SCIENCE CHINA
Information Sciences

• **Supplementary File** •

Reduced-Search Guessing Random Additive Noise Decoding of Polar Codes

Kefan Wang[1], Yuejun Wei[2], Zhenyuan Chen[2], Huarui Yin[3], Liping Li[1*] & Wenyi Zhang[3*]

[1] *Key Laboratory of Intelligent Computing and Signal Processing of the Ministry of Education of China, Anhui University, Hefei 230601, China;*
[2] *Huawei Technologies;*
[3] *Department of Electronic Engineering and Information Science, University of Science and Technology of China*

Appendix A Proof of Theorem 1

Let the received samples be $\boldsymbol{y}$, and $\tilde{\boldsymbol{y}}$ is the corresponding hard decisions.

Theorem 1. To estimate $\widehat{\boldsymbol{u}} = (\tilde{\boldsymbol{y}} \oplus \boldsymbol{e})\boldsymbol{G}$, it is sufficient to query the errors occurring at the subvector $\tilde{\boldsymbol{y}}_{\mathcal{A}}$, namely, the error vector $\boldsymbol{e}_{\mathcal{A}}$ being sufficient.

Proof. Let the error patterns be in the set $\mathcal{E} : \{\boldsymbol{e} = (e_1, e_2, ..., e_N)\}$. The norm of $\mathcal{E}$ is the number of error patterns that the decoder is about to query. GRAND tests an error pattern $\boldsymbol{e} \in \mathcal{E}$ by adding it to $\tilde{\boldsymbol{y}}$: $\hat{\boldsymbol{x}} = \tilde{\boldsymbol{y}} \oplus \boldsymbol{e}$. Then, the source vector $\hat{\boldsymbol{u}}$ is generated as $\widehat{\boldsymbol{u}} = \hat{\boldsymbol{x}}\boldsymbol{G}$. With $\boldsymbol{u}_{\bar{\mathcal{A}}}$ being known (typically all zeros), the estimated source vector $\hat{\boldsymbol{u}}$ only depends on $\hat{\boldsymbol{u}}_{\mathcal{A}}$.

$$\boldsymbol{u}_{\mathcal{A}} = \boldsymbol{x}_{\mathcal{A}}\boldsymbol{G}_{\mathcal{A}\mathcal{A}}^{-1} = \boldsymbol{x}_{\mathcal{A}}\boldsymbol{G}_{\mathcal{A}\mathcal{A}}. \tag{A1}$$

Using equation (A1), $\hat{\boldsymbol{u}}_{\mathcal{A}}$ can be fully recovered from $\hat{\boldsymbol{x}}_{\mathcal{A}}\boldsymbol{G}_{\mathcal{A}\mathcal{A}}$. With $\hat{\boldsymbol{x}}_{\mathcal{A}} = \tilde{\boldsymbol{y}}_{\mathcal{A}} \oplus \boldsymbol{e}_{\mathcal{A}}$, it indicates that only errors at positions specified by $\mathcal{A}$ are necessary.

Appendix B Proof of Corollary 2

Corollary 1. Consider three error patterns: $\boldsymbol{e}' \in \mathcal{E}_I$, $\boldsymbol{e}'' \in \mathcal{E}_F$, and $\boldsymbol{e} = \boldsymbol{e}' + \boldsymbol{e}''$. If $\boldsymbol{e}'$ produces a correct information sequence $\hat{\boldsymbol{u}}_{\mathcal{A}}$, then the error pattern $\boldsymbol{e}$ also produces the same correct information sequence $\hat{\boldsymbol{u}}_{\mathcal{A}}$.

Proof. Let us define two sets containing error patterns:

$$\mathcal{E}_I = \{\boldsymbol{e} = (e_1, e_2, ..., e_N) \mid \text{if } i \in \bar{\mathcal{A}}, \text{ then } e_i = 0\} \tag{B1}$$

$$\mathcal{E}_F = \{\boldsymbol{e} = (e_1, e_2, ..., e_N) \mid \text{if } i \in \mathcal{A} \text{ then } e_i = 0\} \tag{B2}$$

From the definitions, it can be observed that $\mathcal{E}_I$ contains error patterns with errors only over information bit positions, whereas the set $\mathcal{E}_F$ is the set of error patterns with errors that occur only at frozen bit positions. The error pattern $\boldsymbol{e} = \boldsymbol{e}' + \boldsymbol{e}''$ can be decomposed into two parts:

$$\begin{aligned}(\boldsymbol{e}_{\mathcal{A}} \ \ \boldsymbol{e}_{\bar{\mathcal{A}}}) &= (\boldsymbol{e}'_{\mathcal{A}} \ \ \boldsymbol{e}'_{\bar{\mathcal{A}}}) + (\boldsymbol{e}''_{\mathcal{A}} \ \ \boldsymbol{e}''_{\bar{\mathcal{A}}}) \\ &= (\boldsymbol{e}'_{\mathcal{A}} + \boldsymbol{e}''_{\mathcal{A}} \ \ \boldsymbol{e}'_{\bar{\mathcal{A}}} + \boldsymbol{e}''_{\bar{\mathcal{A}}})\end{aligned} \tag{B3}$$

With the definitions of (B1) and (B2), $\boldsymbol{e}'_{\bar{\mathcal{A}}} = \boldsymbol{0}$ since $\boldsymbol{e}' \in \mathcal{E}_I$ and $\boldsymbol{e}''_{\mathcal{A}} = \boldsymbol{0}$ since $\boldsymbol{e}'' \in \mathcal{E}_F$. Then equation (B3) can be written as

$$(\boldsymbol{e}_{\mathcal{A}} \ \ \boldsymbol{e}_{\bar{\mathcal{A}}}) = (\boldsymbol{e}'_{\mathcal{A}} \ \ \boldsymbol{e}''_{\bar{\mathcal{A}}}) \tag{B4}$$

Then, the hard decision of the codeword plus the error pattern $\boldsymbol{e}$ is

$$\begin{aligned}\hat{\boldsymbol{x}} &= \tilde{\boldsymbol{y}} \oplus \boldsymbol{e} \\ (\hat{\boldsymbol{x}}_{\mathcal{A}} \ \ \hat{\boldsymbol{x}}_{\bar{\mathcal{A}}}) &= (\tilde{\boldsymbol{y}}_{\mathcal{A}} \ \ \tilde{\boldsymbol{y}}_{\bar{\mathcal{A}}}) \oplus (\boldsymbol{e}'_{\mathcal{A}} \ \ \boldsymbol{e}''_{\bar{\mathcal{A}}}) \\ &= (\tilde{\boldsymbol{y}}_{\mathcal{A}} \oplus \boldsymbol{e}'_{\mathcal{A}} \ \ \tilde{\boldsymbol{y}}_{\bar{\mathcal{A}}} \oplus \boldsymbol{e}''_{\bar{\mathcal{A}}})\end{aligned} \tag{B5}$$

With Theorem 1, $\hat{\boldsymbol{x}}_{\bar{\mathcal{A}}}$ is not relevant in calculating $\hat{\boldsymbol{u}}_{\mathcal{A}}$. Therefore, the error pattern of $\boldsymbol{e}$ eventually is equivalent as $\boldsymbol{e}'_{\mathcal{A}}$, as expressed in equation (B5). Therefore, it is equivalent to the error pattern $\boldsymbol{e}'$ since $\boldsymbol{e}'_{\bar{\mathcal{A}}} = \boldsymbol{0}$. They both produce the same information sequence: $\hat{\boldsymbol{u}}_{\mathcal{A}} = \hat{\boldsymbol{x}}_{\mathcal{A}}G_{\mathcal{A}\mathcal{A}}$.

* Corresponding author (email: liping_li@ahu.edu.cn, wenyizha@ustc.edu.cn)

Appendix C Adaptive GRAND List Decoders

Algorithm C1 Simplified ORBGRAND List for Polar Codes

Input: $\boldsymbol{y}$,$\boldsymbol{G}_{\mathcal{A}\mathcal{A}}$,$LW_{max}, HW_{max}, \Delta, \delta$

Output: $\widehat{\boldsymbol{u}}_{final}$

$\boldsymbol{y}_{\mathcal{A}} \leftarrow \boldsymbol{y}(\mathcal{A}); \tilde{\boldsymbol{y}}_{\mathcal{A}} \leftarrow sign(\boldsymbol{y}_{\mathcal{A}})$
$\boldsymbol{InfoIndex} \leftarrow sortInfo(\boldsymbol{y}_{\mathcal{A}})$
$\boldsymbol{e}_{\mathcal{A}} \leftarrow \boldsymbol{0}; \boldsymbol{L} \leftarrow \boldsymbol{0};$
while $LW(\boldsymbol{e}_{\mathcal{A}}) <= LW_{max}$ **do**
  if $LW(\boldsymbol{e}_{\mathcal{A}}) > \Delta$ **then**
    break
  end if
  if $HW(\boldsymbol{e}_{\mathcal{A}}) <= HW_{max}$ **then**
    $\widehat{\boldsymbol{u}}_{\mathcal{A}} \leftarrow (\tilde{\boldsymbol{y}}_{\mathcal{A}} \oplus \boldsymbol{e}_{\mathcal{A}}) \cdot \boldsymbol{G}_{\mathcal{A}\mathcal{A}}$
    if $CRC(\widehat{\boldsymbol{u}}_{\mathcal{A}}) == 1$ **then**
      $\boldsymbol{L} \leftarrow AddToList(\widehat{\boldsymbol{u}}_{\mathcal{A}}, \boldsymbol{L})$
      if $LW(\boldsymbol{e}_{\mathcal{A}}) == LW_{max}$ **then**
        $LW_{max} \leftarrow LW_{max} + \delta$
        $HW_{max} \leftarrow HW(\boldsymbol{e}_{\mathcal{A}})$
      end if
    end if
  end if
  if $LW(\boldsymbol{e}_{\mathcal{A}}) == LW_{max}$ and $\boldsymbol{L} == \boldsymbol{0}$ **then**
    $LW_{max} \leftarrow LW_{max} + \delta$
  end if
  $\boldsymbol{e}_{\mathcal{A}} \leftarrow NextInfoErrorPattern(\boldsymbol{e}_{\mathcal{A}}, \boldsymbol{InfoIndex})$
end while
$\widehat{\boldsymbol{u}}_{final} \leftarrow MinDistance(\boldsymbol{L}, \boldsymbol{InfoIndex})$
Return $\widehat{\boldsymbol{u}}_{final}$

To improve the performance of GRAND, the authors in [1] proposed List GRAND (LGRAND). A maximum logistic weight is set to limit the number of queries. For example, the maximum logistic weight 128 is set for $N = 128$ and $R = 0.82$. In the searching process, when an error pattern $\boldsymbol{e}$ produces a valid codeword, the maximum logistic weight is set $\mathrm{LW}(\boldsymbol{e}) + \delta$. Here, $\mathrm{LW}(\boldsymbol{e})$ denotes the logistic weight of $\boldsymbol{e}$ and δ is the additional value to be searched beyond $\mathrm{LW}(\boldsymbol{e})$. Searching stops when the maximum logistic weight is reached. Note that if no error patterns produce valid codewords, the maximum logistic weight is not updated and remains the initial maximum 128. The number of lists for LGRAND is not fixed. The first improvement of LGRAND is immediately available by applying Theorem 1 to LGRAND: for each search, only error patterns in $\mathcal{E}_I$ are tested. This decoder is denoted as LGRAND-Info in the paper.

For LGRAND-Info, an adaptive searching is proposed, called AD-LGRAND-Info, to further increase the performance while reducing the querying number, especially for low SNR regions. This adaptive searching can also be applied to LGRAND. For simplicity, if error patterns are generated based on all the code symbols (the original LGRAND), the decoder is only written as AD-LGRAND. Since adaptive operations are the same for AD-LGRAND and AD-LGRAND-Info (except for the searching space of error patterns), in the following, AD-LGRAND-Info is taken as an example to illustrate the adaptive process.

For AD-LGRAND-Info, the maximum logistic weight is not initially fixed to be a large number (such as 128 for LGRAND in [1]). Instead, the operational maximum logistic weight is adaptively increasing. To introduce this adaptive process, let us first define four parameters, namely, Δ, LW_{max}, HW_{max}, and δ of AD-LGRAND-Info:

- Δ: the global maximum logistic weight of all the error patterns;
- LW_{max}: the current maximum allowed logistic weight of the error patterns;
- HW_{max}: the current maximum allowed Hamming weight of the error patterns;
- δ: the step size to increase the logistic weight when LW_{max} is reached.

In the decoding process, when AD-LGRAND-Info generates an error pattern $\boldsymbol{e}_{\mathcal{A}}$ based on Theorem 1, the logistic weight of the error pattern $\boldsymbol{e}_{\mathcal{A}}$ is verified to determine if it is less than or equal to LW_{max}. If $\mathrm{LW}(\boldsymbol{e}_{\mathcal{A}}) \leqslant LW_{max}$ and $\mathrm{HW}(\boldsymbol{e}_{\mathcal{A}}) \leqslant HW_{max}$ (here $\mathrm{HW}(\boldsymbol{e}_{\mathcal{A}})$ is the Hamming weight of $\boldsymbol{e}_{\mathcal{A}}$), then this pattern $\boldsymbol{e}_{\mathcal{A}}$ is used to check the vector $\widehat{\boldsymbol{u}}_{\mathcal{A}} = (\tilde{\boldsymbol{y}}_{\mathcal{A}} \oplus \boldsymbol{e}_{\mathcal{A}}) \cdot \boldsymbol{G}_{\mathcal{A}\mathcal{A}}$. If $\widehat{\boldsymbol{u}}_{\mathcal{A}}$ passes the CRC check, then it is added to the list $\boldsymbol{L}$. Meanwhile, the current logistic weight is verified to determine if LW_{max} is reached. If yes, then the decoder increases the maximum allowed logistic weight to be

$LW_{max} + \delta$. The maximum allowed Hamming weight is also set to be the current Hamming weight of this error pattern. This process is illustrated in lines 12–14 of Algorithm 2, indicating that, at the current LW_{max}, a codeword is found, and more error patterns should be verified. Further, LW_{max} can be reached, and the list $\boldsymbol{L}$ is empty. In this case, the operational maximum allowed logistic weight is also increased to $LW_{max} + \delta$. This operation is illustrated in lines 18–19 of Algorithm 2, which indicates that up to LW_{max}, no codewords are found yet, and more searching is required. Note that the logistic weight of all error patterns must be less than or equal to the global maximum Δ. When searching is finished, the most likely information sequence from the list $\boldsymbol{L}$ is declared as the final estimated sequence by

$$\boldsymbol{u}_{final} = \arg\min_{\widehat{\boldsymbol{u}} \in \boldsymbol{L}} ||\widehat{\boldsymbol{u}}\boldsymbol{G} - \boldsymbol{y}|| \tag{C1}$$

Here, the minimum distance rule selects the most likely sequence.

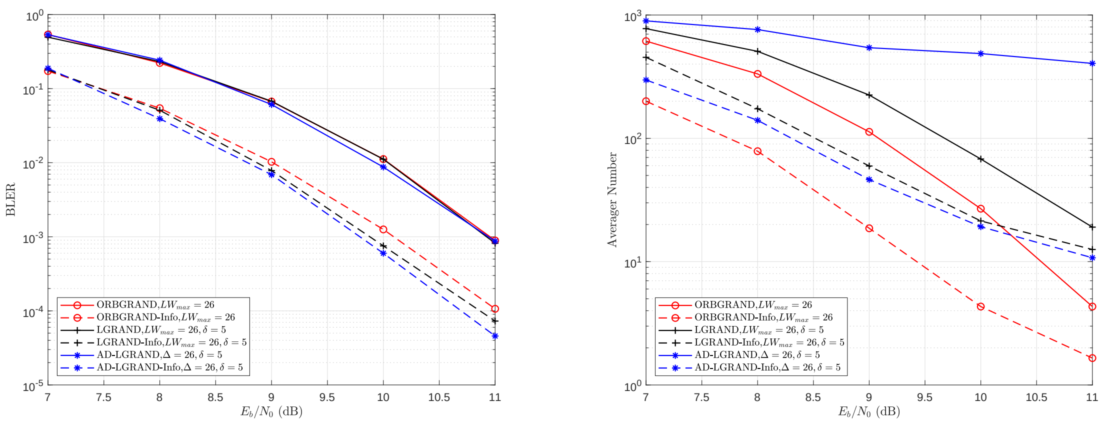
Appendix C.1 Comparison between AD-LGRAND and LGRAND

According to Corollary 1, if the information sequence produced by the error pattern $\boldsymbol{e}' \in \mathcal{E}_I$ passes the CRC check, then the error pattern $\boldsymbol{e} = \boldsymbol{e}' + \boldsymbol{e}''$ must pass the CRC check since errors $\boldsymbol{e}'' \in \mathcal{E}_F$ do not affect the decision of $\boldsymbol{u}_{\mathcal{A}}$. For example, let $N = 4$, $\mathcal{A} = \{3, 4\}$, $\bar{\mathcal{A}} = \{1, 2\}$, the received LLRs$=\{0.5, -0.9, 1.2, 0.2\}$, and $\delta = 1$. The order of the LLR is $\boldsymbol{r} = (2, 3, 4, 1)$. If the first error pattern $\boldsymbol{e}' = \{0, 0, 0, 1\}$ produces an information sequence passing the CRC check, then LGRAND increases LW_{max} to 2. Further, the next error pattern $\boldsymbol{e} = \{1, 0, 0, 0\}$ does not passes the CRC check, and LGRAND ends the decoding procedure. However, AD-LGRAND continues the decoding procedure and increases LW_{max} to 3. Then, the next error pattern $\boldsymbol{e} = \{1, 0, 0, 1\}$ also passes the CRC check and is added to the list. AD-LGRAND repeats the above procedure until the logistic weight of the error pattern is larger than LW_{max}.

With LW($\boldsymbol{e}''$) ranging from 1 to $LW_{max} -$ LW($\boldsymbol{e}'$), the error pattern $\boldsymbol{e}$ very likely can reach LW_{max}, and all the corresponding information sequences can pass the CRC check. Note that every time a valid information sequence is found and LW_{max} is reached, LW_{max} is updated by an additional δ until the global maximum is met. While searching over all code symbols, $\boldsymbol{e}' \in \boldsymbol{e}_{\mathcal{A}}$ may update LW_{max} by an additional δ to $LW_{max} = LW_{max} + \delta$.

By comparison, LGRAND updates LW_{max} only once when the first error pattern produces a valid information sequence: $LW_{max} = LW(\boldsymbol{e}') + \delta$, especially for low-rate codes. In this regard, AD-LGRAND is not efficient in terms of searching efficiency compared with LGRAND, but can potentially increase the performance because of the potentially larger LW_{max}. With an increase in the code rate, the size of $\bar{\mathcal{A}}$ decreases, resulting in fewer error patterns $\boldsymbol{e}'' \in \mathcal{E}_F$ and less possibility of reaching LW_{max} with valid information sequences.

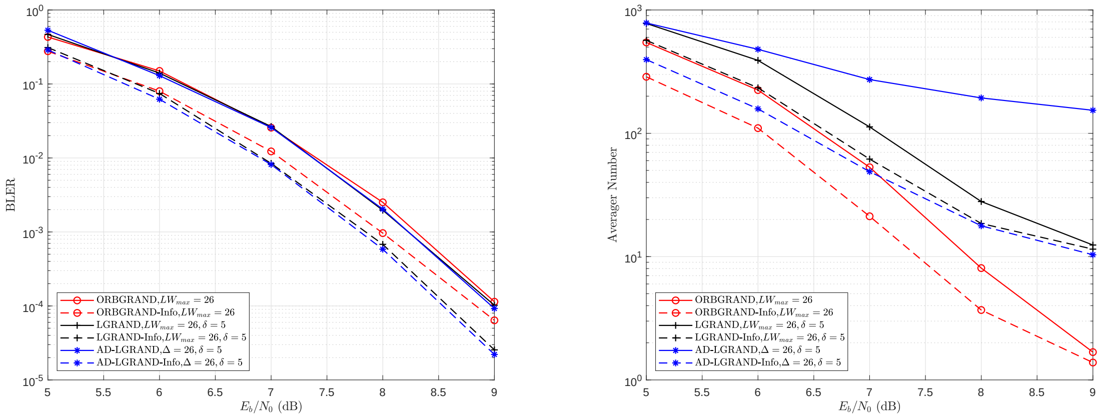
Appendix D Simulation Setting



(a) BLER (b) Average number of queries

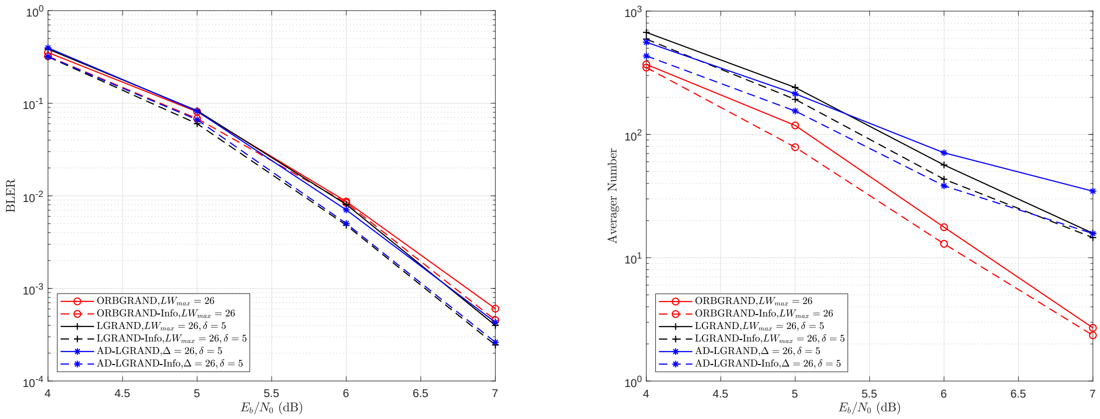
Figure D1 Performance of different GRAND decoders for polar codes (128,32) with 11 CRC bits.

In the experiment, the modulation is binary phase shift keying over additive white Gaussian channels. For all the decoders, the maximum logistic weight is set to be the same: $\Delta = 26$. For AD-LGRAND and AD-LGRAND-Info, the global maximum logistic weight is set to 26 (found from experiments for this configuration). For each E_b/N_0, an initial operation maximum allowed logistic weight LW_{max} can be set. A general rule is that for small E_b/N_0s, a large LW_{max} is required, and vice versa. The initial LW_{max} for the simulated E_b/N_0s in the simulation is $[9, 8, 7, 6, 5]$. The adaptive setting of the maximum allowed logistic weight can help AD-LGRAND (blue solid lines with asterisks) achieve a small gain compared with LGRAND. However, the gain of AD-LGRAND-Info (blue dashed line with asterisks) is the largest compared to the other two new decoders: ORBGRAND-Info and LGRAND-Info. The gain of AD-LGRAND-Info also increases with an increase in E_b/N_0, as shown in the larger gap in Fig. D1(a). Regarding the average number of queries, AD-LGRAND



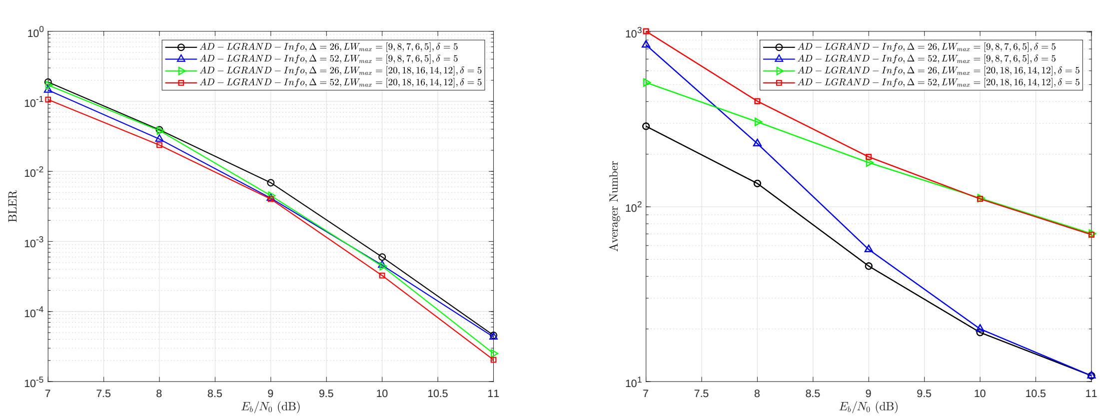
(a) BLER (b) Average number of queries

Figure D2 Performance of different GRAND decoders for polar codes (128,64) with 11 CRC bits.



(a) BLER (b) Average number of queries

Figure D3 Performance of different GRAND decoders for polar codes (128,96) with 11 CRC bits.



(a) BLER (b) Average number of queries

Figure D4 Performance of AD-LGRAND-Info decoders for polar codes (128,32) with different simulation parameters.

requires 20 times that of LGRAND. This is because of the difference in the searching strategy: LGRAND updates the LW_{max} only once (the first time a codeword is found), whereas AD-LGRAND updates the LW_{max} whenever a codeword is found until the global maximum, as explained in Section Appendix C.1. AD-LGRAND-Info requires a much smaller querying number (only 56% of LGRAND) because of the more precise searching over information positions and the early stopping strategy. Overall, among all the new decoders, AD-LGRAND-Info provides the largest gain (1.5 dB compared to that of ORBGRAND at BLER 10^{-3}), and its average number of queries is moderate (2.5 times that of ORBGRAND at the same BLER level). For other code rates, such as $R = 0.5$ and $R = 0.75$, AD-LGRAND-Info also shows improvements, as shown in Figs. D2 and D3. Clearly, the improvement is not as large as that for low code rates because AD-LGRAND-Info searches over the information positions: the lower that code rate, the smaller the searching space, and the larger the gain.

As shown in Fig. D4(a), when the initial maximum logistic weight LW_{max} increases from $[9, 8, 7, 6, 5]$ to $[20, 18, 16, 14, 12]$, the BLER performance is better in the high SNR region (refer to the black circled and the green lines with triangles). This is because the decoder initially can search more codewords (the stop condition is the initial $LW_{max}+\delta$), and in the high SNR region, it enables the correct codeword to be captured before searching stops. When the global maximum logistic weight Δ changes from 26 to 52, the BLER performance is better in the low SNR region (refer to the pair of black circles and blue lines with up triangles or the pair of green lines with triangles and the red line with squares). The reason is that, for low SNR regions (with more errors), the list can be filled so quickly that the correct one has no chance to enter the list and the global maximum Δ is met. Therefore, by increasing the global maximum Δ, the correct codeword has more chance to be captured.

References

1 Syed Mohsin Abbas, Marwan Jalaleddine and Warren J. Gross, “A practical way to achieve Maximum Likelihood Decoding,” [Online]. Available: https://arxiv.org/abs/2109.12225.