# The Capacity of String-Duplication Systems

Farzad Farnoud (Hassanzadeh), Moshe Schwartz, *Senior Member, IEEE*, and Jehoshua Bruck, *Fellow, IEEE*

***Abstract*— It is known that the majority of the human genome consists of duplicated sequences. Furthermore, it is believed that a significant part of the rest of the genome also originated from duplicated sequences and has mutated to its current form. In this paper, we investigate the possibility of constructing an exponentially large number of sequences from a short initial sequence using simple duplication rules, including those resembling genomic-duplication processes. In other words, our goal is to find the capacity, or the expressive power, of these string-duplication systems. Our results include exact capacities, and bounds on the capacities, of four fundamental string-duplication systems. The study of these fundamental biologically inspired systems is an important step toward modeling and analyzing more complex biological processes.**

***Index Terms*— Capacity, DNA, string duplication, formal languages, constrained coding.**

## I. Introduction

MORE than 50% of the human genome consists of *repeated sequences* [9]. An important class of these repeated sequences are *interspersed repeats*, which are caused by *transposons*. A transposon, or a "jumping gene", is a segment of DNA that can "copy and paste" or "cut and paste" itself into new positions of the genome. Currently, 45% of the human genome is known to consist of transposon-driven repeats [9].

A second type of repeats are *tandem repeats*, generally thought to be caused by *slipped-strand mispairings* [16]. A slipped-strand mispairing is said to occur when, during DNA synthesis, one strand in a DNA duplex becomes misaligned with the other. These mispairings may lead to deletions or insertion of a repeated sequence [12]. While tandem repeats are known to constitute only 3% of the human genome, they cause important phenomena such as chromosome fragility, expansion diseases, silencing genes [18], and rapid morphological variation [6].

While interspersed repeats and random repeats together account for a significant part of the human genome, it is conceivable that a substantial portion of the unique genome, the part that is not known to contain repeated sequences, also has its origins in ancient repeated sequences that are no longer recognizable due to change over time [9], [18].

Other processes of DNA mutation include insertion and deletion of subsequences, as well as point mutation which replaces a single element in the DNA sequence with another. These alone can mutate a DNA sequence, over time, and potentially create any target sequence. This is in contrast with duplication mutation processes which, at first glance, do not appear to create new information. Can duplication processes alone account for the diversity seen in nature?

Thus, motivated by the prevalence and the significance of repeated sequences and the fact that much of our unique DNA was likely originally repeated sequences, in this paper we study the *capacity* of *string-duplication systems* with simple duplication rules including those resembling the repeat-producing genomic processes, namely duplication of transposons and duplication caused by slipped-strand mispairings. A string-duplication system, to be defined formally later, consists of an initial sequence, a set of rewriting rules, and all sequences that can be obtained by applying the rules to the initial sequence a finite number of times. The notion of capacity, defined later in the paper, represents the average number of bits per symbol that can asymptotically be encoded by the sequences in a string-duplication system, and thus illustrates the expressive power and the diversity of that system.

In this paper, we consider four duplication rules. The first is the *end-duplication* rule, which allows substrings of a certain length $k$ to be appended to the end of existing sequences. For example, if $k = 3$ we may construct the sequence TCATGCCAT from TCATGC. While the biological motivation for this rule is somewhat tenuous, and it may be seen as an extension of a different rule studied below, we present it first because of the simplicity of proving the related results. In particular, we show that nearly all sequences with the same alphabet as the initial sequence can be generated with this rule.

The second rule is called *tandem duplication* and allows a substring of length $k$ to be duplicated next to its original position. For example, for $k = 3$, from the sequence TCATGC, one can generate TCATCATGC. We show that this rule has capacity zero regardless of the initial sequence. However, if one allows substrings of all lengths larger than a given value to be copied, the capacity becomes positive except in trivial cases.

The third rule is *palindromic duplication*, which is similar to tandem duplication except that the copy is reversed before insertion. For instance, in our previous example, the sequence TCATTACGC can be generated. This rule is biologically inspired, motivated by palindromic repeats in DNA which also

Manuscript received November 24, 2014; revised July 10, 2015; accepted October 26, 2015. Date of publication December 4, 2015; date of current version January 18, 2016. This work was supported by the National Science Foundation within the Expeditions in Computing Program through the Molecular Programming Project. This paper was presented in part at the 2014 IEEE International Symposium on Information Theory.

F. Farnoud and J. Bruck are with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125 USA (e-mail: farnoud@caltech.edu; bruck@paradise.caltech.edu).

M. Schwartz is with the Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Beer Sheva 8410501, Israel (e-mail: schwartz@ee.bgu.ac.il).

Communicated by A. G. Dimakis, Associate Editor for Coding Techniques.
Digital Object Identifier 10.1109/TIT.2015.2505735

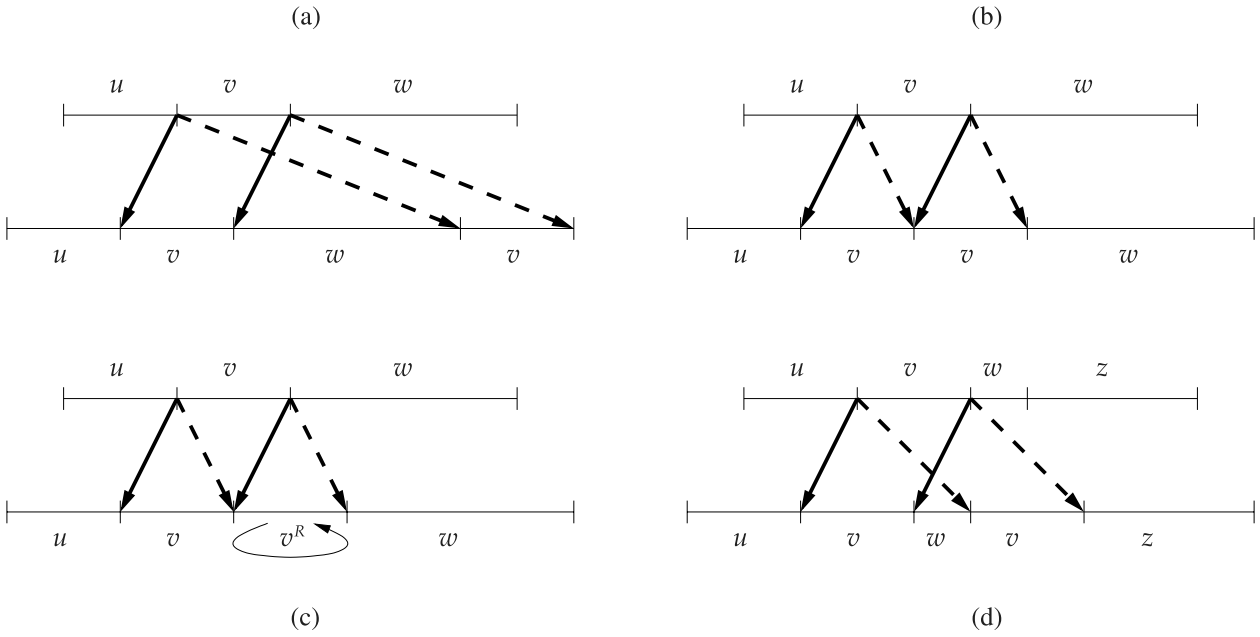


Fig. 1. A depiction of four duplication rules: (a) end duplication, (b) tandem duplication, (c) palindromic duplication, and (d) interspersed duplication.

involve complementing the base pairs (for example, see [13]). To avoid imposing any structure on the underlying alphabet and to allow odd-sized alphabets, in this study we use the simpler model which only reverses the substring, but does not complement it. We also note that reversing a substring without complementing it was studied in the context of evolutionary grammars, e.g., [2]. Here, in the palindromic duplication case, the capacity is zero only in the trivial case in which the initial sequence has only one unique symbol.

The last rule is *interspersed duplication*, where the copy of a substring of a given length $k$ can be inserted after $k'$ symbols. This rule is motivated by the fact that transposons may insert themselves in places far from their original positions. As an example, for $k = 3$ and $k' = 1$, from TCATGC, one can obtain TCATGCATC. For this rule, we show that the capacity is zero if and only if the initial sequence is periodic with period equal to the greatest common divisor of $k$ and $k'$.

The four duplication rules are depicted in Figure 1. Our results regarding the expressive power of string-duplication systems suggest that duplication in genome may be a source of genomic diversity.

We note that the biologically-inspired mathematical model we suggest is not entirely new. Indeed, the special case of tandem duplication has been already studied in a series of papers [3], [4], [10], [11]. However, this was done in the context of the theory of formal languages, and the goal of these studies was mainly to determine their place in the Chomsky hierarchy of formal languages. We also mention the work [15], which aims to enumerate RNA folds that are chemically stable by modeling them as words belonging to context-free languages.

In the next section, we present the preliminaries and in the following four sections, we present the results for each of the aforementioned duplication rules. The concluding remarks are presented in Section VII.

## II. Preliminaries

Let $\Sigma$ be some finite alphabet. We recall some useful notation commonly used in the theory of formal languages. An $n$-string $x = x_1x_2\cdots x_n \in \Sigma^n$ is a finite sequence of alphabet symbols, $x_i \in \Sigma$. We say $n$ is the length of $x$ and denote it by $|x| = n$. For two strings, $x \in \Sigma^n$ and $y \in \Sigma^m$, their concatenation is denoted by $xy \in \Sigma^{n+m}$. The set of all finite strings over the alphabet $\Sigma$ is denoted by $\Sigma^*$. We say $v \in \Sigma^*$ is a *substring* of $x$ if $x = uvw$, where $u, w \in \Sigma^*$. For $s \in \Sigma^*$ and a non-negative integer $k$, we use $s^k$ to denote the sequence obtained by concatenating $k$ copies of $s$. A finite or infinite string $s$ is *periodic* with period $p$ if for all $1 \leqslant i \leqslant |s| - p$, we have $s_i = s_{i+p}$.

The *alpha-representation* of a string $s$, denoted by $R(s)$, is the set of all letters from $\Sigma$ making up $s$. Thus, if $s = s_1s_2\cdots s_n \in \Sigma^n$,

$$R(s) = \{s_1, s_2, \ldots, s_n\} \subseteq \Sigma.$$

The *alpha-diversity* of $s$ is the size of the alpha-representation of $s$, denoted by

$$\delta(s) = |R(s)|.$$

Given a set $S \subseteq \Sigma^*$, we denote

$$S^* = \{w_1w_2\cdots w_m \mid w_i \in S, m \geqslant 0\},$$

whereas

$$S^+ = \{w_1w_2\cdots w_m \mid w_i \in S, m \geqslant 1\}.$$

For any $x \in \Sigma^*$, $|x| = n \geqslant m$, the $m$-suffix of $x$ is $w \in \Sigma^m$, such that $x = vw$ for some $v \in \Sigma^*$. Similarly, the $m$-prefix of $x$ is $u \in \Sigma^m$, where $x = uv$ for some $v \in \Sigma^*$. Finally, we conveniently denote $[n] = \{1, 2, \ldots, n\}$.

A *string system* $S$ is a subset $S \subseteq \Sigma^*$. For any integer $n$, we denote by $N_S(n)$ the number of length-$n$ strings in $S$, i.e.,

$$N_S(n) = \left|S \cap \Sigma^n\right|.$$

An important measure associated with a string system is that of capacity. We use the definition given by Shannon in his analysis of the discrete noiseless channel [17], which is also commonly used in the constrained-coding theory [7], [14].

*Definition 1: The* capacity *of a string system $S \subseteq \Sigma^*$ is defined by*

$$\mathsf{cap}(S) = \limsup_{n\to\infty} \frac{\log_2 N_S(n)}{n}.$$

Since $N_S(n) \leqslant |\Sigma|^n$ we immediately note that

$$\mathsf{cap}(S) \leqslant \log_2 |\Sigma|,$$

for all $S$. Also, for any infinite set $S$,

$$\mathsf{cap}(S) \geqslant 0.$$

We pause to mention [1] which studied a context-free model that included deletion, reversal, transposition, and tandem-duplication operations. A similar question was asked in [1]: what is the number of strings obtainable after a fixed number of operations are applied to a given set of seed strings? Inverse questions were also studied, i.e., given some numerical parameters, can we construct a context-free model that has these parameters? However, the results of [1] do not have any bearing on the current paper.

To further study the capacity of string systems we require some tools used in constrained-coding theory, mostly, Perron-Frobenius theory. For an in-depth study the reader is referred to [14] (in particular, Chapter 4). A way of generating a set of strings is the following: Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{L})$ be a finite directed graph, with vertices $\mathcal{V}$, directed edges $\mathcal{E}$, and a labeling function $\mathcal{L} : \mathcal{E} \to \Sigma$ with $\Sigma$ a finite alphabet. A path in the graph is a sequence of edges $e_1, e_2, \ldots, e_\ell$ such that for each $1 \leqslant i \leqslant \ell - 1$, the destination vertex of $e_i$ is the source vertex of $e_{i+1}$. We say the path generates the word $w \in \Sigma^*$ if $w = \mathcal{L}(e_1)\mathcal{L}(e_2)\ldots\mathcal{L}(e_\ell)$. The string system generated by the graph $S = S(\mathcal{G})$ is the set of all words generated by finite paths in $\mathcal{G}$.

Some other properties of the graph are of interest. We say $\mathcal{G}$ is *irreducible* if for every (ordered) pair of vertices $v_1, v_2 \in \mathcal{V}$, there is a directed path from $v_1$ to $v_2$. An irreducible graph is said to be *primitive* if the gcd of the lengths of cycles in $\mathcal{G}$ is 1. In particular, a self-loop in an irreducible $\mathcal{G}$ forces the graph to be primitive.

A graph $\mathcal{G}$ is said to be *lossless* if any two distinct paths that agree on the starting vertex, and agree on the ending vertex, must generate distinct words. In particular, if for every vertex $v \in \mathcal{V}$, the outgoing edges from $v$ are labeled by distinct elements of $\Sigma$, then the graph is *deterministic* and also lossless.

The adjacency matrix of $\mathcal{G}$, denoted $A_\mathcal{G}$, is the $|\mathcal{V}| \times |\mathcal{V}|$ integer matrix, whose $i, j$ entry is the number of edges directed from $v_i$ to $v_j$. By Perron-Frobenius theory [14, Th. 4.2.3] the *spectral radius* of $A_\mathcal{G}$ (the largest absolute value of any eigenvalue of $A_\mathcal{G}$), which is denoted $\lambda(A_\mathcal{G})$, is also an eigenvalue of $A_\mathcal{G}$. For an irreducible lossless $\mathcal{G}$, we have [14, Th. 4.4.4],

$$\mathsf{cap}(S(\mathcal{G})) = \log_2 \lambda(A_\mathcal{G}).$$

If in addition $\mathcal{G}$ is primitive, then the number of paths (words) of length $n$ starting at vertex $v_i \in \mathcal{V}$ and ending at vertex $v_j \in \mathcal{V}$ is given by [14, Th. 4.5.12]

$$\big(c_{i,j} + o(1)\big)\, \lambda(A_\mathcal{G})^n,$$

where $c_{i,j}$ is a positive constant, and $o(1)$ denotes a function tending to 0 as $n$ grows.

We now turn to describe another property of string systems we will be interested in.

*Definition 2: Let $S \subseteq \Sigma^*$ be a string system. We shall say $S$ is* fully diverse *if for every $v \in \Sigma^*$ there exist $u, w \in \Sigma^*$ such that $uvw \in S$.*

In other words, a string system $S$ is fully diverse if every finite sequence appears as a subsequence of a string in $S$. We observe that a fully-diverse system $S$ does not necessarily have maximum capacity of $\log_2 |\Sigma|$. As an example, let $\Sigma = \{0, 1\}$, and consider $S = \{vv \mid v \in \Sigma^*\}$. Obviously $S$ is fully diverse, but $\mathsf{cap}(S) = \frac{1}{2}$.

In the other direction, however, if a string system $S$ has full capacity, i.e., $\mathsf{cap}(S) = \log_2 |\Sigma|$, then it must be fully diverse. For otherwise, there exists a finite-length sequence $v$ that never appears as a subsequence of a string in $S$. By the theory of constrained systems (for example see [14]), $S$ cannot have full capacity.

A basic component in our setting is a *string-duplication rule*. Such a rule, $T$, is nothing but a function $T : \Sigma^* \to \Sigma^*$. In what follows, these rules will be defined as taking subsequences and duplicating them in various ways under certain conditions, hence the name string-*duplicating* rules. We will also use $\mathcal{T}$ (with appropriate subscripts and superscripts) to denote sets of string-duplication rules, i.e., $\mathcal{T} \subseteq {\Sigma^*}^{\Sigma^*}$.

The following is the main definition used throughout the paper, describing a string system that evolves from a seed sequence using certain rules.

*Definition 3: Let $\Sigma$ be a finite alphabet, $s \in \Sigma^*$ some string, and $\mathcal{T} \subseteq {\Sigma^*}^{\Sigma^*}$ a set of string-duplication rules. A* string-duplication system*, $S$, defined by the tuple $(\Sigma, s, \mathcal{T})$, is the closure of $\mathcal{T}$ operating on $s$, namely, $S \subseteq \Sigma^*$ is the minimal set for which:*

1) $s \in S$.
2) $s' \in S$ *and* $T \in \mathcal{T}$ *imply* $T(s') \in S$.

*We write $S = S(\Sigma, s, \mathcal{T})$.*

We can think of $s$ as a *seed string*, and the strings resulting from applying the functions in $\mathcal{T}$ to $s$, as the *descendants* of $s$. Thus, $S$ contains $s$, its descendants, its descendants' descendants, and so on.

We close this section with two simple lemmas.

*Lemma 4: Let $\Sigma$ be a finite alphabet, and $s, s' \in \Sigma^*$. Consider $S = S(\Sigma, s, \mathcal{T})$ and $S' = S(\Sigma, s', \mathcal{T})$ two string-duplication systems. If $s' \in S$, then*

$$\mathsf{cap}(S') \leqslant \mathsf{cap}(S).$$

*Proof:* Since $s' \in S$ it is trivial that $S' \subseteq S$, and the claim follows. ∎

For the second lemma we need to define a certain kind of duplication rules.

*Definition 5: A set of duplication rules $\mathcal{T}$ is said to be* extension invariant *if for all $s, u, w \in \Sigma^*$ and $T \in \mathcal{T}$, if $T(s) = v$, then there exists $T' \in \mathcal{T}$ such that $T'(usw) = uvw$.*

*Lemma 6: Let $\Sigma$ be a finite alphabet, and $s \in \Sigma^*$. Consider $S = S(\Sigma, s, \mathcal{T})$ a string-duplication system, where $\mathcal{T}$ is extension invariant. Then for any $u, w \in \Sigma^*$ and $S' = S(\Sigma, usw, \mathcal{T})$, we have*

$$\mathsf{cap}(S) \leqslant \mathsf{cap}(S').$$

*Proof:* Since $\mathcal{T}$ is extension invariant, we note that any $v \in S$ implies $uvw \in S'$. Thus,

$$N_S(n) \leqslant N_{S'}(n + |uw|).$$

Since $|uw|$ is a constant independent of $n$, plugging this inequality in the definition of the capacity gives the desired result. ∎

We can tie all of these concepts together in the following theorem.

*Theorem 7: Let $\Sigma$ be a finite alphabet, $s \in \Sigma^*$, and let $S = S(\Sigma, s, \mathcal{T})$ be a fully-diverse string-duplication system with an extension invariant $\mathcal{T}$. Then $\mathsf{cap}(S)$ does not depend on $s$.*

*Proof:* Denote $S_i = S(\Sigma, s_i, \mathcal{T})$ for $i = 1, 2$. Since $S_2$ is fully diverse, there exist $u, w \in \Sigma^*$ such that $us_1w \in S_2$. Thus, since $\mathcal{T}$ is extension invariant, by Lemma 6 and then Lemma 4 we get

$$\mathsf{cap}(S_1) \leqslant \mathsf{cap}(S(\Sigma, us_1w, \mathcal{T})) \leqslant \mathsf{cap}(S_2).$$

Symmetrically, we can show that $\mathsf{cap}(S_2) \leqslant \mathsf{cap}(S_1)$. Hence, $\mathsf{cap}(S_1) = \mathsf{cap}(S_2)$. ∎

We note that throughout the paper, all the duplication rules that are used preserve the alpha representation, i.e., $R(T(w)) = R(w)$ for all rules $T$ and strings $w \in \Sigma^*$. Thus, the requirement of full diversity in Theorem 7, throughout this work, implicitly requires $R(s) = \Sigma$. However, this is not necessarily the case for general rules that do not preserve the alpha representation of strings.

We also mention in passing that we can change some of the requirements of Theorem 7 and still obtain the same claim. This may be done be replacing full diversity with an analogous full suffix diversity (i.e., every finite string is obtainable as a suffix of some word in the system), and also replacing extension invariance with left-extension invariance. However, this will not be useful in this paper.

## III. End Duplication

The first string-duplication system we study is end duplication. We show in this Section that this system has full capacity, as well as full diversity.

We define the end-duplication function, $T^{\text{end}}_{i,k} : \Sigma^* \to \Sigma^*$, as follows:

$$T^{\text{end}}_{i,k}(x) = \begin{cases} uvwv & \text{if } x = uvw,\ |u| = i,\ |v| = k \\ x & \text{otherwise.} \end{cases}$$

Namely, $T^{\text{end}}_{i,k}$ operates on a string by taking a substring of length $k$, starting from the $(i+1)$-st position, and copying it at the end of the string.

*Example 8: Let $s = \mathsf{TCATGC}$. Then*

$$T^{\text{end}}_{1,3}(s) = \mathsf{TCATGCCAT},$$

*by substituting $u = \mathsf{T}$, $v = \mathsf{CAT}$, and $w = \mathsf{GC}$. However,*

$$T^{\text{end}}_{4,3}(s) = \mathsf{TCATGC} = s,$$

*since no partition $s = uvw$ exists with $|u| = 4$ and $|v| = 3$.* □

We also define two sets of these functions which will be used later:

$$\mathcal{T}^{\text{end}}_k = \left\{ T^{\text{end}}_{i,k} \;\middle|\; i \geqslant 0 \right\},$$
$$\mathcal{T}^{\text{end}}_{\geqslant k} = \left\{ T^{\text{end}}_{i,k'} \;\middle|\; i \geqslant 0, k' \geqslant k \right\}.$$

In the case of end duplication we can find the exact capacity, as shown by the following theorem.

*Theorem 9: Let $\Sigma$ be any finite alphabet, $k \geqslant 1$ any integer, and $s \in \Sigma^*$, $|s| \geqslant k$. Then for $S^{\text{end}}_k = S(\Sigma, s, \mathcal{T}^{\text{end}}_k)$,*

$$\mathsf{cap}(S^{\text{end}}_k) = \log_2 \delta(s).$$

*Proof:* First we note that by requiring $|s| \geqslant k$ we avoid the degenerate case of $S^{\text{end}}_k$ containing only $s$. We further note that, by the definition of the duplication functions,

$$R(x) = R(T^{\text{end}}_{i,k}(x))$$

for all non-negative integers $i$ and $k$, and thus, all the strings in $S^{\text{end}}_k$ have the same alpha-representation. Thus, trivially,

$$\mathsf{cap}(S^{\text{end}}_k) \leqslant \log_2 \delta(s).$$

We now turn to prove the inequality in the other direction. We contend that given a string $x \in \Sigma^*$, $|x| \geqslant k$, and some string $w \in \Sigma^k$, $R(w) \subseteq R(x)$, with at most $2k$ duplication steps we can obtain from $x$ a string $y \in \Sigma^*$ ending with $w$, i.e., $y = vw$, for some $v \in \Sigma^*$.

As a first step, we duplicate the $k$-prefix of $x$, i.e., if $x = uv$, for $|u| = k$ and some $v$, then

$$x' = T^{\text{end}}_{0,k}(x) = uvu.$$

By doing so we ensure that for any symbol $a \in R(x)$ there is a $k$-substring of $x'$ starting with $a$, and a $k$-substring of $x'$ ending with $a$.

Let us now denote the symbols of $w$ as $w = w_1 w_2 \cdots w_k$, $w_i \in \Sigma$. Assume that the $k$-substring of $x'$ starting at position $i_1$ ends with $w_1$. We form

$$x_1 = T^{\text{end}}_{i_1-1,k}(x')$$

whose 1-suffix is just $w_1$. Next, assume the $k$-substring of $x'$ starting at position $i_2$ starts with $w_2$. Note that $x'$ is a prefix of $x_1$. We form

$$x_2 = T^{\text{end}}_{|x_1|-k+1,k}\left(T^{\text{end}}_{i_2-1,k}(x_1)\right).$$

It easy to verify $x_2$ has a 2-suffix of $w_1w_2$. Continuing in the same way, let $i_j$ be the starting position of a $k$-substring of $x'$ starting with $w_j$. We form

$$x_j = T^{\text{end}}_{|x_{j-1}|-k+1,k}\left(T^{\text{end}}_{i_j-1,k}(x_{j-1})\right),$$

for $j = 3, \ldots, k$. Note that $x_j$ has a $j$-suffix $w_1, \ldots, w_j$.

It follows that after $2k$ duplication steps we can obtain from any such $x$ a string with any given $k$-suffix $w$, provided $R(w) \subseteq R(x)$. Thus, from the initial string $s$, we can obtain a string $s'$ with all of the strings of $R(s)^k$ appearing as $k$-substrings, using at most $2k\delta(s)^k$ duplication steps,[1] i.e.,

$$|s'| \leqslant |s| + 2k^2\delta(s)^k.$$

After having obtained $s'$, each duplication may duplicate any of the $k$-strings in $R(s)^k$ in a single operation. Thus, for all $n = |s'| + tk$, $t$ a non-negative integer, the number of distinct strings in $S_k^{\text{end}}$ is bounded from below by

$$N_{S_k^{\text{end}}}(n) \geqslant \delta(s)^{n-|s'|}.$$

Since $|s'|$ is a constant, we have

$$\mathsf{cap}(S_k^{\text{end}}) \geqslant \log_2 \delta(s).$$

■

The following is an obvious corollary of Theorem 9.

*Theorem 10: Let $\Sigma$ be any finite alphabet, $k \geqslant 1$ any integer, and $s \in \Sigma^*$, $|s| \geqslant k$. Then for $S_{\geqslant k}^{\text{end}} = S(\Sigma, s, \mathcal{T}_{\geqslant k}^{\text{end}})$,*

$$\mathsf{cap}(S_{\geqslant k}^{\text{end}}) = \mathsf{cap}(S_k^{\text{end}}) = \log_2 \delta(s).$$

*Proof:* Since for all $n \geqslant k$,

$$N_{S_k^{\text{end}}}(n) \leqslant N_{S_{\geqslant k}^{\text{end}}}(n) \leqslant \delta(s)^n,$$

the claim follows. ■

Since by Theorem 9 and Theorem 10, $S_k^{\text{end}}$ and $S_{\geqslant k}^{\text{end}}$ have full capacity if $R(s) = \Sigma$, the following corollary is immediate.

*Corollary 11: Both $S_k^{\text{end}}$ and $S_{\geqslant k}^{\text{end}}$ are fully diverse, provided the seed string, $s$, satisfies $|s| \geqslant k$, and $R(s) = \Sigma$.*

## IV. Tandem Duplication

We now turn to consider tandem duplication. In complete contrast with the previous section, we show that fixed-length tandem duplication has capacity 0 and is never fully diverse. However, tandem duplication with only lower-bounded duplication length has positive capacity and is fully diverse.

We define the tandem-duplication rules, $T_{i,k}^{\tan} : \Sigma^* \to \Sigma^*$, as

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

We also define the sets

$$\mathcal{T}_k^{\tan} = \left\{ T_{i,k}^{\tan} \;\middle|\; i \geqslant 0 \right\},$$
$$\mathcal{T}_{\geqslant k}^{\tan} = \left\{ T_{i,k'}^{\tan} \;\middle|\; i \geqslant 0, k' \geqslant k \right\}.$$

In words, tandem duplication takes a $k$-substring and duplicates it adjacent to itself in the string. The following theorem shows that the capacity of tandem-duplication systems is in complete contrast to end-duplication systems.

*Theorem 12: Let $\Sigma$ be any finite alphabet, $k$ any positive integer, and $s \in \Sigma^*$, with $|s| \geqslant k$. Then for $S_k^{\tan} = S(\Sigma, s, \mathcal{T}_k^{\tan})$,*

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

[1]This bound may be improved, but this will not affect the capacity calculation.

*Proof:* Consider any $n$-string $x \in \Sigma^n$, $n \geqslant k$. Instead of viewing $x = x_1x_2\cdots x_n$ as a sequence of $n$ symbols from $\Sigma$, we can, by abuse of notation, view it as a sequence of $n-k+1$ overlapping $k$-substrings $x = x'_1x'_2\cdots x'_{n-k+1}$, where

$$x'_i = x_ix_{i+1}\cdots x_{i+k-1}.$$

For a $k$-string $y = y_1y_2\cdots y_k$, $y_i \in \Sigma$, its cyclic shift by one position is denoted by

$$Ey = y_2y_3\cdots y_ky_1.$$

A cyclic shift by $j$ positions is denoted by

$$E^jy = y_{j+1}y_{j+2}\cdots y_ky_1y_2\cdots y_j.$$

We say two $k$-strings, $y, z \in \Sigma^k$, are cyclically equivalent if

$$y = E^jz,$$

for some integer $j$. Clearly this is an equivalence relation. Let $\phi(y)$ denote the equivalence class of $y$. If $y$ and $z$ are cyclically equivalent, then $\phi(y) = \phi(z)$.

We now define

$$\Phi(x) = \phi(x'_1)\phi(x'_2)\cdots\phi(x'_{n-k+1}),$$

i.e., $\Phi(x)$ is the image of the overlapping $k$-substrings of $x$ under $\phi$. We also observe that knowing $x'_1$ and $\Phi(x)$ enables a full reconstruction of $x$.

At this point we turn to consider the effect of the duplication $T_{i,k}^{\tan}$ on a string $x \in \Sigma^*$, $|x| \geqslant k$. When viewed as a sequence of overlapping $k$-substrings, as defined above,

$$\begin{aligned} T_{i,k}^{\tan}(x) = {} & x'_1\cdots x'_{i-1} \\ & x'_iEx'_iE^2x'_i\cdots E^{k-1}x'_ix'_i \\ & x'_{i+1}\cdots x'_{n-k+1}. \end{aligned}$$

Since $\phi(x'_i) = \phi(E^j(x'_i))$ for all $j$, we have

$$\begin{aligned} \Phi(T_{i,k}^{\tan}(x)) = {} & \phi(x'_1)\cdots\phi(x'_{i-1}) \\ & \phi(x'_i)\phi(x'_i)\cdots\phi(x'_i) \\ & \phi(x'_{i+1})\cdots\phi(x'_{n-k+1}), \end{aligned}$$

where $\phi(x'_i)$ appears $k+1$ consecutive times.

Thus, we may think of $\phi(x'_i)$ as a bin, and the action of $T_{i,k}^{\tan}$ as throwing $k$ balls into the bin $\phi(x'_i)$. The number of bins does not change throughout the process, and is equal to one more than the number of times $\phi(x'_i) \neq \phi(x'_{i+1})$, where $x = s$ is the original string. If $b$ is the number of bins defined by $s$, then the number of strings obtained by $m$ duplications is exactly $\binom{b+m-1}{b-1}$. Since this number grows only polynomially in the length of the resulting string, we have

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

■

We note that $S_k^{\tan}$ is not necessarily fully diverse. For example, take any $k \in \mathbb{N}$, $\Sigma = \{0, 1, 2\}$, and $s = 0^k1^k2^k$. It is easily seen that in all the strings of $S_k^{\tan} = S(\Sigma, s, T_k^{\tan})$, all the 0's appear before all the 2's. In fact, a stronger statement is provided by the following theorem.

*Theorem 13: Let $\Sigma$ be a finite alphabet, $|\Sigma| \geqslant 2$, and $s \in \Sigma^*$. Then $S_k^{\tan} = S(\Sigma, s, \mathcal{T}_k^{\tan})$ is never fully diverse.*

*Proof:* Like in the proof of Theorem 12, denote $s = s_1 s_2 \cdots s_n$, define $s'_i = s_i s_{i+1} \cdots s_{i+k-1}$, and consider

$$\Phi(s) = \phi(s'_1)\phi(s'_2)\cdots\phi(s'_n - k + 1).$$

If, for convenience, we denote all the possible equivalence classes as $\phi_1, \phi_2, \ldots, \phi_m$, then we can write

$$\Phi(s) = \phi_{i_1}^{j_1}\phi_{i_2}^{j_2}\cdots\phi_{i_\ell}^{j_\ell},$$

for some positive integers $i_1, \ldots, i_\ell$ and $j_1, \ldots, j_\ell$, and where $i_t \neq i_{t+1}$ for all $t$.

By the proof of Theorem 12, every sequence $v \in S_k^{\tan}$ satisfies

$$\Phi(v) = \phi_{i_1}^{j_1+t_1k}\phi_{i_2}^{j_2+t_2k}\cdots\phi_{i_\ell}^{j_\ell+t_\ell k}, \tag{1}$$

for some non-negative integers $t_1, \ldots, t_\ell$. When $|\Sigma| \geqslant 2$ we have $m \geqslant 2$, i.e., there is more than one equivalence class. Thus, we contend that there is a sequence whose $\Phi$ transform is not a subsequence of (1). If such a sequence exists, it does not appear as a subsequence of a string in $S_k^{\tan}$.

We now prove the existence of such a sequence. Let $v \in S_k^{\tan}$ be a string in the system, and $\Phi(v)$ of the form given in (1). Denote $|v| = n$, and $v = v_1 v_2 \cdots v_n$, where $v_i \in \Sigma$. The $k$-suffix of $v$ is $v_{n-k+1}v_{n-k+2}\cdots v_n$. Arbitrarily choose a letter $v_{n+1} \in \Sigma$ such that $v_{n-k+1} \neq v_{n+1}$. Obviously $v_{n-k+1}v_{n-k+2}\cdots v_n$ is not cyclically equivalent to $v_{n-k+2}v_{n-k+3}\cdots v_n v_{n+1}$. Thus, let $i_{\ell+1}$ be an index such that

$$\begin{aligned}\phi_{i_\ell} &= \phi(v_{n-k+1}v_{n-k+2}\cdots v_n)\\ &\neq \phi(v_{n-k+2}v_{n-k+3}\cdots v_n v_{n+1}) = \phi_{i_{\ell+1}}.\end{aligned}$$

It follows that

$$\Phi(vv_{n+1}) = \phi_{i_1}^{j_1+t_1k}\phi_{i_2}^{j_2+t_2k}\cdots\phi_{i_\ell}^{j_\ell+t_\ell k}\phi_{i_{\ell+1}}.$$

Since $i_\ell \neq i_{\ell+1}$, we must have that $\Phi(vv_{n+1})$ is never a subsequence of any sequence of equivalence classes of the form (1). ∎

*Example 14: Set $\Sigma = \{0, 1\}$, and $k = 2$. Denote the equivalence classes under cyclic rotation as*

$$\begin{aligned}\phi_1 &= \phi(00),\\ \phi_2 &= \phi(01) = \phi(10),\\ \phi_3 &= \phi(11).\end{aligned}$$

*Assume we pick $s = 000110$. Then*

$$\Phi(s) = \phi(00)\phi(00)\phi(01)\phi(11)\phi(10) = \phi_1^2\phi_2\phi_3\phi_2.$$

*By Theorem 13, all the strings $v \in S_2^{\tan} = S(\Sigma, s, \mathcal{T}_2^{\tan})$ have the following $\Phi$ transform,*

$$\Phi(v) = \phi_1^{2+2t_1}\phi_2^{1+2t_2}\phi_3^{1+2t_3}\phi_2^{1+2t_4}. \tag{2}$$

*Take for example $w = 11011$. We have*

$$\Phi(w) = \phi_3\phi_2^2\phi_3.$$

*Since $\Phi(w)$ is never a subsequence of (2), $w$ never appears as a subsequence of a string in $S_2^{\tan} = S(\Sigma, s, \mathcal{T}_2^{\tan})$.* □

When considering $S_{\geqslant k}^{\tan} = S(\Sigma, s, \mathcal{T}_{\geqslant k}^{\tan})$ the situation appears to be harder to analyze. We start with the case of $k = 1$.

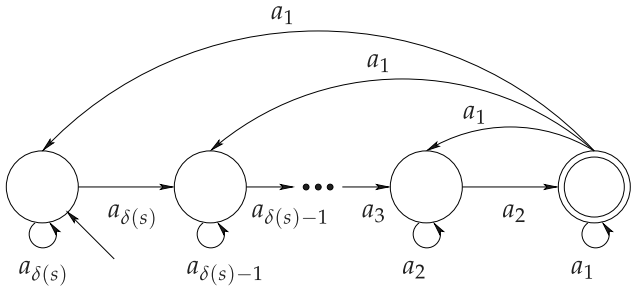


Fig. 2. The finite-state automaton accepting the regular language used in the proof of Theorem 15.

*Theorem 15: For any finite alphabet $\Sigma$, and any string $s \in \Sigma^*$ of nontrivial alpha-diversity, $\delta(s) \geqslant 2$, we have*

$$\mathsf{cap}(S_{\geqslant 1}^{\tan}) \geqslant \log_2(r + 1),$$

*where $r$ is the largest (real) root of the polynomial*

$$f(x) = x^{\delta(s)} - \sum_{i=0}^{\delta(s)-2} x^i.$$

*Proof:* The proof strategy is the following: we shall show that $S_{\geqslant 1}^{\tan}$ contains, among other things, a regular language. The capacity of that regular language will serve as the lower bound we claim.

For the first phase of the proof, assume $i_1 < i_2 < \cdots < i_{\delta(s)}$ are the indices of the $\delta(s)$ distinct alphabet symbols in $s$. We produce a sequence of strings, $s_0, s_1, \ldots, s_{\delta(s)-1}$, with $s_0 = s$, defined iteratively by

$$s_j = T^{\tan}_{i_{\delta(s)-j}-1, i_{\delta(s)}-i_{\delta(s)-j}+j}(s_{j-1}),$$

for $j = 1, 2, \ldots, \delta(s) - 1$. After this set of steps, the $\delta(s)$-substring starting at position $i_{\delta(s)}$ of $s_{\delta(s)-1}$ contains $\delta(s)$ distinct symbols. In what follows we will only use these symbols for duplication, and thus, the constant amount of other symbols in $s_{\delta(s)-1}$ does not affect our calculation of the lower bound on capacity. Thus, for ease of presentation we shall assume from now on that $s = a_{\delta(s)}a_{\delta(s)-1}\cdots a_1$, where $a_j \in \Sigma$ are distinct.

We now perform the following iterations: In iteration $i$, where $i = \delta(s), \delta(s)-1, \ldots, 2$, we duplicate only $i$-substrings equal to $a_i a_{i-1}\cdots a_2 a_1$ (as many times as we want). As a final iteration, we may duplicate 1-substrings without constraining their content. It is easy to verify the resulting strings form the following regular language,

$$S = \left(a^+_{\delta(s)}\left(a^+_{\delta(s)-1}\left(\cdots\left(a_2^+\left(a_1^+\right)^+\right)^+\right)^+\right)^+\right)^+.$$

The construction process implies $S \subseteq S_{\geqslant 1}^{\tan}$.

The finite-state automaton accepting the regular language $S$ is depicted in Figure 2. The graph is primitive and lossless, and thus, for the purpose of calculating the capacity, instead of counting the number of length $n$ words in $S$, we can count the number of length-$n$ paths in the depicted automaton graph $\mathcal{G}$,

starting and ending in the appropriate vertices (see [7], [14]). By Perron-Frobenius theory,

$$\mathsf{cap}(S^{\tan}_{\geqslant 1}) \geqslant \mathsf{cap}(S) = \log_2 \lambda(A_{\mathcal{G}}),$$

where $\lambda(A_{\mathcal{G}})$ is the largest magnitude of an eigenvalue of $A_{\mathcal{G}}$, and where $A_{\mathcal{G}}$ denotes the adjacency matrix of $\mathcal{G}$. We note that $A_{\mathcal{G}}$ is the $\delta(s) \times \delta(s)$ matrix

$$A_{\mathcal{G}} = \begin{pmatrix} 1 & 1 & & & & \\ & 1 & 1 & & & \\ & & 1 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & 1 & 1 \\ 1 & 1 & 1 & \dots & 1 & 1 \end{pmatrix},$$

and its largest eigenvalue is the largest real root of

$$\det(\lambda I - A_{\mathcal{G}}) = (\lambda - 1)^{\delta(s)} - \sum_{i=0}^{\delta(s)-2} (\lambda - 1)^i.$$

Setting $x = \lambda - 1$ we obtain the desired result. ∎

*Remark 16: Before proceeding we pause to reflect on the lower bound of Theorem 15. Let $r$ be the largest root of $f(x)$, defined in Theorem 15, and $\delta = \delta(s)$. We contend that*

$$\frac{1 + \sqrt{5}\sqrt{1 - 4/\left(5\delta^2\right)}}{2\left(1 + 1/\delta\right)} \leqslant r \leqslant \frac{1 + \sqrt{5}}{2}. \tag{3}$$

*This clearly holds if $\delta = 2$, with equality on the left-hand side of* (3)*. Assume $\delta > 2$. Note that*

$$f(x) = x^{\delta} - \frac{x^{\delta-1} - 1}{x - 1} = \frac{x^{\delta+1} - x^{\delta} - x^{\delta-1} + 1}{x - 1}.$$

*So $r$ is the same as the largest root of $g(x) = x^{\delta+1} - x^{\delta} - x^{\delta-1} + 1 = x^{\delta-1}\left(x^2 - x - 1\right) + 1$ which is in turn less than the largest root of $g(x) - 1 = x^{\delta-1}\left(x^2 - x - 1\right)$. Hence $r \leqslant \frac{1+\sqrt{5}}{2}$. For the derivative $g'$ of $g$, we have $g'(x) = x^{\delta-2}\left((\delta+1)x^2 - \delta x - (\delta - 1)\right)$, which has three roots: one positive, one negative, and one equal to zero. Since $g(1) = 0$ and $g'(1) < 0$, the function $g(x)$ has a zero after the largest zero of $g'(x)$, completing the proof of (3).*

*Based on the lower bound given in (3) and the trivial upper bound of $\log_2 \delta(s)$, for $S^{\tan}_{\geqslant 1} = S\left(\Sigma, s, T^{\tan}_{\geqslant 1}\right)$, with $\delta(s) = 2$, 3, and 4, we respectively have*

$$\begin{aligned} 1 &\leqslant \mathsf{cap}(S^{\tan}_{\geqslant 1}) \leqslant 1 \\ 1.12127 &\leqslant \mathsf{cap}(S^{\tan}_{\geqslant 1}) \leqslant 1.58496 \\ 1.18382 &\leqslant \mathsf{cap}(S^{\tan}_{\geqslant 1}) \leqslant 2. \end{aligned}$$

*While these bounds are good for small values of $\delta(s)$, as $\delta(s)$ increases the lower bound becomes close to $\log_2\left(1 + \frac{1+\sqrt{5}}{2}\right) = 1.38848$ and the upper bound grows as $\log_2(\delta(s))$, and so the gap grows large.*

At least in one case, the bound of Theorem 15 is attained with equality, as is shown in the following corollary.

*Corollary 17: For $\Sigma = \{0, 1\}$, and $s \in \Sigma^*$ with $\delta(s) = 2$ we have*

$$\mathsf{cap}(S^{\tan}_{\geqslant 1}) = 1.$$

*Proof:* By applying Theorem 15 we get

$$\mathsf{cap}(S^{\tan}_{\geqslant 1}) \geqslant 1.$$

We also have the trivial upper bound

$$\mathsf{cap}(S^{\tan}_{\geqslant 1}) \leqslant \log_2 |\Sigma| = 1,$$

which completes the proof. ∎

For $S^{\tan}_{\geqslant k}$ and general $k$, we claim a weaker result, that is provided in the following theorem.

*Theorem 18: For any finite alphabet $\Sigma$, and any binary string $s \in \Sigma^*$, $|s| \geqslant k$, of nontrivial alpha-diversity, $\delta(s) \geqslant 2$, we have*

$$\mathsf{cap}(S^{\tan}_{\geqslant k}) \geqslant \log_2 r > 0,$$

*where $r$ is the largest root of the polynomial*

$$f(x) = x^{k+1} - x - 1.$$

*Proof:* The proof strategy is, again, to find a regular language that is a subset of $S^{\tan}_{\geqslant k}$ and use its capacity as a lower bound. We start with the following preparation, by performing the following $k$ duplications,

$$s' = T^{\tan}_{0,2k-1}\left(\dots\left(T^{\tan}_{0,k+1}\left(T^{\tan}_{0,k}(s)\right)\right)\right).$$

If we denote $s' = s'_1 s'_2 \cdots$, where $s'_i \in \Sigma$, then it is easy to verify that

$$s'_{k+1} = s'_{k+2} = \cdots = s'_{2k} = s'_1.$$

Since $\delta(s') = \delta(s) \geqslant 2$, this run of at least $k$ consecutive equal symbols, must end. Without loss of generality, assume $0, 1 \in \Sigma$, and (possibly after an appropriate relabeling of the symbol names) either $0^k1$ or $10^k$ form a substring of $s'$. We shall assume the former, and the proof for the latter is similar. We ignore the rest of the symbols, as they will not affect our calculations of the lower bound on capacity. Thus, we may proceed as if the initial string $s$ is $0^k1$.

We now generate more strings by duplicating only substrings of the form $0^k1$ or $0^{k-1}1$. The resulting set of strings contains the regular language

$$S = \left(\left(0^k1\right)^+\left(0^{k-1}1\right)^+\right)^+.$$

We can follow the same steps as in the proof of Theorem 15 in order to find the capacity of $S$. It is given by the base-2 logarithm of the largest real solution for the equation

$$x^{-(k+1)} + x^{-k} = 1.$$

By rearranging, we get the claim. It is also easy to verify that the claimed $r$ satisfies $r > 1$, and so the capacity is strictly positive. ∎

*Remark 19: Let $r$ denote the largest root of $f(x)$ from Theorem 18. We contend that $2^{1/(k+1)} \leqslant r \leqslant 2^{1/k}$ as follows. Since $f(1) < 0$, we find $r > 1$. We have $r^{k+1} = r + 1 \geqslant 2$ and so $r \geqslant 2^{1/(k+1)}$. On the other hand $r^k - 1 = 1/r \leqslant 1$*

*and so $r \leqslant 2^{1/k}$. Thus, under the conditions of Theorem 18, a weaker lower bound is*

$$\mathsf{cap}(S^{\tan}_{\geqslant k}) \geqslant \frac{1}{k+1}.$$

Unlike the case of $S^{\tan}_{k}$, the following shows $S^{\tan}_{\geqslant k}$ is fully diverse.

*Theorem 20: Let $\Sigma$ be a finite alphabet, and $s \in \Sigma^*$ a string such that $R(s) = \Sigma$ and $|s| \geqslant k$. Then $S^{\tan}_{\geqslant k} = S(\Sigma, s, \mathcal{T}^{\tan}_{\geqslant k})$ is fully diverse.*

*Proof:* The proof follows the same path as that of Theorem 15. Let $v \in \Sigma^{\ell}$ be any string, $v = v_1 v_2 \dots v_{\ell}$, with $v_i \in \Sigma$. Denote $s = s_1 s_2 \dots s_n$, and let $i_1, i_2, \dots, i_{\ell}$ be indices (not necessarily distinct) such that $s_{i_j} = v_j$. This is always possible since $R(s) = \Sigma$, i.e., all the letters of the alphabet appear somewhere in $s$.

As a first step we define $w_0 = T^{\tan}_{0,|s|}(s)$, i.e., create $w_0 = ss$ from $s$. We then iteratively create $w_j$ from $w_{j-1}$ in the following way:

$$w_j = T^{\tan}_{i_j - 1, 2|s| - i_j + j}(w_{j-1}),$$

for $j = 1, 2, \dots, \ell$. We note that the first step of creating $ss$ from $s$, and the fact that $|s| \geqslant k$, guarantee that

$$2|s| - i_j + j \geqslant k,$$

for all $j$. Now, one can easily verify that $w_j$, starting from index $2|s| + 1$, contains $v_1 v_2 \dots v_j$. In particular, $w_{\ell}$ contains $v$, and so $S^{\tan}_{\geqslant k}$ is fully diverse. ∎

*Corollary 21: Let $\Sigma$ be a finite alphabet, and $s \in \Sigma^*$ such that $R(s) = \Sigma$, and $|s| \geqslant k$. Then $\mathsf{cap}(S^{\tan}_{\geqslant k}(s))$ does not depend on $s$.*

*Proof:* This is a combination of Theorem 20, Theorem 7, and the fact that $\mathcal{T}^{\tan}_{\geqslant k}$ is extension invariant. ∎

In this section, we studied the capacity of systems $S^{\tan}_{\geqslant k}$. The capacity and diversity of the analogous $S^{\tan}_{\leqslant k}$ appears to be quite more elaborate, and is studied in the recent [8]. Here, we only mention a straightforward upper bound on this capacity. Consider $S^{\tan}_{\leqslant k} = S(\Sigma, s, \mathcal{T}^{\tan}_{\leqslant k})$ with $\Sigma = \{s_1, \dots, s_m\}$ and $s = s_1 \cdots s_m$ (the symbols of $s$ are all distinct). Every sequence $y \in S^{\tan}_{\leqslant k}$ has the following two properties. First, for each $i > k$, there are no occurrence of $s_{i-k}$ that appears after $s_i$ anywhere in $y$. Second, the last occurrence of $s_i$ for $i < m$ is always followed by $s_{i+1}$. Suppose $y \in S^{\tan}_{\leqslant k}$ has length $n$. We can divide $y$ into $m$ substrings where the $i$th substrings starts after the previous one ends and ends at the last occurrence of $s_i$. Note that the $i$th substring starts and ends with $s_i$ and so consists only of elements $s_i, s_{i+1}, \dots, s_{i+k-1}$. There are at most $\binom{n}{m}$ ways of dividing $y$ into such substrings. Hence, the total number of possibilities for $y$ is at most $\binom{n}{m}k^n$, and for any finite alphabet $\Sigma$ and any $s$ as above,

$$\mathsf{cap}(S^{\tan}_{\leqslant k}) \leqslant \log_2 k. \tag{4}$$

## V. Palindromic Duplication

The next type of string-duplication system we consider uses palindromic duplication rules. Even with fixed-length duplication we show this system has positive capacity and full diversity. We also prove that the capacity of this system depends only on the alpha-diversity of the seed string.

Consider the palindromic-duplication rule $T^{\mathrm{pal}}_{i,k} : \Sigma^* \to \Sigma^*$ defined as

$$T^{\mathrm{pal}}_{i,k}(x) = \begin{cases} uvv^R w & \text{if } x = uvw,\ |u| = i,\ |v| = k, \\ x & \text{otherwise,} \end{cases}$$

where $y^R$ is the reverse of $y$, i.e., $y^R = y_m y_{m-1} \cdots y_1$ for a sequence $y = y_1 y_2 \cdots y_m$, $y_j \in \Sigma$. Furthermore, let

$$\mathcal{T}^{\mathrm{pal}}_{k} = \left\{ T^{\mathrm{pal}}_{i,k} \;\middle|\; i \geqslant 0 \right\}.$$

and use $S^{\mathrm{pal}}_{k} = S(\Sigma, s, \mathcal{T}^{\mathrm{pal}}_{k})$. Since the seed string $s$ will play a crucial role, we shall often use the notation $S^{\mathrm{pal}}_{k}(s)$ to emphasize the choice of $s$.

*Lemma 22: Let $s \in \Sigma^k$ such that $s \neq s^R$. Then*

$$\mathsf{cap}(S^{\mathrm{pal}}_{k}(s)) \geqslant \frac{1}{k}.$$

*Proof:* By repeatedly applying duplication to the last block of $k$ symbols, we can create any sequence of alternating blocks $s$ and $s^R$, starting with $s$. To extend any run of $s$, except the first one, (resp. any run of $s^R$) we can apply duplication to the last block of the previous run, which is an $s^R$ block (resp. $s$). Thus, the regular language

$$S = ss^R \left\{ s, s^R \right\}^*,$$

satisfies $S \subseteq S^{\mathrm{pal}}_{k}(s)$. Since $s \neq s^R$, we easily see that

$$\mathsf{cap}(S^{\mathrm{pal}}_{k}(s)) \geqslant \mathsf{cap}(S) = \frac{1}{k}.$$

∎

Note that the requirement that $s \neq s^R$ implies that $k \geqslant 2$.

The following theorem states that the capacity of palindromic duplication is positive except in trivial cases.

*Theorem 23: Let $\Sigma$ be a finite alphabet. For any $s \in \Sigma^*$, $|s| \geqslant k$, we have $\mathsf{cap}(S^{\mathrm{pal}}_{k}(s)) = 0$ if and only if $\delta(s) = 1$ or $k = 1$.*

*Proof:* It is clear that if $\delta(s) = 1$, then $\mathsf{cap}(S^{\mathrm{pal}}_{k}(s)) = 0$. Also, if $k = 1$ then trivially $S^{\mathrm{pal}}_{1} = S^{\tan}_{1}$, and then, by Theorem 12, $\mathsf{cap}(S^{\mathrm{pal}}_{1}(s)) = \mathsf{cap}(S^{\tan}_{1}) = 0$.

For the other direction, suppose that $\mathsf{cap}(S^{\mathrm{pal}}_{k}(s)) = 0$. If $k = 1$ then we are done. Assuming $k > 1$, we show that $\delta(s) = 1$. We first prove this for $|s| = k$.

Denote $s = s_1 s_2 \cdots s_k$, with $s_i \in \Sigma$. Since $\mathsf{cap}(S^{\mathrm{pal}}_{k}(s)) = 0$, by Lemma 22, we must have that $s = s^R$, or equivalently,

$$s_i = s_{k+1-i}, \quad \forall i \in [k]. \tag{5}$$

If $k = 2$, then (5) implies $\delta(s) = 1$ and we are done again. We therefore assume $k \geqslant 3$. Obviously $ss^R \in S^{\mathrm{pal}}_{k}(s)$, so by Lemma 4

$$\mathsf{cap}(S^{\mathrm{pal}}_{k}(ss^R)) \leqslant \mathsf{cap}(S^{\mathrm{pal}}_{k}(s)) = 0.$$

Also, since $s_2 s_3 \cdots s_k s_k$ is a substring of $ss^R$, and $\mathcal{T}^{\mathrm{pal}}_{k}$ is obviously extension invariant, then by Lemma 6 we get

$$\mathsf{cap}(S^{\mathrm{pal}}_{k}(s_2 s_3 \cdots s_k s_k)) \leqslant \mathsf{cap}(S^{\mathrm{pal}}_{k}(ss^R)) \leqslant 0.$$

Since $S_k^{\text{pal}}(s_2 s_3 \cdots s_k s_k))$ is an infinite set, we necessarily have

$$\mathsf{cap}(S_k^{\text{pal}}(s_2 s_3 \cdots s_k s_k)) = 0.$$

Hence,

$$s_2 = s_k, \tag{6}$$
$$s_{i+2} = s_{k+1-i}, \quad \forall i \in [k-2]. \tag{7}$$

From (5) and (7), it follows that

$$s_i = s_{i+2}, \quad \forall i \in [k-2]. \tag{8}$$

It is also true that $s_1 = s_2$ since $s_1 = s_k$ from (5) and $s_2 = s_k$ from (6). The expressions (8) and $s_1 = s_2$ prove that $\delta(s) = 1$.

We now consider the case of $|s| \geqslant k$. If $s'$ is a $k$-substring of $s$, then by Lemma 6

$$\mathsf{cap}(S_k^{\text{pal}}(s')) \leqslant \mathsf{cap}(S_k^{\text{pal}}(s)) = 0.$$

Since $S_k^{\text{pal}}(s')$ is an infinite set, we have $\mathsf{cap}(S_k^{\text{pal}}(s')) = 0$, and using the above proof for length-$k$ strings, we get $\delta(s') = 1$. Since this is true for every $k$-substring $s'$ of $s$, either $k = 1$ or the $k$-substrings overlap and we must have $\delta(s) = 1$. ∎

We now turn to consider the diversity of $S_k^{\text{pal}}(s)$.

*Theorem 24: For any $x, y \in \Sigma^*$, with $|y| \geqslant k$, if $R(x) \subseteq R(y)$, then there exists $u \in \Sigma^*$ such that $ux \in S_k^{\text{pal}}(y)$. In particular, if $R(y) = \Sigma$ then $S_k^{\text{pal}}(y)$ is fully diverse.*

*Proof:* We prove the lemma by explicitly constructing a string of the form $ux$ that belongs to $S_k^{\text{pal}}(y)$. A simple example of this process is given after the proof.

We first observe that we can always apply $T_{i,k}^{\text{pal}}$ to $y$ for any valid value of $i$, to obtain $\overline{y}$ for which $|\overline{y}| \geqslant 2k$. For ease of notation, we therefore assume $|y| \geqslant 2k$. We also assume $|x| > 0$, or else the claim is trivial.

Suppose that the *last* symbol of $x$ is $a$. We construct a sequence $y''$ from $y$ using the functions $\mathcal{T}_k^{\text{pal}}$ such that $a$ is the last symbol of $y''$, i.e., $a$ is "pushed" to the end. Let $i$ be such that $y_i = a$. Consider the conditions

$$i \geqslant k \text{ and } |y| - i \geqslant k - 1.$$

At most one of the two conditions does not hold. If the former does not hold, let $y' = T_{0,k}^{\text{pal}}(y)$. There is a copy of $a$ at position $i' = 2k - i + 1$ in $y'$, i.e., $y'_{i'} = a$. We have $i' \geqslant k$ and $|y'| - i' \geqslant 3k - (2k - i + 1) \geqslant k - 1$. If the latter does not hold, let $y' = T_{i-k,k}^{\text{pal}}(y)$ and $i' = i$. If both conditions hold, let $y' = y$ and $i' = i$. We thus have $y'_{i'} = a$ with $i' \geqslant k$ and $|y'| - i' \geqslant k - 1$. The significance of these conditions is that they enable us to duplicate blocks of length $k$ containing $a$ without the need to concern ourselves with the boundaries of the sequence.

Let $|y'| - i' = q(k-1) + r$ such that $q$ and $r$ are integers with $q \geqslant 1$ and $0 \leqslant r < k - 1$.

First, suppose $k$ is even. We let $y'' = T_{i'-k/2,k}^{\text{pal}}(y')$. Now there is a copy of $a$ in $y''$ at position $i'' = i' + k + 1$. The distance of this copy from the end of $y''$ is

$$|y''| - i'' = |y'| + k - (i' + k + 1) = |y'| - i' - 1.$$

Hence, the distance is decreased by one, compared with $y'$. We repeat the same procedure and update $y''$ and $i''$ as

$$y'' \leftarrow T_{i''-k/2,k}^{\text{pal}}(y''),$$
$$i'' \leftarrow i'' + k + 1,$$

until we have $|y''| - i'' = q(k-1)$, where for simplicity we have used update rules instead of introducing new notation. At this point we switch to repeating

$$y'' \leftarrow T_{i''-1,k}^{\text{pal}}(y''), \tag{9}$$
$$i'' \leftarrow i'' + 2k - 1, \tag{10}$$

until $a$ becomes the last symbol of $y''$.

Next, suppose that $k$ is odd and $r$ is even. We let $y'' = T_{i'-(k-1)/2}^{\text{pal}}(y')$. Now there is a copy of $a$ in $y''$ at position $i'' = i' + k + 2$. The distance of this copy from the end of $y''$ is

$$|y''| - i'' = |y'| + k - (i' + k + 2) = |y'| - i' - 2.$$

The distance is thus decreased by two, compared with $y'$. Since $r$ is even, by repeating the same procedure and updating $y''$ and $i''$, we can have $|y''| - i'' = q(k-1)$. We then repeat (9) and (10) until $a$ becomes the last symbol of $y''$.

Finally, suppose that $k$ and $r$ are both odd. Let $y'' = T_{i-1,k}^{\text{pal}}(y')$. There is a copy of $a$ in $y''$ at position $i'' = i'$. The distance of this copy from the end of $y''$ is $|y''| - i'' = |y'| + k - i$. Let $|y''| - i'' = q'(k-1) + r'$ where $q'$ and $r'$ are integers with $q' \geqslant 1$ and $0 \leqslant r' < k - 1$. We thus have

$$r' = r + 1 + (q + 1 - q')(k-1).$$

Since $k-1$ is even and $r$ is odd, we find that $r'$ is even. We can then proceed as in the previous case in which $k$ is odd and $r$ is even.

We have shown that any symbol present in $y$ can be "pushed" to the end position. If we denote $x = x_1 x_2 \cdots x_\ell$, $x_i \in \Sigma$, then we can push $x_\ell$ to the end position, disregard it momentarily, and apply the procedure above to recursively push $x_1 x_2 \cdots x_{\ell-1}$ just before it. The final result is a sequence in $S_k^{\text{pal}}(y)$ which ends with $x$. ∎

*Example 25: Consider the system $S_k^{\text{pal}}(y)$ over $\Sigma = \{a, b, c, d\}$ with $y = abcbd$, $k = 3$ and let $x = ac$. Our goal is to find a word in $S_k^{\text{pal}}(y)$ that ends with $x$ with the method described in the proof of the preceding lemma. To do this we first push $c$ to the end in $y$. Note that for $i = 3$, we have $y_i = c$ such that $i \geqslant k$ and $|y| - i \geqslant k - 1$. Since these conditions hold, let $y' = y$ and $i' = i$. The value of $r$ in the equation $|y'| - i' = q(k-1) + r$ is thus $r = 0$. Since $r = 0$, we let $y'' = y'$ and $i'' = i' = 3$ and apply (7) by letting $y'' \leftarrow T_{2,3}^{\text{pal}}(y'')$ to get $y'' = abcbddbc$. Now that $c$ is pushed to the end, we omit it, letting $y = abcbddb$, and proceed by pushing $a$ to the end. This process will follow the steps: $\underline{abc}bddb \rightarrow abccb\underline{abd}db \rightarrow abccbabddb\underline{adb} \rightarrow abccbabddbadbbda$, where we have underlined the segment that is duplicated to get the next string. Putting these together, we have obtained a word in $S_k^{\text{pal}}(y)$ that ends with $x = ac$,*

TABLE I
NUMERICAL RESULTS FOR PALINDROMIC DUPLICATION (SEE EXAMPLE 27)

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $s = 01, k = 2$ | $n$ | 2 | 4 | 6 | 8 | 10 | 12 | 14 |
| | $N(n)$ | 1 | 1 | 3 | 10 | 37 | 145 | 584 |
| $s = 010, k = 3$ | $n$ | 3 | 6 | 9 | 12 | 15 | 18 | 21 |
| | $N(n)$ | 1 | 1 | 3 | 14 | 78 | 467 | 2894 |
| $s = 012, k = 3$ | $n$ | 3 | 6 | 9 | 12 | 15 | 18 | 21 |
| | $N(n)$ | 1 | 1 | 4 | 25 | 182 | 1423 | 11577 |

*as follows:*

$$abcbd \to abcbddb\boldsymbol{c} \to abccbabddb\boldsymbol{c} \to abccbabddbadbb\boldsymbol{c} \to abccbabddbadbbd\boldsymbol{ac},$$

*where the symbols that are pushed to the end are in* ***boldface***. □

*Corollary 26: For all $s \in \Sigma^*$, with $|s| \geqslant k$ and $R(s) = \Sigma$, $\mathsf{cap}(S_k^{\mathrm{pal}}(s))$ does not depend on $s$.*

*Proof:* Again, this is a simple combination of Theorem 24, Theorem 7, and the fact that $\mathcal{T}_k^{\mathrm{pal}}$ is extension invariant. ■

Suppose $s$ is a string of length $k$ such that $s = s^R$. We show that for positive integers $p$ and $q$, we have

$$N_{S_k^{\mathrm{pal}}(s)}(pqk) \geqslant \left(N_{S_k^{\mathrm{pal}}(s)}(pk)\right)^q. \tag{11}$$

To see this, note that from $s$ we can derive $s^q$, i.e., $q$ copies of $s$. Then, from each of these copies, independently, we generate a sequence from $S_k^{\mathrm{pal}}(s) \cap \Sigma^{pk}$. The the result is a sequence of length $pqk$ from $S_k^{\mathrm{pal}}(s)$.

It is clear that (11) also holds for the case in which $s$ is equal to a (bijective) relabeling of $s^R$, e.g., $s = 012$.

If we let $q \to \infty$ in (11), we find that

$$\mathsf{cap}\left(S_k^{\mathrm{pal}}(s)\right) \geqslant \frac{\log_2 N_{S_k^{\mathrm{pal}}(s)}(pk)}{pk}. \tag{12}$$

We use this observation in the following example to obtain numerical lower bounds on capacity.

*Example 27: Using a computer, we obtain Table I for the given values of $s$ and $k$. In the table, $N(n)$ denotes the number of sequences of length $n$ in the system $S_k^{\mathrm{pal}}(s)$. We then use* (12) *to find the following lower bounds on the capacity,*

$$\mathsf{cap}\left(S_2^{\mathrm{pal}}(01)\right) \geqslant \frac{\log_2 584}{14} \geqslant 0.65,$$
$$\mathsf{cap}\left(S_3^{\mathrm{pal}}(010)\right) \geqslant \frac{\log_2 2894}{21} \geqslant 0.54,$$
$$\mathsf{cap}\left(S_3^{\mathrm{pal}}(012)\right) \geqslant \frac{\log_2 11577}{21} \geqslant 0.64.$$

□

## VI. INTERSPERSED DUPLICATION

As a final string system, we consider interspersed-duplication. The picture is more complicated in this case. We show that the capacity of the system is sometime 0, and sometimes positive, depending on the periodicity of the seed string. Unlike all previous systems, for this one we can also show cases in which the capacity is positive, but less than full. Finally, we also prove that the system is fully diverse in some cases.

The interspersed-duplication rules $T_{i,k,k'}^{\mathrm{int}} : \Sigma^* \to \Sigma^*$ are defined as

$$T_{i,k,k'}^{\mathrm{int}}(x) = \begin{cases} uvwvz, & \text{if } x = uvwz,\ |u| = i, \\ & |v| = k,\ |w| = k', \\ x, & \text{otherwise.} \end{cases}$$

We let

$$\mathcal{T}_{k,k'}^{\mathrm{int}} = \left\{ T_{i,k,k'}^{\mathrm{int}} \;\middle|\; i \geqslant 0 \right\},$$

and use $S_{k,k'}^{\mathrm{int}} = S(\Sigma, s, \mathcal{T}_{k,k'}^{\mathrm{int}})$, for some $s \in \Sigma^*$. We may also use $S_{k,k'}^{\mathrm{int}}(s)$ to denote the aforementioned string system. To avoid trivialities, throughout this section, we assume $k, k' \geqslant 1$.

For a sequence $s = s_1 s_2 \cdots$, with $s_i \in \Sigma$, we conveniently denote the $k$-substring starting at position $i$ as $s_{i,k} = s_i s_{i+1} \cdots s_{i+k-1}$. Furthermore, for two sequences of equal length, $s, s' \in \Sigma^k$, we denote their Hamming distance as $d_H(s, s')$, which is the number of coordinates in which $s$ and $s'$ disagree.

*Lemma 28: For all $s \in \Sigma^*$ such that $|s| \geqslant k + k'$, we have*

$$\mathsf{cap}(S_{k,k'}^{\mathrm{int}}(s)) \geqslant \frac{1}{k} \log_2 \left(1 + d_H\left(s_{1,k}, (s^2)_{k+1,k}\right)\right).$$

*Proof:* The proof considers two cases: either $k > k'$, or $k \leqslant k'$. We prove the former and state where the proof of the latter is different. It also suffices to consider only $|s| = k + k'$, since for longer strings we can simply ignore the extra symbols.

For simplicity of notation, let $s = x_1 \cdots x_k y_1 \cdots y_{k'}$, where $x_i, y_i \in \Sigma$. We initially apply $T_{0,k,k'}^{\mathrm{int}}$ to $s$ and obtain

$$s' = T_{0,k,k'}^{\mathrm{int}}(s) = x_1 \cdots x_k y_1 \cdots y_{k'} x_1 \cdots x_k.$$

We then apply $T_{i,k,k'}^{\mathrm{int}}$ to $s'$, for all $0 \leqslant i \leqslant k$, and get the following list of results:

$$\begin{array}{l}
x_1 \cdots x_k\ y_1 \cdots y_{k'}\ x_1 \cdots x_k\ x_1 x_2 x_3 \cdots x_{k'} x_{k'+1} x_{k'+2} \cdots x_k \\
x_1 \cdots x_k\ y_1 \cdots y_{k'}\ x_1 \cdots x_k\ y_1 x_2 x_3 \cdots x_{k'} x_{k'+1} x_{k'+2} \cdots x_k \\
x_1 \cdots x_k\ y_1 \cdots y_{k'}\ x_1 \cdots x_k\ y_1 y_2 x_3 \cdots x_{k'} x_{k'+1} x_{k'+2} \cdots x_k \\
\qquad\qquad \vdots \\
x_1 \cdots x_k\ y_1 \cdots y_{k'}\ x_1 \cdots x_k\ y_1 y_2 y_3 \cdots y_{k'} x_{k'+1} x_{k'+2} \cdots x_k \\
x_1 \cdots x_k\ y_1 \cdots y_{k'}\ x_1 \cdots x_k\ y_1 y_2 y_3 \cdots y_{k'}\ x_1\ x_{k'+2} \cdots x_k \\
\qquad\qquad \vdots \\
x_1 \cdots x_k\ y_1 \cdots y_{k'}\ x_1 \cdots x_k\ y_1 y_2 y_3 \cdots y_{k'}\ x_1\ x_2\ \cdots x_{k-k'}
\end{array}$$

where the 6 explicitly stated sequences correspond to $i = 0, 1, 2, k', k' + 1, k$. From these results, it is clear that we have $1 + d_H(s_{1,k}, (s^2)_{k+1,k})$ distinct sequences. Since the same operation can be repeated, i.e., apply $T_{i,k,k'}^{\mathrm{int}}$ to $s'$, for all $0 \leqslant i \leqslant k$, to all the distinct results of the previous round, the number of sequences in $S_{k,k'}^{\mathrm{int}}$ with length $2k + k' + jk$ is at least

$$N_{S_{k,k'}^{\mathrm{int}}}(2k + k' + jk) \geqslant \left(1 + d_H(s_{1,k}, (s^2)_{k+1,k})\right)^j.$$

This completes the proof for $k > k'$.

For $k \leqslant k'$ the proof is similar except that the result of applying $T^{\text{int}}_{i,k,k'}$ to $s'$, for all $0 \leqslant i \leqslant k$, is the following set of sequences:

$$\begin{array}{cccc} x_1\cdots x_k & y_1\cdots y_{k'} & x_1\cdots x_k & x_1x_2x_3\cdots x_k \\ x_1\cdots x_k & y_1\cdots y_{k'} & x_1\cdots x_k & y_1y_2x_3\cdots x_k \\ & & \vdots & \\ x_1\cdots x_k & y_1\cdots y_{k'} & x_1\cdots x_k & y_1y_2y_3\cdots y_k \end{array}$$

where the 3 explicitly stated sequences correspond to $i = 0, 1, k$. ∎

With an example, we show that the lower bound of Lemma 28 is sharp. Choose $s$ as

$$s = a_1\cdots a_k b a_2\cdots a_k,$$

where $b \neq a_1$. Suppose $t \in S^{\text{int}}_{k,k}(s)$. Each $k$-substring $t_{(i-1)k+1,k}$, for non-negative integers $i \leqslant |t|/k$, either equals $a_1\cdots a_k$ or $ba_2\cdots a_k$. Thus for a non-negative integer $j$ there are no more than $2^j$ sequences of length $jk$ in $S^{\text{int}}_{k,k}(s)$. Hence,

$$\mathsf{cap}(S^{\text{int}}_{k,k}(s)) \leqslant \lim_{j\to\infty} \frac{\log_2 2^j}{jk} = \frac{1}{k}$$

which matches the lower bound given in Lemma 28, and so $\mathsf{cap}(S^{\text{int}}_{k,k}(s)) = \frac{1}{k}$.

The next corollary is an immediate result of the previous lemma.

*Corollary 29: Let $\Sigma$ be some finite alphabet, and assume $\mathsf{cap}(S^{\text{int}}_{k,k'}(s)) = 0$, where $s \in \Sigma^*$ and $|s| \geqslant k + k'$. For any $(k + k')$-substring of $s$, denoted $t = x_1\cdots x_k y_1\cdots y_{k'}$, with $x_i, y_i \in \Sigma$, we have $d_H(t_{1,k}, (t^2)_{k+1,k}) = 0$, i.e.,*

$$\begin{aligned} x_1\cdots x_k &= y_1\cdots y_{k'}x_1\cdots x_{k-k'}, \quad \text{if } k > k', \\ x_1\cdots x_k &= y_1\cdots y_k, \quad \text{if } k \leqslant k'. \end{aligned}$$

This corollary is used in the following theorem.

*Theorem 30: For $\Sigma$ a finite alphabet, $s \in \Sigma^*$, $|s| \geqslant k + k'$, we have $\mathsf{cap}(S^{\text{int}}_{k,k'}(s)) = 0$ if and only if $s$ is periodic with period $\gcd(k, k')$.*

*Proof:* We start with the easy direction. Assume $s$ is periodic with period $\gcd(k, k')$. Note that in this case $S^{\text{int}}_{k,k'}(s)$ contains only one sequence of length $ik + k'$ for each $i \geqslant 1$, which is itself a periodic extension of $s$. No other sequence appears in $S^{\text{int}}_{k,k'}(s)$. Thus, the capacity is 0.

We now turn to the other direction. Assume the capacity is 0. We further assume

$$s = x_1\cdots x_k y_1\cdots y_{k'},$$

with $x_i, y_i \in \Sigma$, has length $k + k'$. The general case, of $|s| > k+k'$, will then follow easily. The proof in this direction is divided into two cases.

For the first case, suppose $k > k'$, and denote $k'' = k - k'$. We show that $s$ is periodic with period $\gcd(k, k')$. From Corollary 29, it follows that

$$y_1\cdots y_{k'} = x_1\cdots x_{k'}$$

so we can write

$$s = x_1\cdots x_k x_1\cdots x_{k'}.$$

Furthermore, said corollary implies that $x_i = x_{k'+i}$ for $i \in [k - k']$ and so

$$s = x_1\cdots x_{k'}x_1\cdots x_{k''}x_1\cdots x_{k'}.$$

By once applying the rule of $T^{\text{int}}_{0,k,k'}$ we obtain

$$t = x_1\cdots x_{k'} \quad x_1\cdots x_{k''} \quad x_1\cdots x_{k'} \quad x_1\cdots x_{k'} \quad x_1\cdots x_{k''}.$$

By Lemma 4, $\mathsf{cap}(S^{\text{int}}_{k,k'}(s)) = 0$ implies $\mathsf{cap}(S^{\text{int}}_{k,k'}(t)) = 0$. Now let us apply Corollary 29 to the substring

$$t' = x_1\cdots x_{k''} \quad x_1\cdots x_{k'} \quad x_1\cdots x_{k'}$$

of $t$ to get

$$x_1\cdots x_{k''}x_1\cdots x_{k'} = x_1\cdots x_{k'}x_1\cdots x_{k''}.$$

That is, the sequence $x_1\cdots x_{k''}x_1\cdots x_{k'}$, which has length $k$, equals itself when cyclically shifted by $k'$. Hence, it is periodic with period $\gcd(k, k')$ and thus $s$ is periodic with the same period.

For the second case, assume $k \leqslant k'$. Denote $x = x_1\cdots x_k$ and $y = y_1\cdots y_{k'}$, so $s = xy$. Find integers $q$ and $r$ such that $k' = qk + r$ and $0 \leqslant r < k$ and let $t$ be the sequence obtained from $s$ by $q + 1$ times applying $T^{\text{int}}_{0,k,k'}$, that is,

$$\begin{aligned} t &= xyx^{q+1} \\ &= x_{1,k}\, y_{1,k}\, y_{k+1,k}\cdots y_{(q-1)k+1,k}\, y_{qk+1,r}\, (x_{1,k})^{q+1}. \end{aligned}$$

Again, by Lemma 4 we have $\mathsf{cap}(S^{\text{int}}_{k,k'}(t)) = 0$. Thus, we can apply Corollary 29 to any $(k + k')$-substring of $t$. For $i = 0, 1, \ldots, q-1$, in that order, applying Corollary 29 to the $(k + k')$-substring $t_{ik+1,k+k'}$ implies that

$$x_{1,k} = y_{ik+1,k}. \tag{13}$$

Next, for the $(k + k')$-substring $t_{qk+1,k+k'}$, we have

$$\begin{aligned} t_{qk+1,k+k'} &= y_{(q-1)k+1,k}\, y_{qk+1,r}\, (x_{1,k})^q \\ &= x_{1,k}\, y_{qk+1,r}\, (x_{1,k})^q, \end{aligned}$$

where the second equality follows from (13). By applying Corollary 29 to this sequence, we find

$$t_{qk+1,k+k'} = x_{1,k}x_{1,r}\,(x_{1,k})^q.$$

Thus, we have

$$t = (x_{1,k})^{q+1}(x_{1,r})(x_{1,k})^{q+1}.$$

Finally, we apply Corollary 29 to the $(k + k')$-substring

$$t_{qk+r+1,k+k'} = x_{r+1}\cdots x_k\, x_1\cdots x_r\, (x_1\cdots x_k)^q\, x_1\cdots x_r$$

which shows that

$$x_{r+1}\cdots x_k\, x_1\cdots x_r = x_1\cdots x_k.$$

Since $x_1\cdots x_k$ equals itself when cyclically shifted by $r$, it is periodic with period $\gcd(k, r) = \gcd(k, k')$. Hence $t$ is periodic with the same period and so is $s$.

We have shown that for the special case of $|s| = k + k'$, if the capacity is zero, then $s$ is periodic with period $\gcd(k, k')$. Now suppose $|s| > k + k'$ and that $\mathsf{cap}(S^{\text{int}}_{k,k'}(s)) = 0$.

Let $d = \gcd(k, k')$ and, for the moment, also suppose that $d$ divides $|s|$. Let

$$C = \left\{ s_{id+1,k+k'} \;\middle|\; 0 \leqslant i \leqslant \frac{|s| - (k+k')}{d} \right\}$$

be a set of $(k + k')$-substrings of $s$ that cover $s$ and each consecutive pair overlap in at least $d$ positions. Since the capacity for each of these $(k + k')$-substrings is also zero, they are periodic with period $d$. Because of their overlaps and the fact that they cover $s$, it follows that $s$ is also periodic with period $d$. To complete the proof it remains to consider the case in which $d$ does not divide $|s|$. In this case, we can repeat the same argument but with adding the substring $s_{|s|-(k+k')+1,(k+k')}$ to the set $C$ to ensure that $s$ is covered by overlapping $(k + k')$-substrings. ∎

We now turn to discuss the dependence of $\mathsf{cap}(S^{\text{int}}_{k,k'}(s))$ on $s$. For a sequence $x \in \Sigma^*$, and two symbols $a, b \in R(x)$, let

$$\Delta_x(a,b) = \left\{ j \in \mathbb{Z} \;\middle|\; \exists i, x_i = a, x_{i+j} = b \right\},$$

be the set of the differences of positions of $a$ and $b$ in $x$. Furthermore, let

$$\rho_{x,\ell}(a,b) = \{(j \bmod \ell) \mid j \in \Delta_x(a,b)\}.$$

*Lemma 31: Let $\Sigma$ be some finite alphabet, $d > 0$ an integer, and $D \subset \{0, 1, \ldots, d-1\}$ some subset, $|D| < d$. Consider the constrained system $S \subseteq \Sigma^*$ such that for every $x \in S$, and every two symbols $a, b \in \Sigma$ (not necessarily distinct), $\rho_{x,d}(a,b) \subseteq D$. Then $\mathsf{cap}(S) < \log_2 |\Sigma|$.*

*Proof:* We begin by constructing a De-Bruijn graph of order $d + 1$ over $\Sigma$, $\mathcal{G}''(V'', E'')$, defined in the following way. We set $V'' = \Sigma^{d+1}$, and a directed edge connects $v = v_1 \cdots v_{d+1} \in V''$ and $v' = v'_1 \cdots v'_{d+1} \in V''$, if $v'_i = v_{i+1}$ for all $i \in [d]$. That edge has label $v'_{d+1} \in \Sigma$. The graph is regular with out-degree $|\Sigma|$. Clearly the set of finite strings read along paths taken in $\mathcal{G}''$ is simply $S'' = \Sigma^*$. In particular, by Perron-Frobenius theory, if $A_{\mathcal{G}''}$ is the adjacency matrix of $\mathcal{G}''$, since $\mathcal{G}''$ is clearly deterministic (and therefore lossless) and primitive,

$$\mathsf{cap}(S'') = \log_2 \lambda(A_{\mathcal{G}''}) = \log_2 |\Sigma|.$$

As the next step, we construct a graph $\mathcal{G}'(V', E')$ from $\mathcal{G}''(V'', E'')$ by setting $V' = V''$, and removing all edges $v \to u$, such that

$$\rho_{v,d}(a,b) \cup \rho_{u,d}(a,b) \not\subseteq D,$$

for some $a, b \in \Sigma$. The labels of the surviving edges remain the same. We define $S'$ to be the set of strings read from finite paths in $\mathcal{G}'$. Since $|D| < d$, $A_{\mathcal{G}'}$ is obtained from $A_{\mathcal{G}''}$ by changing at least one entry from 1 to 0. By Perron-Frobenius theory,

$$\mathsf{cap}(S') \leqslant \log_2 \lambda(A_{\mathcal{G}'}) < \log_2 \lambda(A_{\mathcal{G}''}) = \log_2 |\Sigma|.$$

Finally, since it is clear that $S \subseteq S'$, we get

$$\mathsf{cap}(S) \leqslant \mathsf{cap}(S') < \log_2 |\Sigma|,$$

as claimed. ∎

Using Lemma 31 we obtain the following theorem.

TABLE II
STRING-DUPLICATION SYSTEMS AND THEIR PROPERTIES

| Type | System | Capacity: Zero | Capacity: Partial | Capacity: Full | Full Diversity |
|---|---|---|---|---|---|
| End | $S^{\text{end}}_{k}$ | – | – | ✓ | ✓ |
| | $S^{\text{end}}_{\geqslant k}$ | – | – | ✓ | ✓ |
| Tandem | $S^{\text{tan}}_{k}$ | ✓ | – | – | – |
| | $S^{\text{tan}}_{\geqslant k}$ | – | ? | ? | ✓ |
| Palindromic | $S^{\text{pal}}_{k}$ | – | ? | ? | ✓ |
| Interspersed | $S^{\text{int}}_{k,k'}$ | ✓ | ✓ | ? | ✓ |

Legend:
– No system exists
✓ A system exists
? Unknown

*Theorem 32: Let $\Sigma$ be a finite alphabet, $s \in \Sigma^*$ have length at least $k + k'$, and denote $d = \gcd(k, k')$. If, for some $a, b \in R(s)$, we have $\left|\rho_{s,d}(a,b)\right| < d$ then $\mathsf{cap}(S^{\text{int}}_{k,k'}(s)) < \log_2 \delta(s)$.*

*Proof:* We observe that for any $x, x' \in S^{\text{int}}_{k,k'}(s)$, and for $a, b \in R(s)$, we have

$$\rho_{x,d}(a,b) = \rho_{x',d}(a,b).$$

This can be easily seen by noting that any function in $\mathcal{T}^{\text{int}}_{k,k'}$ changes the differences between positions of two elements by a linear combination (with integer coefficients) of $k$ and $k'$. We then apply Lemma 31. ∎

*Theorem 33: For a finite alphabet $\Sigma$, $s \in \Sigma^*$, $R(s) = \Sigma$, $|s| \geqslant k + k'$, and $\gcd(k, k') = 1$, the system $S^{\text{int}}_{k,k'} = S(\Sigma, s, \mathcal{T}^{\text{int}}_{k,k'})$ is fully diverse.*

*Proof:* The proof is similar to that of Theorem 24. In that light, it suffices to show that in a sequence $y \in \Sigma^*$ of length $m \geqslant k + k'$, a symbol $a \in R(y)$ can be "pushed" to the end. That is, we can find a sequence $y'' \in S^{\text{int}}_{k,k'}(y)$ that ends with $a$.

Suppose $a$ is in position $i$ in $y$. Without loss of generality (similar to Theorem 24), we may assume $i > k$ and $m - i \geqslant k' - 1$.

Let $y' = T^{\text{int}}_{i-1,k,k'}(y)$. There is a copy of $a$ at position $i' = i$ whose distance from the end of $y'$ is $\left|y'\right| - i' = k + m - i$ and this is an increase of size $k$ compared to $y$. We update $y'$ as $y' \leftarrow T^{\text{int}}_{i'-1,k,k'}(y')$. In each step, the distance of $a$ at position $i'$ from the end of $y'$ increases by $k$. We continue until we have $k' \mid \left|y'\right| - i'$. This eventually happens as $\gcd(k, k') = 1$.

Now we let $y'' = T^{\text{int}}_{i'-k,k,k'}(y')$. There is a copy of $a$ in $y''$ at position $i'' = i' + k + k'$. The distance of this copy of $a$ from the end of $y''$ is $\left|y''\right| - i'' = \left|y'\right| - i' - k'$. Thus the distance is decreased by $k'$. We update $y''$ and $i''$ as $y'' \leftarrow T^{\text{int}}_{i''-k,k,k'}(y'')$ and $i'' \leftarrow i'' + k + k'$. We continue until $a$ is the last element of $y''$. The rest of the argument follows along the same lines as those of Theorem 24. ∎

*Corollary 34: For a finite alphabet $\Sigma$, and $s \in \Sigma^*$ with $|s| \geqslant k + k'$ and $R(s) = \Sigma$, if $\gcd(k, k') = 1$, then $\mathsf{cap}(S^{\text{int}}_{k,k'}(s))$ does not depend on $s$.*

*Proof:* Since $\mathcal{T}^{\text{int}}_{k,k'}$ is extension invariant, by Theorem 33 and Theorem 7 the result follows. ∎

## VII. Conclusion

Motivated by the prevalence of repeats in biological sequences and genomic-duplication processes, we studied the capacity of several fundamental string-duplication systems. Generally, the capacity of these systems is zero only under very restrictive conditions and they can otherwise generate an exponential number of strings. Furthermore, in many cases, these systems can generate any string as a substring. An overview highlighting the main results concerning non-trivial systems is given in Table II. Specifically, we do not consider systems with $\delta(s) = 1$, and palindromic systems with $k = 1$. Additionally, partial capacity denotes any capacity falling within the open interval $(0, \log_2 |\Sigma|)$. These results suggest a complex behavior of such systems, as well as the possibility of duplication in genome being a significant source of genomic diversity.

Some open questions remain, the most notable of which is to find non-trivial upper bounds on the capacities of these systems. Another interesting question concerns the problem of finding the exact capacity, either in a closed-form expression or algorithmically.

Other open question involve extension of this model, in order to match it with the complexity of the real world. One such extension introduces probability into the model. The capacity in that case becomes hard to compute, and is upper bounded by the combinatorial capacity introduced in this paper. Steps towards solving this problem are taken in [5].

As a final note, we observe that there is a qualitative difference between the proofs for full diversity of $S_k^{\text{end}}$, $S_{\geqslant k}^{\text{tan}}$, $S_k^{\text{pal}}$, and $S_{k,k'}^{\text{int}}$. Assume we have some $s \in \Sigma^*$, and we are looking for $u, w \in \Sigma^*$ such that $usw$ is in the string system. In the case $S_k^{\text{end}}$, the full-diversity proof constructs $usw$ of length $|usw| = \Theta(|s|)$. However, in the other cases, the proofs only give $|usw| = \Theta(|s|^2)$. We do not know whether this is an inherent property of these strings systems, or whether it is an artifact of our proof method. As a motivation to study this we also observe that if there exists a constant $c$ such that for any $s \in \Sigma^*$ there exist $u, w \in \Sigma^*$ with $usw \in S$ in the string system and $|usw| \leqslant c\,|s|$, then

$$\sum_{i=|s|}^{c|s|} (i - |s| + 1) N_S(i) \geqslant |\Sigma|^{|s|}.$$

This in turn implies that for some $|s| \leqslant i \leqslant c\,|s|$,

$$(c\,|s|)^2 N_S(i) \geqslant |\Sigma|^{|s|},$$

and therefore

$$\mathsf{cap}(S) \geqslant \frac{1}{c} \log_2 |\Sigma|. \tag{14}$$

## Acknowledgements

The authors wish to thank the anonymous reviewers, whose comments helped improve the presentation of this paper. In particular, parts of the analysis in Remark 16 and Remark 19 were suggested by one of the anonymous reviewers, as were (4) and (14).

## References

[1] J. Dassow, "Numerical parameters of evolutionary grammars," in *Jewels Are Forever*, J. Karhumäki, H. Maurer, G. Păun, and G. Rozenberg, Eds. Berlin, Germany: Springer, 1999, pp. 171–181.

[2] J. Dassow and V. Mitrana, "Evolutionary grammars: A grammatical model for genome evolution," in *Bioinformatics*, vol. 1278. Berlin, Germany: Springer, 1997, pp. 199–209.

[3] J. Dassow, V. Mitrana, and G. Păun, "On the regularity of duplication closure," *Bull. EATCS*, vol. 69, pp. 133–136, Oct. 1999.

[4] J. Dassow, V. Mitrana, and A. Salomaa, "Operations and language generating devices suggested by the genome evolution," *Theoretical Comput. Sci.*, vol. 270, nos. 1–2, pp. 701–738, 2002.

[5] F. Farnoud, M. Schwartz, and J. Bruck, "A stochastic model for genomic interspersed duplication," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Hong Kong, Jun. 2015, pp. 904–908.

[6] J. W. Fondon, III, and H. R. Garner, "Molecular origins of rapid and continuous morphological evolution," *Proc. Nat. Acad. Sci. USA*, vol. 101, no. 52, pp. 18058–18063, 2004.

[7] K. A. S. Immink, *Codes for Mass Data Storage Systems*. Eindhoven, The Netherlands: Shannon Foundation Publishers, 2004.

[8] S. Jain, F. Farnoud, and J. Bruck, "Capacity and expressiveness of genomic tandem duplication," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Hong Kong, Jun. 2015, pp. 1946–1950.

[9] E. S. Lander *et al.*, "Initial sequencing and analysis of the human genome," *Nature*, vol. 409, no. 6822, pp. 860–921, 2001.

[10] P. Leupold, C. Martín-Vide, and V. Mitrana, "Uniformly bounded duplication languages," *Discrete Appl. Math.*, vol. 146, no. 3, pp. 301–310, 2005.

[11] P. Leupold, V. Mitrana, and J. M. Sempere, "Formal languages arising from gene repeated duplication," in *Aspects of Molecular Computing*. Berlin, Germany: Springer, 2004, pp. 297–308.

[12] G. Levinson and G. A. Gutman, "Slipped-strand mispairing: A major mechanism for DNA sequence evolution," *Molecular Biol. Evol.*, vol. 4, no. 3, pp. 203–221, 1987.

[13] D. M. Lilley, "The inverted repeat as a recognizable structural feature in supercoiled DNA molecules," *Proc. Nat. Acad. Sci. USA*, vol. 77, no. 11, pp. 6468–6472, Nov. 1980.

[14] D. Lind and B. Marcus, *An Introduction to Symbolic Dynamics and Coding*. Cambridge, U.K.: Cambridge Univ. Press, 1985.

[15] O. Milenkovic, "Constrained coding for context-free languages with applications to genetic sequence modelling," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Nice, France, Jun. 2007, pp. 1686–1690.

[16] N. I. Mundy and A. J. Helbig, "Origin and evolution of tandem repeats in the mitochondrial DNA control region of shrikes (*Lanius* spp.)," *J. Molecular Evol.*, vol. 59, no. 2, pp. 250–257, 2004.

[17] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, Jul. 1948.

[18] K. Usdin, "The biological effects of simple tandem repeats: Lessons from the repeat expansion diseases," *Genome Res.*, vol. 18, no. 7, pp. 1011–1019, 2008.

**Farzad Farnoud (Hassanzadeh)** is a postdoctoral scholar at the California Institute of Technology. He received his MS degree in Electrical and Computer Engineering from the University of Toronto in 2008. From the University of Illinois at Urbana- Champaign, he received his MS degree in mathematics and his PhD in Electrical and Computer Engineering in 2012 and 2013, respectively. His research interests include the information-theoretic and algorithmic analysis of genomic evolutionary processes, ranking-based information processing, and coding for flash memory. He is a recipient of the Robert T. Chien Memorial Award for demonstrating excellence in research in electrical engineering from the University of Illinois at Urbana-Champaign.

**Moshe Schwartz** (M'03–SM'10) is an associate professor at the Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Israel. His research interests include algebraic coding, combinatorial structures, and digital sequences.

Dr. Schwartz received the B.A. (summa cum laude), M.Sc., and Ph.D. degrees from the Technion - Israel Institute of Technology, Haifa, Israel, in 1997, 1998, and 2004 respectively, all from the Computer Science Department. He was a Fulbright post-doctoral researcher in the Department of Electrical and Computer Engineering, University of California San Diego, and a post-doctoral researcher in the Department of Electrical Engineering, California Institute of Technology. While on sabbatical 2012-2014, he was a visiting scientist at the Massachusetts Institute of Technology (MIT).

Dr. Schwartz received the 2009 IEEE Communications Society Best Paper Award in Signal Processing and Coding for Data Storage, and the 2010 IEEE Communications Society Best Student Paper Award in Signal Processing and Coding for Data Storage.

**Jehoshua Bruck** (S'86–M'89–SM'93–F'01) is the Gordon and Betty Moore Professor of computation and neural systems and electrical engineering at the California Institute of Technology (Caltech). His current research interests include information theory and systems and the theory of computation in nature.

Dr. Bruck received the B.Sc. and M.Sc. degrees in electrical engineering from the Technion-Israel Institute of Technology, in 1982 and 1985, respectively, and the Ph.D. degree in electrical engineering from Stanford University, in 1989. His industrial and entrepreneurial experiences include working with IBM Research where he participated in the design and implementation of the first IBM parallel computer; cofounding and serving as Chairman of Rainfinity (acquired in 2005 by EMC), a spin-off company from Caltech that created the first virtualization solution for Network Attached Storage; as well as cofounding and serving as Chairman of XtremIO (acquired in 2012 by EMC), a start-up company that created the first scalable all-flash enterprise storage system.

Dr. Bruck is a recipient of the Feynman Prize for Excellence in Teaching, the Sloan Research Fellowship, the National Science Foundation Young Investigator Award, the IBM Outstanding Innovation Award and the IBM Outstanding Technical Achievement Award.