On Codes for the Noisy Substring Channel

Yonatan Yehezkeally, *Member, IEEE* and Nikita Polyanskii, *Member, IEEE*

***Abstract*—We consider the problem of coding for the substring channel, in which information strings are observed only through their (multisets of) substrings. Due to existing DNA sequencing techniques and applications in DNA-based storage systems, interest in this channel has renewed in recent years. In contrast to existing literature, we consider a noisy channel model where information is subject to noise *before* its substrings are sampled, motivated by in-vivo storage.**

We study two separate noise models, substitutions or deletions. In both cases, we examine families of codes which may be utilized for error-correction and present combinatorial bounds on their sizes. Through a generalization of the concept of repeat-free strings, we show that the added required redundancy due to this imperfect observation assumption is sublinear, either when the fraction of errors in the observed substring length is sufficiently small, or when that length is sufficiently long. This suggests that no asymptotic cost in rate is incurred by this channel model in these cases. Moreover, we develop an efficient encoder for such constrained strings in some cases.

Finally, we show how a similar encoder can be used to avoid formation of secondary-structures in coded DNA strands, even when accounting for *imperfect* structures.

***Index Terms*—DNA storage, Sequence reconstruction, Error-correcting codes, Insertion/deletion-correcting codes, Constrained codes**

I. Introduction

DNA as a medium for data storage offers high density and longevity, far greater than those of electronic media [1]. Among its applications, data storage in DNA may offer a protected medium for long-period data storage [2], [3]. In particular, it has recently been demonstrated that storage in the DNA of living organisms (henceforth, *in-vivo* DNA storage) is now feasible [4]; the envelope of a living cell affords some level of protection to the data, and even offers propagation, through cell replication. Among its varied usages, in-vivo DNA storage allows watermarking genetically modified organisms (GMOs) [5]–[7] to protect intellectual property, or labeling research material [3], [8]. It may even conceal sensitive information, as it may appear indistinguishable from the organism's own genetic information [9].

Similarly to other media, information stored over this medium is subjected to noise due to mutations, creating errors in data, which accumulate over time and replication cycles. Examples of such noise include symbol insertions or deletion, in addition to substitutions (point-mutations) [10], [11]; the latter is the focus of the vast majority of classical error-correction research, and the former have also been studied. Interestingly, however, the very methods we currently use to store and later retrieve data from DNA inherently introduce new constraints on information reconstruction. While desired sequences may be synthesized (albeit, while suffering from errors, e.g., substitution noise), the process of DNA sequencing, i.e., retrieving the DNA sequence of an organism, only observes that sequence as the (likely incomplete) multiset of its substrings (practically, up to a certain substring length) [12]. Thus, information contained in the order of these substrings might be irrevocably lost. As a result of these constraints, conventional and well-developed error-correction approaches cannot simply be applied.

To overcome these effects, one approach in existing literature is to add redundancy in the form of indexing, in order to recover the order of substrings (see, e.g., [13]–[15]). A different approach, potentially more applicable to in-vivo DNA storage, is to add redundancy in the form of constraints on the long information string, such that it can be uniquely reconstructed by knowledge of its substrings of a given length (or range of lengths). The combinatorial problem of recovering a sequence from its substrings has attracted attention in recent years [16]–[23], and coding schemes involving only these substrings (including the incidence frequency of each substring) were studied [12], [15], [24]–[26].

However, works dedicated to overcoming this obstacle, inherent to the technology we use, have predominantly focused on storage outside of living cells (i.e., *in-vitro* DNA storage). Likewise, works focused on error-correction for in-vivo DNA data storage (e.g., [27]–[29]) have disregarded the technical process by which data is to be read. However, in real applications varied distinct noise mechanisms act on stored data concurrently. Hence, in practice, both sets of challenges have to be collectively overcome in order to robustly store information using in-vivo DNA.

The aim of this work is to protect against errors in the information string (caused by mutations over the replication process of cells), when channel outputs are given by the multisets of their substrings, of a predetermined length, rather than entire strings. This models the process of DNA sequencing, once information needs to be read from the medium. We shall study the required redundancy of this model, and devise coding

Manuscript received 25 September 2023; revised 6 February 2024; accepted 24 March 2024. This work has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 801434). The work of Yonatan Yehezkeally was supported by the Alexander von Humboldt Foundation under a Carl Friedrich von Siemens Post-Doctoral Research Fellowship. The work of Nikita Polyanskii was supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) under Grant No. WA3907/1-1. An earlier version of this paper was presented in part at the 2021 IEEE International Symposium on Information Theory (ISIT) [DOI: 10.1109/ISIT45174.2021.9517943]. *(Corresponding author: Yonatan Yehezkeally.)*

Yonatan Yehezkeally is with the Institute for Communications Engineering, School of Computation, Information and Technology, Technical University of Munich (TUM), 80333 Munich, Germany (e-mail: yonatan.yehezkeally@tum.de). Nikita Polyanskii is with the Research and Development department, IOTA Foundation, Berlin, Germany.

strategies, under the assumption of two different error types: substitution and deletion noise.

Another application for this line of research is secondary-structure avoidance. Secondary structures are complex spatial structures that can form in a chemically active single-stranded DNA, as a result of the strand folding upon itself to allow two sub-segments to bond via complementary-base-pair hybridization [30]. Their formation renders the DNA strand chemically inactive and is therefore detrimental for sequencing and DNA-based computation, hence a number of recent works have looked to avoid them through coding [31]–[34]. Herein we focus on relatively long structures, but unlike recent works, we do not consider only perfect structures, but also attempt to avoid ones which contain impairments, i.e., imperfect structures. We show that this problem is closely connected to the above-described channel; thus, we are able to also present an efficient encoder for this setting.

The paper is organized as follows. In Section II, we discuss the main contribution of this paper, in context of related works. In Section III we then present necessary notation. Then, in Section IV we study the suggested model with substitution errors, and in Section V with deletion errors. Finally, in Section VI we develop an encoder for avoiding the formation of even imperfect secondary structures.

II. Related works and main contribution

Given a string of length n, the problem of reconstructing it from the multiset of (all-, or, in some works, most-) its substrings of a fixed length $\ell \leqslant n$, has been studied in literature. Assuming no errors occur in $\boldsymbol{x}$ prior to sampling of its substrings, the problem of interest is identifying a set of constraints on the information string, equivalent or sufficient, for such reconstruction to be achievable.

It was observed in [16] that under certain circumstances, distinct information strings in which repetitions of ℓ-substrings appear in different positions, exhibit the same multisets of $(\ell+1)$-substrings. These observations indicate that care must be taken when including code-words which contain repeating ℓ-substrings (where observations are made via the multiset of ℓ'-substrings, for some $\ell' \leqslant \ell+1$). On the other hand, if every ℓ-substring of $\boldsymbol{x}$ is unique, then $\boldsymbol{x}$ is uniquely reconstructible from the multiset of its $(\ell+1)$-substrings (and in fact, ℓ'-substrings, for all $\ell' > \ell$), as evident from a greedy reconstruction algorithm (which at each stage searches for the next/previous character in the information string). This observation motivates the study of *repeat-free strings*; $\boldsymbol{x}$ is said to be ℓ-repeat-free if every ℓ-substring of $\boldsymbol{x}$ is unique (put differently, if $\boldsymbol{x}$ is of length n, then it contains $n-\ell+1$ distinct ℓ-substrings).

Focus on repeat-free strings is further justified by the following results. It was observed in [19], via introduction of *profile vectors*, that over an alphabet of size q, where the length n of strings grows, if $\ell < \frac{\log_q(n)}{1+\epsilon}$ then the rate of all possible ℓ-substring multisets vanishes. Conversely, it was demonstrated in [21] using probabilistic arguments that the asymptotic redundancy of the code-book consisting of all ℓ-repeat-free strings of length n (which, as noted above, is an upper bound for the redundancy of a code assuring reconstruction from $(\ell+1)$-substrings), is $O(n^{2-\ell/\log_q(n)})$; thus, when $\ell > (1+\epsilon)\log_q(n)$, the rate of repeat-free strings alone is 1.

In this paper, we extend the setting of previous works by allowing information strings to suffer a bounded number of errors, prior to the sampling of their substrings. We study this model under two separate error models: substitution (Hamming) errors, and deletion errors. In both cases we show (see Theorems 13 and 27) that when $\ell > (1+\epsilon)\log_q(n)$ and the fraction of errors in the substring length ℓ is sufficiently small, the rate of generalized repeat-free strings, dubbed *resilient-repeat-free*, suffers no penalty from the process of sampling, or from the presence of noise (when compared to the results of [21]); i.e., the required added redundancy is sub-linear. In the case of Hamming noise, we also show that when the fraction of errors is too large, resilient-repeat-free strings do not exist. However, it is left for future works to determine the precise transition between the two regimes. Further, we develop an efficient encoder for resilient-repeat-free sequences (see Algorithm 1), although our encoder does not output sequences of a fixed length n, but rather only guarantees that the output is of length *at most* n.

It should be noted that [20] presented almost explicit encoding/decoding algorithms for codes with a similar noise model. However, in that paper's setting, substitution noise affects individual substrings *after* sampling; the codes it constructs are capable of correcting a constant number of errors in each substring, but requires the assumption that errors do not affect the same information symbol in a majority of the substrings that reflect it. Therefore, its setting is incompatible with the one considered herein, whereby each error occurring *before* sampling affects ℓ consecutive substrings. [22] also developed codes with full rate, capable of correcting a fixed number of errors, occurring in substrings independently after sampling. It replaced the aforementioned restriction by a constraint on the number of total erroneous substrings, which is at most logarithmic in the information string's length. Hence, the total number of errors in its setting remains asymptotically smaller than the one incurred in the setting considered here.

Finally, as mentioned above we exploit the similarity between the aforementioned setting and channel model and the problem of avoiding secondary structures. We focus on hairpin-loop structures with long stems (scaling logarithmically in the length of the sequence), and unlike recent works [31]–[34] the encoder we develop prevents the formation of such structures even when the underlying complementary-base-pair hybridization (in a region called the *stem* of the structure) is imperfect, that is, it contains at most a δ-fraction of mismatched nucleobases (which cannot stably hybridize), while asymptotically achieving full rate.

III. Preliminaries

Let Σ^* be the set of finite strings over an alphabet Σ, which for convenience we assume to be a finite unital ring of size q (e.g., $\mathbb{Z}_q$). For $\boldsymbol{x} = x(0)x(1)\cdots x(n-1) \in \Sigma^*$, we let $|\boldsymbol{x}| = n$ denote the *length* of $\boldsymbol{x}$. We note that indices in the sequel are numbered $0, 1, \ldots$. For $\boldsymbol{x}, \boldsymbol{y} \in \Sigma^*$, we let $\boldsymbol{x}\boldsymbol{y}$ be their concatenation. For $I \subseteq \mathbb{N}$ (we follow the convention $0 \in \mathbb{N}$) and $\boldsymbol{x} \in \Sigma^*$, we denote by $\boldsymbol{x}_I$ the restriction of $\boldsymbol{x}$ to indices in I (excluding any indices $|\boldsymbol{x}| \leqslant i \in I$), ordered according to the naturally inherited order on I.

We let $|A|$ denote the size of a finite set A. For a code $C \subseteq \Sigma^n$, we define its *redundancy* $\operatorname{red}(C) \triangleq n - \log_q |C|$, and *rate* $R(C) \triangleq \frac{1}{n} \log_q |C| = 1 - \frac{\operatorname{red}(C)}{n}$.

For $n \in \mathbb{N}$, denote $[n] \triangleq \{0, 1, \ldots, n-1\}$. Although perhaps confusable, for $m \leqslant n \in \mathbb{N}$ we use the common notation $[m, n] \triangleq \{m, m+1, \ldots, n\}$. We shall interpret $\boldsymbol{x}_I$ as enumerated by $[|I|]$, i.e., $x_I(0) = x(\min I)$, etc. Where it is convenient, we will also assume $I \subseteq \mathbb{N}$ to be enumerated by $[|I|]$, such that the order of elements is preserved; i.e., $I = \{I(i) : i \in [|I|]\}$, and for all $i \in [|I| - 1]$ one has $I(i) < I(i+1)$. Under this convention we have, e.g., $x_I(0) = x(I(0))$. We follow the standard group notation in denoting for $j \in \mathbb{N}$ and $I \subseteq \mathbb{N}$, the *coset* $j + I \triangleq \{j + i : i \in I\}$.

Example 1 *Consider the string* $\boldsymbol{x} = 0000111101100101$ *of length* $n = 16$*, and the set* $I = [7, 10] = \{7, 8, 9, 10\} = 7 + [4]$*. Then,* $\boldsymbol{x}_I = 1011$*, and in particular* $\boldsymbol{x}_I(1) = \boldsymbol{x}(I(1)) = \boldsymbol{x}(8) = 0$.

For $\boldsymbol{x} \in \Sigma^*$ and $i, \ell \in \mathbb{N}$, where $i + \ell \leqslant |\boldsymbol{x}|$, we say that $\boldsymbol{x}_{i+[\ell]}$ is the length-ℓ *substring* of $\boldsymbol{x}$ at index i, or ℓ*-mer* (at index i) for short. Using notation from [16], for $\boldsymbol{x} \in \Sigma^*$ and $\ell \in \mathbb{N}$ we denote the multiset of ℓ-mers of $\boldsymbol{x}$ by

$$Z_\ell(\boldsymbol{x}) \triangleq \{\!\{ \boldsymbol{x}_{i+[\ell]} : 0 \leqslant i \leqslant |\boldsymbol{x}| - \ell \}\!\}.$$

We follow [21] in denoting the set of ℓ*-repeat-free* strings

$$\mathcal{RF}_\ell(n) \triangleq \{ \boldsymbol{x} \in \Sigma^n : i < j \implies \boldsymbol{x}_{i+[\ell]} \neq \boldsymbol{x}_{j+[\ell]} \}.$$

We can now more formally state the objectives of Sections IV and V. Assuming an underlying error model, known in context but yet to be determined, we let $B_t(\boldsymbol{x})$, for some $\boldsymbol{x} \in \Sigma^*$, be the set of strings $\boldsymbol{y} \in \Sigma^*$ which may be the product of at most t errors occurring to $\boldsymbol{x}$. Using this notation, our aim shall be to study and design codes $C \subseteq \Sigma^n$, such that given $\boldsymbol{x} \in C$ and $\boldsymbol{y} \in B_t(\boldsymbol{x})$, for some fixed (or bounded) t, $\boldsymbol{x}$ can be uniquely reconstructed given only $Z_\ell(\boldsymbol{y})$. We shall study constraints, generalizing the notion of repeat-free strings, which allow unique reconstruction of $\boldsymbol{y}$, ascertain their required redundancy utilizing a probabilistic method, devise explicit encoding/decoding algorithms when possible, and state in Corollary 25 specific cases where this in turn allows reconstruction of $\boldsymbol{x}$.

Our analysis of the number of constrained sequences is aided in both the Hamming-errors and deletions cases by the following notation:

Definition 2 *For positive* $\ell \leqslant n$*, denote* $\binom{[n]}{\ell} \subseteq 2^{[n]}$ *the collection of* ℓ*-subsets of* $[n]$*. A pair of subsets* $(I, J) \in \binom{[n]}{\ell}^2$ *is said to be* observable *if* $I(k) < J(k)$ *for all* $k \in [\ell]$.

Given a string $\boldsymbol{x} \in \Sigma^n$, known from context, we will denote for an observable pair $(I, J) \in \binom{[n]}{\ell}^2$

$$\boldsymbol{u}_{I,J} \triangleq \boldsymbol{x}_I - \boldsymbol{x}_J \in \Sigma^\ell. \tag{1}$$

We also denote $\Gamma_I \triangleq \{(P, Q) : (P, Q)$ is observable, $(P \cup Q) \cap I \neq \emptyset\}$. To simplify notation, where some $\ell \leqslant n$ is also given, we shall abbreviate $\boldsymbol{u}_{i,j} \triangleq \boldsymbol{u}_{i+[\ell], j+[\ell]}$ and

$$\Gamma_i \triangleq \{(p, q) : \min(|i - p|, |i - q|) < \ell\}, \tag{2}$$

for any $0 \leqslant i < j \leqslant n - \ell$.

Then the following lemma will prove useful when bounding the redundancy of constrained sequences:

Lemma 3 *Take* $\ell \leqslant n$ *and an observable pair* $(I, J) \in \binom{[n]}{\ell}^2$*. Further, let* $\boldsymbol{x} \in \Sigma^n$ *be chosen uniformly at random. Then* $\boldsymbol{u}_{I,J}$ *is distributed uniformly and mutually independent of* $\{\boldsymbol{u}_{P,Q} : (P, Q) \notin \Gamma\}_I$.

Proof: First, since $\boldsymbol{u}_{I,J}$ is the image of $\boldsymbol{x}$ under a linear map (more precisely, a module homomorphism), the pre-image of any point is a coset of the map's kernel and, thus, of equal size; as a result, $\boldsymbol{u}_{I,J}$ is distributed uniformly on the map's range. Since (I, J) is observable, the map is surjective onto Σ^ℓ, hence the first part is completed.

Second, observe that $\boldsymbol{x}_I$ is independent of $\boldsymbol{x}_{[n] \setminus I}$, hence mutually independent of $\{\boldsymbol{u}_{P,Q} : (P, Q) \notin \Gamma_I\}$. Since given $\boldsymbol{x}_{[n] \setminus I}$, there exist a bijection between $\boldsymbol{x}_I$ and $\boldsymbol{u}_{I,J}$, the proof is concluded. ■

Finally, our proof strategy for bounding the redundancy of said constraints is based on Lovász's local lemma (LLL), which we slightly rephrase below.

Theorem 4 *[35, Th. 1.1] Let* $\{A_{i,j}\}_{i,j}$ *be events in a probability space* Ω*. If for all* i, j *there exist constants* $0 < f_{i,j} < 1$ *such that*

$$\Pr(A_{i,j}) \leqslant f_{i,j} \prod_{(p,q) \in \Gamma_{i,j}} (1 - f_{p,q}),$$

where $\Gamma_{i,j}$ *is such that the event* $A_{i,j}$ *is mutually independent of events* $\{A_{p,q} : (p, q) \notin \Gamma_{i,j}\}$*, then*

$$\Pr\Big(\Omega \setminus \bigcup\nolimits_{i,j} A_{i,j}\Big) \geqslant \prod_{i,j} (1 - f_{i,j}).$$

To the best of authors' knowledge, this application of the lemma is novel to the conference version of this work; it then also appeared in similar form in a concurrent journal version of [21]. Before continuing, we derive a corollary of Theorem 4 which is less tight, but more easily utilized.

Corollary 5 *Let* $\{A_{i,j}\}_{i,j}$ *be events in a probability space* Ω*. If for all* i, j *there exist constants* $0 < \phi_{i,j} < 1$ *such that*

$$\Pr(A_{i,j}) \leqslant \phi_{i,j} \exp\Big(-\sum_{(p,q) \in \Gamma_i} \phi_{p,q} - \phi_{i,j}\Big),$$

where Γ_i *is such that the event* $A_{i,j}$ *is mutually independent of events* $\{A_{p,q} : (p, q) \notin \Gamma_i\}$*, then*

$$\Pr\Big(\Omega \setminus \bigcup\nolimits_{i,j} A_{i,j}\Big) \geqslant \exp\Big(-\sum_{i,j} \phi_{i,j}\Big).$$

Proof: The proof follows from the inequality $1-f \geqslant e^{-f/(1-f)}$ for $0<f<1$. Then, denoting $f_{i,j} \triangleq \frac{\phi_{i,j}}{1+\phi_{i,j}}$ and $\Gamma_{i,j} \triangleq \Gamma_i$ for all i,j we have, for all i,j:

$$\begin{aligned} f_{i,j}\prod_{(p,q)\in\Gamma_{i,j}}(1-f_{p,q}) &= \phi_{i,j}(1-f_{i,j})\prod_{(p,q)\in\Gamma_{i,j}}(1-f_{p,q}) \\ &\geqslant \phi_{i,j}\exp\Big(-\phi_{i,j}-\sum_{(p,q)\in\Gamma_i}\phi_{p,q}\Big) \geqslant \Pr(A_{i,j}). \end{aligned}$$

It then follows from Theorem 4 that

$$\begin{aligned} \Pr\Big(\Omega\setminus\bigcup\nolimits_{i,j}A_{i,j}\Big) &\geqslant \prod_{i,j}\Big(1-\frac{\phi_{i,j}}{1+\phi_{i,j}}\Big) \\ &= \prod_{i,j}\frac{1}{1+\phi_{i,j}}, \end{aligned}$$

which, together with $1+\phi \leqslant e^{\phi}$ for all ϕ, concludes the proof. ■

Our aim in the next two sections will be to give a precise definition to the resilient-repeat-free constraint in the contexts of Hamming-errors and deletions respectively, apply Corollary 5 to bound their redundancies, then study explicit encoders (in Section IV) and the cases in which error-correcting codes can be embedded in these constraints.

IV. Substitution noise

In this section we consider substitution noise, with error balls $B^{\mathrm{s}}_t(\boldsymbol{x}) \triangleq \{\boldsymbol{y} : d_{\mathrm{H}}(\boldsymbol{x},\boldsymbol{y}) \leqslant t\}$, where $d_{\mathrm{H}}(\boldsymbol{x},\boldsymbol{y})$ denotes the Hamming distance between $\boldsymbol{x}$ and $\boldsymbol{y}$. Observe that the superscript s denotes *substitution* noise, and is not a parameter in this notation.

We present and study a family of repeat-free strings which are resilient to substitution errors:

Definition 6 *We say that $\boldsymbol{x}\in\Sigma^*$ is (t,ℓ)-*resilient repeat free *if the result of any t substitution errors to $\boldsymbol{x}$ is ℓ-repeat-free. More precisely, we define*

$$\mathcal{RRF}^{\mathrm{s}}_{t,\ell}(n) \triangleq \{\boldsymbol{x}\in\Sigma^n : B^{\mathrm{s}}_t(\boldsymbol{x}) \subseteq \mathcal{RF}_\ell(n)\}.$$

Throughout the paper, we shall abbreviate our notation to $\mathcal{RRF}^{\mathrm{s}}(n)$, given that t,ℓ are known from context.

Example 7 *The sequence $\boldsymbol{x} = 0000111101100101$ from Example 1 is 4-repeat-free, since all of its substrings of length 4 are unique. It is not, however, $(1,4)$-resilient-repeat-free, since after a single substitution one may derive $\boldsymbol{y} = 000011110\underline{0}100101$, and $\boldsymbol{y}_{7+[4]} = 1001 = \boldsymbol{y}_{10+[4]}$.*

A. Rate of resilient-repeat-free strings

In the following section we dedicate ourselves to study $\mathrm{red}(\mathcal{RRF}^{\mathrm{s}}(n))$, where t,ℓ are taken to be functions of n. In particular, we will be interested in developing sufficient (and to a lesser degree, necessary) conditions on t,ℓ that assure $R(\mathcal{RRF}^{\mathrm{s}}(n)) = 1-o_n(1)$.

Recall that [21] showed that if $\ell = a\log(n)+o(\log(n))$, then

$$R(\mathcal{RF}_\ell(n)) = \begin{cases} o_n(1), & a<1; \\ 1-o_n(1), & a>1. \end{cases}$$

Since $\mathcal{RRF}^{\mathrm{s}}_{t,\ell}(n) \subseteq \mathcal{RRF}^{\mathrm{s}}_{0,\ell}(n) = \mathcal{RF}_\ell(n)$, then with the above scaling of ℓ, having $a<1$ implies that $R(\mathcal{RRF}^{\mathrm{s}}_{t,\ell}(n)) = o_n(1)$ as well, for all t; we shall see that when $a>1$, then for sufficiently small t we still have $R(\mathcal{RRF}^{\mathrm{s}}_{t,\ell}(n)) = 1-o_n(1)$.

A particular notion that will aid in our analysis is the following: for $0<k\leqslant\ell$, denote

$$\mathcal{A}^{\ell}_t(k) \triangleq \big\{\boldsymbol{x}\in\Sigma^{\ell+k} : \exists\boldsymbol{y}\in B^{\mathrm{s}}_t(\boldsymbol{x}) : \boldsymbol{y}_{[\ell]} = \boldsymbol{y}_{k+[\ell]}\big\}.$$

We let $\pi^{\ell}_t(k) \triangleq q^{-(\ell+k)}\big|\mathcal{A}^{\ell}_t(k)\big|$ (i.e., $\pi^{\ell}_t(k) = \Pr\big(\boldsymbol{x}\in\mathcal{A}^{\ell}_t(k)\big)$ where $\boldsymbol{x}\in\Sigma^{\ell+k}$ is chosen uniformly at random). This notion captures the pertinent range of k, since as $k\geqslant\ell$ grows, $\pi^{\ell}_t(k)$ is clearly fixed and no longer changes with k. For convenience, when ℓ,t are known from context, we also abbreviate:

$$\pi \triangleq \pi^{\ell}_t(\ell); \qquad \pi' \triangleq \max_{0<k<\ell}\pi^{\ell}_t(k). \tag{3}$$

The usefulness of the notation in (3) is substantiated in the following theorem.

Theorem 8 *Let $\ell=\ell(n), t=t(n)$ be integer functions, and assume $t\leqslant\ell\leqslant n$. If for all sufficiently large n it holds that $3\ell^2\pi' + 2\ell n\pi \leqslant 1/e$, then*

$$\mathrm{red}(\mathcal{RRF}^{\mathrm{s}}(n)) = O\big(n\log(n)\pi' + n^2\pi\big).$$

Proof: As mentioned above, we shall rely on Corollary 5, for which we need to define the sets $\{A_{i,j}\}$, determine the constants $\{\phi_{i,j}\}$ and establish the independence property for the sets $\{\Gamma_i\}$. We define for all $0\leqslant i<j\leqslant n-\ell$ the sets

$$A_{i,j} \triangleq \big\{\boldsymbol{x}\in\Sigma^n : \exists\boldsymbol{y}\in B^{\mathrm{s}}_t(\boldsymbol{x}) : \boldsymbol{y}_{i+[\ell]} = \boldsymbol{y}_{j+[\ell]}\big\}.$$

Note that $\Sigma^n\setminus\mathcal{RRF}^{\mathrm{s}}(n) = \bigcup_{i,j}A_{i,j}$.

We let $\boldsymbol{x}\in\Sigma^n$ be chosen uniformly at random. Then $\Pr(\boldsymbol{x}\in A_{i,j}) = \pi^{\ell}_t(\min\{\ell,j-i\})$. Further,

$$|\mathcal{RRF}^{\mathrm{s}}(n)| = q^n\cdot\Pr(\boldsymbol{x}\in\mathcal{RRF}^{\mathrm{s}}(n)),$$

and hence

$$\mathrm{red}(\mathcal{RRF}^{\mathrm{s}}(n)) = -\log_q\Pr(\boldsymbol{x}\in\mathcal{RRF}^{\mathrm{s}}(n)).$$

Note that, in our notation, $\Pr(\boldsymbol{x}\in\mathcal{RRF}^{\mathrm{s}}(n)) = \Pr\Big(\boldsymbol{x}\notin\bigcup_{i,j}A_{i,j}\Big)$.

Recalling (2), we claim for $0\leqslant i<j\leqslant n-\ell$ that the event $\{\boldsymbol{x}\in A_{i,j}\}$ is mutually independent of the events $\{\{\boldsymbol{x}\in A_{p,q}\} : (p,q)\notin\Gamma_i\}$. Indeed, Lemma 3 then implies that $\boldsymbol{u}_{i,j}$ is mutually independent of $\{\boldsymbol{u}_{p,q} : (p,q)\notin\Gamma_i\}$. By abuse of notation, consider the mapping $U_{i,j} : \boldsymbol{x}\mapsto\boldsymbol{u}_{i,j}$; then $\boldsymbol{x}\in A_{i,j}$ if and only if $\boldsymbol{u}_{i,j}\in U_{i,j}B^{\mathrm{s}}_t\big(U^{-1}_{i,j}\mathbf{0}\big)$, where $U^{-1}_{i,j}\mathbf{0} = \{\boldsymbol{y} : U_{i,j}\boldsymbol{y} = \mathbf{0}\}$, $B^{\mathrm{s}}_t(A) = \bigcup_{\boldsymbol{y}\in A}B^{\mathrm{s}}_t(\boldsymbol{y})$, and $U_{i,j}A = \{U_{i,j}\boldsymbol{y} : \boldsymbol{y}\in A\}$. Since the sets $U_{i,j}B^{\mathrm{s}}_t\big(U^{-1}_{i,j}\mathbf{0}\big)$ depend only on t,i,j but not $\boldsymbol{x}$, the independence property holds.

Observe that the number of pairs $(p,q)\in\Gamma_i$ satisfying $|p-q|<\ell$ is over-counted as all choices of $\alpha\in[i-\ell+1,i+\ell-1]$ and $\beta\in[\alpha-\ell+1,\alpha+\ell-1]\setminus\{\alpha\}$ (then, $(p,q) = (\min\{\alpha,\beta\},\max\{\alpha,\beta\})$); in fact, this way one counts all pairs in $[i-\ell+1,i]$, and all pairs in $[i,i+\ell-1]$, twice (in fact, more pairs are counted twice, but the precise number is immaterial). I.e., that number is at most $(2\ell-1)(2\ell-2)-2\binom{\ell}{2}=(3\ell-2)(\ell-1)<3\ell^2$. The number of pairs $(p,q)\in\Gamma_i$ such that $|q-p|\geqslant\ell$ can also be counted as above, allowing $\beta\in[n]\setminus[\alpha-\ell+1,\alpha+\ell-1]$ (which at worst, when $\alpha\in\{0,n-1\}$, allows for $n-\ell+1$ distinct choices), i.e., it is at most $(2\ell-1)(n-\ell+1)<2\ell n$.

We shall apply an almost symmetric version of Corollary 5, where

$$\phi_{i,j}=\begin{cases} e\pi, & j-i\geqslant\ell,\\ e\pi', & j-i<\ell.\end{cases}$$

Then, for any i,j we observe

$$\begin{aligned}\phi_{i,j}\exp\Big(-\sum_{(p,q)\in\Gamma_i}\phi_{p,q}-\phi_{i,j}\Big) &> \phi_{i,j}e^{-(3\ell^2\pi'+2\ell n\pi)e}\\ &\geqslant \pi_t^\ell(\min\{\ell,j-i\})\\ &= \Pr(\boldsymbol{x}\in A_{i,j}),\end{aligned}$$

where the last inequality is justified by $3\ell^2\pi'+2\ell n\pi<1/e$, for large enough n. It follows from Corollary 5 that

$$\begin{aligned}\operatorname{red}(\mathcal{RRF}^{\mathrm{s}}(n)) &= -\log_q\Pr(\boldsymbol{x}\in\mathcal{RRF}^{\mathrm{s}}(n))\\ &= -\log_q\Pr\Big(\boldsymbol{x}\notin\bigcup\nolimits_{i,j}A_{i,j}\Big)\\ &\leqslant \log_q(e)\sum\nolimits_{i,j}\phi_{i,j}\\ &< e\log_q(e)\big(n\ell\pi'+n^2\pi\big),\end{aligned}$$

which concludes the proof. ■

Based on the last theorem, it is of interest to bound π,π' from above. Our strategy will be twofold. First, we will devise sufficient conditions for $\boldsymbol{x}\in\mathcal{RRF}^{\mathrm{s}}(n)$; second, we make tighten the resulting bounds by taking advantage of the periodicity implied for $\boldsymbol{y}\in B_t^{\mathrm{s}}(\boldsymbol{x})$, by $\boldsymbol{y}_{[\ell]}=\boldsymbol{y}_{k+[\ell]}$. To that end, we note the following result.

Lemma 9 *Take $t\leqslant\ell\leqslant n\in\mathbb{N}$, $\boldsymbol{x}\in\Sigma^n$. If for all $0\leqslant i<j\leqslant n-\ell$ it holds that*

$$d_{\mathrm{H}}\big(\boldsymbol{x}_{i+[\ell]},\boldsymbol{x}_{j+[\ell]}\big)>t+\max\{0,\min\{t,\ell-j+i\}\},$$

then $\boldsymbol{x}\in\mathcal{RRF}^{\mathrm{s}}(n)$.

Proof: The proof follows from applying the triangle inequality by cases on $(i+[\ell])\cap(j+[\ell])$. Assume to the contrary that there exist $\boldsymbol{y}\in B_t^{\mathrm{s}}(\boldsymbol{x})$ and $0\leqslant i<j\leqslant n-\ell$ such that $\boldsymbol{y}_{i+[\ell]}=\boldsymbol{y}_{j+[\ell]}$. Note that

$$\begin{aligned}d_{\mathrm{H}}\big(\boldsymbol{x}_{i+[\ell]},\boldsymbol{x}_{j+[\ell]}\big) &\leqslant d_{\mathrm{H}}\big(\boldsymbol{x}_{i+[\ell]},\boldsymbol{y}_{i+[\ell]}\big)+\\ &\quad d_{\mathrm{H}}\big(\boldsymbol{y}_{i+[\ell]},\boldsymbol{y}_{j+[\ell]}\big)+\\ &\quad d_{\mathrm{H}}\big(\boldsymbol{y}_{j+[\ell]},\boldsymbol{x}_{j+[\ell]}\big)\\ &= d_{\mathrm{H}}\big(\boldsymbol{x}_{i+[\ell]},\boldsymbol{y}_{i+[\ell]}\big)+\\ &\quad d_{\mathrm{H}}\big(\boldsymbol{y}_{j+[\ell]},\boldsymbol{x}_{j+[\ell]}\big).\end{aligned}$$

We continue by cases.

If $j-i\geqslant\ell$ then, since $(i+[\ell])\cap(j+[\ell])=\emptyset$, then $d_{\mathrm{H}}\big(\boldsymbol{x}_{i+[\ell]},\boldsymbol{y}_{i+[\ell]}\big)+d_{\mathrm{H}}\big(\boldsymbol{y}_{j+[\ell]},\boldsymbol{x}_{j+[\ell]}\big)\leqslant t$, which contradicts the theorem's assumption.

On the other hand, if $j-i\leqslant\ell-t$, then we may simply bound $d_{\mathrm{H}}\big(\boldsymbol{x}_{i+[\ell]},\boldsymbol{y}_{i+[\ell]}\big)+d_{\mathrm{H}}\big(\boldsymbol{y}_{j+[\ell]},\boldsymbol{x}_{j+[\ell]}\big)\leqslant 2t$, again in contradiction.

Finally, suppose $\ell-t<j-i<\ell$. Note that

$$\begin{aligned}&d_{\mathrm{H}}\big(\boldsymbol{x}_{i+[\ell]},\boldsymbol{y}_{i+[\ell]}\big)+d_{\mathrm{H}}\big(\boldsymbol{y}_{j+[\ell]},\boldsymbol{x}_{j+[\ell]}\big)\\ &\quad= d_{\mathrm{H}}\big(\boldsymbol{x}_{[i,j-1]},\boldsymbol{y}_{[i,j-1]}\big)+2d_{\mathrm{H}}\big(\boldsymbol{x}_{[j,i+\ell-1]},\boldsymbol{y}_{[j,i+\ell-1]}\big)+\\ &\qquad d_{\mathrm{H}}\big(\boldsymbol{y}_{[i+\ell,j+\ell-1]},\boldsymbol{x}_{[i+\ell,j+\ell-1]}\big).\end{aligned}$$

Since $[i,j-1],[j,i+\ell-1],[i+\ell,j+\ell-1]$ are pairwise disjoint,

$$\begin{aligned}&d_{\mathrm{H}}\big(\boldsymbol{x}_{[i,j-1]},\boldsymbol{y}_{[i,j-1]}\big)+d_{\mathrm{H}}\big(\boldsymbol{x}_{[j,i+\ell-1]},\boldsymbol{y}_{[j,i+\ell-1]}\big)+\\ &\qquad\qquad d_{\mathrm{H}}\big(\boldsymbol{y}_{[i+\ell,j+\ell-1]},\boldsymbol{x}_{[i+\ell,j+\ell-1]}\big)\leqslant t.\end{aligned}$$

Hence, denoting $\Delta\triangleq d_{\mathrm{H}}\big(\boldsymbol{x}_{[j,i+\ell-1]},\boldsymbol{y}_{[j,i+\ell-1]}\big)$, we have

$$d_{\mathrm{H}}\big(\boldsymbol{x}_{i+[\ell]},\boldsymbol{x}_{j+[\ell]}\big)\leqslant t+\Delta\leqslant t+(\ell-j+i),$$

once more in contradiction. This concludes the proof. ■

Observe in particular that Lemma 9 applies to $n=\ell+k$, and its proof can be applied specifically for $(i,j)=(0,k)$. That is, if $\operatorname{wt}(\boldsymbol{u}_{0,k})=d_{\mathrm{H}}(\boldsymbol{x}_{[\ell]},\boldsymbol{x}_{k+[\ell]})>t+\min\{t,\ell-k\}$ then $\boldsymbol{x}\notin\mathcal{A}_t^\ell(k)$. Vice versa, $\pi_t^\ell(k)=\Pr\big(\boldsymbol{x}\in\mathcal{A}_t^\ell(k)\big)\leqslant\Pr(\operatorname{wt}(\boldsymbol{u}_{0,k})\leqslant t+\min\{t,\ell-k\})$, which leads to the following bound:

Corollary 10 $\pi_t^\ell(k)\leqslant q^{-\ell}\sum_{i=0}^{t+\min\{t,\ell-k\}}\binom{\ell}{i}(q-1)^i$.

Proof: By Lemma 3 $\boldsymbol{u}_{0,k}\in\Sigma^\ell$ is distributed uniformly, hence from the above observation the proof is concluded. ■

The bound of Corollary 10 can be improved upon in some cases, depending on k (thus improving the upper bound on π'):

Lemma 11

$$\pi_t^\ell(k)\geqslant q^{-\ell}\sum_{i=0}^{t}\binom{\ell}{i}(q-1)^i,$$

and

$$\pi_t^\ell(k)\leqslant\begin{cases} q^{-\ell}\sum_{i=0}^{t}\binom{\ell+k}{i}(q-1)^i, & k\leqslant\frac{\ell}{2};\\ q^{-\ell}\sum_{i=0}^{t}\binom{2\ell-k}{i}(q-1)^i & k>\frac{\ell}{2}.\end{cases}$$

Proof: Take integers $p\geqslant 2$ and $0\leqslant r<k$ such that $\ell+k=pk+r$. For $\boldsymbol{x}\in\mathcal{A}_t^\ell(k)$, there exists $\boldsymbol{y}\in B_t^{\mathrm{s}}(\boldsymbol{x})$ such that $\boldsymbol{y}_{[\ell]}=\boldsymbol{y}_{k+[\ell]}$. The method of our proof utilizes the observation $\boldsymbol{y}_{[\ell]}=\boldsymbol{y}_{k+[\ell]}$ implies that $\boldsymbol{y}$ is k-periodic, i.e., can be determined by its first k coordinates:

$$\boldsymbol{y}=(\underbrace{\boldsymbol{y}_{[k]},\ldots,\boldsymbol{y}_{[k]}}_{p\text{ times}},\boldsymbol{y}_{[r]}).$$

Observe for each $\boldsymbol{y}\in\Sigma^{\ell+k}$, satisfying $\boldsymbol{y}_{[\ell]}=\boldsymbol{y}_{k+[\ell]}$ (of which we have seen there exist precisely q^k distinct possibilities, corresponding to a free choice of $\boldsymbol{y}_{[k]}$), that one may form a unique $\boldsymbol{x}\in\mathcal{A}_t^\ell(k)$ by changing at most t of the symbols $\boldsymbol{y}_{k+[\ell]}$. It follows that

$$\big|\mathcal{A}_t^\ell(k)\big|\geqslant q^k\sum_{i=0}^{t}\binom{\ell}{i}(q-1)^i.$$

On the other hand, it is also straightforward that

$$\big|\mathcal{A}_t^\ell(k)\big|\leqslant q^k\sum_{i=0}^{t}\binom{\ell+k}{i}(q-1)^i,$$

by changing up to t symbols of the whole of $\boldsymbol{y}$.

When $p=2$ or, equivalently, $k>\frac{\ell}{2}$, we shall improve the above bound. Take $\boldsymbol{x}\in\mathcal{A}_k$ and $\boldsymbol{y}\in B^{\mathrm{s}}_t(\boldsymbol{x})$ satisfying $\boldsymbol{y}_{[\ell]}=\boldsymbol{y}_{k+[\ell]}$. Define the intervals $I_1\triangleq[\ell-k]$, $I_2\triangleq[\ell-k,k-1]$, $I_3\triangleq[k,\ell-1]$, $I_4\triangleq[\ell,2k-1]$, $I_5\triangleq[2k,k+\ell-1]$. Using this notation, we have $\boldsymbol{y}_{I_2}=\boldsymbol{y}_{I_4}$ and $\boldsymbol{y}_{I_1}=\boldsymbol{y}_{I_3}=\boldsymbol{y}_{I_5}$, i.e.,

$$\boldsymbol{y}=\boldsymbol{y}_{I_1}\boldsymbol{y}_{I_2}\boldsymbol{y}_{I_1}\boldsymbol{y}_{I_2}\boldsymbol{y}_{I_1}.$$

Consider the string $\boldsymbol{y}'\triangleq\boldsymbol{y}_{I_1}\boldsymbol{x}_{I_2}\boldsymbol{y}_{I_1}\boldsymbol{x}_{I_2}\boldsymbol{y}_{I_1}$ and note that

$$\begin{aligned}d_{\mathrm{H}}(\boldsymbol{x},\boldsymbol{y}')&=d_{\mathrm{H}}\big(\boldsymbol{x}_{I_1\cup I_3\cup I_5},\boldsymbol{y}'_{I_1\cup I_3\cup I_5}\big)+\\&\quad d_{\mathrm{H}}\big(\boldsymbol{x}_{I_2},\boldsymbol{y}'_{I_2}\big)+d_{\mathrm{H}}\big(\boldsymbol{x}_{I_4},\boldsymbol{y}'_{I_2}\big)\\&=d_{\mathrm{H}}(\boldsymbol{x}_{I_1\cup I_3\cup I_5},\boldsymbol{y}_{I_1\cup I_3\cup I_5})+\\&\quad d_{\mathrm{H}}(\boldsymbol{x}_{I_2},\boldsymbol{x}_{I_2})+d_{\mathrm{H}}(\boldsymbol{x}_{I_4},\boldsymbol{x}_{I_2})\\&=d_{\mathrm{H}}(\boldsymbol{x}_{I_1\cup I_3\cup I_5},\boldsymbol{y}_{I_1\cup I_3\cup I_5})+\\&\quad 0+d_{\mathrm{H}}(\boldsymbol{x}_{I_4},\boldsymbol{x}_{I_2}).\end{aligned}$$

Applying the triangle inequality on the last addend,

$$\begin{aligned}d_{\mathrm{H}}(\boldsymbol{x},\boldsymbol{y}')&\leqslant d_{\mathrm{H}}(\boldsymbol{x}_{I_1\cup I_3\cup I_5},\boldsymbol{y}_{I_1\cup I_3\cup I_5})+\\&\quad d_{\mathrm{H}}(\boldsymbol{x}_{I_4},\boldsymbol{y}_{I_4})+d_{\mathrm{H}}(\boldsymbol{y}_{I_4},\boldsymbol{x}_{I_2})\\&=d_{\mathrm{H}}(\boldsymbol{x}_{I_1\cup I_3\cup I_5},\boldsymbol{y}_{I_1\cup I_3\cup I_5})+\\&\quad d_{\mathrm{H}}(\boldsymbol{x}_{I_4},\boldsymbol{y}_{I_4})+d_{\mathrm{H}}(\boldsymbol{y}_{I_2},\boldsymbol{x}_{I_2})\\&=d_{\mathrm{H}}(\boldsymbol{x},\boldsymbol{y})\leqslant t.\end{aligned}$$

Therefore, for any $\boldsymbol{x}\in\mathcal{A}^{\ell}_t(k)$ there exists $\boldsymbol{y}'\in B^{\mathrm{s}}_t(\boldsymbol{x})$ of the form

$$\boldsymbol{y}'=\boldsymbol{y}_{I_1}\boldsymbol{x}_{I_2}\boldsymbol{y}_{I_1}\boldsymbol{x}_{I_2}\boldsymbol{y}_{I_1},$$

and in particular $\boldsymbol{x}_{I_2}=\boldsymbol{y}'_{I_2}$. This implies an improved upper bound, by freely choosing $\boldsymbol{y}_{I_1}\boldsymbol{x}_{I_2}\in\Sigma^k$ and subsequently at most t coordinates from $[\ell+k]\setminus I_2$ to change, as follows

$$\big|\mathcal{A}^{\ell}_t(k)\big|\leqslant q^k\sum_{i=0}^{t}\binom{2\ell-k}{i}(q-1)^i. \qquad\blacksquare$$

With the results of Corollary 10 and Lemma 11, we can now bound π,π' to facilitate the application of Theorem 8.

Corollary 12

$$\pi=q^{-\ell}\sum_{i=0}^{t}\binom{\ell}{i}(q-1)^i\leqslant q^{-\ell\left(1-H_q\left(\min\left\{\frac{q-1}{q},\frac{t}{\ell}\right\}\right)\right)},$$

and

$$\begin{aligned}\pi'&\leqslant q^{-\ell}\sum_{i=0}^{t}\binom{\lfloor 3\ell/2\rfloor}{i}(q-1)^i\\&\leqslant q^{-\ell\left(1-\frac{3}{2}H_q\left(\min\left\{\frac{q-1}{q},\frac{2t}{3\ell}\right\}\right)\right)},\end{aligned}$$

where $H_q(\delta)=\delta\log_q(q-1)-\delta\log_q(\delta)-(1-\delta)\log_q(1-\delta)$ is the q-ary entropy function.

Proof: First observe the equality on the first line follows from the lower bound of Lemma 11, together with the upper bound of either Corollary 10 or Lemma 11. Similarly, the first inequality on the second line follows from the upper bound of Lemma 11.

The second inequality on both lines follows from the standard bound on the size of the q-ary Hamming ball (see, e.g., [36, Lem. 4.7]); in particular observe that

$$\sum_{i=0}^{t}\binom{\lfloor 3\ell/2\rfloor}{i}(q-1)^i\leqslant q^{\lfloor 3\ell/2\rfloor H_q\left(\min\left\{\frac{q-1}{q},\frac{t}{\lfloor 3\ell/2\rfloor}\right\}\right)},$$

and since $x\mapsto xH_q(1/x)$ is increasing for $x\geqslant 1$, the claim follows. $\blacksquare$

We note before continuing that applying the upper bound of Corollary 10 instead of Lemma 11 would result in an inferior upper bound on π'.

Motivated by the discussion at the beginning of this section, we fix the values of t,ℓ for the reminder of this paper. Take $a>1$ and a real number $\delta>0$; we let

$$\begin{aligned}\ell&\triangleq\big\lfloor a\log_q(n)\big\rfloor;\\t&\triangleq\lfloor\delta\ell\rfloor=\big\lfloor\delta\big\lfloor a\log_q(n)\big\rfloor\big\rfloor,\end{aligned}\tag{4}$$

as n grows.

Inspired by Corollary 12, we also denote by $\widetilde{\delta}_q$ the (unique) real number $0<\widetilde{\delta}_q<\frac{q-1}{q}$ satisfying

$$H_q\left(\tfrac{2}{3}\widetilde{\delta}_q\right)=\tfrac{2}{3}.$$

We observe by substitution that $\widetilde{\delta}_q>\frac{q-1}{2q}$, and provide $\widetilde{\delta}_q$ for some small values of q:

q	$\frac{q-1}{2q}$	$\widetilde{\delta}_q$	$\frac{q-1}{q}$
2	0.25	0.2609	0.5
3	0.3333	0.3723	0.6667
4	0.375	0.4375	0.75
5	0.4	0.4817	0.8
6	0.4167	0.5141	0.8333

Applying the result of Corollary 12 to Theorem 8, we can now obtain the following result.

Theorem 13 *Fix $a>1$, $0<\delta<\widetilde{\delta}_q$. Then, as $n\to\infty$,*

$$\operatorname{red}(\mathcal{RRF}^{\mathrm{s}}(n))=O(n^{2-a(1-H_q(\delta))}).$$

Proof: If $a\leqslant(1-H_q(\delta))^{-1}$ the proposition vacuously holds.

Otherwise, let $\boldsymbol{x}\in\Sigma^{\ell+k}$ be chosen uniformly at random. Based on Corollary 12 (recalling again that $x\mapsto xH_q(1/x)$ is increasing for $x\geqslant 1$), we observe for $\delta<\widetilde{\delta}_q$ that

$$\begin{aligned}\pi&\leqslant q\cdot n^{-a(1-H_q(\delta))};\\\pi'&\leqslant q\cdot n^{-a\left(1-\frac{3}{2}H_q\left(\frac{2}{3}\delta\right)\right)}.\end{aligned}$$

Hence, for sufficiently large n it holds that $3\ell^2\pi'+2\ell n\pi<1/e$, satisfying the conditions of Theorem 8. Since we also have $n\log(n)\pi'=o\big(n^{2-a(1-H_q(\delta))}\big)$, the claim follows from Theorem 8. $\blacksquare$

Corollary 14 *Take $0<\delta<\widetilde{\delta}_q$. If $a>(1-H_q(\delta))^{-1}$ then $R(\mathcal{RRF}^{\mathrm{s}}(n))=1-o(1)$, and if $a\geqslant 2(1-H_q(\delta))^{-1}$, then $\mathcal{RRF}^{\mathrm{s}}(n)$ incurs a constant number of redundant symbols.*

The last corollary can be viewed in the context of related works; as mentioned above, [21] demonstrated that if $a > 1$ then $R(\mathcal{RF}_\ell(n)) = 1 - o_n(1)$, and if $a \geqslant 2$ then $\operatorname{red}(\mathcal{RF}_\ell(n)) = O_n(1)$. Corollary 14 demonstrates that if $a > 1$ (respectively $a \geqslant 2$), then for all sufficiently small $\delta > 0$ it holds that $R(\mathcal{RRF}^{\mathrm{s}}(n)) = 1 - o_n(1)$ (respectively, $\operatorname{red}(\mathcal{RRF}^{\mathrm{s}}(n)) = O_n(1)$). That is, resilient-repeat-free sequences for a number of substitutions errors logarithmic in the string length (linear in the substring length) incur no additional asymptotic cost.

Up until here, we have focused on demonstrating conditions sufficient for the rate of resilient-repeat-free strings to be asymptotically optimal. In the sequel, we pursue the converse, or more precisely, necessary conditions for such strings to obtain non-vanishing rate.

Definition 15 *For a real δ, $0 \leqslant \delta < 1$, and an integer $\ell > 0$, let $M_q(\ell, \delta)$ be the maximum number of code-words in a code $C \subseteq \Sigma^\ell$ such that $d_{\mathrm{H}}(\boldsymbol{x}, \boldsymbol{y}) \geqslant \delta\ell$ for any distinct $\boldsymbol{x}, \boldsymbol{y} \in C$. For a given $\delta > 0$, define the maximum achievable rate by*

$$R_q(\delta) \triangleq \limsup_{\ell\to\infty} \tfrac{1}{\ell} \log_q M_q(\ell, \delta).$$

For completeness, we state the well-known Gilbert-Varshamov and Elias-Bassalygo bounds (see, e.g., [36, Thm.4.9-12]) for $\delta \leqslant \frac{q-1}{q}$,

$$1 - H_q(\delta) \leqslant R_q(\delta) \leqslant 1 - H_q\Big(\tfrac{q-1}{q}\Big(1 - \sqrt{1 - \tfrac{q}{q-1}\delta}\Big)\Big).$$

The following lemma states a converse bound to Corollary 14.

Lemma 16 *If $a < R_q(\delta)^{-1}$, then for sufficiently large $n \in \mathbb{N}$*

$$\mathcal{RRF}^{\mathrm{s}}(n) = \emptyset.$$

In particular, the statement holds if $t \geqslant \frac{q-1}{q}\ell$, for all a.

Proof: Take, on the contrary, some $\boldsymbol{x} \in \mathcal{RRF}^{\mathrm{s}}(n)$. By Definition 6, the ℓ-mers

$$\big\{\boldsymbol{x}_{i\ell+[\ell]} : 0 \leqslant i \leqslant \lfloor n/\ell \rfloor - 1\big\} \subseteq \Sigma^\ell$$

form a code of size $\lfloor n/\ell \rfloor$ and minimum distance $d > t \geqslant \delta\ell$. By Definition 15 we obtain

$$\frac{\log\lfloor n/\ell \rfloor}{\ell} \leqslant R_q(\delta) + o(1).$$

Recalling $\ell = \lfloor a \log n \rfloor$ yields that

$$\tfrac{1}{a} \leqslant R_q(\delta) + o(1),$$

in contradiction to the assumption. ■

It should be noted that Lemma 16 specifically pertains to resilient-repeat-free strings, which the reader will observe are not necessarily required for successful reconstruction of information. Nevertheless, it might be conjectured, based on the noiseless case, that resilient-repeat-free sequences may achieve optimum asymptotic rate.

Before concluding, we note that a twofold gap remains between Theorem 13 and the converse of Lemma 16. First, $\operatorname{red}(\mathcal{RRF}^{\mathrm{s}}(n))$ is not characterized when $R_q(\delta)^{-1} \leqslant a \leqslant (1 - H_q(\delta))^{-1}$; and second, it is not found when $\delta \geqslant \widetilde{\delta}_q$.

B. Encoding resilient-repeat-free codes

In this section, we present an explicit encoder of resilient-repeat-free strings, in the hope that it may then be utilized in constructing error-correcting codes for the noisy substring channel.

We first discuss how elements of the ball $B^{\mathrm{s}}_t(\mathbf{0})$ (which throughout this discussion we assume to contain length-ℓ sequences; observe that we opt to use ℓ instead of n here since our analysis will later be applied to ℓ-substring of a longer length-n string) may be enumerated. Observe that given any $\boldsymbol{x}_{[k]} \in \Sigma^k$ with $\operatorname{wt}(\boldsymbol{x}_{[k]}) \leqslant t$,

$$\begin{aligned} n(\boldsymbol{x}_{[k]}) &\triangleq \big|\big\{\boldsymbol{y} \in B^{\mathrm{s}}_t(\mathbf{0}) : \boldsymbol{y}_{[k]} = \boldsymbol{x}_{[k]}\big\}\big| \\ &= \sum_{j=0}^{t-\operatorname{wt}(\boldsymbol{x}_{[k]})} \binom{\ell-k}{j}(q-1)^j. \end{aligned}$$

Example 17 *We take $q = 2, \ell = 7, t = 3, k = 4$ and $\boldsymbol{x} = 0110010 \in \Sigma^7 \cap B^{\mathrm{s}}_3(\mathbf{0})$. Then, the number of elements $\boldsymbol{y} \in B^{\mathrm{s}}_3(\mathbf{0})$ such that $\boldsymbol{y}_{[4]} = \boldsymbol{x}_{[4]} = 0110$ equals*

$$\begin{aligned} n(\boldsymbol{x}_{[4]}) &= \sum_{j=0}^{3-\operatorname{wt}(0110)} \binom{7-4}{j}(2-1)^j \\ &= \sum_{j=0}^{1} \binom{3}{j} = 1 + 3 = 4. \end{aligned}$$

These elements are

$$\{0110000, 0110001, 0110010, 0110100\}.$$

Assuming a total order $<$ on Σ, denote $\|x\| \triangleq \big|\{y \in \Sigma : y < x\}\big|$ for all $x \in \Sigma$. It was shown in [37] that the lexicographic index of $\boldsymbol{x} \in B^{\mathrm{s}}_t(\mathbf{0})$ equals

$$i(\boldsymbol{x}) = \sum_{k\in[\ell]} \sum_{\alpha<x(k)} n(\boldsymbol{x}_{[k-1]}\alpha), \tag{5}$$

where we let $\boldsymbol{x}_{[0]}$ be the empty string, with $\operatorname{wt}(\boldsymbol{x}_{[0]}) \triangleq 0$.

Example 18 *We use the natural order $0 < 1$ with $q = 2$. Then, using $\boldsymbol{x} = 0110010$ from Example 17 we ascertain its lexicographic index in $B^{\mathrm{s}}_3(\mathbf{0})$:*

$$\begin{aligned} i(\boldsymbol{x}) &= \sum_{k=0}^{6} \sum_{\alpha<x(k)} n(\boldsymbol{x}_{[k]}\alpha) \\ &= n(\boldsymbol{x}_{[1]}0) + n(\boldsymbol{x}_{[2]}0) + n(\boldsymbol{x}_{[5]}0) \\ &= n(00) + n(010) + n(011000) \\ &= \sum_{j=0}^{3-\operatorname{wt}(00)} \binom{7-2}{j} + \sum_{j=0}^{3-\operatorname{wt}(010)} \binom{7-3}{j} + \\ &\quad \sum_{j=0}^{3-\operatorname{wt}(011000)} \binom{7-6}{j} \\ &= \sum_{j=0}^{3} \binom{5}{j} + \sum_{j=0}^{2} \binom{4}{j} + \sum_{j=0}^{1} \binom{1}{j} \\ &= (1 + 5 + 10 + 10) + (1 + 4 + 6) + (1 + 1) = 39. \end{aligned}$$

Lemma 19 *The lexicographic index of $\boldsymbol{x} \in B^{\mathrm{s}}_t(\mathbf{0})$ equals*

$$\begin{aligned} i(\boldsymbol{x}) = \sum_{k\in[\ell]} &\|x(k)\| \cdot n(\boldsymbol{x}_{[k+1]}) + \\ &\mathrm{wt}(x(k)) \cdot \binom{\ell-k-1}{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} (q-1)^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})}. \end{aligned}$$

Proof: Observe that if $x(k) \neq 0$, then

$$\begin{aligned} n(\boldsymbol{x}_{[k]}0) &= \sum_{j=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]}0)} \binom{\ell-(k+1)}{j} (q-1)^j \\ &= \sum_{j=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} \binom{\ell-k-1}{j} (q-1)^j, \end{aligned}$$

hence

$$\begin{aligned} n(\boldsymbol{x}_{[k+1]}) &= \sum_{j=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k+1]})} \binom{\ell-(k+1)}{j} (q-1)^j \\ &= \sum_{j=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})-1} \binom{\ell-k-1}{j} (q-1)^j \\ &= n(\boldsymbol{x}_{[k]}0) - \binom{\ell-k-1}{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} (q-1)^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})}. \end{aligned}$$

It is also immediate that for any $\alpha \in \Sigma \setminus \{0\}$ it holds that $n(\boldsymbol{x}_{[k]}\alpha) = n(\boldsymbol{x}_{[k+1]})$. The claim now follows from (5). ■

Example 20 *Repeating Example 18 with Lemma 19 we can find*

$$\begin{aligned} i(\boldsymbol{x}) = \sum_{k\in[7]} &\|x(k)\| \cdot n(\boldsymbol{x}_{[k+1]}) + \\ &\mathrm{wt}(x(k)) \cdot \binom{6-k}{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} (q-1)^{3-\mathrm{wt}(\boldsymbol{x}_{[k]})} \\ = n(\boldsymbol{x}_{[2]}) &+ \binom{6-1}{3-\mathrm{wt}(\boldsymbol{x}_{[1]})} + \\ n(\boldsymbol{x}_{[3]}) &+ \binom{6-2}{3-\mathrm{wt}(\boldsymbol{x}_{[2]})} + \\ n(\boldsymbol{x}_{[6]}) &+ \binom{6-5}{3-\mathrm{wt}(\boldsymbol{x}_{[5]})} \\ = n(01) &+ \binom{5}{3} + \\ n(011) &+ \binom{4}{2} + \\ n(011001) &+ \binom{1}{1} \\ = \sum_{j=0}^{2} \binom{5}{j} &+ \binom{5}{3} + \\ \sum_{j=0}^{1} \binom{4}{j} &+ \binom{4}{2} + \\ \binom{1}{0} &+ \binom{1}{1} = 39, \end{aligned}$$

matching the result of Example 18.

Computationally, the most taxing expression to calculate in the sum of Lemma 19 is $n(\boldsymbol{x}_{[k+1]})$; however, one might employ a recursive approach to obtaining the sum. Indeed, from the Pascal identity we observe

$$\begin{aligned} &n(\boldsymbol{x}_{[k]}) \\ &= \sum_{j=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} \binom{\ell-k}{j} (q-1)^j \\ &= \sum_{j=1}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} \binom{\ell-k-1}{j-1} (q-1)^j + \\ &\quad \sum_{j=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} \binom{\ell-k-1}{j} (q-1)^j \\ &= (q-1) \sum_{j'=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})-1} \binom{\ell-k-1}{j'} (q-1)^{j'} + \\ &\quad \sum_{j=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} \binom{\ell-k-1}{j} (q-1)^j \\ &= q \sum_{j=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})-1} \binom{\ell-k-1}{j} (q-1)^j + \\ &\quad \binom{\ell-k-1}{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} (q-1)^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})}. \end{aligned}$$

By incrementing the upper limit of the sum through $t - \mathrm{wt}(\boldsymbol{x}_{[k]})$ and subtracting the corresponding addend separately, the last line can also be restated

$$\begin{aligned} n(\boldsymbol{x}_{[k]}) = q &\sum_{j=0}^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} \binom{\ell-k-1}{j} (q-1)^j - \\ &(q-1) \binom{\ell-k-1}{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} (q-1)^{t-\mathrm{wt}(\boldsymbol{x}_{[k]})}. \end{aligned}$$

Partitioning into cases by $\mathrm{wt}(x(k))$, we find

$$\begin{aligned} &n(\boldsymbol{x}_{[k]}) \\ &= q \cdot n(\boldsymbol{x}_{[k+1]}) - \\ &\quad (-1)^{\mathrm{wt}(x(k))} \binom{\ell-k-1}{t-\mathrm{wt}(\boldsymbol{x}_{[k]})} (q-1)^{t-\mathrm{wt}(\boldsymbol{x}_{[k+1]})+1}, \end{aligned} \tag{6}$$

where trivially $n(\boldsymbol{x}_{[\ell]}) = n(\boldsymbol{x}) = 1$.

Example 21 *In Examples 17 and 20 we have found, in $B^{\mathrm{s}}_3(\mathbf{0})$,*

$$\begin{aligned} n(01) &= \sum_{j=0}^{2} \binom{5}{j} = 16; \\ n(011) &= \sum_{j=0}^{1} \binom{4}{j} = 5; \\ n(0110) &= \sum_{j=0}^{1} \binom{3}{j} = 4. \end{aligned}$$

We can now confirm that indeed

$$\begin{aligned} n(011) &= 2 \cdot n(0110) - (-1)^{\mathrm{wt}(0)} \binom{6-3}{3-2} \\ &= 2 \cdot 4 - 3, \end{aligned}$$

and

$$\begin{aligned} n(01) &= 2 \cdot n(011) - (-1)^{\mathrm{wt}(1)} \binom{6-2}{3-1} \\ &= 2 \cdot 5 + 6. \end{aligned}$$

It now follows from Lemma 19 and (6) that computing the sum for the index $i(\boldsymbol{x})$ can be done for $k \in [\ell]$ in descending order, where at each addend it is required to:

1) Compute binomial coefficients of the form $\binom{\ell-k}{j}$ for $j \leqslant t$ (all of order at most $\binom{\ell}{t} \leqslant \frac{\ell^t}{t!} \leqslant (\frac{e\ell}{t})^t$). Observe each binomial coefficient requires $\log(\frac{e\ell}{t})^t < t\log(\ell)$ symbols, and at most $t+1$ need to be stored at a time, so that $\left\{\binom{\ell-k-1}{j} : j \leqslant t\right\}$ could be computed via the Pascal identity from $\left\{\binom{\ell-k}{j} : j \leqslant t\right\}$. This stage hence requires $O(t^2\log(\ell))$ operations, and $O(t^2\log(\ell))$ space. Further, obtaining $\left\{\binom{\ell}{j} : j \leqslant t\right\}$ for initialization requires at most $O(t^2\log(\ell)\ell)$ operations, if it is performed similarly.
2) Multiplying a binomial coefficient by at most q^t, which may practically be performed in $O(t\log(\ell)\log\log(\ell)\log\log\log(\ell))$ operations.
3) Computing $n(\boldsymbol{x}_{[k]})$, which has seen above requires $O(t\log(\ell))$ space and $O(t\log(\ell)\log\log(\ell)\log\log\log(\ell))$ operations.
4) Summing the results requires $O(t\log(\ell))$ operations and $O(t\log(\ell))$ space.

The entire algorithm therefore requires at most $O(t^2\log(\ell)\ell)$ operations and $O(t^2\log(\ell))$ space. That is, if $t,\ell = O(\log(n))$, at most $O((\log(n))^3\log\log(n))$ operations and $O((\log(n))^2\log\log(n))$ space.

The inverse operation, obtaining $\boldsymbol{x} \in B_t^{\mathrm{s}}(\mathbf{0})$ such that $i(\boldsymbol{x}) = i$, for some given i, is also due to [37]: starting with the empty sequence for $k = 0$, assume $\boldsymbol{x}_{[k-1]}$ has already been constructed for some $0 < k \leqslant \ell$. Going over $\alpha \in \Sigma$ in increasing order (assuming the same total order as before), if $i \leqslant n(\boldsymbol{x}_{[k-1]}\alpha)$ then set $x(k-1) \triangleq \alpha$ and update $i \leftarrow i - n(\boldsymbol{x}_{[k-1]}\alpha)$; otherwise increase α and repeat; the maximum element of Σ can be filled in without comparison, if the algorithm arrives at it. Again, the limiting step of the algorithm is obtaining the representation of the binomial coefficients, and while the algorithm might require $t\ell$ steps in the worst case, these do not need to be recalculated unless k is increased. Thus, calculating the inverse also requires at most $O((\log(n))^3\log\log(n))$ operations and $O((\log(n))^2\log\log(n))$ space, for $t,\ell = O(\log(n))$.

In summary, we have obtained an explicit and invertible enumerator of $B_t^{\mathrm{s}}(\mathbf{0})$, with the aforementioned complexity, which we denote $\mathrm{en}(\boldsymbol{x})$. Recall that $|B_t^{\mathrm{s}}(\mathbf{0})| \leqslant q^{\ell H_q(t/\ell)}$, i.e., $\mathrm{en}\colon B_t^{\mathrm{s}}(\mathbf{0}) \to \Sigma^{\lceil \ell H_q(t/\ell)\rceil}$.

Equipped with an (efficient) enumeration algorithm for $B_t^{\mathrm{s}}(\mathbf{0})$, we may now propose an explicit encoder of resilient-repeat-free sequences. Our encoder has the drawback that it produces sequences in $\bigcup_{m\leqslant n}\mathcal{RRF}_{t,\ell}^{\mathrm{s}}(m)$ (here, t,ℓ are specified to stress that they are invariant in m, depending only on n) rather than solely in $\mathcal{RRF}^{\mathrm{s}}(n)$; however, in practice this does not seem too onerous for applications, were shorter sequences may be stored just as easily, as long as data is recoverable.

Algorithm 1: Resilient-repeat-free Encoder

```
Input: $\boldsymbol{x} \in \Sigma^n$ containing no 0-run of length $z$
Output: $\mathrm{Enc}_1(\boldsymbol{x}) \in \bigcup_{m\leqslant n}\mathcal{RRF}^{\mathrm{s}}(m)$
$j \leftarrow 1$
while $j \leqslant |\boldsymbol{x}| - \ell'$ do
    for $i = j-1,\dots,0$ do
        if $\exists \boldsymbol{y} \in B_t^{\mathrm{s}}(\boldsymbol{x})\colon \boldsymbol{y}_{i+[\ell']} = \boldsymbol{y}_{j+[\ell']}$ then
            Replace $\boldsymbol{x}_{j+[\ell']}$ with $\boldsymbol{s}$ from (8)
            $j \leftarrow \max\{0, j-\ell'+1\}$
            break
        end
    end
    $j \leftarrow j+1$
end
return $\boldsymbol{x}$
```

Our construction is summarized in Algorithm 1; its main idea is a generalization of [21, Alg. 3], as follows. Assume $a(1-H_q(\delta)) > 1$, and choose $\epsilon > 0$ such that $\zeta \triangleq a(1-H_q(\delta)-\epsilon)-1 > 0$. Let $z = \lfloor \zeta\log_q(n)\rfloor$. An information string is first encoded into a length-n string $\boldsymbol{x}$ containing no 0-run of length z, which may be done in linear time using $\left\lceil \frac{q}{q-2}n^{1-\zeta}\right\rceil = O(n^{2-a(1-H_q(\delta)-\epsilon)})$ redundant symbols [38, Lem. 4]. Interestingly, this allows us to achieve redundancy which is arbitrarily close, in orders of magnitude, to the result of Theorem 13. Next, using

$$\ell' \triangleq 11 + \left\lceil 2\log_q(\ell) + a(1-\epsilon)\log_q(n)\right\rceil \leqslant \ell, \tag{7}$$

where the last inequality holds for all sufficiently large n, it is then iteratively checked whether $\boldsymbol{x}_{[\ell'+j]} \in \mathcal{RRF}_{t,\ell'}^{\mathrm{s}}(\ell'+j)$, for $j \in [n-\ell'+1]$ in increasing order.

If in some iteration it is determined that $\boldsymbol{x}_{[\ell'+j]} \notin \mathcal{RRF}_{t,\ell'}^{\mathrm{s}}(\ell'+j)$, then the algorithm deletes $\boldsymbol{x}_{j+[\ell']}$ from $\boldsymbol{x}$ and replaces it with a sequence with the following form:

$$\boldsymbol{s} \triangleq 0^z 1 \circ E(j-i) \circ 10^{z'}1 \circ E(\mathrm{en}(\boldsymbol{e})) \circ 1, \tag{8}$$

where

- j,i are the loop-indices at any specific iteration of Algorithm 1, and by abuse of notation we take $(j-i)$ to represent the q-ary expansion of the difference, using only as many symbols as required (since $j > i$, the all-zero representation 0^k would be taken to stand for 2^k instead of 0);
- $\boldsymbol{e} \triangleq \boldsymbol{x}_{j+[\ell']} - \boldsymbol{x}_{i+[\ell']} \in \Sigma^{\ell'}$ when $j-i \geqslant \ell'$, or $\boldsymbol{e} \triangleq \boldsymbol{x}_{i+[\ell'+k]} - \boldsymbol{y}_{i+[\ell'+k]} \in \Sigma^{\ell'+k}$ when $j-i < \ell'$; recall, however, that from the proof of Lemma 11 it follows that we may always assume $|\mathrm{supp}(\boldsymbol{e})| \leqslant \lfloor 3\ell'/2\rfloor$ is known, even for $k > \lfloor \ell'/2\rfloor$. In both cases $\mathrm{wt}(\boldsymbol{e}) \leqslant t$, and $\boldsymbol{x}_{j+[\ell']}$ is recoverable from $\boldsymbol{e}, \boldsymbol{x}_{i+[j-i]}$; and

- $z' \triangleq \lceil \log_q(\ell) \rceil + 2$ and $E(\cdot)$ is the explicit and efficient encoder described in [39, Alg. 1]; it can accept sequences of lengths at most $q\ell$, and returns an encoded version containing no 0-runs of length z', utilizing only a single redundant symbol. Observe that both $\log_q(j - i), |\mathrm{en}(\boldsymbol{e})| \leqslant \frac{3}{2}\ell < q\ell$.

We note that since $E(\cdot)$ may accept sequences of varying (sufficiently small) lengths, so too is $|\boldsymbol{s}|$ not constant. Our next aim is to bound $|\boldsymbol{s}|$ from above, as this affects the correct operation of Algorithm 1, namely, its termination condition. This behavior, and the operation of Algorithm 1, are demonstrated in the next example.

Example 22 *A complete run of Algorithm 1 will be tedious to track. We therefore limit ourselves to demonstrate a single step of the algorithm, for the follwing toy example:* $q = 2, n = 16384, a = 4, \delta = 0.025$. *Then,* $\ell = 56, t = 1$. *If we take* $\epsilon = 0.41$*, then we have* $z = 9$. *Also note* $z' = 8$ *and* $\ell' = 56 = \ell$. *Observe a sequence beginning with*

$$\begin{aligned}\boldsymbol{x} = {} & 01010111010101110101011101010111 \\ & 01010111010101110101011101011111\ldots\end{aligned}$$

which contains no run of zeros of length $z = 9$. *Assume Algorithm 1 reaches* $j = 8$ *to consider this prefix of* $\boldsymbol{x}$*; also observe that*

$$\begin{aligned}\boldsymbol{y} = {} & 01010111010101110101011101010111 \\ & 0101011101010111010101110101\underline{0}111\ldots\end{aligned}$$

satisfies $\boldsymbol{y}_{0+[\ell']} = \boldsymbol{y}_{0+[56]} = \boldsymbol{y}_{8+[56]} = \boldsymbol{y}_{j+[\ell']}$. *Therefore, Algorithm 1 replaces* $\boldsymbol{y}_{j+[\ell']}$ *with a substring* $\boldsymbol{s}$ *composed of*

$$\boldsymbol{s} \triangleq 0^9 1 \circ E(8) \circ 10^8 1 \circ E(\mathrm{en}(\boldsymbol{e})) \circ 1,$$

where

- 8 *is represented by* 000, *and* $E(000) = 0001$ *contains no zero-run of length* $z' = 8$ *and uses a single redundant symbol (we refrain from tracing [39, Alg. 1] for brevity; suffice to note that* $|E(000)| = 3 + 1 = 4$, *that it contains no zero-run of length* $z' = 8$, *and that* 000 *can be decoded from it).*
- $\boldsymbol{e} = \boldsymbol{x}_{0+[56+8]} - \boldsymbol{y}_{0+[56+8]} = 0^{60}10^3 \in \Sigma^{56+8}$, *and its indexing in* $B_1^{\mathrm{s}}(\mathbf{0})$ *is* $i(\boldsymbol{e}) = 4$. *That is,* $\mathrm{en}(\boldsymbol{e}) \in \Sigma^{\lceil \frac{3}{2}\ell' H_q(2t/3\ell') \rceil} = \Sigma^8$ *is represented* 00000100. *Finally,* $E(\mathrm{en}(\boldsymbol{e})) = 000001001$ *(again, we do not trace [39, Alg. 1]).*

Finally,

$$\boldsymbol{s} = 0^9 1 \circ 0001 \circ 10^8 1 \circ 000001001 \circ 1,$$

and its length is $34 < 56 = \ell'$ *(this fact is key to the algorithm's termination condition, as will be discussed next).*

As a matter of convenience, we denote moving forward

$$\bar{H}_q(\delta) = H_q\big(\min\{\tfrac{q-1}{q}, \delta\}\big). \tag{9}$$

Theorem 23 *If* $\frac{3}{2}\bar{H}_q(\frac{2}{3}\delta) - H_q(\delta) \leqslant \frac{1}{a}$, *then Algorithm 1 terminates,* $\mathrm{Enc}_1(\boldsymbol{x}) \in \bigcup_{m \leqslant n} \mathcal{RRF}^{\mathrm{s}}_{t,\ell}(m)$, *and* $\boldsymbol{x}$ *can be decoded from it.*

Proof: First observe that if the last iteration of Algorithm 1 terminates, then its output is resilient-repeat-free.

Next, we will show that the inserted substring in (8) is strictly less than ℓ, hence each replacement that the algorithm performs *shortens* $\boldsymbol{x}$. As a consequence, the algorithm must terminate. Indeed, observe that

$$\begin{aligned}|\boldsymbol{s}| &= 8 + \lfloor \zeta \log_q(n) \rfloor + \lceil \log_q(j - i) \rceil + \lceil \log_q(\ell) \rceil + |\mathrm{en}(\boldsymbol{e})| \\ &< 10 + \log_q(\ell) + \zeta \log_q(n) + \log_q(j - i) + |\mathrm{en}(\boldsymbol{e})|.\end{aligned}$$

If $j - i \geqslant \ell'$, then we bound $\log(j - i) \leqslant \log(n)$, and we have seen that

$$\begin{aligned}|\mathrm{en}(\boldsymbol{e})| &\leqslant \lceil \ell' \bar{H}_q(t/\ell') \rceil \leqslant \lceil \ell \bar{H}_q(t/\ell) \rceil \\ &\leqslant \lceil \ell H_q(\delta) \rceil < a H_q(\delta) \log_q(n) + 1,\end{aligned}$$

hence in this case

$$\begin{aligned}|\boldsymbol{s}| &< 11 + \log_q(\ell) + (\zeta + 1 + a H_q(\delta)) \log_q(n) \\ &= 11 + \log_q(\ell) + a(1 - \epsilon) \log_q(n) < \ell',\end{aligned}$$

as required. Otherwise we bound $\log_q(j - i) \leqslant \log_q(\ell') \leqslant \log_q(\ell)$ and

$$\begin{aligned}|\mathrm{en}(\boldsymbol{e})| &\leqslant \lceil (\ell' + k) \bar{H}_q(t/(\ell' + k)) \rceil \\ &< \tfrac{3}{2}\ell \bar{H}_q(\tfrac{2}{3}\delta) + 1,\end{aligned}$$

where $k \triangleq \min\{j - i, \lfloor \ell'/2 \rfloor\}$. Hence,

$$\begin{aligned}|\boldsymbol{s}| &< 11 + 2\log_q(\ell) + \big(\zeta + \tfrac{3}{2} a \bar{H}_q(\tfrac{2}{3}\delta)\big) \log_q(n) \\ &\leqslant \ell' + \big(a(\tfrac{3}{2}\bar{H}_q(\tfrac{2}{3}\delta) - H_q(\delta)) - 1\big) \log_q(n).\end{aligned}$$

Under the assumption of the theorem, it also holds in this case that $|\boldsymbol{s}| < \ell'$.

Lastly, observe that, iterating over $j \in [|\mathrm{Enc}_1(\boldsymbol{x})| - \ell']$ in decreasing order, the first observed instance of $0^z 1$ is always the last to have been inserted by Algorithm 1; this holds because after each replacement, j is decreased only by $\ell' - 1$, hence any later replacements, say at index j', either satisfy $j' > j$ or they overwrite the first 0 of $0^z 1$ (observe that $\boldsymbol{s}$ ends with a 1). Further, by observing the first instance of $0^{z'}$ following that instance of $0^z 1$, it is possible to uniquely deduce the coordinates of $E(j - i)$, and therefore to deduce i. Now, given $E(\mathrm{en}(\boldsymbol{e}))$ one obtains $\boldsymbol{e}$, and with $\boldsymbol{e}, \boldsymbol{x}_{i+[j-i]}$ is is uniquely possible to reconstruct the removed segment $\boldsymbol{x}_{j+[\ell]}$. Since every replacement of Algorithm 1 is reversible, and the process can be tracked in reverse, $\boldsymbol{x}$ can be reconstructed. ■

Lemma 24 *The run-time of Algorithm 1 is* $O(n^2 \log(n)^2)$.

Proof: In any iteration, if there exists $\boldsymbol{y} \in B_t^{\mathrm{s}}(\boldsymbol{x}_{[\ell'+j]}) \setminus \mathcal{RF}_{\ell'}(\ell' + j)$, and j is minimal such that this occurs, then there necessarily exists $i < j$ such that $\boldsymbol{y}_{i+[\ell']} = \boldsymbol{y}_{j+[\ell']}$. By Lemma 9, if $i \leqslant j - \ell'$, the existence of such $\boldsymbol{y}$ is equivalent to $\mathrm{wt}(\boldsymbol{x}_{j+[\ell']} - \boldsymbol{x}_{i+[\ell']}) \leqslant t$, which may be verified in at most $\ell'(j - \ell')$ operations. On the other hand, for each $0 < k < \ell'$ we check whether there exists $\boldsymbol{y} \in B_t^{\mathrm{s}}(\boldsymbol{x}_{[\ell'+j]})$ with $i = j - k$; as seen in the proof of Lemma 11, this implies that $\boldsymbol{y}_{i+[\ell'+k]}$ is k-periodic. The following procedure verifies whether such $\boldsymbol{y}$ exists: denote for convenience $\boldsymbol{u} \triangleq \boldsymbol{x}_{i+[\ell'+k]}$; for each $p \in [k]$, we define the multiset

$U_p \triangleq \{\!\{u(q) : q \equiv p \pmod{k}\}\!\}$. If the most frequent element in U_p is some $x \in \Sigma$, denote by t_p the number of occurrences of all other elements in U_p; clearly, there exists $\boldsymbol{y}$ with the given i if and only if $\sum_{p\in[k]} t_p \leqslant t$. This algorithm requires at most $O(\ell\log(\ell))$ operations for each k (due to the summation). For $\ell = O(\log(n))$, any iteration requires at most $O(n\log(n))$ operations in total, for both cases.

Finally, observe that there could be at most $n\ell' \leqslant n\ell$ iterations of Algorithm 1, completing the proof. ■

C. Error-correcting codes for the noisy substring channel

Based on Corollary 14, we can demonstrate the existence of error-correcting codes for the noisy substring channel, which achieve at most a constant redundancy over that of classical error-correcting codes for Hamming noise.

Corollary 25 *Let $C \subseteq \Sigma^n$ be an error-correcting code, capable of correcting t substitution errors, and denote, for some $\boldsymbol{z} \in \Sigma^n$, $\bar{C}_{\boldsymbol{z}} \triangleq (\boldsymbol{z}+C)\cap\mathcal{RRF}^{\mathrm{s}}(n)$. Then for any $\boldsymbol{x} \in \bar{C}_{\boldsymbol{z}}$ and $\boldsymbol{y} \in B^{\mathrm{s}}_t(\boldsymbol{x})$, it is possible to uniquely decode $\boldsymbol{x}$ observing only $Z_{\ell+1}(\boldsymbol{y})$. Further, decoding is possible through a greedy algorithm for reconstruction of $\boldsymbol{y}$, followed by application of any decoding scheme for C.*

Finally, in the cases indicated in Corollary 14, where $\mathrm{red}(\mathcal{RRF}^{\mathrm{s}}(n)) = O(1)$, there exists $\boldsymbol{z}$ satisfying $\mathrm{red}(\bar{C}_{\boldsymbol{z}}) = \mathrm{red}(C) + O(1)$.

Note that Corollary 25 is unfortunately nonconstructive. It is our hope that the encoder of Algorithm 1 may be combined with error-correction techniques to yield explicit code constructions for this channel. However, achieving this goal seems to require new ideas, and we leave it for future study.

V. Deletion noise

This section is dedicated to the study of resilient-repeat-free sequences under deletion, rather than Hamming, errors. We demonstrate that the same probabilistic tools can be used to bound from above the redundancy of such sequences. We remark that the same method can be used to study insertion errors, even though the equivalence of insertion/deletion-correction does not extend in a straightforward manner to our setting.

For $\boldsymbol{x} \in \Sigma^n$, let $S^{\mathrm{d}}_t(\boldsymbol{x}) \subseteq \Sigma^{n-t}$ denote the set of strings generated from $\boldsymbol{x}$ by t deletions. Again, superscript d marks *deletion* noise, and does not serve as a parameter.

Definition 26 *For integers $t, \ell \leqslant n$, define a family of repeat-free strings which is resistant to deletion noise:*

$$\mathcal{RRF}^{\mathrm{d}}_{t,\ell}(n) \triangleq \big\{\boldsymbol{x} \in \Sigma^n : S^{\mathrm{d}}_t(\boldsymbol{x}) \subseteq \mathcal{RF}_\ell(n-t)\big\}.$$

Again, we fix t, ℓ as in (4), and omit them from $\mathcal{RRF}^{\mathrm{d}}(n)$ whenever possible. Then we have the following:

Theorem 27 *For all $a > 1$ and $\delta > 0$ it holds that*

$$\mathrm{red}\Big(\mathcal{RRF}^{\mathrm{d}}(n)\Big) = O\Big(n^{2-a+\frac{2a(1+\delta)}{\log_2(q)}H_2(\delta/(1+\delta))}\Big/\log(n)\Big).$$

Proof: We follow a similar strategy as in Theorem 8, but apply a symmetric bound in Corollary 5, i.e., utilizing a fixed constant $\phi_{I,J} \equiv \phi$. Note that a sufficient condition for $\boldsymbol{x} \in \mathcal{RRF}^{\mathrm{d}}(n)$ is that for every observable pair $(I,J) \in \binom{[n]}{\ell_a}^2$, such that

$$I(\ell-1) - I(0) < \ell + t \tag{10}$$

(and similarly for J), it holds that $\boldsymbol{x}_I \neq \boldsymbol{x}_J$. For such a pair, denote

$$A_{I,J} \triangleq \{\boldsymbol{x} \in \Sigma^n : \boldsymbol{x}_I = \boldsymbol{x}_J\} = \{\boldsymbol{x} \in \Sigma^n : \boldsymbol{u}_{I,J} = 0\}.$$

Again, we let $\boldsymbol{x} \in \Sigma^n$ be chosen uniformly at random, implying $\mathrm{red}\big(\mathcal{RRF}^{\mathrm{d}}(n)\big) = -\log_q \Pr\big(\boldsymbol{x} \in \mathcal{RRF}^{\mathrm{d}}(n)\big)$.

In order to apply Corollary 5 we need to determine the constant ϕ, the neighborhoods $\Gamma_{I,J}$ (establishing an independence condition) and their sizes. For any observable pair (I,J), note that $\Pr(\boldsymbol{x} \in A_{I,J}) = q^{-\ell} \leqslant q \cdot n^{-a}$, and for convenience denote $\pi_{\mathrm{d}} \triangleq q \cdot n^{-a}$ and $\phi \triangleq e\pi_{\mathrm{d}}$. Next, by Lemma 3 it suffices that $\Gamma_{I,J}$ consists of all $(P,Q) \in \Gamma_I$ satisfying (10). Thus, to determine P, it suffices to choose

1) a single element of I (which shall be a member of $P\cap I$);
2) an interval of length $\ell+t$ containing the chosen element; and
3) any $\ell-1 < \ell$ additional elements of the chosen interval.

Then Q can be chosen from any interval of length $\ell+t$. The same holds for a suitable choice of $Q \cap I \neq \emptyset$. Thus, $|\Gamma_{I,J}| \leqslant \ell(\ell+t)n\binom{\ell+t}{\ell}^2$.

Now, in order to satisfy the conditions of Corollary 5 we observe from $\binom{s}{r} \leqslant \sqrt{\frac{s}{r(s-r)}}2^{sH_2(r/s)}$ (a relaxation of, e.g., [40, Ch.10, Sec.11, Lem.7]) that

$$\binom{\ell+t}{t}^2 \leqslant \frac{\ell+t}{\ell t} n^{\frac{2a(1+\delta)}{\log_2(q)}H_2(\delta/(1+\delta))}.$$

If $\frac{2(1+\delta)}{\log_2(q)}H_2\big(\frac{\delta}{1+\delta}\big) \geqslant 1$ or $a < \big(1 - \frac{2(1+\delta)}{\log_2(q)}H_2\big(\frac{\delta}{1+\delta}\big)\big)^{-1}$, the theorem vacuously holds. Otherwise, we note that

$$\begin{aligned}(|\Gamma_{I,J}|+1)\pi_{\mathrm{d}} &\leqslant qn^{-a} + q\frac{(\ell+t)^2}{t}n^{1-a+\frac{2a(1+\delta)}{\log_2(q)}H_2\left(\frac{\delta}{1+\delta}\right)} \\ &= o_n(1).\end{aligned}$$

Then, we observe

$$\begin{aligned}\phi_{I,J}\exp\Big(-\sum_{(P,Q)\in\Gamma_{I,J}}\phi_{P,Q} - \phi_{I,J}\Big) &= \pi_{\mathrm{d}}e^{1-e(|\Gamma_{I,J}|+1)\pi_{\mathrm{d}}} \\ &> \pi_{\mathrm{d}} \geqslant \Pr(\boldsymbol{x} \in A_{I,J}).\end{aligned}$$

Finally, one needs also note that the number of observable pairs (I,J) satisfying Eq. (10) is no more than $\binom{n-\ell-t}{2} \cdot \binom{\ell+t}{\ell}^2 < n^2\binom{\ell+t}{\ell}^2$. From Corollary 5 it follows that

$$\Pr\Big(\boldsymbol{x} \notin \bigcup_{I,J} A_{I,J}\Big) \geqslant \exp\bigg(-e\pi_{\mathrm{d}}n^2\binom{\ell+t}{\ell}^2\bigg),$$

and hence

$$\begin{aligned}\mathrm{red}\Big(\mathcal{RRF}^{\mathrm{d}}(n)\Big) &= -\log_q \Pr\Big(x \in \mathcal{RRF}^{\mathrm{d}}(n)\Big) \\ &\leqslant e\log(e)\pi_{\mathrm{d}}n^2\binom{\ell+t}{\ell}^2 \\ &\leqslant e\log(e)q\frac{\ell+t}{\ell t}n^{2-a+\frac{2a(1+\delta)}{\log_2(q)}H_2(\delta/(1+\delta))},\end{aligned}$$

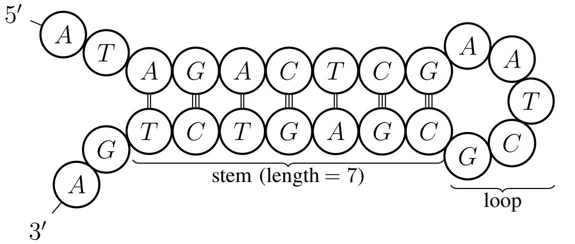

Figure 1. Formation of a hairpin-loop secondary structure in an oligonucleotide.

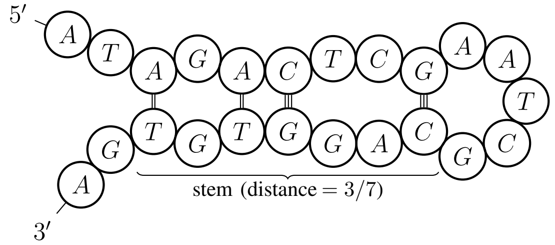

Figure 2. Imperfect stem in a hairpin-loop structure.

which completes the proof. ■

Corollary 28 *If* $a > \big(1 - \frac{2(1+\delta)}{\log_2(q)} H_2\big(\frac{\delta}{1+\delta}\big)\big)^{-1}$, *for any* $\delta > 0$, *then* $R(\mathcal{RRF}^{\mathrm{d}}(n)) = 1 - o_n(1)$, *and if* $a \geqslant 2\big(1 - \frac{2(1+\delta)}{\log_2(q)} H_2\big(\frac{\delta}{1+\delta}\big)\big)^{-1}$ *then* $\mathrm{red}\Big(\mathcal{RRF}^{\mathrm{d}}(n)\Big) = O_n(1)$.

Note again that if $a > 1$ (respectively $a > 2$), then for all sufficiently small $\delta > 0$ it holds that $R(\mathcal{RRF}^{\mathrm{d}}(n)) = 1 - o_n(1)$ (respectively, $\mathrm{red}\big(\mathcal{RRF}^{\mathrm{d}}(n)\big) = O_n(1)$). Before concluding, we also note that a parallel statement to Corollary 25 holds in this setting, as well.

VI. Secondary structure avoidance

In this section, we leverage Algorithm 1 to protect against the formation of secondary structures in coded DNA strands. We focus on a special type of secondary structure, called *hairpin-loop* (see Figure 1). Unlike recent works, our analysis does not require a perfect binding in the stem region of the hairpin structures (see Figure 2), and we show that Algorithm 1 can be utilized to avoid the formation of such structures. However, we rely in this section on the Hamming metric rather than the Levenshtein metric as was suggested in [30], thus we do not consider the formation of so-called *bulge-loops* due to the elasticity of the DNA sugar-phosphate backbone. We remark that given an efficient enumeration of the Levenshtein ball about any point, these methods can be extended to utilize that metric, too.

In order to define hairpin-loop-avoiding sequences, we first present some notation. An involution on Σ is a mapping $x \mapsto \bar{x}$ such that for all $x \in \Sigma$ it holds that $\bar{\bar{x}} = x$; we now assume Σ to be equipped with such an involution (we allow fixed points, in order to account for odd q, which shall not affect our analysis). For example, DNA is composed of four *nucleotide bases*: the *purines*, adenine ($\boldsymbol{A}$) and guanine ($\boldsymbol{G}$), are respectively the complements of the *pyrimidines* thymine ($\boldsymbol{T}$) and cytosine ($\boldsymbol{C}$); when forming a *double helix* (or *duplex*) structure, each base can only stably bond (*hybridize*) with its complement. For $\boldsymbol{x} \in \Sigma^n$, denote $\bar{\boldsymbol{x}} \triangleq \bar{x}(0)\bar{x}(1)\cdots\bar{x}(n-1)$.

Algorithm 2: Hairpin-avoiding Encoder

```
Input: $\boldsymbol{x} \in \Sigma^n$ containing no 0-run of length $z$
Output: $\mathrm{Enc}_1(\boldsymbol{x}) \in \bigcup_{m \leqslant n} \mathcal{RRF}^{\mathrm{s}}_{t_\delta, \ell_a}(m)$
$j' \leftarrow 2(\ell_a - t_\delta)$
while $j' \leqslant |\boldsymbol{x}|$ do
    for $i = j' - 2(\ell_a - t_\delta), \ldots, 0$ do
        $\ell' \leftarrow \min\{\ell_a, \lfloor \frac{j'-i}{2} \rfloor\}$
        $j \leftarrow j' - \ell'$
        if $d\big((\bar{\boldsymbol{x}}_{j+[\ell']})^{\mathrm{r}}, \boldsymbol{x}_{i+[\ell']}\big) \leqslant t_\delta - (\ell_a - \ell')$ then
            Replace $\boldsymbol{x}_{j+[\ell']}$ with $\boldsymbol{s}$ from (11)
            $j' \leftarrow \max\{2(\ell_a - t_\delta), j' - \ell' + 1\}$
            break
        end
    end
    $j' \leftarrow j' + 1$
end
return $\boldsymbol{x}$
```

DNA strands are also oriented: each nucleotide is composed of one of four nitrogenous bases, together with a pentose sugar and a phosphate group; the phosphate groups connect the sugar rings of adjacent nucleotides 5’-end to 3’-end (referring to the five-carbon sites of the sugar rings) to form a long chain (*oligonucleotide*), and thus the orientation can be observed from any segment of the chain. Stable duplexes only form between oligonucleotides of reverse orientations, and therefore coiled-loop secondary structures cannot appear. To capture this notion, we denote for $\boldsymbol{x} \in \Sigma^n$ the *reverse* sequence $\boldsymbol{x}^{\mathrm{r}} \triangleq x(n-1)\cdots x(1)x(0)$.

For integers $t \leqslant \ell$, we define the set of length-n (t,ℓ)-*hairpin avoiding* strings to contain those strings that do not have the potential for the formation of a loop with stem-length ℓ, of which at least $\ell - t$ symbols are hybridized. More precisely,

$$\mathcal{HA}_{t,\ell}(n) \triangleq \left\{ \boldsymbol{x} \in \Sigma^n : \begin{array}{c} \forall 0 \leqslant i < j < n \\ \forall \ell - t \leqslant \ell' \leqslant \min\{\ell, j-i, n-j\} \\ d_{\mathrm{H}}(\boldsymbol{x}_{i+[\ell']}, (\bar{\boldsymbol{x}}_{j+[\ell']})^{\mathrm{r}}) > t - (\ell - \ell') \end{array} \right\}.$$

Observe in the above definition, that for $0 < i < j < n-1$,

$$\begin{aligned} & d_{\mathrm{H}}(\boldsymbol{x}_{i-1+[\ell'+1]}, (\bar{\boldsymbol{x}}_{j+[\ell'+1]})^{\mathrm{r}}) > t - (\ell - \ell') + 1 \\ & \quad \Longrightarrow d_{\mathrm{H}}(\boldsymbol{x}_{i+[\ell']}, (\bar{\boldsymbol{x}}_{j+[\ell']})^{\mathrm{r}}) > t - (\ell - \ell'), \end{aligned}$$

hence some conditions in the above definition are redundant.

As before, for fixed real numbers $a > 1$ and $0 < \delta < 1$, we also make the notation $\mathcal{HA}_{\delta,a}(n)$. We will show that when $a > (1 - \bar{H}_q(\delta))^{-1}$ (for $\delta < 1 - \frac{1}{q}$) then Algorithm 1 can, with slight necessary adjustments, encode into $\bigcup_{m \leqslant n} \mathcal{HA}_{t_\delta, \ell_a}(m)$ with redundancy $O(n^{2 - a(1 - H_q(\delta) - \epsilon)})$ for arbitrarily small $\epsilon > 0$. We leave the interesting problem of stating an analogue of Lemma 16 in this case for future study.

Indeed, the encoder presented in Algorithm 2 differs from Algorithm 1 only in the type of condition in the inner loop (ranges for j, i are adjusted accordingly); if a replacement is required, instead of necessarily replacing an entire ℓ-substring, in the case $i > j-\ell$ (i.e., $\ell' < \ell$) only the ℓ'-suffix is replaced, with the substring

$$\boldsymbol{s} \triangleq 0^z 1 \circ E(j-i) \circ 10^{z'} 1 \circ E(\mathrm{en}(\boldsymbol{e})) \circ 1. \tag{11}$$

For convenience we repeat the previous definitions for the following expressions:

- $\zeta = a(1 - H_q(\delta) - \epsilon) - 1 > 0$;
- $z = \lfloor \zeta \log_q(n) \rfloor$;
- $j - i$ represents the q-ary expansion of the difference (using only as many symbols as required),

and we adjust the following definitions:

- $\boldsymbol{e} \triangleq (\bar{\boldsymbol{x}}_{j+[\ell']})^{\mathrm{r}} - \boldsymbol{x}_{i+[\ell']}$;
- $z' \triangleq \lceil \log_q(\ell) \rceil + 1$; and, finally,
- $E(\cdot)$ is an explicit and efficient encoder into strings containing no 0-runs of length z', accepting inputs of lengths at most ℓ and requiring a single redundant symbol [39, Alg. 1]. We shall see below that both $\log(j-i), |\mathrm{en}(\boldsymbol{e})| \leqslant \ell$.

The analysis of Algorithm 2 is much similar to that of Algorithm 1 in Section IV-B. We summarize the result in the following theorem.

Theorem 29 *If $(1-\bar{H}_q(\delta))^{-1} < a < \delta^{-1}(1-\bar{H}_q(\delta))^{-1}$, then for sufficiently large n Algorithm 2 terminates, $\mathrm{Enc}_2(\boldsymbol{x}) \in \bigcup_{m \leqslant n} \mathcal{HA}_{t,\ell}(m)$, and $\boldsymbol{x}$ can be decoded from it.*

Proof: We prove only the first part; the latter two follow exactly as in the proof of Theorem 23. As before,

$$|\boldsymbol{s}| < 9 + \log(\ell) + \zeta \log_q(n) + \log_q(j-i) + |\mathrm{en}(\boldsymbol{e})|.$$

Repeating the analysis of Theorem 23, if $j-i \geqslant \ell$ (i.e., $\ell' = \ell$), then (bounding $\log_q(j-i) \leqslant \log_q(n)$) we have

$$\begin{aligned} |\mathrm{en}(\boldsymbol{e})| &\leqslant \lceil \ell \bar{H}_q(t/\ell) \rceil \leqslant \lceil \ell_a H_q(\delta) \rceil \\ &< a H_q(\delta) \log_q(n) + 1, \end{aligned}$$

as before, and hence again (for sufficiently large n)

$$\begin{aligned} |\boldsymbol{s}| &< 9 + \log_q(\ell) + (\zeta + 1 + a H_q(\delta)) \log_q(n) \\ &= 9 + \log_q(\ell) + a(1-\epsilon) \log_q(n) < \ell. \end{aligned}$$

It follows that such an iteration of Algorithm 2 also shortens $\boldsymbol{x}$.

Otherwise, when $\ell' < \ell$ we again bound $\log_q(j-i) \leqslant \log_q(\ell') < \log_q(\ell)$ and

$$\begin{aligned} |\mathrm{en}(\boldsymbol{e})| &\leqslant \lceil \ell' \bar{H}_q\big(\tfrac{t-(\ell-\ell')}{\ell'}\big) \rceil \\ &\leqslant \ell' \bar{H}_q\big(\tfrac{\ell'-(1-\delta)\ell}{\ell'}\big) + 1 \\ &< \ell' H_q(\delta) + 1. \end{aligned}$$

Hence,

$$\begin{aligned} |\boldsymbol{s}| &< 10 + 2\log_q(\ell) + \zeta \log_q(n) + H_q(\delta)\ell' \\ &= 10 + 2\log_q(\ell) - (\epsilon a + 1)\log_q(n) \\ &\quad + (1 - H_q(\delta)) a \log_q(n) + H_q(\delta)\ell' \\ &\leqslant 11 + 2\log_q(\ell) - (\epsilon a + 1)\log_q(n) \\ &\quad + (1 - H_q(\delta))\ell + H_q(\delta)\ell' \\ &\leqslant 11 + 2\log_q(\ell) - (\epsilon a + 1)\log_q(n) \\ &\quad + (1 - H_q(\delta))(\ell - \ell') + \ell' \\ &\leqslant (11 + 2\log_q(\ell) - \epsilon a \log_q(n)) + \ell' \\ &\quad + ((1 - H_q(\delta))\delta a - 1)\log_q(n) < \ell', \end{aligned}$$

where the last inequality again holds for sufficiently large n, and relies on the theorem's assumption. ■

Corollary 30 *For all $a > 1$ and sufficiently small $\delta > 0$, there exists an efficient (explicit) encoder from $\Sigma^{n-o(n)}$ into $\bigcup_{m \leqslant n} \mathcal{HA}_{t,\ell}(m)$, for sufficiently large n.*

The problems of encoding directly into $\mathcal{HA}_{t,\ell}(n)$, more precisely bounding its redundancy, as well as generalization to the Levenshtein metric (hence, considering also the formation of bulge-loop secondary structures), are left for future study.

Acknowledgments

The authors gratefully acknowledge the invaluable insights provided by the anonymous reviewers and associate editor, which were instrumental in improving the presentation of the paper.

References

[1] G. M. Church, Y. Gao, and S. Kosuri, "Next-generation digital information storage in DNA," *Science*, vol. 337, no. 6102, pp. 1628–1628, 2012.

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

[3] 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.

[4] S. L. Shipman, J. Nivala, J. D. Macklis, and G. M. Church, "CRISPR-Cas encoding of a digital movie into the genomes of a population of living bacteria," *Nature*, vol. 547, p. 345, Jul. 2017.

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

[6] D. Heider and A. Barnekow, "DNA-based watermarks using the DNA-Crypt algorithm," *BMC Bioinf.*, vol. 8, no. 1, pp. 176–185, May 2007.

[7] 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, 2012.

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

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

[10] R. Heckel, G. Mikutis, and R. N. Grass, "A characterization of the DNA data storage channel," *Scientific reports*, vol. 9, no. 1, p. 9663, Jul. 2019.

[11] O. Sabary, Y. Orlev, R. Shafir, L. Anavy, E. Yaakobi, and Z. Yakhini, "SOLQC: Synthetic oligo library quality control tool," *Bioinformatics*, vol. 37, no. 5, pp. 720–722, Mar. 2021.

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

[13] A. Lenz, P. H. Siegel, A. Wachter-Zeh, and E. Yaakobi, "Coding over sets for DNA storage," *IEEE Trans. Inf. Theory*, vol. 66, no. 4, pp. 7682–7696, Apr. 2020.

[14] J. Sima, N. Raviv, and J. Bruck, “Robust indexing - optimal codes for DNA storage,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Los Angeles, CA, USA, Jun. 2020, pp. 717–722.

[15] ——, “On coding over sliced information,” *IEEE Trans. Inf. Theory*, vol. 67, no. 5, pp. 2793–2807, May 2021.

[16] E. Ukkonen, “Approximate string-matching with q-grams and maximal matches,” *Theoretical Computer Science*, vol. 92, no. 1, pp. 191–211, Jan. 1992.

[17] J. Acharya, H. Das, O. Milenkovic, A. Orlitsky, and S. Pan, “String reconstruction from substring compositions,” *SIAM J. Discrete Math.*, vol. 29, no. 3, pp. 1340–1371, 2015.

[18] I. Shomorony, T. A. Courtade, and D. Tse, “Fundamental limits of genome assembly under an adversarial erasure model,” *IEEE Trans. Mol., Bio. and Multi-Scale Commun.*, vol. 2, no. 2, pp. 199–208, Dec. 2016.

[19] Z. Chang, J. Chrisnata, M. F. Ezerman, and H. M. Kiah, “Rates of DNA sequence profiles for practical values of read lengths,” *IEEE Trans. Inf. Theory*, vol. 63, no. 11, pp. 7166–7177, Nov. 2017.

[20] R. Gabrys and O. Milenkovic, “Unique reconstruction of coded strings from multiset substring spectra,” *IEEE Trans. Inf. Theory*, vol. 65, no. 12, pp. 7682–7696, Dec. 2019.

[21] O. Elishco, R. Gabrys, M. Médard, and E. Yaakobi, “Repeat-free codes,” *IEEE Trans. Inf. Theory*, vol. 67, no. 9, pp. 5749–5764, Sep. 2021.

[22] S. Marcovich and E. Yaakobi, “Reconstruction of strings from their substrings spectrum,” *IEEE Trans. Inf. Theory*, vol. 67, no. 7, pp. 4369–4384, Jul. 2021.

[23] J. Chrisnata, H. M. Kiah, S. Rao Karingula, A. Vardy, E. Yaakobi, and H. Yao, “On the number of distinct k-decks: Enumeration and bounds,” *Advances in Mathematics of Communications*, vol. 17, no. 4, pp. 960–978, Aug. 2023.

[24] A. Lenz, P. H. Siegel, A. Wachter-Zeh, and E. Yaakobi, “Anchor-based correction of substitutions in indexed sets,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, Jul. 2019, pp. 757–761.

[25] N. Raviv, M. Schwartz, and E. Yaakobi, “Rank-modulation codes for DNA storage with shotgun sequencing,” *IEEE Trans. Inf. Theory*, vol. 65, no. 1, pp. 50–64, Jan. 2019.

[26] N. Beeri and M. Schwartz, “Improved rank-modulation codes for DNA storage with shotgun sequencing,” *IEEE Trans. Inf. Theory*, vol. 68, no. 6, pp. 3719–3730, Jun. 2022.

[27] S. Jain, F. Farnoud, M. Schwartz, and J. Bruck, “Noise and uncertainty in string-duplication systems,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 3120–3124.

[28] N. Alon, J. Bruck, F. Farnoud, and S. Jain, “Duplication distance to the root for binary sequences,” *IEEE Trans. Inf. Theory*, vol. 63, no. 12, pp. 7793–7803, Dec. 2017.

[29] F. Farnoud, M. Schwartz, and J. Bruck, “Estimation of duplication history under a stochastic model for tandem repeats,” *BMC Bioinf.*, vol. 20, no. 1, pp. 64–74, Feb. 2019.

[30] O. Milenkovic and N. Kashyap, “On the design of codes for DNA computing,” in *Proc. Int. Workshop on Coding and Cryptography (WCC), 2005, Bergen, Norway*, ser. Lecture Notes in Computer Science, Ø. Ytrehus, Ed., vol. 3969. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2006, pp. 100–119.

[31] ——, “DNA codes that avoid secondary structures,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Adelaide, SA, Australia, Sep. 2005, pp. 288–292.

[32] K. G. Benerjee and A. Banerjee, “On homopolymers and secondary structures avoiding, reversible, reversible-complement and GC-balanced DNA codes,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Espoo, Finland, Jun. 2022, pp. 204–209.

[33] T. T. Nguyen, K. Cai, H. M. Kiah, D. T. Dao, and K. A. Schouhamer Immink, “On the design of codes for DNA computing: Secondary structure avoidance codes,” *arXiv preprint arXiv:2302.13714v1*, Feb. 2023. [Online]. Available: https://arxiv.org/abs/2302.13714v1

[34] D. Bar-Lev, A. Kobovich, O. Leitersdorf, and E. Yaakobi, “Universal framework for parametric constrained coding,” *arXiv preprint arXiv:2212.09314v1*, Apr. 2023. [Online]. Available: https://arxiv.org/abs/2304.01317v1

[35] J. Spencer, “Asymptotic lower bounds for Ramsey functions,” *Discrete Math.*, vol. 20, pp. 69–76, 1977.

[36] R. M. Roth, *Introduction to Coding Theory*. Cambridge Univ. Press, 2006.

[37] T. M. Cover, “Enumerative source encoding,” *IEEE Trans. Inf. Theory*, vol. 19, no. 1, pp. 73–77, Jan. 1973.

[38] Y. Yehezkeally, D. Bar-Lev, S. Marcovich, and E. Yaakobi, “Generalized unique reconstruction from substrings,” *IEEE Trans. Inf. Theory*, vol. 69, no. 9, pp. 5648–5659, Sep. 2023.

[39] M. Levy and E. Yaakobi, “Mutually uncorrelated codes for DNA storage,” *IEEE Trans. Inf. Theory*, vol. 65, no. 6, pp. 3671–3691, Jun. 2019.

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

Yonatan Yehezkeally (S’12–M’20) received the B.Sc. degree (*cum laude*) in mathematics and the M.Sc. (*summa cum laude*) and Ph.D. degrees in electrical and computer engineering from Ben-Gurion University of the Negev, Beer-Sheva, Israel, in 2013, 2017, and 2020 respectively. He is currently a Carl Friedrich von Siemens Post-Doctoral Research Fellow of the Alexander von Humboldt Foundation, with the Associate Professorship of Coding and Cryptography (Prof. Wachter-Zeh), School of Computation, Information and Technology, Technical University of Munich. His research interests include coding theory and algorithms, particularly with applications to novel storage media, with a focus on DNA-based storage and nascent sequencing technologies. They further include combinatorial analysis and structures, as well as algebraic structures.

Nikita Polyanskii (Member, IEEE) received the Specialist degree in mathematics and the Ph.D. degree in mathematics from the Lomonosov Moscow State University, Russia, in 2013 and 2016, respectively. From 2015 to 2017, he was a Researcher with the Institute for Information Transmission Problems, Russia, and a Senior Research Engineer with the Huawei Moscow Research Center, Russia. From 2017 to 2021, he was a Post-Doctoral Researcher with the Technion—Israel Institute of Technology, the Skolkovo Institute of Science and Technology, and the Technical University of Munich. He is currently a Senior Research Engineer with the IOTA Foundation, Germany. His research interests include the theory of error-correcting codes and distributed ledger technologies.