Average Redundancy of Variable-Length Balancing Schemes à la Knuth

Duc Tu Dao*, Han Mao Kiah*, and Tuan Thanh Nguyen†

*D School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore
†Singapore University of Technology and Design
email: {daoductu001, hmkiah}@ntu.edu.sg, tuanthanhnguyen@sutd.edu.sg

Abstract — We study and propose schemes that map messages onto constant-weight codewords using variable-length prefixes. We provide polynomial-time computable formulas that estimate the average number of redundant bits incurred by our schemes. In addition to the exact formulas, we perform an asymptotic analysis and show that our scheme uses \(\frac{1}{2} \log_2 n + O(1)\) redundant bits to encode into length-\(n\) words with weight \((n/2) + q\) for constant \(q\).

I. INTRODUCTION

The imbalance of a binary word \(x\) refers to the difference\(^1\) between the number of ones and the number of zeros in \(x\). A word of even length is balanced if its imbalance is exactly zero and a code is balanced if all its codewords are balanced. Due to their applications in various recording systems, balanced codes have been extensively studied (see [1] for a survey). In recent years, interest in balanced codes has rekindled because of the emergence of DNA macromolecules as a next-generation data storage medium with its unprecedented density, durability, and replication efficiency [2], [3]. Now, a DNA string comprises four bases: A, T, C, G, and a string is GC-rich if a high proportion of the bases corresponds to either G or C. Since GC-rich or GC-poor DNA strings are prone to both synthesis and sequencing errors [4], [5], we aim to reduce the difference between the number of G, C, and the number of A, T on every DNA codeword. This requirement turns out to be equivalent to reducing the imbalance of a related binary word (see for example [5], [6]).

In his seminal paper [7], Knuth proposed a simple and elegant linear-time algorithm that transforms an arbitrary binary length-\(n\) message into a balanced length-\(n\) codeword. To allow the receiver to recover the message, a \(\log_2 n\)-bit prefix must be transmitted and hence, Knuth’s method incurs a redundancy of \(\log_2 n\) bits. This differs from the minimum required by a multiplicative factor of two (see (1)). Later, Alon et al. [8] demonstrated that, under certain assumptions of the encoding scheme, \(\log_2 n\) redundant bits are necessary (see discussion in Section II for more details). Hence, it appears unlikely that we can improve Knuth’s balancing technique if we insist on transmitting prefixes of a fixed length.

Therefore, in [9], [10], Immink and Weber proposed balancing schemes that transmit variable-length prefixes and studied the average redundancy of their proposals. Specifically, in [9], Weber and Immink provided two variable-length balancing schemes whose average redundancy are asymptotically equal to \(\log_2 n\) and \(\frac{1}{2} \log_2 n + 0.916\), respectively. The work in [9] is extended to \(q\)-ary case in [24]. Later in [10], Immink and Weber proposed another variable-length balancing scheme which we study closely in this paper. In [10], Immink and Weber provided closed formulas for the average redundancy of their scheme and computed these values for \(n \leq 8192\). While numerically the redundancy values are close to the optimal value given in (1), a tight asymptotic analysis was not provided. A slightly different approach was considered by Swart and Weber in [22], where messages have variable length and codewords have fixed length. However, for long messages, its redundancy is \(O(\sqrt{n})\). In this work, we make modifications to the scheme in [10] and demonstrate that the average redundancy is at most \(\frac{1}{2} \log_2 n + 0.526\) asymptotically.

Even though the average redundancy of our scheme differs from the optimal \((1)\) by an additive constant of approximately 0.2, our scheme and its accompanying analysis can be easily extended to the case where the imbalance is fixed to some positive constant. Formally, for an even integer \(n\) and some fixed integer \(q\), we say that a length-\(n\) word is \(q\)-balanced if its imbalance is exactly \(q\), that is, its weight is exactly \((n/2) + q\). A code is \(q\)-balanced if all words are \(q\)-balanced. Since all words in an \(q\)-balanced have the same weight, such codes are also known as constant-weight codes and are used in a variety of communication and data storage scenarios. Recent applications involve data storage in crossbar resistive memory arrays [11], [12] and live DNA [13] (see also [14] for a survey).

While there is extensive research on constructing constant-weight codes with distance properties, simple efficient encoding methods are less well-known. In fact, this problem was posed by MacWilliams and Sloane as Research Problem 17.3 [15]. To the best of our knowledge, there are three encoding approaches: the enumerative method of Schalkwijk [16], the geometric approach of Tian et al. [17], and the Knuth-like method of Skachek and Immink [18]. The Knuth-like method is also viable to construct \(q\)-ary constant weight sequences with redundancy \(\log_q n + O(1)\) [23]. For the case where \(q\) is constant, the first two methods encode in quadratic time \(O(n^2)\), while the third method runs in linear time. However, the third method incurs \(\log_q n\) redundant bits and this is the regime that we study in this work. Specifically, when \(q\) is a positive constant, we show that there is a linear-time variable-length \(q\)-balancing scheme that incurs average redundancy of at most \(\frac{1}{2} \log_2 n + 2.526\) redundant bits.

In summary, our contributions are two-fold. First, we adapt the variable-length balancing schemes in [10] to encode into \(q\)-balanced words for a fixed \(q \geq 0\). Second, and more crucially, we provide a detailed analysis of the average redundancy of our variable-length \(q\)-balancing schemes. To this end, we borrow tools from lattice-path combinatorics and provide closed formulas for the upper bounds on the average redundancy of both schemes A and B (described in Sections III and V). Unfortunately, as with [10], we are unable to complete the asymptotic analysis for Scheme A. Hence, we introduce Scheme B which uses slightly more redundant bits, and show that Scheme B incurs average redundancy of at most \(\frac{1}{2} \log_2 n + 2.526\) redundant bits asymptotically when \(q > 0\). Interestingly, for the case \(q = 0\), the average redundancy of Scheme B can be reduced to \(\frac{1}{2} \log_2 n + 0.526\) and this is better than the schemes given in [9].

II. PRELIMINARIES

Throughout this paper, we fix \(n\) to be an even integer and let \([n]\) denote the set \(\{0, 1, 2, \ldots, n\}\). For a binary word \(x\) and \(j \in [n]\), we let \(x^{(j)}\) and \(x_{(j)}\) denote its length-\(j\) prefix and length-\(j\) suffix, respectively. Also, for two binary words \(x\) and \(y\), we use \(xy\) to denote their concatenation and \(x^n\) to denote the concatenation of \(n\) copies of \(x\). Finally, \(\mathcal{X}\) denotes the complement of \(x\).

Now, since \(n\) is even, we set \(n = 2m\) and fix a non-negative integer \(q \leq m = n/2\). Recall that a length-\(n\) word \(x\) is \(q\)-balanced if its weight is exactly \(m + q = n/2 + q\). In other words, the imbalance of \(x\) is exactly \(2q\). The collection of all \(q\)-balanced words of length \(n\) is denoted by \(\mathcal{B}(n, q)\). If \(q = 0\), then we simply write \(\mathcal{B}(n)\) as \(\mathcal{B}(n)\) and refer to these words as balanced words. Here, our goal is to efficiently map arbitrary binary messages into codewords in collection \(\mathcal{B}(n, q)\), while incurring as few redundant bits as possible. Formally, we have the following.

---

\(^1\)For ease of exposition in later sections, we use the term “imbalance” to refer to the (signed) difference, instead of absolute difference.
TABLE I: Average Redundancy of Variable-Length 0-Balancing Schemes

| n   | Scheme in [9] | Scheme in [10] | Scheme B | Opt. Redundancy |
|-----|---------------|----------------|----------|-----------------|
| 64  | 5.37          | 3.37           | 3.36     | 3.32            |
| 128 | 6.32          | 4.25           | 3.86     | 4.02            |
| 256 | 7.29          | 4.78           | 4.36     | 4.53            |
| 512 | 8.27          | 5.31           | 4.86     | 5.03            |

Definition 1. An \((n, k, \rho; q)\)-variable-length balancing scheme is a pair of encoding and decoding maps \((E, F)\) such that: (i) \(E\) is an injective encoding map from \(\{0, 1\}^k \times \{0, 1\}^n\) to \(\mathbb{B}(n, q) \times \{0, 1\}^n\). In other words, \(E(x) = (c, p)\) with \(c \in \mathbb{B}(n, q)\) and \(p \in \{0, 1\}^n\), and we refer to \(p\) as the prefix. (ii) \(D\) is a decoding map from \(\mathbb{E}(n) \times \{0, 1\}^k\) such that \(D \circ E(x) = x\) for all \(x \in \{0, 1\}^k\).

Here, \(\rho\) denotes the average redundancy and it is given by the quantity \[\sum_{x \in \{0, 1\}^k} (|E(x)| - k)\]

Clearly, \[|\mathbb{B}(n, q)| = \left(\frac{n}{2} + q\right) = 2^{\frac{n}{2} - \log_2 n + O\left(\frac{1}{n}\right)}\], where the latter asymptotic estimate follows from Stirling’s approximation (see for example, [19, Thm 4.6]). Therefore, the redundancy of \(\mathbb{B}(n, q)\), or, the minimum redundancy, is given by

\[n - \log_2 \left(\frac{n}{2} + q\right) \approx \frac{1}{2} \log_2 n + \Delta, \text{ where } \Delta \approx 0.326 \ldots \quad (1)\]

Here, \(q\) is a constant and asymptotics are taken with respect to \(n\). That is, \(f(n) \sim g(n)\) means that \(\lim_{n \to \infty} f(n)/g(n) = 1\).

As mentioned earlier, when \(q = 0\), we have celebrated Knuth’s balancing technique [7]. A crucial ingredient to this technique is the following simple flipping operation. For any word \(x \in \{0, 1\}^n\) and \(j \in [n]\), we define \(\text{Flip}(x, j) = x_1 x_2 \ldots x_{j-1} \hat{x}_j x_{j+1} \ldots x_n\), where \(\hat{x}_j = 1 - x_j\). Knuth’s technique eliminates any \(j\)-th flips on a word if \(x\) belongs to \(\mathbb{B}(n, q)\). Formally, we say that \(j\) is a q-balancing index for \(x\) if \(\text{Flip}(x, j) \in \mathbb{B}(n, q)\) and we use \(x(\tau)\) to denote the set of all q-balancing indices of \(x\). In other words, \(T(x, q) = \{j \in [n] : \text{Flip}(x, j) \in \mathbb{B}(n, q)\}\). Knuth’s key observation is that \(T(x, 0)\) is always nonempty, or equivalently, a 0-balancing index always exists [7].

Let \(\text{Flip}(x, \tau) = c\). In order to receive the message \(x\), the sender needs to transmit both \(c\) and some representation of \(\tau\). As the set of all possible balancing indices has size \(n\), we require a \([\log_2 n] = \log_2 n + O(1)\)-bit prefix \(p\) to represent the index \(\tau\). Hence, Knuth’s balancing technique results in an \((n, n, \rho; 0)\)-variable-length balancing scheme with \(\rho = \log_2 n\). Note that this is in fact a fixed-length scheme because the prefix \(p\) is always of length \(\rho\). However, \(\rho\) is twice the optimal quantity given in (1). One way to reduce this redundancy is to use a different and possibly smaller set of possible balancing words. Unfortunately, this is not possible [8]. So, in [9], [10], Immink and Weber turn their attention to variable-length balancing schemes. We summarize their results here.

Theorem 1 ([9], [10]). Let \(\ell \in \{1, 2, 3\}\). There exists explicit \((n, n, \rho; \ell)\)-variable-length balancing schemes with average redundancies\(^2\) as follows:

\[\rho_1 = \frac{2}{3} \log_2 n,\]

\[\rho_2 = \frac{1}{2} \log_2 n + 0.916 \ldots \]

\[\rho_3 = \frac{1}{2^{n/3}} \sum_{i=1}^{n/3} i_{\gamma}(i, n) \log i, \]

where

\[\gamma(i, n) = 2^n \left(\sum_{j=1}^{i-2} \cos^n \frac{\pi}{2} j + \sum_{j=1}^{i-1} \cos^n \frac{\pi}{2} \right). \quad (5)\]

We refer to these schemes as Schemes 1, 2, and 3, respectively.

When \(q > 0\), the set of q-balancing indices may be empty. That is, there exist binary words \(x\) with \(T(x, q) = \emptyset\) and we refer to such words as bad words. Hence, a different encoding rule must be applied to these bad words, and simple linear-time methods were proposed and studied by Skachev and Immink [18]. While their q-balancing schemes are in fact variable-length schemes, Skachev and Immink did not provide an analysis of the average redundancy of their schemes and instead argued that \(\log_2 n + O(1)\) redundant bits are sufficient in the worst case (when \(q\) is constant).

Our Contributions.

(I) In this paper, we amalgamate the variable-length scheme in [10] with the q-balancing schemes in [18] to obtain new variable-length q-balancing schemes. We formally describe Schemes A and B in Sections III and V, respectively.

(II) Crucially, our objective is to provide a sharp analysis of the average redundancy of our q-balancing schemes. In Section III, we outline our analysis strategy. Section IV then provides the connection with lattice path combinatorics, a detailed proof that the fraction of bad words is negligible, and finally, a closed expression (8) for an upper bound on the average redundancy.

(III) Unfortunately, we were unable to give an asymptotic estimate for (8). Hence, we make a small modification to obtain Scheme B and demonstrate Theorem 2. In Table I, we compare the average redundancy of Scheme B with those in prior work when \(q = 0\).

Theorem 2. Fix \(q\) to be constant. Scheme B is an \((n, n-1, \rho_B; q)\)-variable-length balancing scheme where

\[\rho_B \leq \frac{1}{2} \log_2 n + 0.526 \ldots, \text{ if } q > 0, \]

\[\sim \frac{1}{2} \log_2 n + 0.526 \ldots, \text{ if } q = 0. \quad (6)\]

Here, we write \(f(n) \leq g(n)\) if \(\lim_{n \to \infty} f(n)/g(n) \leq 1\).

III. FIRST VARIABLE-LENGTH q-BALANCING SCHEME

In this section, we first present Scheme A and outline a strategy to analyze the average redundancy of Scheme A. As mentioned earlier, the main obstacle in the direct application of Knuth’s technique is the existence of bad words. Next, we define bad and good words, and classify good words into two types.

Definition 2. A word \(x\) is bad if \(T(x, q) = T(\bar{x}, q) = \emptyset\). Otherwise, \(x\) is good. Furthermore, \(x\) is Type-1-good if \(T(x, q) \neq \emptyset\); and \(x\) is Type-0-good if \(T(x, q) = \emptyset\) and \(T(\bar{x}, q) \neq \emptyset\).

Lemma 3. If \(\text{wt}(x) \geq n/2 + q\) or \(\text{wt}(x) \leq n/2 - q\), then \(x\) is Type-1-good.

Here, we use Lemma 3 to provide a simple way to obtain a good word from a bad one using 2q redundant bits. We remark that in [18], more sophisticated techniques are used to handle these bad words with less redundancy.

Lemma 4. Suppose that \(x\) is bad. If \(x' = x^{[n-2q]}\), then either \(x'^{(2q)}\) or \(x'^{(n-2q)}\) is Type-1-good. We say that \(x\) is Type-i-bad if \(x'^{(2q)}\) or \(x'^{(n-2q)}\) is Type-1-good.

With this lemma, we are ready to define our first variable-length q-balancing scheme. Henceforth, for expository purposes, we assume that all logarithmic functions return integer values.

Scheme A: An \((n, n, \rho_A; q)\)-balancing scheme

\[\text{INPUT: } x \in \{0, 1\}^n\]

\[\text{OUTPUT: } c \in \mathbb{B}(n, q), p \in \{0, 1\}^n\]

(I) Determine if \(x\) is Type-i-good or Type-i-bad.
If $x$ is Type-1-good, set $\hat{x} = x$.
If $x$ is Type-0-good, set $\hat{x} = \bar{x}$.
If $x$ is Type-1-bad, set $\hat{x} = x^{[n-2q]} q_{2q}$.

By Lemmas 3 and 4, we have that $T(\hat{x}, q) \neq \emptyset$.

(II) Determine the q-balanced word $c$.

- $c \leftarrow \text{Flip}(\hat{x}, \tau)$, where $\tau = \text{min } T(\hat{x}, q)$

(III) Determine the prefix $p$.

- Compute $\Gamma_q(c)$ using (11) (see Section IV).
- $z \leftarrow \text{length-}r$ binary representation of the index of $\tau$ in $\Gamma_q(c)$. Here, $r = \log_2 |\Gamma_q(c)|$.
- If $x$ is Type-i-good, we set $p \leftarrow 0iz$.
- If $x$ is Type-i-bad, we set $p \leftarrow 1izx^{[2q]}$.

Remark 3. When $q = 0$, all words are Type-1-good by Lemma 3. Hence, we omit Step (I) and in Step (II), we simply set $p$ to be $z$.

This case, we recover Immink and Weber’s variable scheme in [10].

Average Redundancy Analysis. We now analyze the average redundancy of Scheme A. To this end, consider a message $x$ and let $z(x)$ and $p(x)$ be the resulting index representation and prefix, respectively. Observe from Scheme A, when the word $x$ is good, the prefix $p(x)$ has length $2 + |z(x)|$. On the other hand, when the word $x$ is bad, the prefix $p(x)$ has length $2 + |z(x)| + 2q \leq 2 + 2q + \log_2 n$. Therefore, if we denote the number of bad words in $\{0, 1\}^n$ by $D(n, q)$, we have that

$$\rho_A \leq 2 + \frac{1}{2^n} \left( \sum_{x \text{ is good}} |z(x)| \right) + \frac{D(n, q)}{2^n} \cdot \frac{2q + \log_2 n}{(2q + \log_2 n)}.$$  

(7)

Hence, in the next section, we use lattice path combinatorics to analyse both $D(n, q)$ and $\sum_{x \text{ is good}} |z(x)|$. For the former quantity, we have the following proposition whose proof is deferred to Section IV.

Proposition 5. For fixed $q$, we have $D(n, q) = O(2^n)$.

Therefore, it remains to study the latter quantity and we derive a closed formula in Section IV. Then, together with (7), we have the following estimate for the average redundancy of Scheme A.

Theorem 6. Scheme A is an $(n, n, \rho_A; q)$ variable-length balancing scheme where

$$\rho_A \leq \frac{1}{2^n} \sum_{i=0}^{n+1} \gamma_q(i, n) \log_2 i,$$

and $\gamma_q(i, n)$ is computed using Theorem 13.

IV. ANALYSIS USING LATTICE PATH COMBINATORICS

In this section, we complete the analysis of $\rho_A$. Results in this section are based on a classic combinatorial problem – lattice path enumeration. We remark that Immink and Weber applied similar methods for their analysis of variable-length 0-balancing schemes [9], [10]. Here, we not only extend the analysis to the case where $q > 0$, but also use lattice path combinatorics to enumerate bad words. Due to space constraints, we omit certain technical proofs. A longer version of this paper with detailed proofs is available here [20].

Definition 4. A path in the integer lattice plane $\mathbb{Z}^2$ is simple if it starts from a lattice point and consists of horizontal $\rightarrow$ and vertical $\uparrow$ unit steps in the positive direction.

Since the 1850s, lattice paths have been extensively studied and we refer the reader to Krattenthaler [21] for a comprehensive survey. Here, we state a selection of results.

Theorem 7 ([21]). Suppose that $a$, $b$, $c$, $d$, $s$, and $t$ are integers such that $a \leq c$, $b \leq d$, $a + s \leq b \leq a + t$, and $c + s \leq d \leq c + t$. Set $\Delta_x = c - a$, $\Delta_y = d - b$, and $\Delta_h = t - s + 2$. Then the number of simple paths from $(a, b)$ to $(c, d)$ that stay below\(^4\) the line $Y = X + t$ and above the line $Y = X + s$ is given by

$$\sum_{k=1}^{[(\Delta_h - 1)/2]} 4 \cdot 2 \cos \frac{\pi k}{\Delta_h} \Delta_{x+y} \frac{\pi k(a - b + t + 1)}{\Delta_h} \sin \frac{\pi k(c - d + t + 1)}{\Delta_h}.$$  

(9)

Next, we describe a transformation II that maps binary words $\{0, 1\}^n$ of certain weight to length-$n$ simple paths between certain lattice points in $\mathbb{Z}^2$. Specifically, for any binary word $x = x_1 x_2 \ldots x_n$, we define a simple path $\Pi(x)$ where

- $\Pi(x)$ starts from $(0, w(x) - m)$ (recall that $n = 2m$);
- When $x_1 = 0$, we move a vertical unit $\uparrow$;
- When $x_1 = 1$, we move a horizontal unit $\rightarrow$.

In Figure 1, we provide examples of the transformation II. The following lemma summarizes certain properties of II.

Lemma 8. Fix $q$ and recall that $\mathcal{B}(n, q)$ is the set of words with weight $m + q$. Let $\mathcal{P}$ be the set of all simple paths from $(0, q)$ to $(m + q, m)$. Then II is a bijection from $\mathcal{B}(n, q)$ to $\mathcal{P}$.

Enumeration of Bad Words. First, we apply Lemma 8 to characterize good and bad words.

Lemma 9. Let $x \in \{0, 1\}^n$. Then $x$ is bad if and only if $\Pi(x)$ is strictly between $Y = X + q$ and $Y = X - q$.

Therefore, the number of bad words is immediate from Lemma 9 and Theorem 7.

Proposition 10. We have that

$$D(n, q) = \sum_{y=1-q}^{q-1} \frac{1}{2^n} \sum_{k=1}^{[\Delta_h]} 4 \cdot 2 \cos \frac{\pi k}{\Delta_h} \Delta_{x+y} \frac{\pi k(q-y)}{\Delta_h} \sin \frac{\pi k(q+y)}{2q}.$$  

(10)

Next, we estimate asymptotically the number of bad words. Note that each summand is at most $\frac{4}{\Delta_h} \left( \frac{2 \cos \frac{\pi k}{\Delta_h}}{2q} \right)^n$. Now, since $0 < \frac{\pi k}{\Delta_h} < \frac{\pi}{2}$ and so, each summand is at most $\frac{4}{\Delta_h} \left( \frac{2 \cos \frac{\pi}{2}}{2q} \right)^n$. Since there are at most $\frac{4(q-1)^2}{q}$ summands, we have $D(n, q) \leq \frac{4(q-1)^2}{q} \left( \frac{2 \cos \frac{\pi}{2}}{2q} \right)^n = o(2^n)$, proving Proposition 5.

Average Redundancy of Good Words. Here, we provide a closed formula for the quantity $\sum_{x \text{ is good}} |z(x)|$ defined in (7). Immink and Weber first investigated this quantity in their proposed variable-length 0-balancing scheme [10]. In their paper, instead of studying the set of balancing indices associated with a message $x$, they investigated the set of 0-balancing indices that can be received with a codeword $c$.

Formally, for $c \in \mathcal{B}(n, q)$, we consider the following set: $\Gamma_q(c) \triangleq \{ j \in [n] : \text{Flip}(x, j) = c, \tau_q(x) = j \}$. Here, $\tau_q(x)$ is defined to be $\min T(x, q)$. It turns out that given $c \in \mathcal{B}(n, q)$, we can determine $\Gamma_q(c)$ efficiently. Specifically, when $q = 0$, Immink and Weber provided a simple characterization of $\Gamma_q(c)$ using the notion of running sum.

Definition 5. Let $x \in \{0, 1\}^n$. The running sum of $x$, denoted by $R(x)$, is a length-$(n + 1)$ integer-valued vector defined by $R(x)_0 = 0$ and $R(x)_i = R(x)_{i-1} + (-1)^{i-1} x_i$, if $i > 0$.

Next, we generalize a result in [10] for the case where $q \geq 0$.

Proposition 11. Let $c \in \mathcal{B}(n, q)$. Then

$$\Gamma_q(c) = \{ i \in [n] : R(c)_i \neq R(c)_{i+1}, \text{ for all } j < i \}.$$  

(11)

Next, we consider the set $\mathcal{E}_q(i, n) = \{ c \in \mathcal{B}(n, q) : \Gamma_q(c) = i \}$ and determine its size $\gamma_q(i, n) \triangleq | \mathcal{E}_q(i, n) |$. To characterize the words in $\mathcal{E}_q(i, n)$, we introduce the notion of width.

4Here, we include simple paths that touch both lines $Y = X + s$ and $Y = X + t$.  

Copyright (C) 2022 by IEICE

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Definition 6. Consider a lattice path \( \pi \). Suppose that \( t_{\text{min}} \) and \( s_{\text{max}} \) are the smallest and largest integers such that \( \pi \) lies in between \( Y = X + t_{\text{min}} \) and \( Y = X + s_{\text{max}} \). Then the width of \( \pi \) is defined to be \( t_{\text{min}} - s_{\text{max}} \).

Lemma 12. Fix \( q \) and \( 1 \leq i \leq n + 1 \). Then \( x \in \mathcal{E}_q(i, n) \) if and only if the path \( \Pi(x) \) has width exactly \( i - 1 \).

Hence, determining \( \gamma_q(i, n) \) is equivalent to enumerating lattice paths with a certain width. Therefore, using principles of inclusion and exclusion with Theorem 7, we have the following result.

Theorem 13. We have that

\[
\gamma_q(i, n) = \sum_{t=0}^{i-1} \left( G(i-1, t) - G(i-2, t) - G(i-2, t-1) + G(i-3, t-1) \right),
\]

where \( G(i, t) \) is given by the expression

\[
\frac{2^{n+2}}{i+2} \sum_{k=1}^{\lfloor (i+1)/2 \rfloor} \frac{\pi_k}{i+2} \sin \frac{\pi_k(t+1)}{i+2} \sin \frac{\pi_k(t+1-2q)}{i+2}.
\]

Note that when \( q = 0 \), we recover (5). Inserting the expression of \( \gamma_q(i, n) \) into (8), we obtain a closed formula for an upper bound of \( \rho_A \). As (13) appears unamenable to asymptotic analysis, we introduce our second variable-length balancing scheme.

V. SECOND VARIABLE-LENGTH Q-BALANCING SCHEME

In this section, we define Scheme B and analyze its average redundancy with methods similar to previous sections. We then complete the asymptotic analysis and prove Theorem 2. The main difference of Scheme B is that we consider a length-(\( n - 1 \)) message, and first encode into a codeword \( c' \) with weight \( m + q - 1 \) or \( m + q \). Then, we append one extra redundant bit so that the resulting codeword \( c \) belongs to \( \mathcal{B}(n, q) \). Surprisingly, this modification simplifies the asymptotic analysis and allows us to prove that the average redundancy of Scheme B is within an additive constant from the optimal value in (1) (when \( q = 0 \)).

As before, for a word \( x \in \{0, 1\}^{n-1} \), we define the set of indices \( T_B(x, q) \) and use it to classify as good or bad.

Definition 7. Let \( x \in \{0, 1\}^{n-1} \). Then we set \( T_B(x, q) = \{ j \in [n-1] : \text{wt}(\text{Flip}(j, x)) \in \{m + q - 1, m + q\} \} \). Then \( x \) is bad if \( T_B(x, q) = T_B(x, q) = \emptyset \). Otherwise, \( x \) is Type-1-good if \( T_B(x, q) \neq \emptyset \); and \( x \) is Type-0-good if \( T_B(x, q) = \emptyset \) and \( T_B(x, q) = \emptyset \).

Using similar methods, we obtain the analogue of Lemma 4.

Lemma 14. Suppose that \( x \) is bad. If \( x' = x^{(n-2q-3)} \), then either \( x' \) or \( x' \) is Type-1-good. We say that \( x \) is Type-i-bad if \( x' \) is Type-1-good.

Scheme B: An \((n, n-1, \rho_B; q)\)-balancing scheme

INPUT: \( x \in \{0, 1\}^{n-1} \)

OUTPUT: \( c' \in \mathcal{B}(n, q) \), \( p \in \{0, 1\}^* \)

(I) Determine if \( x \) is Type-i-good or Type-i-bad.

- If \( x \) is Type-1-good, set \( \bar{x} = x \).
- If \( x \) is Type-0-good, set \( \bar{x} = \overline{x} \).
- If \( x \) is Type-i-bad, set \( \bar{x} = x^{[n-2q-3], 2q-2} \).

(II) Determine the \( q \)-balanced word \( c' \).

- \( c \leftarrow \text{Flip}(x, \tau) \) where \( \tau = \text{min} \ T_B(\bar{x}, q) \).
- If \( \text{wt}(c) = m + q - 1 \), append 1 to \( c \) to obtain \( c' \).
- If \( \text{wt}(c) = m + q \), append 0 to \( c \) to obtain \( c' \).

(III) Determine the prefix \( p \).

- Compute \( \Gamma_A(c) \) using Proposition 16
- If \( x \) is Type-i-good, we set \( p \leftarrow \text{0iz} \).
- If \( x \) is Type-i-bad, we set \( p \leftarrow \text{1izx}_{(2q-2)} \).

As before, to estimate the average redundancy \( \rho_B \), we determine the number of bad words and the average redundancy of good words. Let \( x \) be a message and set \( z(x) \) and \( p(x) \) to be the resulting index representation and prefix, respectively. If we denote the number of bad words by \( D_B(n-1, q) \), we have that

\[
\rho_B \leq \frac{3}{2^{n-1}} \left( \sum_{x \text{ is good}} \left| z(x) \right| \right) + \frac{D_B(n-1, q)}{2^{n-1}} (2q - 2 + \log_2 n).
\]

Using similar methods as before, we have the following corollary.

Corollary 15. For fixed \( q \), \( D_B(n-1, q) \leq D(n, q) = o(2^n) \).

Next, we estimate \( \sum_{x \text{ is good}} \left| z(x) \right| \). To do so, we set \( \tau_q(x) = \min T_B(x, q) \) and \( \Gamma_q(c) = \{ j \in [n-1] : \text{Flip}(j, x) = c, \tau_q(x) = j \} \) for \( x \in \{0, 1\}^{n-1} \). As before, we characterize a \( \Gamma_q(c) \) using the running sum of \( c \).

Proposition 16. Suppose that \( \text{wt}(c) \in \{m + q - 1, m + q\} \).

\[
\Gamma_q(c) = \begin{cases} \{ i \in [n-1] : R(c_i) \geq 0, R(c_i) \neq R(c_i) \text{ for } j < i \}, & \text{if } \text{wt}(c) = n/2 + q - 1, \\ \{ i \in [n-1] : R(c_i) \leq 0, R(c_i) \neq R(c_i) \text{ for } j < i \}, & \text{if } \text{wt}(c) = n/2 + q. \end{cases}
\]

As before, we consider the set of length-(\( n - 1 \)) words: \( \mathcal{E}_q(i, n-1) = \{ \text{wt}(c) = m + q - 1, m + q \} \). We then estimate the size \( \gamma_q(i, n-1) = \left| \mathcal{E}_q(i, n-1) \right| \). We then characterize codewords using its lattice path representations and apply Theorem 7 to a surprisingly clean expression for \( \gamma_q(i, n-1) \).

Theorem 17. Fix \( q > 0 \). Then \( \gamma_q(i, n-1) \) is given by

\[
\sum_{x \text{ is good}} \left| z(x) \right| = \frac{1}{2^{n-1}} \sum_{x \text{ is good}} \frac{m+q}{m+i-q} \frac{2i - 2q}{2m} \frac{2m}{m+i+q} (i \log i) + \frac{1}{2^{n-1}} \sum_{x \text{ is bad}} \frac{m+q}{m+i-q} \frac{2i - 2q}{2m} \frac{2m}{m+i+q} (i \log i).
\]
We present the main result of this section.

**Theorem 18.** Fix $q$. Then

$$
\frac{1}{2^{n-1}} \sum_{x \text{ is good}} |z(x)| \preceq \frac{1}{2} \log n + \beta,
$$

where $\beta = \frac{2 - \ln 4 \cdot \gamma}{\ln 4} - \frac{1}{2} \approx -0.474\ldots$ and $\gamma$ is the Euler-Mascheroni constant.

**Proof Sketch.** Let $Z_1$ be the first summand of (15) and we estimate this quantity. First, via standard manipulations, we have

$$Z_1 \leq \frac{2}{m} \sum_{k=1}^{m} (k + q)^2 \log(k + q) \left( \frac{2m}{m_k} \right)^m. \tag{17}$$

Next, we have Stirling's estimate [19]:

$$\sum_{k=1}^{m} (k + q)^2 \log(k + q) \left( \frac{2m}{m_k} \right)^m \sim \frac{1}{\sqrt{m}}. \tag{20}$$

Then setting

$$F(x) = (x + q)^2 \log(x + q) \text{ in [19, Theorem 4.9]},$$

we have that

$$\sum_{k=1}^{m} (k + q)^2 \log(k + q) \left( \frac{2m}{m_k} \right)^m \sim \int_0^\infty e^{-x^2/m} (x + q)^2 \log(x + q) \, dx \approx m^{3/2} \int_0^\infty e^{-z^2} \log zdz + \frac{m^{3/2}}{2} \int_0^\infty e^{-z^2} z^2 \, dz.$$  

Similarly, for the second summand in (15), we obtain the exact same expression. Now, we have

$$\int_0^\infty e^{-z^2} \log zdz = \frac{\sqrt{\pi} \ln 4}{4 \ln 4} \quad \text{and} \quad \int_0^\infty e^{-z^2} z^2 \, dz = \frac{\pi}{4}. \tag{21}$$

So, combining everything, we have that the right-hand side of (15) tends to \( \frac{1}{2} \log n + \left( \frac{2 - \ln 4 \cdot \gamma}{\ln 4} - \frac{1}{2} \right) \approx \frac{1}{2} \log n - 0.474\ldots \), as required.

Together with (14), we obtain the desired upper bound for $P_B$ and obtain Theorem 2. Moreover, when $q = 0$, we have that all words are good. In this case, we need not prepend $z$ with the two bits and thus, the average redundancy is simply given by $1 + \frac{1}{2} \log n + \beta \approx \frac{1}{4} \log n + 0.526$.

**References**

[1] K. A. S. Immink, *Codes for mass data storage systems*. Shannon Foundation Publisher, 2004.

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

[3] I. Shomorony, R. Heckel, “Information-theoretic foundations of dna data storage,” *Foundations and Trends Commun. and Inform. Theory*, vol. 19, no. 1, pp. 1–106, 2022.

[4] M. G. Ross, C. Russ, M. Costello, A. Hollinger, N. J. Lennon, R. Hegarty, C. Nusbaum, and D. B. Jaffe, “Characterizing and measuring bias in sequence data,” *Genome biology*, vol. 14, no. 5, pp. 1–20, 2013.

[5] K. A. S. Immink, K. Cai, “Properties and constructions of constrained codes for DNA-based data storage,” *IEEE Access*, vol. 8, pp. 49523–49531, 2020.

[6] K. Cai, Y. M. Chee, R. Gabrys, H. M. Kiah, T. T. Nguyen, “Correcting a single indel/edit for DNA-based data storage: Linear-time encoders and order-optimality,” *IEEE Trans. Inform. Theory*, vol. 67, no. 6, pp. 3438–3451, 2021.

[7] D. Knuth, “Efficient balanced codes,” *IEEE Trans. Inform. Theory*, vol. 32, no. 1, pp. 51–53, 1986.

[8] N. Alon, E. E. Bergmann, D. Coppersmith, A. M. Odlyzko, “Balancing sets of vectors,” *IEEE Trans. Inform. Theory*, vol. 34, no. 1, pp. 128–130, 1988.

[9] J. H. Weber, K. A. S. Immink, “Knuth’s balanced codes revisited,” *IEEE Trans. Inform. Theory*, vol. 56, no. 4, pp. 1673–1679, 2010.

[10] K. A. S. Immink, J. H. Weber, “Very efficient balanced codes,” *IEEE Journal Selected Areas Commun.*, vol. 28, no. 2, pp. 188–192, 2010.

[11] C. D. Nguyen, Y. K. Vu, K. Cai, “Two-dimensional weight-constrained codes for crossbar resistive memory arrays,” *IEEE Commun. Letters*, vol. 25, no. 5, pp. 1435–1438, 2021.

[12] T. T. Nguyen, K. Cai, K. A. S. Immink, Y. M. Chee, “Efficient design of capacity-approaching two-dimensional weight-constrained codes,” in *2021 IEEE Int. Symp. Inform. Theory*. 2021, pp. 2930–2935.