IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 27, 2020, accepted March 7, 2020, date of publication March 11, 2020, date of current version March 19, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2980036

Properties and Constructions of Constrained Codes for DNA-Based Data Storage

KEES A. SCHOUHAMER IMMINK[1], (Life Fellow, IEEE), AND KUI CAI[2], (Senior Member, IEEE)

[1]Turing Machines Inc., 3016 DK Rotterdam, The Netherlands
[2]Singapore University of Technology and Design (SUTD), Singapore 487372

Corresponding author: Kees A. Schouhamer Immink (immink@turing-machines.com)

This work was supported by the Singapore Ministry of Education Academic Research Fund Tier 2 under Grant MOE2016-T2-2-054.

ABSTRACT We describe properties and constructions of constraint-based codes for DNA-based data storage which account for the maximum repetition length and AT/GC balance. Generating functions and approximations are presented for computing the number of sequences with maximum repetition length and AT/GC balance constraint. We describe routines for translating binary runlength limited and/or balanced strings into DNA strands, and compute the efficiency of such routines. Expressions for the redundancy of codes that account for both the maximum repetition length and AT/GC balance are derived.

INDEX TERMS Constrained coding, maximum runlength, balanced words, storage systems, DNA-based storage.

I. INTRODUCTION

The first large-scale archival DNA-based storage architecture was implemented by Church *et al.* [1] in 2012. Blawat *et al.* [2] described successful experiments for storing and retrieving data blocks of 22 Mbyte of digital data in synthetic DNA. Erlich and Zielinski [3] further explored the limits of storage capacity of DNA-based storage architectures. Recent examples of experimental work on DNA-base storage can be found in [4]–[6].

Naturally occurring DNA consists of four types of *nucleotides*: adenine (A), cytosine (C), guanine (G), and thymine (T). A DNA strand (or *oligonucleotides*, or *oligo* in short) is a linear sequence of these four nucleotides that are composed by DNA synthesizers. Binary source, or user, data are translated into the four types of nucleotides, for example, by mapping two binary source into a single nucleotide, in short *nt*.

Strings of nucleotides should satisfy a few elementary conditions, called *constraints*, in order to be less error prone. Repetitions of the same nucleotide, a *homopolymer run*, significantly increase the chance of sequencing errors [7], [8], so that such long runs should be avoided. For example, in [8], experimental studies show that once the homopolymer run is larger than four nt, the sequencing error rate starts increasing significantly. In addition, [8] also reports that oligos with large unbalance between GC and AT content exhibit high dropout rates and are prone to polymerase chain reaction (PCR) errors, and should therefore be avoided.

Blawat's format [2] incorporates a constrained code that uses a look-up table for translating binary source data into strands of nucleotides with a homopolymer run of length at most three. Blawat's format did not incorporate an AT/GC balance constraint. Strands that do not satisfy both the maximum homopolymer run requirement and the weak balance constraint are barred in Erlich's coding format [3].

In this paper, we describe properties and constructions of quaternary constraint-based codes for DNA-based storage which account for a maximum homopolymer run and maximum unbalance between AT and GC contents. Binary 'balanced' and runlength limited sequences have found widespread use in data communication and storage practice [9]. We show that constrained binary sequences can easily be translated into constrained quaternary sequences, which opens the door to a wealth of efficient binary code constructions for application in DNA-based storage [10]–[13]. A further advantage of the binary-to-binary translation instead of a 'direct' binary-to-quaternary translation is the lower complexity of encoding and decoding look-up tables.

The associate editor coordinating the review of this manuscript and approving it for publication was Nadeem Iqbal.

The disadvantage is, as we show, the loss in information capacity of the binary versus the quaternary approach.

We start in Section II with a description of the limiting properties and code constructions that impose a maximum homopolymer run. We specifically compute and compare the information capacity of binary versus 'direct' quaternary coding techniques. In Section III, we enumerate the number of binary and quaternary sequences with combined AT and GC contents and run-length constraints. Section IV concludes the paper.

II. MAXIMUM RUNLENGTH CONSTRAINT

Long repetitions of the same nucleotide (nt), called a *homopolymer run* or *runlength*, may significantly increase the chance of sequencing errors [7], [8], and should be avoided. Avoiding long runs of the same nucleotide will result in loss of information capacity, and codes are required for translating arbitrary source data into constrained quaternary strings. Binary runlength limited (RLL) codes have found widespread application in digital communication and storage devices since the 1950s [9], [14]. MacLauhlin *et al.* [15] studied multi-level runlength limited codes for optical recording. A string of n-nucleotide oligo's of 4-ary symbols can be seen as two parallel *binary* strings of length n, where the 4-ary symbol is represented by two binary symbols. Such a system of multiple parallel data streams with joint constraints is reminiscent of 'two-dimensional' track systems, which have been studied by Marcellin and Weber [16].

We start in the next subsection with the counting of q-ary sequences that satisfy a maximum runlength, followed by subsections where we describe limiting properties and code constructions that avoid $m+1$ repetitions of the same nucleotide.

A. COUNTING q-ARY SEQUENCES, CAPACITY

Let the number of q-ary n-length sequences having a maximum run, m, of the same symbol be denoted by $N_q(m, n)$. The number $N_q(m, n)$ is found by using the recursive relation [17, Part 1]:

$$N_q(m,n) = \begin{cases} q^n, & n \le m, \\ (q-1)\sum_{k=1}^{m} N_q(m, n-k), & n > m. \end{cases} \tag{1}$$

For $n \le m$ the above is trivial as all sequences satisfy the maximum runlength constraint. For $n > m$ we follow Shannon's approach [17] for the discrete noiseless channel. The runlength of k symbols a can be seen as a 'phrase' a of length k. After a phrase a has been emitted, a phrase of symbols $b \neq a$ of length k can be emitted without violating the maximum runlength constraint imposed. The total number of allowed sequences, $N_q(m, n)$, is equal to $(q-1)$ times the sum of the numbers of sequences ending with a phrase of length $k = 1, 2, \ldots, m$, which are equal to $N_q(m, n-k)$. Addition of these numbers yields (1), which proves (1). Using the above expression, we may easily compute the feasibility of a q-ary m-constrained code for relatively small values of n where a coding look-up table is practicable, see Subsection II-C for more details.

1) GENERATING FUNCTIONS

Generating functions are a very useful tool for enumerating constrained sequences [18], and they offer tools for approximating the number of constrained sequences for asymptotically large values of the sequence length n. The series of numbers $\{N_q(m, n)\}$, $n = 1, 2 \ldots$, in (1), can be compactly written as the coefficients of a formal power series $H_{q,m}(x) = \sum N_q(m, i)x^i$, where x is a dummy variable. There is a simple relationship between the generating function, $H_{q,m}(x)$, and the linear homogenous recurrence relation (1) with constant coefficients that defines the same series [18]. We first define a generating function

$$G(x) = \sum g_i x^i. \tag{2}$$

Let the operation $[x^n]G(x)$ denote the *extraction* of the coefficient of x^n in the formal power series $G(x)$, that is, define

$$[x^n]\left(\sum g_i x^i\right) = g_n. \tag{3}$$

Let

$$T(x) = \sum_{i=1}^{m} x^i. \tag{4}$$

The generating function for the number of q-ary sequences with a maximum runlength m is

$$qT(x) + q(q-1)T(x)^2 + q(q-1)^2T(x)^3 + \cdots.$$

We may rewrite the above as

$$\frac{qT(x)}{1-(q-1)T(x)},$$

so that the number of n-symbol m-constrained q-ary words is

$$N_q(m,n) = [x^n]\frac{qT(x)}{1-(q-1)T(x)}. \tag{5}$$

2) ASYMPTOTICAL BEHAVIOR

For asymptotically large codeword length n, the maximum number of (binary) user bits that can be stored per q-ary symbol, called *(information) capacity*, denoted by $C_q(m)$, is given by [17]

$$C_q(m) = \lim_{n\to\infty}\frac{1}{n}\log_2 N_q(m,n) = \log_2 \lambda_q(m), \tag{6}$$

where $\lambda_q(m)$, is the largest real root of the characteristic equation [15], [17]

$$x^{m+1} - qx^m + q - 1 = 0. \tag{7}$$

Table 1 shows the information capacities $C_2(m)$ and $C_4(m)$ versus maximum allowed (homopolymer) run m. For asymptotically large n we may approximate $N_q(m, n)$ by [18]

$$N_q(m,n) \approx A_q(m)\lambda_q^n(m). \tag{8}$$

TABLE 1. Capacity $C_2(m)$ and $C_4(m)$ versus m.

m	$C_2(m)$	$C_4(m)$
1	0.0000	$1.5850(=\log_2 3)$
2	0.6942	1.9227
3	0.8791	1.9824
4	0.9468	1.9957
5	0.9752	1.9989
6	0.9881	1.9997

TABLE 2. Coefficient $A_2(m)$ and $A_4(m)$ versus m.

m	$A_2(m)$	$A_4(m)$
1		$1.3333(=4/3)$
2	1.4477	1.1031
3	1.2368	1.0341
4	1.1327	1.0110
5	1.0759	1.0034
6	1.0435	1.0010

The coefficient $A_q(m)$ is found, see [14, page 157-158], by rewriting $H_{q,m}(x)$ as a quotient of two polynomials, or $H_{q,m}(x) = \frac{r(x)}{p(x)}$. Then

$$A_q(m) = -\lambda_q(m)\frac{r(1/\lambda_q(m))}{p'(1/\lambda_q(m))}. \tag{9}$$

Table 2 shows the coefficients $A_2(m)$ and $A_4(m)$ versus m. For $m = 1$, we simply find $N_4(1, n) = 4.3^{n-1}$. We found that the approximation (8) is remarkably accurate. For a typical example, $N_4(2, 10) = 676836$, while the approximation using (8) yields $N_4(2, 10) \approx 676835.9769$. The redundancy of a 4-ary string of length n with a maximum runlength m, denoted by $r_4(m, n)$, is, using (8),

$$\begin{aligned} r_4(m, n) &= 2n - \log_2 N_4(m, n) \\ &\approx n\,(2 - C_4(m)) - \log_2 A_4(m). \end{aligned} \tag{10}$$

B. BINARY-BASED RLL CODE CONSTRUCTION, CONSTRUCTION I

Yazdi *et al.* [19] and Taranalli *et al.* [20] showed that we may exploit binary maximum runlength limited (RLL) codes for constructing quaternary RLL codes. Their construction, denoted by Construction 1, exemplifies such a technique for $m > 1$. The construction is simple, but we show below that this simplicity has its price in terms of extra redundancy.

Construction 1: Let $\boldsymbol{u} = (u_1, \ldots, u_n)$ be an n-bit RLL string. We merge the RLL n-bit string, $\boldsymbol{u}$, with an n-bit source string $\boldsymbol{y} = (y_1, \ldots, y_n)$, by using the addition $v_i = u_i + 2y_i$, $1 \le i \le n$, where $\boldsymbol{v} = (v_1, \ldots, v_n)$, $v_i \in \mathcal{Q}$ is the 4-ary output string. It is easily verified that the 4-ary output string, $\boldsymbol{v}$, has maximum allowed run m, the same as the binary string $\boldsymbol{u}$.

The number of distinct 4-ary sequences, $\boldsymbol{v}$, of Construction 1 equals $2^n N_2(m, n)$, so that the redundancy, denoted by $r_2(m, n)$, is

$$r_2(m, n) \approx n\,(1 - C_2(m)) - \log_2 A_2(m). \tag{11}$$

TABLE 3. Asymptotic rate efficiency, $\eta(m)$, of binary Construction 1 versus maximum homopolymer run, m.

m	$\eta(m)$
2	0.881
3	0.948
4	0.975
5	0.988
6	0.994
7	0.997

TABLE 4. Rate efficiency, $R_{m,0}/C_4(m)$, of binary Construction 1 versus strand length, n, and maximum homopolymer run, m.

n	$m = 2$	$m = 3$	$m = 4$
5	0.832	0.807	0.802
6	0.780	0.841	0.835
7	0.817	0.865	0.859
8	0.845	0.883	0.877
9	0.809	0.897	0.891
10	0.832	0.908	0.902

The rate efficiency with respect to the runlength limited 4-ary channel, denoted by $\eta(m)$, is expressed by

$$\eta(m) = \frac{1 + C_2(m)}{C_4(m)}. \tag{12}$$

Table 3 lists results of computations. We may notice that Construction 1 will suffer a loss of up to 12 % for $m = 2$. For larger values of m, however, the loss is negligible.

The above asymptotic efficiency of Construction 1, $\eta(m)$, is valid for very large values of the strand length n. It is of practical interest to assess the efficiency for smaller values of the strand length. Construction 1 can be used with any binary RLL code, and there are many binary code constructions for generating maximum runlength constrained sequences, see [14] for an overview. We propose here, for the efficiency assessment, a simple two-mode block code of codeword length n. Runlength constrained codewords in the first mode start with a symbol 'zero', while codewords in the second mode start with a 'one'. When the previous sent codeword ends with a 'one' we use the codewords from the first mode and *vice versa*. The number of binary source words that can be accommodated with Construction 1 equals $2^{n-1}N_2(m, n)$, so that the code rate, denoted by $R_{m,0}$, is

$$R_{m,0} = \frac{1}{n}\left(n - 1 + \lfloor \log_2 N_2(m, n) \rfloor\right), \tag{13}$$

where we truncated the code size to the largest power of two. Table 4 shows selected outcomes of computations of the rate efficiency $R_{m,0}/C_4(m)$ versus m and n.

C. ENCODING OF QUATERNARY SEQUENCES WITHOUT BINARY STEP

In this subsection, we investigate two simple constructions of codes that transform binary source words directly (that is, without an intermediate binary coding step) into 4-ary maximum homopolymer constrained codewords. An example of a simple 4-ary block code was presented by

TABLE 5. Rate efficiency, $R_{m,1}/C_4(m)$, of the two-mode code construction versus strand length, n, and maximum homopolymer run, m.

n	$m=1$	$m=2$	$m=3$	$m=4$
5	0.883	0.832	0.807	0.802
6	0.841	0.867	0.841	0.835
7	0.901	0.892	0.865	0.859
8	0.946	0.910	0.883	0.877
9	0.911	0.925	0.897	0.891
10	0.946	0.936	0.908	0.902

Blawat *et al.* [2]. The code converts 8 source bits into a 4-ary word of 5 nt. The 5-nt words can be cascaded without violating the prescribed $m = 3$ maximum homopolymer run. The rate of Blawat's construction is $R = 8/5 = 1.6$. As $C_4(m = 3) = 1.9824$, see Table 1, the (rate) efficiency of the construction is $R/C_4(m) = 0.807$. Alternative, and more efficient, constructions are described below.

In the first construction, denoted by *two-mode construction*, each source word can be represented by one of two possible codewords, where the codeword sent is chosen to satisfy the runlength constraint at the junction of two cascaded codewords. Decoding is accomplished by observing the n-symbol codeword. In the second, slightly more efficient, construction, denoted by *four-mode construction*, a source word can be represented by four possible codewords. Decoding is accomplished by observing the n-symbol codeword plus the last symbol of the previous codeword.

1) TWO-MODE CONSTRUCTION

In this format, a source word can be represented by two n-symbol 4-ary m-constrained codewords, where the alternative representations differ at the first position. In case we append a new codeword to the previous codeword, we are always able to choose (at least) one representation whose first symbol differs from the last symbol of the previous codeword. Then, clearly, the cascaded string of 4-ary symbols satisfies the prescribed maximum homopolymer run constraint. The rate of this two-mode construction, denoted by $R_{m,1}$, is

$$R_{m,1} = \frac{1}{n}(\lfloor \log_2(N_4(m,n)) \rfloor - 1), \tag{14}$$

where we truncated the code size to the largest power of two possible. Table 5 shows outcomes of computations of the rate efficiency $R_{m,1}/C_4(m)$ versus m and n. We observe that, for $m = 2$, the 'quaternary' efficiency $R_{2,1}/C_4(2)$ is slightly better than the 'binary' $R_{2,0}/C_4(2)$, see Table 4. For $m > 2$, both approaches have the same efficiency. The conversion of the binary source symbols into the 4-ary n-nt strands and vice versa can be accomplished using two look-up tables of complexity 4^n.

2) FOUR-MODE CONSTRUCTION

In the above two-mode construction, the encoded codeword depends on the last symbol of the previous codeword. Decoding, however, is based on the observation of the n symbols of the retrieved codeword. In the second construction, the codeword also depends on the last symbol of the previous codeword. Decoding, however, is accomplished by observing the n symbols of the retrieved codeword plus the last symbol of the previous codeword. To that end, we define four tables of codewords, denoted by $L(i, a)$, where i, $1 \le i \le K$, denotes the decimal representation of the source word to be encoded, K denotes the size of the table, and a denotes the last symbol of the previous codeword. The four tables are constructed in such a way that the codewords in each table $L(i, a)$ do not start with the symbol a. As a result, the encoder always generates a symbol transition between the tail and nose symbols of consecutive codewords. The maximum size of the four tables equals $K = \frac{3}{4}N_4(m, n)$ (note that $N_4(m, n)$ is a multiple of 4). Table 6 shows a simple example of the encoding tables of a four-mode code for $n = 2$ and $m = 2$. The size of this code equals $K = 12$. Let, for example, the source sequence be '0', '1', '3', '6'. Then, using the table, the encoded sequence is '10', '11', '03', '22'. We may simply verify that the maximum runlength is $m = 2$. The code size $K = 12$, while the code size of the two-mode code $m = n = 2$ described above equals $16/2 = 8$. The table shows that the codeword '00' is assigned to three source words, namely '0', '4', and '8', so that '00' cannot be decoded unambiguously by observing the codeword. Observation of the retrieved codeword plus the last symbol of te previous codeword solves the ambiguouty.

TABLE 6. Encoding tables of a four-mode code for $n = 2$ and $m = 2$. The parameter i denotes the (decimal) representation of the source word. The tables $L(i, a)$, $a = 0, 1, 2, 3$, show the corresponding codeword, where a denotes the last symbol of the previous codeword.

i	$L(i,0)$	$L(i,1)$	$L(i,2)$	$L(i,3)$
0	10	00	10	10
1	11	01	11	11
2	12	02	12	12
3	13	03	13	13
4	20	20	00	20
5	21	21	01	21
6	22	22	02	22
7	23	23	03	23
8	30	30	30	00
9	31	31	31	01
10	32	32	32	02
11	33	33	33	03

The rate of this four-mode construction, denoted by $R_{m,2}$, is

$$R_{m,2} = \frac{1}{n}\left\lfloor \log_2\left(\frac{3}{4}N_4(m,n)\right)\right\rfloor. \tag{15}$$

Table 7 shows the rate efficiency of the four-mode construction. The efficiency improvement with respect to the two-mode construction, see Table 5, is obtained at the cost of four look-up tables instead of two.

Example: Let (as in Blawat's code [2]) $n = 5$ and $m = 3$. We simply find, using (1), $N_4(3, 5) = 996$, so that the code may accommodate $K = 3/4 \times 996 = 747$ binary source words. Since $K > 512 = 2^9$ we may implement a code of rate 9/5, which is 12% higher than that of Blawat's code of

TABLE 7. Rate efficiency, $R_{m,2}/C_4(m)$, of the four-mode construction versus strand length, n, and maximum homopolymer run, m.

n	$m=1$	$m=2$	$m=3$	$m=4$
5	0.883	0.936	0.908	0.902
6	0.946	0.954	0.925	0.919
7	0.991	0.966	0.937	0.931
8	0.946	0.975	0.946	0.940
9	0.981	0.982	0.953	0.946
10	0.946	0.936	0.958	0.952

rate 8/5. As we have the freedom of deleting $747-512=235$ redundant codewords, we may, for example, bar the words with the highest unbalance.

In the next section, we take a look at the combined AT and GC contents balance and maximum polymer run constrained codes.

III. COMBINED WEIGHT AND MAXIMUM RUN CONSTRAINED CODES

Oligos with large unbalance between GC and AT content exhibit high dropout rates and are prone to polymerase chain reaction (PCR) errors, and should therefore be avoided. Avoidance of such undesired sequences implies an extra redundancy. In this section, we compute the redundancy of binary and quaternary codes with combined RLL and AT/GC constraints.

A. DEFINITION AT/GC CONTENT, BALANCE, AND WEIGHT

We use the nucleotide alphabet $\mathcal{Q} = \{0, 1, 2, 3\}$, where we propose the following relation between the four decimal symbols and the nucleotides: $G = 0, C = 1, A = 2$, and $T = 3$. The *AT/GC content* constraint stipulates that around half of the nucleotides should be either an A or a T nucleotide. In order to study AT-balanced nucleotides, we start with a few definitions. We define the *weight* or *AT-content*, denoted by $w_4(\boldsymbol{x})$, of the n-nucleotide oligo $\boldsymbol{x} = (x_1, \ldots, x_n)$, $x_i \in \mathcal{Q}$, as the number of occurrences of A or T, or

$$w_4(\boldsymbol{x}) = \sum_{i=1}^{n} \varphi(x_i), \tag{16}$$

where

$$\varphi(u) = \begin{cases} 0, & u < 2, \\ 1, & u > 1. \end{cases} \tag{17}$$

The weight of a binary word $\boldsymbol{x} = (x_1, \ldots, x_n)$, $x_i \in \{0, 1\}$, denoted by $w_2(\boldsymbol{x})$, is defined by

$$w_2(\boldsymbol{x}) = \sum_{i=1}^{n} \varphi(2x_i) = \sum_{i=1}^{n} x_i. \tag{18}$$

If we write the 4-ary word $\boldsymbol{x} = (x_1, \ldots, x_n)$, $x_i \in \mathcal{Q}$, as $\boldsymbol{x} = \boldsymbol{y} + 2\boldsymbol{z}$, where y_i and $z_i \in \{0, 1\}$ then

$$w_4(\boldsymbol{x}) = \sum_{i=1}^{n} \varphi(x_i) = \sum_{i=1}^{n} \varphi(2z_i) = w_2(\boldsymbol{z}). \tag{19}$$

Kerpez *et al.* [21], Braun and Immink [22], and Kurmaev [23] analyzed properties and constructions of binary combined weight and runlength constrained codes. Their results are straightforwardly applied to the quaternary case at hand. In the next subsections, we count binary and quaternary sequences that satisfy combined maximum runlength and weight constraints. We start by counting the number of binary sequences, $\boldsymbol{x}$, of length n that satisfy a maximum runlength constraint m and have weight $w = w_2(\boldsymbol{x})$. Paluncic and Maharaj [24] enumerated this number for the balanced case $w = w_2(\boldsymbol{x}) = n/2$.

B. COUNTING BINARY RLL SEQUENCES OF GIVEN WEIGHT

Define the bi-variate generating function $H(x, y)$ in the dummy variables x and y by

$$H(x, y) = \sum_{i,j} h_{i,j} x^i y^j, \tag{20}$$

and let $[x^{n_1} y^{n_2}]h(x, y)$ denote the extraction of the coefficient of $x^{n_1} y^{n_2}$ in the formal power series $\sum h_{i,j} x^i y^j$, or

$$[x^{n_1} y^{n_2}] \left(\sum h_{i,j} x^i y^j \right) = h_{n_1,n_2}. \tag{21}$$

Define

$$T_1(x, y) = \sum_{i=1}^{m} x^i y^i. \tag{22}$$

Let the sequence start with a runlength of zero's, then the generating function for the number of binary sequences with a maximum runlength m is

$$T(x) + T(x)T_1(x, y) + T(x)^2 T_1(x, y) + T(x)^2 T_1(x, y)^2 + \cdots.$$

In case the sequence starts with a run of one's, we obtain for the generating function

$$T_1(x) + T(x)T_1(x, y) + T(x)T_1(x, y)^2 + T(x)^2 T_1(x, y)^2 + \cdots.$$

The generating function for the number of binary sequences with a maximum runlength m starting with a one or a zero runlength is the sum of the two above generating functions. Working out the sum yields

$$\frac{T_1(x, y) + T(x) + 2T_1(x, y)T(x)}{1 - T_1(x, y)T(x)},$$

so that the number of n-bit codewords, $\boldsymbol{x}$, with maximum runlength m, denoted (with a slight abuse of notational convention by adding an extra parameter) by $N_2(m, w, n)$, that satisfy a given unbalance constraint $w = w_2(\boldsymbol{x})$ is given by

$$N_2(m, w, n) = [x^n y^w] \frac{T_1(x, y) + T(x) + 2T_1(x, y)T(x)}{1 - T_1(x, y)T(x)}. \tag{23}$$

With the above bi-variate generating function, we may exactly compute the number of binary m-constrained words of weight w.

More insight is gained by an approximation of $N_2(m, w, n)$. For a given maximum runlength, m, and asymptotically large n, we are specifically interested in the distribution of $\lim_{n\to\infty} N_2(m, w, n)/N(m, n)$ versus the weight w. The weight w of a binary sequence of length n is the sum of the runlengths of ones. The runlengths are random variables, so that for asymptotically large n, according to the Central Limit Theorem [18], the weight distribution approaches a Gaussian distribution with mean $\frac{n}{2}$ and variance denoted by $\sigma_2^2(m, n)$. Then

$$N_2(m, w, n) \approx \mathcal{G}\left(w; \frac{n}{2}, \sigma_2^2(m, n)\right) N_2(m, n), \quad n \gg 1, \quad (24)$$

where

$$\mathcal{G}(u; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{u-\mu}{\sigma})^2}, \quad (25)$$

denotes the Gaussian distribution. The variance, $\sigma_2^2(m, n)$, of the Gaussian distribution is computed below.

1) COMPUTATION OF THE VARIANCE, $\sigma_2^2(m, n)$

Let $\boldsymbol{x}$ be an infinitely long binary m-constrained sequence, where the probabilities of occurrence of the runlengths of zeros and ones are chosen to maximize the information rate (entropy) of the sequence. The probability of occurrence of a runlength of length l, $l \le m$, in a maxentropic sequence equals $\lambda_2^{-l}(m)$, see [14, Chapter 4], where for $q = 2$, see (7), $\sum_{l=1}^{m} \lambda_2^{-l}(m) = 1$. The average runlength, denoted by $\bar{l}$, equals

$$\bar{l} = \sum_{i=1}^{m} i\lambda_2^{-i}(m). \quad (26)$$

The runlength variance of an m-constrained sequence, denoted by $\mathrm{Var}(l)$, is

$$\mathrm{Var}(l) = \sum_{i=1}^{m} (i - \bar{l})^2 \lambda_2^{-i}(m). \quad (27)$$

The weight variance, $\sigma_2^2(m, n)$, of the m-constrained sequence is

$$\sigma_2^2(m, n) = \gamma_2(m)\frac{n}{4}, \quad (28)$$

where

$$\gamma_2(m) = \frac{\mathrm{Var}(l)}{\bar{l}}.$$

Table 8 shows results of computations (note that the parameter $\gamma_4(m)$ is explained in Section III-C). In order to verify the accuracy of the Gaussian approximation, we have numerically compared it with the (accurate) outcomes of the generating function. Figure 1 shows a comparison between the accurate and approximate distributions, $N_2(m, w, n)/N_2(m, n)$, for $n = 100$ and $m = 2, 3, 4$. Except for the discrepancy in the tails of the distributions, the accuracy of the Gaussian approximation is quite sufficient for engineering applications. The Gaussian approximation is accurate within a few percent within the two-sigma limits of the distribution.

TABLE 8. Coefficient $\gamma_2(m)$ and $\gamma_4(m)$ versus maximum homopolymer run m.

m	$\gamma_2(m)$	$\gamma_4(m)$
1		0.5000
2	0.1708	0.7410
3	0.3449	0.8796
4	0.5059	0.9497
5	0.6426	0.9808
10	0.9565	0.9999
∞	1	1

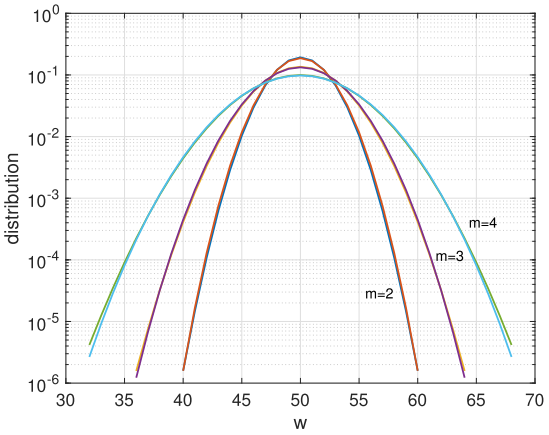


FIGURE 1. Comparison of the weight distribution of $N_2(m, w, n)/N_2(m, n)$, using (a) the Gaussian distribution (24) and (b) generating functions for $n = 100$ and $m = 2, 3, 4$.

C. COUNTING QUATERNARY RLL SEQUENCES OF GIVEN WEIGHT

We count the number of n-tuples $\boldsymbol{x}$ of 4-ary symbols that satisfy a maximum runlength constraint, m, and have weight $w = w_4(\boldsymbol{x})$, denoted (with a slight abuse of notational convention) by $N_4(m, w, n)$.

1) MAXIMUM RUNLENGTH CONSTRAINT

For the special case $m = 1$, Limbachiya *et al.* [25] presented a closed-form expression of $N_4(1, w, n)$. For other values of the prescribed maximum runlength, m, we may readily compute the number of 4-ary sequences, $N_4(m, w, n)$, versus weight, $w = w_4(\boldsymbol{x})$, by applying generating functions.

The 4-ary symbols are generated by a constrained data source that can be modelled as a four-state Moore-type finite-state machine. The machine steps from state to state where when state $i \in \mathcal{Q}$ is visited a sequence of k, $1 \le k \le m$, symbols 'i' are emitted. After visiting state i, the data source may not return to state i (and so forbidding to again emit a sequence of the same symbol 'i'), but it enters state $j \ne i$, $j \in \mathcal{Q}$. When the machine enters state 3 or 4, the word weight, w, is incremented by k, where k, $1 \le k \le m$, denotes the run of symbols '3' or '4'. When, on the other hand, states 1 or 2 are entered, the weight increment is nil. The resulting 4×4 one-step *skeleton* or *state-transition* matrix, $D(x, y)$, of the finite-state machine is

$$D(x, y) = \begin{bmatrix} 0 & a_0 & a_0 & a_0 \\ a_0 & 0 & a_0 & a_0 \\ a_1 & a_1 & 0 & a_1 \\ a_1 & a_1 & a_1 & 0 \end{bmatrix}, \quad (29)$$

TABLE 9. Number of balanced words, $N_4(m, \frac{n}{2}, n)$, versus m and n.

n	$m=1$	$m=2$	$m=3$
6	424	1160	1280
8	3352	14696	17608
10	27208	190848	248360
12	224760	2520888	3563392
14	1880040	33704904	51751168
16	15873240	454767840	758461240

where $a_0 = T(x)$ and $a_1 = T_1(x, y)$. We are now in the position to write a general expression for $N_4(m, w, n)$. The number of 4-ary sequences of length n with maximum runlength constraint m and weight w equals

$$N_4(m, w, n) = [x^n y^w] \frac{1}{3} \sum_{i,j} \sum_{k=1}^{n} d_{i,j}^{[k]}(x, y), \tag{30}$$

where $d_{i,j}^{[k]}(x, y)$ denotes the entries of $D^k(x, y)$. The entries $d_{i,j}^{[k]}(x, y)$ of $D^k(x, y)$ are equal to the number of sequences (paths) of k runlengths starting in state i and ending in state j. Summation for all possible runlengths $k \leq n$ and matrix entries, and division by three yields the generating function of $N_4(m, w, n)$, which proves (30).

Balanced codewords with $w = n/2$, n even, play an important role. Table 9 shows outcomes of computations of $N_4(m, \frac{n}{2}, n)$ using (30), for $m = 1, 2$, and 3. The case $m = 1$ was earlier presented in [25]. Note that the integer sequence $N_4(m = 1, \frac{n}{2}, n)$ versus n is also known as OEIST sequence A085363 (multiplied by 2), for which an alternative generating function is presented in [26].

Generating functions (30) allow us to accurately compute $N_4(m, w, n)$. For some applications, we may sacrifice accuracy for simplicity of the expression. In the next subsection, we derive a simple approximation to $N_4(m, w, n)$ valid for asymptotically large n and small relative weight w/n.

2) ESTIMATE OF THE WEIGHT DISTRIBUTION

The weight $w_4(\boldsymbol{x})$ is the number of nucleotides A and T in the sequence $\boldsymbol{x}$, see (19). Then, as in the binary case above, for asymptotically large n, according to the Central Limit Theorem, the weight distribution is approximately Gaussian, that is, we may conveniently approximate $N_4(m, w, n)$ by

$$N_4(m, w, n) \approx \mathcal{G}\left(w; \frac{n}{2}, \sigma_4^2(m, n)\right) N_4(m, n), \quad n \gg 1, \tag{31}$$

where $\sigma_4^2(m, n)$ denotes the variance of the Gaussian weight distribution. The variance $\sigma_4^2(m, n)$ can be computed as follows.

3) COMPUTATION OF THE VARIANCE $\sigma_4^2(m, n)$

Let u_i, $i = 1, 2, \ldots$, $u_i \in \mathcal{Q}$, be an infinitely long 4-ary sequence generated by a maxentropic source that satisfies a prescribed maximum runlength m. Although the 4-ary sequence u_i, $i = 1, 2, \ldots$, satisfies a limited runlength constraint, m, the runs of the binary weight sequence $v_i = \varphi(u_i)$, $i = 1, 2, \ldots$, see definition (17), are without any limit. The variance, $\sigma_4^2(m, n)$, of the Gaussian weight distribution is governed by the runlength distribution, $P(k)$, of the binary sequence v_i, where $P(k)$, $k > 0$, denotes the probability of occurrence of a runlength k. Clearly, $\sum_{k>0} P(k) = 1$. The probability $P(k)$ is proportional to the number of binary m-sequences of length k, $N_2(m, k)$, times the probability of such a sequence, λ_4^{-k}, or

$$P(k) = cN_2(m, k)\lambda_4^{-k}, \quad k \geq 1, \tag{32}$$

where the normalization constant c is chosen such that $\sum_{k=1}^{\infty} P(k) = 1$. The term $N_2(m, k)$ is the number of AT combinations of length k, which may exist of a single A or T run or a plurality of alternating A and T runs. Then we have

$$\sigma_4^2(m, n) = \gamma_4(m)\frac{n}{4}, \tag{33}$$

where, see [14, Chapter 4],

$$\gamma_4(m) = \frac{1}{\bar{l}} \sum_{k=1}^{\infty} (k - \bar{l})^2 P(k) \tag{34}$$

and

$$\bar{l} = \sum_{k=1}^{\infty} kP(k). \tag{35}$$

Table 8 shows results of computations of $\gamma_4(m)$ versus m. We infer from (31) and Table 8 that, for n fixed, the weight distribution becomes wider with increasing maximum runlength m, see also Figure 1. Note that the above outcome is not consistent with the results by Erlich and Zielinski [3], as they assume a Gaussian balance distribution whose variance equals $n/4$, independent of m.

An estimate of the number of balanced codewords, $N_4(m, \frac{n}{2}, n)$, is

$$N_4\left(m, \frac{n}{2}, n\right) \approx \frac{\sqrt{2}}{\sqrt{\pi\gamma_4(m)n}} N_4(m, n), \quad n \text{ even}. \tag{36}$$

For the case $m = 1$ we have, (see [26], sequence A085363, for a similar result)

$$N_4\left(1, \frac{n}{2}, n\right) \approx \frac{8}{\sqrt{\pi n}} 3^{n-1}, \quad n \text{ even}. \tag{37}$$

Using the above approximation, we obtain, for example, that $N_4(1, 8, 16) \approx 16191008$, which is 2% higher than its exact value, 15873240, listed in Table 9.

D. REDUNDANCY OF BINARY AND QUATERNARY CODES WITH COMBINED RLL AND AT/GC BALANCE CONSTRAINTS

For DNA-based storage, we do not require that the strands of the codebook, $\mathcal{S}$, are strictly balanced, as a small unbalance, that is $\alpha_{\mathcal{S}} \ll 1$, between the GC and AT content is permitted without affecting the error performance. Such a constraint is called a *weak balance constraint*. The *relative unbalance* of a word, $\alpha(\boldsymbol{x})$, is defined by $\alpha(\boldsymbol{x}) = \left|\frac{w_4(\boldsymbol{x})}{n} - \frac{1}{2}\right|$. An n-nucleotide oligo is said to be balanced if $\alpha(\boldsymbol{x}) = 0$. Code

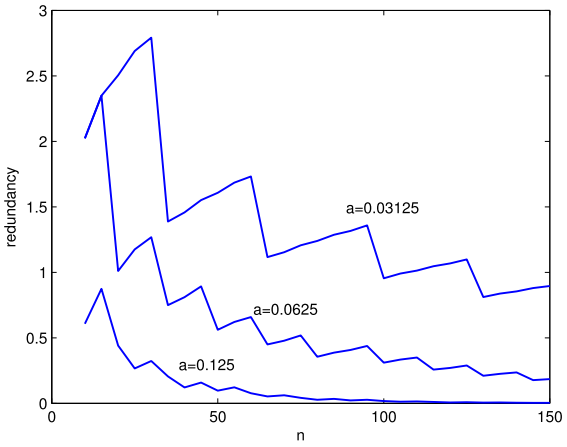


FIGURE 2. Redundancy (bits), $r_4(a, n)$, versus word length, n, with the relative unbalance, a, as a parameter. The raggedness of the curves is caused by the truncation effects in the summation in (39).

constructions for combined RLL and weak balanced codes have been published in [3], and for $m = 3$ [27], [28].

We first study the balance of sequences without and m-constraint. The number of 4-ary words of length n with balance $w = w_4(\boldsymbol{x})$, denoted by $N(w, n)$, equals

$$N_4(w, n) = \binom{n}{w} 2^n. \tag{38}$$

The number of oligo's, denoted by $N_{4,a}(n)$, of length n, whose relative unbalance, $\alpha(\boldsymbol{x}) \leq a$, is given by

$$N_{4,a}(n) = \sum_{|\frac{w}{n} - \frac{1}{2}| < a} N_4(w, n) = 2^n \sum_{|\frac{w}{n} - \frac{1}{2}| < a} \binom{n}{w}. \tag{39}$$

The redundancy of 4-ary nearly balanced strands, denoted by $r_4(a, n)$, equals

$$r_4(a, n) = \log_2 \frac{4^n}{N_{4,a}(n)}. \tag{40}$$

Figure 2 shows examples of computations of the redundancy, $r_4(a, n)$, versus n with the relative unbalance, a, as a parameter. The raggedness of the curves is caused by the truncation effects in the summation in (39). The distribution for asymptotically large n of $N_4(w, n)$ versus w is approximately Gaussian shaped, that is

$$N_4(w, n) \approx \mathcal{G}\left(w; \frac{n}{2}, \frac{n}{4}\right) 4^n, \quad n \gg 1, \tag{41}$$

so that the redundancy equals

$$r_{4,a}(n) \approx -\log_2[1 - 2Q(2a\sqrt{n})], \quad n \gg 1, \tag{42}$$

where the Q-function is defined by

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{u^2}{2}} du. \tag{43}$$

We now study q-ary sequences with both an m-constraint and a given weight w. As in Construction 1, let the quaternary word $\boldsymbol{x} = (x_1, \ldots, x_n)$, $x_i \in \mathcal{Q}$, be written as $\boldsymbol{x} = \boldsymbol{y} + 2\boldsymbol{z}$, where the constituting elements y_i and $z_i \in \{0, 1\}$. If the binary sequence $\boldsymbol{z}$ is m-constrained and has weight $w = w_2(\boldsymbol{z})$, then $\boldsymbol{x}$ is m-constrained and it has weight $w_4(\boldsymbol{z}) = w$. Using (11), (24), and (31), we obtain for $n \gg 1$, that the redundancy of q-ary sequences with combined RLL and balance constraints, denoted by $r_{q,a}(m, n)$, equals

$$r_{q,a}(m, n) \approx r_q(m, n) - \log_2 \left[1 - 2Q\left(2a\sqrt{\frac{n}{\gamma_q(m)}}\right)\right]. \tag{44}$$

A numerical analysis of the above expression shows that the redundancy difference due to the balance (right hand) term is around 0.5-1 bit for $m = 2$. For larger values of the homopolymer run m the extra redundancy is negligible for $n > 10$. The redundancy difference, $r_2(m, n) - r_4(m, n)$, due to the imposed runlength constraint is much larger for $n > 10$ than the redundancy due the balance constraint.

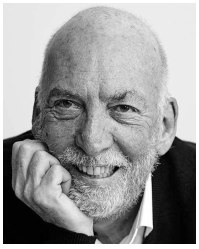
IV. CONCLUSION

We have compared two coding approaches for constraint-based coding of DNA strings. In the first approach, an intermediate, 'binary', coding step is used, while in the second approach we 'directly' translate source data into constrained quaternary sequences. The binary approach is attractive as it yields a lower complexity of encoding and decoding look-up tables. The redundancy of the binary approach is higher than that of the quaternary approach for generating combined weight and run-length constrained sequences. The redundancy difference is small for larger values of the maximum homopolymer run. We have found exact and approximate expressions for the number of binary and quaternary sequences with combined weight and run-length constraints.

REFERENCES

[1] G. M. Church, Y. Gao, and S. Kosuri, "Next-generation digital information storage in DNA," *Science*, vol. 337, no. 6102, p. 1628, Sep. 2012.
[2] M. Blawat, K. Gaedke, I. Hutter, X. Cheng, B. Turczyk, S. Inverso, B. W. Pruitt, and G. M. Church, "Forward error correction for DNA data storage," in *Proc. Int. Conf. Comput. Sci. (ICCS)*, vol. 80, 2016, pp. 1011–1022.
[3] Y. Erlich and D. Zielinski, "DNA fountain enables a robust and efficient storage architecture," *Science*, vol. 355, no. 6328, pp. 950–954, Mar. 2017.
[4] J. Koch, S. Gantenbein, K. Masania, W. J. Stark, Y. Erlich, and R. N. Grass, "A DNA-of-things storage architecture to create materials with embedded memory," *Nature Biotechnol.*, vol. 38, no. 1, pp. 39–43, Jan. 2020.
[5] Y. Wang, M. Noor-A-Rahim, J. Zhang, E. Gunawan, Y. L. Guan, and C. L. Poh, "High capacity DNA data storage with variable-length oligonucleotides using repeat accumulate code and hybrid mapping," *J. Biol. Eng.*, vol. 13, no. 1, p. 89, Dec. 2019.
[6] L. Ceze, J. Nivala, and K. Strauss, "Molecular digital data storage using DNA," *Nature Rev. Genet.*, vol. 20, no. 8, pp. 456–466, Aug. 2019.
[7] J. Bornholt, R. Lopez, D. M. Carmean, L. Ceze, and G. Seelig, "A DNA-based archival storage system," *ACM SIGOPS Oper. Syst. Rev.*, vol. 50, pp. 637–649, 2016.
[8] 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 Biol.*, vol. 14, no. 5, p. R51, 2013.
[9] K. W. Cattermole, "Principles of digital line coding," *Int. J. Electron.*, vol. 55, pp. 3–33, Jul. 1983.
[10] K. A. Schouhamer Immink and K. Cai, "Design of capacity-approaching constrained codes for DNA-based storage systems," *IEEE Commun. Lett.*, vol. 22, no. 2, pp. 224–227, Feb. 2018.
[11] Y.-S. Kim and S.-H. Kim, "New construction of DNA codes with constant-GC contents from binary sequences with ideal autocorrelation," in *Proc. IEEE Int. Symp. Inf. Theory Process.*, Jul. 2011, pp. 1569–1573.

[12] Y. M. Chee and S. Ling, "Improved lower bounds for constant GC-content DNA codes," *IEEE Trans. Inf. Theory*, vol. 54, no. 1, pp. 391–394, Jan. 2008.
[13] K. A. Schouhamer Immink and K. Cai, "Efficient balanced and maximum homopolymer-run restricted block codes for DNA-based data storage," *IEEE Commun. Lett.*, vol. 23, no. 10, pp. 1676–1679, Oct. 2019.
[14] K. A. S. Immink, *Codes for Mass Data Storage Systems*, 2nd ed. Eindhoven, The Netherlands: Shannon Foundation, 2004.
[15] S. W. MacLauhlin, J. Luo, and Q. Xie, "On the capacity of M-ary Runlength-limited codes," *IEEE Trans. Inf. Theory*, vol. 41, no. 5, pp. 1508–1511, Sep. 1995.
[16] M. W. Marcellin and H. J. Weber, "Two-dimensional modulation codes," *IEEE J. Sel. Areas Commun.*, vol. 10, no. 1, pp. 254–266, Jan. 1992.
[17] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948.
[18] P. Flajolet and R. Sedgewick, *Analytic Combinatorics*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[19] S. M. Hossein, T. Yazdi, H. M. Kiah, and O. Milenkovic, "Weakly mutually uncorrelated codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 2649–2653.
[20] V. Taranalli, H. Uchikawa, and P. H. Siegel, "Error analysis and inter-cell interference mitigation in multi-level cell flash memories," in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, U.K., Jun. 2015, pp. 271–276.
[21] K. J. Kerpez, A. Gallopoulos, and C. Heegard, "Maximum entropy charge-constrained run-length codes," *IEEE J. Sel. Areas Commun.*, vol. 10, no. 1, pp. 242–253, Jan. 1992.
[22] V. Braun and K. A. Schouhamer Immink, "An enumerative coding technique for DC-free runlength-limited sequences," *IEEE Trans. Commun.*, vol. 48, no. 12, pp. 2024–2031, Dec. 2000.
[23] O. F. Kurmaev, "Constant-weight and constant-charge binary run-length limited codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 7, pp. 4497–4515, Jul. 2011.
[24] F. Paluncic and B. T. J. Maharaj, "Using bivariate generating functions to count the number of balanced runlength-limited words," in *Proc. GLOBECOM - IEEE Global Commun. Conf.*, Singapore, Dec. 2017, pp. 4–8.
[25] D. Limbachiya, M. K. Gupta, and V. Aggarwal, "Family of constrained codes for archival DNA data storage," *IEEE Commun. Lett.*, vol. 22, no. 10, pp. 1972–1975, Oct. 2018.
[26] N. J. A. Sloane. (2019). *The On-Line Encyclopedia of Integer Sequences*. [Online]. Available: http://oeis.org
[27] Y. Wang, M. Noor-A-Rahim, E. Gunawan, Y. L. Guan, and C. L. Poh, "Construction of bio-constrained code for DNA data storage," *IEEE Commun. Lett.*, vol. 23, no. 6, pp. 963–966, Jun. 2019.
[28] W. Song, K. Cai, M. Zhang, and C. Yuen, "Codes with run-length and GC-content constraints for DNA-based data storage," *IEEE Commun. Lett.*, vol. 22, no. 10, pp. 2004–2007, Oct. 2018.



KEES A. SCHOUHAMER IMMINK (Life Fellow, IEEE) is currently a Founder and the President of Turing Machines Inc., an innovative start-up focused on coding and signal processing for DNA-based storage. He received the 2017 IEEE Medal of Honor for his for pioneering contributions to video, audio, and data recording technology, the Knighthood, in 2000, the Personal Emmy Award, in 2004, the 1999 Audio Engineering Society's (AES) Gold Medal, the 2004 SMPTE Progress Medal, the 2014 Eduard Rhein Prize for Technology, and the 2015 IET Faraday Medal. He received an Honorary Doctorate from the University of Johannesburg, in 2014. He was inducted into the Consumer Electronics Hall of Fame, elected into the Royal Netherlands Academy of Arts and Sciences, and the (US) National Academy of Engineering. He has served the profession as a Governor for the IEEE Information Theory and Consumer Electronics Societies and the President for the Audio Engineering Society.

KUI CAI (Senior Member, IEEE) received the B.E. degree in information and control engineering from Shanghai Jiao Tong University, Shanghai, China, and the joint Ph.D. degree in electrical engineering from the Technical University of Eindhoven, The Netherlands, and the National University of Singapore. She is currently an Associate Professor with the Singapore University of Technology and Design (SUTD). Her main research interests are in the areas of coding theory, information theory, signal processing for various data storage systems, and digital communications. She received the 2008 IEEE Communications Society Best Paper Award in Coding and Signal Processing for Data Storage. She has served as the Vice-Chair (Academia) for the IEEE Communications Society and the Data Storage Technical Committee (DSTC), from 2015 to 2016.