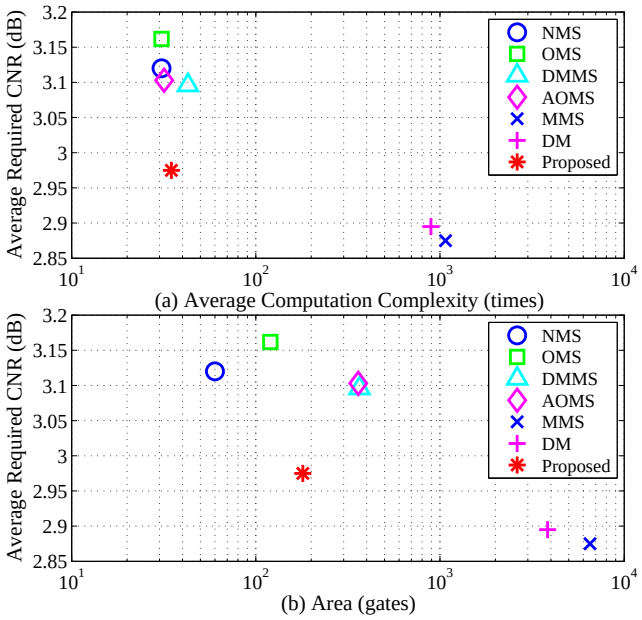

Figure 6: Performance comparison: (a) average required CNR vs. average computation complexity, and (b) average required CNR vs. area

5. CONCLUSION

In this paper, in order to achieve high BER performance for satellite transmission services, a novel self-adjustable offset min-sum algorithm is proposed with the check node operation approximating BP algorithm. The correctness of the approximation is proved by mathematical induction through using Jacobian logarithm iteratively. The proposed algorithm is hardware-friendly compared to the BP-based algorithms and the simulation results show that the proposed algorithm can achieve an average of 0.15dB gain compared to Min-sum based algorithms.

REFERENCES

[1] R. G. Gallager, *Low-Density Parity-Check Codes*. MIT Press, Cambridge, MA, 1963.

[2] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, March 1999.

[3] A. Hashimoto and Y. Suzuki, "A new transmission system for the advanced satellite broadcast," *IEEE Trans. Consum. Electron.*, vol. 54, Issue 2, pp. 353–360, May 2008.

[4] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Trans. Commun.*, vol. 47, no. 5, pp. 673–680, May 1999.

[5] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1288–1299, August 2005.

[6] S. L. Howard, C. Schlegel, and V. C. Gaudet, "Degree-matched check node decoding for regular and irregular LDPCs," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 53, no. 10, pp. 1054–1058, October 2006.

[7] M. Jiang, C. Zhao, L. Zhang, and E. Xu, "Adaptive offset min-sum algorithm for low-density parity check codes," *IEEE Communications Letters*, vol. 10, no. 6, pp. 483–485, June 2006.

[8] A. Anastasopoulos, "A comparison between the sum-product and the min-sum iterative detection algorithms based on density evolution," *IEEE Global Telecommunications Conference (GLOBECOM)*, San Antonio, TX, vol. 2, pp. 1021–1025, November 2001.

[9] L. Sakai, W. Matsumoto, and H. Yoshida, "Reduced complexity decoding based on approximation of update function for low-density parity-check codes," *IEICE Trans. Fundamentals*, vol. J90-A, no. 2, pp. 83–91, Feberary 2007.

[10] M. M. Mansour and N. R. Shanbhag, "High-throughput LDPC decoders," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 6, pp. 976–996, December 2003.

[11] Y. Suzuki, A. Hashimoto, M. Kojima, S. Tanaka, T. Kimura, and T. Saito, "LDPC codes for the advanced digital satellite broadcasting system," *IEICE Technical Report*, vol. 109, no. 212, pp. 19–24, September 2009.