Rates of DNA Sequence Profiles for Practical Values of Read Lengths

Zuling Chang, Johan Chrisnata, Martianus Frederic Ezerman, and Han Mao Kiah

***Abstract*—A recent study by one of the authors has demonstrated the importance of profile vectors in DNA-based data storage. We provide exact values and lower bounds on the number of profile vectors for finite values of alphabet size q, read length ℓ, and word length n. Consequently, we demonstrate that for $q \geq 2$ and $n \leq q^{\ell/2-1}$, the number of profile vectors is at least $q^{\kappa n}$ with κ very close to 1. In addition to enumeration results, we provide a set of efficient encoding and decoding algorithms for certain families of profile vectors.**

***Index Terms*—DNA-based data storage, profile vectors, Lyndon words, synchronization, de Bruijn sequences.**

1. Introduction

Despite advances in traditional data recording techniques, the emergence of Big Data platforms and energy conservation issues impose new challenges to the storage community in terms of identifying high volume, nonvolatile, and durable recording media. The potential for using macromolecules for ultra-dense storage was recognized as early as in the 1960s. Among these macromolecules, DNA molecules stand out due to their biochemical robustness and high storage capacity.

In the last few decades, the technologies for synthesizing (writing) artificial DNA and for massive sequencing (reading) have reached attractive levels of efficiency and accuracy. Building upon the rapid growth of DNA synthesis and sequencing technologies, two laboratories recently outlined architectures for archival DNA-based storage [2], [3]. The first architecture achieved a density of 700 TB/gram, while the second approach raised the density to 2.2 PB/gram. To further protect against errors, Grass *et al.* later incorporated Reed-Solomon error-correction schemes and encapsulated the DNA media in silica [4]. Yazdi *et al.* recently proposed a completely different approach that provided a random access and rewritable DNA-based storage system [5], [7]. More recently, a series of experiments by various groups [8]–[10] demonstrate the feasibility of DNA-based storage systems.

To control specialized errors arising from sequencing platforms, two families of codes were introduced by Gabrys *et al.* [11] and Kiah *et al.* [12]. The former looked at miniaturized nanopore sequencers such as MinION, while the latter focused on errors arising from high-throughput sequencers such as Illumina, which is arguably the more mature technology. The latter forms the basis for this work. In particular, we examine the concept of *DNA profile vectors* introduced by Kiah *et al.* [12].

In this channel model, to store and retrieve information in DNA, one starts with a desired information sequence encoded into a sequence or word defined over the nucleotide alphabet $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$. The *DNA storage channel* models a physical process which takes as its input the sequence of length n, and synthesizes (writes) it physically into a macromolecule string. To retrieve the information, the user can choose from several read technologies. The most common sequencing process, implemented by Illumina, makes numerous copies of the string or amplifies the string, and then fragments all copies of the string into a collection of substrings (reads) of approximately the same length ℓ, so as to produce a large number of overlapping *reads*. Since the concentration of all (not necessarily) distinct substrings within the mix is usually assumed to be uniform, one may normalize the concentration of all subsequences by the concentration of the least abundant substring. As a result, one actually observes substring concentrations reflecting the frequency of the substrings in *one copy* of the original string. Therefore, we model the output of the channel as an *unordered subset of reads*. This set may be summarized by its multiplicity vector, which we call the *output profile vector*.

We assume an errorless channel and observe that it is possible for different words or strings to have an identical profile vector. Hence, even without errors, the channel is unable to distinguish between certain pairs of words. This work enumerates all distinct profile vectors for fixed values of n and ℓ over a q-ary alphabet. The number of distinct profile vectors is therefore the maximum number of messages that can be sent over this channel. Our study proves a fundamental limit for this *noiseless* channel. The study of capacity of a noisy version of this channel remains to be determined. Nevertheless, in Section 4-C, we construct a set of profile vectors and propose an associated assembly algorithm that reconstructs the original message in the presence of *coverage* errors (see Section 4-C for definition).

In the case of arbitrary ℓ-substrings, the problem of enumerating all valid profile vectors was addressed by Jacquet *et al.* in

Z. Chang is with the School of Mathematics and Statistics, Zhengzhou University, China, e-mail: zuling_chang@zzu.edu.cn.

J. Chrisnata, M. F. Ezerman, and H. M. Kiah are with the School of Physical and Mathematical Sciences, Nanyang Technological University, 21 Nanyang Link, Singapore 637371, e-mails: {jchrisnata, fredezerman, hmkiah}@ntu.edu.sg.

The work of Z. Chang was supported by the National Natural Science Foundation of China under Grants 61772476 and U1304604. Research Grant TL-9014101684-01 supported the work of M. F. Ezerman. J. Chrisnata's work was supported in part by the Singapore Ministry of Education under Grant MOE2016-T1-001-156. The research of H. M. Kiah was supported by the Singapore Ministry of Education under Grants MOE2015-T2-2-086 and MOE2016-T1-001-156.

Earlier results of this paper were presented at ISIT 2016 [1].

the context of "Markov types" [13]. Kiah *et al.* then extended the enumeration results to profiles with specific ℓ-substring constraints so as to address certain considerations in DNA sequence design [12]. In particular, for fixed values of q and ℓ, the number of profile vectors is known to be $\Theta\left(n^{q^\ell - q^{\ell-1}}\right)$.

However, determining the coefficient for the dominating term $n^{q^\ell - q^{\ell-1}}$ is a computationally difficult task. It has been determined for only very small values of q and ℓ in [12], [13]. Furthermore, it is unclear how accurate the asymptotic estimate $\Theta\left(n^{q^\ell - q^{\ell-1}}\right)$ is for practical values of n. Indeed, most current DNA storage systems do not use string lengths n exceeding several thousands nucleotides (nts) due to the high cost of synthesis. On the other hand, current sequencing systems have read length ℓ between 100 to 1500 nts.

This paper adopts a different approach and looks for lower bounds for the number of profile vectors given moderate values of q, ℓ, and n. Surprisingly, for fixed $q \geq 2$ and moderately large values $n \leq q^{\ell/2-1}$, the number of profile vectors is at least $q^{\kappa n}$ with κ very close to 1. As an example, when $q = 4$ (the number of DNA nucleotide bases) and $\ell = 100$ (a practical read length), our results show that there are at least $4^{0.99n}$ distinct 100-gram profile vectors for $1000 \leq n \leq 10^6$. In other words, for practical values of read and word lengths, we are able to obtain a set of distinct profile vectors with rates *close to one*. In addition to enumeration results, we propose a set of linear-time encoding and decoding algorithms for each of two particular families of profile vectors.

2. Preliminaries and Main Results

Let $[\![q]\!]$ denote the set of integers $\{0, 1, \ldots, q-1\}$ and $[i, j]$ denote the set of integers $\{i, i+1, \ldots, j\}$. Consider a word $\mathbf{x} = x_1 x_2 \cdots x_n$ of length n over $[\![q]\!]$. For $1 \leq i < j \leq n$, we denote the entry x_i by $\mathbf{x}[i]$, the *substring* $x_i x_{i+1} \cdots x_j$ of length $(j - i + 1)$ by $\mathbf{x}[i, j]$, and the length of $\mathbf{x}$ by $|\mathbf{x}|$. The *concatenation* of two strings $\mathbf{x}$ and $\mathbf{y}$ is denoted by $\mathbf{xy}$.

For $\ell \leq n$ and $1 \leq i \leq n-\ell+1$, we call the substring $\mathbf{x}[i, i+\ell-1]$ an *ℓ-gram* of $\mathbf{x}$. For $\mathbf{z} \in [\![q]\!]^\ell$, let $p(\mathbf{x}, \mathbf{z})$ denote the number of occurrences of $\mathbf{z}$ as an ℓ-gram of $\mathbf{x}$. Let $\mathbf{p}(\mathbf{x}, \ell) \triangleq \Big(p(\mathbf{x}, \mathbf{z})\Big)_{\mathbf{z} \in [\![q]\!]^\ell}$ be the (ℓ-gram) *profile vector* of length q^ℓ, indexed by all words of $[\![q]\!]^\ell$ ordered lexicographically. Let $\mathcal{F}(\mathbf{x}, \ell)$ be the set of ℓ-grams of $\mathbf{x}$. In other words, $\mathcal{F}(\mathbf{x}, \ell)$ is the support for the vector $\mathbf{p}(\mathbf{x}, \ell)$.

Example 2.1. Let $q = 2$, $n = 5$ and $\ell = 2$. Then $p(10001, 01) = p(10001, 10) = 1$, while $p(10001, 00) = 2$. So, $\mathbf{p}(10001, 2) = (2, 1, 1, 0)$ and $\mathcal{F}(10001, 2) = \{00, 01, 10\}$.

Consider the words 00010 and 00101. Then $\mathbf{p}(10001, 2) = \mathbf{p}(00010, 2)$ while $\mathcal{F}(10001, 2) = \mathcal{F}(00010, 2) = \mathcal{F}(00101, 2)$.

As illustrated by Example 2.1, different words may have the same profile vector. We define a relation on $[\![q]\!]^n$ where $\mathbf{x} \sim \mathbf{x}'$ if and only if $\mathbf{p}(\mathbf{x}, \ell) = \mathbf{p}(\mathbf{x}', \ell)$. It can be shown that $\sim$ is an equivalence relation and we denote the number of equivalence classes by $P_q(n, \ell)$. We further define the *rate of profile vectors* to be $R_q(n, \ell) = \log_q P_q(n, \ell)/n$.

The asymptotic growth of $P_q(n, \ell)$ as a function of n is given as below.

Theorem 2.1 (Jacquet *et al.* [13], Kiah *et al.* [12])**.** Fix $q \geq 2$ and ℓ. Then

$$P_q(n, \ell) = \Theta\left(n^{q^\ell - q^{\ell-1}}\right).$$

Hence, $\lim_{n\to\infty} R_q(n, \ell) = 0$.

Theorem 2.1 states that the rate of profile vectors tends to zero for fixed q and ℓ. However, it was unclear how fast the rate converges. We demonstrate in this work that the rate converges very slowly and that for moderate values of n, the rates are close to one. In fact, as long as $n \leq q^{\ell/2-1}$, we show that $R_q(n, \ell)$ is at least $1 - o(n)$.

Formally, we summarize our main contributions in Theorem 2.2. They provide exact values and lower bounds for $P_q(n, \ell)$ for finite values of n, q and ℓ. As the proofs of Equations (1), (2), (3), and (4) require a variety of combinatorial tools, we defer the detailed proofs to Sections 3, 4, and 5. Here, we discuss the significance of our results.

Theorem 2.2. Fix $q \geq 2$. Let μ be the Möbius function and $F_q(n, b)$ be as defined by (9) in Section 4-B.

(i) If $\ell \leq n < 2\ell$, then

$$\begin{aligned} P_q(n, \ell) &= q^n - \sum_{r|n-\ell+1} \sum_{t|r} \left(\frac{r-1}{r}\right) \mu\left(\frac{r}{t}\right) q^t \\ &> q^{n-1}(q-1). \end{aligned} \tag{1}$$

(ii) Choose a and m such that $2a \leq \ell$ and $m \leq q^{a-1}$. If $n = m\ell$, then

$$P_q(n, \ell) \geq (q-1)^{m(\ell-a)}. \tag{2}$$

(iii) Choose b, a, and m such that $2a \leq \ell$, $b + 1 < a$ and $m \leq F_q(a - b - 1, b)$. If $n = m\ell$, then

$$P_q(n, \ell) \geq \left(F_q(n - a - 1, b)(q-1)\right)^m. \tag{3}$$

(iv) If $n \geq \ell$,

$$P_q(n, \ell) > \frac{1}{n}\left(\sum_{t|n} \mu\left(\frac{n}{t}\right) q^t - \binom{n}{2} q^{n-\ell+1}\right). \tag{4}$$

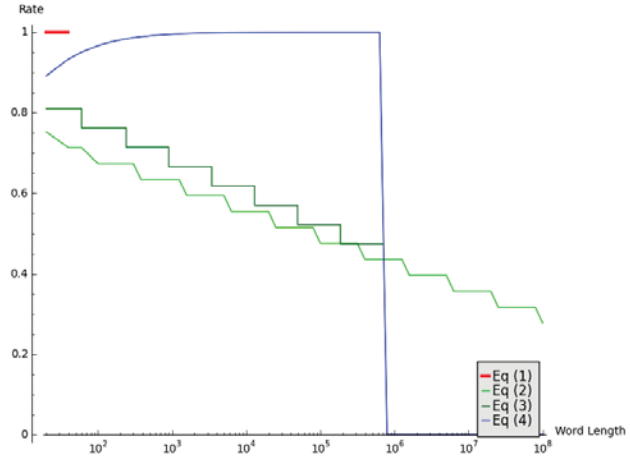
Suppose further that $\ell \geq 2\log_q n + 2$ and $n \geq 8$. Then $P_q(n, \ell) > q^{n-1}/n$. Hence, for all $0 < \kappa < 1$, we have $P_q(n, \ell) \geq q^{\kappa n}$ for sufficiently large n.

Discussion on Theorem 2.2

We compare the *rates* provided in Theorem 2.2 with $q = 4$ in Figures 1 and 2.

Rates for Moderate Read Lengths. Figure 1(b) shows that for a practical read length $\ell = 100$ and word lengths $n \leq 10^8$, the rates of the profile vectors is very close to one. In fact, computations show that $R_4(n, \ell) \geq 0.99$ for $1000 \leq n \leq 10^6$. Even for a shorter read length $\ell = 20$, Figure 1(a) illustrates that the rates are close to one for word lengths $n \leq 10^5$. On the other hand, Kosuri and Church [14] reported that current technologies are only able to synthesize DNA strings of length up to 10^4 bases at cost of 1000 USD per base, and DNA strings of length up to 100-200 bases at cost less than one USD per base. Therefore, for practical values of read and word lengths,

(a) The rates $R_4(n, 20)$ for $20 \leq n \leq 10^8$



(b) The rates $R_4(n, 100)$ for $100 \leq n \leq 10^8$

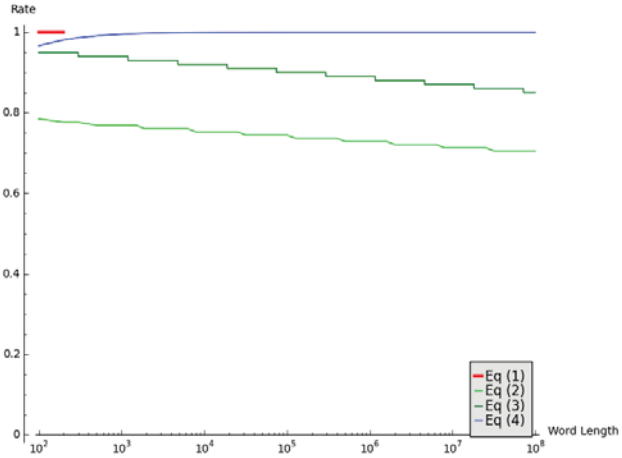
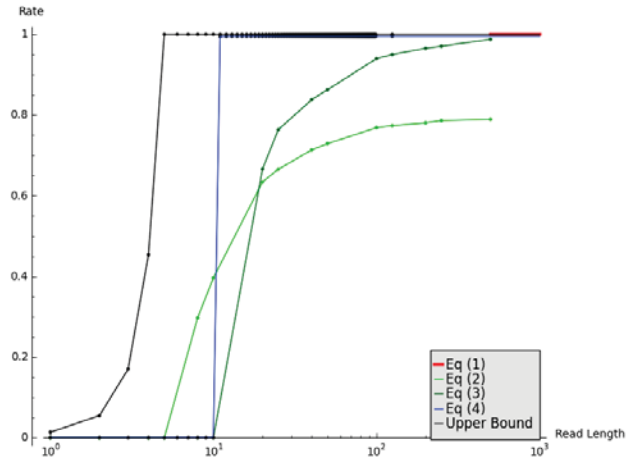


Fig. 1. Rate of profile vectors for fixed values of ℓ.

(a) The rates $R_4(10^3, \ell)$ for $1 \leq \ell \leq 10^3$



(b) The rates $R_4(10^6, \ell)$ for $1 \leq \ell \leq 10^6$

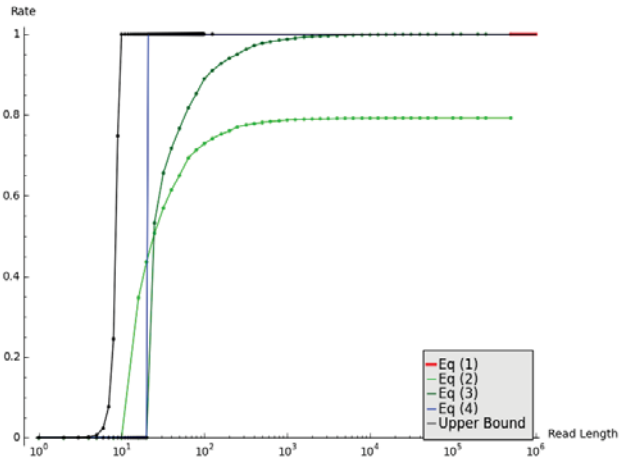


Fig. 2. Rate of profile vectors for fixed values of n.

we obtain a set of distinct profile vectors with rates close to one.

Sharpness of Estimates. In Figure 2, we plot an upper bound for $R_q(n, \ell)$, given by

$$R_q(n, \ell) \leq \frac{1}{n} \log_q \binom{n - \ell + q^\ell}{q^\ell - 1}. \tag{5}$$

Here, the inequality follows from the fact that a profile vector is an integer-valued vector of length q^ℓ, whose entries sum to $n-\ell+1$. Such vectors are also known as *weak q^ℓ-compositions of* $(n - \ell + 1)$ and their number is given by $\binom{n-\ell+q^\ell}{q^\ell-1}$ (see for example Stanley [15, Section 1.2]), and hence, the inequality follows.

Figure 2 illustrates that if we fix the word length n, we have $R_q(n, \ell) \approx 1$ for $\ell \geq 2 \log_q n$ and $R_q(n, \ell) \approx 0$ for $\ell \leq \log_q n$. In other words, we have determined $R_q(n, \ell)$ except for a small regime where $\log_q n \leq \ell \leq 2 \log_q n$ We will state this observation formally later in Corollary 2.3 and provide an asymptotic analysis for the rates of profile vectors.

Efficient Encoding and Decoding. Define encoding and decoding as follows. Consider a given family of profile vectors $\mathcal{C}$ and set of data strings $\mathcal{X}$ such that $|\mathcal{C}| = |\mathcal{X}|$. Two maps $\texttt{encode} : \mathcal{X} \to \mathcal{C}$ and $\texttt{decode} : \mathcal{C} \to \mathcal{X}$ form an *encoding and decoding pair* if $\texttt{decode} \circ \texttt{encode}(\mathbf{c}) = \mathbf{c}$ for all $\mathbf{c} \in \mathcal{X}$.

Figures 1 and 2 shows that, for $n \leq q^{\ell/2-1}$, *i.e.* $\ell \geq 2 \log_q n + 2$, Equation (4) provides a significantly better lower bound than Equations (2) and (3). Unfortunately, the proof for Equation (4) is nonconstructive and we are only able to demonstrate a set of efficient encoding and decoding algorithms for the families of profile vectors associated with Equations (1), (2), and (3). Furthermore, we provide an efficient sequence reconstruction algorithm for the family of profile vectors associated with Equations (2) and (3).

Asymptotic Rates. Let n be a function of ℓ, denoted by $n = f(\ell)$, such that n increases with ℓ. We then define the *asymptotic rate of profile vectors with respect to* n via the equation

$$\alpha(n, q) \triangleq \limsup_{\ell \to \infty} R_q(n, \ell). \tag{6}$$

Suppose that ℓ is a system parameter determined by current sequencing technology. Then $n = f(\ell)$ determines how long we can set our codewords so that $\alpha(n, q)$ remains to be the information rate of the DNA storage channel. From Theorem 2.2, we derive the following result on the asymptotic rates.

Corollary 2.3 (Asymptotic rates)**.** Fix $q \geq 2$ and let $n = f(\ell)$.

(i) Suppose that $\ell \leq n \leq q^{\ell/2-1}$ for all ℓ. Then $\alpha(n, q) = 1$.
(ii) Let $\epsilon > 0$. Suppose that $n \geq q^{(1+\epsilon)\ell}$ for all ℓ. Then $\alpha(n, q) = 0$.

Proof. If $n \leq q^{\ell/2-1}$, or $\ell \geq 2\log_q n+2$, then apply Theorem 2.2(iv) to obtain $R_q(n, \ell) \geq (n - 1 - \log_q n)/n$. After taking limits, we have $\alpha(n, q) = 1$, proving (i). Next from (5), we have that, for $n \geq 2$, $\ell \geq 2$,

$$P_q(n, \ell) \leq \binom{n - \ell + q^\ell}{q^\ell - 1} = \prod_{j=1}^{q^\ell - 1} \frac{n - \ell + j + 1}{j} \leq n^{q^\ell - 1}.$$

Then $R_q(n, \ell) \leq (q^\ell - 1)(\log_q n)/n$. Fix $\epsilon > 0$. If $n \geq q^{(1+\epsilon)\ell}$, then $q^\ell \leq n^{1/(1+\epsilon)}$. So,

$$R_q(n, \ell) \leq \frac{(q^\ell - 1)\log_q n}{n} \leq \frac{n^{1/(1+\epsilon)}\log_q n}{n} = \frac{\log_q n}{n^{\epsilon/(1+\epsilon)}}.$$

After taking limits, we have $\alpha(n, q) = 0$, proving (ii). □

Finally, we comment that our results on asymptotic rates cover the instance where $n = \beta\ell$, where β is a constant greater than one. In other words, the read length ℓ is a fraction of the word length. Indeed, if $n = f(\ell) = \beta\ell$, then $f(\ell) \leq q^{\ell/2-1}$ for sufficiently large ℓ. Hence, Corollary 2.3(i) applies and we have $\alpha(n, q) = 1$.

3. Exact Enumeration of Profile Vectors

We extend the methods of Tan and Shallit [16], where the number of possible $\mathcal{F}(\mathbf{x}, \ell)$ was determined for $\ell \leq n < 2\ell$. Specifically, we determine $P_q(n, \ell)$ for $\ell \leq n < 2\ell$. Our strategy is to first define an equivalence relation using the notions of root conjugates so that the number of equivalence classes yields $P_q(n, \ell)$. We then find this number using standard combinatorial methods.

Definition 3.1. Let $\mathbf{x}$ be a q-ary word. A *period* of $\mathbf{x}$ is a positive integer r such that $\mathbf{x}$ can be *factorized* as

$$\mathbf{x} = \underbrace{\mathbf{u}\,\mathbf{u}\,\cdots\,\mathbf{u}}_{k \text{ times}}\mathbf{u}', \text{ with } |\mathbf{u}| = r, \mathbf{u}' \text{ a prefix of } \mathbf{u}, \text{ and } k \geq 1.$$

Let $\pi(\mathbf{x})$ denote the *minimum period* of $\mathbf{x}$. The *root* of $\mathbf{x}$ is given by $\mathbf{h}(\mathbf{x}) = \mathbf{x}[1, \pi(\mathbf{x})]$, which is the prefix of $\mathbf{x}$ with length $\pi(\mathbf{x})$. Two words $\mathbf{x}$ and $\mathbf{x}'$ are said to be *root-conjugate* if $\mathbf{h}(\mathbf{x}) = \mathbf{uv}$ and $\mathbf{h}(\mathbf{x}') = \mathbf{vu}$ for some words $\mathbf{u}$ and $\mathbf{v}$, or $\mathbf{h}(\mathbf{x})$ is a *rotation* of $\mathbf{h}(\mathbf{x}')$.

Example 3.1. 10010010 has minimum period three and its root is 100. Also, 01001001 has minimum period three and its root is 010. Therefore, 10010010 and 01001001 are root-conjugates.

Observe that two words which are root-conjugates necessarily have the same minimum period and it can be shown that being root-conjugates form an equivalence relation. In addition, we have the following technical lemma.

Lemma 3.1. Let $\ell < n < 2\ell$. Let $\mathbf{x}$ be a word of length n with $\pi(\mathbf{x}) \leq n - \ell + 1 \leq \ell$. Then, for $1 \leq i < j \leq \pi(\mathbf{x})$, we have $\mathbf{x}[i, i + \ell - 1] \neq \mathbf{x}[j, j + \ell - 1]$.

Proof. Suppose that $\mathbf{x}[i, i + \ell - 1] = \mathbf{x}[j, j + \ell - 1]$. Setting $k = j - i$, we have $\mathbf{x}[s] = \mathbf{x}[s + k]$ for $i \leq s \leq i + \ell - 1$. Since $\pi(\mathbf{x}) < \ell$, then $\mathbf{x}[s] = \mathbf{x}[s + k]$ for $1 \leq s \leq \pi(\mathbf{x})$. Therefore, $\mathbf{x}[s] = \mathbf{x}[s + d]$ for $1 \leq s \leq \pi(\mathbf{x})$ where $d = \gcd(k, \pi(\mathbf{x})) \leq k = j - i < \pi(\mathbf{x})$. In other words, $\mathbf{x}$ has a period $d < \pi(\mathbf{x})$, contradicting the minimality of $\pi(\mathbf{x})$. □

Tan and Shallit proved the following result that characterized $\mathcal{F}(\mathbf{x}, \ell)$ when $|\mathbf{x}| < 2\ell$.

Lemma 3.2 (Tan and Shallit [16, Theorem 15])**.** Suppose that $\ell \leq n < 2\ell$ and $\mathbf{x}$ and $\mathbf{x}'$ are distinct q-ary words of length n. Then $\mathcal{F}(\mathbf{x}, \ell) = \mathcal{F}(\mathbf{x}', \ell)$ if and only if $\mathbf{x}, \mathbf{x}'$ are root-conjugates with $\pi(\mathbf{x}) \leq n - \ell + 1$.

Using Lemma 3.1, we extend Lemma 3.2 to characterize the profile vectors when $n < 2\ell$.

Theorem 3.3. Let $\mathbf{x}$ and $\mathbf{x}'$ be distinct q-ary words of length n. If $\mathbf{x}, \mathbf{x}'$ are root-conjugates with $\pi(\mathbf{x}) \mid n - \ell + 1$, then $\mathbf{p}(\mathbf{x}, \ell) = \mathbf{p}(\mathbf{x}', \ell)$. Conversely, if $\ell \leq n < 2\ell$ and $\mathbf{p}(\mathbf{x}, \ell) = \mathbf{p}(\mathbf{x}', \ell)$, then $\mathbf{x}, \mathbf{x}'$ are root-conjugates with $\pi(\mathbf{x}) \mid n - \ell + 1$.

Proof. Suppose that $\mathbf{x}$ and $\mathbf{x}'$ are root-conjugates with $\pi(\mathbf{x}) = r$ and $n - \ell + 1 = rs$ for some s. Then it can be verified that $\mathcal{F}(\mathbf{x}, \ell) = \mathcal{F}(\mathbf{x}', \ell) = \{\mathbf{x}[i, i + \ell - 1] : 1 \leq i \leq r\}$ and $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}', \mathbf{z}) = s$ for all $\mathbf{z} \in \mathcal{F}(\mathbf{x}, \ell)$. Therefore, $\mathbf{p}(\mathbf{x}, \ell) = \mathbf{p}(\mathbf{x}', \ell)$.

Conversely, let $\mathbf{p}(\mathbf{x}, \ell) = \mathbf{p}(\mathbf{x}', \ell)$. Then $\mathcal{F}(\mathbf{x}, \ell) = \mathcal{F}(\mathbf{x}', \ell)$. By Lemma 3.2, $\mathbf{x}, \mathbf{x}'$ are root-conjugates with $\pi(\mathbf{x}) \leq n - \ell + 1$. Let $r = \pi(\mathbf{x})$. It remains to show that $r \mid n - \ell + 1$.

Suppose otherwise and let $n - \ell + 1 = rs + t$ with $1 \leq t \leq r - 1$. Let the roots of $\mathbf{x}$ and $\mathbf{x}'$ be $\mathbf{uv}$ and $\mathbf{vu}$, respectively. Therefore, we can write $\mathbf{x}$ and $\mathbf{x}'$ as

$$\mathbf{x} = \overbrace{\underbrace{\mathbf{uv}}_{r}\underbrace{\mathbf{uv}}_{r}\cdots\underbrace{\mathbf{uv}}_{r}}^{s \text{ times}}\underbrace{\mathbf{w}}_{t+\ell-1} \text{ and}$$

$$\mathbf{x}' = \overbrace{\underbrace{\mathbf{vu}}_{r}\underbrace{\mathbf{vu}}_{r}\cdots\underbrace{\mathbf{vu}}_{r}}^{s \text{ times}}\underbrace{\mathbf{w}'}_{t+\ell-1}.$$

We have the following cases.

(i) If $1 \leq t < |\mathbf{u}|$, let $\mathbf{z}'$ be the ℓ-length prefix of $\mathbf{x}'$. Since $|\mathbf{w}'| = t + \ell - 1 \geq \ell$ and $\mathbf{z}'$ is a prefix of $\mathbf{w}'$, we have $p(\mathbf{x}', \mathbf{z}') \geq s + 1$. On the other hand, from Lemma 3.1, the ℓ-gram of $\mathbf{z}'$ can only appear after the first $|\mathbf{u}|$ coordinates of $\mathbf{w}$. However, $|\mathbf{w}| - |\mathbf{u}| < t + (\ell - 1) - t < \ell$, and so, there is no occurrence of $\mathbf{z}'$ as an ℓ-gram of $\mathbf{w}$. Therefore, $p(\mathbf{x}, \mathbf{z}') = s < p(\mathbf{x}', \mathbf{z}')$, contradicting the assumption that $\mathbf{p}(\mathbf{x}, \ell) = \mathbf{p}(\mathbf{x}', \ell)$.
(ii) If $|\mathbf{u}| \leq t \leq r - 1$, let $\mathbf{z} = \mathbf{x}[|\mathbf{u}|, |\mathbf{u}| + \ell - 1]$. Since $|\mathbf{w}| = t + \ell - 1 \geq |\mathbf{u}| + \ell - 1$, we have $p(\mathbf{x}, \mathbf{z}) \geq s + 1$. With the same considerations as before, we check that there is no occurence of $\mathbf{z}$ as an ℓ-gram of $\mathbf{w}'$, So, $p(\mathbf{x}', \mathbf{z}) = s < p(\mathbf{x}, \mathbf{z})$, a contradiction.

We conclude $t = 0$. Hence, $r \mid n - \ell + 1$. □

Hence, for $\ell \leq n < 2\ell$, we have $\mathbf{x} \sim \mathbf{x}'$ if and only if $\mathbf{x}$ and $\mathbf{x}'$ are root-conjugates with $\pi(\mathbf{x}) \mid n-\ell+1$. We compute the number of equivalence classes using this characterization.

A word is said to be *aperiodic* if it is not equal to any of its nontrivial rotations. An aperiodic word of length r is said to be *Lyndon* if it is the lexicographically least word amongst all of its r rotations. Here, the ordering of the symbols follows directly from their order as integers. The number of Lyndon words [17] of length r is given by

$$L_q(r) = \frac{1}{r} \sum_{t|r} \mu\left(\frac{r}{t}\right) q^t. \tag{7}$$

For any integer $r \mid n - \ell + 1$ and any word $\mathbf{x}$, if $\pi(\mathbf{x}) = r$ and $\mathbf{h}(\mathbf{x})$ is its root, then $\mathbf{h}(\mathbf{x})$ is aperiodic and is a rotation of some Lyndon word $\mathbf{u}(\mathbf{x})$. Let $\mathbf{u}(\mathbf{x})$ be the representative of the equivalence class of $\mathbf{x}$. Since there are r rotations of $\mathbf{u}(\mathbf{x})$, there are r words in the equivalence class of $\mathbf{x}$. Therefore, the number of equivalence classes is

$$q^n - \sum_{r|n-\ell+1} (r-1) L_q(r),$$

and, consequently, we obtain (1). We rewrite its statement here for convenience.

Theorem 3.4. Fix $q \geq 2$. Let μ be the Möbius function. If $n < 2\ell$, then

$$\begin{aligned} P_q(n,\ell) &= q^n - \sum_{r|n-\ell+1} \sum_{t|r} \left(\frac{r-1}{r}\right) \mu\left(\frac{r}{t}\right) q^t \\ &> q^{n-1}(q-1). \end{aligned}$$

Example 3.2. Let $n = 5$, $\ell = 4$, and $q = 2$. Consider the words $\mathbf{x} = 10101$ and $\mathbf{x}' = 01010$, which are root-conjugates with minimum period two. Since $2|n - \ell + 1$, it follows from Theorem 3.3 that $\mathbf{x}$ and $\mathbf{x}'$ have the same profile vector.

Conversely, if there are two distinct words $\mathbf{x}$ and $\mathbf{x}'$ such that $\mathbf{x} \sim \mathbf{x}'$, then Theorem 3.3 states that the minimum period of $\mathbf{x}$ divides two. It is then not difficult to argue that the pair of words $\mathbf{x}$ and $\mathbf{x}'$ must be 10101 and 01010. Therefore, the number of distinct profile vectors $P_2(5,4)$ is 31. More generally, in the case $\ell = n - 1$, (1) reduces to $P_q(n,\ell) = q^n - \binom{q}{2}$.

From Theorem 3.3, if $\mathbf{x}$ and $\mathbf{x}'$ are root-conjugates with $\pi(\mathbf{x}) \mid n - \ell + 1$, we have $\mathbf{p}(\mathbf{x},\ell) = \mathbf{p}(\mathbf{x}',\ell)$ for *all* values of n. In other words, the number of equivalence classes computed above provides an upper bound for the number of profile vectors. Formally, we have the following corollary.

Corollary 3.5. For $n \geq 2\ell$,

$$P_q(n,\ell) \leq q^n - \sum_{r|n-\ell+1} \sum_{t|r} \left(\frac{r-1}{r}\right) \mu\left(\frac{r}{t}\right) q^t.$$

Next, we assume $n < 2\ell$ and provide efficient methods to encode and decode messages into q-ary words of length n with distinct ℓ-gram profile vectors. To do so, we make use of the following simple observation from Theorem 3.3.

Lemma 3.6. Let $n < 2\ell$ and $\mathbf{x}$ be a q-ary word of length n such that $\mathbf{x}[\ell-1] \neq \mathbf{x}[n]$. If $\mathbf{x} \sim \mathbf{x}'$, then $\mathbf{x} = \mathbf{x}'$.

Proof. Suppose otherwise that $\mathbf{x}'$ and $\mathbf{x}$ are distinct. Then Theorem 3.3 implies that $\pi(\mathbf{x}) | n-\ell+1$. In other words, $n-\ell+1$ is a period of $\mathbf{x}$ and hence $\mathbf{x}[n] = \mathbf{x}[n-(n-\ell+1)] = \mathbf{x}[\ell-1]$, yielding a contradiction. □

Lemma 3.6 then motivates the encoding and decoding methods presented as Algorithms 1 and 2, respectively, with $\mathcal{C}$ being the image of $\texttt{encode}_1$.

Algorithm 1 $\texttt{encode}_1(\mathbf{c})$

Input: Data string $\mathbf{c}$, where $\mathbf{c} \in [\![q]\!]^{n-1} \times \{1, 2, \ldots, q-1\}$.
Output: $\mathbf{x} \in [\![q]\!]^n$ such that $\mathbf{x}[\ell-1] \neq \mathbf{x}[n]$.

```
if c[n] ≠ c[ℓ − 1] then
    x ← c
else
    x ← append c[1, n − 1] with 0
end if
return x
```

Algorithm 2 $\texttt{decode}_1(\mathbf{x})$

Input: Codeword $\mathbf{x} \in [\![q]\!]^n$.
Output: $\mathbf{c} \in [\![q]\!]^{n-1} \times \{1, 2, \ldots, q-1\}$.

```
if x[n] ≠ 0 then
    c ← x
else
    c ← append x[1, n − 1] with x[ℓ − 1]
end if
return c
```

Example 3.3. Set $q = 2$, $n = 5$, and $\ell = 4$.

Suppose that we encode $\mathbf{c} = 00001 \in [\![2]\!]^4 \times \{1\}$. Applying Algorithm 1, since $\mathbf{c}[3] \neq \mathbf{c}[5]$, then $\mathbf{x} = \texttt{encode}_1(\mathbf{c}) = \mathbf{c}$.

On the other hand, suppose that we encode $\mathbf{c} = 00101 \in [\![2]\!]^4 \times \{1\}$. Applying Algorithm 1, since $\mathbf{c}[3] = \mathbf{c}[5]$, then $\mathbf{x} = 00100$ by setting the last bit of $\mathbf{c}$ to zero.

We compute $\texttt{encode}_1(\mathbf{c})$ for all $\mathbf{c} \in [\![2]\!]^4 \times \{1\}$ to obtain the code $\mathcal{C}$.

$\mathbf{c}$	$\texttt{encode}_1(\mathbf{c})$	$\mathbf{c}$	$\texttt{encode}_1(\mathbf{c})$
00001	00001	10001	10001
00011	00011	10011	10011
00101	00100	10101	10100
00111	00110	10111	10110
01001	01001	11001	11001
01011	01011	11011	11011
01101	01100	11101	11100
01111	01110	11111	11110

.

Each word in $\mathcal{C}$ has its third coordinate different from its last coordinate, and no two words in $\mathcal{C}$ share the same profile vector.

We summarize our observations in the following proposition.

Proposition 3.7. Let $n < 2\ell$. Consider the maps $\texttt{encode}_1$ and $\texttt{decode}_1$ defined by Algorithms 1 and 2 and the code $\mathcal{C}$. Then

$\mathbf{p}(\mathbf{x},\ell) \neq \mathbf{p}(\mathbf{x}',\ell)$ for any two distinct words $\mathbf{x},\mathbf{x}' \in \mathcal{C}$ and $\texttt{decode}_1 \circ \texttt{encode}_1(\mathbf{c}) = \mathbf{c}$ for all $\mathbf{c} \in [\![q]\!]^{n-1} \times \{1,2,\ldots,q-1\}$. Hence, $\texttt{encode}_1$ is injective and $|\mathcal{C}| = q^{n-1}(q-1)$. Furthermore, $\texttt{decode}_1$ and $\texttt{encode}_1$ output their respective strings in $O(n)$ time.

Therefore, for $n < 2\ell$, we have $P_q(n,\ell) \geq q^{n-1}(q-1)$. Now, observe that Algorithm 1 encodes $(n-1)\log_2 q + \log_2(q-1)$ bits of information, while the set of all q-ary words of length n has the capacity to encode $n\log_2 q$ bits of information. Hence, by imposing the constraint that the words have distinct profile vectors, we lose only $\log_2 q - \log_2(q-1) \leq 1$ bit of information. Furthermore, any set of M messages with $M \leq q^{n-1}(q-1)$ can be efficiently converted to data strings in $[\![q]\!]^{n-1} \times \{1,2,\ldots,q-1\}$. Consider the message $i \in [1,M]$. We *reverse* the q-ary base expansion of i to obtain a q-ary sequence $\mathbf{c}$. Since $M \leq q^{n-1}(q-1)$, the last symbol of $\mathbf{c}$ (or the first symbol in the base expansion) is necessarily in $[\![q-1]\!]$. We simply add one to the last symbol of $\mathbf{c}$ to obtain a data string in $[\![q]\!]^{n-1} \times \{1,2,\ldots,q-1\}$.

To end this section, we remark that this family of profile vectors is *not* robust against coverage errors (see Section 4-C for the definition). Consider two words 00001 and 00011. Suppose the 3-gram 000 is lost from the former. Then the remaining reads $000, 001$ can come from either 00001 or 00011 and we are unable to distinguish the two. In the next section, we provide families of profile vectors whose read lengths are shorter than half the word length and are robust against such errors.

4. Distinct Profile Vectors from Addressable Codes

Borrowing ideas from synchronization, we construct a set of words with different profile vectors and prove (2) and (3). Our strategy is to mimic the concept of *watermark* and *marker codes* [18]–[20], where a marker pattern is distributed throughout a codeword. Due to the unordered nature of the short reads, instead of a single marker pattern, we consider a set of patterns.

More formally, suppose that $0 < 2a \leq \ell \leq n$. Let $\mathcal{A} = \{\mathbf{u}_1,\mathbf{u}_2,\ldots,\mathbf{u}_M\} \subseteq [\![q]\!]^a$ be a set of M sequences of length a. Elements of $\mathcal{A}$ are called *addresses*. A word $\mathbf{x} = \mathbf{z}_1\mathbf{z}_2\cdots\mathbf{z}_M$, where $|\mathbf{z}_i| = \ell$ for all $1 \leq i \leq M$, is said to be $(\mathcal{A},\ell)$-*addressable* if the following properties hold.

(C1) The prefix of length a of $\mathbf{z}_i$ is equal to $\mathbf{u}_i$ for all $1 \leq i \leq M$. In other words, $\mathbf{z}_i[1,a] = \mathbf{u}_i$.

(C2) $\mathbf{z}_i[j,j+a-1] \notin \mathcal{A}$ for all $1 \leq i \leq M$ and $2 \leq j \leq \ell - a + 1$.

Conditions (C1) and (C2) imply that the address $\mathbf{u}_i \in \mathcal{A}$ appears exactly once as the prefix of $\mathbf{z}_i$ and does not appear as an a-gram of any substring $\mathbf{z}_j$ with $j \neq i$. A code $\mathcal{C}$ is $(\mathcal{A},\ell)$-*addressable* if all words in $\mathcal{C}$ are $(\mathcal{A},\ell)$-addressable. Intuitively, given an $(\mathcal{A},\ell)$-addressable word $\mathbf{x}$, we can make use of the addresses in $\mathcal{A}$ to identify the position of each ℓ-gram in $\mathbf{x}$ and hence, reconstruct $\mathbf{x}$. We formalize this idea in the following theorem.

Theorem 4.1. Let $2a \leq \ell$ and $\mathcal{A} = \{\mathbf{u}_1,\mathbf{u}_2,\ldots,\mathbf{u}_M\}$ be a set of addresses of length a. Suppose that $\mathcal{C}$ is an $(\mathcal{A},\ell)$-addressable code. For distinct words $\mathbf{x},\mathbf{x}' \in \mathcal{C}$, we have $\mathcal{F}(\mathbf{x},\ell) \neq \mathcal{F}(\mathbf{x}',\ell)$. Therefore, $\mathbf{p}(\mathbf{x},\ell) \neq \mathbf{p}(\mathbf{x}',\ell)$ and $P_q(n,\ell) \geq |\mathcal{C}|$.

Proof. Let $\mathbf{x} = \mathbf{z}_1\mathbf{z}_2\cdots\mathbf{z}_M$ and $\mathbf{x}' = \mathbf{z}'_1\mathbf{z}'_2\cdots\mathbf{z}'_M$ be distinct $(\mathcal{A},\ell)$-addressable words in $\mathcal{C}$. Without loss of generality, we assume $\mathbf{z}_1 \neq \mathbf{z}'_1$. Observe that $\mathbf{z}_1 \in \mathcal{F}(\mathbf{x},\ell)$. To prove the theorem, it suffices to show that $\mathbf{z}_1 \notin \mathcal{F}(\mathbf{x}',\ell)$.

Suppose otherwise that $\mathbf{z}_1$ appears as an ℓ-gram in $\mathbf{x}'$. By Conditions (C1) and (C2), we have that the prefix $\mathbf{u}_1$ of $\mathbf{z}_1$, which is an address that can only appear on the intersection of $\mathbf{z}'_{i-1}$ and $\mathbf{z}'_i$ for some $i \neq 1$. Since otherwise if $\mathbf{u}_1$ is entirely contained in either $\mathbf{z}'_{i-1}$ or $\mathbf{z}'_i$, then it will contradict condition (C2). Hence, we consider the picture below.

$$\mathbf{x}' = \cdots \overbrace{\circ\circ\underbrace{\oplus\oplus\cdots\oplus}_{|\mathbf{u}_i|=a}\oplus\oplus}^{|\mathbf{z}_1|=\ell} + + \cdots \text{ for some } i \neq 1.$$

Here, $\circ$'s and $+$'s represent the ℓ-grams $\mathbf{z}_1$ and $\mathbf{z}'_i$, respectively, and $\oplus$'s indicate the symbols that are in the overlap of the two ℓ-grams. Since $2a \leq \ell$, $\mathbf{u}_i$ must be in $\mathbf{z}_1$ as an a-gram, contradicting Condition (C2). □

To employ Theorem 4.1, we consider two sets of addresses in the following subsections. As observed in Fig. 1(a), the first set of addresses (corresponding to (2)) yields a larger set of distinct profile vectors when the word length is big. However, the number of distinct profile vectors necessarily have rates smaller than $\log_q(q-1)$, and hence, we propose the second set of addresses to obtain larger rates for certain word lengths.

A. First Set of Addresses

Consider the set of addresses

$$\mathcal{A}^* \triangleq \left\{(u_1,u_2,\ldots,u_a) : \sum_{i=1}^{a} u_i = 0 \bmod q\right\}. \tag{8}$$

So, $\mathcal{A}^*$ is a set of $M = q^{a-1}$ addresses listed as $\mathbf{u}_1,\mathbf{u}_2,\ldots,\mathbf{u}_M$. To construct an $(\mathcal{A}^*,\ell)$-addressable code, we consider the map $\texttt{encode}_2 : \{1,2,\ldots,q-1\}^{(\ell-a)M} \to [\![q]\!]^{M\ell}$ given in Algorithm 3 and define $\mathcal{C}$ to be the image of $\texttt{encode}_2$. Conversely, we consider the map $\texttt{decode}_2 : \mathcal{C} \to \{1,2,\ldots,q-1\}^{(\ell-a)M}$ given in Algorithm 4.

Example 4.1. For $q = 4$ and $a = 2$, the address set is $\mathcal{A}^* = \{00,13,22,31\}$ by (8). Consider $\ell = 5$ and the data string $\mathbf{c} = 111\ 123\ 222\ 321$. Applying Algorithm 3 to construct $\mathbf{z}_1$ with $\mathbf{c}_1 = 111$, we start with $\mathbf{z}_1 = 00$. Then $z_{\text{bad}} = 0$ and we choose the first element of $\{1,2,3\}$ to append to $\mathbf{z}_1$ to get 001. In the next iteration, we have $z_{\text{bad}} = 3$ and append 0 to $\mathbf{z}_1$ to get 0010. Repeating this, we then obtain $\mathbf{z}_1 = \underline{00}101$. Completing the process for all i, we have

$$\mathbf{z}_1 = \underline{00}101,\ \mathbf{z}_2 = \underline{13}023,\ \mathbf{z}_3 = \underline{22}111,\ \mathbf{z}_4 = \underline{31}210,$$

and so, $\texttt{encode}_2(\mathbf{c}) = (00101,13023,22111,31210) = \mathbf{x}$. We check that $\mathbf{x}$ is indeed $(\mathcal{A}^*,\ell)$-addressable, and verify that

Algorithm 3 $\texttt{encode}_2(\mathbf{c}, \mathcal{A}^*)$

Input: Data string $\mathbf{c} = \mathbf{c}_1\mathbf{c}_2\cdots\mathbf{c}_M$, where $\mathbf{c}_i \in \{1, 2, \ldots, q-1\}^{(\ell-a)}$ for $1 \le i \le M$, and $\mathcal{A}^*$ is defined by (8).
Output: $\mathbf{x} = \mathbf{z}_1\mathbf{z}_2\cdots\mathbf{z}_M \in [\![q]\!]^{M\ell}$, where $\mathbf{x}$ is $(\mathcal{A}^*, \ell)$-addressable.

for $1 \le i \le M$ **do**
  $\mathbf{z}_i \leftarrow \mathbf{u}_i$ ($\mathbf{u}_i$ has length a)
  for $a+1 \le j \le \ell$ **do**
    $z_{\text{bad}} \leftarrow -\sum_{s=1}^{a-1} \mathbf{z}_i[j-s] \bmod q$ (negative of the sum of the last $a-1$ entries modulo q)
    $z \leftarrow z_{\text{bad}} + \mathbf{c}_i[j-a] \bmod q$
    append $\mathbf{z}_i$ with z
  end for
end for
return $\mathbf{z}_1\mathbf{z}_2\cdots\mathbf{z}_M$

Algorithm 4 $\texttt{decode}_2(\mathbf{z}_1\mathbf{z}_2\cdots\mathbf{z}_M)$

Input: Codeword $\mathbf{z}_1\mathbf{z}_2\cdots\mathbf{z}_M \in \mathcal{C}$.
Output: $\mathbf{c}_1\mathbf{c}_2\cdots\mathbf{c}_M \in \{1, 2, \ldots, q-1\}^{(\ell-a)M}$.

for $1 \le i \le M$ **do**
  for $a+1 \le j \le \ell$ **do**
    $z_{\text{bad}} \leftarrow -\sum_{s=1}^{a-1} \mathbf{z}_i[j-s] \bmod q$ (negative of the sum of the last $a-1$ entries modulo q)
    $\mathbf{c}_i[j-a] \leftarrow \mathbf{z}_i[j] - z_{\text{bad}} \bmod q$
  end for
end for
return $\mathbf{c}_1\mathbf{c}_2\cdots\mathbf{c}_M$

$\texttt{decode}_2(\mathbf{x}) = \mathbf{c}$. Since there are 3^{12} possible data strings, $|\mathcal{C}| = 3^{12} \approx 4^{9.51}$.

Algorithm 3 bears similarities with a *linear feedback shift register* [21]. The main difference is that we augment our codeword with a symbol that is *not equal* to the value defined by the linear equation. This then guarantees that we have no a-grams belonging to $\mathcal{A}^*$. More formally, we have the following proposition.

Proposition 4.2. Consider the maps $\texttt{encode}_2$ and $\texttt{decode}_2$ in Algorithms 3 and 4 and the code $\mathcal{C}$. Then $\mathcal{C}$ is an $(\mathcal{A}^*, \ell)$-addressable code and $\texttt{decode}_2 \circ \texttt{encode}_2(\mathbf{c}) = \mathbf{c}$ for all $\mathbf{c} \in \{1, 2, \ldots, q-1\}^{(\ell-a)M}$. Hence, $\texttt{encode}_2$ is injective and $|\mathcal{C}| = (q-1)^{M(\ell-a)}$. Furthermore, $\texttt{decode}_2$ and $\texttt{encode}_2$ compute their respective strings in $O(qM\ell)$ time.

Since $M = q^{a-1}$, Theorem 4.1 and Proposition 3.7 then imply that $P_q(n, \ell) \ge (q-1)^{q^{a-1}(\ell-a)}$ for $n = q^{a-1}\ell$ and $2a \le \ell$. In other words, for $n = q^{a-1}\ell$ and $2a \le \ell$, we have

$$R_q(n, \ell) \ge \left(1 - \frac{a}{\ell}\right) \log_q(q-1).$$

We now modify our construction to derive addressable codes for all values of $n = m\ell$ where $m \le q^{a-1}$. Choose a subset $\mathcal{B}^*$ of $\mathcal{A}^*$ of size m for the address set and a straightforward modification of Algorithm 3 then yields $(\mathcal{B}^*, \ell)$-addressable words. The size of this $(\mathcal{B}^*, \ell)$-addressable code can be computed to be $(q-1)^{m(\ell-a)}$ and we obtain the following corollary which is a restatement of Theorem 2.2(ii).

Corollary 4.3. Suppose that $n = m\ell$ with $m \le q^{a-1}$ and $2a \le \ell$. Then $P_q(n, \ell) \ge (q-1)^{m(\ell-a)}$ and

$$R_q(n, \ell) \ge (1 - a/\ell) \log_q(q-1).$$

Example 4.2. Setting $q = 4$, $a = 5$, and $\ell = 100$ in (2) yields $P_4(25600, 100) \ge 3^{24320} \approx 4^{19273}$. In other words, $R_4(25600, 100) \ge 0.753$. Applying Corollary 4.3 and varying $a \in \{2, 3, 4, 5\}$, we have

$$R_4(100m, 100) \ge \begin{cases} 0.776, & \text{for } 1 < m \le 4, \\ 0.768, & \text{for } 4 < m \le 16, \\ 0.760, & \text{for } 16 < m \le 64, \\ 0.752, & \text{for } 64 < m \le 256. \end{cases}$$

Observe that the rates obtained via this construction have values less than $\log_4 3 \approx 0.792$. In the next subsection, we propose another set of addresses to obtain rates close to one.

B. Second Set of Addresses

To define the second set of addresses, let $b+1 < a$ and let $\mathcal{R}_q(n, b)$ be the set of all q-ary strings of length n that have no run of b consecutive zeroes. In the literature, $\mathcal{R}_q(n, b)$ is said to satisfy the $(0, b-1)$-*runlength limited* (RLL) constraint [22], [23]. Codes satisfying the constraint have been extensively studied. In particular, the size of $\mathcal{R}_q(n, b)$ and the recursive formula for $F_q(b, n) \triangleq |\mathcal{R}_q(n, b)|$ is well known [23].

$$F_q(n, b) = \begin{cases} q^n, & \text{if } n < b, \\ (q-1)\sum_{i=1}^{b} F_q(n-i, b), & \text{otherwise.} \end{cases} \tag{9}$$

For all values of q, n and b, polynomial time encoding and decoding algorithms for $\mathcal{R}_q(n, b)$ may be obtained using techniques of enumerative coding [23, Chapter 6]. Recently, when $n \le 2^{b-2}$, Levy and Yaakobi [24, Algorithm 1] provided linear time encoding and decoding algorithms for RLL codes using a single bit of redundancy. A straightforward modification of their algorithms yield efficient encoding and decoding algorithms for general $q \ge 2$. In particular, for $n \le q^{b-2}$, there exists linear time encoding and decoding schemes for $\mathcal{R}_q(n, b)$.

Equipped with results regarding RLL codes, we describe the next set of addresses.

$$\mathcal{A}^R(b) \triangleq \{(\underbrace{0, 0, \ldots, 0}_{b \text{ times}}, x)\mathbf{a} : x \ne 0 \text{ and } \mathbf{a} \in \mathcal{R}_q(a-b-1, b)\}. \tag{10}$$

In other words, $\mathcal{A}^R(b)$ is the set of words of length a, where the prefix of each word is a run of b zeroes followed by a nonzero symbol, and no run of b zeroes appear elsewhere in these addresses. Therefore, the size of $\mathcal{A}^R(b)$ is given by $M = (q-1)F_q(a-b-1, b)$ and again, we list the addresses as $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_M$.

Proposition 4.4. Define $\mathcal{A}^R(b)$ as in (10) and the code $\mathcal{C} \triangleq \{\mathbf{z}_1 \cdots \mathbf{z}_M : \mathbf{z}_i = \mathbf{u}_i x \mathbf{v}_i$, where $x \neq 0$ and $\mathbf{v}_i \in \mathcal{R}_q(\ell - a - 1, b)\}$. Then $\mathcal{C}$ is an $(\mathcal{A}^R(b), \ell)$-addressable code of size $((q-1)F_q(\ell - a - 1, b))^M$ and has polynomial time encoding and decoding algorithms. Furthermore, if $\ell - a - 1 \leq q^{b-2}$, there exists encoding and decoding algorithms that compute their respective strings in $O(M\ell)$ time.

Proof. We check that any word in $\mathcal{C}$ satisfies both (C1) and (C2). By definition, any word satisfies condition (C1). For condition (C2), consider the ℓ-gram $\mathbf{z}_i = \mathbf{u}_i x \mathbf{v}_i$ with $x \neq 0$ and $\mathbf{v}_i \in \mathcal{R}_q(\ell - a - 1, b)$ and we have the following cases.

(i) Since $\mathbf{v}_i$ contains no run of b consecutive zeroes, no address in $\mathcal{A}^R(b)$ can begin in $\mathbf{v}_i$. In other words, $\mathbf{z}_i[j, j+a-1] \notin \mathcal{A}^R(b)$ for all $j \geq a+2$.

(ii) Similarly, since $\mathbf{u}_i$ contains no run of b consecutive zeroes, except as its prefix, we have that $\mathbf{z}_i[j, j+a-1] \notin \mathcal{A}^R(b)$ for all $2 \leq j \leq a-b+1$.

(iii) Finally, it remains to verify that $\mathbf{z}_i[j, j+a-1] \notin \mathcal{A}^R(b)$ for all $a-b+2 \leq j \leq a+1$. In this case, the nonzero element x is contained in the first b symbols of $\mathbf{z}_i[j, j+a-1]$, and therefore, the latter a-gram does not belong to $\mathcal{A}^R(b)$.

Hence, an $(\mathcal{A}^R(b), \ell)$-addressable code and its size follow from (9). The encoding and decoding algorithms follow directly from Immink [23] and Levy and Yaakobi [24]. □

As before, we may choose a subset of $\mathcal{A}^R(b)$ of size m with $m \leq (q-1)F_q(a-b-1, b)$ and select it to be the address set. We then obtain the following corollary which is a restatement of Theorem 2.2(iii).

Corollary 4.5. Suppose that $2a \leq \ell$, $b+1 < a$ and $m \leq F_q(a-b-1, b)$. If $n = m\ell$, then

$$P_q(n, \ell) \geq \left(F_q(n-a-1, b)(q-1)\right)^m. \tag{11}$$

We now compare the two lower bounds provided by Corollaries 4.3 and 4.5.

Example 4.3. We revisit Example 4.2 and set $q = 4$ and $\ell = 100$. We fix $b = 3$. Then we let $a = 8$ and check that $m = 256 \leq F_4(4, 3) = 747$. Applying (3) yields $P_4(25600, 100) \geq 4^{91 \times 256} \approx 4^{23296}$. In other words, $R_4(25600, 100) \geq 0.910$, which is significantly larger than the lower bound in Example 4.2. Applying Corollary 4.5 and varying $a \in \{4, 5, 6, 7, 8\}$,

$$R_4(100m, 100) \geq \begin{cases} 0.950, & \text{for } 1 < m \leq 3; \\ 0.940, & \text{for } 3 < m \leq 12; \\ 0.930, & \text{for } 12 < m \leq 48; \\ 0.920, & \text{for } 48 < m \leq 189; \\ 0.910, & \text{for } 189 < m \leq 747. \end{cases}$$

Example 4.4. Set $q = 4$ and $\ell = 20$. We compare the values of n for which Corollaries 4.3 and 4.5 apply.

We first consider Corollary 4.5. Varying a and b such that $a \geq b+1$ and $2a \leq \ell$, the maximum possible value of m is $F_4(7, 2) = 35316$ when $a = 10$ and $b = 2$. Hence, the largest possible value n is 706320.

On the other hand, for Corollary 4.3, the maximum possible value of m is $4^9 = 262144$ when $a = 10$. Hence, the largest possible value of n is 5242880. Therefore, we observe that Corollary 4.3 provides lower bounds $P_q(n, \ell)$ for a wider range of n.

We improve the lower bound for $R_4(n, 100)$ in the next section (Section 5). Nevertheless, the families of profile vectors obtained in this section have efficient encoding and decoding algorithms. Furthermore, we demonstrate a simple assembly algorithm in the next subsection.

C. Assembly of $(\mathcal{A}, \ell)$-Addressable Words in the Presence of Coverage Errors

Let $\mathcal{A}$ be a set of addresses, $\mathcal{C}$ be a set of $(\mathcal{A}, \ell)$-addressable words, and $\mathbf{x} \in \mathcal{C}$. This subsection presents an algorithm that takes the set of reads provided by $\mathbf{p}(\mathbf{x}, \ell)$ and correctly assembles $\mathbf{x}$. We also observe that correct assembly is possible even if some reads are lost.

We use the formal definition of the DNA storage channel given by Kiah *et al.* [12] and reproduce here the notion of *coverage errors*. Suppose that the data of interest is encoded by a vector $\mathbf{x} \in [\![q]\!]^n$ and let $\widehat{\mathbf{p}}(\mathbf{x})$ be the output profile vector of the channel. Coverage errors occur when not all ℓ-grams are observed during fragmentation and subsequently sequenced. For example, suppose that $\mathbf{x} = 10001$ from Example 2.1, and that $\widehat{\mathbf{p}}(\mathbf{x})$ is the channel output 2-gram profile vector. The coverage loss of one 2-gram results in the count of 00 in $\widehat{\mathbf{p}}(\mathbf{x})$ to be one instead of two.

Let $\mathcal{B}(\mathbf{x}, e)$ be the set of all possible output profile vectors arising from at most e coverage errors with input vector $\mathbf{x}$. Then for a code $\mathcal{C} \subseteq [\![q]\!]^n$, a map $\Phi : \mathbb{Z}_{\geq 0}^{q^\ell} \to \mathcal{C} \cup \{\text{Fail}\}$ is an *assembly algorithm for* $\mathcal{C}$ *that corrects* e *coverage errors* if for all $\mathbf{x} \in \mathcal{C}$, $\Phi(\widehat{\mathbf{p}}) = \mathbf{x}$ for all $\widehat{\mathbf{p}} \in \mathcal{B}(\mathbf{x}, e)$.

Let $\widehat{\mathbf{p}}$ represent a (possibly incomplete) set of reads obtained from $\mathbf{x}$. Our broad strategy of assembly is as follows. For each read $\mathbf{r} \in \widehat{\mathbf{p}}$, we first attempt to *align* $\mathbf{r}$ by guessing the index j such that $\mathbf{r} = \mathbf{x}[j, j+\ell-1]$. After which, we ensure that all symbols in $\mathbf{x}$ are *covered* by some correctly aligned read so that the entire word $\mathbf{x}$ can be reconstructed.

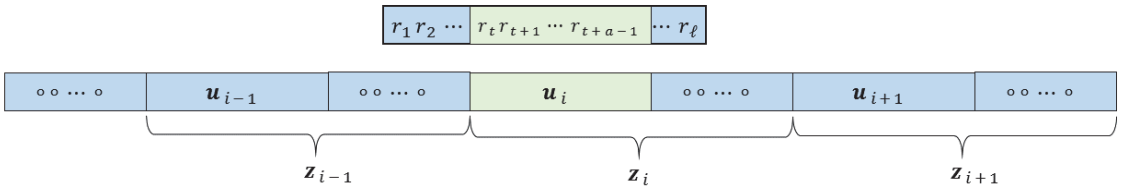
We now describe in detail the alignment step. Let $\mathbf{x}$ be an $(\mathcal{A}, \ell)$-addressable word. Recall that $\mathbf{x}$ can be written as

$$\mathbf{x} = \underbrace{\overbrace{x_1 \cdots x_a}^{\mathbf{u}_1} \cdots x_\ell}_{\mathbf{z}_1} \cdots \underbrace{\overbrace{x_{(i-1)\ell+1} \cdots x_{(i-1)\ell+a}}^{\mathbf{u}_i} \cdots x_{i\ell}}_{\mathbf{z}_i} \cdots \underbrace{\overbrace{x_{(M-1)\ell+1} \cdots x_{(M-1)\ell+a}}^{\mathbf{u}_M} \cdots x_{M\ell}}_{\mathbf{z}_M},$$

so that $\mathbf{x}[(i-1)\ell+1, (i-1)\ell+a] = \mathbf{u}_i$ for all $1 \leq i \leq M$.

Let $\mathbf{r}$ be a read obtained from $\mathbf{x}$ and we say that $\mathbf{r}$ is *correctly aligned at* j for some $1 \leq j \leq n-\ell+1$ if $\mathbf{x}[j, j+\ell-1] = \mathbf{r}$. To align the read $\mathbf{r}$, our plan is to look for the address that occurs last in $\mathbf{r}$, say $\mathbf{u}_i$, and match it to the corresponding index $(i-1)\ell+1$. Specifically, suppose that $\mathbf{r}$ is a read that contains an address as a substring. Let t be the largest index such that $\mathbf{r}[t, t+a-1] = \mathbf{u}_i \in \mathcal{A}$. Then we align $\mathbf{r}$ in a way such that $\mathbf{r}[t, t+a-1]$ matches the address $\mathbf{u}_i$ (see Figure 3(a)).

(a)



(b)

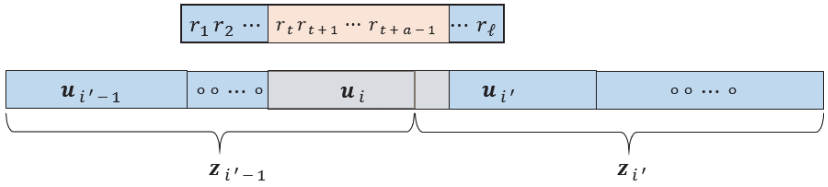


Fig. 3. Possible ways of obtaining a read containing the address $\mathbf{u}_i$: (a) The string $\mathbf{u}_i$ is the prefix of the ith component $\mathbf{z}_i$ of $\mathbf{x}$. (b) The string $\mathbf{u}_i$ has an overlap with the prefix of the i'th component $\mathbf{z}_{i'}$ of $\mathbf{x}$.

However, not all reads can be aligned correctly. To remedy the situation, we define a special type of read that can be proven to be always aligned correctly.

Definition 4.1. Let $\mathbf{r} \in [\![q]\!]^\ell$ be an ℓ-gram or read. Define $L(\mathbf{r})$ to be the the largest index t of $\mathbf{r}$ such that $\mathbf{r}[t, t+a-1] \in \mathcal{A}$. If such a t does not exist, then we set $L(\mathbf{r}) = \infty$. We say that $\mathbf{r}$ is a *Type I read* if $L(\mathbf{r}) \leq \ell - 2a + 2$.

Example 4.5. In Definition 4.1, observe that we can characterize a Type I read without knowing $\mathbf{x}$ as illustrated in the examples below.

(i) Consider the set of parameters in Example 4.1 with $q = 4$, $a = 2$, $\mathcal{A}^* = \{00, 13, 22, 31\}$, and $\ell = 5$. Consider further the $(\mathcal{A}^*, \ell)$-addressable word $\mathbf{x} = (00101, 13023, 22111, 31210)$. The 5-gram 01130 is a Type I read, since $L(01\underline{13}0) = 3 \leq 3$ On the other hand, the 5-grams 30232 and 21113 are *not* Type I reads, since $L(30232) = \infty$ and $L(211\underline{13}) = 4 > 3$.

(ii) Consider $q = 4$, $a = 3$, $\ell = 8$, and $\mathcal{A}^*$ given by (8). The 8-gram 101$\underline{220}$01 is a Type I read, while the 8-grams 32122133 and 2130$\underline{130}$3 are *not* Type I reads. Note that $L(21301303) = 5$, since 130 is the last 3-gram even though 301 and 013 also occur as 3-grams.

Now, our alignment method may fail when a read is *not* a Type I read. For example, if we consider $\mathbf{x}$ in Example 4.5(i) and the read $\mathbf{r} = 211\underline{13}$, our method aligns $\mathbf{r}$ to index 3, which is incorrect. In contrast, if a read is of Type I, we prove that the alignment is always correct.

Lemma 4.6. Let $\mathbf{r}$ be a Type I read of an $(\mathcal{A}, \ell)$-addressable word $\mathbf{x}$. If $L(\mathbf{r}) = t$ and $\mathbf{r}[t, t+a-1] = \mathbf{u}_i$, then $\mathbf{r}$ is correctly aligned at $j = (i-1)\ell - t + 2$.

Proof. It suffices to show that if $\mathbf{x}[j', j'+\ell-1] = \mathbf{r}$ for some index j', then j' is necessarily equal to j.

Now, $\mathbf{r}[t, t+a-1] = \mathbf{u}_i$, or equivalently, $\mathbf{x}[j'+t-1, j'+t+a-2] = \mathbf{u}_i$. From the definition of an $(\mathcal{A}, \ell)$-addressable word, $\mathbf{u}_i$ cannot be properly contained in any of the substrings $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_M$, with the exception of $\mathbf{z}_i$, where $\mathbf{u}_i$ is the prefix. Therefore, we only have the following two possibilities for the index $j' + t - 1$ (see Figure 3).

(a) $\mathbf{u}_i$ is a prefix of $\mathbf{z}_i$. Then $j'+t-1 = (i-1)\ell+1$, implying that $j' = (i-1)\ell - t + 2$.

(b) The substring $\mathbf{r}[t, t+a-1] = \mathbf{x}[j'+t-1, j'+t+a-2] = \mathbf{u}_i$ has an overlap with some address $\mathbf{u}_{i'} \in \mathcal{A}$. Specifically, $j'+t-1 \pmod{\ell} \in \{0, \ell-a+2, \ell-a+3, \ldots, \ell-1\}$. Since $\mathbf{r}$ is a Type I read, we have that $t \leq \ell - 2a + 2$. Hence, there are at least $a-1$ symbols in $\mathbf{r}$ after the substring $\mathbf{u}_i$. Since $\mathbf{u}_i$ has some overlap with $\mathbf{u}_{i'}$, the substring $\mathbf{u}_{i'}$ is necessarily contained in $\mathbf{r}$. This contradicts the maximality of t.

In conclusion, the only possibility is $j' = j$. □

It remains to show that all symbols in $\mathbf{x}$ is covered by a Type I read. More formally, let $1 \leq i \leq n$ and we say that a read $\mathbf{r} = \mathbf{x}[j, j+\ell-1]$ *covers the symbol* $\mathbf{x}[i]$ if $j \leq i \leq j+\ell-1$. The following lemma characterizes Type I reads.

Lemma 4.7. Let $1 \leq j \leq n-\ell+1$. The ℓ-gram $\mathbf{x}[j, j+\ell-1]$ is a Type I read if and only if $j \pmod{\ell} \notin \{2, 3, \ldots, 2a-1\}$.

Proof. Let $\mathbf{r} := \mathbf{x}[j, j+\ell-1]$. Let $j = k + \ell \cdot m$ for some nonnegative integer m. If $k = 1$, then $L(\mathbf{r}) = 1$. When $2a \leq k \leq \ell$, notice that $\mathbf{r}$ contains the address $\mathbf{u}_{m+1}$. Hence $L(\mathbf{r}) \neq \infty$. Since the last possible address is $\mathbf{u}_{m+1} = \mathbf{x}[1 + (m+1)\ell, a + (m+1)\ell]$, we have $L(\mathbf{r}) = (1 + (m+1)\ell) - (k + \ell \cdot m) + 1 = \ell - k + 2$. Thus, $2 \leq L(\mathbf{r}) \leq \ell - 2a + 2$. In both cases, $\mathbf{r}$ is a Type I read.

For the converse, let $j = k + \ell \cdot m$ for some nonnegative integer m. Consider the case where $2 \leq k \leq a$. If $L(\mathbf{r}) = \infty$, then it is not Type I. Otherwise, there exists a t such that $\mathbf{x}[t, t+a-1] \in \mathcal{A}$ with $t \geq (m+1)\ell - a + 2$. Therefore, $L(\mathbf{r}) \geq \ell - a - k + 3 \geq \ell - 2a + 3$. Next, let $a + 1 \leq k \leq 2a - 1$. The last possible address is $\mathbf{u}_{m+1} = \mathbf{x}[1 + (m+1)\ell, a + (m+1)\ell]$. Hence, $L(\mathbf{r}) = (1 + (m+1)\ell) - (k + \ell \cdot m) + 1 = \ell - k + 2 \geq \ell - 2a + 3$. In both cases, $\mathbf{r}$ is not a Type I read. □

The next corollary then follows from a straightforward computation.

Corollary 4.8. Let $\mathbf{x}$ be a q-ary word of length n. Each symbol in $\mathbf{x}$ is covered by at least one Type I read. In particular, if $\ell \leq i \leq n - \ell$, then $\mathbf{x}[i]$ is covered by exactly $\ell - 2a + 2$ Type I reads.

Lemma 4.6 and Corollary 4.8 imply the correctness of our assembly algorithm for $\mathcal{C}$, provided that no reads are lost. In order for the assembly algorithm to correct coverage errors, we simply fix the values of the first ℓ and the last ℓ symbols of all codewords. Then each of the other $n - 2\ell$ symbols is covered by exactly $\ell - 2a + 2$ Type I reads. Therefore, if at most $\ell - 2a + 1$ reads are lost, we are guaranteed that all of these $n - 2\ell$ symbols are covered by at least one Type I read. We obtain the following theorem.

Theorem 4.9. Let $\mathcal{A} = \{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_M\}$ be a set of addresses. Suppose that $\mathbf{z}_1^*, \mathbf{z}_M^* \in [\![q]\!]^\ell$ obey the following properties:

(1) $\mathbf{z}_1^*[1, a] = \mathbf{u}_1$ and $\mathbf{z}_M^*[1, a] = \mathbf{u}_M$;
(2) $\mathbf{z}_1^*[j, j + a - 1]^* \notin \mathcal{A}$ and $\mathbf{z}_M^*[j, j + a - 1] \notin \mathcal{A}$ for all $2 \leq j \leq \ell - a + 1$.

Let $\mathcal{C}^*$ be a set of $(\mathcal{A}, \ell)$-addressable words such that $\mathbf{x}[1, \ell] = \mathbf{z}_1^*$ and $\mathbf{x}[(M-1)\ell + 1, M\ell] = \mathbf{z}_M^*$ for all $\mathbf{x} \in \mathcal{C}^*$. Then there exists an assembly algorithm for $\mathcal{C}^*$ that corrects $\ell - 2a + 1$ coverage errors.

Suitable modifications to the code $\mathcal{C}^*$ in Theorem 4.9 yield addressable codes that correct more than $\ell - 2a + 1$ coverage errors. We sketch only the main ideas as they follow from the well-known concept of code concatenation treated in [25] and [26, Section 5.5]. More concretely, we assume two codes: an *outer* code $\mathcal{D}_{\text{out}}$ of length $M - 2$ over an alphabet Σ, and an *inner* code $\mathcal{D}_{\text{in}}$ of length $\ell - a$ over the alphabet $[\![q]\!]$. They satisfy the following conditions.

(D1) $|\Sigma| \leq |\mathcal{D}_{\text{in}}|$, and so, there exists an injective map $\chi : \Sigma \to \mathcal{D}_{\text{in}}$.
(D2) Recall the definition of $\mathbf{z}_1^*$ and $\mathbf{z}_M^*$ in Theorem 4.9. Define $\mathcal{D}^*$ to be the q-ary code of length $M\ell$ given by

$$\mathcal{D}^* \triangleq \{\mathbf{z}_1^* \mathbf{z}_2 \cdots \mathbf{z}_{M-1} \mathbf{z}_M^* : \mathbf{z}_j = \mathbf{u}_j \chi(d_j), \\ 2 \leq j \leq M - 1, (d_2, \ldots, d_{M-1}) \in \mathcal{D}_{\text{out}}\}.$$

We require $\mathcal{D}^*$ to be $(\mathcal{A}, \ell)$-addressable.

Therefore, if the outer code $\mathcal{D}_{\text{out}}$ and inner code $\mathcal{D}_{\text{in}}$ are able to correct up to $(D-1)$ and $(d-1)$ erasures, respectively, then the assembly algorithm for $\mathcal{D}^*$ is able to correct $dD(\ell - 2a + 2) - 1$ coverage errors.

Remark 1. We demonstrate that there exist inner codes that satisfy condition (D2). Since (D1) can be satisfied easily by choosing an outer code over a sufficiently large alphabet Σ, the concatenated code $\mathcal{D}^*$ can be made $(\mathcal{A}, \ell)$-addressable.

Consider the address set $\mathcal{A}^R(b)$ defined by (10). For the inner code, consider a code $\mathcal{C}^R \subseteq \mathcal{R}_q(n, b)$ of length $\ell - a - 1$ that corrects up to $(d-1)$ erasures. In other words, $\mathcal{C}^R$ is an erasure-correcting code that satisfies the $(0, b-1)$-RLL constraint. Such codes have been studied extensively in the literature of constrained coding (see Marcus *et al.* [22, Chapter 9] and the references therein).

Next, we set $\mathcal{D}_{\text{in}} = \{x\mathbf{v} : x \neq 0, \mathbf{v} \in \mathcal{C}^R\}$. Hence, for any symbol $d \in \Sigma$, the image $\chi(d)$ is of the form $x\mathbf{v}$ where $x \neq 0$ and $\mathbf{v} \in \mathcal{R}_q(n, b)$. Following a similar argument in the proof of Proposition 4.4, we have that $\mathcal{D}^*$ is $(\mathcal{A}^R(b), \ell)$-addressable.

It is unclear if the above construction is optimal. The question of optimality and the investigation of other possible address sets and their corresponding inner codes are deferred to future work.

5. Distinct Profile Vectors from Partial de Bruijn Sequences

We borrow classical results on de Bruijn sequences and certain results from Maurer [27] to provide detailed proof for (4). Unfortunately, while we derive a strong lower bound for $R_q(n, \ell)$, the proof described here is nonconstructive and we do not obtain any efficient encoding and decoding algorithms as in the previous sections. We first define partial ℓ-de Bruijn sequences.

Definition 5.1. A q-ary word $\mathbf{x}$ is a *partial ℓ-de Bruijn sequence* if every q-ary word of length ℓ appears at most once in $\mathbf{x}$. In other words, $p(\mathbf{x}, \mathbf{z}) \leq 1$. A partial ℓ-de Bruijn sequence is *complete ℓ-de Bruijn* if every q-ary word of length ℓ appears exactly once in $\mathbf{x}$.

By definition, a complete ℓ-de Bruijn sequence has length $q^\ell + \ell - 1$. The number of distinct complete ℓ-de Bruijn sequences was explicitly determined by van Aardenne-Ehrenfest and de Bruijn [28] as a special case of a more general result on the number of trees in certain graphs. They built upon a previous work done by Tutte and Smith (see the note at the end of [28]). Their combined effort led to a formula of counting the number of trees in a directed graph as the value of a certain determinant. The formula is now known as BEST Theorem. The acronym refers to the four surnames, namely de **B**ruijn, **E**hrenfest, **S**mith, and **T**utte.

Theorem 5.1 (BEST [28])**.** The number of distinct complete ℓ-de Bruijn words is $(q!)^{q^{\ell-1}}$.

Remark 2. Theorem 5.1 is usually stated in terms of Eulerian circuits in a de Bruijn graph of order ℓ. We refer the interested reader to van Aardenne-Ehrenfest and de Bruijn [28] for the formal definitions. Specifically, the number of Eulerian circuits is known to be $(q!)^{q^{\ell-1}}/q^\ell$. Consider a circuit represented by the q-ary word $\mathbf{y}$. To obtain q^ℓ complete ℓ-de Bruijn sequences, we simply consider the q^ℓ rotations of $\mathbf{y}$ and append to each rotation $\mathbf{y}'$ its prefix of length $\ell - 1$.

Partial de Bruijn sequences are of deep and sustained interest in graph theory, combinatorics, and cryptography. In the first two, their inherent structures are the focus of attention, while, in cryptography, the interest is mainly on the case of $q = 2$ to generate random looking sequences to be used as keystream in additive stream ciphers [29, Sect. 6.3]. Maurer established the following enumeration result.

Theorem 5.2 ([27, Theorem 3.1])**.** The number of partial ℓ-de Bruijn sequences is larger than $L_q(n) - \binom{n}{2} q^{n-\ell}/n$.

Recall that $L_q(n)$ is the number of q-ary Lyndon words of length n given by (7). Next, we make the connection to profile vectors based on the following observation of Ukkonen.

Lemma 5.3 ([30, Theorem 2.2])**.** Let $\mathbf{x}$ be a q-ary sequence of length n such that every $(\ell-1)$-gram appears at most once. If $\mathbf{x}'$ is a q-ary sequence such that $\mathbf{p}(\mathbf{x}, \ell) = \mathbf{p}(\mathbf{x}', \ell)$, then $\mathbf{x} = \mathbf{x}'$.

It follows from Lemma 5.3 that two distinct partial $(\ell - 1)$-de Bruijn sequences have distinct ℓ-gram profile vectors. Therefore, the number of distinct partial $(\ell - 1)$-de Bruijn sequences is a lower bound for $P_q(n, \ell)$. Hence, we establish (4) and also the following corollary to BEST Theorem.

Corollary 5.4. Let $n = q^\ell + \ell - 1$. Then $P_q(n, \ell) \geq (q!)^{q^{\ell-1}}$. Hence, $\alpha(n, q) \geq \frac{1}{q} \log_q(q!)$.

Finally, we conduct an analysis similar to Maurer's to complete the proof of Theorem 2.2(iv). We restate the theorem here for convenience.

Theorem 5.5. If $n \geq \ell$, then

$$P_q(n, \ell) > \frac{1}{n} \left(\sum_{t|n} \mu\left(\frac{n}{t}\right) q^t - \binom{n}{2} q^{n-\ell+1} \right).$$

Suppose further that $\ell \geq 2 \log_q n + 2$ and $n \geq 8$. Then $P_q(n, \ell) > q^{n-1}/n$.

Proof. As mentioned earlier, since the number of distinct partial $(\ell - 1)$-de Bruijn sequences is a lower bound for $P_q(n, \ell)$, the first inequality follow from Theorem 5.2 and Lemma 5.3.

Next, assume that $\ell \geq 2 \log_q n + 2$ with $n \geq 8$. Then the following two inequalities can be established.

$$\binom{n}{2} q^{n-\ell+1} < \frac{q^{n-1}}{2}, \text{ since } q^{\ell-2} \geq n^2, \text{ and}$$

$$\sum_{\substack{d<n \\ d|n}} \mu\left(\frac{n}{d}\right) q^d < \frac{n}{2} q^{n/2} \leq \frac{1}{2} q^{n-1} \text{ for } q \geq 2 \text{ and } n \geq 8.$$

Therefore,

$$nL_q(n) - \binom{n}{2} q^{n-\ell+1} \geq q^n - \frac{1}{2} q^{n-1} - \frac{1}{2} q^{n-1} \geq q^{n-1}.$$

Thus, $P_q(n, \ell) > q^{n-1}/n$. □

Remark 3. Recently, Gabrys and Milenkovic [31] improved the lower bound in Theorem 5.2 in the context of reconstruction from substring spectrum. Specifically, they showed that when $\ell \geq 2 \lceil \log_q n \rceil + 5$, then $P_q(n, \ell) \geq q^{n-2}$.

6. Conclusion

We provided exact values and lower bounds for the number of profile vectors given moderate values of q, ℓ, and n. Surprisingly, for fixed $q \geq 2$ and moderately large values $n \leq q^{\ell/2-1}$, the number of profile vectors is at least $q^{\kappa n}$ with κ very close to 1. In other words, for practical values of read and word lengths, we are able to obtain a set of distinct profile vectors with rates close to one. In addition to enumeration results, we propose a set of linear-time encoding and decoding algorithms for certain families of profile vectors.

In our future work, we aim to provide sharper estimates on the asymptotic rate of profile vectors $\alpha(n, q)$ (see (6)) when $q^\ell/2 \leq n \leq q^{\ell(1+\epsilon)}$ (for some $\epsilon > 0$). and to examine the number of profile vectors with specific ℓ-gram constraints *a la* Kiah *et al.* [12].

Acknowledgement

The authors thank the anonymous reviewers and Professor Moshe Schwartz for their comments and suggestions to improve the presentation of this work.

References

[1] Z. Chang, J. Chrisnata, M. F. Ezerman, and H. M. Kiah, "On the number of DNA sequence profiles for practical values of read lengths," in *Proc. IEEE International Symp. Inform. Theory*, 2016, pp. 2654–2658.

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

[3] N. Goldman, P. Bertone, S. Chen, C. Dessimoz, E. M. LeProust, B. Sipos, and E. Birney, "Towards practical, high-capacity, low-maintenance information storage in synthesized DNA," *Nature*, vol. 494, pp. 77–80, 2013.

[4] R. N. Grass, R. Heckel, M. Puddu, D. Paunescu, and W. J. Stark, "Robust chemical preservation of digital information on DNA in silica with error-correcting codes," *Angewandte Chemie International Edition*, vol. 54, no. 8, pp. 2552–2555, 2015.

[5] S. Yazdi, Y. Yuan, J. Ma, H. Zhao, and O. Milenkovic, "A rewritable, random-access DNA-based storage system," *Scientific Reports*, vol. 5, no. 14138, 2015.

[6] S. Yazdi, H. M. Kiah, E. R. Garcia, J. Ma, H. Zhao, and O. Milenkovic, "DNA-based storage: Trends and methods," *IEEE Trans. Molecular, Biological, Multi-Scale Commun.*, vol. 1, no. 3, pp. 230–248, 2015.

[7] S. Yazdi, H. M. Kiah, E. R. Garcia, J. Ma, H. Zhao, and O. Milenkovic, "DNA-based storage: Trends and methods," *IEEE Trans. Molecular, Biological, Multi-Scale Commun.*, vol. 1, no. 3, pp. 230–248, 2015.

[8] J. Bornholt, R. Lopez, D. M. Carmean, L. Ceze, G. Seelig, and K. Strauss, "A DNA-based archival storage system", in *Proc. Twenty-First International Conf. Architectural Support for Programming Languages and Operating Systems*, 2016, pp. 637–649.

[9] Y. Erlich, and D. Zielinski, "DNA Fountain enables a robust and efficient storage architecture", *Science*, vol. 355, no. 6328, pp. 950–954, 2017.

[10] S. Yazdi, R. Gabrys, and O. Milenkovic, "Portable and error-free DNA-based data storage", *Scientific Reports*, no. 5011, vol. 7, 2017.

[11] R. Gabrys, H. M. Kiah, and O. Milenkovic, "Asymmetric Lee distance codes for DNA-based storage," *arXiv preprint arXiv:1506.00740*, 2015.

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

[13] P. Jacquet, C. Knessl, and W. Szpankowski, "Counting Markov types, balanced matrices, and Eulerian graphs," *IEEE Trans. Inform. Theory*, vol. 58, no. 7, pp. 4261–4272, 2012.

[14] S. Kosuri. and G. M. Church, "Large-scale de novo DNA synthesis: technologies and applications," *Nature Methods*, no. 11, pp. 499–507, 2014.

[15] R. P. Stanley, *Enumerative Combinatorics*. Cambridge University Press, 2011, vol. 1.

[16] S. Tan and J. Shallit, "Sets represented as the length-n factors of a word," in *Combinatorics on Words*. Springer, 2013, pp. 250–261.

[17] R. C. Lyndon, "On Burnside's problem," *Transactions of the American Mathematical Society*, vol. 77, no. 2, pp. 202–215, 1954.
[18] F. Sellers, "Bit loss and gain correction code," *IRE Trans. Inform. Theory*, vol. 8, no. 1, pp. 35–38, 1962.
[19] N. Kashyap and D. L. Neuhoff, "Codes for data synchronization with timing," in *Proc. Data Compression Conference*. IEEE, 1999, pp. 443–452.
[20] M. C. Davey and D. J. MacKay, "Reliable communication over channels with insertions, deletions, and substitutions," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 687–698, 2001.
[21] S. W. Golomb, *Shift Register Sequences*. Aegean Park Press, 1982.
[22] B. H. Marcus, R. M. Roth, and P. H. Siegel, "An Introduction to Coding for Constrained Systems," unpublished lecture notes.
[23] K. A. S. Immink, *Codes for Mass Data Storage Systems*. Shannon Foundation Publisher, 2004.
[24] M. Levy and E. Yaakobi, "Mutually uncorrelated codes for DNA storage," in *Proc. IEEE International Symp. Inform. Theory*, 2017, pp. 3115–3119.
[25] G. D. Forney Jr, "Concatenated Codes." Research Monograph no. 37, 1966.
[26] W. C. Huffman and V. Pless, *Fundamentals of Error-Correcting Codes*. Cambridge University Press, 2010.
[27] U. M. Maurer, "Asymptotically-tight bounds on the number of cycles in generalized de Bruijn-Good graphs," *Discrete Applied Mathematics*, vol. 37, pp. 421 – 436, 1992.
[28] T. van Aardenne-Ehrenfest and N. G. de Bruijn, "Circuits and trees in oriented linear graphs," *Simon Stevin*, vol. 28, pp. 203–217, 1951.
[29] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot, *Handbook of Applied Cryptography*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 1996.
[30] E. Ukkonen, "Approximate string-matching with q-grams and maximal matches," *Theoretical Computer Science*, vol. 92, no. 1, pp. 191–211, 1992.
[31] R. Gabrys, and O. Milenkovic, "Unique Reconstruction of Coded Strings from their Substring Spectrum", accepted for *DNA23*, 2017.

Zuling Chang received his B.Sc. and Ph.D. degrees, both in mathematics, from Nankai University, China, in 1998 and 2003 respectively. He did his postdoctoral research in Beijing University of Posts and Telecommunications, China, from 2003 to 2005. He then joined the faculty of the School of Mathematics and Statistics, Zhengzhou University, China, where he is currently an Associate Professor.

His research interests cover the design of cryptographic sequences and information theory.

Johan Chrisnata received his B.Sc. degree in mathematics from Nanyang Technological University (NTU), Singapore, in 2015. Since then, he has been a Project Officer at NTU. His research interests include enumerative combinatorics and coding theory.

Martianus Frederic Ezerman grew up in East Java Indonesia. He received the B.A. degree in philosophy and the B.Sc. degree in mathematics in 2005 and the M.Sc. degree in mathematics in 2007, all from Ateneo de Manila University, Philippines. In 2011 he obtained the Ph.D. degree in mathematics from Nanyang Technological University (NTU), Singapore. After research fellowships at Laboratoire d'Information Quantique, Université Libre de Bruxelles, Belgium and at the Centre for Quantum Technologies (CQT), National University of Singapore, he returned in March 2014 to NTU where he is currently a Senior Research Fellow.

He is interested in coding theory, cryptography, and quantum information processing.

Han Mao Kiah received his Ph.D. degree in mathematics from Nanyang Technological University (NTU), Singapore, in 2014. From 2014 to 2015 he was a Postdoctoral Research Associate at the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign. Currently he is a Lecturer at the School of Physical and Mathematical Sciences, NTU, Singapore.

His research interests include combinatorial design theory, coding theory, and enumerative combinatorics.