Noise and Uncertainty in String-Duplication Systems

Siddharth Jain*, Farzad Farnoud (Hassanzadeh)†, Moshe Schwartz‡, and Jehoshua Bruck*
*Electrical Engineering, California Institute of Technology
Pasadena, CA 91125, U.S.A., {sidjain, bruck}@caltech.edu
†Electrical & Computer Engineering, University of Virginia
Charlottesville, VA 22904, U.S.A., farzad@virginia.edu
‡Electrical and Computer Engineering, Ben-Gurion University of the Negev
Beer Sheva 8410501, Israel, schwartz@ee.bgu.ac.il

***Abstract*—Duplication mutations play a critical role in the generation of biological sequences. Simultaneously, they have a deleterious effect on data stored using in-vivo DNA data storage. While duplications have been studied both as a sequence-generation mechanism and in the context of error correction, for simplicity these studies have not taken into account the presence of other types of mutations. In this work, we consider the capacity of duplication mutations in the presence of point-mutation noise, and so quantify the generation power of these mutations. We show that if the number of point mutations is vanishingly small compared to the number of duplication mutations of a constant length, the generation capacity of these mutations is zero. However, if the number of point mutations increases to a constant fraction of the number of duplications, then the capacity is nonzero. Lower and upper bounds for this capacity are also presented. Another problem that we study is concerned with the mismatch between code design and channel in data storage in the DNA of living organisms with respect to duplication mutations. In this context, we consider the uncertainty of such a mismatched coding scheme measured as the maximum number of input codewords that can lead to the same output.**

I. Introduction

With advances in sequencing and biological sequence synthesis, deoxyribonucleic acid (DNA) is emerging as a strong contender for satisfying future data storage needs. Its advantages include high data density and its longevity. Ideally, it can hold 230 exabyte/gram of data and last for hundreds of thousands of years. It is also the medium of choice for data storage in nature and thus artificial DNA data storage could be used as memory for synthetic-biology algorithms.

DNA storage can be divided into two types, ex vivo and in vivo. In the former, synthesized DNA is stored in a stable non-biological environment while in the latter, it is recombined with the DNA of a living organism. Applications proposed for in-vivo storage include watermarking genetically-modified organisms (GMOs) [1], [7], [13], labeling organisms in biological studies [15], long-term storage in a protected environment [2], [15], and biosteganography [3].

As with any other storage medium, DNA is not immune to noise and error. These errors may arise during the DNA synthesis and sequencing processes. They may also occur as spontaneous changes in the stored DNA in ex-vivo storage, and as a result of biological mutations in in-vivo storage. These mutations result in substitution, insertion, deletion, translocation, and duplication errors. Protecting the data in this setting, requires us to consider the *code design* problems arising from errors caused by mutations. In some cases, this provides a new motivation for classical challenging problems (e.g., deletions/insertions error correction), and in others introduces new problems (e.g., duplication and translocation error correction).

Mutations in general and duplications in particular [8] can also be viewed as sequence editing processes that give rise to biological sequences. This vantage point leads to questions regarding the ability of such mutations in generating a large family of sequences and the compressibility of the resulting sequences, quantified through the notion of the *capacity* of *string systems*. A string system is defined via a *seed* string and a set of *generation rules*, representing a given family of mutations, and includes all strings that can be obtained from the seed via a finite application of the generation rules. The capacity of a string system (defined formally later) is the asymptotic number of bits per string symbol required to represent a string from the string system, and thus measures both the generation power of the processes creating the sting system and the compressibility of the strings that are generated.

In our recent previous works, we have considered both the capacity problem [4], [6], [9], [10] and the code design problem [11], [12] for duplication mutations. In this case, the string systems are called the *string-duplication systems*. Specifically, we have studied *tandem*-duplication mutations, which change a string of the form $u = xyz$, where x,y, and z are substrings of u, into $v = xyyz$. For example, under tandem duplication $ACCGTG$ may become $ACCGT\underline{CGT}G$, where the extra copy resulting from the mutation is underlined.

In [4], [6], [9], [10] exact capacities, and bounds on the capacities, of string-duplication systems in various settings are given. In particular we show in [6] that when the length of the duplicated substring is fixed (referred to as the *duplication length*), one can only generate a polynomial number of strings,

This work was supported in part by the NSF Expeditions in Computing Program (The Molecular Programming Project).

and thus the capacity is 0. Furthermore, if the duplication length in a ternary system is bounded by 3, a capacity of $\log_3 \frac{3+\sqrt{5}}{2} \simeq 0.88$ trits/symbol can be achieved [10].

Error-correcting codes for errors caused by tandem duplications were studied in [11], [12]. In particular, an optimal family of codes for correcting errors due to tandem duplications of a fixed length and any number of errors was presented. We also studied codes for correcting tandem duplications of length up to a given constant k, where we primarily focused on the cases of $k = 2, 3$.

While the capacity of tandem-duplication systems have been studied, these mutations do not occur in isolation and other types of mutations, such as point mutations, are typically also present. Furthermore, in the case of duplication-correcting codes, the channel through which the sequences are passed is not always known. In this paper, we study two problems related to these shortcomings. First, we study the capacity of noisy tandem duplication where both tandem duplications and point mutations occur. We show in particular that if the number of point mutations is small compared to the number of duplications with a fixed length, the capacity is still 0. However, a linear number of point mutations results in a nonzero capacity. The related problem of code design for noisy tandem duplication is postponed to a future work. The second problem that we study here is the mismatch between code design and channel characteristics for tandem-duplication channels with respect to the maximum length of the duplication errors. In this case, we quantify the uncertainty resulting from this mismatch, i.e., the maximum number of possible inputs for one output, for channels in which the maximum duplication length is bounded by 3.

The rest of the paper is organized as follows. Section II presents the required definitions and notations. In Section III, we study the capacity of noisy tandem-duplication systems and related problems. Our results concerning the uncertainty resulting from the mismatch between the code and the channel are given in Section IV. We present conclusions and open problems in Section V.

II. Preliminaries

Given a string $x \in \Sigma^*$, a *tandem duplication of length* k is a process by which a contiguous substring of x of length k is copied next to itself. More precisely, we define the tandem-duplication rules, $T_{i,k} : \Sigma^* \to \Sigma^*$, for all $k \in \mathbb{N}$, $i \in \mathbb{N}_0$, as

$$T_{i,k}(x) \triangleq \begin{cases} uvvw & \text{if } x = uvw,\ |u| = i,\ |v| = k \\ x & \text{otherwise.} \end{cases}$$

We note that the "otherwise" case applies exactly when $|x| < k + i$, and therefore x cannot be decomposed into a prefix u of length i, an inner part v of length k, and some suffix w. A specific set of duplication rules that would be of interest to us throughout the paper is

$$\mathcal{T}_k \triangleq \left\{ T_{i,k} \;\middle|\; i \geqslant 0 \right\}.$$

Another operation of interest is that of point mutation. This operation overwrites a symbol in a string with another symbol (perhaps the same, in which case, no change happens). More precisely, we define the point-mutation rules, $P_{i,a} : \Sigma^* \to \Sigma^*$, for all $i \in \mathbb{N}_0$, $a \in \Sigma$, as

$$P_{i,a}(x) \triangleq \begin{cases} uaw & \text{if } x = ubw,\ |u| = i,\ b \in \Sigma, \\ x & \text{otherwise,} \end{cases}$$

and then define

$$\mathcal{P} \triangleq \left\{ P_{i,a} \mid i \geqslant 0, a \in \Sigma \right\}.$$

Given $x, y \in \Sigma^*$, if there exist $f \in \mathcal{T}_k \cup \mathcal{P}$ such that $y = f(x)$, then we say y is a direct descendant of x. If a sequence of $t + p$ operations $f_1, \dots, f_{t+p} \in \mathcal{T}_k \cup \mathcal{P}$, exactly p of which are point mutations from $\mathcal{P}$ and the rest are t k-tandem duplications from $\mathcal{T}_k$, such that $y = f_{t+p}(\dots(f_1(x))\dots)$, then we say y is an $(t+p)$-descendant of x and denote it by $x \xRightarrow[k]{t,p} y$. Finally, if there exists a finite sequence of transformations from $\mathcal{T}_k \cup \mathcal{P}$, transforming x into y, we say y is a descendant of x and denote it by $x \xRightarrow[k]{*,*} y$. We note that x is its own descendant via an empty sequence of transformations.

We define the *descendant cone* of $x \in \Sigma^*$ as

$$D_k^{*,*}(x) \triangleq \left\{ y \in \Sigma^* \;\middle|\; x \xRightarrow[k]{*,*} y \right\}.$$

In a similar fashion we define the (t, p)*-descendant cone* $D_k^{t,p}(x)$ by replacing $\xRightarrow[k]{*,*}$ with $\xRightarrow[k]{t,p}$ in the definition of $D_k^{*,*}(x)$. Additionally, whenever $p = 0$, i.e., no point mutations are involved, we use the simpler notation of $\xRightarrow[k]{*}$, $\xRightarrow[k]{t}$, D_k^*, and D_k^t.

We are now ready to define the *noisy tandem-duplication system*, denoted $S_k^p(s)$ over the alphabet Σ, for all tandem-duplication length $k \in \mathbb{N}$, amount of point-mutation $p : \mathbb{N} \to \mathbb{N}_0$, and initial seed string $s \in \Sigma^*$,

$$S_k^p(s) \triangleq \bigcup_{m \geqslant 0} D_k^{t,p(t)}(s),$$

i.e., it is the set of all the descendants of s obtained by using m transformations, of which $p(m)$ are point mutations. Using this notation, the tandem-duplication system studied in [6], [11], [12] is simply S_k^0.

An important figure of merit associated with any string system $S \subseteq \Sigma^*$ is its capacity, which intuitively measures the average information content in a symbol from a string in S. Assuming $|\Sigma| = q$, the *capacity* of $S \subseteq \Sigma^*$ is defined by

$$\mathsf{cap}(S) \triangleq \limsup_{n \to \infty} \frac{1}{n} \log_q |S \cap \Sigma^n|.$$

Another property of interest to us is expressiveness. We say a string-duplication system is *fully expressive* if for every $v \in \Sigma^*$ there exist $u, w \in \Sigma^*$ such that

$$s \xRightarrow[k]{*,*} uvw.$$

Namely, every given sequences v appears as a subsequence of some sequence derived from s.

Some strings cannot be derived from other strings. Formally, a string $s \in \Sigma^*$ is said to be *irreducible* with respect to $\mathcal{T}_k$ if there is no $s' \in \Sigma^*$, $s' \neq s$, such that $s' \overset{*}{\underset{k}{\Longrightarrow}} s$. We emphasize the fact that no point mutations are considered when defining irreducible strings. The set of all irreducible strings is denoted by Irr_k, and the set of all irreducible strings of length n is denoted by $\mathrm{Irr}_k(n)$.

Finally, we also consider the set of tandem-duplication rules,

$$\mathcal{T}_{\leqslant k} \triangleq \left\{ T_{i,k'} \mid i \geqslant 0, k \geqslant k' \right\}.$$

Replacing $\mathcal{T}_k$ by $\mathcal{T}_{\leqslant k}$ in all the definitions above gives us another string-duplication system of interest. All notation remains the same except k is replaced by $\leqslant k$ whenever appropriate.

III. Capacity and Expressiveness in S_k^p

The study of k-tandem duplication as a source of noise, using only duplication rules from $\mathcal{T}_k$, was described in [6], [11], [12]. In this section we consider a mix of k-tandem duplication together with point mutation, as a model for channel noise. In particular, we are interested in the capacity of error patterns, and the expressiveness of such a system.

Let us assume throughout this section that $\Sigma = \mathbb{Z}_q$. This does not constrain us in any way, and provides a structure we can use. An important tool in analyzing k-tandem-duplication systems is the transform domain defined by ϕ_k, which was described in [11], [12]. We define the mapping in a slightly different form, which keeps its essence but simplifies notation. Define $\phi_k : \mathbb{Z}_q^{\geqslant k} \to \mathbb{Z}_q^{\geqslant k}$ by,

$$\phi_k(x) \triangleq x0^k - 0^k x,$$

where subtraction is performed entry-wise over $\mathbb{Z}_q$. We comment that to obtain the original definition of ϕ_k of [11], [12] we delete the last k symbols and separate the sequence into its k-prefix and $|x| - k$ suffix.

Another mapping defined in [11], [12] injects k consecutive zeros into a string. We adjust the definition of $\zeta_{i,k}$ to match the change in ϕ_k. We therefore define $\zeta_{i,k} : \mathbb{Z}_q^{\geqslant k} \to \mathbb{Z}_q^{\geqslant k}$ by

$$\zeta_{i,k}(uv) \triangleq u0^k v,$$

where $u \in \Sigma^k$, $v \in \Sigma^*$.

The following lemma was proved in [11], [12].

Lemma 1. *For every string $x \in \mathbb{Z}_q^{\geqslant k}$, $0 \leqslant i \leqslant |x|$,*

$$\phi_k(T_{i,k}(x)) = \zeta_{i,k}(\phi_k(x)).$$

Intuitively, tandem-duplication operations of length k in the original domain appear as injections of 0^k in the transform domain. Thus, during many of the proofs it will be easier for us to consider strings in the transform domain, and only at the end use the reverse transform to obtain the result in the original domain.

One easily observes that the ϕ_k transform is linear, i.e., for all $x, x' \in \mathbb{Z}_q^{\geqslant k}$, $|x| = |x'|$,

$$\phi_k(x + x') = \phi_k(x) + \phi_k(x').$$

The same also holds for $\zeta_{i,k}$.

It was shown in [6] that $\mathsf{cap}(S_k^0) = 0$, regardless of the size of the alphabet Σ, and the starting string $s \in \Sigma^*$. We now show this changes when noisy duplication is present.

Theorem 2. *For any finite alphabet $\Sigma = \mathbb{Z}_q$, a seed string $s \in \Sigma^{\geqslant k}$, a tandem-duplication length $k \in \mathbb{N}$, and amount of point mutations $p : \mathbb{N} \to \mathbb{N}_0$, if $p(t) = o(t)$, then*

$$\mathsf{cap}(S_k^p(s)) = 0.$$

Proof: Let e_i denote a vector of all zeros except for the ith position which is 1, and whose length is implicit and understood from the context. The basis for the proof is the observation that in the transform domain, is a sequence with two non-zero elements: a 1 in the ith position, and a -1 in the $(k+i)$th position. Additionally, the elements of $\phi_k(e_i)$ sum to 0 over $\mathbb{Z}_q$.

Linearity of the transform ϕ_k guarantees that a single symbol change in the ith position of a sequence x, becomes

$$\phi_k(x + a \cdot e_i) = \phi_k(x) + a \cdot \phi_k(e_i),$$

for any $a \in \mathbb{Z}_q$.

Assume $s \overset{t,p(t)}{\underset{k}{\Longrightarrow}} x$, where we have

$$n \triangleq |x| = |s| + tk.$$

Further assume that in the derivation of x from s, the jth k-tandem duplication used is $T_{i_j,k}$. Additionally, the jth point mutation affects the ℓ_j coordinate by increasing it by a_j. Moreover, the jth point mutation occurs before the m_jth tandem duplications occurred. It now follows that

$$\begin{aligned}\phi_k(x) = {} & \zeta_{i_t,k}(\zeta_{i_{t-1},k}(\dots \zeta_{i_1,k}(\phi_k(s))\dots)) \\ & + \sum_{j=1}^{p(t)} a_j \cdot \zeta_{i_t,k}(\zeta_{i_{t-1},k}(\dots \zeta_{m_j,k}(\phi_k(e_{\ell_j}))\dots)).\end{aligned}$$

Hence, each derivation may be decomposed as a sum of a "noiseless" k-tandem-duplication process

$$\zeta_{i_t,k}(\zeta_{i_{t-1},k}(\dots \zeta_{i_1,k}(\phi_k(s))\dots))$$

and a "noise" component due to point mutations

$$\sum_{j=1}^{p(t)} a_j \cdot \zeta_{i_t,k}(\zeta_{i_{t-1},k}(\dots \zeta_{m_j,k}(\phi_k(e_{\ell_j}))\dots)).$$

The latter is a vector in the transform domain of length $n + k$ that has at most $2p(t)$ non-zero entries. The number of such noise vectors is upper bounded by

$$\Phi_{2p(t)}^{n+k} \triangleq \sum_{j=0}^{2p(t)} \binom{n+k}{j} (q-1)^j.$$

which is the size of a ball in the Hamming metric over $\mathbb{Z}_q^{n+k}$ and radius $2p(t)$.

We now have

$$\left|D_k^{t,p}(s)\cap\mathbb{Z}_q^n\right| \leqslant \left|D_k^{t,0}(s)\cap\mathbb{Z}_q^n\right|\cdot\Phi_{2p(t)}^{n+k}. \tag{1}$$

By [6, Th. 12],

$$\left|D_k^{t,0}(s)\cap\mathbb{Z}_q^n\right| \leqslant \binom{|s|-k+t}{|s|-1} = q^{n\cdot o(1)}.$$

Additionally, when $p(t)=o(1)$ we have (see [14])

$$\Phi_{2p(t)}^{n+k} = q^{n\cdot o(1)}.$$

Combining these together we obtain

$$\mathsf{cap}(S_k^p(s)) = 0.$$

■

Theorem 3. *For any finite alphabet $\Sigma=\mathbb{Z}_q$, a seed string $s\in\Sigma^{\geqslant k}$, a tandem-duplication length $k\in\mathbb{N}$, and amount of point mutations $p:\mathbb{N}\to\mathbb{N}_0$, if $p(t)=\alpha tk$, $\alpha\geqslant 0$, then*

$$\mathsf{cap}(S_k^p(s)) \leqslant \begin{cases} H_q(2\alpha) & 0\leqslant\alpha\leqslant\frac{1}{2}\left(1-\frac{1}{q}\right), \\ 1 & \alpha>\frac{1}{2}\left(1-\frac{1}{q}\right), \end{cases}$$

and

$$\mathsf{cap}(S_k^p(s)) \geqslant \begin{cases} H_q(\alpha) & 0\leqslant\alpha\leqslant 1-\frac{1}{q}, \\ 1 & \alpha>1-\frac{1}{q}, \end{cases}$$

where $H_q(\alpha)$ denotes the q-ary entropy function,

$$H_q(\alpha)\triangleq\alpha\log_q(q-1)-\alpha\log_q\alpha-(1-\alpha)\log_q(1-\alpha).$$

Proof: For the upper bound we use (1) again. Let $s\xRightarrow[k]{t,p(t)}x$, and $n\triangleq|x|=|s|+tk$. However now, when $p(t)=\alpha tk$, we have (see [14])

$$\Phi_{2p(t)}^{n+k} = \begin{cases} q^{nH_q(2\alpha)(1+o(1))} & 0\leqslant\alpha\leqslant\frac{1}{2}\left(1-\frac{1}{q}\right), \\ q^{n(1+o(1))} & \alpha>\frac{1}{2}\left(1-\frac{1}{q}\right). \end{cases}$$

For the lower bound, fix a single noiseless derivation, $s\xRightarrow[k]{t,0}x$, denoting $n\triangleq|x|=|s|+tk$. In the original domain, the number of (distinct) sequences obtainable from x by changing at most $p(t)$ positions is exactly $\Phi_{p(t)}^n$. Thus,

$$\left|D_k^{t,p(t)}(s)\right| \geqslant \Phi_{p(t)}^n = \begin{cases} q^{nH_q(\alpha)(1+o(1))} & 0\leqslant\alpha\leqslant 1-\frac{1}{q}, \\ q^{n(1+o(1))} & \alpha>1-\frac{1}{q}. \end{cases}$$

■

The lower bound of Theorem 3 may be improved by carefully constructing more strings. This is shown in the following.

Lemma 4. *For any finite alphabet $\Sigma=\mathbb{Z}_q$, a seed string $s\in\mathbb{Z}_q^{\geqslant k}$, a tandem-duplication length $k\in\mathbb{N}$, amount of point mutations $p:\in\mathbb{N}\to\mathbb{N}_0$, and any real constant $\beta\in[1,k]$, if $p(t)=\alpha tk$, $\alpha\in[0,\frac{\beta}{2k}]$, then*

$$\begin{aligned}\mathsf{cap}(S_k^p(s)) \geqslant{} & \frac{1}{k}H_2\left(\frac{2\alpha k}{\beta}\right)\log_q 2+\frac{2\alpha}{\beta}\log_q 2 \\ & +\frac{\alpha}{\beta}H_2\left(\beta-\lfloor\beta\rfloor\right)\log_q 2 \\ & +\frac{\alpha}{\beta}\log_q\left(\binom{k}{\lfloor\beta\rfloor}(q-1)^{\lfloor\beta\rfloor}-1\right) \\ & +\frac{\alpha}{\beta}\left(\beta-\lfloor\beta\rfloor\right)\log_q\frac{\binom{k}{\lfloor\beta\rfloor+1}(q-1)^{\lfloor\beta\rfloor+1}-1}{\binom{k}{\lfloor\beta\rfloor}(q-1)^{\lfloor\beta\rfloor}-1}.\end{aligned}$$

Proof: Our proof strategy relies on our ability to generate many distinct strings from the seed string s. We shall only use the last k symbols of s, and therefore we may assume w.l.o.g. that $|s|=k$. The choice of symbols of s will be immaterial. We shall additionally apply only tandem duplication operations $T_{i,k}$ where $k|i$. Thus, we may think of the seed string, as well as any intermediary string as a string comprised of a concatenation of blocks of length k, where each tandem-duplication operates on these blocks only. After t tandem-duplication operations we shall obtain a string of length $(t+1)k$, i.e., a string of $t+1$ blocks.

In addition, we have a budget of αtk point mutations. Whenever we use point mutations, we shall do so immediately after a block is tandem duplicated, mutating only symbols in the newly created block.

We encode each derivation sequence using a string made up of block identifiers of the form X_i, and delimiters from the set $\{(,),.\}$ (a left and right parentheses and a dot), with exactly one delimiter between adjacent block identifiers. Thus,

$$X_1.X_2(X_3(X_4)X_5.X_6)X_7 \tag{2}$$

is such a possible string. The string should have balanced parentheses, i.e., in every prefix of the string the number of left parentheses is at least the number of right parentheses. The encoding matches the derivation sequence using the following rules:

1) $X_i.X_{i+1}$ means the block X_{i+1} was tandem duplicated from X_i without any point mutations.
2) $X_i(X_{i+1}$ means the block X_{i+1} was tandem duplicated from X_i with at least one point mutation in X_{i+1}.
3) $X_{i_1}(\xi)X_{i_2}$, where ξ is a balanced string, means that X_{i_2} was tandem duplicated from X_{i_1} without any point mutations.
4) Whenever we have $X_{i_1}(\xi X_{i_2}(X_{i_3}$, where ξ is a balanced string, then $X_{i_1}\neq X_{i_3}$.

The derivation proceeds from the outer parentheses nesting level to the innermost, and within each nesting level, from left to right. Thus, the string of (2) uniquely describes a derivation depicted in Figure 1. The requirements also imply $X_1=X_2=X_7$, $X_3\neq X_2$, $X_4\neq X_3$, $X_4\neq X_2$, and $X_3=X_5=X_6$.

We first note that two derivations with the same encoding string but different point mutations obviously create distinct final sequences. Moreover, we contend that two derivations

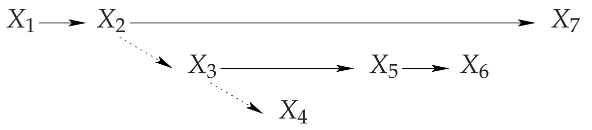

Figure 1. The derivation process of (2). Solid arrows represent tandem duplication. Dashed arrows represent tandem duplication with at least one point mutation. The process proceeds from top to bottom, and left to right.

with distinct encoding strings result in distinct final sequences. To show the latter we prove by simple induction that given the values of the blocks X_i, we can uniquely find the missing delimiters. The induction is on the index i. The base case is obvious. Assume we correctly found the missing delimiters given $X_1 \dots X_i$ and we are now given X_{i+1}. If $X_{i+1} = X_i$, then necessarily we have $X_i.X_{i+1}$. Otherwise $X_{i+1} \neq X_i$ and we distinguish between two cases depending on the largest $j < i$ such that $X_j(\xi X_i$ where ξ is a balanced string. If no such j exists, i.e., X_i is at the outermost nesting level of parentheses, then we must have $X_i(X_{i+1}$. If $X_{i+1} = X_j$, then by the third and fourth requirements we must have $X_i)X_{i+1}$. Otherwise, we have $X_i(X_{i+1}$.

At this point we introduce the parameter β. We shall consider only derivations whose encoding strings contain exactly $\frac{\alpha tk}{\beta}$ pairs of parentheses. Since each left parenthesis implies at least one point mutation, we shall distribute our budget of αtk point mutations as evenly as possible between the left parentheses. Thus, in $\frac{\alpha tk}{\beta}(1 - \beta + \lfloor \beta \rfloor)$ of the left parentheses we shall use $\lfloor \beta \rfloor$ point mutations, and in the remaining $\frac{\alpha tk}{\beta}(\beta - \lfloor \beta \rfloor)$ left parentheses we shall use $\lfloor \beta \rfloor + 1$ point mutations. Indeed, the two sum to the total budget,

$$\frac{\alpha tk}{\beta}(1 - \beta + \lfloor \beta \rfloor)\lfloor \beta \rfloor + \frac{\alpha tk}{\beta}(\beta - \lfloor \beta \rfloor)(\lfloor \beta \rfloor + 1) = \alpha tk.$$

We are now ready to count the number of sequences resulting from the process that was described above. We have an encoding string with tk delimiters, of which $2\alpha tk/\beta$ positions hold $\alpha tk/\beta$ pairs of parentheses. We have

$$\binom{t}{\frac{2\alpha tk}{\beta}} = 2^{tH_2(2\alpha k/\beta)(1+o(1))}$$

ways of choosing these positions. These $2\alpha tk/\beta$ positions are to be filled with a balanced sequence of parentheses, which may be done in

$$C_{\alpha tk/\beta} \triangleq \frac{1}{\frac{\alpha tk}{\beta}+1}\binom{\frac{2\alpha tk}{\beta}}{\frac{\alpha tk}{\beta}} = 2^{\frac{2\alpha tk}{\beta}(1+o(1))},$$

where C_i denotes the ith Catalan number. Next, of the $\alpha tk/\beta$ left parentheses we need to choose $\alpha tk/\beta \cdot (\beta - \lfloor \beta \rfloor)$ positions to have $\lfloor \beta \rfloor + 1$ point mutations each. This may be done in

$$\binom{\frac{\alpha tk}{\beta}}{\frac{\alpha tk}{\beta}(\beta - \lfloor \beta \rfloor)} = 2^{\frac{\alpha tk}{\beta}H_2(\beta - \lfloor \beta \rfloor)(1+o(1))}$$

distinct ways. Now, given a single tandem duplication with $\lfloor \beta \rfloor$ point mutations, we have

$$\binom{k}{\lfloor \beta \rfloor}(q-1)^{\lfloor \beta \rfloor} - 1$$

ways to choose the point mutations (choosing $\lfloor \beta \rfloor$ positions from the k positions in the block, and choosing for each point mutation a mutated value different from the current one). We note that we subtract one since we would like to eliminate an option that contradicts the fourth requirement. A similar expression is derived for blocks with $\lfloor \beta \rfloor + 1$ point mutations.

Combining all of the expressions above and substituting them in the expression for the capacity we get the desired lower bound. ■

An example showing the improved lower bounds is given in Figure 2.

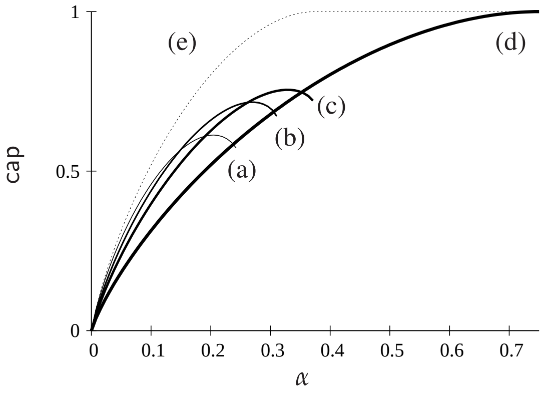

Figure 2. Bounds on $\mathsf{cap}(S_k^p(s))$ with $\Sigma = \mathbb{Z}_4$, $k = 2$, and $p(t) = \alpha kt$. The improved lower bound of Lemma 4 with (a) $\beta = 1$, (b) $\beta = 1.25$, (c) $\beta = 1.5$, as well as (d) the lower bound of Theorem 3 and (e) the upper bound of Theorem 3.

Theorem 5. *For any finite alphabet $\Sigma = \mathbb{Z}_q$, a seed string $s \in \Sigma^*$, a tandem-duplication length $k \in \mathbb{N}$, and amount of point mutations $p : \mathbb{N} \to \mathbb{N}_0$, the system $S_k^p(s)$ is fully expressive if and only if $p(t) = \omega(1)$.*

Proof: We are given that $p(t) = \omega(1)$, i.e., we have an unbounded budget of point mutations. Then, for any $v \in \mathbb{Z}_q^*$, there exists $t \in \mathbb{N}$ such that $\min(p(t), |s| + tk) \geqslant |v|$. Consider now the following derivation: first employ t tandem duplications $T_{0,k}$ starting from s, resulting in a string x whose length satisfies $|x| \geqslant |v|$. Now apply $|v| \leqslant p(t)$ point mutations to the first $|v|$ positions of x, so that they are mutated into v. Thus, $s \overset{t,p(t)}{\underset{k}{\Longrightarrow}} vw$, for some $w \in \mathbb{Z}_q^*$, and the system is fully expressive.

In the other direction, suppose $p(t) = O(1)$. Then for any $s \overset{t,p(t)}{\underset{k}{\Longrightarrow}} v$ we have that $\phi_k(v)$ contains at most $2(|s| + p(t)) < M$ non-zero elements, where M is some constant. In particular, we have no descendant of s whose ϕ_k-transform has at least M non-zero elements. ■

We point out some interesting observations. First we note that $S_k^p(s)$ is the first natural string-duplication system that is fully expressive, yet without full capacity (cf. [6], [9]). Second, note that as long as $p(t) = o(t)$ and $p(t) = \omega(1)$, we have a natural fully expressive system with capacity 0.

IV. Uncertainty in Coding for $S_{\leqslant k}$

Suppose we have an error-correcting code designed specifically to correct tandem duplications from $\mathcal{T}_{\leqslant k}$. However, if we incorrectly estimate the value of k, we may end up transmitting our codewords over a channel the tandem-duplicates using rules from $\mathcal{T}_{\leqslant k'}$, $k' \neq k$. This mismatch between the design parameter and the actual channel parameter may cause mis-decoding. We quantify the number of possible mis-decodings as the channel-code uncertainty, which we study in this section.

Error-correcting codes for uniform and bounded tandem-duplication errors were given in [11], [12]. In particular, the following code was constructed for correcting any number of tandem-duplication errors of length $\leqslant k$, for $k = 2$ and $k = 3$,

$$C_{\leqslant k}(n) \triangleq \bigcup_{i=1}^{n} \left\{ \xi_{n-i}(x) \mid x \in \operatorname{Irr}_{\leqslant k}(i) \right\}.$$

where for any sequence $x = x_0 \dots x_{n-1} \in \Sigma^+$, $x_i \in \Sigma$, its ℓ-suffix-extension

$$\xi_\ell(x) \triangleq x x_{n-1}^{\ell},$$

i.e., the sequence x with its last symbol repeated an extra ℓ times.

Motivated by the error-correcting code defined above, in this section we consider the problem of sending codewords of the form

$$C_U(n) \triangleq \bigcup_{i=1}^{n} \left\{ \xi_{n-i}(x) \mid x \in \operatorname{Irr}_U(i) \right\} \tag{3}$$

through a $\mathcal{T}_{\leqslant 3}$ channel. Here $U \subseteq \mathbb{N}$ is a finite set of positive integers, and Irr_U denotes the set of strings not containing a tandem repeat of length appearing in U. We quantify this problem by defining *uncertainty*, which measures the size of the maximum subset $M_U(n) \subseteq C_U(n)$, such that $D^*_{\leqslant 3}(c) = y$ for every $c \in M_U(n)$ and some $y \in \Sigma^*$. Mathematically,

$$\operatorname{Unc}_U(n) \triangleq \max_{y \in \Sigma^*} \left| \left\{ x \in C_U(n) \mid y \in D^*_{\leqslant 3}(x) \right\} \right|. \tag{4}$$

We first recall some results from [11] which we summarize in the following lemma.

Lemma 6. *[11] For any $y \in \Sigma^*$ there exists a unique $x \in \operatorname{Irr}_{\leqslant 3}$ such that $y \in D^*_{\leqslant 3}(x)$. Additionally, for $y_1, y_2 \in \Sigma^*$, we have $D^*_{\leqslant 3}(y_1) \cap D^*_{\leqslant 3}(y_2) \neq \emptyset$ if and only if there exists $x \in \operatorname{Irr}_{\leqslant 3}$ such that $y_1, y_2 \in D^*_{\leqslant 3}(x)$.*

We also make the following trivial observation. For all $k, k' \in \mathbb{N}$, $k \leqslant k'$, we have $\operatorname{Irr}_{\leqslant k'} \subseteq \operatorname{Irr}_{\leqslant k}$, and therefore $C_{\leqslant k'}(n) \subseteq C_{\leqslant k}(n)$.

The next lemma provides a characterization of the uncertain codewords, and gives an expression for the uncertainty.

Lemma 7. *For all $n \in \mathbb{N}$, and finite $U \subseteq \mathbb{N}$,*

$$\operatorname{Unc}_U(n) = \max_{s \in \operatorname{Irr}_{\leqslant 3}} \left| D^*_{\leqslant 3}(s) \cap C_U(n) \right|.$$

Proof: By (4) we are looking for elements of $C_U(n)$ that share some descendant when using $\mathcal{T}_{\leqslant 3}$. By Lemma 6 they must also share a unique irreducible ancestor (under $\mathcal{T}_{\leqslant 3}$). ■

Before proceeding we need another simple lemma.

Lemma 8. *Let $U \subseteq \mathbb{N}$ be finite, with $\{1,2,3\} \cap U \in \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}\}$. Then for any $s \in \Sigma^*$,*

$$D^*_{\leqslant 3}(s) \cap \operatorname{Irr}_U = D^*_{\{1,2,3\} \setminus U}(s) \cap \operatorname{Irr}_U.$$

Proof: The right-hand side of the equation is trivially contained in the left-hand side. Thus, we only need to prove any $x \in D^*_{\leqslant 3}(s) \cap \operatorname{Irr}_U$ may be derived from s using tandem-duplication operations with lengths from $\{1,2,3\} \setminus U$. This, again, is trivial when $\{1,2,3\} \cap U \in \{\emptyset, \{1,2,3\}\}$.

We now observe that once a tandem duplication of length 1 is employed, all future descendants contain a substring aa, $a \in \Sigma$. Thus, if $\{1,2,3\} \cap U = \{1\}$, and $x \in D^*_{\leqslant 3}(s) \cap \operatorname{Irr}_U$, i.e., x does not contain a substring aa, $a \in \Sigma$, then deriving x from s using $\mathcal{T}_{\leqslant 3}$ does not require a tandem duplication of length 1. Similarly, if a tandem duplication of length 2 is employed, a substring of the form $abab$, $a, b \in \Sigma$ is created. The only way to eliminate it is by tandem duplication of length 1, which in itself cannot be later eliminated. Thus, the case of $\{1,2,3\} \cap U = \{1,2\}$ is handled as well. ■

This brings us to the following corollary, which provides an upper bound on the uncertainty.

Corollary 9. *Let $U \subseteq \mathbb{N}$ be finite, with $\{1,2,3\} \cap U \in \{\emptyset, \{1\}, \{1,2\}, \{1,2,3\}\}$. Then for any $s \in \Sigma^*$,*

$$\operatorname{Unc}_U(n) \leqslant \max_{s \in \operatorname{Irr}_{\leqslant 3}} \left| D^*_{\{1,2,3\} \setminus U}(s) \cap \Sigma^{\leqslant n} \right|. \tag{5}$$

Proof: We have,

$$\begin{aligned}
\operatorname{Unc}_U(n) &\overset{(a)}{=} \max_{s \in \operatorname{Irr}_{\leqslant 3}} \left| D^*_{\leqslant 3}(s) \cap C_U(n) \right| \\
&\overset{(b)}{\leqslant} \max_{s \in \operatorname{Irr}_{\leqslant 3}} \sum_{i=1}^{n} \left| D^*_{\leqslant 3}(s) \cap \operatorname{Irr}_U(i) \right| \\
&\overset{(c)}{=} \max_{s \in \operatorname{Irr}_{\leqslant 3}} \sum_{i=1}^{n} \left| D^*_{\{1,2,3\} \setminus U}(s) \cap \operatorname{Irr}_U(i) \right| \\
&\leqslant \max_{s \in \operatorname{Irr}_{\leqslant 3}} \left| D^*_{\{1,2,3\} \setminus U}(s) \cap \Sigma^{\leqslant n} \right|,
\end{aligned}$$

where (a) is by Lemma 7, (b) is by (3), and (c) is by Lemma 8. ■

We note that for large values of n, the right-hand side of (5) is dominated by a term of the form $|\Sigma|^{cn}$, where $c = \mathsf{cap}(S^0_{\{1,2,3\} \setminus U}(s))$, which by [6], [9], is known in some cases.

We now turn to another issue of importance. A mismatch between a code designed for the $\mathcal{T}_{\leqslant k}$-channel and an actual $\mathcal{T}_{\leqslant k'}$-channel, $k < k'$, may be avoided if we can determine the value of k'. As a first step, we set out to find substrings that may never occur in $D^*_{\leqslant k}(s)$, but may appear as substrings of $D^*_{\leqslant k+1}(s)$. An occurrence of such substrings in a channel's output can be used as a marker indicating it is a $\mathcal{T}_{\leqslant k'}$-channel, with $k' \geqslant k+1$.

A closely related work is [9], where the expressiveness of bounded tandem duplication string systems was characterized. The relevant results of [9] are summarized in Table I.

TABLE I
EXPRESSIVENESS OF $S_{\leqslant k}(s) = S(\Sigma, s, \mathcal{T}_{\leqslant k})$.

Σ	s	k	fully expressive
$\{1,2\}$	12	$\geqslant 2$	Yes
$\{1,2,3\}$	arbitrary	$\leqslant 3$	No
$\{1,2,3\}$	123	$\geqslant 4$	Yes
$\lvert\Sigma\rvert \geqslant 4$	arbitrary	arbitrary	No

From Table I, we observe the following:

- For $|\Sigma| = 2$, $S_{\leqslant k}$ is fully expressive for any $k > 1$ and was shown to generate any binary string from some seed with $k > 1$ in [9]. Hence there exists no binary string which cannot be generated by $S_{\leqslant k}(s)$ system but can be generated by $S_{\leqslant k+1}(s)$ system for $k > 1$.
- For $|\Sigma| = 3$, $S_{\leqslant k}$ is fully expressive for $k > 3$, but it is not known whether an $S_{\leqslant k+1}$ system can generate anything new over $S_{\leqslant k}$ for $k > 3$. Since, the system is fully expressive this problem is intuitively harder compared to higher alphabets.
- For $|\Sigma| > 3$, since we do not have full expressiveness for any k, finding examples which can be generated by $S_{\leqslant k+1}(s)$ but not by $S_{\leqslant k}(s)$ is intuitively easier.

To summarize, for $|\Sigma| > 3$, we should gain in expressiveness by increasing k. The next two lemmas prove this statement. We recall that a string is called *squarefree* if it does not contain a substring of the form ww, with $w \in \Sigma^+$.

Lemma 10. *Let $\Sigma = \mathbb{Z}_q$, $q \geqslant 4$, and $k > 0$. If $z \in (\mathbb{Z}_q \setminus \{0\})^k$ is squarefree, and $s \in \mathbb{Z}_q^*$ does not contain $w \triangleq 0z0$ as a substring, then there is no $y \in D^*_{\leqslant k}(s)$ which contains w as a substring.*

Proof: Let $x \in D^*_{\leqslant k}(s)$ and assume $v = v_1 v_2 \dots v_\ell$ is a substring of x, where $\ell > k$. If $v \in \mathrm{Irr}_{\leqslant k}$, then by [9], either $v_1 = v_{1+i}$ for some $2 \leqslant i \leqslant k$, or $v_\ell = v_{\ell-j}$ for some $2 \leqslant j \leqslant k$. We note that $w \triangleq 0z0$ does not satisfy these requirements. On the other hand, $w \in \mathrm{Irr}_{\leqslant k}$, and the claim follows. ■

Lemma 11. *Let $\Sigma = \mathbb{Z}_q$, $q \geqslant 4$, and $k > 0$. If $z \in (\mathbb{Z}_q \setminus \{0\})^k$ is squarefree, and $s \in \mathbb{Z}_q^*$ contains $w' \triangleq 0z$ as a substring, then there exists $y \in D^*_{\leqslant k+1}(s)$ which contains $w \triangleq 0z0$ as a substring.*

Proof: Simply duplicate the w' part (of length $k+1$) to obtain a string with w as a substring. ■

V. DISCUSSION

The capacity of the tandem-duplication system without point mutations, $\mathsf{cap}(S^0_k(s))$, was proven to be 0 in [6]. This capacity determines the exponential growth rate of descendant cones, when using only tandem-duplication operations. When building an error-correcting code to protect against tandem duplications, these descendant cones take on the role of error spheres, and an error-correcting code is therefore a packing of these spheres. Even though their capacity (without point mutations) is 0, the channel capacity (determined by the size of the optimal code) is not full, as was shown in [12].

As shown in this paper, in the presence of point mutations, the growth rate of the tandem-duplication descendant cones is positive, $\mathsf{cap}(S^p_k(s))$, as long as the fraction of point mutations does not vanish. Thus, we may expect the channel capacity in the model with point mutations and tandem duplications, to drop, and perhaps vanish. This channel capacity, and the construction of coding schemes for this channel, is part of ongoing work by the authors.

An additional source of errors we considered in this paper is due to a mismatch between the channel parameters and the error-correcting code we employ. We focused on the the bounded tandem-duplication system $S_{\leqslant k}(s)$, and studied the effects of sending codewords from a code designed for $S_{\leqslant k}(s)$ through a channel that uses $S_{\leqslant k+1}(s)$. A particular goal, which is the subject of ongoing research, is determining an unknown value of k used by the channel. Steps toward solving this combinatorial problem were given here. In combination with [5], a probabilistic framework for the estimation of k is a future research interest.

REFERENCES

[1] M. Arita and Y. Ohashi, "Secret signatures inside genomic DNA," *Biotechnology Progress*, vol. 20, no. 5, pp. 1605–1607, 2004.

[2] F. Balado, "Capacity of DNA data embedding under substitution mutations," *IEEE Trans. Inform. Theory*, vol. 59, no. 2, pp. 928–941, Feb. 2013.

[3] T. D. P. Brunet, "Aims and methods of biosteganography," *Journal of biotechnology*, vol. 226, pp. 56–64, 2016.

[4] F. Farnoud, M. Schwartz, and J. Bruck, "The capacity of string-duplication systems," in *Proceedings of the 2014 IEEE International Symposium on Information Theory (ISIT2014), Honolulu, HI, USA*, Jun. 2014, pp. 1301–1305.

[5] ——, "A stochastic model for genomic interspersed duplication," in *Proceedings of the 2015 IEEE International Symposium on Information Theory (ISIT2015), Hong Kong, China SAR*, Jun. 2015, pp. 1731–1735.

[6] ——, "The capacity of string-duplication systems," *IEEE Trans. Inform. Theory*, vol. 62, no. 2, pp. 811–824, Feb. 2016.

[7] D. Heider and A. Barnekow, "DNA-based watermarks using the DNA-Crypt algorithm," *BMC Bioinformatics*, vol. 8, no. 1, pp. 1–10, 2007.

[8] International Human Genome Sequencing Consortium, "Initial sequencing and analysis of the human genome," *Nature*, vol. 409, no. 6822, pp. 860–921, 2001. [Online]. Available: http://www.nature.com/nature/journal/v409/n6822/abs/409860a0.html

[9] S. Jain, F. Farnoud, and J. Bruck, "Capacity and expressiveness of genomic tandem duplication," in *Proceedings of the 2015 IEEE International Symposium on Information Theory (ISIT2015), Hong Kong, SAR China*, Jun. 2015, pp. 1946–1950.

[10] ——, "Capacity and expressiveness of genomic tandem duplication," Paradise Laboratory, California Institute of Technology, Tech. Rep. ETR127, Jan. 2015. [Online]. Available: http://www.paradise.caltech.edu/papers/etr127.pdf

[11] S. Jain, F. Farnoud, M. Schwartz, and J. Bruck, "Duplication-correcting codes for data storage in the DNA of living organisms," Paradise Laboratory, California Institute of Technology, Tech. Rep. ETR131, Jan. 2016. [Online]. Available: http://www.paradise.caltech.edu/papers/etr131.pdf

[12] ——, "Duplication-correcting codes for data storage in the DNA of living organisms," in *Proceedings of the 2016 IEEE International Symposium on Information Theory (ISIT2016), Barcelona, Spain*, Jul. 2016, pp. 1028–1032.

[13] M. Liss, D. Daubert, K. Brunner, K. Kliche, U. Hammes, A. Leiherer, and R. Wagner, "Embedding permanent watermarks in synthetic genes," *PLoS ONE*, vol. 7, no. 8, p. e42465, 08 2012.

[14] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. North-Holland, 1978.

[15] P. C. Wong, K.-k. Wong, and H. Foote, "Organic data memory using the DNA approach," *Commun. ACM*, vol. 46, no. 1, pp. 95–98, Jan. 2003.