Duplication-Correcting Codes for Data Storage in the DNA of Living Organisms

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

***Abstract*—The ability to store data in the DNA of a living organism has applications in a variety of areas including synthetic biology and watermarking of patented genetically-modified organisms. Data stored in this medium is subject to errors arising from various mutations, such as point mutations, indels, and tandem duplication, which need to be corrected to maintain data integrity. In this paper, we provide error-correcting codes for errors caused by tandem duplications, which create a copy of a block of the sequence and insert it in a tandem manner, i.e., next to the original. In particular, we present two families of codes for correcting errors due to tandem-duplications of a fixed length; the first family can correct any number of errors while the second corrects a bounded number of errors. We also study codes for correcting tandem duplications of length up to a given constant k, where we are primarily focused on the cases of $k = 2, 3$.**

I. Introduction

Data storage in the DNA of living organisms (henceforth *live DNA*) has a multitude of applications. For example, it can enable in vivo synthetic biology methods and algorithms that need "memory," e.g., to store information about their state or record changes in the environment. Embedding data in live DNA also allows watermarking genetically modified organisms (GMOs) to verify authenticity and to track unauthorized use [1], [8], [17], as well as, labeling organisms in biological studies [22]. DNA watermarking can also be used to tag infectious agents used in research laboratories to identify sources of potential malicious use or accidental release [11]. Furthermore, live DNA can serve as a protected medium for storing large amounts of data in a compact format for long periods of time [2], [22]. An additional advantage of using DNA as a medium is that data can be disguised as part of the organisms original DNA, thus providing a layer of secrecy [3].

While the host organism provide a level of protection to the data-carrying DNA molecules as well as a method for replication, the integrity of the stored information suffers from mutations such as tandem duplications, point mutations, insertions, and deletions. Furthermore, since each DNA replication may introduce new mutations, the number of such deleterious events increases with the number of generations. As a result, to ensure decodability of the stored information, the coding/decoding scheme must be capable of a level of error correction. Motivated by this problem, we study designing codes that can correct errors arising from tandem duplications. In addition to improving the reliability of data storage in live DNA, studying such codes may help to acquire a better understanding of how DNA stores and protects biological information in nature.

Tandem duplication is the process of inserting a copy of a segment of the DNA adjacent to its original position, resulting in a *tandem repeat*. A process that may lead to a tandem duplication is *slipped-strand mispairings* [18] during DNA replication, where one strand in a DNA duplex is displaced and misaligned with the other. Tandem repeats constitute about 3% of the human genome [13] and may cause important phenomena such as chromosome fragility, expansion diseases, silencing genes [21], and rapid morphological variation [6].

Different approaches to the problem of error-control for data stored in live DNA have been proposed in the literature. In the work of Arita and Ohashi [1], each group of five bits of information is followed by one parity bit for error detection. Heider and Barnekow [8] use the extended (8,4) Hamming code or repetition coding to protect the data. Yachie et al. [23] propose to enhance reliability by inserting multiple copies of the data into multiple regions of the genome of the host organism. Finally, Haughton and Balado [7] present an encoding method satisfying certain biological constraints, which is studied in a substitution mutation model. None of the aforementioned encodings, with the possible exception of repetition coding, are designed to combat tandem duplications, which is the focus of this paper. While repetition coding can correct duplication errors, it is not an efficient method because of its high redundancy.

It should also be noted that error-control for storage in live DNA is inherently different from that in DNA that is stored

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

outside of a living organism (see [24] for an overview), since the latter is not concerned with errors arising during organic DNA replication.

In this work, we ignore the potential biological effects of embedding data into the DNA. Furthermore, constructing codes that in addition to tandem duplication errors can combat other types of errors, such as substitutions, are postponed to a future work.

We also note that tandem duplication, as well as other duplication mechanisms, were studied in the context of information theory [4], [5], [10]. However, these works used duplications as a generative process, and attempted to measure its capacity and diversity. In contrast, we consider duplications as a noise source, and design error-correcting codes to combat it.

When a sequence has been corrupted by a tandem duplication channel, the challenge arises in finding the squarefree *root* sequence from which the corrupted sequence could be generated. For example, for the sequence $AC\underline{GTGT}$, with $GTGT$ as a tandem duplication error, the root sequence would be $ACGT$ since $ACGTGT$ can be generated from $ACGT$ by doing a tandem duplication of length 2 on GT. But there can be sequences which have more than one root. For example, the sequence $\underline{ACGCACGCG}$ can be generated from **ACG** by doing a tandem duplication of CG first, followed by a tandem duplication of $ACGC$ in $A\underline{CGCG}$ and can also be generated from **ACGCACG** by doing a tandem duplication of the suffix CG. Hence, $ACGCACGCG$ has two squarefree roots. But if we restrict the length of duplication to 2 in the previous example, then $ACGCACGCG$ has only one root i.e., $ACGCACG$. This means that the number of roots that a sequence can have depends on the set of duplication lengths that are allowed. In fact, we find that tandem duplication channels which have unique roots are the ones that allow duplications of fixed length k and the other which allow duplications of lengths bounded by 2 or 3. For all other cases, we prove in Section V, that the duplication roots are not necessarily unique. This unique root property for fixed length, 2- bounded and 3- bounded duplication channels allows us to construct error correcting codes for them.

We will first consider the tandem duplication channel with duplications of a fixed length k. For example with $k = 3$, after a tandem duplication, the sequence $ACAGT$ may become $ACAG\underline{CAG}T$, which may further become $ACA\underline{ACA}GCAGT$ where the copy is underlined. In our analysis, we provide a mapping in which tandem duplications of length k are equivalent to insertion of k zeros. Using this mapping, we demonstrate the strong connection between codes that correct duplications of a fixed length and Run-Length Limited (RLL) systems. We present constructions for codes that can correct an unbounded number of tandem duplications of a fixed length and show that our construction is optimal, i.e., of largest size.

We then turn our attention to codes that correct t tandem duplications (as opposed to an unbounded number of duplications), and show that these codes are closely related to constant-weight codes in the ℓ_1 metric.

Finally, we consider codes for correcting duplications of bounded length. Here, our focus will be on duplication errors of length at most 2 or 3, for which we will present a construction that corrects any number of such errors. In the case of duplication length at most 2 the codes we present are optimal.

The paper is organized as follows. The preliminaries and notation are described in Section II. In Sections III and IV we present the results concerning duplications of a fixed length k and duplications of length at most k, respectively. In Section V, we characterize tandem duplication channels which do not necessarily have a unique root.

II. Preliminaries

We let Σ denote some finite alphabet, and Σ^* denote the set of all finite strings (words) over Σ. The unique empty word is denoted by ϵ. Given two words $x, y \in \Sigma^*$, their concatenation is denoted by xy, and x^t denotes the concatenation of t copies of x, where t is some positive integer. By convention, $x^0 = \epsilon$. We normally index the letters of a word starting with 1, i.e., $x = x_1x_2\dots x_n$, with $x_i \in \Sigma$. With this notation, the t-prefix and t-suffix of x are defined by

$$\mathrm{Pref}_t(x) = x_1x_2\dots x_t,$$
$$\mathrm{Suff}_t(x) = x_{n-t+1}x_{n-t+2}\dots x_n.$$

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, $T_{i,k} : \Sigma^* \to \Sigma^*$, as

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

Two specific sets of duplication rules would be of interest to us throughout the paper.

$$\mathcal{T}_k = \left\{ T_{i,k} \;\middle|\; i \geqslant 0 \right\},$$
$$\mathcal{T}_{\leqslant k} = \left\{ T_{i,k'} \;\middle|\; i \geqslant 0, 1 \leqslant k' \leqslant k \right\}.$$

Given $x, y \in \Sigma^*$, if there exist i and k such that

$$y = T_{i,k}(x),$$

then we say y is a direct descendant of x, and denote it by

$$x \underset{k}{\Longrightarrow} y.$$

If a sequence of t tandem duplications of length k is employed to reach y from x we say y is a t-descendant of x and denote it by

$$x \overset{t}{\underset{k}{\Longrightarrow}} y.$$

More precisely, we require the existence of t non-negative integers $i_1, i_2, \dots, i_t$, such that

$$y = T_{i_t,k}(T_{i_{t-1},k}(\dots T_{i_1,k}(x)\dots)).$$

Finally, if there exists a finite sequence of tandem duplications of length k transforming x into y, we say y is a descendant of x and denote it by

$$x \overset{*}{\underset{k}{\Longrightarrow}} y.$$

We note that x is its own descendant via an empty sequence of tandem duplications.

Example 1. *Let* $\Sigma = \{0,1,2,3\}$ *and* $x = 02123$. *Since,* $T_{1,2}(x) = 0212123$ *and* $T_{0,2}(0212123) = 020212123$, *the following hold*

$$02123 \underset{2}{\Longrightarrow} 0212123, \qquad 02123 \overset{2}{\underset{2}{\Longrightarrow}} 020212123,$$

where in both expressions, the relation could be replaced with $\Longrightarrow_2^*$. □

We define the *descendant cone* of x as

$$D_k^*(x) = \left\{ y \in \Sigma^* \;\middle|\; x \overset{*}{\underset{k}{\Longrightarrow}} y \right\}.$$

In a similar fashion we define the *t-descendant cone* $D_k^t(x)$ by replacing $\Longrightarrow_k^*$ with $\Longrightarrow_k^t$ in the definition of $D_k^*(x)$.

The set of definitions given thus far was focused on tandem-duplication rules of substrings of length exactly k, i.e., for rules from $\mathcal{T}_k$. These definitions as well as others in this section are extended in the natural way for tandem duplication rules of length up to k, i.e., $\mathcal{T}_{\leqslant k}$. We denote these extensions by replacing the k subscript with the $\leqslant k$ subscript. Thus, we also have $D_{\leqslant k}^*(x)$ and $D_{\leqslant k}^t(x)$.

Example 2. *Consider* $\Sigma = \{0,1\}$ *and* $x = 01$. *It is not difficult to see that*

$$\begin{aligned} D_1^2(x) &= \{0001, 0011, 0111\}, \\ D_1^*(x) &= \left\{ 0^i 1^j \;\middle|\; i,j \in \mathbb{N} \right\}, \\ D_2^*(x) &= \left\{ (01)^i \;\middle|\; i \in \mathbb{N} \right\}, \\ D_{\leqslant 2}^*(x) &= \{0s1 \mid s \in \Sigma^*\}. \end{aligned}$$

□

Using the notation D_k^*, we restate the definition of the *tandem string-duplication system* given in [4]. Given a finite alphabet Σ, a seed string $s \in \Sigma^*$, the tandem string-duplication system is given by

$$S_k = S(\Sigma, s, \mathcal{T}_k) = D_k^*(s),$$

i.e., it is the set of all the descendants of s under tandem duplication of length k.

The process of tandem duplication can be naturally reversed. Given a string $y \in \Sigma^*$, for any positive integer, $t > 0$, we define the *t-ancestor cone* as

$$D_k^{-t}(y) = \left\{ x \in \Sigma^* \;\middle|\; x \overset{t}{\underset{k}{\Longrightarrow}} y \right\},$$

or in other words, the set of all words for which y is a t-descendant.

Yet another way of viewing the t-ancestor cone is by defining the *tandem-deduplication rules*, $T_{i,k}^{-1} : \Sigma^* \to \Sigma^*$, as

$$T_{i,k}^{-1}(y) = \begin{cases} uvw & \text{if } y = uvvw,\ |u| = i,\ |v| = k \\ \epsilon & \text{otherwise,} \end{cases}$$

where we recall ϵ denotes the empty word. This operation takes an adjacently-repeated substring of length k, and removes one of its copies. Thus, a string x is in the t-ancestor cone of y (where we assume $x, y \neq \epsilon$ to avoid trivialities) iff there is a sequence of of t non-negative integers $i_1, i_2, \dots, i_t$, such that

$$x = T_{i_t,k}^{-1}(T_{i_{t-1},k}^{-1}(\dots T_{i_1,k}^{-1}(y)\dots)).$$

In a similar fashion we define the *ancestor cone* of y as

$$D_k^{-*}(y) = \left\{ x \in \Sigma^* \;\middle|\; x \overset{*}{\underset{k}{\Longrightarrow}} y \right\}.$$

By flipping the direction of the derivation arrow, we let $\Longleftarrow$ denote deduplication. Thus, if y may be deduplicated to obtain x in a single step we write

$$y \underset{k}{\Longleftarrow} x.$$

For multiple steps we add $*$ in superscript.

Example 3. *We have*

$$0212123 \underset{2}{\Longleftarrow} 02123, \qquad 020212123 \overset{2}{\underset{2}{\Longleftarrow}} 02123,$$

and

$$\begin{aligned} &D_2^{-*}(020212123) \\ &\qquad = \{020212123, 0212123, 0202123, 02123\}. \end{aligned}$$

□

A word $y \in \Sigma^*$ is said to be *irreducible* if there is nothing to deduplicate in it, i.e., y is its only ancestor, meaning

$$D_k^{-*}(y) = \{y\}.$$

The set of irreducible words is denoted by Irr_k. We will find it useful to denote the set of irreducible words of length n by

$$\mathrm{Irr}_k(n) = \mathrm{Irr}_k \cap \Sigma^n.$$

The ancestors of $y \in \Sigma^*$ that cannot be further deduplicated, are called the *roots* of y, and are denoted by

$$R_k(y) = D_k^{-*}(y) \cap \mathrm{Irr}_k.$$

Note that since the aforementioned definitions extend to tandem duplication rules of length up to k, we also have $S_{\leqslant k}$, $D_{\leqslant k}^{-t}(y)$, $D_{\leqslant k}^{-*}(y)$, $\mathrm{Irr}_{\leqslant k}$, $\mathrm{Irr}_{\leqslant k}(n)$, and $R_{\leqslant k}(y)$. In some previous works (e.g., [15]), S_k is called the *uniform-bounded-duplication system*, whereas $S_{\leqslant k}$ is called the *bounded-duplication system*.

Example 4. *For the binary alphabet* $\Sigma = \{0,1\}$,

$$\mathrm{Irr}_{\leqslant 2} = \{0, 1, 01, 10, 010, 101\},$$

and for any alphabet that contains $\{0,1,2,3\}$,

$$R_2(020212123) = \{02123\}\,,$$
$$R_{\leqslant 4}(012101212) = \{012, 0121012\}\,.$$

□

Inspired by the DNA-storage scenario, we now define error-correcting codes for tandem string-duplication systems.

Definition 5. *An* $(n, M; t)_k$ *code* C *for the* k*-tandem-duplication channel is a subset* $C \in \Sigma^n$ *of size* $|C| = M$*, such that for each* $x, y \in C$, $x \neq y$,

$$D_k^t(x) \cap D_k^t(y) = \emptyset.$$

Here t *stands for either a non-negative integer, or* $*$*. In the former case we say the code can correct* t *errors, whereas in the latter case we say the code can correct all errors. In a similar fashion, we can define an* $(n, M; t)_{\leqslant k}$ *by replacing all "k" subscripts by "*$\leqslant k$*".*

Assume the size of the finite alphabet is $|\Sigma| = q$. We then denote the size of the largest $(n, M; t)_k$ code over Σ by $A_q(n; t)_k$. The capacity of the channel is then defined as

$$\mathsf{cap}_q(t)_k = \limsup_{n\to\infty} \frac{1}{n} \log_q A_q(n; t)_k.$$

Analogous definitions are obtained by replacing k with $\leqslant k$ or by replacing t with $*$.

III. k-TANDEM-DUPLICATION CODES

In this section we consider tandem string-duplication systems where the substring being duplicated is of a constant length k. Such systems were studied in the context of formal languages [15] (also called *uniform-bounded-duplication systems*), and also in the context of coding and information theory [4].

In [15] it was shown that for any finite alphabet Σ, and any word $x \in \Sigma^*$, under k-tandem duplication x has a unique root, i.e.,

$$|R_k(x)| = 1.$$

Additionally, finding the unique root may be done efficiently, even by a greedy algorithm which searches for occurrences of ww as substrings of x, with $|w| = k$, removing one copy of w, and repeating the process. This was later extended in [14], where it was shown that the roots of a regular languages also form a regular language. In the section that follows we give an alternative elementary proof to the uniqueness of the root. This proof will enable us to easily construct codes for k-tandem-duplication systems, as well as to state bounds on their parameters.

We also mention [4], in which S_k was studied from a coding and information-theoretic perspective. It was shown there that the capacity of all such systems is 0. This fact will turn out to be extremely beneficial when devising error-correcting codes for k-tandem-duplication systems.

Throughout this section, without loss of generality, we assume $\Sigma = \mathbb{Z}_q$. We also use $\mathbb{Z}_q^*$ to denote the set of all finite strings of $\mathbb{Z}_q$ (not to be confused with the non-zero elements of $\mathbb{Z}_q$), and $\mathbb{Z}_q^{\geqslant k}$ to denote the set of all finite strings over $\mathbb{Z}_q$ of length k or more.

We shall require the following mapping, $\phi_k : \mathbb{Z}_q^{\geqslant k} \to \mathbb{Z}_q^k \times \mathbb{Z}_q^*$. The mapping is defined by,

$$\phi_k(x) = (\mathrm{Pref}_k(x), \mathrm{Suff}_{|x|-k}(x) - \mathrm{Pref}_{|x|-k}(x)),$$

where subtraction is performed entry-wise over $\mathbb{Z}_q$. We easily observe that ϕ_k is a bijection between $\mathbb{Z}_q^n$ and $\mathbb{Z}_q^k \times \mathbb{Z}_q^{n-k}$ by noting that we can recover x from $\phi_k(x)$ in the following manner: first set $x_i = \phi_k(x)_i$, for all $1 \leqslant i \leqslant k$, and for $i = k+1, k+2, \dots$, set $x_i = x_{i-k} + \phi_k(x)_i$, where $\phi_k(x)_i$ denotes the ith symbol of $\phi_k(x)$. Thus, ϕ_k^{-1} is well defined.

Another mapping we define is one that injects k consecutive zeros into a string. More precisely, we define $\zeta_{i,k} : \mathbb{Z}_q^k \times \mathbb{Z}_q^* \to \mathbb{Z}_q^k \times \mathbb{Z}_q^*$, where

$$\zeta_{i,k}(x, y) = \begin{cases} (x, u0^k w) & \text{if } y = uw,\ |u| = i \\ (x, y) & \text{otherwise.} \end{cases}$$

The following lemma will form the basis for the proofs to follow.

Lemma 6. *The following diagram commutes:*

$$\begin{array}{ccc} \mathbb{Z}_q^{\geqslant k} & \xrightarrow{T_{i,k}} & \mathbb{Z}_q^{\geqslant k} \\ \big\downarrow \phi_k & & \big\downarrow \phi_k \\ \mathbb{Z}_q^k \times \mathbb{Z}_q^* & \xrightarrow{\zeta_{i,k}} & \mathbb{Z}_q^k \times \mathbb{Z}_q^* \end{array}$$

i.e., for every string $x \in \mathbb{Z}_q^{\geqslant k}$,

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

Before presenting the proof, we provide an example for the diagram of the lemma.

Example 7. *Assume* $\Sigma = \mathbb{Z}_4$*. Starting with* 02123 *and letting* $i = 1$ *and* $k = 2$ *leads to*

$$\begin{array}{ccc} 02123 & \xrightarrow{T_{1,2}} & 021\underline{21}23 \\ \big\downarrow \phi_2 & & \big\downarrow \phi_2 \\ (02, 102) & \xrightarrow{\zeta_{1,2}} & (02, 1\underline{00}02) \end{array}$$

where the inserted elements are underlined. □

Proof: Let $x \in \mathbb{Z}_q^{\geqslant k}$ be some string, $x = x_1 x_2 \dots x_n$. Additionally, let $\phi_k(x) = (y, z)$ with $y = y_1 \dots y_k$, and $z = z_1 \dots z_{n-k}$. We first consider the degenerate case, where $i \geqslant n - k + 1$. In that case, $T_{i,k}(x) = x$, and then by definition $\zeta_{i,k}(y, z) = (y, z)$ since z does not have a prefix of length at least $n - k + 1$. Thus, for $i \geqslant n - k + 1$ we indeed have

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

We are left with the case of $0 \leqslant i \leqslant n - k$. We now write

$$T_{i,k}(x) = x_1 x_2 \dots x_{i+k} x_{i+1} x_{i+2} \dots x_n.$$

Thus, if we denote $\phi_k(T_{i,k}(x)) = (y,z)$, then

$$\begin{aligned} y &= x_1 \dots x_k = \mathrm{Pref}_k(x), \\ z &= x_{k+1} - x_1, \dots, x_{k+i} - x_i, 0^k, \\ &\quad x_{k+i+1} - x_{i+1}, \dots, x_n - x_{n-k}. \end{aligned}$$

This is exactly an insertion of 0^k after i symbols in the second part of $\phi_k(x)$. It therefore follows that

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

as claimed. ∎

Recalling that ϕ_k is a bijection between $\mathbb{Z}_q^n$ and $\mathbb{Z}_q^k \times \mathbb{Z}_q^{n-k}$, together with Lemma 6 gives us the following corollary.

Corollary 8. *For any $x \in \mathbb{Z}_q^{\geqslant k}$, and for any sequence of non-negative integers $i_1, \dots, i_t$,*

$$T_{i_t,k}(\dots T_{i_1,k}(x)\dots) = \phi_k^{-1}(\zeta_{i_t,k}(\dots \zeta_{i_1,k}(\phi_k(x))\dots)).$$

Example 9. *Continuing Example 7, let $x = 02123$, $k = t = 2$, $i_1 = 1$, and $i_2 = 0$. Then*

$$\begin{aligned} &T_{0,2}(T_{1,2}(02123)) \\ &= T_{0,2}(0212123) \\ &= 020212123 \\ &= \phi_k^{-1}((02, 0010002)) \\ &= \phi_k^{-1}(\zeta_{0,2}((02, 10002))) \\ &= \phi_k^{-1}(\zeta_{0,2}(\zeta_{1,2}((02, 102)))) \\ &= \phi_k^{-1}(\zeta_{0,2}(\zeta_{1,2}(\phi_k(02123)))). \end{aligned}$$

□

Corollary 8 paves the way to working in the ϕ_k-transform domain. In this domain, a tandem-duplication operation of length k translates into an insertion of a block of k consecutive zeros. Conversely, a tandem-deduplication operation of length k becomes a removal of a block of k consecutive zeros.

The uniqueness of the root, proved in [15], now comes for free. In the ϕ_k-transform domain, given $(x,y) \in \mathbb{Z}_q^k \times \mathbb{Z}_q^*$, as long as y contains a substring of k consecutive zeros, we may perform another deduplication. The process stops at the unique outcome in which the length of every run of zeros in y is reduced modulo k.

This last observation motivates us to define the following operation on a string in $\mathbb{Z}_q^*$. We define $\mu_k : \mathbb{Z}_q^* \to \mathbb{Z}_q^*$ which reduces the lengths of runs of zeros modulo k in the following way. Consider a string $x \in \mathbb{Z}_q^*$, where

$$x = 0^{m_0} w_1 0^{m_1} w_2 \dots w_t 0^{m_t},$$

where m_i are non-negative integers, and $w_1, \dots, w_t \in \mathbb{Z}_q \setminus \{0\}$, i.e., $w_1, \dots, w_t$ are single non-zero symbols. We then define

$$\mu_k(x) = 0^{m_0 \bmod k} w_1 0^{m_1 \bmod k} w_2 \dots w_t 0^{m_t \bmod k}.$$

For example, for $z = 0010002$,

$$\mu_2(z) = 102.$$

Additionally, we define

$$\sigma_k(x) = \left(\left\lfloor \frac{m_0}{k} \right\rfloor, \left\lfloor \frac{m_1}{k} \right\rfloor, \dots, \left\lfloor \frac{m_t}{k} \right\rfloor \right) \in (\mathbb{N} \cup \{0\})^*$$

and call $\sigma(x)$ the *zero signature* of x. For z given above,

$$\sigma_2(z) = (1,1,0).$$

We note that $\mu_k(x)$ and $\sigma(x)$ together uniquely determine x.

We also observe some simple properties. First, the Hamming weight of a vector, denoted wt_H, counts the number of non-zero elements in a vector. By definition we have for every $x \in \mathbb{Z}_q^n$,

$$\mathrm{wt}_H(x) = \mathrm{wt}_H(\mu_k(x)).$$

Additionally, the length of the vector $\sigma_k(x)$, denoted $|\sigma_k(x)|$, is given by

$$|\sigma_k(x)| = \mathrm{wt}_H(x) + 1 = \mathrm{wt}_H(\mu_k(x)) + 1. \qquad (1)$$

Note that for $z = 0010002$ as above, we have

$$|\sigma_2(z)| = 3 = \mathrm{wt}_H(z) + 1 = \mathrm{wt}_H(102) + 1.$$

Thus, our previous discussion implies the following corollary.

Corollary 10. *For any string $x \in \mathbb{Z}_q^{\geqslant k}$,*

$$R_k(x) = \left\{ \phi_k^{-1}(y, \mu_k(z)) \;\middle|\; \phi_k(x) = (y,z) \right\}.$$

We recall the definition of the $(0, k-1)$-RLL system over $\mathbb{Z}_q$ (for example, see [9], [16]). It is defined as the set of all finite strings over $\mathbb{Z}_q$ that do not contain k consecutive zeros. We denote this set as $C_{\mathrm{RLL}_q(0,k-1)}$. In our notation,

$$C_{\mathrm{RLL}_q(0,k-1)} = \left\{ x \in \mathbb{Z}_q^* \;\middle|\; \sigma_k(x) \in 0^* \right\}.$$

By convention, $C_{\mathrm{RLL}_q(0,k-1)} \cap \mathbb{Z}_q^0 = \{\epsilon\}$. The following is another immediate corollary.

Corollary 11. *For all $n \geqslant k$,*

$$\mathrm{Irr}_k(n) = \left\{ \phi_k^{-1}(y,z) \;\middle|\; y \in \mathbb{Z}_q^k, z \in C_{\mathrm{RLL}_q(0,k-1)} \cap \mathbb{Z}_q^{n-k} \right\}.$$

Proof: The proof is immediate since x is irreducible iff no deduplication action may be applied to it. This happens iff for $\phi_k(x) = (y,z)$, z does not contain k consecutive zeros, i.e., $z \in C_{\mathrm{RLL}_q(0,k-1)} \cap \mathbb{Z}_q^{n-k}$. ∎

Given two strings, $x, x' \in \mathbb{Z}_q^{\geqslant k}$, we say x and x' are k-congruent, denoted $x \sim_k x'$, if $R_k(x) = R_k(x')$. It is easily seen that $\sim_k$ is an equivalence relation.

Example 12. *For instance,* 02123, 0212323, 0212123, *and* 020212123 *are all 2-congruent, since they have the unique root* 02123. □

Corollary 13. *Let $x, x' \in \mathbb{Z}_q^*$ be two strings, and denote $\phi_k(x) = (y,z)$ and $\phi_k(x') = (y',z')$. Then $x \sim_k x'$ iff $y = y'$ and $\mu_k(z) = \mu_k(z')$.*

Proof: This is immediate when using Corollary 10 to express the roots of x and x'. ∎

For all sequences x in the preceding Example, if we let $\phi_2(x) = (y,z)$, then $y = 02$ and $\mu_2(z) = 102$.

The following lemma appeared in [15, Proposition 2]. We restate it and give an alternative proof.

Lemma 14. *For all $x, x' \in \mathbb{Z}_q^{\geqslant k}$, we have*

$$D_k^*(x) \cap D_k^*(x') \neq \emptyset$$

if and only if $x \sim_k x'$.

Proof: In the first direction, assume $x \not\sim_k x'$. By the uniqueness of the root, let us denote $R_k(x) = \{u\}$ and $R_k(x') = \{u'\}$, with $u \neq u'$. If there exists $w \in D_k^*(x) \cap D_k^*(x')$, then w is a descendant of both u and u', therefore $u, u' \in R_k(w)$, which is a contradiction. Hence, no such w exists, i.e., $D_k^*(x) \cap D_k^*(x') = \emptyset$.

In the other direction, assume $x \sim_k x'$. We construct a word $w \in D_k^*(x) \cap D_k^*(x')$. Denote $\phi_k(x) = (y,z)$ and $\phi_k(x') = (y',z')$. By Corollary 13 we have

$$y = y', \\ \mu_k(z) = \mu_k(z').$$

Let us then denote

$$z = 0^{m_0} v_1 0^{m_1} v_2 \dots v_t 0^{m_t}, \\ z' = 0^{m'_0} v_1 0^{m'_1} v_2 \dots v_t 0^{m'_t},$$

with v_i a non-zero symbol, and

$$m_i \equiv m'_i \pmod{k},$$

for all i. We now define

$$z'' = 0^{\max(m_0,m'_0)} v_1 0^{\max(m_1,m'_1)} v_2 \dots v_t 0^{\max(m_t,m'_t)}.$$

Since z'' differs from z and z' by insertion of blocks of k consecutive zeros, it follows that

$$w = \phi_k^{-1}(y, z'') \in D_k^*(x) \cap D_k^*(x'),$$

which completes the proof. ■

We now turn to constructing error-correcting codes. The first construction is for a code capable of correcting all errors.

Construction A. *Fix $\Sigma = \mathbb{Z}_q$ and $k \geqslant 1$. For any $n \geqslant k$ we construct*

$$C = \bigcup_{i=0}^{\lfloor n/k \rfloor - 1} \left\{ \phi_k^{-1}(y, z0^{ki}) \;\middle|\; \phi_k^{-1}(y,z) \in \mathrm{Irr}_k(n - ik) \right\}.$$

Theorem 15. *The code C from Construction A is an $(n, M; *)_k$ code, with*

$$M = \sum_{i=0}^{\lfloor n/k \rfloor - 1} q^k M_{\mathrm{RLL}_q(0,k-1)}(n - (i+1)k).$$

Here $M_{\mathrm{RLL}_q(0,k-1)}(m)$ denotes the number of strings of length m which are $(0,k-1)$-RLL over $\mathbb{Z}_q$, i.e.,

$$M_{\mathrm{RLL}_q(0,k-1)}(m) = \left| C_{\mathrm{RLL}_q(0,k-1)} \cap \mathbb{Z}_q^m \right|.$$

Proof: The size of the code is immediate, by Corollary 11. Additionally, the roots of distinct codewords are distinct as well, since we constructed the code from irreducible words with blocks of k consecutive zeros appended to their end. Thus, by Lemma 14, the descendant cones of distinct codewords are disjoint. ■

We can say more about the size of the code we constructed.

Theorem 16. *The code C from Construction A is optimal, i.e., it has the largest cardinality of any $(n;*)_k$ code.*

Proof: By Lemma 14, any two distinct codewords of an $(n;*)_k$ code must belong to different equivalence classes of $\sim_k$. The code C of Construction A contains exactly one codeword from each equivalence class of $\sim_k$, and thus, it is optimal. ■

The code C from Construction A also allows a simple decoding procedure, whose correctness follows from Corollary 10. Assume a word $x' \in \mathbb{Z}_q^{\geqslant k}$ is received, and let $\phi_k(x') = (y', z')$. The decoded word is simply

$$\tilde{x} = \phi_k^{-1}(y', \mu_k(z') 0^{n-k-|\mu_k(z')|}), \tag{2}$$

where n is the length of the code C. In other words, the decoding procedure recovers the unique root of the received x', and in the ϕ_k-transform domain, pads it with enough zeros.

Example 17. *Let $n = 4$, $q = 2$, and $k = 1$. By inspection, the code C of Construction A can be shown to equal*

$$C = \{\underline{0}000, \underline{01}11, \underline{010}0, \underline{0101}, \underline{1}111, \underline{10}00, \underline{101}1, \underline{1010}\},$$

where in each codeword the k-irreducible part is underlined. As an example of decoding, both 01100 and 01000 decode to 0100. Specifically for the former case, $x' = 01100$, we have $\phi_k(x') = (y', z') = (0, 1010)$. So $\mu_k(z') = 11$ and

$$\tilde{x} = \phi_k^{-1}(0, 110) = 0100.$$

□

Encoding may be done in any of the many various ways for encoding RLL-constrained systems. The reader is referred to [9], [16] for further reading. After encoding the RLL-constrained string z, a string $y \in \mathbb{Z}_q^k$ is added, and ϕ_k^{-1} employed, to obtain a codeword.

Finally, the asymptotic rate of the code family may also be obtained, thus, obtaining the capacity of the channel.

Corollary 18. *For all $q \geqslant 2$ and $k \geqslant 1$,*

$$\mathsf{cap}_q(*)_k = \mathsf{cap}(\mathrm{RLL}_q(0, k-1)),$$

where $\mathsf{cap}(\mathrm{RLL}_q(0,k-1))$ is the capacity of the q-ary $(0,k-1)$-RLL constrained system.

Proof: We use C_n to denote the code from Construction A, where the subscript n is used to denote the length of the code. It is easy to see that for $n \geqslant k$,

$$q^k M_{\mathrm{RLL}_q(0,k-1)}(n-k) \leqslant |C_n| \leqslant n q^k M_{\mathrm{RLL}_q(0,k-1)}(n-k).$$

Then by standard techniques [16] for constrained coding,

$$\lim_{n\to\infty}\frac{1}{n}\log_2|C_n| = \mathsf{cap}(\mathrm{RLL}_q(0,k-1)) = \log_2\lambda(A_q(k-1)),$$

where $\lambda(A_q(k-1))$ is the largest eigenvalue of the $k\times k$ matrix $A_q(k-1)$ defined as

$$A_q(k-1) = \begin{pmatrix} q-1 & 1 & & & \\ q-1 & & 1 & & \\ \vdots & & & \ddots & \\ q-1 & & & & 1 \\ q-1 & & & & \end{pmatrix}. \quad (3)$$

■

As a side note, we comment that an asymptotic (in k) expression for the capacity may be given by

$$\mathsf{cap}(\mathrm{RLL}_q(0,k)) = \log_2 q - \frac{(q-1)\log_2 e}{q^{k+2}}(1+o(1)). \quad (4)$$

This expression agrees with the expression for the binary case $q=2$ mentioned in [12] without proof or reference. For completeness, we bring a short proof of this claim in the appendix.

Having considered $(n,M;*)_k$ codes, we now turn to study $(n,M;t)_k$ codes for $t\in\mathbb{N}\cup\{0\}$. We note that $\mathbb{Z}_q^n$ is an optimal $(n,q^n;0)_k$ code. Additionally, any $(n,M;*)_k$ code is trivially also an $(n,M;t)_k$ code, though not necessarily optimal.

We know by Lemma 14 that the descendant cones of two words overlap if and only if they are k-congruent. Thus, the strategy for constructing $(n,M;*)_k$ codes was to pick single representatives of the equivalence classes of $\sim_k$ as codewords. However, the overlap that is guaranteed by Lemma 14 may require a large amount of duplication operations. If we are interested in a small enough value of t, then an $(n,M;t)_k$ code may contain several codewords from the same equivalence class. This observation will be formalized in the following, by introducing a metric on k-congruent words, and applying this metric to pick k-congruent codewords.

Fix a length $n\geqslant 1$, and let $x,x'\in\mathbb{Z}_q^n$, $x\sim_k x'$, be two k-congruent words of length n. We define the distance between x and x' as

$$d_k(x,x') = \min\left\{t\geqslant 0 \;\middle|\; D_k^t(x)\cap D_k^t(x')\neq\varnothing\right\}.$$

Since x and x' and k-congruent, Lemma 14 ensures that d_k is well defined.

Lemma 19. *Let $x,x'\in\mathbb{Z}_q^n$, $x\sim_k x'$, be two k-congruent strings. Denote $\phi_k(x)=(y,z)$ and $\phi_k(x')=(y,z')$. Additionally, let*

$$\sigma_k(z) = (s_0,s_1,\dots,s_r),$$
$$\sigma_k(z') = (s'_0,s'_1,\dots,s'_r).$$

Then

$$d_k(x,x') = \sum_{i=0}^{r}\left|s_i-s'_i\right| = d_{\ell_1}(\sigma_k(z),\sigma_k(z')),$$

where d_{ℓ_1} stands for the ℓ_1-distance function.

Proof: Let x and x' be two strings as required. By Corollary 13 we indeed have $y=y'$, and $\mu_k(z)=\mu_k(z')$. In particular, the length of the vectors of the zero signatures of z and z' are the same,

$$\left|\sigma_k(z)\right| = \left|\sigma_k(z')\right| = r+1.$$

We now observe that the action of a k-tandem duplication on x corresponds to the addition of a standard unit vector e_i (an all-zero vector except for the ith coordinate which equals 1) to $\sigma_k(z)$.

Let $\tilde{x}$ denote a vector that is a descendant both of x and x', and that requires the least number of k-tandem duplications to reach from x and x'. If we denote $\phi_k(\tilde{x})=(\tilde{y},\tilde{z})$, then we have

$$\tilde{y} = y = y',$$
$$\mu_k(\tilde{z}) = \mu_k(z) = \mu_k(z'),$$
$$\sigma_k(\tilde{z}) = (\max(s_0,s'_0),\dots,\max(s_r,s'_r)).$$

Thus,

$$d_k(x,x') = \sum_{i=0}^{r}(\max(s_i,s'_i)-s_i) = \sum_{i=0}^{r}(\max(s_i,s'_i)-s'_i) = \sum_{i=0}^{r}\left|s_i-s'_i\right| = d_{\ell_1}(\sigma_k(z),\sigma_k(z')).$$

■

From Lemma 19 we also deduce that d_k is a metric over any set of k-congruent words of length n.

The following theorem shows that a code is $(n;t)_k$ if and only if the zero signatures of the z-part of k-congruent codewords in the ϕ_k-transform domain, form a constant-weight code in the ℓ_1-metric with distance at least $t+1$. We recall that the ℓ_1-metric weight of a vector $s=s_1s_2\dots s_n\in\mathbb{Z}^n$ is defined as the ℓ_1-distance to the zero vector, i.e.,

$$\mathrm{wt}_{\ell_1}(s) = \sum_{i=1}^{n}|s_i|.$$

Theorem 20. *Let $C\subseteq\mathbb{Z}_q^n$, $n\geqslant k$, be a subset of size M. Then C is an $(n,M;t)_k$ code if and only if for each $y\in\mathbb{Z}_q^k$, $z\in\mathbb{Z}_q^{n-k}$, the following sets*

$$C(y,z) = \left\{\sigma_k(z') \;\middle|\; z'\in\mathbb{Z}_q^{n-k}, \mu_k(z)=\mu_k(z'), \phi_k^{-1}(y,z')\in C\right\}$$

are constant-weight $(n(y,z),M(y,z),t+1)$ codes in the ℓ_1-metric, with constant weight

$$\mathrm{wt}_{\ell_1}(\sigma(z)) = \frac{n-k-|\mu_k(z)|}{k},$$

and length

$$n(y,z) = \mathrm{wt}_H(z)+1 = \mathrm{wt}_H(\mu_k(z))+1,$$

where wt_H *denotes the Hamming weight.*

Proof: In the first direction, let C be an $(n,M;t)_k$ code. Fix y and z, and consider the set $C(y,z)$. Assume to the contrary that there exist distinct $\sigma_k(z'),\sigma_k(z'') \in C(y,z)$, $z',z'' \in \mathbb{Z}_q^{n-k}$, such that $d_{\ell_1}(\sigma_k(z'),\sigma_k(z'')) \leqslant t$.

The length of the code, $n(y,z)$, is obvious given (1). We note that $\sigma_k(z') \neq \sigma_k(z'')$ implies $z' \neq z''$. By definition, we have

$$\mu_k(z) = \mu_k(z') = \mu_k(z'').$$

Thus,

$$\begin{aligned}\mathrm{wt}_{\ell_1}(\sigma(z)) &= \mathrm{wt}_{\ell_1}(\sigma(z')) = \mathrm{wt}_{\ell_1}(\sigma(z''))\\ &= \frac{n-k-|\mu_k(z)|}{k},\end{aligned}$$

where $|\mu_k(z)|$ denotes the length of the vector $\mu_k(z)$. Additionally, the two codewords

$$c' = \phi_k^{-1}(y,z') \in C \qquad \text{and} \qquad c'' = \phi_k^{-1}(y,z'') \in C$$

are k-congruent and distinct. By Lemma 19,

$$d_k(c',c'') = d_{\ell_1}(\sigma_k(z'),\sigma_k(z'')) \leqslant t. \tag{5}$$

However, that contradicts the code parameters since we have (5) imply $D_k^t(c') \cap D_k^t(c'') \neq \emptyset$, whereas in an $(n,M;t)_k$ code, the t-descendant cones of distinct codewords have an empty intersection.

In the other direction, assume that for every choice of y and z, the corresponding $C(y,z)$ is a constant-weight code with minimum ℓ_1-distance of $t+1$. Assume to the contrary C is not an $(n,M;t)_k$ code. Therefore, there exist two distinct codewords, $c',c'' \in C$ such that $d_k(c',c'') \leqslant t$.

By Lemma 14 we conclude that c' and c'' are k-congruent. Thus, there exist $y \in \mathbb{Z}_q^k$ and $z \in \mathbb{Z}_q^{n-k}$ (z is not necessarily unique) such that,

$$\begin{aligned}\phi_k(c') &= (y,z')\\ \phi_k(c'') &= (y,z'')\\ \mu_k(z) &= \mu_k(z') = \mu_k(z'').\end{aligned}$$

We can now use Lemma 19 and obtain

$$d_{\ell_1}(\sigma_k(z'),\sigma_k(z'')) = d_k(c',c'') \leqslant t,$$

which contradicts the minimal distance of $C(y,z)$. ■

With the insight given by Theorem 20 we now give a construction for $(n,M;t)_k$ codes.

Construction B. *Fix* $\Sigma = \mathbb{Z}_q$, $k \geqslant 1$, $n \geqslant k$, *and* $t \geqslant 0$. *Furthermore, for all*

$$\begin{aligned}1 &\leqslant m \leqslant n-k+1,\\ 0 &\leqslant w \leqslant \left\lfloor \frac{n-k}{k} \right\rfloor,\end{aligned}$$

fix ℓ_1*-metric codes over* $\mathbb{Z}_q$*, denoted* $C_1(m,w)$*, which are of length* m*, constant* ℓ_1*-weight* w*, and minimum* ℓ_1*-distance* $t+1$*. We construct*

$$C = \left\{ \phi_k^{-1}(y,z) \;\middle|\; \begin{aligned}& y \in \mathbb{Z}_q^k, z \in \mathbb{Z}_q^{n-k},\\ & \sigma_k(z) \in C_1\left(\mathrm{wt}_H(\mu_k(z))+1, \frac{n-k-|\mu_k(z)|}{k}\right)\end{aligned} \right\}.$$

Corollary 21. *The code* C *from Construction* B *is an* $(n,M;t)_k$ *code.*

Proof: Let $c,c' \in C$ be two k-congruent codewords, i.e., $\phi_k(c) = (y,z)$, $\phi_k(c') = (y,z')$, and $\mu_k(z) = \mu_k(z')$. It follows, by construction, that $\sigma_k(z)$ and $\sigma_k(z')$ belong to the same ℓ_1-metric code with minimum ℓ_1-distance at least $t+1$. By Theorem 20, C is an $(n,M;t)_k$ code. ■

Due to Theorem 20, a choice of optimal ℓ_1-metric codes in Construction B will result in optimal $(n,M;t)_k$ codes. We are unfortunately unaware of explicit construction for such codes. However, we may deduce such a construction from codes for the similar Lee metric (e.g., [19]), while applying a standard averaging argument for inferring the existence of a constant-weight code. We leave the construction of such codes for a future work.

IV. $\leqslant k$-TANDEM-DUPLICATION CODES

In this Section, we consider error-correcting codes that correct duplications of length at most k, which correspond to $S_{\leqslant k}$, i.e., bounded tandem string-duplication systems, where the substring being duplicated is of maximum length k. In particular, we present constructions for codes that can correct any number of duplications of length $\leqslant 3$ as well as a lower bound on the capacity of the corresponding channel. In the case of duplications of length $\leqslant 2$ we give optimal codes, and obtain the exact capacity of the channel.

It is worth noting that the systems $S_{\leqslant k}$ were studied in the context of formal languages [15] and also in the context of coding and information theory [10]. In [15], it was shown that $S_{\leqslant k}$, with $k \geqslant 4$, are not a regular language for alphabet size $|\Sigma| \geqslant 3$. However, it was proved in [10] that $S_{\leqslant 3}$ is indeed a regular language irrespective of the starting string and the alphabet size.

In this paper, we will show that strings that can be generated by bounded tandem string-duplication systems with maximum duplication length 3 have a unique duplication root, a fact that will be useful for our code construction. Theorem 24 formalizes this statement. Before stating Theorem 24, we define the following.

Definition 22. *Let two squares* $y_1 = \alpha\alpha \in \Sigma^*$ *and* $y_2 = \beta\beta \in \Sigma^*$ *appear as substrings of some string* $u \in \Sigma^*$*, i.e.,*

$$u = x_1 y_1 z_1 = x_2 y_2 z_2,$$

with $|x_1| = i$*,* $|x_2| = j$*. We say* y_1 *and* y_2 *are* overlapping squares in u *if the following conditions both hold:*

1) $i \leqslant j \leqslant i + 2|\alpha| - 1$.

2) *If $i = j$, then $\alpha \neq \beta$.*

Example 23. *As an example, consider the sequence u,*

$$u = 01\underbrace{2323}_{\beta_1\beta_1}\overbrace{\underbrace{452452}_{\beta_2\beta_2}323452}^{\alpha\alpha}\underbrace{456245}_{\beta_3\beta_3}67,$$

where $\alpha\alpha$ and $\beta_i\beta_i$ for each $i \in \{1,2,3\}$ are overlapping squares. □

The following theorem shows that every word has a unique root under tandem deduplication of length up to 3.

Theorem 24. *For any $z \in \Sigma^*$ we have $|R_{\leqslant 3}(z)| = 1$.*

Proof: Fix some $z \in \Sigma^*$, and assume z has exactly k distinct roots, $R_{\leqslant 3}(z) = \{y_1, y_2, \dots, y_k\}$. Let us assume to the contrary that $k \geqslant 2$.

Let us follow a deduplication sequence starting at $x_0 = z$. At each step, we deduplicate $x_i \Longleftarrow_{\leqslant 3} x_{i+1}$, and we must have $|R_{\leqslant 3}(x_i)| \geqslant |R_{\leqslant 3}(x_{i+1})|$. At each step, out of the possible immediate ancestors of x_i, we choose x_{i+1} to be one with $|R_{\leqslant 3}(x_{i+1})| \geqslant 2$ if possible. Since the end-point of a deduplication process is an irreducible sequence, we must reach a sequence x in the deduplication process with the following properties:

1) $z \Longleftarrow^*_{\leqslant 3} x$
2) $|R_{\leqslant 3}(x)| \geqslant 2$
3) For each $x' \in \Sigma^*$ such that $x \Longleftarrow_{\leqslant 3} x'$, $|R_{\leqslant 3}(x')| = 1$.
4) There exist $v, w \in \Sigma^*$ such that $x \Longleftarrow_{\leqslant 3} v$ and $x \Longleftarrow_{\leqslant 3} w$ with $|R_{\leqslant 3}(v)| = |R_{\leqslant 3}(w)| = 1$.
5) $R_{\leqslant 3}(v) = \{y_i\} \neq \{y_j\} = R_{\leqslant 3}(w)$.

Intuitively, in the deduplication process starting from z, we reach a sequence x with more than one root, but any following single deduplication moves us into a single descendant cone of one of the roots of z. We note that all ancestors of v must have a single root y_i, and all ancestors of w must have a single root y_j.

Thus, x must contain a square $u_v u_v$ whose deduplication results in v, and a square $u_w u_w$ whose deduplication results in w. We contend that the squares $u_v u_v$ and $u_w u_w$ overlap. Otherwise, if $u_v u_v$ and $u_w u_w$ do not overlap in x, we may deduplicate them in any order to obtain the same result. Hence, there exists $t \in \Sigma^*$ such that $v \Longleftarrow_{\leqslant 3} t$ and $w \Longleftarrow_{\leqslant 3} t$. But then, since t is an ancestor both of v and w,

$$\{y_i\} = R_{\leqslant 3}(v) = R_{\leqslant 3}(t) = R_{\leqslant 3}(w) = \{y_j\},$$

a contradiction.

We now know that $u_v u_v$ and $u_w u_w$ must overlap. We also note $|u_v|, |u_w| \leqslant 3$. Let $a, b, c \in \Sigma$ be three distinct symbols. If the alphabet is smaller, then some of the cases below may be ignored, and the proof remains the same. We use brute force to enumerate the following cases: (each string describes the shortest subsequence that contains the overlapping squares)

1) $|u_v| = 1, |u_w| = 1$: *aaa*.
2) $|u_v| = 1, |u_w| = 2$: *aaaaa*, *aabab*.
3) $|u_v| = 1, |u_w| = 3$: *aaaaaa*, *aaaaaaa*, *aabaaba*, *abaaba*, *aabcabc*.
4) $|u_v| = 2, |u_w| = 2$: *aaaaa*, *ababab*, *ababbbb*, *ababa*, *bcbcaca*.
5) $|u_v| = 2, |u_w| = 3$: *aaaaaa*, *aaaaaaa*, *aaaaaaaa*, *aaaaaaaaa*, *abaabaaaa*, *abaababa*, *abaabab*, *abcabcccc*, *abcabcaca*, *abcabcbcb*, *abcabcbc*.
6) $|u_v| = 3, |u_w| = 3$: *aaaaaaa*, *aaaaaaaa*, *aaaaaaaaa*, *aaaaaaaaaa*, *aaaaaaaaaaa*, *abaabaaaaaa*, *abaababaaba*, *abaabacaaca*, *abaababcabc*, *abaabacbacb*, *abaabaabaa*, *abaababbab*, *abaabacbac*, *abaabaaba*, *abaabaab*, *abaabaa*, *abcabcaacaa*, *abcabcbbcbb*, *abcabcbccbc*, *abcabcaccac*, *abcabccbccb*, *abcabccacca*, *abcabccbcc*, *abcabcbbcb*, *abcabcabca*, *abcabcabc*, *abcabcab*, *abcabca*.

All other cases left are symmetric (by relabeling the alphabet symbols) to one of the above listed case. For example, if $u_v = abc$ and $u_w = cbc$, the corresponding string appears in case 6) as *abcabcbccbc*. It is tedious, yet easy, to check that each of the above listed cases has a unique root if deduplication of maximum length 3 is allowed. In the above example, indeed, the only possible root is *abc*,

$$\underline{abcabc}bccbc \Longleftarrow_{\leqslant 3} abcbccbc \Longleftarrow^*_{\leqslant 3} abc,$$
$$abcab\underline{cbccbc} \Longleftarrow_{\leqslant 3} abcabcbc \Longleftarrow^*_{\leqslant 3} abc.$$

Let $x = \alpha\beta\gamma \in \Sigma^*$, where β covers exactly the overlapping squares, and is one of the above listed cases. Then, by deduplication of $u_v u_v$ from β in x, we get v, and by deduplication of $u_w u_w$ from β in x, we get w. However, since β has a unique root, we may deduplicate v and w to the same word $t = \alpha\beta'\gamma \in \Sigma^*$, where $R(\beta) = \{\beta'\}$, i.e., β' is the unique root of β. Thus, t is an ancestor of both v and w. Again,

$$\{y_i\} = R_{\leqslant 3}(v) = R_{\leqslant 3}(t) = R_{\leqslant 3}(w) = \{y_j\},$$

which is a contradiction. ■

Corollary 25. *For any $z \in \Sigma^*$ we also have $|R_{\leqslant k}(z)| = 1$ for $k = 1, 2$.*

In a similar fashion to the previous section, we define the following relation. We say $x, x' \in \Sigma^*$ are $\leqslant 3$-congruent, denoted $x \sim_{\leqslant 3} x'$, if $R_{\leqslant 3}(x) = R_{\leqslant 3}(x')$. Clearly $\sim_{\leqslant 3}$ is an equivalence relation. Having shown any sequence has a unique root when duplicating up to length 3, we obtain the following corollary.

Corollary 26. *For any two words $x, x' \in \Sigma^*$, if*

$$D^*_{\leqslant 3}(x) \cap D^*_{\leqslant 3}(x') \neq \emptyset$$

then $x \sim_{\leqslant 3} x'$.

We note that unlike Lemma 14, we do not have $x \sim_{\leqslant 3} x'$ necessarily imply that their descendant cones intersect. Here is a simple example illustrating this case. Fix $q = 3$, and let $x = 012012$ and $x' = 001122$. We note that $x \sim_{\leqslant 3} x'$, since

$$R_{\leqslant 3}(x) = R_{\leqslant 3}(x') = \{012\}.$$

However, $D^*_{\leqslant 3}(x) \cap D^*_{\leqslant 3}(x') = \varnothing$ since all the descendants of x have a 0 to the right of a 2, whereas all the descendants of x' do not.

We are missing a simple operator which is required to define an error-correcting code. For any sequence $x \in \Sigma^+$, we define its k-suffix-extension to be

$$\xi_k(x) = x(\mathrm{Suff}_1(x))^k,$$

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

Construction C. *Let Σ be some finite alphabet. The constructed code is*

$$C = \bigcup_{i=1}^{n} \left\{ \xi_{n-i}(x) \mid x \in \mathrm{Irr}_{\leqslant 3}(i) \right\}.$$

Theorem 27. *The code C from Construction C is an $(n, M; *)_{\leqslant 3}$ code, where*

$$M = \sum_{i=1}^{n} \left| \mathrm{Irr}_{\leqslant 3}(i) \right|.$$

Proof: The parameters of the code are obvious. Since the last letter duplication induced by the suffix extension may be deduplicated, we clearly have exactly one codeword from each equivalence class of $\sim_{\leqslant 3}$. By Corollary 26, the descendant cones of the codewords do not intersect and the code can indeed correct all errors. ■

For the remainder of the section we denote by $\mathrm{Irr}_{q;\leqslant 3}$ the set of irreducible words with respect to $\Longleftarrow_{\leqslant 3}$ over $\mathbb{Z}_q$, in order to make explicit the dependence on the size of the alphabet. We also assume $q \geqslant 3$, since $q = 2$ is a trivial case with

$$\mathrm{Irr}_{2;\leqslant 3} = \{0, 1, 01, 10, 010, 101\}. \qquad (6)$$

We observe that $\mathrm{Irr}_{q;\leqslant 3}$ is a regular language. Indeed, it is defined by a finite set of subsequences we would like to avoid. This set is exactly

$$\mathcal{F}_q = \left\{ uu \in \mathbb{Z}_q^* \;\middle|\; 1 \leqslant |u| \leqslant 3 \right\}.$$

We can easily construct a finite directed graph with labeled edges such that paths in the graph generate exactly $\mathrm{Irr}_{q;\leqslant 3}$. This graph is obtained by taking the De Bruijn graph $\mathcal{G}_q = (\mathcal{V}_q, \mathcal{E}_q)$ of order 5 over $\mathbb{Z}_q$, i.e., $\mathcal{V}_q = \mathbb{Z}_q^5$, and edges of the form $(a_1, a_2, a_3, a_4, a_5) \to (a_2, a_3, a_4, a_5, a_6)$, for all $a_i \in \mathbb{Z}_q$. Thus, each edge is labeled with a word $w = (a_1, a_2, a_3, a_4, a_5, a_6) \in \mathbb{Z}_q^6$. We then remove all edges labeled by words $\alpha\beta\gamma \in \mathbb{Z}_q^6$ such that $\beta \in \mathcal{F}_q$. We call the resulting graph $\mathcal{G}'_q$. It is easy verify that each path in $\mathcal{G}'_q$ generates a sequence of sliding windows of length 6. Reducing each window to its first letter we get exactly $\mathrm{Irr}_{q;\leqslant 3}$. An example showing $\mathcal{G}'_3$ is given in Figure **??**. Finally, it follows that using known techniques [16], we can calculate $\mathsf{cap}(\mathrm{Irr}_{q;\leqslant 3})$.

Corollary 28. *For all $q \geqslant 3$,*

$$\mathsf{cap}_q(*)_{\leqslant 3} \geqslant \mathsf{cap}(\mathrm{Irr}_{q;\leqslant 3}).$$

Proof: Let M_n denote the size of the length n code over $\mathbb{Z}_q$ from Construction C. By definition, $A_q(n;*)_{\leqslant 3} \geqslant M_n$. We note that trivially

$$M_n = \sum_{i=1}^{n} \left| \mathrm{Irr}_{q;\leqslant 3}(i) \right| \geqslant \left| \mathrm{Irr}_{q;\leqslant 3}(n) \right|.$$

Plugging this into the definition of the capacity gives us the desired claim. ■

Example 29. *Using the constrained system presented in Figure **??** that generates $\mathrm{Irr}_{3;\leqslant 3}$, we can calculate*

$$\mathsf{cap}_3(*)_{\leqslant 3} \geqslant 0.347934.$$

□

Stronger statements may be given when the duplication length is upper bounded by 2 instead of 3.

Lemma 30 *For all $x, x' \in \Sigma^*$, we have*

$$D^*_{\leqslant 2}(x) \cap D^*_{\leqslant 2}(x') \neq \varnothing$$

if and only if $x \sim_{\leqslant 2} x'$.

Proof: In the first direction, assume $x \nsim_{\leqslant 2} x'$. By the uniqueness of the root from Corollary 25, let us denote $R_{\leqslant 2}(x) = \{u\}$ and $R_{\leqslant 2}(x') = \{u'\}$, with $u \neq u'$. If there exists $w \in D^*_{\leqslant 2}(x) \cap D^*_{\leqslant 2}(x')$, then w is a descendant of both u and u', therefore u and $u' \in R_{\leqslant 2}(w)$, which is a contradiction. Hence, no such w exists, i.e., $D^*_{\leqslant 2}(x) \cap D^*_{\leqslant 2}(x') = \varnothing$.

In the other direction, assume $x \sim_{\leqslant 2} x'$. We construct a word $w \in D^*_{\leqslant 2}(x) \cap D^*_{\leqslant 2}(x')$. Let $R_{\leqslant 2}(x) = R_{\leqslant 2}(x') = \{v\}$, and denote $v = a_1 a_2 \dots a_m$, where $a_i \in \Sigma$. Consider a tandem duplication string system $S_{\leqslant 2} = (\Sigma, v, \mathcal{T}_{\leqslant 2})$. Using [10], the regular expression for the language generated by $S_{\leqslant 2}$ is given by

$$a_1^+ a_2^+ (a_1^+ a_2^+)^* a_3^+ (a_2^+ a_3^+)^* \cdots a_m^+ (a_{m-1}^+ a_m^+)^*.$$

Since $x, x' \in S$, we have

$$x = \bigoplus_{i=1}^{\alpha_1} (a_1^{p_{1i}} a_2^{q_{1i}}) a_3^{q_{21}} \bigoplus_{i=2}^{\alpha_2} (a_2^{p_{2i}} a_3^{q_{2i}}) \dots a_m^{q_{(m-1)1}} \bigoplus_{i=2}^{\alpha_{m-1}} (a_{m-1}^{p_{(m-1)i}} a_m^{q_{(m-1)i}}),$$

and

$$x' = \bigoplus_{i=1}^{\beta_1} (a_1^{e_{1i}} a_2^{f_{1i}}) a_3^{f_{21}} \bigoplus_{i=2}^{\beta_2} (a_2^{e_{2i}} a_3^{f_{2i}}) \dots a_m^{f_{(m-1)1}} \bigoplus_{i=2}^{\beta_{m-1}} (a_{m-1}^{e_{(m-1)i}} a_m^{f_{(m-1)i}}),$$

where $\bigoplus$ represents concatenation and $p_{ji}, q_{ji}, e_{ji}, f_{ji}, \alpha_j, \beta_j \geqslant 1$. Now, it is easy to observe that we can obtain

$$w = \bigoplus_{i=1}^{\gamma_1} (a_1^{g_1} a_2^{h_1}) a_3^{h_2} \bigoplus_{i=2}^{\gamma_2} (a_2^{g_2} a_3^{h_2}) \cdots a_m^{h_{m-1}} \bigoplus_{i=2}^{\gamma_{m-1}} (a_{m-1}^{g_{m-1}} a_m^{h_{m-1}})$$

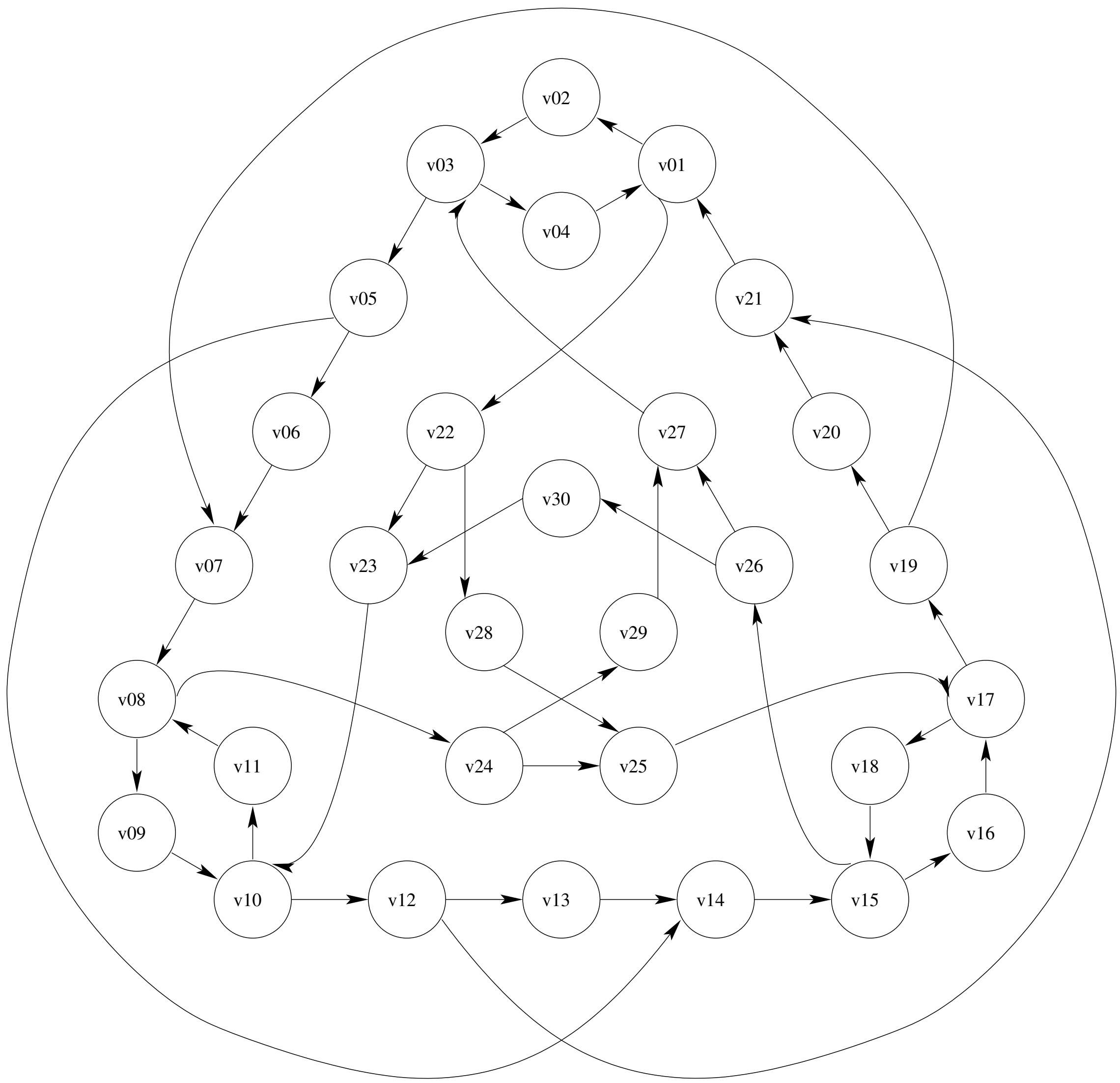

Figure 1. The graph $\mathcal{G}_3'$ producing the set of ternary irreducible words $\mathrm{Irr}_{3;\leqslant 3}$. Vertices without edges were removed as well.

by doing tandem duplication of length up to 2 on x and x', and choosing $\gamma_j = \max\{\alpha_j, \beta_j\}$, $g_j = \max_i\{p_{ji}, e_{ji}\}$, and $h_j = \max_i\{q_{ji}, f_{ji}\}$. Note, p_{ji} and q_{ji} are assumed to be 0 for $i > \alpha_j$ and e_{ji} and f_{ji} are assumed to be 0 for $i > \beta_j$. Thus, $w \in D_{\leqslant 2}(x) \cap D_{\leqslant 2}(x')$. ■

Construction D. *Let Σ be some finite alphabet. The constructed code is*

$$C = \bigcup_{i=1}^{n} \{\xi_{n-i}(x) \mid x \in \mathrm{Irr}_{\leqslant 2}(i)\}.$$

Theorem 31. *The code C from Construction* D *is an optimal $(n, M; *)_{\leqslant 2}$ code, where*

$$M = \sum_{i=1}^{n} |\mathrm{Irr}_{\leqslant 2}(i)|.$$

Proof: The parameters of the code are obvious. Since the last letter duplication induced by the suffix extension may be deduplicated, we clearly have exactly one codeword from each equivalence class of $\sim_{\leqslant 2}$. By Lemma 30, the descendant cones of the codewords do not intersect and the code can indeed correct errors.

By Lemma 30, any two distinct codewords of an $(n; *)_{\leqslant 2}$ code must belong to different equivalence classes of $\sim_{\leqslant 2}$. The code C of Construction D contains exactly one codeword from each equivalence class of $\sim_{\leqslant 2}$, and thus, it is optimal. ■

Corollary 32. *For all $q \geqslant 3$,*

$$\mathsf{cap}_q(*)_{\leqslant 2} = \mathsf{cap}(\mathrm{Irr}_{q;\leqslant 2}).$$

Proof: Let M_n denote the size of the length n code over $\mathbb{Z}_q$ from Construction D. By definition, $A_q(n; *)_{\leqslant 2} \geqslant M_n$. We note that trivially

$$M_n = \sum_{i=1}^{n} \left|\mathrm{Irr}_{q;\leqslant 2}(i)\right| \geqslant \left|\mathrm{Irr}_{q;\leqslant 2}(n)\right|.$$

Additionally, $\left|\mathrm{Irr}_{q;\leqslant 2}\right|(n)$ is monotone increasing in n since any irreducible length-n word x may be extended to an irreducible word of length $n+1$ by adding a letter that is not one of the last two letters appearing in x. Thus,

$$M_n = \sum_{i=1}^{n} \left|\mathrm{Irr}_{q;\leqslant 2}(i)\right| \leqslant n \left|\mathrm{Irr}_{q;\leqslant 2}(n)\right|.$$

Plugging this into the definition of the capacity gives us the desired claim. ■
f

V. Duplication Roots

In Section III, we stated that if the duplication length is uniform (i.e., a constant k), then every sequence has a unique root. Further in Section IV, we proved in Theorem 24 that if the duplication length is bounded by 3 (i.e. $\leqslant 3$), then again every sequence will have a unique root. In fact, the two cases proved in the paper are the only cases of tandem duplication channels that have a unique root given a sequence, namely, in all other cases, the duplication root is not necessarily unique. The characterization is stated in Theorem 40. Before moving to Theorem 40, consider the following example:

Example 33. *Let $U = \{2,3,4\}$ be a set of duplication lengths and $\Sigma = \{1,2,3\}$. Consider*

$$z = \overbrace{1\,2\,3\,2\,1\,\underbrace{2\,3\,2\,3}_{\beta\beta}}^{\alpha\alpha}$$

z has two tandem repeats $\alpha\alpha$ and $\beta\beta$ with $|\alpha| = 4$ and $|\beta| = 2$. If we deduplicate $\alpha\alpha$ first from z , we get

$$123212323 \underset{4}{\Longleftarrow} 12323 \underset{2}{\Longleftarrow} 123 = x.$$

Howevever, if we deduplicate $\beta\beta$ first from z we get

$$123212323 \underset{2}{\Longleftarrow} 1232123 = y.$$

□

Theorem 40 generalizes the statement presented in example above to any set of duplication lengths. We first state the following lemmas to arrive at the statement of Theorem 40.

Lemma 34 *Given Σ, $k > 1$ and a set $U = \{k, k+1\}$, there exists a $z \in \Sigma^*$ with $|R_U(z)| > 1$, i.e., z has more than one root.*

Proof: Consider $z = \{a_1 a_2 \cdots a_{k+1}\}^2 a_2 \cdots a_{k-1} a_k$, where $a_i \in \Sigma$. Let $a_1 = a_{k+1} = a$, then z can be rewritten as

$$\begin{aligned} z &= aa_2 \cdots a_k aaa_2 \cdots a_k aa_2 \cdots a_k \\ &= \{aa_2 \cdots a_k a\}^2 a_2 \cdots a_k \\ &= aa_2 \cdots a_k a \{aa_2 \cdots a_k\}^2. \end{aligned} \tag{7}$$

As is evident in (7), there are two squares in z, one of length $2k$ and the other of length $2k+2$. Deduplicating length $2k+2$ square first gives:

$$\begin{aligned} \{aa_2 \cdots a_k a\}^2 a_2 \cdots a_k &\underset{U}{\Longleftarrow} aa_2 \cdots a_k aa_2 \cdots a_k \\ &\underset{U}{\Longleftarrow} aa_2 \cdots a_k = y. \end{aligned}$$

Deduplicating length $2k$ square first gives

$$aa_2 \cdots a_k a \{aa_2 \cdots a_k\}^2 \underset{U}{\Longleftarrow} aa_2 \cdots a_k aaa_2 \cdots a_k = x.$$

Since $k > 1$, no deduplication is possible in x above since only squares of length k or $k+1$ can be deduplicated and there is no square of length $2k$ or $2k+2$ in x.

As $x \neq y$, we have constructed a z that has more than one duplication root. ■

In the proof of Lemma 34 above, if a deduplication of length 1 was allowed or in other words $U = \{1, k, k+1\}$, then x can be deduplicated to y by deduplicating a square of length 1 first and then a square of length k as shown below:

$$\begin{aligned} x = aa_2 \cdots a_k aaa_2 \cdots a_k &\underset{U}{\Longleftarrow} aa_2 \cdots a_k aa_2 \cdots a_k \\ &\underset{U}{\Longleftarrow} aa_2 \cdots a_k = y. \end{aligned}$$

Hence, if the set $U = \{1, k, k+1\}$, the counter example constructed in the proof of Lemma 34 does not work. This gives rise to the question of whether there exists $z \in \Sigma^*$ which has more than one duplication root if the deduplication set is $\{1, k, k+1\}$. We answer this question in lemma 35 and corollary 36 below

Lemma 35 *Given $|\Sigma| \geqslant 2$, $k > 2$ and $U = \{1, k\}$, there exists a $z \in \Sigma^*$ with $|R_U(z)| > 1$.*

Proof: Consider $z = a_1 a_2 \cdots a_k a_1 a_2 \cdots a_k$. Let $a_1 = a_k = a$. Then z can be rewritten as

$$\begin{aligned} z &= aa_2 \cdots aaa_2 \cdots a \\ &= \{aa_2 \cdots a\}^2 \\ &= aa_2 \cdots a_{k-1} a^2 a_2 \cdots a_{k-1} a. \end{aligned} \tag{8}$$

Deduplicating length $2k$ square in z first gives

$$\{aa_2 \cdots a\}^2 \underset{U}{\Longleftarrow} aa_2 \cdots a = y.$$

Deduplicating, length 2 square in z first gives

$$aa_2 \cdots aaa_2 \cdots a_{k-1} a \underset{U}{\Longleftarrow} aa_2 \cdots aa_2 \cdots a_{k-1} a = x.$$

Since $k > 2$ and $|\Sigma| \geqslant 2$, x cannot be further deduplicated in general to get y.

As $x \neq y$, we have constructed a z which has more than one duplication root. ■

An immediate corollary follows from the proof of Lemma 35

Corollary 36 *Given $|\Sigma| \geqslant 2, k > 1$, for any $V \subseteq \{i : i > k\}$, if $U = \{1, k\} \cup V$, then for some $z \in \Sigma^*$ $|R_U(z)| > 1$.*

Lemma 37 *Given $|\Sigma| \geqslant 3$, for $U = \{1,2\} \bigcup V$ and $|U| > 2$, where $V \subseteq \{4, 5, \cdots\}$, then for some $z \in \Sigma^*$, $|R_U(z)| > 1$.*

Proof: Let a, b, c be distinct symbols $\in \Sigma$ and $m = \min_{p \in V} p$.

Consider $z = ab^{m-3}caab^{m-3}ca$. z can be rewritten as $z = \{ab^{m-3}ca\}^2$. Deduplicating this square of length $2m$ first in z, we can get the following root

$$z = \{ab^{m-3}ca\}^2 \underset{U}{\Longleftarrow} ab^{m-3}ca \overset{*}{\underset{U}{\Longleftarrow}} abca = x.$$

However, z can also be rewritten as $z = ab^{m-3}c\{a\}^2b^{m-3}ca$, deduplicatiing this square of length 2 in z, we can get the following root

$$z = ab^{m-3}c\{a\}^2b^{m-3}ca \underset{U}{\Longleftarrow} ab^{m-3}cab^{m-3}ca \overset{*}{\underset{U}{\Longleftarrow}} abcabca = y.$$

y cannot be further deduplicated since the only sqaure in y is $\{abc\}^2$, which is of length 3 and $3 \notin U$. Hence, $|R_U(z)| > 1$. ∎

Lemma 38 *Given $|\Sigma| \geqslant 3$, for $U = \{1,2,3\} \bigcup V$ and $|U| > 3$, where $V \subseteq \{4,5,6\cdots\}$, then for some $z \in \Sigma^*$, $|R_U(z)| > 1$.*

Proof: Let a, b, c be 3 distinct symbols $\in \Sigma$. Consider the string $z = ab^mcbab^mcbc$, where $m = \min_{p\in V} p - 3$. z can be rewritten as $\{ab^mcb\}^2c$. One possible way of deduplicating z is

$$\{ab^mcb\}^2c \underset{U}{\Longleftarrow} ab^mcbc \overset{*}{\underset{U}{\Longleftarrow}} abcbc \underset{U}{\Longleftarrow} abc.$$

z can also be rewritten as $z = ab^mcbab^{m-1}\{bc\}^2$. Another possible way of deduplicating z can be

$$ab^mcbab^{m-1}\{bc\}^2 \underset{U}{\Longleftarrow} ab^mcbab^mc \overset{*}{\underset{U}{\Longleftarrow}} abcbabc.$$

Hence $|R_U(z)| > 1$. ∎

Lemma 39 *Given $|\Sigma| \geqslant 3$, $k > 1$, $m > 1$ and $U = \{k, k+m\}$, there exists a $z \in \Sigma^*$ with $|R_U(z)| > 1$.*

Proof: Consider

$$z = a_1a_2\cdots a_m\cdots a_{k+m}a_1a_2\cdots a_m\cdots a_{k+m}a_{m+1}\cdots a_{k+m-1}.$$

Let $v = a_1a_2a_3\cdots a_{k+m}$ be squarefree [20] and $a_m = a_{k+m}$. z can be rewritten as

$$\begin{aligned} z &= a_1a_2\cdots a_m\cdots a_{k+m-1}a_ma_1a_2\cdots \\ &\quad a_m\cdots a_{k+m-1}a_ma_{m+1}\cdots a_{k+m-1} \\ &= \{a_1a_2\cdots a_m\cdots a_{k+m-1}a_m\}^2a_{m+1}\cdots a_{k+m-1} \\ &= a_1a_2\cdots a_{k+m-1}a_ma_1a_2\cdots a_{m-1}\{a_ma_{m+1}\cdots a_{k+m-1}\}^2 \end{aligned} \tag{9}$$

As is evident in (9), there are two squares in z, one of which is of length $2k$ and the other is of length $2k+2m$. Deduplicating the square of length $2k+2m$ in z first gives

$$\begin{aligned} &\{a_1a_2\cdots a_m\cdots a_{k+m-1}a_m\}^2a_{m+1}\cdots a_{k+m-1} \\ &\underset{U}{\Longleftarrow} a_1a_2\cdots a_m\cdots a_{k+m-1}a_ma_{m+1}\cdots a_{k+m-1} \\ &\underset{U}{\Longleftarrow} a_1a_2\cdots a_m\cdots a_{k+m-1} = y. \end{aligned}$$

Deduplicating the square of length $2k$ first gives

$$\begin{aligned} &a_1a_2\cdots a_{k+m-1}a_ma_1a_2\cdots a_{m-1}\{a_ma_{m+1}\cdots a_{k+m-1}\}^2 \\ &\underset{U}{\Longleftarrow} a_1a_2\cdots a_{k+m-1}a_ma_1a_2\cdots a_{k+m-1} = x. \end{aligned}$$

Observe that $|x| = 2k+2m-1$ and $|y| = k+m-1$. If $m \neq \lambda k$ for some $\lambda \geqslant 1$ and there is a length $2k$ square in x, then for any $w \in D_U^{-*}(x)$, $|w| \neq |y|$.

If $m = \lambda k$, we can remove squares of length $2k$ from x to get to the length $k+m-1$, however we will show that if we can do so then $v = a_1a_2\cdots a_{k+m}$ has a repeat which is a contradiction.

We analyse the following two cases:

1) $\lambda = 1$: $v = a_1a_2\cdots a_k\cdots a_{2k-1}a_{2k}$. $x = a_1a_2\cdots a_ka_{k+1}\cdots a_{2k-1}a_{2k}a_1a_2\cdots a_{2k-1}$, since v is squarefree, for a square of length $2k$ to exist in x, it must start after the index 1 in x. Let it start at index μ ($2k \geqslant \mu > 1$) in x. We have the following tow cases:
   a) $1 < \mu \leqslant k$: Here, $a_\mu = a_{\mu+k}, a_{\mu+1} = a_{\mu+k+1}, \cdots, a_k = a_{2k}, a_{k+1} = a_1, \cdots, a_{k+\mu-1} = a_{\mu-1}$.
   b) $k < \mu \leqslant 2k$: Here, $a_\mu = a_{\mu-k}, a_{\mu+1} = a_{\mu-k+1}, \cdots, a_{2k} = a_k, a_1 = a_{k+1}, \cdots, a_{\mu-k-1} = a_{\mu-1}$.

   Both the cases above imply that, $a_i = a_{k+i}\ \forall\ i \leqslant k$, hence v has a repeat which is a contradiction as v is assumed to be squarefree.
2) $\lambda > 1$: Since v is squarefree, for a square of length $2k$ to exist in x, it must start after the index $(\lambda-1)k+1$ and end before the index $(\lambda+3)k$ in x. Let it start at index $(\lambda-1)k+\mu$. We have the following two cases for the range of μ:
   a) $1 < \mu \leqslant k$: Here, we will have $a_{(\lambda-1)k+\mu} = a_{\lambda k+\mu}, a_{(\lambda-1)k+\mu+1} = a_{\lambda k+\mu+1}, \cdots, a_{\lambda k} = a_{(\lambda+1)k}, a_{\lambda k+1} = a_1, \cdots, a_{\lambda k+\mu-1} = a_{\mu-1}$. Using these set of equalities, if $\alpha_1 = a_1a_2\cdots a_{\mu-1}, \beta_1 = a_{(\lambda-1)k+\mu}\cdots a_{\lambda k}, \gamma_1 = a_\mu a_{\mu+1}\cdots a_{(\lambda-1)k+\mu-1}$, then $v = \alpha_1\gamma_1\beta_1\alpha_1\beta_1$. Let $\alpha_1 = \alpha_1'a_{\mu-1}$ and $\beta_1 = \beta_1'a_{\lambda k}$, then $x = v\alpha_1\gamma_1\beta_1\alpha_1\beta_1' = \alpha_1\gamma_1\beta_1\alpha_1\beta_1\alpha_1\gamma_1\beta_1\alpha_1\beta_1'$. Deduplicating the square $\{\beta_1\alpha_1\}^2$ of length $2k$ from x gives $x^* = \alpha_1\gamma_1\beta_1\alpha_1\gamma_1\beta_1\alpha_1\beta_1'$. It is notable here that $x = \alpha_1\delta_1\beta_1\alpha_1\delta_1\beta_1'$, where $\delta_1 = \gamma_1\beta_1\alpha_1$.
   Let $x_1^* = \alpha_1x_1'a_{\mu-1}\beta_1'$, where $x_1' = \gamma_1\beta_1\alpha_1\gamma_1\beta_1\alpha_1'$.
   b) $k < \mu \leqslant 2k$: Here, we will have $a_{(\lambda-1)k+\mu} = a_{\mu-k}, a_{(\lambda-1)k+\mu+1} = a_{\mu-k+1}, \cdots, a_{(\lambda+1)k} = a_k, a_1 = a_{k+1}, \cdots, a_{\mu-k-1} = a_{\mu-1}$. Using these set of equalities, if $\alpha_2 = a_1a_2\cdots a_{\mu-k-1}, \beta_2 = a_{\mu-k}\cdots a_k, \gamma_2 = a_\mu a_{\mu+1}\cdots a_{(\lambda-1)k+\mu-1}$. Therefore, $v = \alpha_2\beta_2\alpha_2\gamma_2\beta_2$. Let $\beta_2 = \beta_2'a_k$, then $x = v\alpha_2\beta_2\alpha_2\gamma_2\beta_2' = \alpha_2\beta_2\alpha_2\gamma_2\beta_2\alpha_2\beta_2\alpha_2\gamma_2\beta_2'$. Deduplicating the square $\{\beta_2\alpha_2\}^2$ of lengfth $2k$ from x gives $x^* = \alpha_2\beta_2\alpha_2\gamma_2\beta_2\alpha_2\gamma_2\beta_2'$. It is notable here that $x = \alpha_2\delta_2\beta_2\alpha_2\delta_2\beta_2'$, where $\delta_2 = \beta_2\alpha_2\gamma_2$.
   Let $x_2^* = \alpha_2\beta_2x_2'$, where $x_2' = \alpha_2\gamma_2\beta_2\alpha_2\gamma_2\beta_2'$.

Now, we see in both case $a)$ and $b)$ above, for x_i^* ($i \in \{1,2\}$) to have a repeat, x_i' should have a repeat. x_1' is similar in structure to x with δ_1 replaced by β_1, α_1 replaced by γ_1 and β_1 replaced by α_1 and x_2' is similar in structure to x with δ_2 replaced by γ_2. The length of x_i' is $2|\lambda k| - 1$.

Depending on the case in which x_i' falls, we can apply the same method of deduplication on x_i' now to get a new x_1' or x_2', until the length of the new x_1' or x_2' is $4k-1$, where we will land up in the case of $\lambda = 1$, by which we find a repeat of length $2k$ in the first $2k$ symbols of this final x_1' or x_2'. Since the string formed by first $2k$ symbols in the final x_1' or x_2' is a substring of v, therefore v has a repeat which is a contradiction. ■

Now, we state a Theorem about non-uniqueness of root for an arbitrary non-empty set U of deduplication lengths.

Theorem 40 *Given Σ with $|\Sigma| \geqslant 3$ and a non-empty set U, there exists a sequence $z \in \Sigma^*$ with $|R_U(z)| > 1$ given $U \neq \{k\}$ for some $k \geqslant 1$, $U \neq \{1,2\}$ and $U \neq \{1,2,3\}$.*

Proof:

- $1 \in U$ and $U \neq \{1\}$ or $U \neq \{1,2\}$ or $\{1,2,3\}$: Theorem 40 holds from corollary 36, Lemma 37 and 38.
- $1 \notin U$: Let p and q be the first and second minimum values in the set U respectively. If $q-p=1$, then Theorem 1 holds from Lemma 34 by putting $k=p$, otherwise it holds from Lemma 39 by putting $k=p$.

■

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] C. T. Clelland, V. Risca, and C. Bancroft, "Hiding messages in DNA microdots," *Nature*, vol. 399, no. 6736, pp. 533–534, 06 1999.

[4] F. Farnoud, M. Schwartz, and J. Bruck, "The capacity of string-duplication systems," *IEEE Trans. Inform. Theory*, accepted.

[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] J. W. Fondon and H. R. Garner, "Molecular origins of rapid and continuous morphological evolution," *Proceedings of the National Academy of Sciences*, vol. 101, no. 52, pp. 18 058–18 063, 2004.

[7] D. Haughton and F. Balado, "BioCode: Two biologically compatible algorithms for embedding data in non-coding and coding regions of DNA," *BMC Bioinformatics*, vol. 14, no. 1, pp. 1–16, 2013.

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

[9] K. A. S. Immink, *Coding Techniques for Digital Recorders*. Prentice-Hall, 1991.

[10] 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.

[11] D. C. Jupiter, T. A. Ficht, J. Samuel, Q.-M. Qin, and P. de Figueiredo, "DNA watermarking of infectious agents: Progress and prospects," *PLoS Pathog*, vol. 6, no. 6, p. e1000950, 06 2010.

[12] A. Kato and K. Zeger, "On the capacity of two-dimensional run-length constrained channels," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1527–1540, Jul. 1999.

[13] E. S. Lander, L. M. Linton, B. Birren, C. Nusbaum, M. C. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh *et al.*, "Initial sequencing and analysis of the human genome," *Nature*, vol. 409, no. 6822, pp. 860–921, 2001.

[14] P. Leupold, "Duplication roots," *Lecture Notes in Computer Science*, vol. 4588, pp. 290–299, 2007.

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

[16] D. Lind and B. H. Marcus, *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, 1985.

[17] 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.

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

[19] R. M. Roth and P. H. Siegel, "Lee-metric BCH codes and their application to constrained and partial-response channels," *IEEE Trans. Inform. Theory*, vol. 40, no. 4, pp. 1083–1096, Jul. 1994.

[20] A. Thue, "Uber unendliche zeichenreihen," *Kra. Vidensk. Selsk. Skrifter. I. Mat.-Nat. Kl., Cristiana*, vol. 7, 1906.

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

[22] 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.

[23] N. Yachie, Y. Ohashi, and M. Tomita, "Stabilizing synthetic data in the DNA of living organisms," *Systems and Synthetic Biology*, vol. 2, no. 1-2, pp. 19–25, 2008.

[24] S. M. H. T. Yazdi, H. M. Kiah, E. R. Garcia, J. Ma, H. Zhao, and O. Milenkovic, "DNA-based storage: Trends and methods," *arXiv preprint: http://arxiv.org/abs/1507.01611*, 2015.

Appendix

We provide a short proof of (4). We need to estimate the largest eigenvalue of $A_q(k)$ from (3), i.e., to estimate the largest root λ of its characteristic polynomial

$$\chi_{A_q(k)}(x) = \frac{x^{k+2} - qx^{k+1} + q - 1}{x-1}.$$

Since this largest root is strictly greater than 1, we can alternatively find the largest root of the polynomial

$$f(x) = x^{k+2} - qx^{k+1} + q - 1.$$

We shall require the following simple bounds. Taking the first term in the Taylor expansion of e^x, and the error term, we have for all $x > 0$,

$$e^x = 1 + xe^{x'},$$

for some $x' \in [0, x]$. Since $x > 0$ and e^x is increasing, we have

$$e^x = 1 + xe^{x'} \leqslant 1 + xe^x,$$

or alternatively,

$$1 - e^x \geqslant -xe^x. \tag{10}$$

Similarly, taking the first two terms of the Taylor expansion, for all $x > 0$, we get the well known

$$e^x > 1 + x. \tag{11}$$

We return to the main proof. In the first direction, let us first examine what happens when we set

$$x = qe^{-\frac{q-1}{q^{k+2}}}.$$

Then

$$\begin{aligned} f(x) &= q^{k+2} e^{-\frac{q-1}{q^{k+2}}(k+2)} - q^{k+2} e^{-\frac{q-1}{q^{k+2}}(k+1)} + q - 1 \\ &= q^{k+2} e^{-\frac{q-1}{q^{k+2}}(k+2)} \left(1 - e^{\frac{q-1}{q^{k+2}}}\right) + q - 1 \\ &\overset{(a)}{\geqslant} (q-1)\left(1 - e^{-\frac{q-1}{q^{k+2}}(k+1)}\right) \\ &> 0, \end{aligned}$$

where (a) follows by an application of (10).

In the other direction, we examine the value of $f(x)$ when we set

$$x = qe^{-\frac{q-1}{q^{k+2}}\alpha},$$

where α is a constant depending on q and k. To specify α we recall $W(z)$, $z \geqslant -\frac{1}{e}$, denotes the Lambert W-function, defined by

$$W(z)e^{W(z)} = z.$$

We define

$$\alpha = \frac{W\left(-\frac{q-1}{q^{k+2}}(k+2)\right)}{-\frac{q-1}{q^{k+2}}(k+2)} = e^{-W\left(-\frac{q-1}{q^{k+2}}(k+2)\right)}.$$

Except for $k = 1$ and $q = 2$, for all other values of the parameters we have

$$-\frac{q-1}{q^{k+2}}(k+2) \geqslant -\frac{1}{e},$$

rendering the use of the W function valid. We also note that for these parameters we have $\alpha \geqslant 1$.

Let us calculate $f(x)$,

$$\begin{aligned} f(x) &= q^{k+2}e^{-\frac{q-1}{q^{k+2}}(k+2)\alpha} - q^{k+2}e^{-\frac{q-1}{q^{k+2}}(k+1)\alpha} + q - 1 \\ &= q^{k+2}e^{-\frac{q-1}{q^{k+2}}(k+2)\alpha}\left(1 - e^{\frac{q-1}{q^{k+2}}\alpha}\right) + q - 1 \\ &\overset{(a)}{<} (q-1)\left(1 - \alpha e^{-\frac{q-1}{q^{k+2}}(k+2)\alpha}\right) \\ &\overset{(b)}{=} (q-1)(1-1) = 0, \end{aligned}$$

where (a) follows by an application of (11), and (b) follows by substituting the value of α.

In summary, $f(x)$ is easily seen to be decreasing in the range $[1, (k+1)q/(k+2)]$, and increasing in the range $[(k+1)q/(k+2), \infty)$, and therefore, its unique largest root λ is in the range

$$qe^{-\frac{q-1}{q^{k+2}}\alpha} \leqslant \lambda \leqslant qe^{-\frac{q-1}{q^{k+2}}}.$$

It is easy to verify that $\alpha = 1 + o(1)$, where $o(1)$ denotes a function decaying to 0 as $k \to \infty$. Hence,

$$\lambda = qe^{-\frac{q-1}{q^{k+2}}(1+o(1))},$$

and therefore

$$\begin{aligned} \mathsf{cap}(\mathrm{RLL}_q(0,k)) &= \log_2 \lambda \\ &= \log_2 q - \frac{(q-1)\log_2 e}{q^{k+2}}(1 + o(1)). \end{aligned}$$