Coding over Sets for DNA Storage

Andreas Lenz*, **Paul H. Siegel**†, **Antonia Wachter-Zeh***, and **Eitan Yaakobi**‡
*Institute for Communications Engineering, Technical University of Munich, Germany
†Department of Electrical and Computer Engineering, CMRR, University of California, San Diego, California
‡Computer Science Department, Technion – Israel Institute of Technology, Haifa, Israel
Emails: andreas.lenz@mytum.de, psiegel@ucsd.edu, antonia.wachter-zeh@tum.de, yaakobi@cs.technion.ac.il

***Abstract*—In this paper, we study error-correcting codes for the storage of data in synthetic deoxyribonucleic acid (DNA). We investigate a storage model where data is represented by an unordered set of M sequences, each of length L. Errors within that model are losses of whole sequences and point errors inside the sequences, such as substitutions, insertions and deletions. We propose code constructions which can correct these errors with efficient encoders and decoders. By deriving upper bounds on the cardinalities of these codes using sphere packing arguments, we show that many of our codes are close to optimal.**

I. Introduction

DNA-based storage has attracted significant attention due to recent demonstrations of the viability of storing information in macromolecules. This recent increased interest was paved by significant progress in synthesis and sequencing technologies. The main advantages of DNA-based storages over classical storage technologies are very high data densities and long-term reliability without electrical supply. Given the trends in cost decreases of DNA synthesis and sequencing, it is now acknowledged that within the next 10–15 years DNA storage may become a highly competitive archiving technology.

A DNA storage system consists of three important entities (see Fig. 1): (1) a DNA synthesizer that produces the strands that encode the data to be stored in DNA. In order to produce strands with acceptable error rate the length of the strands is typically limited to no more than 250 nucleotides; (2) a storage container with compartments that store the DNA strands, although in an unordered manner; (3) a DNA sequencer that reads the strands and transfers them back to digital data. The encoding and decoding stages are external processes to the storage system which convert the binary user data into strands of DNA in such a way that even in the presence of errors, it is possible to reconstruct the original data.

The first large scale experiments that demonstrated the potential of *in vitro* DNA storage were reported by Church et al. who recovered 643 KB of data [3] and Goldman et al. who accomplished the same task for a 739 KB message [5]. Later, in [6], Grass et al. stored and recovered successfully an 81 KB message by using error-correcting codes. Since then, several groups have built similar systems, storing ever larger amounts of data. Among these, Erlich and Zielinski [4] stored 2.11MB of data with high storage rate, Blawat et al. [1] successfully stored 22MB, and more recently Organick et al. [13] stored 200MB. Yazdi et al. [19] developed a method that offers both random access and rewritable storage.

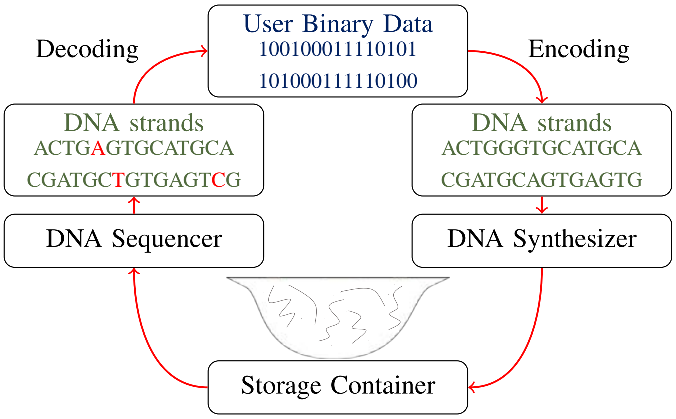

Fig. 1. Illustration of a DNA-based storage system.

DNA as a storage system has several attributes which distinguish it from any other storage system. The most prominent one is that the strands are not ordered in the memory and thus it is not possible to know the order in which they were stored. One way to address this problem is using block addresses, also called indices, that are stored as part of the strand. Errors in DNA are typically substitutions, insertions, and deletions, where most published studies report that either substitutions or deletions are the most common ones, depending upon the specific technology for synthesis and sequencing [1], [4], [9], [13], [14], [20]. While codes correcting substitution errors were widely studied, much less is known for codes correcting insertions and deletions. The task of error correction becomes even more challenging taking into account the lack of ordering of the strands. The goal of this work is to study and to design error-correcting codes which are specifically targeted towards the special structure of DNA storage systems.

II. DNA Channel Model

We consider the storage of data in synthetic DNA and build upon the analysis of [7] and [8]. In such a system, data is stored as an unordered *set*

$$\mathcal{S} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\} \subseteq \mathbb{Z}_2^L,$$

with distinct *sequences* $\mathbf{x}_i$. The parameter M denotes the number of stored sequences and L is the length of each sequence $\mathbf{x}_i$. The set of all possible data sets is therefore

$$\mathcal{X}_M^L = \{\mathcal{S} \subseteq \mathbb{Z}_2^L : |\mathcal{S}| = M\},$$

and note that $|\mathcal{X}_M^L| = \binom{2^L}{M}$. Representing data words as unordered sets is inherently natural, as any information about ordering of the data sequences is lost during the storage.

When a data set $\mathcal{S} \in \mathcal{X}_M^L$ is read from the storage medium, a subset of $M - s$ sequences is obtained, of which additionally t

This work was done in part while A. Lenz and E. Yaakobi were visiting the Center for Memory and Recording Research (CMRR), UC San Diego. This work was supported by the Institute for Advanced Study (IAS), Technische Universität München (TUM), with funds from the German Excellence Initiative and the European Union's Seventh Framework Program (FP7) under grant agreement no. 291763. This work was also supported by NSF Grant CCF-BSF-1619053 and by the United States-Israel BSF grant 2015816.

978-1-5386-4780-6/18/$31.00©2018 IEEE

are erroneous. This received data set $\mathcal{S}'$ is considered to be the output of a pre-processing algorithm, which produces estimates of the stored sequences with a reconstruction algorithm.

Denote by $(\mathcal{U}, \mathcal{L}, \mathcal{F})$ a partition of $\mathcal{S}$ such that:

- the set $\mathcal{U}$ corresponds to the $M - t - s$ sequences that have been received without errors,
- $\mathcal{L}$ is the set of s sequences that have not been read at all,
- $\mathcal{F}$ is the set of t sequences that are read with errors.

Hence, the channel output is $\mathcal{S}' = \mathcal{U} \cup \mathcal{F}'$, where $\mathcal{F}' = \{\mathbf{x}'_{f_1}, \ldots, \mathbf{x}'_{f_t}\}$ is the set of received sequences that are in error. In each erroneous sequence $\mathbf{x}'_{f_i}$, there are at most ϵ substitution or insertion and deletion errors. Note that in contrast to [10], where the storage of *multisets* with full sequence errors ($\epsilon = L$) are discussed, we consider unordered *sets* with point errors ($\epsilon \leq L$). Since the erroneous sequences $\mathbf{x}'_{f_i}$ are not necessarily distinct from each other or from the sequences in $\mathcal{U}$, the size of the received set satisfies $M - t - s \leq |\mathcal{S}'| \leq M - s$. In view of our channel model, we will refer to the following definition of an error-correcting code for a DNA storage system.

Definition 1. *A code $\mathcal{C} \subseteq \mathcal{X}_M^L$ is called an $(s, t, \epsilon)_{\mathcal{H}}$* ***error-correcting code****, if it can correct s (or fewer) losses of sequences and ϵ (or fewer) substitutions in each of t (or fewer) sequences. Similarly, an $(s, t, \epsilon)_{\mathcal{L}}$* ***error-correcting code*** *is defined for insertion/deletion errors, where the erroneous sequences have at most ϵ insertions and deletions.*

Here, the subscripts $\mathcal{H}, \mathcal{L}$ abbreviate the underlying Hamming, respectively Levenshtein metric. With this definition a code $\mathcal{C}$ is a set of codewords, where each codeword is itself a set of M sequences of length L. One of the main challenges associated with errors in such codewords is the loss of ordering of the code sequences. Throughout the paper we will use the following definition for the redundancy of a code.

Definition 2. *The* redundancy *of a code $\mathcal{C} \subseteq \mathcal{X}_M^L$ is*

$$r(\mathcal{C}) = \log |\mathcal{X}_M^L| - \log |\mathcal{C}| = \log \binom{2^L}{M} - \log |\mathcal{C}|.$$

Here and in the rest of the paper, we take the logarithm to the base 2. We summarize several comments on the DNA storage model in the following remark.

Remark 1. 1) *We choose to work here with sets and not multisets of sequences because sequences are assumed to be replicated prior to reading, and the reading process does not necessarily recover all of the copies. Thus it is not possible to distinguish how many times each sequence was stored. For more details, see [7].*

2) *Even though there is no order of the sequences in the set $\mathcal{S} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$, for notational purposes we assume they are listed in lexicographic order in the set representation of $\mathcal{S}$. However, this ordering information is not available when reading the sequences. A common and efficient solution to combat the lack of ordering of the sequences is to add an index for each sequence [19], [7]. This requires an index of $\lceil \log M \rceil$ bits in each sequence, which limits the maximum number of information bits to be $M(L - \lceil \log M \rceil)$. While this solution is attractive for its simplicity, it introduces already a redundancy of*

$$\log \binom{2^L}{M} - M(L - \lceil \log M \rceil) = c_M M,$$

where $c_M = (\lceil \log M \rceil - \log M) + \log \mathrm{e} - \nu$ with $\nu = o(1)$ and $\nu \leq 1 + \log \mathrm{e}$. Hence, every solution which uses indexing already incurs a redundancy of at best roughly $M \log \mathrm{e}$ bits. However, the indexing construction is asymptotically optimal with increasing L [7]. Note that the suboptimality of indexing for multiset codes has been shown in [10].

3) *While the value of L is moderate, e.g., in the order of a few hundreds, M is significantly larger. In this work we assume that $L = o(M)$ and usually M can be polynomial in L or exponential in βL for some $0 < \beta < 1$. In any event we require $M \leq 2^L$.*

4) *We present the results in this work for binary sequences, however most or all of them can be extended to the non-binary case (and, in particular, the quaternary case).*

The results about the redundancy of the proposed constructions and their lower bounds are summarized in Table I.

TABLE I
REDUNDANCY OF CONSTRUCTIONS AND BOUNDS

Error correction	Construction	approx. Bound
$(s, t, L)_{\mathcal{H}}$ or $(s, t, L)_{\mathcal{L}}$	$c_M M + (s + 2t)(L - \lceil \log M \rceil)$	$(s + t)L +$ $t \log M$
	$(s + 2t)L$	
	$M^c(s + 2t)(L - \log M + \log \mathrm{e}) +$ $M^{1-c} \log \left(\mathrm{e} M^{\frac{c}{2}}\right) - (s + 2t) \log \mathrm{e}$	
$(0, 1, 1)_{\mathcal{L}}$	$\log(L + 1)$	$\log(L) - 1$
$(0, 1, 1)_{\mathcal{H}}$	$2L$	$\log L$
$(0, M, 1)_{\mathcal{L}}$	$M \log(L + 1)$	$M(\log L - 1)$
$(0, M, \epsilon)_{\mathcal{H}}$	$M\epsilon \lceil \log L \rceil$	$M\epsilon \log(L/\epsilon)$

III. Code Constructions

A. An Index-Based Construction

The following construction is based on adding an index in front of all sequences $\mathbf{x}_i$ and using a maximum distance separable (MDS) code over the M sequences. For all n and k, where $k \leq n$ we denote by $\mathsf{MDS}[n, k]$ an MDS code over any field of size at least $n - 1$. For all $1 \leq i \leq M$ we will use $\mathbf{I}(i) \in \mathbb{Z}_2^{\lceil \log M \rceil}$ to denote the binary representation of the index i with $\lceil \log M \rceil$ bits.

In Construction 1, the sequences $\mathbf{x}_i = (\mathbf{I}(i), \mathbf{u}_i)$ of each codeword set are constructed by writing a binary representation of the index, $\mathbf{I}(i)$, of length $\lceil \log M \rceil$ in the first part of each sequence. Then, the remaining part $\mathbf{u}_i$ is viewed as a symbol over the extension field $\mathbb{F}_{2^{L - \lceil \log M \rceil}}$, and $(\mathbf{u}_1, \ldots, \mathbf{u}_M)$ will form a codeword in some MDS code[1]. In this construction and in the rest of the paper whenever we write the set $\mathcal{S}$ we assume it is a set of M sequences denoted by $\mathcal{S} = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\} \in \mathcal{X}_M^L$. The following construction is based on the findings in [7], where index-based constructions are analyzed for the correction of losses only.

Construction 1. *For all M, L, and a positive integer δ, let $\mathcal{C}_1(M, L, \delta)$ be the code defined by*

$$\mathcal{C}_1(M, L, \delta) = \{\mathcal{S} \in \mathcal{X}_M^L : \mathbf{x}_i = (\mathbf{I}(i), \mathbf{u}_i), \\ (\mathbf{u}_1, \ldots, \mathbf{u}_M) \in \mathsf{MDS}[M, M - \delta]\}.$$

[1] Note that we assume $M \leq \sqrt{2^L}$ in this section to guarantee the existence of the MDS code. However, the case $M > \sqrt{2^L}$ can always be addressed by employing non-MDS codes.

Lemma 1. *For all M, L, δ, the code $\mathcal{C}_1(M, L, \delta)$ is an $(s, t, L)_{\mathcal{H}}$ error-correcting code for all $s + 2t \leq \delta$.*

Proof. To begin with, we observe that if we can recover the MDS codeword $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_M)$, we can also recover $\mathcal{S}$. Given $\mathcal{S}'$, we create the received word $\mathbf{U}'$ by declaring position j to be an erasure if the index $\mathbf{I}(j)$ does not appear or appears more than once in $\mathcal{S}'$. The remaining positions in $\mathbf{U}'$ are filled with the corresponding symbols $\mathbf{u}'_j$. We will show that the number of erasures s' and the number of errors t' in $\mathbf{U}'$ satisfy $s' + 2t' \leq \delta$. Denote by $\mathcal{U}_I, \mathcal{L}_I, \mathcal{F}_I$ the sets of indices (first $\lceil \log M \rceil$ bits) of sequences in $\mathcal{S}$ corresponding to $\mathcal{U}$, $\mathcal{L}$, or $\mathcal{F}$, respectively. Further, $\mathcal{F}'_I$ is the set of indices of received sequences in $\mathcal{S}'$ that are the erroneous outcomes of the sequences in $\mathcal{F}$. First, we have $s' \leq s + t - t' + |\mathcal{F}'_I \cap \mathcal{U}_I|$ where $|\mathcal{F}'_I \cap \mathcal{U}_I|$ accounts for the situation when an erroneous sequence has the same index as an error-free one. Secondly, the number of errors satisfies $t' \leq |\mathcal{F}'_I \cap (\mathcal{F}_I \cup \mathcal{L}_I)|$. Hence, $s' + 2t' \leq s + t + t' + |\mathcal{F}'_I \cap \mathcal{U}_I| \leq s + 2t \leq \delta$. □

Similarly we obtain the error-correction capability of Construction 1 with respect to insertion and deletion errors.

Lemma 2. *For all M, L, δ, the code $\mathcal{C}_1(M, L, \delta)$ is an $(s, t, L)_{\mathcal{L}}$ error-correcting code for all $s + 2t \leq \delta$.*

Note that for the practically important case of losses and combinations of substitution and deletion errors, $\mathcal{C}_1(M, L, \delta)$ is error-correcting, if $s + 2t_{\mathcal{H}} + t_{\mathcal{D}} \leq \delta$, where $t_{\mathcal{H}}$ is the number of sequences suffering from substitution errors only and $t_{\mathcal{D}}$ is the number of sequences with deletion errors. The same also holds for combinations of substitution and insertion errors. For all M, L, δ, the redundancy of the code $\mathcal{C}_1(M, L, \delta)$ is

$$r(\mathcal{C}_1(M, L, \delta)) = c_M M + \delta(L - \lceil \log M \rceil).$$

B. A Construction Based On Constant Weight Codes

Imposing an ordering (e.g., lexicographic) onto the sequences in $\mathbb{Z}_2^L$, every data set $S \in \mathcal{X}_M^L$ can be represented by a binary vector $\mathbf{v}(\mathcal{S})$ of length 2^L, where each non-zero entry in $\mathbf{v}(\mathcal{S})$ indicates that a specific sequence is contained in the set $\mathcal{S}$. The possible data sets can therefore be represented[2] by the set of constant-weight binary vectors of length 2^L,

$$\mathcal{V}_M^L = \{\mathbf{v} \in \{0,1\}^{2^L} : \mathrm{wt}(\mathbf{v}) = M\},$$

where $\mathrm{wt}(\mathbf{v})$ denotes the *Hamming weight* of $\mathbf{v}$, i.e., the number of non-zero entries inside the vector $\mathbf{v}$. Using this representation, a sequence loss corresponds to an asymmetric $1 \to 0$ error inside $\mathbf{v}(\mathcal{S})$. Errors inside a sequence are either single errors in the Johnson graph, see e.g. [2], or single asymmetric $1 \to 0$ errors, if the erroneous sequence coincides with an already present sequence in $\mathcal{S}$. This suggests the following construction.

Construction 2. *For all M, L and positive integers s, t, let $\mathcal{C}_M^L(s, t) \subseteq \mathcal{V}_M^L$ be an M-constant-weight code of length 2^L, which corrects s asymmetric $1 \to 0$ errors and t errors in the Johnson graph. We then define the following code*

$$\mathcal{C}_2(M, L, s, t) = \{\mathcal{S} \in \mathcal{X}_M^L : \mathbf{v}(\mathcal{S}) \in \mathcal{C}_M^L(s, t)\}.$$

Lemma 3. *For all M, L and positive integers s, t, the code $\mathcal{C}_2(M, L, s, t)$ is an $(s, t, L)_{\mathcal{H}}$ error-correcting code.*

Proof. Denote by $\mathcal{S}'$ the received set after at most s losses of sequences and errors in at most t sequences. Let s' be the number of asymmetric errors and t' be the number of errors in $\mathbf{v}(\mathcal{S})$ with $s' + t' \leq s + t$ and $t' \leq t$. Note that $s' = M - \mathrm{wt}(\mathbf{v}(\mathcal{S}'))$ is detectable by the decoder. If $s' \leq s$, then the decoder can directly decode $s' \leq s$ losses and $t' \leq t$ errors in the Johnson graph. If $s' > s$, the decoder adds $s' - s$ (arbitrarily placed) ones to $\mathbf{v}(\mathcal{S}')$, resulting in exactly s losses and at most $t' + s' - s \leq t$ errors in the Johnson graph. □

Since asymmetric and Johnson graph errors can be represented by one, respectively two substitutions, we can use an M-constant-weight subset of any standard error correcting code, which corrects $\tau = s + 2t$ errors for $\mathcal{C}_M^L(s, t)$. In [15, ch. 5.5] it is shown that a τ-error-correcting binary alternant code code of length 2^L has dimension at least $2^L - \tau L$. Due to the pigeonhole principle, there is one coset of the alternant code that contains at least $\binom{2^L}{M}/2^{\tau L}$ words with constant weight M. Hence, there exists a code $\mathcal{C}_M^L(s, t)$, such that

$$r(\mathcal{C}_2(M, L, s, t)) \leq (s + 2t)L.$$

This redundancy is smaller than the redundancy of Construction 1, especially for the case $L = o(M)$.

C. An Improved Indexed-Based Construction

Construction 1, which uses indexing, is beneficial for its simplicity in the encoding and decoding procedure, however its redundancy is larger than that of Construction 2. On the other hand, Construction 2 does not provide an efficient encoder and decoder. In this section, we present a construction which introduces ideas from both of these methods.

The main idea of this construction is to reduce the number of bits allocated for indexing each sequence. This allows a trade-off in redundancy with respect to L and M. To simplify notation, we assume here that $M = 2^z$ for some $z \in \mathbb{N}$.

Construction 3. *Denote by $\mathbf{I}_c(i) \in \mathbb{Z}_2^{(1-c)\log M}$ the $(1 - c)\log M$ most significant bits of the binary representation $\mathbf{I}(i)$ of i, where $0 \leq c < 1$ and $c \log M \in \mathbb{N}_0$. Further, for $0 \leq i \leq M^{1-c} - 1$, let $\mathbf{U}_i = \{\mathbf{u}_{iM^c+1}, \ldots, \mathbf{u}_{(i+1)M^c}\}$ denote a set of distinct sequences with the same index $\mathbf{I}_c(i)$, which are ordered lexicographically and form a symbol over a field. For $\delta \geq 0$, let $\mathcal{C}_3(M, L, c, \delta)$ be the code defined by*

$$\begin{aligned}\mathcal{C}_3(M, L, c, \delta) = \{\mathcal{S} \in \mathcal{X}_M^L : \mathbf{x}_i = (\mathbf{I}_c(i), \mathbf{u}_i),& \\ (\mathbf{U}_1, \ldots, \mathbf{U}_{M^{1-c}}) \in \mathsf{MDS}[M^{1-c}, M^{1-c} - \delta]\}.&\end{aligned}$$

Note that there are M^{1-c} groups of sequences which use the same index. [3]

Lemma 4. *For all M, L, c, δ, the code $\mathcal{C}_3(M, L, c, \delta)$ is an (s, t, L) error-correcting code for all $s + 2t \leq \delta$.*

Lemma 1 is proven similiar to Lemma 4. The redundancy of $\mathcal{C}_3$ can be shown to be approximately

$$r(\mathcal{C}_3) \approx M^c \delta(L - \log M + \log \mathrm{e}) + M^{1-c} \log\left(\mathrm{e} M^{\frac{c}{2}}\right) - \delta \log \mathrm{e}.$$

[2] This representation has been used as a proof technique in [7].

[3] The symbols of the MDS code are symbols of a finite field with size $\binom{2^L M^{c-1}}{M^c}$ and we therefore require $M^{1-c} \leq \binom{2^L M^{c-1}}{M^c}$.

D. Special Constructions

We begin with an observation about the equivalence of $(0, t, \epsilon)_{\mathcal{L}}$ codes for the case where there are either only insertion or only deletion errors inside the sequences.

Lemma 5. *A code $\mathcal{C} \subseteq \mathcal{X}_M^L$ is $(0, t, \epsilon)$ insertion-only correcting if and only if it is $(0, t, \epsilon)$ deletion-only correcting.*

Note that a $(0, t, \epsilon)_{\mathcal{L}}$ deletion-only (or insertion-only) code, with $\epsilon \geq 2$, is in general not insertion *and* deletion correcting. A counterexample is the code $\mathcal{C} = \{\{0000, 1111, 1000\}, \{0000, 1111, 0111\}\}$, which is both $(0, 1, 2)$ insertion-only and deletion-only correcting, but not $(0, 1, 2)_{\mathcal{L}}$ insertion and deletion correcting.

The following construction is based on Varshamov-Tenengolts (VT) codes [17], [11] that correct a single insertion/deletion in one of the M sequences. This code can be extended to an arbitrary alphabet size q by applying non-binary VT codes [16]. The construction employs the idea of using single-erasure-correcting code over the checksums. The insertion/deletion can then be corrected using the corresponding checksum. Note that this idea is similar to the concept of tensor product codes [18].

Definition 3. *The checksum $s_L(\mathbf{x})$ of $\mathbf{x} \in \mathbb{Z}_2^L$ is defined by*

$$s_L(\mathbf{x}) = \sum_{i=1}^{L} i x_i \bmod (L+1).$$

Construction 4. *For an integer a, with $0 \leq a \leq L$, the code construction $\mathcal{C}_4(M, L, a)$ is given by*

$$\mathcal{C}_4(M, L, a) = \left\{ \mathcal{S} \in \mathcal{X}_M^L : \sum_{i=1}^{M} s_L(\mathbf{x}_i) \equiv a \bmod (L+1) \right\}.$$

Lemma 6. *For all M, L, a, the code $\mathcal{C}_4(M, L, a)$ is a $(0, 1, 1)_{\mathcal{L}}$ error-correcting code.*

Proof. Assume there has been a single insertion or deletion in the k-th sequence. After the reading process, the $M-1$ error-free sequences can be identified as they have length exactly L. The checksum deficiency is given by

$$a - \sum_{i \in \mathcal{U}} s_L(\mathbf{x}_i) \bmod (L+1) = s_L(\mathbf{x}_k).$$

The error in $\mathbf{x}_k$ is corrected by decoding into the VT code with checksum $s_L(\mathbf{x}_k)$. □

Based on the pigeonhole principle there exists $0 \leq a \leq L$ such that the redundancy of the code $\mathcal{C}_4(M, L, a)$ satisfies

$$r(\mathcal{C}_4(M, L, a)) \leq \log(L+1).$$

As we will show in Theorem 2, the redundancy of any $(0, 1, 1)_{\mathcal{L}}$ error-correcting code is at least $\log(L+2) - 1$, and thus Construction 4 is close to optimal.

Using VT codes, we propose another construction of $(0, M, 1)_{\mathcal{L}}$ error-correcting codes. That is, the code can correct a single deletion or insertion in every sequence.

Construction 5. *Let $a \in \mathbb{N}_0$, with $0 \leq a \leq L$. Then,*

$$\mathcal{C}_5(M, L, a) = \{\mathcal{S} \in \mathcal{X}_M^L : s_L(\mathbf{x}_i) \equiv a \bmod (L+1), \forall 1 \leq i \leq M\}.$$

Lemma 7. *The code $\mathcal{C}_5(M, L, a)$ is a $(0, M, 1)_{\mathcal{L}}$ error-correcting code.*

By Construction 5, all sequences $\mathbf{x}_i$ have the same checksum a, which allows to correct single insertions or deletions in each sequence. It is known [11] that the number of words satisfying $s_L(\mathbf{x}) = 0 \bmod (L+1)$ is at least $2^L/(L+1)$. Therefore the redundancy of Construction 5 is at most

$$r(\mathcal{C}_5(M, L, 0)) \leq M \left(\log(L+1) + \frac{M \log \mathrm{e}}{2^L/(L+1) - M} \right).$$

With our assumption $M = 2^{\beta L}$, we obtain a redundancy of $r(\mathcal{C}_5(M, L, 0)) \approx M \log(L+1)$. The next construction can be used to correct ϵ substitution errors in each sequence. Let $\mathcal{C}[L, \epsilon]$ a binary ϵ-error-correcting code of length L.

Construction 6. *For all M, L, and ϵ we define the code*

$$\mathcal{C}_6(M, L, \epsilon) = \{\mathcal{S} \in \mathcal{X}_M^L : \mathcal{S} \subseteq \mathcal{C}[L, \epsilon]\},$$

Lemma 8. *The code $\mathcal{C}_6(M, L, \epsilon)$ is a $(0, M, \epsilon)_{\mathcal{H}}$ error-correcting code.*

The proof is immediate, since every sequence is a codeword of a code that can correct ϵ errors. For $\mathcal{C}[L, \epsilon]$ we use a binary ϵ-error correcting alternant code of length L, which has redundancy at most $\epsilon \lceil \log L \rceil$ [15, ch. 5.5] and thus obtain a code $\mathcal{C}_6(M, L, \epsilon)$ with redundancy at most

$$r(\mathcal{C}_6(M, L, \epsilon)) \leq M \left(\epsilon \lceil \log L \rceil + \frac{M \log \mathrm{e}}{2^{L - \epsilon \lceil \log L \rceil} - M} \right),$$

if $M \leq 2^{L - \epsilon \lceil \log L \rceil}$. With our assumption $M = 2^{\beta L}$, the redundancy is roughly $r(\mathcal{C}_6(M, L, \epsilon)) \approx M \epsilon \lceil \log L \rceil$.

IV. Upper Bounds

In this section we derive non-asymptotic sphere packing upper bounds on codes within the presented storage model.

Definition 4. *The error ball $B_{t,\epsilon}^{\mathcal{H}}(\mathcal{S})$ $[B_{t,\epsilon}^{\mathcal{L}}(\mathcal{S})]$ is defined to be the set of all possible received sets $\mathcal{S}' = \mathcal{U} \cup \mathcal{F}'$ after t (or fewer) sequences of $\mathcal{S} \in \mathcal{X}_M^L$ have been distorted by ϵ (or fewer) substitution [insertion/deletion] errors each.*

Definition 5. *The error ball $B_\epsilon^{\mathcal{H}}(\mathbf{x})$ $[B_\epsilon^{\mathcal{L}}(\mathbf{x})]$ around $\mathbf{x} \in \mathbb{Z}_2^L$ is defined to be the set of all possible received vectors $\mathbf{x}' \neq \mathbf{x}$, after ϵ (or fewer) substitutions [insertions/deletions].*

Theorem 1. *The cardinality of any $(0, t, \epsilon)_{\mathcal{H}}$ error-correcting code $\mathcal{C} \subseteq \mathcal{X}_M^L$ satisfies*

$$|\mathcal{C}| \leq \frac{\sum_{i=M-t}^{M} \binom{2^L}{i}}{(B_\epsilon^{\mathcal{H}} - (t-1) N_\epsilon^{\mathcal{H}})^t},$$

where $B_\epsilon^{\mathcal{H}} = \sum_{i=1}^{\epsilon} \binom{L}{i}$ is the size of the ϵ-error ball and $N_\epsilon^{\mathcal{H}} = \sum_{i=0}^{\epsilon-1} \binom{L-1}{i}$ is the maximum intersection size of two ϵ-error balls around two distinct words.

Proof. We derive a lower bound on $|B_{t,\epsilon}^{\mathcal{H}}(\mathcal{S})|$. To each of the t erroneous sequences we can associate a unique set of at least $B_\epsilon^{\mathcal{H}} - (t-1) N_\epsilon^{\mathcal{H}}$ distinct words in the substitution error ball. This is because there are $B_\epsilon^{\mathcal{H}}$ elements in the substitution ball and there are at most $N_\epsilon^{\mathcal{H}}$ elements in common with each of the $t-1$ other erroneous sequences. Therefore, we get $(B_\epsilon^{\mathcal{H}} - (t-1) N_\epsilon^{\mathcal{H}})^t$ possible unique received sets. The nominator counts all possible received sets of size $M-t$ to M, which yields the bound by a sphere packing argument. The value for $N_\epsilon^{\mathcal{H}}$ is known from [12]. □

Using this bound yields for small t and $\epsilon = 1$ a minimum redundancy of approximately $t\log(L)$.

Theorem 2. *The cardinality of any $(0,t,\epsilon)_{\mathcal{L}}$ error-correcting code $\mathcal{C} \subseteq \mathcal{X}_M^L$ satisfies*

$$|\mathcal{C}| \leq \frac{\binom{2^L}{M-t}\binom{2^{L+\epsilon}}{t}}{\binom{M}{t}(S_\epsilon^{\mathcal{I}} - (t-1)N_\epsilon^{\mathcal{I}})^t},$$

where $S_\epsilon^{\mathcal{I}} = \sum_{i=0}^{\epsilon}\binom{L+\epsilon}{i}$ is the size of the ϵ-insertion sphere and $N_\epsilon^{\mathcal{I}} = \sum_{i=0}^{\epsilon-1}\binom{L+\epsilon}{i}(1-(-1)^{\epsilon-i})$ is the maximum intersection of two ϵ-insertion spheres of two distinct words.

The proof of Theorem 2 follows the same idea as the proof of Theorem 1. For small t and $\epsilon = 1$, this bound implies a minimum redundancy of approximately $t(\log(L+2)-1)$.

A. Asymptotic bounds

We now derive asymptotic bounds for large L on the redundancy for $(0,M,\epsilon)_{\mathcal{H}}$ and $(0,M,\epsilon)_{\mathcal{L}}$ error-correcting codes.

Lemma 9. *Denote by $\mathcal{Y} \subseteq \mathcal{S} \in \mathcal{X}_M^L$ the largest set such that $\mathcal{Y}$ is an ϵ-substitution correcting code. Then,*

$$|B_{t,\epsilon}^{\mathcal{H}}(\mathcal{S})| \geq \begin{cases} \left(B_\epsilon^{\mathcal{H}}\right)^{|\mathcal{Y}|}, & \textit{if} |\mathcal{Y}| \leq t \\ \binom{|\mathcal{Y}|}{t}\left(B_\epsilon^{\mathcal{H}}\right)^t, & \textit{else} \end{cases}$$

where $B_\epsilon^{\mathcal{H}} = \sum_{i=1}^{\epsilon}\binom{L}{i}$.

Proof. In each of the distinct error balls $B_\epsilon^{\mathcal{H}}(\mathbf{x})$, $\mathbf{x} \in \mathcal{Y}$ we have at least $B_\epsilon^{\mathcal{H}} = |B_\epsilon^{\mathcal{H}}(\mathbf{x})|$ possible patterns of unique outcomes for $B_{t,\epsilon}^{\mathcal{H}}(\mathcal{S})$ by either adding an error to $\mathbf{x}$ such that a sequence in $B_\epsilon^{\mathcal{H}}(\mathbf{x}) \setminus \mathcal{S}$ is obtained or by adding an error to a sequence in $B_\epsilon^{\mathcal{H}}(\mathbf{x}) \cap \mathcal{S}$ such that $\mathbf{x}$ is obtained. □

Theorem 3. *The redundancy of any $(0,M,\epsilon)_{\mathcal{H}}$ error-correcting code $\mathcal{C} \subseteq \mathcal{X}_M^L$ satisfies asymptotically*

$$r(\mathcal{C}) \gtrsim cM\log(B_\epsilon^{\mathcal{H}}),$$

for any $0 \leq c < 1$, when $L \to \infty$ and $M = 2^{\beta L}$, $0 < \beta < 1$.

Proof. Denote by $D(c)$ the number of data words $\mathcal{S} \in \mathcal{X}_M^L$ which have a ball size $|B_{M,\epsilon}^{\mathcal{H}}(\mathcal{S})| < (B_\epsilon^{\mathcal{H}})^{cM}$, where $0 \leq c < 1$. By Lemma 9, $D(c)$ is at most the number of data sets, which do not contain an ϵ-error correcting code $\mathcal{Y} \subseteq \mathcal{S}$ of size at least ct. By a sphere packing argument, it follows that any $(0,M,\epsilon)_{\mathcal{H}}$ correcting code $\mathcal{C} \subseteq \mathcal{X}_M^L$ satisfies

$$|\mathcal{C}| \leq \frac{\sum_{i=cM}^{M}\binom{2^L}{i}}{(B_\epsilon^{\mathcal{H}})^{cM}} + D(c).$$

It can be shown that the first term in this sum dominates the bound for all $0 \leq c < 1$, when $M = 2^{\beta L}$, with $0 < \beta < 1$. □

Theorem 4. *The redundancy of any $(0,M,\epsilon)_{\mathcal{L}}$ error-correcting code $\mathcal{C} \subseteq \mathcal{X}_M^L$ satisfies asymptotically*

$$r(\mathcal{C}) \gtrsim cM(\log(S_\epsilon^{\mathcal{I}}) - \epsilon),$$

for any $0 \leq c < 1$, when $L \to \infty$ and $M = 2^{\beta L}$, $0 < \beta < 1$.

Theorem 4 can be shown by noting that most balls $B_{M,\epsilon}^{\mathcal{L}}(\mathcal{S})$ have size at least $(S_\epsilon^{\mathcal{I}})^{cM}$, similar to the proof of Theorem 3.

B. Bound for losses and errors

Theorem 5. *The redundancy of any $(s,t,L)_{\mathcal{H}}$ or $(s,t,L)_{\mathcal{L}}$ correcting code $\mathcal{C} \subseteq \mathcal{X}_M^L$ satisfies*

$$r(\mathcal{C}) \geq (s+t)\log(2^L-M-t)+t\log(M-s-t)-\log(t!(s+t)!).$$

Proof. Choosing $s+t$ sequences to be erroneous and letting each of the t erroneous ones be one of the $2^L - M$ sequences in $\mathcal{X}_M^L \setminus \mathcal{S}$, we can use a sphere-packing argument to show that any $(s,t,L)_{\mathcal{H}}$ or $(s,t,L)_{\mathcal{L}}$ correcting code $\mathcal{C} \subseteq \mathcal{X}_M^L$ satisfies $|\mathcal{C}| \leq \binom{2^L}{M-s}/(\binom{M}{t+s}\binom{2^L-M}{t})$. □

References

[1] M. Blawat, K. Gaedke, I. Hütter, X. M. Chen, B. Turczyk, S. Inverso, B. W. Pruitt, and G. M. Church, "Forward error correction for DNA data storage," in *Int. Conf. Computational Science*, San Diego, Jun. 2016, pp. 1011–1022.

[2] A. E. Brouwer, J. B. Shearer, N. J. A. Sloane, and W. D. Smith, "A new table of constant weight codes," *IEEE Trans. Inf. Theory*, vol. 36, no. 6, pp. 1334–1380, Nov. 1990.

[3] G. M. Church, Y. Gao, and S. Kosuri, "Next-generation digital information storage in DNA," *Science*, no. 6102, pp. 1628–1628, Sep. 2012.

[4] Y. Erlich and D. Zielinski, "DNA fountain enables a robust and efficient storage architecture," *Science*, no. 6328, pp. 950–954, Mar. 2017.

[5] N. Goldman, P. Bertone, S. Chen, C. Dessimoz, E. M. LeProust, B. Sipos, and E. Birney, "Towards practical, high-capacity, low-maintenance information storage in synthesized DNA," *Nature*, no. 7435, pp. 77–80, Jan. 2013.

[6] R. N. Grass, R. Heckel, M. Puddu, D. Paunescu, and W. J. Stark, "Robust chemical preservation of digital information on DNA in silica with error-correcting codes," *Angewandte Chemie Int. Edition*, no. 8, pp. 2552–2555, Feb. 2015.

[7] R. Heckel, I. Shomorony, K. Ramchandran, and D. N. C. Tse, "Fundamental limits of DNA storage systems," in *IEEE Int. Symp. Inform. Theory*, Aachen, Germany, Jun. 2017, pp. 3130–3134.

[8] H. M. Kiah, G. J. Puleo, and O. Milenkovic, "Codes for DNA sequence profiles," *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 3125–3146, Jun. 2016.

[9] S. Kosuri and G. Church, "Large-scale de novo DNA synthesis: technologies and applications," *Nature Methods*, no. 5, pp. 499–507, May 2014.

[10] M. Kovačević and V. Y. F. Tan, "Codes in the space of multisets – coding for permutation channels with impairments," *IEEE Trans. Inf. Theory*, Jan. 2018, (early access).

[11] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Soviet Physics Doklady*, vol. 10, no. 8, pp. 707–710, Feb. 1966.

[12] ——, "Efficient reconstruction of sequences," *IEEE Trans. Inf. Theory*, vol. 47, no. 1, pp. 2–22, Jan. 2001.

[13] L. Organick, S. D. Ang, Y. J. Chen, R. Lopez, S. Yekhanin, K. Makarychev, M. Z. Racz, G. Kamath, P. Gopalan, B. Nguyen, C. Takahashi, S. Newman, H. Y. Parker, C. Rashtchian, G. G. K. Stewart, R. Carlson, J. Mulligan, D. Carmean, G. Seelig, L. Ceze, , and K. Strauss, "Scaling up DNA data storage and random access retrieval," *bioRxiv*, Mar. 2017.

[14] M. G. Ross, C. Russ, M. Costello, A. Hollinger, N. Lennon, R. Hegarty, N. Nusbaum, and D. Jaffe, "Characterizing and measuring bias in sequence data," *Genome Biol.*, no. 5, May 2013.

[15] R. M. Roth, *Introduction to Coding Theory*. New York: Cambridge University Press, 2006.

[16] G. M. Tenengolts, "Nonbinary codes, correcting single deletion or insertion," *IEEE Trans. Inf. Theory*, vol. 30, no. 5, pp. 766–769, 1984.

[17] R. R. Varshamov and G. M. Tenengolts, "Codes which correct single asymmetric errors," *Automation Remote Control*, vol. 26, no. 2, pp. 286–290, 1965.

[18] J. K. Wolf, "An introduction to tensor product codes and applications to digital storage systems," in *IEEE Inform. Theory Workshop*, Chengdu, China, Oct. 2006, pp. 6–10.

[19] S. M. H. T. Yazdi, Y. Yuan, J. Ma, H. Zhao, and O. Milenkovic, "A rewritable, random-access DNA-based storage system," *Nature Scientific Reports*, no. 14138, Aug. 2015.

[20] A. K. Yim, A. C. S. Yu, J. W. Li, A. I. C. Wong, J. F. C. Loo, K. Chan, S. K. Kong, and T. F. Chan, "The essential component in DNA-based information storage system: Robust error-tolerating module," *Frontiers in Bioengineering and Biotechnology*, no. 49, pp. 1–5, Nov. 2014.