Research Article

Novel Early Termination Method of an ADMM-Penalized Decoder for LDPC Codes in the IoT

Biao Wang

School of Mathematics and Information Science, Baoji University of Arts and Sciences, Baoji 721013, China

Correspondence should be addressed to Biao Wang; wangbiao401@sina.com

Received 10 January 2022; Revised 3 August 2022; Accepted 17 August 2022; Published 14 October 2022

Academic Editor: Marimuthu Karuppiah

Copyright © 2022 Biao Wang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

As a critical communication technology, low-density parity-check (LDPC) codes are widely concerned with the Internet of things (IoT). To increase the convergence rate of the alternating direction method of multiplier (ADMM)-penalized decoder for LDPC codes, a novel early termination (ET) method is presented by computing the average sum of the hard decision (ASHD) during each ADMM iteration. In terms of the flooding scheduling and layered scheduling ADMM-penalized decoders, the simulation results show that the proposed ET method can significantly reduce the average number of iterations at low signal-to-noise ratios (SNRs) with negligible decoding performance loss.

1. Introduction

The Internet of things (IoT) is an important part of the new generation of information technology, and it also represents an important stage of development in the era of “information technology” [1–4]. With the era of big data coming, IoT is widely used in many fields and will have a very beneficial impact on business, human lifestyle, and health [5]. As there are many vehicles and few parking spaces, the difficulty of parking is a common problem faced by many cities at present. Based on IoT, Abirami et al. designed a smart parking system that can well alleviate the parking problem [6]. The quality and supply efficiency of food are the key issues in the food industry and directly affect the healthy life of human beings. A dynamic food supply chain, which not only ensures the food quality but also provides intelligent vehicle path tracking, is proposed based on IoT [7].

As low-density parity-check (LDPC) codes exhibit excellent decoding performance, they are widely employed in IoT [8–11]. However, as certain nodes do not actively participate in forwarding messages, the communication cost increases [12]. To control the communication cost, it is necessary to study the related technology for communication systems.

Based on linear programming (LP), an alternative decoding method is designed for LDPC codes [13]. Though the LP method’s decoding performance is better than the message passing (MP) decoding, its decoding complexity is high. To reduce the decoding complexity of the LP method, an efficient LP decoding method based on the alternating direction method of multiplier (ADMM) (called the ADMM-based LP decoding method) is presented [14]. The complexity of the ADMM-based LP decoding method is similar to that of the belief propagation (BP) decoding method, but its decoding performance is worse at low signal-to-noise ratios (SNRs). Recently, many scholars have studied ADMM-based LP decoding to improve performance or reduce complexity.

To improve the performance of the ADMM-based LP decoding method at low SNRs, an improved ADMM decoding method (called the ADMM-penalized decoding method) that increases the cost of pseudocodewords is proposed [15]. A novel ADMM-penalized decoding method that significantly improves the decoding performance is also devised based on the weighted penalized factor [16]. The performance of the ADMM-penalized decoder with the improved penalty function devised by combining the $l_1$ and $l_2$ penalty functions can also be improved [17].
The most complex operation of the ADMM-based LP decoding method is the Euclidean projection onto the check polytope. Wei et al. reduced this method’s complexity by directly reducing the number of Euclidean projections [18]. Many other studies have also reduced the complexity by improving the Euclidean projection operation. For example, introducing the cut search algorithm [19], projecting a vector onto a simplex [20], using look-up tables to implement the Euclidean projection [21], removing the sorting operation [22], replacing the check polytope projection through line segment projection [23], exploiting the LP model’s orthogonality structure [24], and alternately adopting even-vertex and other accurate projection algorithms [25] have all helped reduce the complexity.

In addition, inspired by deep learning, Wei et al. presented a deep neural network-aided decoding algorithm that outperforms the original ADMM-penalized decoder [26]. Based on ADMM, Bai et al. proposed an efficient quadratic programming decoder that can eliminate the Euclidean projections [27]. To further improve the decoding performance, Jiao et al. derived the $\ell_2$-box ADMM decoding method for certain intersymbol interference (ISI) channels [28].

Another way to reduce the ADMM-based LP decoder’s complexity is the message scheduling strategy. For the ADMM decoding method, horizontal-layered scheduling [29], vertical-layered scheduling [30], and node-wise scheduling [31] are presented to increase the convergence rate.

The early termination (ET) method terminates the decoding process as early as possible before reaching the maximum number of iterations, and it can therefore increase the decoding method’s convergence rate. The authors in [32–36] provided many useful ET methods that have reduced the BP decoder’s complexity by avoiding unnecessary decoding processes. At present, the standard $\varepsilon$ rule and the ET scheme of $\mathbf{H}\mathbf{x}^T = 0$ are two primary stopping methods for ADMM decoding.

For LDPC codes, this paper designs a novel ET method that increases the convergence rate of the ADMM-penalized decoder, and its primary contributions are as follows:

1. An algorithm that computes the average sum of the hard decision (ASHD) of the ADMM-penalized decoding method during each iteration is presented.
2. Using the ASHD of the codeword bits at each decoding iteration, this paper designs a novel ET method that increases the ADMM penalized decoder’s convergence rate.
3. According to our simulations, the designed ET method can significantly reduce the average number of iterations at low SNRs with negligible decoding performance degradation.

The remainder of the paper is organized as follows: In terms of LDPC codes, the ADMM decoding problem model is briefly reviewed in Section 2. Section 3 designs a novel ET method for the ADMM-penalized decoder. The experimental simulations illustrate the effectiveness of the designed ET method in Section 4. Finally, Section 5 concludes this paper.
2.3. ADMM Penalized Decoding. After adding a penalty function, problem (2) becomes the ADMM penalized decoding:

\[
\begin{align*}
\min & \quad \gamma^T x + \alpha \sum_{i=1}^{N} g(x_i), \\
\text{s.t} & \quad T_j x = z_j, z_j \in P_{d_j}, \quad \forall j \in J,
\end{align*}
\]

(5)

where \( \alpha \) and \( g(x) \) are the penalty parameter and penalty function, respectively.

The flooding scheduling ADMM-penalized decoding method is expressed in Algorithm 1 [15], where \( \nabla_{x} f \) denotes the differential of \( f \) with respect to \( x \).

At present, the stop criterion in Algorithm 1 primarily adopts the standard \( \epsilon \) rule and the \( Hx^T = 0 \) ET rule. The first method compares the prime residual \( \sum_{j \in J} \| T_j x^k - z_j \|_2 \) and the dual residual \( \sum_{j \in J} \| T_j x^k - z_j^k \|_2 \) with the error tolerance \( \epsilon \) (\( \epsilon > 0 \)), and the ADMM decoding process terminates when their values are all less than \( \epsilon \) [14], namely, when \( \sum_{j \in J} \| T_j x^k - z_j \|_2 < \epsilon \) and \( \sum_{j \in J} \| z_j^k - z_j^{k-1} \|_2 < \epsilon \) are all true. The second term verifies the parity-check equation \( Hx^T = 0 \) at the end of each iteration, and if the equation is satisfied, the ADMM decoding process terminates [19].

3. Designed Early Termination Method

A novel ET method of the ADMM-penalized decoder is designed in this section. The LDPC codes \( C_1 \) (576, 288) and \( C_2 \) (1152, 288) are used to show the effectiveness of the designed ET method.

3.1. Motivation. The ADMM-penalized decoder is an iterative decoding method, so we can easily compute the hard decision information for the codeword bits at the end of each decoding iteration. At the \( k \)-th iteration, the \( i \)-th bit hard decision information \( H_i^k \) is decided as follows:

\[
H_i^k = \begin{cases} 
0, & \text{if } x_i^k < 0.5, \\
1, & \text{if } x_i^k \geq 0.5,
\end{cases}
\]

(6)

where \( x_i^k \) represents the soft value of \( x_i \) at the \( k \)-th iteration in Algorithm 1. According to \( H_i^k \), the ASHD at each decoding iteration can be computed as described in Algorithm 2. The ASHD, which describes the iterative information of \( x \) at each ADMM iteration, can monitor the decoding process.

By running the flooding scheduling ADMM-penalized decoder, we obtained the changing ASHD trends for \( C_1 \) and \( C_2 \) under different SNRs, and the results are shown in Figure 1. It can be seen that the ASHD value for \( C_1 \) is relatively high at the 6-th, 7-th, and 8-th iterations. However, when the number of iterations increases, there is an obvious difference between the changing ASHD trends at different SNRs. For example, when SNR = 3.4 dB, the ASHD is close to stabilizing and has a small value, which may indicate that the exact solution vector has been found; therefore, we can prematurely stop decoding. However, the ASHD at SNR = 2.6 dB is still large and dispersed, which represents that a fraction of the hard decision information keeps changing and further iterations are needed to find the exact solution vector. Therefore, if the ASHD values remain relatively high, the decoding process can be terminated. Similar results were obtained for the \( C_2 \) ASHD, and they are also presented in Figure 1.

The changing ASHD trends for \( C_1 \) and \( C_2 \) with the layered scheduling [29] ADMM-penalized decoder under different SNRs is shown in Figure 2. Compared with Figure 1, the changing ASHD trends for \( C_1 \) and \( C_2 \) are roughly the same. The lower the number of iterations, the larger the value of the ASHD. When the number of iterations increases, the ASHD value gradually decreases; in addition, the value of the ASHD at higher SNRs is relatively stable, while the ASHD value at lower SNRs remains relatively dispersed. The difference between Figures 1 and 2 is that with layered scheduling, the number of iterations required to obtain roughly the same ASHD value is lower than that with flooding scheduling.

3.2. Algorithm Details. There are two critical parameters \( \text{Iter} \) and \( \text{Thre} \) that should be considered when employing the designed ET method. \( \text{Iter} \) denotes the number of iterations at which the ADMM decoder starts the ASHD check, and \( \text{Thre} \) is the ASHD threshold used to check whether the decoding process should be terminated. It should be noted that the value of \( \text{Iter} \) is very important in the designed ET method. As the ASHD value is relatively high at the 6-th, 7-th, and 8-th iterations, \( \text{Iter} \) should not be too small. However, \( \text{Iter} \) should also not be too large; otherwise, it will affect the designed ET method’s operations. To achieve a better decoding effect, these two parameters can be optimized by simulations using the designed ET method.

In terms of LDPC codes, Algorithm 3 describes the designed ET method for the ADMM-penalized decoder. Step 4(a) checks whether the number of iterations \( k \) exceeds the maximum number of decoding iterations \( \text{Iter}_{\text{max}} \). Sum\([k] = 0 \) in Step 4(b) implies that this iteration’s solution vector is the correct codeword. Finally, Step 4(c) verifies whether the conditions of the designed ET method are met.

It is worth noting that the designed ET method can be applied to both the flooding scheduling [15] and layered scheduling [29] ADMM-penalized decoders.

4. Simulation Results

We compared the frame error rate (FER) performance and the average number of iterations (ANIs) for the ADMM-penalized decoder with the designed ET method, the standard \( \epsilon \) rule, and the \( Hx^T = 0 \) ET rule. During simulations, the maximum number of iterations, error tolerance, and over-relaxation parameter are set to 40, \( 10^{-5} \), and 1.9, respectively. The \( l_1 \) penalized function is used in the simulation, and its penalty parameters are optimized by the method presented in [14].

4.1. Choice of \( \text{Iter} \) and \( \text{Thre} \). As \( \text{Iter} \) and \( \text{Thre} \) are the two critical parameters in the designed ET method, it is
particularly important to choose appropriate values. Therefore, we select appropriate parameter values for Iter and Thre in terms of the flooding scheduling and layered scheduling ADMM-penalized decoders, respectively.

4.1.1. For the Flooding Scheduling ADMM-Penalized Decoder. First, the influence of Thre is investigated when Iter = 10 for C1. Figure 3 shows the FER performance and the ANIs of the designed ET method with different values of
Figure 2: Changing the ASHD trends for $C_1$ and $C_2$ when using the layered scheduling ADMM-penalized decoder.

**Algorithm 3:** Designed ET method of the ADMM-penalized decoder for LDPC codes.

**Input:** $x = \{x_i|i \in I\}$ from the $k$-th ADMM iteration, the parameter values of $\text{Iter}$ and $\text{Thre}$.

1. For each component $x_i$, compute its hard decision $H^k_i$ at the end of the $k$-th iteration.
2. Compute the sum of hard decision, sum = $\sum_{i=1}^{n} H^k_i$, at the end of the $k$-th iteration.
3. Compute the $k$-th iteration’ average sum of hard decision, $AS[k] = \text{Sum}[k]/k$.
4. (a) Check whether $k > \text{Iter}_{\text{max}}$;
   (b) Check whether $\text{Sum}[k] = 0$;
   (c) Check whether $k > \text{Iter}$ and $AS[k] > \text{Thre}$.
5. If 4(a) is true, or 4(b) is true, or 4(c) is true, then the decoding process can be terminated; otherwise, the next ADMM iteration is performed.

$\text{Thre}$, the standard $\epsilon$ rule, and the $\text{Hx}^T = 0$ ET rule. From Figure 3(a), we can see that there is little difference in the FER performance among the three termination methods at lower SNRs, while at higher SNRs, the FER performance of the designed ET method with large values of $\text{Thre}$ is better than that with small values of $\text{Thre}$. However, all of these results are worse than those for the standard $\epsilon$ rule and the $\text{Hx}^T = 0$ ET rule. On the contrary, Figure 3(b) shows that at lower SNRs, the ANIs of the designed ET method with different $\text{Thre}$ values are all less than those of the other two termination methods. In addition, the diminution in ANIs is more obvious for small $\text{Thre}$ values than that for large values, but the ANIs for different $\text{Thre}$ are nearly the same as those of the $\text{Hx}^T = 0$ ET rule at higher SNRs. Furthermore, all of the ANI results are noticeably lower than those of the standard $\epsilon$ rule. Therefore, an elastic trade-off between $\text{Thre}$ and the FER performance is needed.

Second, the influence of $\text{Iter}$ is studied when $\text{Thre} = 7$. The FER performance and the ANIs of the three termination methods are plotted in Figure 4. From Figure 4(a), it can be seen that the FER performance of the three termination methods is nearly the same at lower SNRs; however, the FER performance of the other two termination methods is better than that of the designed ET methods with different $\text{Iter}$ values at higher SNRs. Additionally, the larger the $\text{Iter}$ value is, the better the FER performance is for the designed ET method. Figure 4(b) illustrates that small $\text{Iter}$ values cause obvious diminution of the ANIs for the designed ET method, and all ANIs for the designed ET method are lower than those for the other two termination methods at lower SNRs. Furthermore, the ANIs for the designed ET method are nearly identical to those for the $\text{Hx}^T = 0$ ET rule at higher SNRs, and they are also all lower than those for the standard $\epsilon$ rule. Therefore, we also need to make a trade-off between $\text{Iter}$ and the FER performance.

Similarly, we set $\text{Iter} = 12$ and study the FER performance and ANIs of $C_2$ when $\text{Thre}$ is 11, 12, 13, 14, 15, 16,
Figure 3: Comparison of the FER and ANIs for $C_1$ when using the flooding scheduling ADMM-penalized decoder with different $Thre$ values and $Iter = 10$.

Figure 4: Comparison of the FER and ANIs for $C_1$ when using the flooding scheduling ADMM-penalized decoder with different $Iter$ values and $Thre = 7$. 
and 17, respectively (as shown in Figure 5). The same conclusions that are drawn about $C_1$ are similarly obtained for $C_2$; that is, the higher the $\text{Thre}$ value, the better the FER performance, but the ANIs at lower SNRs are higher. Then, $\text{Thre}$ is set to 15. Figure 6 shows the FER and ANIs when $\text{Iter}$ is 9, 10, 11, 12, 13, 14, and 15, respectively. Similarly, to obtain a better FER performance, more $\text{Iter}$ is needed, and inevitably, the ANIs at lower SNRs are also higher.

4.1.2. For the Layered Scheduling ADMM-Penalized Decoder.
The FER performance and ANIs for $C_1$ with different $\text{Thre}$ values and $\text{Iter}=9$, as well as with different $\text{Iter}$ values and $\text{Thre}=7$, are shown in Figures 7 and 8, respectively. Similarly, Figures 9 and 10 illustrate the FER performance and ANIs for $C_2$ with different $\text{Thre}$ values and $\text{Iter}=11$, as well as with different $\text{Iter}$ values and $\text{Thre}=15$, respectively. It can be seen that the results are similar to the conclusions drawn regarding scheduling; that is, the larger $\text{Thre}$ value (or $\text{Iter}$ value), the better the FER performance, but it also incurs more ANIs at lower SNRs.

4.2. Comparison of FER Performance and ANIs. For $C_1$, according to the trade-off between the FER performance and ANIs, we choose $\text{(Thre, Iter)} = (7, 11)$ for the designed ET method according to the simulation results from the flooding scheduling ADMM-penalized decoder. The FER performance and ANIs of the flooding scheduling ADMM-penalized decoder with the designed ET method, the standard $\epsilon$ rule, and the $\text{Hx}^T = 0$ ET rule are shown in Figure 11. Compared with the other two termination methods, the designed ET method obviously reduces the ANIs at low SNRs with negligible FER performance loss. When $\text{SNR} = 1.6 \text{ dB}$, the designed ET method can reduce the number of iterations by approximately 12 and 16 compared with the standard $\epsilon$ rule and the $\text{Hx}^T = 0$ ET rule, respectively.

For $C_2$, we choose $\text{(Thre, Iter)} = (15, 13)$ for the designed ET method with the flooding scheduling ADMM-penalized decoder. Figure 11 also shows the FER performance and ANIs for $C_2$ with the three termination methods, and we can observe similar results. Compared with the standard $\epsilon$ rule and the $\text{Hx}^T = 0$ ET rule, the designed ET method can reduce the number of iterations by approximately 6.5 and 10 when $\text{SNR} = 2.8 \text{ dB}$.
Figure 6: Comparison of the FER and ANIs for $C_2$ when using the flooding scheduling ADMM-penalized decoder with different $Iter$ values and $Thre = 15$.

Figure 7: Comparison of the FER and ANIs for $C_1$ when using the layered scheduling ADMM-penalized decoder with different $Thre$ values and $Iter = 9$. 

Security and Communication Networks
**Figure 8:** Comparison of the FER and ANIs for $C_1$ when using the layered scheduling ADMM-penalized decoder with different Iter values and $Thre = 7$.

**Figure 9:** Comparison of the FER and ANIs for $C_2$ when using the layered scheduling ADMM-penalized decoder with different Thre values and Iter = 11.
When using the layered scheduling ADMM-penalized decoder with the designed ET method, we choose the optimized parameter combinations \((\text{Thre}, \text{Iter}) = (7, 10)\) and \((\text{Thre}, \text{Iter}) = (15, 12)\) for \(C_1\) and \(C_2\), respectively. Figure 12 shows the FER performance and ANIs of the three termination methods for \(C_1\) and \(C_2\). Similarly, at low SNRs, the
designed ET method can clearly reduce the ANIs with little FER performance loss. According to the designed ET method, the number of iterations is reduced by approximately 3 and 5 at 1.6 dB for $C_1$ and 3 and 3.5 at 2.8 dB for $C_2$, respectively.

5. Conclusion

By computing the ASHD at each iteration, the ET method presented in this paper can increase the convergence rate of the ADMM-penalized decoder for LDPC codes. Compared with the two existing termination methods, the designed ET method clearly reduces the average number of iterations at low SNRs with negligible decoding performance loss. In addition, both the flooding scheduling and layered scheduling ADMM-penalized decoders can adopt the designed ET method.

Data Availability

The data used to support the findings of this study are included within the supplementary information file.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Natural Science Basic Research Plan in Shaanxi Province of China (No. 2021JM-515).

Supplementary Materials

The supplementary information file contains the data that are used to support the findings of this study. Figure 1: the changing ASHD trends for $C_1$ and $C_2$ when using the flooding scheduling ADMM-penalized decoder. Figure 2: the changing ASHD trends for $C_1$ and $C_2$ when using the layered scheduling ADMM-penalized decoder. Figure 3: the comparison of FER and ANIs for $C_1$ when using the flooding scheduling ADMM-penalized decoder with different $Thre$ values and $Iter = 10$. Figure 4: the comparison of FER and ANIs for $C_1$ when using the flooding scheduling ADMM-penalized decoder with different $Iter$ values and $Thre = 7$. Figure 5: the comparison of FER and ANIs for $C_2$ when using the flooding scheduling ADMM-penalized decoder with different $Thre$ values and $Iter = 12$. Figure 6: the comparison of FER and ANIs for $C_2$ when using the flooding scheduling ADMM-penalized decoder with different $Iter$ values and $Thre = 15$. Figure 7: the comparison of FER and ANIs for $C_1$ when using the layered scheduling ADMM-penalized decoder with different $Thre$ values and $Iter = 9$. Figure 8: the comparison of FER and ANIs for $C_1$ when using the layered scheduling ADMM-penalized decoder with different $Iter$ values and $Thre = 7$. Figure 9: the comparison of FER and ANIs for $C_2$ when using the layered scheduling ADMM-penalized decoder with different $Thre$ values and $Iter = 11$. Figure 10: the comparison of FER and ANIs for $C_2$ when using the layered scheduling ADMM-penalized decoder with different $Iter$ values and $Thre = 15$. Figure 11: the comparison of FER and ANIs for $C_1$ and $C_2$ when using the flooding scheduling ADMM-penalized decoder with the three termination methods. Figure 12: the comparison of FER and ANIs for $C_1$ and $C_2$ when using the layered scheduling ADMM-penalized decoder with the three termination methods.
scheduling ADMM-penalized decoder with the three termination methods. (Supplementary Materials)

References


Security and Communication Networks
Communication and Signal Processing (WCSP), Hangzhou, China, October 2018.


