LDPC Codes for Portable DNA Storage

Peng Fei
Center for Pervasive Communications and Computing
University of California, Irvine
Email: pfei1@uci.edu

Zhiying Wang
Center for Pervasive Communications and Computing
University of California, Irvine
Email: zhiying@uci.edu

Abstract—DNA becomes an attractive storage medium in recent years for its ultra-high density, millennial-long endurance, and efficient replication. In this work we consider DNA storage that uses the affordable and portable nanopore sequencing as the reading mechanics. Unlike traditional data storage systems, errors occur asymmetrically among the four types of nucleotide bases of DNA. Quaternary codes can be employed for error correction, but suffer from high complexity. In this paper, we design binary LDPC codes with a turbo-like decoder for the DNA storage channel. Simulation results show that our binary LDPC codes have a similar bit-error rate but with a speed up by an order of magnitude compared to quaternary codes.

I. INTRODUCTION

With the fast growth of data, DNA storage attracts much attention for its extremely large data densities, integrity in non-ideal conditions over millennial, and efficient replication (e.g., [1]–[6]). In DNA storage, the writing process is called synthesis, which joins nucleotide symbols and produces the desired DNA string. Reading is completed through a DNA sequencer that reads the string and translates it to digital data. Among the DNA sequencing technologies, nanopore sequencing performed on the MinION device [7] is a promising one due to its low cost, scalability, and portability, and has been demonstrated suitable for DNA storage [5]. In nanopore sequencing, a DNA sequence is passed through a nano-sized hole and current variation is measured to estimate the sequence symbols.

Errors in nanopore sequencing [8] should be addressed in order to maintain data integrity. In [9] burst deletion errors are considered and associated codes are constructed. In [10] a channel model of nanopore sequencing is established capturing non-linear inter-symbol interference, deletions, and substitutions. In [11] the measured current is modeled as a quaternary amplitude modulation with additive white Gaussian noise. In [12] it is observed that substitution errors occur asymmetrically among the four bases of nucleotides \((A,T,G,C)\), and the minimum asymmetric Lee distance are studied.

The goal of this work is to design LDPC codes for the asymmetric substitutions of nanopore sequencing. In stead of studying the minimum code distance as in [12], we focus on the performance in terms of bit error rate, and LDPC codes are chosen for its near-capacity performance and low encoding/decoding complexity. As the DNA string is a sequence of alphabet size 4, a quaternary code is a natural candidate for error correction. Although the LDPC code supports non-binary alphabets, it suffers from the magnificent slow speed in decoding compared to the binary case. Hence, we focus on binary LDPC codes for DNA storage.

In this paper we present an asymmetric nanopore channel model. We propose to use two binary LDPC codes for one quaternary DNA sequence, and show decoding algorithms that exchange information between the two LDPC codes. The information exchange step is inspired by Turbo codes, and addresses the asymmetry in the channel errors. Simulation results show that our proposed binary codes have bit error rate comparable to quaternary codes, but has a 10x speed up.

II. PRELIMINARIES

We first introduce the asymmetric channel model of nanopore sequencing. In an \((n,k)\) DNA code, \(2k\) bits of information are encoded into a DNA codeword \(x = (x_1, x_2, \ldots, x_n)\), \(x_i \in \{A, T, G, C\}, 1 \leq i \leq n\). Nanopore sequencing can read the DNA codeword by monitoring small changes in the ionic current flowing through the pore and dwell time in the nanopore [8], the distribution of which is shown approximately in Fig. 1. Hard estimation decisions are made from the current and time measurements with substitution errors. In other words, the channel is discrete and memoryless. The received word is denoted by \(y = (y_1, y_2, \ldots, y_n)\), \(y_i \in \{A, T, G, C\}, 1 \leq i \leq n\).

From Fig. 1, we can see that the nanopore channel is asymmetric among the possible erroneous transitions: transitions between \(T\) and \(C\) are more often, however less so between \(G\) and \(A\). The error probabilities are shown as in Fig. 2, for some \(p_3 \ll p_1 < p_2\). This is a slight generalization of the model in [12], and we assume \(p_1, p_2, p_3\) satisfy the specific relation that

\[
p_1 = \lambda, p_2 = 3\lambda, p_3 \approx 0, \quad (1)
\]
for some parameter \(0 < \lambda < 1/3\). We note that for other choices of error probabilities such that \(p_3 \ll p_1 \ll p_2\), our proposed coding scheme works similarly.

In order to approach channel capacity, we need to optimize the mutual information of the channel input and output, among the possible distributions for the transmitted symbols. We now demonstrate that the uniform distribution is near optimal. For this purpose, we consider the error probabilities as in (1). Denote by \(C_{ap}\) the channel capacity, and \(X,Y\) the random variables of the channel input and output. Due to the symmetry of \(A,G\) and \(C,T\), we set \(q_1 = q_3, q_2 = q_4\) as the probabilities for the transmitted symbols \(A,G,C,T\), respectively. The capacity

\[
C_{ap} = \max_{q_1,q_2} I(X;Y) = \max_{q_1,q_2} H(Y) - H(Y|X)
\]

\[
= -2(q_1(1 - p_1) + 2q_2p_1)\log(q_1(1 - p_1) + 2p_1q_2) - 2(2q_1p_3 + 2q_3(1 - p_2))\log(2q_1p_3 + 2q_3(1 - p_2)) + 2q_1((1 - 2p_1 - p_3)\log(1 - 2p_1 - p_3) + p_1\log(p_1) + p_3\log(p_3))
\]

\[
+ 2q_3((1 - 2p_1 - p_2)\log(1 - 2p_1 - p_2) + p_1\log(p_1) + p_2\log(p_2)).
\]

is maximized when the percentage of \(A\) is from 25 to 31, for \(0 < \lambda \leq 0.04, p_3 = 0\). We can see this probability is close to \(1/4\). Accordingly, all other 3 symbols also have a probability close to \(1/4\). Thus one can use the uniform distribution for the transmitted symbols, hence a linear block code with uniform information symbols is suitable for such channels.

We map the four types of nucleotides to two bits. When a quaternary code is used, these two bits correspond to the vector representation of elements in the finite field \(GF(4)\); when a binary code is used, they are viewed as a 2-bit vector in \(\{0,1\}^2\). In this paper, the mapping is chosen to be \(A \mapsto (0,0), G \mapsto (1,1), T \mapsto (1,0), C \mapsto (0,1)\). According to the channel model in Fig. 2, another (non-equivalent) mapping can be \(A \mapsto (0,1), G \mapsto (1,1), T \mapsto (0,0), C \mapsto (1,0)\). When the symbols are viewed as 2 bits, the choice of the mapping affects the raw bit error rate (BER). Under uniform distribution, the former mapping gives the raw BER

\[
BER = \frac{1}{4}(p_1 + 2p_2 + p_1) + \frac{1}{4}(p_1 + 2p_2 + p_1)
\]

\[
+ \frac{1}{4}(p_1 + 2p_2 + p_1) + \frac{1}{4}(p_1 + 2p_2 + p_1)
\]

\[
= 2p_1 + p_2 + p_3.
\]

For the latter mapping, we have a raw BER

\[
BER = 4p_1 + 2p_2 + \frac{1}{2}p_3.
\]

Since \(p_3\) is small, we ignore the last term in both BERs. In this paper, we only consider the former mapping, and assume that the condition \(p_2 < 4p_1\) is satisfied.

Next, we briefly review the LDPC decoding sum-product algorithm (SPA) (see, e.g., [13]). For simplicity, we describe the algorithm for the binary case. Consider a binary LDPC code defined by an \((n - k) \times n\) parity-check matrix \(H = (h_{i,j})\). The code can be represented by a Tanner graph, with \(n\) variable nodes (VNs) and \(n - k\) check nodes (CNs). An edge exists between the \(j\)-th variable \(V N_j\) and the \(i\)-th check \(C N_i\) if and only if \(h_{i,j} = 1\). Let \(N(i)\) denote the set of neighboring nodes of the node \(i\). When the Tanner graph has no cycles, SPA can be proved to be the bit-wise maximum a posteriori (MAP) decoder [14].

Let \(x^i = (x_1^i, x_2^i, \ldots, x_n^i)\), \(y^i = (y_1^i, y_2^i, \ldots, y_n^i)\) be the transmitted and received binary words, respectively. In SPA, the log-likelihood ratio (LLR), \(\log p[y_1^i=0|x^i]\), is passed between neighboring variable nodes and check nodes in iterations. The LLR passed from node \(i\) to node \(j\) in the algorithm, called extrinsic information, is the aggregated information of the incoming messages to node \(i\) from its neighbors except \(j\), i.e., \(N(i) - \{j\}\).

Let \(i^\text{ter}\) be the index of the current iteration. Define the information passed from the variable node \(V N_j\) to the check node \(C N_i\) by \(L_{j \rightarrow i}^{i^\text{ter}}\), where \(1 \leq j \leq n\) and \(i \in N(i)\). Define the information passed from the check node \(V N_j\) to the variable node \(V N_j\) by \(L_{i \rightarrow j}^{i^\text{ter}}\), where \(1 \leq i \leq n - k\) and \(j \in N(i)\). In the 0-th iteration, \(L_{j \rightarrow i}^{0} = Ch_j\), where for the given received \(y_j\), \(Ch_j\) is the channel information defined by

\[
Ch_j = \log \frac{p(y_j^i|x_j^i = 0)}{p(y_j^i|x_j^i = 1)}.
\]

The message from a CN to a VN is given by

\[
L_{i \rightarrow j}^{i^\text{ter}} = 2\tanh^{-1}(\prod_{j' \in N(i)-\{j\}}\tanh(\frac{1}{2}L_{j' \rightarrow i}^{i^\text{ter}-1})).
\]

The message from a VN to a CN is updated by

\[
L_{j \rightarrow i}^{i^\text{ter}} = Ch_j + \sum_{i' \in N(j)-\{i\}} L_{i' \rightarrow j}^{i^\text{ter}-1}.
\]

Then the overall soft information is obtained for each \(V N_j\):

\[
L_{j}^{\text{soft},i^\text{ter}} = Ch_j + \sum_{i \in N(j)} L_{i \rightarrow j}^{i^\text{ter}-1},
\]
and the corresponding symbols is decoded as \( \hat{x}_{i,\text{iter}}^1 = 1 \) if \( L_{i,\text{iter}}^{\text{soft}} < 0 \), and decoded as 0 otherwise. The algorithm stops if \( \hat{x}_{i,\text{iter}}^1 \cdot H^T = 0 \) or the maximum number of iterations is reached.

### III. Binary LDPC Codes for DNA Storage

In this section, we present the binary LDPC codes for DNA storage. We propose our decoder that is inspired by Turbo decoder.

The construction of the binary codes is straightforward: we take \( k \) bits of information, and encode them into a binary LDPC codeword of length \( n \). Repeat for a second binary LDPC code. The two LDPC codes are denoted by \( C^1 \), \( C^2 \), and have the same rate \( k/n \). The two codewords are denoted by \( x^i = (x^i_1, x^i_2, \ldots, x^i_n) \), for \( i = 1, 2 \), respectively. Then we store \( x_j = (x^1_j, x^2_j) \) as one DNA quaternary symbol, for all \( 1 \leq j \leq n \). After the nanopore channel, we receive the corrupted \( j \)-th DNA symbol, \( y_j = (y^1_j, y^2_j) \), \( 1 \leq j \leq n \), and then form the two binary received words \( y^i = (y^i_1, y^i_2, \ldots, y^i_n) \), \( i = 1, 2 \).

Below, we use notations similar to the sum-product algorithm described in Section II, except that we add a superscript 1 or 2 to denote the corresponding notation for the first or the second LDPC code. For example, denote by \( VN^1_j, VN^2_j \) the \( j \)-th variable node in the first and the second LDPC codes, respectively; denote by \( L^1_{i,\text{iter}}, L^2_{i,\text{iter}} \) the information passed from note \( j \) to node \( i \) in the two codes.

**The baseline decoder.** We propose a baseline decoder that runs two SPAs on the two codewords independently, with some modifications to the channel information \( Ch^1_j, Ch^2_j \). Conditioned on the received quaternary \( y \), the bit-wise MAP rule of the first LDPC is:

\[
\hat{x}_i^1 = \arg \max_{x_i^1 \in \{0,1\}} p(x_i^1 | y) = \arg \max_{x_i^1 \in \{0,1\}} \frac{p(y | x_i^1)}{p(x_i^1)} \tag{16}
\]

\[
= \arg \max_{x_i^1 \in \{0,1\}} \sum_{x_i^1} p(x_i^1 | y) \tag{17}
\]

\[
= \arg \max_{x_i^1 \in \{0,1\}} \sum_{x_i^1} p(y | x_i^1) \tag{18}
\]

\[
= \arg \max_{x_i^1 \in \{0,1\}} \sum_{x_i^1} \prod_{j=1}^n p(y_j | x_i^1) \mathbb{1}_{x_i \in C^1}, \tag{19}
\]

where \( \sim x_i^1 \) means all possible binary vectors \( x^1 \in \{0,1\}^n \) such that \( x_i^1 \) is fixed, and \( \mathbb{1}_{x_i \in C^1} \) is the indicator function that \( x^1 \) is a codeword of the first LDPC code \( C^1 \). Here (18) follows from Bayes’ rule and \( y \) being fixed, and (19) follows from the memoryless channel and the uniformity of the transmitted codewords. In a conventional binary LDPC, the bit-wise MAP rule has the term \( \prod_j p(y_j | x_i^1) \) instead of \( \prod_j p(y_j | x_i^1) \) in (19), corresponding to the channel information. Effectively, the new channel has the binary input \( x_i^1 \) and the quaternary output \( y_i = (y^1_i, y^2_i) \), and (12) is modified to

\[
Ch_j^1 = \log \frac{p(y_j | x_i^1 = 0)}{p(y_j | x_i^1 = 1)}. \tag{20}
\]

When \( y_j = C \),

\[
Ch_j^1 = \log \frac{p(y_j = C | x_j = C) + p(y_j = C | x_j = A)}{p(y_j = C | x_j = G) + p(y_j = C | x_j = T)} \tag{21}
\]

\[
= \log \frac{1 - p_1 - p_2}{p_1 + p_2}. \tag{22}
\]

Similarly, when \( y_j = A \),

\[
Ch_j^1 = \log \frac{p(y_j = A | x_j = A) + p(y_j = A | x_j = C)}{p(y_j = A | x_j = G) + p(y_j = A | x_j = T)} \tag{23}
\]

\[
= \log \frac{1 - p_1 - p_3}{p_1 + p_3}. \tag{24}
\]

When \( y_j = T \),

\[
Ch_j^1 = \log \frac{p(y_j = T | x_j = C) + p(y_j = T | x_j = A)}{p(y_j = T | x_j = G) + p(y_j = T | x_j = T)} \tag{25}
\]

\[
= \log \frac{1 - p_1 + p_2}{p_1 - p_2}. \tag{26}
\]

When \( y_j = G \),

\[
Ch_j^1 = \log \frac{p(y_j = G | x_j = C) + p(y_j = G | x_j = A)}{p(y_j = G | x_j = G) + p(y_j = G | x_j = T)} \tag{27}
\]

\[
= \log \frac{p_1 + p_3}{1 - p_1 - p_3}. \tag{28}
\]

The same analysis can be applied to the second LDPC. Much like conventional LDPC, for cycle-free Tanner graphs, the baseline decoder is a bit-wise MAP decoder, conditioned on \( y \), for each individual LDPC.

However, this baseline decoder can be improved upon. Observe that in Fig. 2, if the received symbol is \( y_j = C = (0, 1) \), and the first bit is decoded to be 1, then it is much more likely that the transmitted symbol is \( x_j = T = (1, 0) \) than the case of \( x_j = G = (1, 1) \). One can verify that given \( y_j = C \) or \( T \), the probability of having complementary bits \( (x^1_j = 1 - x^2_j) \) is much larger than identical bits \( (x^1_j = x^2_j) \). We note that Turbo codes (e.g., [13]) have a similar property. In particular, the Turbo codes can be viewed as two separate convolutional codes, sharing identical but permuted information bits. The two identical information bits pass extrinsic information between each other hence connecting the two convolutional codes. Inspired by Turbo codes, we propose to pass auxiliary information between the two bits when receiving \( C \) or \( T \).

In our proposed decoders, the two LDPC codes run their sum-product algorithms as in Section II concurrently, with the modifications as below. If a received symbol \( y_j = C \) or \( T \), then an auxiliary term is passed from \( VN^1_j \) to \( VN^2_j \) in each iteration. We note that different from Turbo codes, the auxiliary term does not appear as an addend in the overall soft information of (15) due to two reasons: (i) the two bits \( y^1_j, y^2_j \) are not deterministically complementary of each other, (ii) the function in (13) is nonlinear. Therefore, we cannot eliminate the auxiliary term entirely when information is passed back from \( VN^2_j \) to \( VN^1_j \), and the algorithm for the auxiliary information exchange is not straightforward. Regardless, we call the auxiliary term extrinsic information to highlight the
resemblance to Turbo codes. Next, we propose three decoding algorithms.

**Algorithm 1.** Our first decoding algorithm is shown in Algorithm 1. In this algorithm, we pass the extrinsic information from $V N_j^1$ to $V N_j^2$ if the received quaternary symbol $y_j = C$ or $T$, and the extrinsic information of the $j$-th variable node is defined as

$$L_{e_j}^{1 \rightarrow 2} = \sum_{i' \in N(j)} L_{i' \rightarrow j}^{1, \text{iter}-1}.$$  \hfill (29)

In every iteration, the extrinsic information is passed from the first LDPC to the second LDPC. To generate the message from a variable node $V N_j^2$ to a check node $CN_j^2$ in the second LDPC code, we combine the above extrinsic information, the channel information of (20), and the messages from $V N_j^2$, $i' \in N^2(j)$. Specifically, equation (14) is modified to

$$L_{j \rightarrow i}^{2, \text{iter}} = C h_j^2 + \sum_{i' \in N^2(j) - \{i\}} L_{i' \rightarrow j}^{2, \text{iter}-1} - \alpha L_{e_j}^{1 \rightarrow 2},$$  \hfill (30)

for some constant parameter $0 < \alpha < 1$. Moreover, to generate the overall soft information of $V N_j^2$, (15) is modified to

$$L_{j}^{2, \text{soft}, \text{iter}} = C h_j^2 + \sum_{i' \in N^2(j)} L_{i' \rightarrow j}^{2, \text{iter}-1} - \alpha L_{e_j}^{1 \rightarrow 2}. \hfill (31)$$

Equivalent to (30) (31), define the new channel information as

$$C h_{j}^{2, \text{iter}} = C h_j^2 - \alpha L_{e_j}^{1 \rightarrow 2},$$  \hfill (32)

and simply replace $C h_j^2$ with $C h_{j}^{2, \text{iter}}$ in the sum product algorithm of Section II. The extrinsic information from $V N_j^2$ to $V N_j^1$ is defined similarly, and the algorithm of the first LDPC uses the new channel information similar to (32).

In expression (29) of the extrinsic information, the coefficient $-\alpha$ captures the fact that $x^1_j$ and $x^2_j$ are complementary with a large probability. Note that $L_{e_j}^{1 \rightarrow 2}$ is the overall soft information at $V N_j^2$ as in (15) except the channel information term. The channel information is excluded because it is unchanged during the iterations, and would be amplified over time since $\alpha < 1$.

**Algorithm 2.** In the second algorithm, we derive an alternative channel information expression. In the following, we assume $y_j = C$, but the derivation for the case $y_j = T$ is similar and hence omitted.

We define $L_{e_j}^{1 \rightarrow 2}$ the same as (29), and we treat it as the LLR of $x^2_j$ given $y_j = C$:

$$\log \frac{p(x^2_j = 0|y_j = C)}{p(x^2_j = 1|y_j = C)} = L_{e_j}^{1 \rightarrow 2}.$$  \hfill (33)

Since $x^2_j \in \{0, 1\}$, the corresponding probability can be computed from LLR:

$$p(x^2_j|y_j = C) = \frac{e^{-L_{e_j}^{1 \rightarrow 2}}}{1 + e^{-L_{e_j}^{1 \rightarrow 2}}} e^{\frac{s_j L_{e_j}^{1 \rightarrow 2}}{2}}.$$  \hfill (34)

**Algorithm 1 Binary LDPC decoder for DNA storage**

**Initialize:**

1: For $1 \leq j \leq n$, $i \in N(j)$ initialize $L_{j \rightarrow i}^{1, \text{iter}} = C h_j^1$ and $L_{j \rightarrow i}^{2, \text{iter}} = C h_j^2$ by (20)

2: for $\text{iter} = 1 : I_{\text{max}}$ do

3: Stop if $x^{1, \text{iter}}(H^1)T = 0$, $x^{2, \text{iter}}(H^2)T = 0$ or $\text{iter} = I_{\text{max}}$

Message passing between two codes

4: for $1 \leq j \leq n$ do

5: if $y_j = C = (0, 1)$ or $y_j = T = (1, 0)$ then

6: $C h_j^{1, \text{iter}} = C h_j^1 - \alpha \sum_{i' \in N(j)} L_{i' \rightarrow j}^{2, \text{iter}-1}$

7: $C h_j^{2, \text{iter}} = C h_j^2 - \alpha \sum_{i' \in N(j)} L_{i' \rightarrow j}^{2, \text{iter}-1}$

8: else

9: $C h_j^{1, \text{iter}} = C h_j^1$

10: $C h_j^{2, \text{iter}} = C h_j^2$

11: end if

12: end for

Check node update

13: for $1 \leq i \leq n$ do

14: $L_i^{1, \text{iter}} = 2 \tanh^{-1}(\prod_{j' \in N(i) - \{j\}} \tanh(\frac{1}{2} L_{j' \rightarrow i}^{1, \text{iter}}))$

15: $L_i^{2, \text{iter}} = 2 \tanh^{-1}(\prod_{j' \in N(i) - \{j\}} \tanh(\frac{1}{2} L_{j' \rightarrow i}^{2, \text{iter}}))$

16: end for

Variable node update

17: for $1 \leq j \leq n$, $i \in N(j)$ do

18: $L_{j \rightarrow i}^{1, \text{iter}} = C h_j^{1, \text{iter}} + \sum_{i' \in N(j) - i} L_{i' \rightarrow j}^{2, \text{iter}}$

19: $L_{j \rightarrow i}^{2, \text{iter}} = C h_j^{2, \text{iter}} + \sum_{i' \in N(j) - i} L_{i' \rightarrow j}^{1, \text{iter}}$

20: end for

21: for $i = 1 : n$ do

22: $L_{i}^{1, \text{soft}, \text{iter}} = C h_i^{1, \text{iter}} + \sum_{i' \in N(j)} L_{i' \rightarrow j}^{1, \text{iter}}$

23: $x^2_i = \begin{cases} \begin{array}{ll} 0, & \text{if } L_{i}^{1, \text{soft}, \text{iter}} < 0, \\ 1, & \text{else}. \end{array} \end{cases}$

24: Compute $x^2_i$ similarly

25: end for

26: end for

Assume $y_j = C$, from (20), a new channel information from the extrinsic information $L_{e_j}^{1 \rightarrow 2}$ in iteration $\text{iter}$ is set to be

$$C h_{j}^{1 \rightarrow 2, \text{iter}} = \log \frac{p(y_j = C|x^2_j = 0)}{p(y_j = C|x^2_j = 1)} = \log \frac{p(x^2 = 0|y_j = C)}{p(x^2 = 1|y_j = C)}.$$  \hfill (35)

which follows from Bayes’ rule and the uniformity of $x^2_j$. Moreover,

$$p(x^2_j = 0|y_j = C) = \sum_{\gamma \in \{0, 1\}} p(x^2_j = 0|x^1_j = \gamma, y_j = C) p(x^1_j = \gamma|y_j = C).$$  \hfill (36)

When $\gamma = 0$,

$$p(x^2_j = 0|x^1_j = 0, y_j = C)$$  \hfill (37)

(39)

(40)
\[
\frac{p(x_j^2 = 0, x_j^1 = 0, y_j = C)}{p(x_j^2 = 0, y_j = C)} = 1 - \frac{2p_1 - p_2}{p_1 + p_2} \frac{p(x_j^1 = 0|y_j = C)}{p(x_j^1 = 1|y_j = C)} + \frac{p_1}{p_1 + p_2} \frac{p(x_j^1 = 0|y_j = C)}{p(x_j^1 = 1|y_j = C)}.
\]

for \(j = 1, 2\), the observed symbol \(y_j\) is \(C\) or \(T\), and new nodes \(u_j, e_j\) are added.

\[
p(x_j^2 = 0|y_j = T) = \frac{p_1}{1 - p_1 - p_2} \frac{p(x_j^2 = 0|y_j = T)}{p_1 + 2p_2} \frac{p(x_j^1 = 1|y_j = T)}{p(x_j^1 = 1|y_j = T)}.
\]

**Algorithm 3.** In the third decoding algorithm, we add an imaginary variable node \(u_j\) and an imaginary check node \(e_j\) in the Tanner graph, for every \(j\) such that \(y_j = C\) or \(T\). The imaginary nodes serve the purpose of extrinsic information exchange. As Fig. 3 shows, after iteration \(\text{iter} - 1\), we pass the variable message from \(V_N^j\) and \(u_j\) to the check node \(e_j\). Then we pass the check message from \(e_j\) to \(V_N^j\). The variable node \(u_j\) indicates that \(V_N^j\) and \(V_N^j\) are complementary with a large probability. In particular,

\[
u_j + x_j^1 + x_j^2 = 0,
\]

where \(u_j\) is set to have a fixed LLR of

\[
L_{u_j} = \log \frac{p(u_j = 0|y_j = C)}{p(u_j = 1|y_j = C)} = \log \frac{p(u_j = 0, y_j = C|p(y_j = C)}{p(u_j = 1, y_j = C|p(y_j = C)} + \frac{1}{p_1 + p_2} \frac{p(x_j = 0|y_j = C)}{p(x_j = 1|y_j = C)} + \frac{p_1}{p_1 + p_2} \frac{p(x_j = 1|y_j = T)}{p(x_j = 1|y_j = T)} - \frac{1}{p_1 + p_2} \frac{p(x_j = 1|y_j = T)}{p(x_j = 1|y_j = T)}.
\]

**IV. SIMULATION**

We construct LDPC with the degree distribution pair \(\rho(X) = \sum_{d=1}^{d_{\min}} p_d X^{d-1}\) and \(\lambda(X) = \sum_{d=1}^{d_{\max}} \lambda_d X^{d-1}\), where \(\rho_d\) denotes the fraction of all edges connected to degree-\(d\) check nodes, \(d_{\min}\) is the maximum check node degree, and \(\lambda_d\) and \(d_{\max}\) are defined similarly for variable nodes. A good distribution pair is found using density evolution [13], according to which a random parity-check matrix is picked.

The bit error rate of quaternary codes, the 3 proposed decoding algorithms, and the baseline binary decoder is shown in Fig. 4. The code has \(n = 1000, k = 500\). For Algorithm 1, we pick the best parameter \(\alpha = 0.1\). We can see that the error rate of Algorithms 2

\[\text{LDPC 1 Check Nodes} \quad \text{LDPC 1 Variable Nodes} \quad \text{LDPC 2 Variable Nodes} \quad \text{LDPC 2 Check Nodes}\]

\[\text{CN}_1, \text{CN}_2, \text{CN}_3, \text{CN}_4, \text{CN}_5, \text{CN}_6, \text{CN}_7, \text{CN}_8, \text{CN}_9, \text{CN}_{10}\]

\[\text{VN}_1, \text{VN}_2, \text{VN}_3, \text{VN}_4, \text{VN}_5, \text{VN}_6, \text{VN}_7, \text{VN}_8, \text{VN}_9, \text{VN}_{10}\]
TABLE I
AVERAGE RUNNING TIME COMPARISON BETWEEN OUR METHODS AND QUATERNARY LDPC.

<table>
<thead>
<tr>
<th></th>
<th>Method 1</th>
<th>Method 2</th>
<th>Method 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>one iteration</td>
<td>0.038s</td>
<td>0.06s</td>
<td>0.05s</td>
</tr>
<tr>
<td>one codeword</td>
<td>1.08s</td>
<td>1.05s</td>
<td>1.05s</td>
</tr>
</tbody>
</table>

Fig. 4. Bit error rate comparison between our three algorithms, quaternary LDPC and the baseline algorithm. Parameter $\lambda$ is shown in (1).

and 3 are better than the baseline binary decoder, and comparable to the quaternary codes. Algorithm 1 has a comparable error rate to the quaternary codes when $\lambda$ is small. Algorithm 3 performs best because it has the most interactions between the two binary LDPC codes.

Moreover, the simulation time of the algorithms are shown in Table I. Our binary codes are 14 times faster on each iteration and 8 times faster on each codeword than quaternary codes. Given the demand of high-volume data storage, the speed-up can improve system performance significantly.

V. CONCLUSION

In conclusion, we present binary LDPC codes for DNA storage to combat asymmetric sequencing errors. Our decoding algorithms utilize the extrinsic information exchange to accommodate the dependency of the errors of the two bits in a nucleotide symbol. We demonstrate the desirable performance of our codes in terms of bit error rate and decoding speed. One possible future work is to investigate error correction codes with soft information from sequencers.

REFERENCES