Single-Error Detection and Correction for Duplication and Substitution Channels

Yuanyuan Tang*, Yonatan Yehezkeally†, Moshe Schwartz‡, and Farzad Farnoud (Hassanzadeh)§
*Electrical & Computer Engineering, University of Virginia, yt5tz@virginia.edu
†Electrical & Computer Engineering, Ben-Gurion University of the Negev, yonatany@post.bgu.ac.il
‡Electrical & Computer Engineering, Ben-Gurion University of the Negev, schwartz@ee.bgu.ac.il
§Electrical & Computer Engineering, University of Virginia, farzad@virginia.edu

***Abstract*—Motivated by mutation processes occurring in in-vivo DNA-storage applications, a channel that mutates stored strings by duplicating substrings as well as substituting symbols is studied. Two models of such a channel are considered: one in which the substitutions occur only within the duplicated substrings, and one in which the location of substitutions is unrestricted. Both error-detecting and error-correcting codes are constructed, which can handle correctly any number of tandem duplications of a fixed length k, and at most a single substitution occurring at any time during the mutation process.**

I. Introduction

Recent advances in DNA sequencing and synthesis technologies have increased the potential of DNA as a data storage medium. In addition to its high data density, data storage in DNA provides a long-lasting alternative to current storage media. Furthermore, given the need for accessing biological data stored in DNA of living organisms, technologies for retrieving data from DNA will not become obsolete, unlike flash memory, magnetic disks, and optical disks. Data can be stored in DNA *in vitro* or *in vivo*. While the former will likely provide a higher density, the latter can provide a more reliable and cost-effective replication method, as well as a protective shell [11]. In-vivo storage also has applications such as watermarking genetically modified organisms. This technology was recently demonstrated experimentally using CRISPR/Cas gene editing [10], [11]. One of the challenges of this technology is that a diverse set of errors are possible, including substitutions, duplications, insertions, and deletions. Duplication errors, in particular, have been previously studied by a number of recent works, including [3]–[6], [9], among others. This paper focuses on error-control codes for duplication and substitution errors.

In a (tandem) duplication event, a substring of the DNA sequence, the *template*, is duplicated and the resulting *copy* is inserted into the sequence next to the template [12]. Evidence of this process is found in the genomes of many organisms as patterns that are repeated multiple times [2]. In a substitution event, a symbol in the sequence is changed to another symbol of the alphabet. It has been observed that point mutations such as substitutions are more common in tandem repeat regions of the genomes [7]. We consider two models for combined duplication and substitution errors. In the first model, called the *noisy-duplication model*, the copy is a noisy version of the template. Noisy duplications in this model can be viewed as exact duplications followed by substitutions that are restricted to the newly added copy. We also consider an *unrestricted-substitution model*, which relaxes the noisy duplication model by allowing substitutions at any position in the sequence.

In this paper we construct both error-detecting and error-correcting codes, that are capable of correctly handling any number of tandem duplications of a fixed length k, and at most a single substitution error. The main approach in both is to reverse the duplication process while accounting for the single substitution (which may spuriously create the appearance of a duplication that never happened, or eliminate one that did). Different challenges are also presented by the possible locations for substitutions. Namely, we study both the case where the substitution is restricted to occur within a duplicated substring, as well as the case of unrestricted location for the substitution. We bring these difference to light by providing a construction for an error-detecting code for the restricted substitution model, and an error-correcting code for the unrestricted substitution model.

This paper is organized as follows. In Section II we provide the notation as well as relevant background and known results. In Section III we construct error-detecting codes for the restricted substitution model. Finally, in Section IV we give a construction for an error-correcting code for the unrestricted substitution model. Most proofs and technical lemmas are omitted or sketched due to the page limit, and will be available in the full version of this paper.

II. Notation and Preliminaries

Throughout the paper, we assume that the alphabet Σ is a unital ring of size $q \geqslant 2$ (e.g., $\mathbb{Z}_q$ or, when q is a prime power, $\mathbb{F}_q$). Thus, addition (or subtraction) and multiplications of letters from the alphabet are well-defined. The set of finite strings and strings of length at least k over Σ is denoted Σ^* and $\Sigma^{\geqslant k}$, respectively. The concatenation of two strings, $u, v \in \Sigma^*$ is denoted by uv, and u^k denotes concatenating k copies of u. The length (number of letters) of u is denoted by $|u|$, and its Hamming weight by $\mathrm{wt}(u)$. We say $y \in \Sigma^*$ is a substring of $w \in \Sigma^*$ if there exist $x, z \in \Sigma^*$ such that $w = xyz$.

A (tandem) *duplication* of length k duplicates a substring of length k and inserts it in tandem into the string, namely, the copy immediately follows the template. For example, from

This work was supported in part by NSF grants under grant nos. 1816409 and 1755773, and a BSF grant under grant no. 2017652.

uvw, where $|v| = k$, we may obtain $uvvw$. As an example for $k = 3$ and alphabet $\Sigma = \mathbb{Z}_3$, consider

$$x = 1012121 \to x' = 1012\underline{012}121, \tag{1}$$

where the underline part is the copy.

The analysis of duplication errors will be facilitated by the *k-discrete-derivative* transform, defined in [1] in the following way. For $x \in \Sigma^{\geqslant k}$, we define $\phi(x) \triangleq \hat{\phi}(x)\bar{\phi}(x)$, where

$$\hat{\phi}(x) \triangleq x_1 \cdots x_k, \quad \bar{\phi}(x) \triangleq x_{k+1} \cdots x_n - x_1 \cdots x_{n-k},$$

in which subtraction is performed entry-wise over Σ. We note that $\phi(\cdot)$ is a bijection. The duplication length k is implicit in the definition of ϕ. For a set of strings S, we define $\phi(S) \triangleq \{\phi(s) \mid s \in S\}$.

Let x' be obtained through a tandem duplication of length k from x. It is not difficult to see that $\hat{\phi}(x) = \hat{\phi}(x')$ and that $\bar{\phi}(x')$ can be obtained from $\bar{\phi}(x)$ by inserting 0^k in an appropriate position [3]. For the example given in (1),

$$x = 1012121 \to x' = 1012\underline{012}121$$
$$\phi(x) = 101, 1112 \to \phi(x') = 101, 1\underline{000}112$$

Here, a comma separates the parts of ϕ for clarity.

Sometimes duplications are *noisy* and the duplicated symbols are different from the original symbols. (Unless otherwise stated duplications are assumed to be exact.) We only consider the case where a single symbol is different. We view a noisy duplication as a duplication followed by a substitution in the duplicated substring. Continuing the example, the duplication resulting in x' may be followed by a substitution,

$$x' = 1012\underline{0}12121 \to x'' = 1012\underline{1}12121,$$
$$\phi(x') = 101, 1\underline{000}112 \to \phi(x'') = 101, 1\underline{100}012.$$

We also consider unrestricted substitutions, which can occur at any position in the string, rather than only in a substring that is duplicated by the previous duplication. A substitution may be considered as the mapping $x \to x + ae_i$, where $e_i \in \Sigma^n$ is a standard unit vector at index i, and $a \in \Sigma$, $a \neq 0$. Since ϕ is linear over Σ (i.e., $\phi(x+ae_i) = \phi(x)+a\phi(e_i)$), we denote the transform of e_i as $\epsilon_i \triangleq \phi(e_i)$, and observe that $\epsilon_i = e_i - e_{i+k}$ for $i \leqslant n-k$ and $\epsilon_i = e_i$ for $n-k < i \leqslant n$. We note that substitutions might affect two positions in the transform domain. When the length n is unclear from the context, we shall indicate it in the superscript as $e_i^{(n)}$ and $\epsilon_i^{(n)}$.

Let $D_k^{t(p)}(x)$ (for $t \geqslant p$) denote the set of strings that can be obtained from x through t tandem duplications, p of which are noisy, with each noisy duplication containing a single substitution. $D_k^{t(p)}$ is called a *descendant cone* of x. We further define

$$D_k^{*(p)}(x) \triangleq \bigcup_{t=p}^{\infty} D_k^{t(p)}(x), \quad D_k^{*(P)}(x) \triangleq \bigcup_{p \in P} D_k^{*(p)}(x), \tag{2}$$

where P is a subset of non-negative integers. We denote $P = \{0, 1\}$ as $\leqslant 1$.

We define $D_k^{t,p}(x)$ to be the set of strings obtained from x through t tandem duplications and p substitutions, where substitutions can occur in any position (and so we do not require $t \geqslant p$). We extend this definition similar to (2).

For a string $z \in \Sigma^*$, $\mu(z)$ is obtained by removing all copies of 0^k from z. Specifically, for

$$z = 0^{m_0} w_1 0^{m_1} w_2 \cdots w_d 0^{m_d},$$

where m_i are non-negative integers and $w_i \in \Sigma \setminus \{0\}$ are nonzero symbols, we define

$$\mu(z) \triangleq 0^{m_0 \bmod k} w_1 0^{m_1 \bmod k} w_2 \cdots w_d 0^{m_d \bmod k}.$$

Define the *duplication root* $\mathrm{drt}(x)$ of x as the unique string obtained from x by removing all tandem repeats of length k. Note that $\phi(\mathrm{drt}(x)) = \hat{\phi}(x)\mu(\bar{\phi}(x))$ (see [3]). For a set of strings S, we define $\mathrm{drt}(S) \triangleq \{\mathrm{drt}(s) \mid s \in S\}$.

A string x is *irreducible* if $x = \mathrm{drt}(x)$. The set of irreducible strings of length n is denoted $\mathrm{Irr}(n)$, where the duplication length k is again implicit. We denote by $\mathrm{RLL}(m)$ the set of strings in Σ^m that do not contain 0^k as a substring. A string x of length n is irreducible if and only if $\bar{\phi}(x) \in \mathrm{RLL}(n-k)$.

Finally, we define the *redundancy* of a code $C \subseteq \Sigma^n$ as

$$r(C) \triangleq n - \log_q |C| = n - \log_{|\Sigma|} |C|,$$

and the code's *rate* as $R(C) \triangleq 1 - \frac{r(C)}{n}$.

III. Restricted Error-Detecting Codes

In this section, we consider the case of noisy-duplication errors. Our goal is to correct errors consisting of any number of exact duplications and detect errors consisting of any number of exact duplications and one noisy duplication, where the noisy duplication contains one substitution. We refer to codes with this capability as *1-noisy duplication (1ND)* detecting.

A code $C \subseteq \Sigma^n$ is 1ND-detecting if and only if for any two distinct codewords $c_1, c_2 \in C$, we have

$$D_k^{*(\leqslant 1)}(c_1) \cap D_k^{*(0)}(c_2) = \varnothing.$$

It can be shown that this is equivalent to

$$\mathrm{drt}(c_2) \neq \mathrm{drt}(c_1), \quad \mathrm{drt}(c_2) \notin \mathrm{drt}(D_k^{*(1)}(c_1)). \tag{3}$$

As a result of the substitution in the noisy duplication, the length of the duplication root may change. One way to simplify the code design is to restrict ourselves to codes whose codewords all have duplication roots with the same length. Then error patterns that modify this length can be easily detected and we can focus on patterns that keep the duplication-root length the same. We thus consider codes of length n whose codewords are irreducible.

For an irreducible string x of length $n > 2k$, our goal is to study the set $\mathrm{drt}(D_k^{*(1)}(x)) \cap \Sigma^n$, corresponding to errors that do not change the length of the root of x. We refer to a substitution error that does not change the length of the root as an *ambiguous substitution*.

Given that $\hat{\phi}(x)$ is not altered by duplications or substitution errors embedded in noisy duplications, we only study $z = \bar{\phi}(x)$. Let $\bar{z}$ be obtained from z by an arbitrary number of insertions of 0^k (equivalent to k-duplications) and have the form

$$\bar{z} = u0^{pk+m} b_1 b_2 \cdots b_k v,$$

where $m, p \in \mathbb{N}$ such that $0 \leqslant m < k$; $u, v \in \Sigma^*$ such that the last element of u is nonzero; and $b_1, \ldots, b_k$ are symbols such that $b_1 \neq 0$. Assume that the next duplication is noisy and thus includes a substitution. Without loss of generality, this duplication inserts 0^k between u and b_1, resulting in z',

$$z' = u0^{(p+1)k+m} b_1 b_2 \cdots b_k v.$$

Since 0^k may be inserted at any position among the existing 0s before b_1, a substitution can take place at any position among the resulting $m + (p+1)k$ 0s.

An ambiguous substitution can only happen among the last k 0s before b_1 in z', since otherwise the length of root before b_1 will increase while the length after b_1 will stay the same. After the substitution, we obtain

$$z'' = u0^{pk} 0^{m+i-1} b 0^{k-i} b_1 \cdots b_{i-1} (b_i - b) b_{i+1} \cdots b_k v, \quad (4)$$

where $1 \leqslant i \leqslant k$ and $b \neq 0$. Let the length of the run of 0s on the left side of b_i be m_1 and the length of the run of 0s on the right of b_i be m_2 (excluding b_i). Note that $0 \leqslant m_1 \leqslant k-2$. It can be shown that an ambiguous substitution can happen only in the following two cases:

- C.1 $1 < i \leqslant k - m$, $b_i = b$, and $\lfloor \frac{m_2}{k} \rfloor < \lfloor \frac{m_1+m_2+1}{k} \rfloor$. In this case, the length of the root before b_1 increases by k and the length of the root after b_1 decreases by k.
- C.2 $k - m < i \leqslant k$ and ($b_i \notin \{0, b\}$ or $\lfloor \frac{m_2}{k} \rfloor = \lfloor \frac{m_1+m_2+1}{k} \rfloor$). Then the length of the root before and after b_1 remain constant.

Further duplications do not affect the duplication-root. Hence, we use the analysis above to find lower bounds on the size of 1ND-detecting codes. For a string x in the space of irreducible strings of length n, let $V(x, 1)$ denote the size of the sphere of radius 1 substitution and many duplications centered at x. That is,

$$V(x, 1) \triangleq |\mathrm{drt}(D_k^{*(\leqslant 1)}(x)) \cap \Sigma^n|.$$

Lemma 1 For $x \in \mathrm{Irr}(n)$, where $n > 2k$,

$$V(x, 1) \leqslant (n-k)(q-1) - \mathrm{wt}\big(\bar{\phi}(x)\big)(q-2).$$

Proof (sketch): Only the errors described in C.1 and C.2 above generate descendants with roots of length n. We enumerate the contributions of these cases. It can be shown that C.1 contributes at most $\mathrm{wt}(\bar{\phi}(x)) - 1$ and C.2 contributes at most $(n - k - \mathrm{wt}(\bar{\phi}(x)))(q-1)$ to $V(x, 1)$. ∎

To find a lower bound on the size of the code, we apply the Gilbert-Varshamov (GV) bound with the average size of the sphere (see, e.g., [8]).

Lemma 2 Let x be a randomly chosen irreducible string of length n. If $n > 2k$, then $\mathbb{E}[V(x, 1)] \leqslant 2(n-k)(q-1)/q$.

The above lemma leads to the lower bound in the following theorem. The upper bound follows from the fact that the code must be able to correct any number of duplication errors and from [3] where such codes are discussed.

Theorem 3 For $n > 2k$ and $q + k \geqslant 4$, the maximum size $A_{1\mathrm{ND}}(n, q, k)$ of a 1ND-detecting codes of length n over $\mathbb{Z}_q$ satisfies

$$\frac{M}{4(n-k)} \leqslant A_{1\mathrm{ND}}(n, q, k) \leqslant M,$$

where $M \triangleq \sum_{i=0}^{\lfloor n/k \rfloor - 1} |\mathrm{Irr}(n - ik)| = \sum_{i=1}^{\lfloor n/k \rfloor} q^k |\mathrm{RLL}(n - ik)|$ is the number of irreducible words whose descendant cones intersect Σ^n.

We now turn to construct 1ND-detecting codes. As before, we consider codes that consist of irreducible strings of length n. We thus need to devise a method to detect ambiguous substitutions.

When $k = 1$, it can be shown that ambiguous substitutions cannot occur. So $\mathrm{Irr}(n)$ is a 1ND-detecting code. For $k > 1$, two types of ambiguous substitutions are possible. Suppose an irreducible string $x \in \Sigma_q^n$, with $z = \bar{\phi}(x)$, is stored and assume the retrieved string, after a substitution error, is x'', with $z'' = \bar{\phi}(x'')$. With i, b_1, m, m_1, b defined as in (4), it can be shown that in Case C.1, we have

$$\begin{aligned} \mu(z) &= t\, \underline{b_1 \cdots b_{i-1-m_1}} 0^{m_1}\, b_i\, 0^{k-i}\ \ \underline{0^{i-1-m_1}}\ \ w \\ \mu(z'') &= t\ \ \underline{0^{i-1-m_1}}\ 0^{m_1}\, b_i\, 0^{k-i}\, \underline{b_1 \cdots b_{i-1-m_1}}\, w \end{aligned} \quad (5)$$

and in Case C.2,

$$\begin{aligned} \mu(z) &= t\ 0^{m+i-1-k} \underline{0} 0^{k-i}\ b_1 \cdots b_{i-1}\ \ \underline{b_i}\ \ \ w \\ \mu(z'') &= t\ 0^{m+i-1-k} \underline{b} 0^{k-i}\ b_1 \cdots b_{i-1} \underline{(b_i - b)}\ w \end{aligned} \quad (6)$$

for some $t, w \in \Sigma^*$. The differences between the stored and retrieved strings are marked. We make several observations that will help us in designing codes. First, in Case C.1, substrings $b_1 \cdots b_{i-1-m_1}$ and 0^{i-1-m_1}, which are at distance k, are swapped. As a result, the number of 0s before b_i and after b_i increase and decrease, respectively, by δ, where δ is the number of 0s in $b_1 \cdots b_{i-1-m_1}$. Furthermore, the affected region has length $k + i - 1 - m_1 < 2k$. In Case C.2, a 0 changes to a nonzero element b, and a symbol b_i, which is k positions after b, changes to $b_i - b$. As a result, the number of 0s before b_1 decreases by 1 and after b_1 it increases by at most 1.

These observations indicate that the effect of errors can be observed in the position and the number of 0s in the string, and thus motivate the following definition.

Definition 4 Given a string $x \in \Sigma_q^*$ and $z = \bar{\phi}(x)$, we divide $\mu(z)$ into blocks of length k (excepting, perhaps, the last), numbered as $j = 0, 1, 2, \ldots, \left\lceil \frac{|\mu(z)|}{k} \right\rceil$. For $0 \leqslant i \leqslant 3$, let $Z_i = Z_i(x)$ be the total number of 0s in blocks with index j such that $j \equiv i \bmod 4$.

Figure 1 illustrates the preceding definition. As an example, for $k = 2$ and $\mu(z) = 012034050$, we have $Z_0 = 2$, $Z_1 = 1$, $Z_2 = 0$, and $Z_3 = 1$.

Construction A Let p be the smallest odd integer larger than $k - 1$. For $0 \leqslant i, j < p$, define

$$C_{i,j} = \big\{ x \in \mathrm{Irr}(n) \ \big|\ Z_0(x) + 2Z_2(x) \bmod p = i,\ Z_1(x) + 2Z_3(x) \bmod p = j \big\}.$$

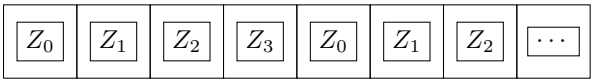

Figure 1. The number of 0's in the jth block of length k contributes to Z_i where $i = (j \bmod 4)$.

Theorem 5 Let q be the alphabet size, k the duplication length, and p the smallest odd integer larger than $k-1$. For $0 \leqslant i, j < p$, the code $C_{i,j}$ of Construction A is a 1ND-detecting code. If $q + k \geqslant 4$, there exist i, j such that $|C_{i,j}| \geqslant \frac{M}{2(k+1)^2}$, where M is given in Theorem 3.

Note that the lower bound on code size given in this theorem may be tighter than the one given in Theorem 3.

Proof: If $k = 1$, then $C_{0,0} = \mathrm{Irr}(n)$ is the only code and the theorem is immediate.

Assume $k > 1$. Let $x, c \in C_{i,j}$ be distinct. By definition of $C_{i,j}$, $\mathrm{drt}(x) = x$ and $\mathrm{drt}(c) = c$. To prove the error detection capability of the code, based on (3), it suffices to show that for any $x'' \in D_k^{*(1)}(x)$, we have $c \neq \mathrm{drt}(x'')$. If $\mathrm{drt}(x'') = \mathrm{drt}(x) = x$, then clearly $c \neq \mathrm{drt}(x'')$. So we assume $\mathrm{drt}(x'') \neq x$. It is then sufficient to show that $\mathrm{drt}(x'') \notin C_{i,j}$. This is obvious if $|\mathrm{drt}(x'')| \neq n$. We thus only consider $|\mathrm{drt}(x'')| = n$.

For $0 \leqslant i \leqslant 3$, let $\Delta_i = Z_i(x'') - Z_i(x)$. Furthermore, for $0 \leqslant i \leqslant 1$, let $F_i = \Delta_i + 2\Delta_{i+2}$. To prove $\mathrm{drt}(x'') \notin C_{i,j}$, it is sufficient to show

$$F_0 \bmod p \neq 0 \text{ or } F_1 \bmod p \neq 0. \tag{7}$$

Let $z = \bar{\phi}(x)$ and $z'' = \bar{\phi}(x'')$. Based on (5) and (6) and the discussion that follows them, we consider three different cases for F_0 and F_1.

First, if the number of zeros changes in 2 consecutive blocks, then one of the pairs (Δ_0, Δ_1), (Δ_1, Δ_2), (Δ_2, Δ_3), (Δ_3, Δ_0) equals $(\delta, -\delta)$ or $(\delta, 0)$ for $0 < |\delta| < k$, and the other Δ_i are equal to 0. Then, $|F_0| = |\delta|$ or $|F_0| = 2|\delta|$. In the former case $F_0 \bmod p \neq 0$ since $0 < |\delta| < k \leqslant p$. In the latter case, $F_0 \bmod p \neq 0$ since $0 < 2|\delta| < 2p$ and $2\delta \neq p$ (recall that p is odd).

Second, if the number of zeros changes in two non-consecutive blocks, then only one of the pairs (Δ_0, Δ_2) and (Δ_1, Δ_3) equals $(\delta, -\delta)$ for $0 < |\delta| < k$, and the other equals $(0, 0)$. Then, either $|F_0| = |\delta|$ or $|F_1| = |\delta|$, and in both cases (7) is satisfied.

Third, if the change of number of zeros occurs in three consecutive blocks, then there exists i such that $\Delta_i = \delta' \neq 0$ and $\Delta_{2+i} = 0$, where $0 < |\delta'| < k$ and $2|\delta'| \neq p$. Then either F_0 or F_1 takes on the value of δ' or $2\delta'$. But $\delta' \bmod p \neq 0$ and $2\delta' \bmod p \neq 0$, implying that (7) is satisfied.

The size of the code is at least $\frac{|\mathrm{Irr}(n)|}{p^2}$, which follows from the pigeonhole principle and the fact that there are p^2 possible values for (i, j). It can be shown that $|\mathrm{Irr}(n)| \geqslant M/2$ if $q + k \geqslant 4$. The asymptotic form is derived immediately from the asymptotics of M (see [3]), which completes the proof. ■

IV. Unrestricted Error-Correcting Codes

Substitution mutations might also occur independently of duplications. In what follows, we consider a single substitution error occurring in addition to however many duplications, but not necessarily in a duplicated substring. We refer to codes able to correct such errors as a *single-substitution correcting* (1S-correcting) code. A code C is 1S-correcting if and only if for any two distinct codewords $c_1, c_2 \in C$, we have

$$D_k^{*,\leqslant 1}(c_1) \cap D_k^{*,\leqslant 1}(c_2) = \varnothing.$$

In this context, we will find it easier to consider strings in the transform domain. We also define the *substitution distance* $\sigma(u, v)$ to measure the number of substitutions required to transform one string into the other, when u, v are assumed to be in the transform domain. More precisely, if $u, v \in \Sigma^n$ and $v - u = \sum_{i=1}^n a_i e_i$, then $\sigma(u, v) \triangleq |\{i \in [n] \mid a_i \neq 0\}|$.

A. Error-correcting codes

When considering the combination of duplications with even a single substitution in the transform domain, we come across the following example:

Example 6 Set $\Sigma = \mathbb{Z}_2$ and $k = 3$, and observe the following two strings of duplication and substitution:

$$u \triangleq 111010111 \to 111010111\underline{000} \to 111\underline{0}00\underline{1}0\underline{1}000$$
$$v \triangleq 111101010 \to 111\underline{000}101010 \to 111000101\underline{0}\underline{0}0$$

It is clear that if $C \subseteq \Sigma^{\geqslant k}$ is a code correcting even a single duplication and a single substitution, even given the order in which they occur, then u, v (or rather, the strings $111101010, 111010000$ whose 3-discrete-derivatives are u, v) cannot both belong to C. Observing that $u, v \in \mathrm{RLL}(9)$ and $\sigma(u, v) = 4$, however, we find that $C \triangleq \{\phi^{-1}(u), \phi^{-1}(v)\}$ can correct any number of duplications, or correct a single substitution. Why it cannot do both at once, then, is not immediately apparent. □

In what follows, we propose a constrained-coding approach which resolves the issue demonstrated in the last example. It relies on the following observation: substitution noise might create a 0^k substring in the transform domain–that is not due to a duplication–as well as break a run of zeros that is; However, a constrained system exists which allow us to de-couple the effects of duplication and substitution noise.

More precisely, we denote

$$\mathcal{W} = \left\{u \in \Sigma^{\geqslant k} \;\middle|\; \forall \text{ substring } v \text{ of } u, |v| = k : \mathrm{wt}(v) > 1\right\}.$$

We aim to show that restricting codewords to be taken from $\mathcal{W}$ (in the trasform domain), the following holds.

Lemma 7 Take an irreducible $x \in \Sigma^{\geqslant k}$, and $y \in D_k^{*,\leqslant 1}(x)$. If $v \triangleq \phi(y)$ contains a 0^k substring, and $\bar{v}$ is derived from v by removing that substring, and if $\phi(x) \in \mathcal{W}$, then $\bar{v} \in \phi\Big(D_k^{*,\leqslant 1}(x)\Big)$.

Proof sketch: We denote by y' the descendant of x derived by the same string of duplications as y, without

substitutions. If $u' \triangleq \phi(y')$ contains 0^k at the same index, the claim is trivial. Otherwise, since v suffered no more than a single substitution, u' necessarily has a substring of length k and weight 1; since $\phi(x) \in \mathcal{W}$, that substring in u' is due to duplications, i.e., we may deduce that u' has a run of zeros of length at least k. Consequently, removing that 0^k substring, we are able to deduce the manner in which the resulting string (also a member of $\phi(D_k^{*,0}(x))$) differs from $\bar{v}$, and show that they are both in $\phi(D_k^{*,\leqslant 1}(c))$ for some $c \in D_k^{*,0}(x)$. ∎

Recall from [3] that a decoder for correcting an unbounded number of duplications simply has to remove incidents of 0^k from a noisy string. This lemma shows that the same approach can be taken with the addition of a single substitution–without increasing the substitution distance–provided that coding is done in $\mathcal{W}$.

Next, we consider the case where a substitution breaks a run of zeros (in the transform domain).

Lemma 8 Suppose $u \in \Sigma^{\geqslant k}$ contains a substring 0^k starting at index i, and suppose $v = u + a\epsilon_l$ for some $i \leqslant j < i + k$, $0 \neq a \in \Sigma$, and $l \in \{j, j-k\}$ (so that $v_j \neq 0$). Note that $v' \triangleq v - v_j\epsilon_j$ has a 0^k substring at index i (like u); We remove that substring from both u, v' to produce $\bar{u}, \bar{v}$, respectively. Then, irrespective of what value l takes, $\sigma(\bar{u}, \bar{v}) \leqslant 1$.

The lemma is straightforward to prove by case for v. It allows us to remove appearances of $0^j a 0^{k-1-j}$ form a noisy string (by applying an appropriate substitution) without increasing the substitution distance.

It is therefore seen that a restriction to $\mathcal{W}$ allows the correction of the substitution error without encountering the issue demonstrated in Example 6. This fact is more precisely stated in the theorem below:

Theorem 9 If $C \in \Sigma^{\geqslant k}$ is an error-correcting code for a single substitution, and $\phi(C) \subseteq \mathcal{W}$, then C is a 1S-correcting code.

B. Code Construction and Size

In this section we construct a family of codes satisfying Theorem 9. We also study the redundancy and rate of the proposed construction. We start by bounding the rate loss of using constrained coding by restricting codes to $\mathcal{W}$:

Lemma 10 $\frac{r(\mathcal{W} \cap \Sigma^n)}{n} \leqslant \frac{2}{k} \log_q \frac{q}{q-1}$.

Proof: We note that $C_n \subseteq \mathcal{W} \cap \Sigma^n$, where C_n is the set of length-n strings in which, divided into blocks of length k, every block ends with two non-zero elements. Hence, $\frac{r(\mathcal{W} \cap \mathbb{Z}_q^n)}{n} \leqslant \frac{r(C_n)}{n} = \frac{1}{n}\left(\left\lfloor \frac{n}{k} \right\rfloor + \left\lfloor \frac{n+1}{k} \right\rfloor\right) \leqslant \frac{2}{k} \log_q \frac{q}{q-1}$. ∎

Theorem 11 If q is a prime power, $r \geqslant 2$, and $n = \frac{q^r-1}{q-1} + \left\lceil \frac{2r}{k} \right\rceil$, then a 1S-correcting k-duplication code $C \subseteq \mathcal{W} \cap \mathbb{F}_q^n$ exists, with

$$R(C) \geqslant 1 - \frac{2}{k} \log_q \frac{q}{q-1} - o(1).$$

Proof: We begin by encoding data into $\mathcal{W} \cap \mathbb{F}_q^{\frac{q^r-1}{q-1}-r}$, incurring by Lemma 10 redundancy

$$r\left(\mathcal{W} \cap \mathbb{F}_q^{\frac{q^r-1}{q-1}-r}\right) \leqslant \left(\frac{q^r-1}{q-1} - r\right)\frac{2}{k} \log_q \frac{q}{q-1}.$$

Next, a systematic encoder for the $\left[\frac{q^r-1}{q-1}, r, 3\right]$ Hamming code (under the change of basis to $\{\epsilon_i\}$) can encode $\mathcal{W} \cap \mathbb{F}_q^{\frac{q^r-1}{q-1}-r} \to \mathbb{F}_q^{\frac{q^r-1}{q-1}}$, incurring r additional symbols of redundancy, and resulting in a code which can correct a single substitution.

Note, due to the systematic encoding, that the projection of this code onto the first $\frac{q^r-1}{q-1} - r$ coordinates is contained in $\mathcal{W}$. We may simply cushion the last r symbols with $\left\lceil \frac{2r}{k} \right\rceil$ interleaved 1's (two per k data symbols) to achieve a code $C \subseteq \mathcal{W} \cap \mathbb{F}_q^n$ which may still correct a single substitution. ∎

Taking $n \to \infty$, we can compare the rate obtained by the code in Theorem 11 to a simple upper bound of the best codes correcting only tandem duplications of length k (see [3]),

$$R(C) \leqslant 1 - \frac{(q-1)\log_q e}{q^{k+2}} + o(1).$$

While both the upper bound and lower bound approach 1 as $k \to \infty$, the lower bound does so as $\Theta(k^{-1})$ whereas the upper bound is much faster as $\Theta(q^{-k})$, implying a gap yet to be resolved.

REFERENCES

[1] F. Farnoud, M. Schwartz, and J. Bruck, "The capacity of string-duplication systems," *IEEE Transactions on Information Theory*, vol. 62, no. 2, pp. 811–824, 2016.

[2] ——, "Estimation of Duplication History under a Stochastic Model for Tandem Repeats," *BMC Bioinformatics*, vol. 20, no. 1, 2019. [Online]. Available: https://doi.org/10.1186/s12859-019-2603-1

[3] S. Jain, F. Farnoud, M. Schwartz, and J. Bruck, "Duplication-Correcting Codes for Data Storage in the DNA of Living Organisms," *IEEE Transactions on Information Theory*, vol. 63, no. 8, pp. 4996–5010, Aug. 2017.

[4] M. Kovačević and V. Y. F. Tan, "Asymptotically Optimal Codes Correcting Fixed-Length Duplication Errors in DNA Storage Systems," *IEEE Communications Letters*, vol. 22, no. 11, pp. 2194–2197, Nov. 2018.

[5] A. Lenz, A. Wachter-Zeh, and E. Yaakobi, "Duplication-Correcting Codes," *arXiv:1712.09345 [cs, math]*, Dec. 2017.

[6] H. Mahdavifar and A. Vardy, "Asymptotically optimal sticky-insertion-correcting codes with efficient encoding and decoding," in *2017 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2017, pp. 2683–2687.

[7] D. Pumpernik, B. Oblak, and B. Borštnik, "Replication slippage versus point mutation rates in short tandem repeats of the human genome," *Molecular Genetics and Genomics*, vol. 279, no. 1, pp. 53–61, 2008. [Online]. Available: http://dx.doi.org/10.1007/s00438-007-0294-1

[8] F. Sala, R. Gabrys, and L. Dolecek, "Gilbert-Varshamov-like lower bounds for deletion-correcting codes," in *2014 IEEE Information Theory Workshop (ITW 2014)*, Nov. 2014, pp. 147–151.

[9] F. Sala, R. Gabrys, C. Schoeny, and L. Dolecek, "Exact reconstruction from insertions in synchronization codes," *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 2428–2445, 2017.

[10] S. L. Shipman, J. Nivala, J. D. Macklis, and G. M. Church, "Molecular recordings by directed CRISPR spacer acquisition," *Science*, Jun. 2016.

[11] ——, "CRISPR–Cas encoding of a digital movie into the genomes of a population of living bacteria," *Nature*, vol. 547, no. 7663, pp. 345–349, Jul. 2017.

[12] K. Zhou, A. Aertsen, and C. W. Michiels, "The role of variable DNA tandem repeats in bacterial adaptation," *FEMS Microbiology Reviews*, vol. 38, no. 1, pp. 119–141, Jan. 2014.