Hindawi
Security and Communication Networks
Volume 2022, Article ID 4599105, 13 pages
https://doi.org/10.1155/2022/4599105

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

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 l_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 ε 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. Preliminaries

2.1. LP Decoding. An $m \times n$ parity-check matrix $\mathbf{H}$ of a binary LDPC code C indicates that its code length is n and the number of check equations is m. Meanwhile, n and m are also the number of variable nodes which are represented by $I = \{1, 2, \ldots, n\}$, and the number of check nodes which are represented by $J = \{1, 2, \ldots, m\}$, respectively. The element $\mathbf{H}_{ji} = 1$ in $\mathbf{H}$ means that the check node c_j and the variable node v_i are related. The set of all check nodes related to variable node v_i is denoted by $N_v(i)$, and the degree of v_i is denoted by $d_{v_i} = |N_v(i)|$. The set of all variable nodes related to the check node c_j is denoted by $N_c(j)$, and the degree of c_j is denoted by $d_{c_j} = |N_c(j)|$.

Suppose that the sender transmits a codeword vector denoted by $\mathbf{x} = \{x_i \in \{0, 1\} | i \in I\}$ and that the receiving end is provided a vector denoted by $\mathbf{r} = \{r_i | i \in I\}$ over a noisy channel. The vector of the log-likelihood ratios (LLRs) is denoted by $\gamma = \{\gamma_i | i \in I\}$, where $\gamma_i = \log\Pr(r_i|0)/\Pr\ (r_i|1)$.

Definition 1. The LP decoding problem is defined as follows [13]:

$$\begin{cases} \min, & \boldsymbol{\gamma}^T \mathbf{x}, \\ \text{s.t}, & \mathrm{T}_j \mathbf{x} \in \mathrm{P}_{d_{c_j}}, \forall j \in J, \end{cases} \tag{1}$$

where the check polytope $\mathrm{P}_{d_{c_j}}$ is the convex hull of all even-parity binary vectors of length d_{c_j}, and the d_{c_j} components of $\mathbf{x}$ that participate in the j-th check node are elected out by the $d_{c_j} \times n$ binary transfer matrix T_j.

2.2. ADMM Decoding. By introducing $\mathbf{z}_j \in R^{d_{c_j}} (j \in J)$, which represents the "replica" variables, problem (1) is rewritten as the following ADMM-based LP problem [14]:

$$\begin{cases} \min, & \boldsymbol{\gamma}^T \mathbf{x}, \\ \text{s.t}, & \mathrm{T}_j \mathbf{x} = \mathbf{z}_j, \mathbf{z}_j \in \mathrm{P}_{d_{c_j}}, \forall j \in J. \end{cases} \tag{2}$$

The augmented Lagrangian of problem (2) is as follows:

$$L_\mu(x, z, y) = \gamma^T x + \frac{\mu}{2} \sum_{j \in J} \left\| \mathrm{T}_j x - z_j + y_j \right\|_2^2 - \frac{\mu}{2} \sum_{j \in J} \left\| y_j \right\|_2^2, \tag{3}$$

where $\mathbf{y}_j \in R^{d_{c_j}}$ denotes the scaled dual variables corresponding to each c_j, the penalty parameter μ is a constant, and $\mu > 0$.

Then, the ADMM iterations are as follows [14]:

$$\begin{aligned} \mathbf{x}^{k+1} &:= \operatorname{argmin}_{\mathbf{x} \in \mathbf{X}} \left(\boldsymbol{\gamma}^{\mathrm{T}} \mathbf{x} + \frac{\mu}{2} \sum_{j \in J} \left\| \mathrm{T}_j \mathbf{x} - \mathbf{z}_j^k + \mathbf{y}_j^k \right\|_2^2 \right), \\ \mathbf{z}_j^{k+1} &:= \prod_{\mathrm{P}_{d_{c_j}}} \left(\mathrm{T}_j \mathbf{x}^{k+1} + \mathbf{y}_j^k \right), \\ \mathbf{y}_j^{k+1} &:= \mathbf{y}_j^k + \mathrm{T}_j \mathbf{x}^{k+1} - \mathbf{z}_j^{k+1}, \end{aligned} \tag{4}$$

where $\mathbf{X} = [0, 1]^n$, and $\prod_{\mathrm{P}_{d_{c_j}}} (u)$ represents the Euclidean projection of the vector u onto $p_{d_{c_j}}$.

2.3. ADMM Penalized Decoding. After adding a penalty function, problem (2) becomes the ADMM penalized decoding:

$$\begin{cases} \min & \boldsymbol{\gamma}^T \mathbf{x} + \alpha \sum_{i\in I} g(x_i), \\ \text{s.t} & \mathrm{T}_j \mathbf{x} = \mathbf{z}_j, \mathbf{z}_j \in P_{d_{c_j}}, \quad \forall j \in J, \end{cases} \tag{5}$$

where α 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 ε rule and the $\mathbf{Hx}^T = 0$ ET rule. The first method compares the prime residual $\sum_{j\in J}\|\mathrm{T}_j\mathbf{x}^k - \mathbf{z}_j^k\|_2$ and the dual residual $\sum_{j\in J}\|\mathrm{T}_j\mathbf{x}^k - \mathbf{z}_j^k\|_2$ with the error tolerance ε ($\varepsilon > 0$), and the ADMM decoding process terminates when their values are all less than ε [14], namely, when $\sum_{j\in J}\|\mathrm{T}_j\mathbf{x}^k - \mathbf{z}_j^k\|_2 < \varepsilon$ and $\sum_{j\in J}\|\mathbf{z}_j^k - \mathbf{z}_j^{k-1}\|_2 < \varepsilon$ are all true. The second term verifies the parity-check equation $\mathbf{Hx}^T = 0$ 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:

$$H_i^k = \begin{cases} 0, \text{if } x_i^k < 0.5, \\ 1, \text{if } x_i^k \geq 0.5, \end{cases} \tag{6}$$

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 $\mathbf{x}$ 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 *Iter* and *Thre* that should be considered when employing the designed ET method. *Iter* denotes the number of iterations at which the ADMM decoder starts the ASHD check, and *Thre* is the ASHD threshold used to check whether the decoding process should be terminated. It should be noted that the value of *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, *Iter* should not be too small. However, *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 $iter_{max}$. $\mathrm{Sum}[k] = 0$ 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 ε rule, and the $\mathbf{Hx}^T = 0$ 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 Iter and Thre. As *Iter* and *Thre* are the two critical parameters in the designed ET method, it is

```
Input: Received vector $\mathbf{r} = \{r_i | i \in I\}$.
Output: Decoded vector $\widehat{x} = \{\widehat{c}_i | i \in I\}$.
(1)   $\forall j \in J,\ i \in N_c(j)$: $\mathbf{z}_j^0 = 0.5$, $\mathbf{y}_j^0 = 0$, $\mathbf{L}_{j\longrightarrow i}^0 = 0.5$
(2)   for all $k = 1 \longrightarrow iter_{max}$ when stop criterion = false do
(3)       for all $i \in I,\ j \in N_v(i)$ do
(4)           $x_i = \prod_{[0,1]}[1/d_{v_i}(\sum_{j\in \mathbf{N}_v(i)}(\mathbf{z}_j^{(i)} - 1/\mu\mathbf{y}_j^{(i)}) - 1/\mu(\nabla_x f)_i)]$
(5)       end for
(6)       for all $j \in \mathbf{J},\ i \in N_c(j)$ do
(7)           $\mathbf{w}_j = \mathbf{T}_j\mathbf{x} + \mathbf{y}_j/\mu$
(8)           $\mathbf{z}_j = \prod_{\mathrm{P}_{d_{c_j}}}(\mathbf{w}_j)$
(9)           $\mathbf{y}_j = \mathbf{y}_j + \mu(\mathbf{T}_j\mathbf{x} - \mathbf{z}_j)$
(10)      end for
(11)  end for
```

Algorithm 1: ADMM-penalized decoding algorithm.

```
Input: $\mathbf{x} = \{x_i | i \in I\}$ from the $k$-th ADMM iteration.
Output: ASHD$[k]$ of all the $k$-th ADMM iteration.
(1)   For each component $x_i$, compute its hard decision, $H_i^k$, at the end of the $k$-th iteration.
(2)   Compute the sum of hard decision, sum = $\sum_{i=1}^{n} H_i^k$, at the end of the $k$-th iteration.
(3)   Compute all the $k$-th iteration' sum of hard decision, Sum$[k]$ = Sum$[k]$ + sum.
(4)   Count the number of all $k$-th iteration, counter$[k]$ + +.
(5)   Compute all the $k$-th iteration' average sum of hard decision, ASHD$[k]$ = sum$[k]$/counter$[k]$.
```

Algorithm 2: Compute the ASHD of the ADMM-penalized decoding method at each iteration.

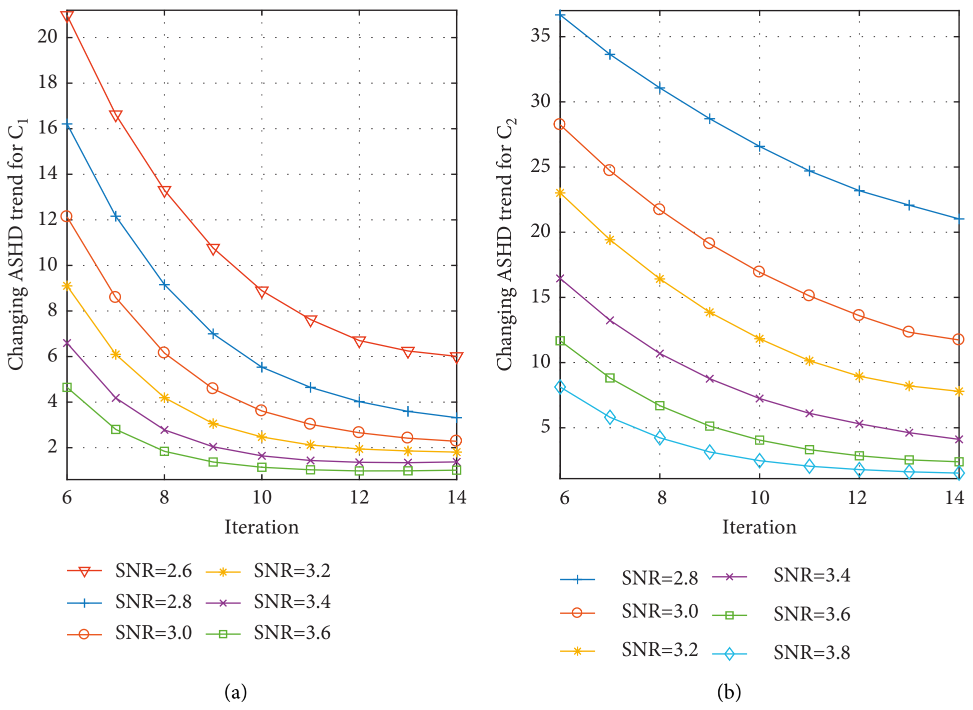

Figure 1: Changing the ASHD trends for C_1 and C_2 when using the flooding scheduling ADMM-penalized decoder.

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 C_1. Figure 3 shows the FER performance and the ANIs of the designed ET method with different values of

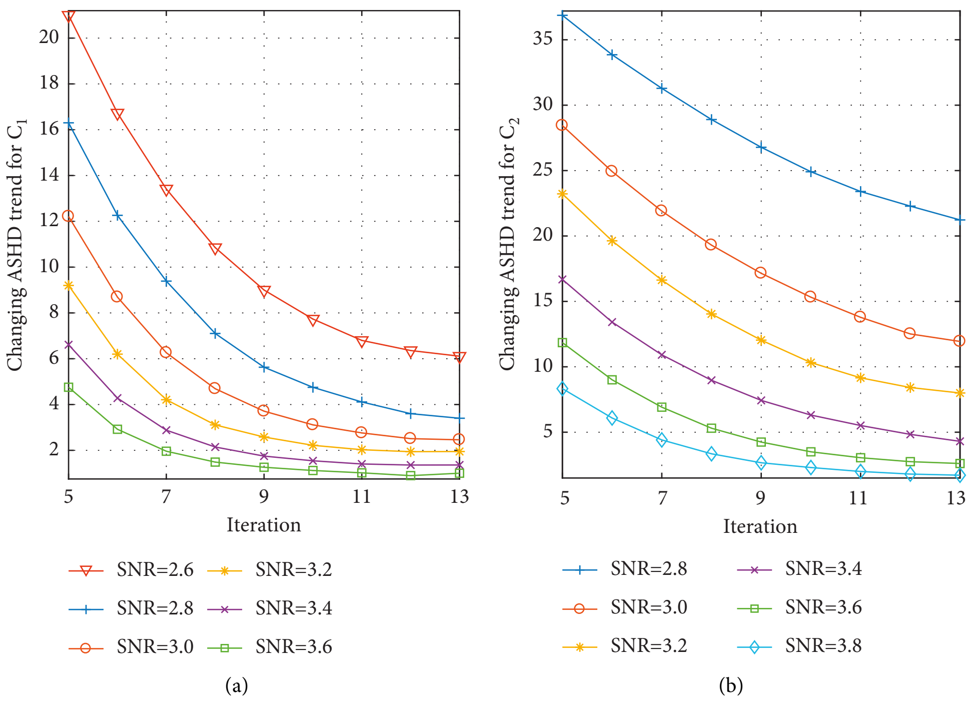

FIGURE 2: Changing the ASHD trends for C_1 and C_2 when using the layered scheduling ADMM-penalized decoder.

```
Input: $\mathbf{x} = \{x_i | i \in I\}$ from the $k$-th ADMM iteration, the parameter values of $Iter$ and $Thre$.
(1)  For each component $x_i$, compute its hard decision $H_i^k$ at the end of the $k$-th iteration.
(2)  Compute the sum of hard decision, sum $= \sum_{i=1}^{n} H_i^k$, at the end of the $k$-th iteration.
(3)  Compute the $k$-th iteration' average sum of hard decision, $\mathrm{AS}[k] = \mathrm{Sum}[k]/k$.
(4)  (a) Check whether $k > iter_{max}$;
     (b) Check whether $\mathrm{Sum}[k] = 0$;
     (c) Check whether $k > Iter$ and $\mathrm{AS}[k] > Thre$.
(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.
```

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

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

Second, the influence of *Iter* is studied when $Thre = 7$. 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 *Iter* values at higher SNRs. Additionally, the larger the *Iter* value is, the better the FER performance is for the designed ET method. Figure 4(b) illustrates that small *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 $\mathbf{Hx}^T = 0$ ET rule at higher SNRs, and they are also all lower than those for the standard ε rule. Therefore, we also need to make a trade-off between *Iter* and the FER performance.

Similarly, we set $Iter = 12$ and study the FER performance and ANIs of C_2 when *Thre* is 11, 12, 13, 14, 15, 16,

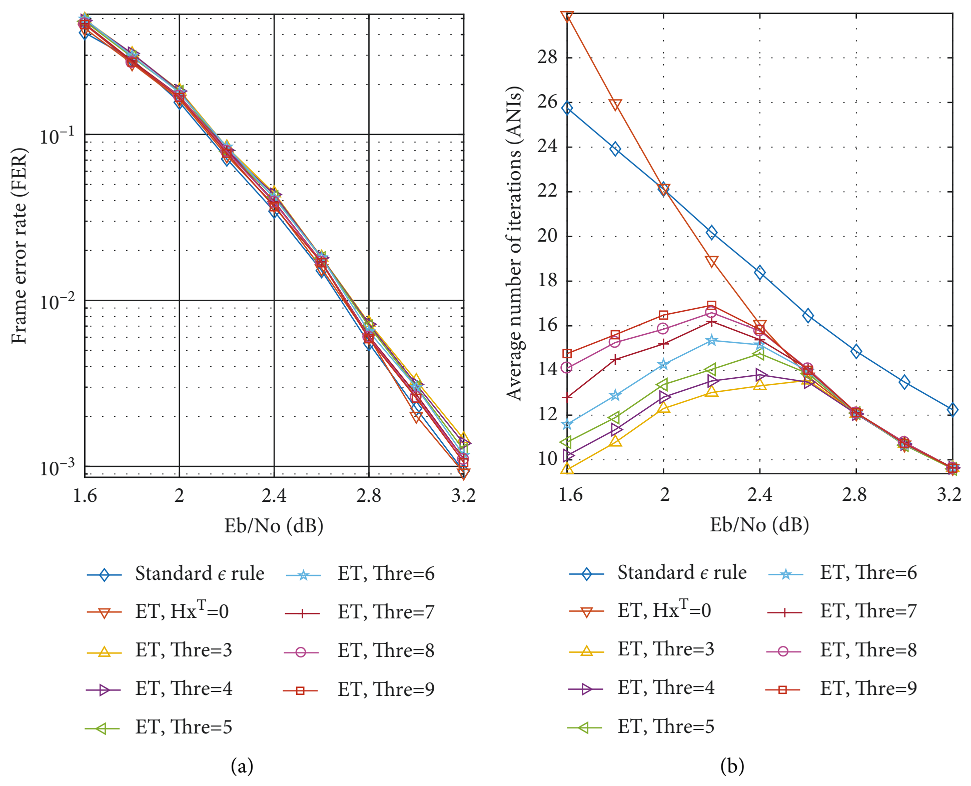

Figure 3: Comparison of the FER and ANIs for C_1 when using the flooding scheduling ADMM-penalized decoder with different *Thre* values and *Iter* = 10.

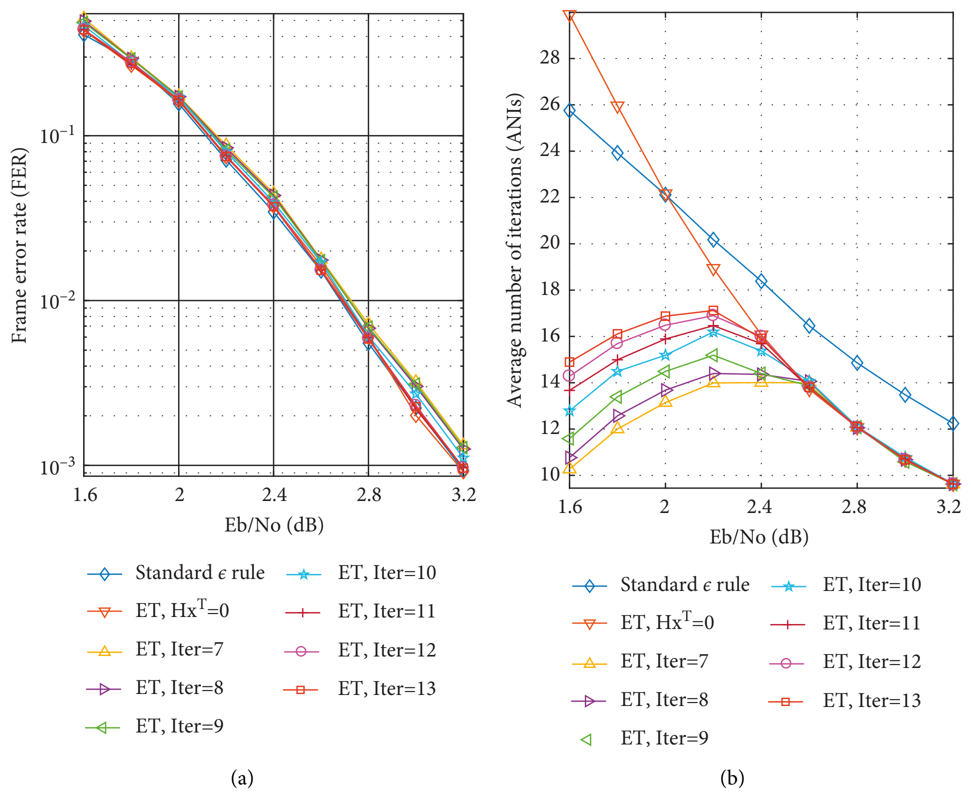

Figure 4: Comparison of the FER and ANIs for C_1 when using the flooding scheduling ADMM-penalized decoder with different *Iter* values and *Thre* = 7.

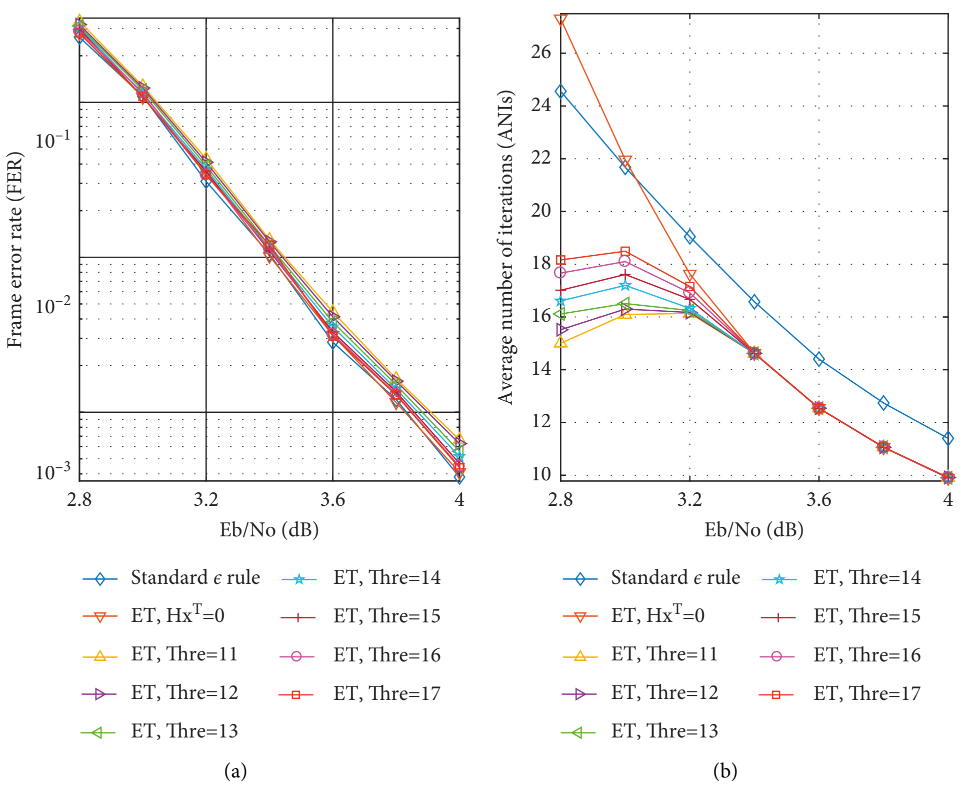

FIGURE 5: Comparison of the FER and ANIs for C_2 when using the flooding scheduling ADMM-penalized decoder with different *Thre* values and *Iter* = 12.

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 *Thre* value, the better the FER performance, but the ANIs at lower SNRs are higher. Then, *Thre* is set to 15. Figure 6 shows the FER and ANIs when *Iter* is 9, 10, 11, 12, 13, 14, and 15, respectively. Similarly, to obtain a better FER performance, more *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 *Thre* values and *Iter* = 9, as well as with different *Iter* values and *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 *Thre* values and *Iter* = 11, as well as with different *Iter* values and *Thre* = 15, respectively. It can be seen that the results are similar to the conclusions drawn regarding scheduling; that is, the larger *Thre* value (or *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 (*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 ε rule, and the $\mathbf{Hx}^T = 0$ 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 SNR = 1.6 dB, the designed ET method can reduce the number of iterations by approximately 12 and 16 compared with the standard ε rule and the $\mathbf{Hx}^T = 0$ ET rule, respectively.

For C_2, we choose (*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 ε rule and the $\mathbf{Hx}^T = 0$ ET rule, the designed ET method can reduce the number of iterations by approximately 6.5 and 10 when SNR = 2.8 dB.

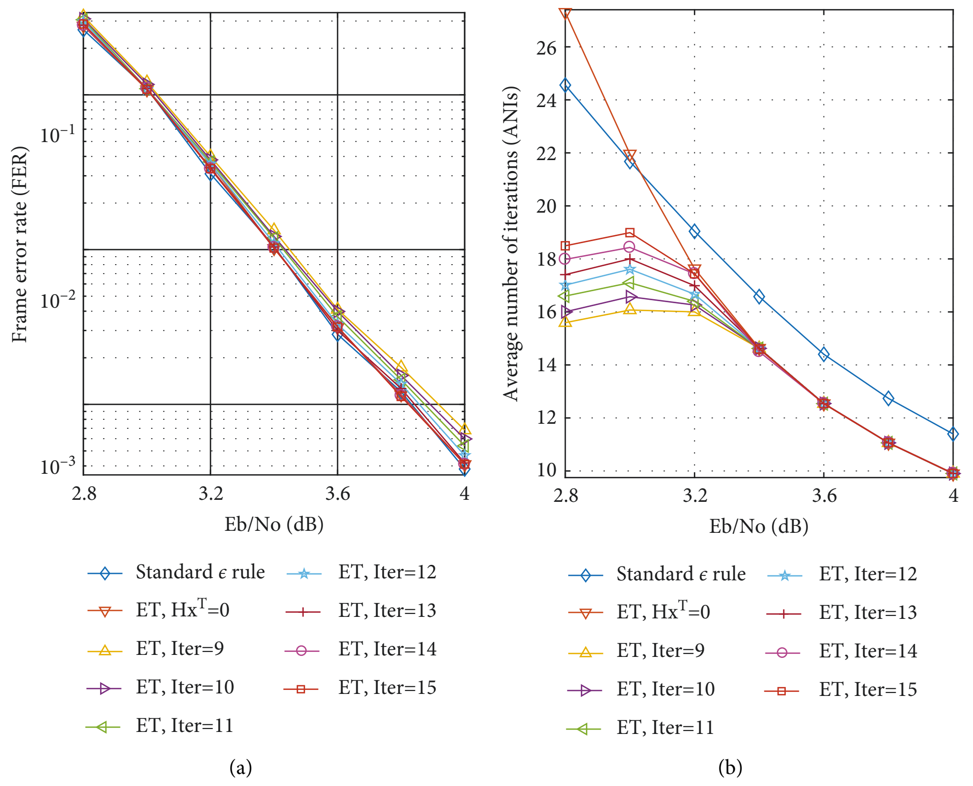

FIGURE 6: Comparison of the FER and ANIs for C_2 when using the flooding scheduling ADMM-penalized decoder with different *Iter* values and *Thre* = 15.

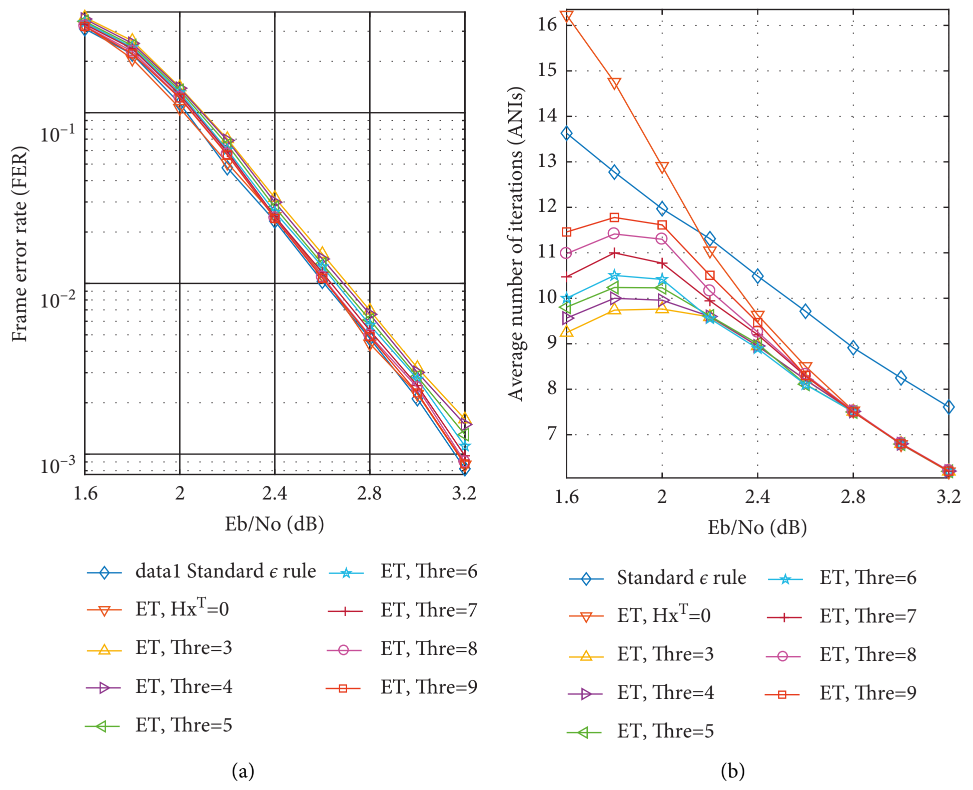

FIGURE 7: Comparison of the FER and ANIs for C_1 when using the layered scheduling ADMM-penalized decoder with different *Thre* values and *Iter* = 9.

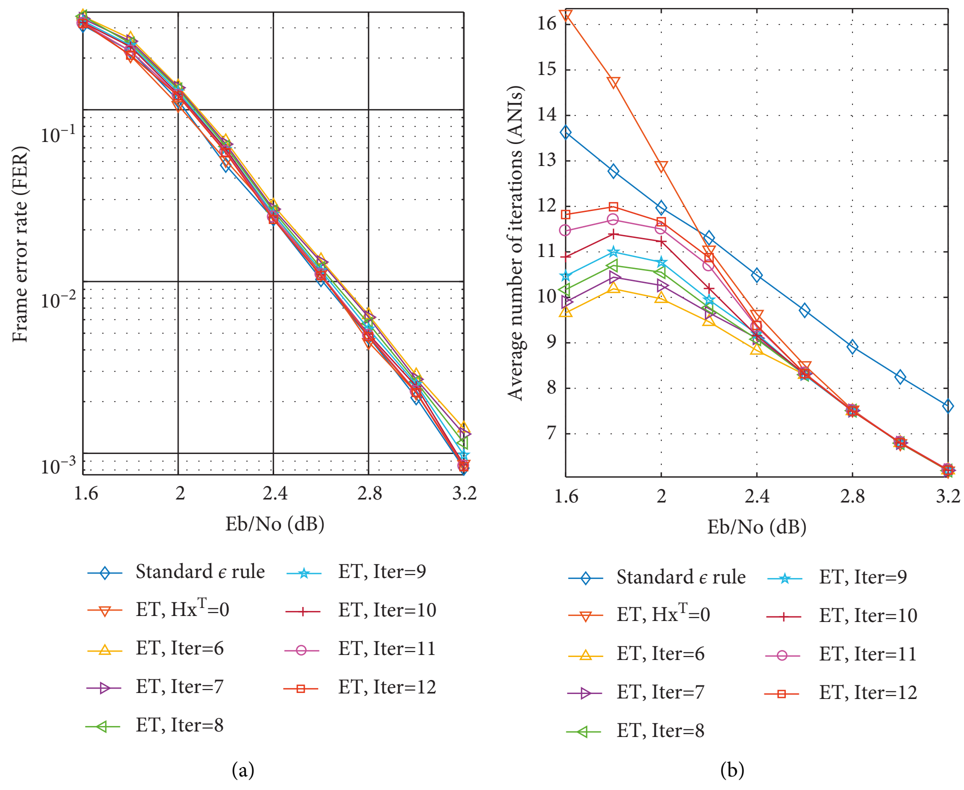

FIGURE 8: Comparison of the FER and ANIs for C_1 when using the layered scheduling ADMM-penalized decoder with different *Iter* values and *Thre* = 7.

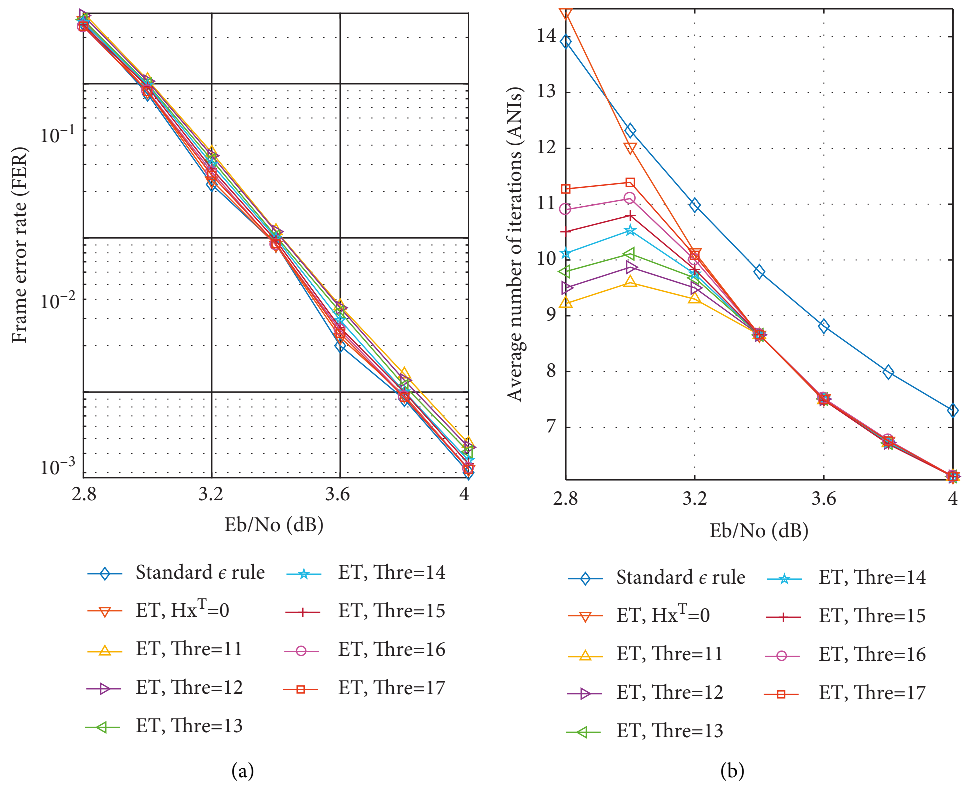

FIGURE 9: Comparison of the FER and ANIs for C_2 when using the layered scheduling ADMM-penalized decoder with different *Thre* values and *Iter* = 11.

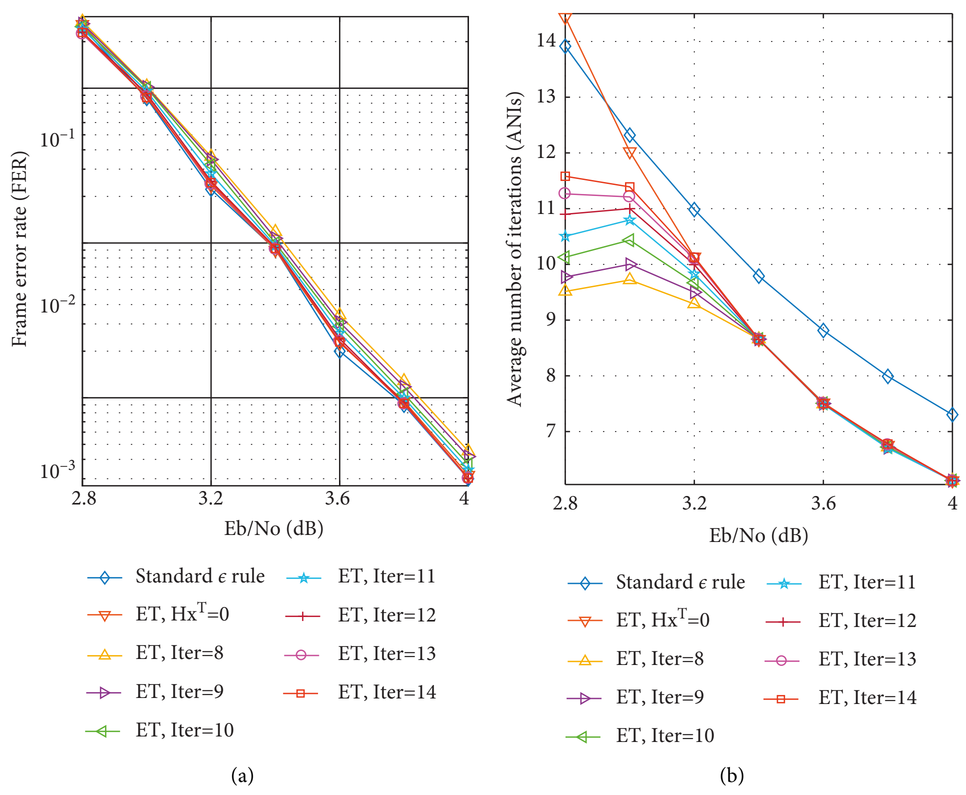

Figure 10: Comparison of the FER and ANIs for C_2 when using the layered scheduling ADMM-penalized decoder with different *Iter* values and *Thre* = 15.

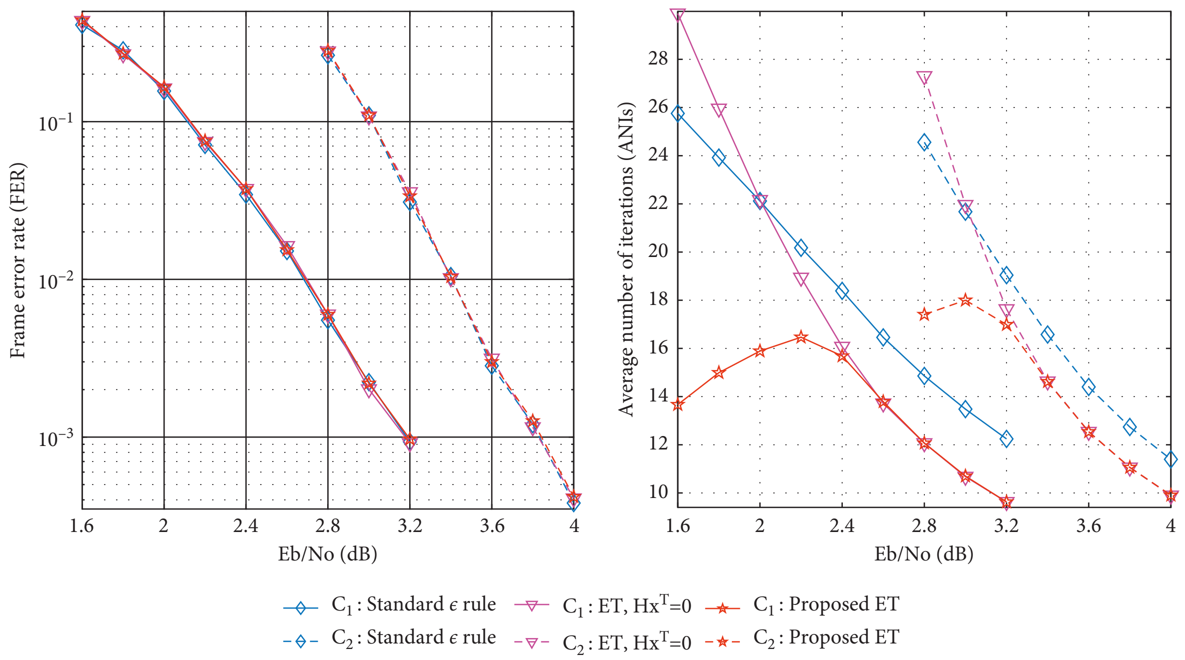

Figure 11: Comparison of the FER and ANIs for C_1 and C_2 when using the flooding scheduling ADMM-penalized decoder with the three termination methods.

When using the layered scheduling ADMM-penalized decoder with the designed ET method, we choose the optimized parameter combinations (*Thre*, *Iter*) = (7, 10) and (*Thre*, *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

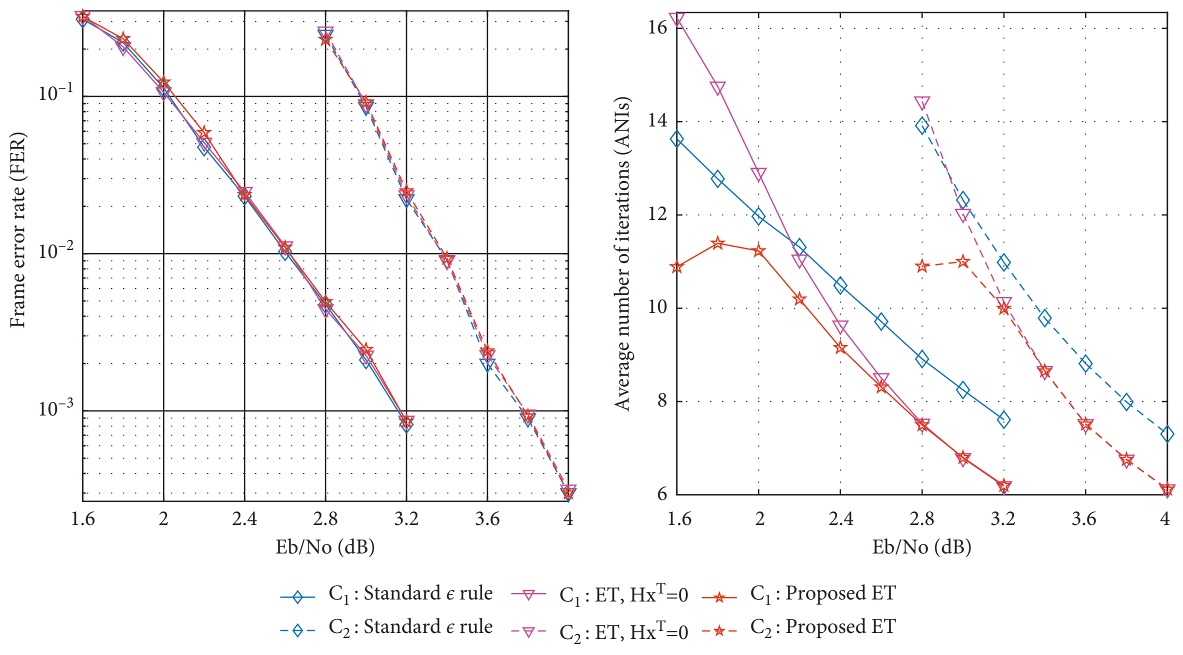

Figure 12: Comparison of the FER and ANIs for C_1 and C_2 when using the layered scheduling ADMM-penalized decoder with the three termination methods.

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. (*Supplementary Materials*)

References

[1] M. Zheng, S. Chen, W. Liang, and M. Song, "NSAC: a novel clustering protocol in cognitive radio sensor networks for internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5864-5865, 2019.

[2] M. P. K. Reddy and M. R. Babu, "Implementing self adaptiveness in whale optimization for cluster head section in internet of things," *Cluster Computing*, vol. 22, no. 1, pp. 1361–1372, 2019.

[3] Y. Liu, K. Wang, K. Qian, M. Du, and S. Guo, "Tornado: enabling blockchain in heterogeneous internet of things through a space-structured approach," *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 1273–1286, 2020.

[4] H. Gao, L. Yu, I. A. Khan, Y. Wang, Y. Yang, and H. Shen, "Visual object detection and tracking for internet of things devices based on spatial attention powered multi-domain network," *IEEE Internet of Things Journal*, p. 1, 2021.

[5] D. G. Gopal, X. Z. Gao, and J. A. A. Alzubi, "IoT and big data impact on various engineering applications," *Recent Patents on Engineering*, vol. 15, no. 2, pp. 119-120, 2021.

[6] M. Abirami, M. A. Jerlin, and D. G. Gopal, "A smart parking system using IoT," *World Review of Entrepreneurship Management and Sustainable Development*, vol. 15, no. 3, pp. 335–345, 2019.

[7] S. M. Nagarajan, G. G. Deverajan, P. Chatterjee, W. Alnumay, and V. Muthukumaran, "Integration of IoT based routing process for food supply chain management in sustainable smart cities," *Sustainable Cities and Society*, vol. 76, pp. 103448–103510, 2022.

[8] H. Y. Kwak, J. S. No, and H. Park, "Design of irregular SC-LDPC codes with non-uniform degree distributions by linear programming," *IEEE Transactions on Communications*, vol. 67, no. 4, pp. 2632–2646, 2019.

[9] M. R. Sadeghi, "Optimal search for girth-8 quasi cyclic and spatially coupled multiple-edge LDPC codes," *IEEE Communications Letters*, vol. 23, no. 9, pp. 1466–1469, 2019.

[10] J. W. Hu, M. Baldi, P. Santini, N. Zeng, S. Ling, and H. X. Wang, "Lightweight key encapsulation using LDPC codes on FPGAs," *IEEE Transactions on Computers*, vol. 69, no. 3, pp. 327–341, 2020.

[11] Z. Yang, Y. Fang, Y. Cheng, P. Chen, and D. J. Almakhles, "Protograph LDPC-coded BICM-ID with irregular mapping: an emerging transmission technique for massive internet of things," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 3, pp. 1051–1065, 2021.

[12] R. Saravanan and D. G. Gopal, "Selfish node detection based on evidence by trust authority and selfish replica allocation in DANET," *International Journal of Information and Communication Technology*, vol. 9, no. 4, pp. 473–491, 2016.

[13] J. Feldman, M. J. Wainwright, and D. R. Karger, "Using linear programming to decode binary linear codes," *IEEE Transactions on Information Theory*, vol. 51, no. 3, pp. 954–972, 2005.

[14] S. Barman, X. Liu, S. C. Draper, and B. Recht, "Decomposition methods for large scale LP decoding," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 7870–7886, 2013.

[15] X. Liu and S. C. Draper, "The ADMM penalized decoder for LDPC codes," *IEEE Transactions on Information Theory*, vol. 62, no. 6, pp. 2966–2984, 2016.

[16] X. Jiao, H. Wei, J. Mu, and C. Chen, "Improved ADMM penalized decoder for irregular low-density parity-check codes," *IEEE Communications Letters*, vol. 19, no. 6, pp. 913–916, 2015.

[17] B. Wang and Z. Wang, "ADMM penalized decoding method based on improved penalty function for LDPC codes in the IoTs," *Computer Communications*, vol. 154, pp. 197–203, 2020.

[18] H. Wei, X. Jiao, and J. Mu, "Reduced-complexity linear programming decoding based on ADMM for LDPC codes," *IEEE Communications Letters*, vol. 19, no. 6, pp. 909–912, 2015.

[19] X. Zhang and P. H. Siegel, "Efficient iterative LP decoding of LDPC codes with alternating direction method of multipliers," in *Proceedings of the IEEE International Symposium on Information Theory*, Istanbul, Turkey, July 2013.

[20] G. Zhang, R. Heusdens, and W. B. Kleijn, "Large scale LP decoding with low complexity," *IEEE Communications Letters*, vol. 17, no. 11, pp. 2152–2155, 2013.

[21] X. Jiao, J. Mu, Y. C. He, and C. Chen, "Efficient ADMM decoding of LDPC codes using lookup tables," *IEEE Transactions on Communications*, vol. 65, no. 4, pp. 1425–1437, 2017.

[22] H. Wei and A. H. Banihashemi, "An iterative check polytope projection algorithm for ADMM-based LP decoding of LDPC codes," *IEEE Communications Letters*, vol. 22, no. 1, pp. 29–32, 2018.

[23] Q. Xia, Y. Lin, S. Tang, and Q. Zhang, "A fast approximate check polytope projection algorithm for ADMM decoding of LDPC codes," *IEEE Communications Letters*, vol. 23, no. 9, pp. 1520–1523, 2019.

[24] J. Bai, Y. Wang, and F. C. M. Lau, "Minimum-polytope-based linear programming decoder for LDPC codes via ADMM approach," *IEEE Wireless Communications Letters*, vol. 8, no. 4, pp. 1032–1035, 2019.

[25] Q. Xia, X. Wang, H. Liu, and Q. L. Zhang, "A hybrid check polytope projection algorithm for ADMM decoding of LDPC codes," *IEEE Communications Letters*, vol. 25, no. 1, pp. 108–112, 2021.

[26] Y. Wei, M. M. Zhao, M. J. Zhao, and M. Lei, "ADMM-based decoder for binary linear codes aided by deep learning," *IEEE Communications Letters*, vol. 24, no. 5, pp. 1028–1032, 2020.

[27] J. Bai, Y. Wang, and Q. Shi, "Efficient QP-ADMM decoder for binary LDPC codes and its performance analysis," *IEEE Transactions on Signal Processing*, vol. 68, pp. 503–518, 2020.

[28] X. Jiao, H. Liu, J. Mu, and Y. He, "l_2-Box ADMM decoding for LDPC codes over ISI channels," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 4, pp. 3966–3971, 2021.

[29] I. Debbabi, B. L. Gal, N. Khouja, F. Tlili, and C. Jego, "Fast converging ADMM-penalized algorithm for LDPC decoding," *IEEE Communications Letters*, vol. 20, no. 4, pp. 648–651, 2016.

[30] I. Debbabi, B. L. Gal, N. Khouja, F. Tlili, and C. Jego, "Comparison of diffferent schedulings for the ADMM based LDPC decoding," in *Proceedings of the International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*, Brest, France, October 2016.

[31] X. Jiao, J. Mu, and H. Wei, "Reduced complexity node-wise scheduling of ADMM decoding for LDPC codes," *IEEE Communications Letters*, vol. 21, no. 3, pp. 472–475, 2017.

[32] H. Wu, F. Wang, and Y. Yuan, "A distributed CRC early termination scheme for high throughput QC-LDPC codes," in *Proceedings of the International Conference on Wireless*

Communications and Signal Processing (WCSP), Hangzhou, China, October 2018.

[33] J. Frenzel, S. Mller-Weinfurtner, J. B. Huber, and R. R. Mller, “Convergence behavior of LDPC decoding and application to early termination,” in *Proceedings of the IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Bologna, Italy, September 2018.

[34] D. Declercq, V. Savin, O. Boncalo, and F. Ghaffari, “An imprecise stopping criterion based on in-between layers partial syndromes,” *IEEE Communications Letters*, vol. 22, no. 1, pp. 13–16, 2018.

[35] C. Lin, Y. Wu, and C. Song, “Energy-efficient LDPC codec design using cost-effective early termination scheme,” *IET Computers & Digital Techniques*, vol. 13, no. 2, pp. 118–125, 2019.

[36] E. Cho, H. Lee, H. Ju, C. Yoon, and S. H. Kim, “Early termination method for LDPC codes based on majority-voted codeword,” in *Proceedings of the International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju, Korea (South), December 2020.