Coding schemes for locally balanced constraints

Chen Wang,* **Ziyang Lu**,* **Zhaojun Lan**,† **Gennian Ge**,† and **Yiwei Zhang***
*Key Laboratory of Cryptologic Technology and Information Security of Ministry of Education,
School of Cyber Science and Technology, Shandong University, Qingdao, Shandong, 266237, China
†School of Mathematical Sciences, Capital Normal University, Beijing 100048, China
{cwang2021, zylu}@mail.sdu.edu.cn, zjlan@cnu.edu.cn, gnge@zju.edu.cn, ywzhang@sdu.edu.cn

***Abstract*—Motivated by applications in DNA-based storage, we study explicit encoding and decoding schemes of binary strings satisfying locally balanced constraints, where the (ℓ, δ)-locally balanced constraint requires that the weight of any consecutive substring of length ℓ is between $\frac{\ell}{2} - \delta$ and $\frac{\ell}{2} + \delta$. In this paper we present coding schemes for the strongly locally balanced constraints and the locally balanced constraints, respectively. Moreover, we introduce an additional result on the linear recurrence formula of the number of binary strings which are $(6, 1)$-locally balanced, as a further attempt to both capacity characterization and new coding strategies for locally balanced constraints.**

I. Introduction

With the rapid development of DNA synthesis and DNA sequencing technology, DNA storage is becoming a promising direction for future data storage [1]. Error-correcting codes for DNA storage differ from traditional coding theory for communications or current storage mediums in many aspects. In particular, specific synthesis and sequencing methods and the biochemical properties of DNA strings bring in many additional constraints on the codewords. Most common constraints are either *run-length-limited (RLL) constraints* which limit the length of consecutive repeated symbols, or *global GC-content constraints* which require that the percentage of guanine (G) and cytosine (C) in a DNA string must be bounded by a given interval. Constructing codes with RLL constraints and/or global GC-content constraints is not a new topic and it belongs to the well-established field of constrained coding theory, see for example, [2], [3]. Constrained coding problems also have applications in fields other than DNA storage. For example, constrained codes are extremely useful in communication and storage systems which require simultaneous energy and information transfer [4].

In addition to the run-length and global GC-content constraints, during DNA storage some PCR amplification techniques require local GC-content constraints on DNA strings [5]. That is, within each consecutive substring of a given length, the percentage of guanine and cytosine is also bounded. Motivated by this application, Gabrys et al. [6] considered the binary case and proposed the so-called *locally balanced constraints* and *strongly locally balanced constraints*, where the (ℓ, δ)-locally balanced constraint requires that in a binary string of length n any consecutive substring of length ℓ should have weight between $\frac{\ell}{2}-\delta$ and $\frac{\ell}{2}+\delta$. In [6] the authors studied the capacity of binary strings meeting such constraints via a spectral graph theory method. While some capacity results were given in [6], explicit constructions of codes achieving the capacity and related encoding and decoding algorithms were not considered. The explicit coding schemes, when ℓ and δ are fixed constants, are the main objective of this paper.

It should be noted that another related work by Nguyen et al. [7] considered the coding schemes for locally balanced constraints (in the name of sliding window-constrained codes). However, their results hold only when $\ell = \Omega(\log n)$ and $\delta = \Theta(\ell)$ and their sequence replacing techniques cannot be easily generalized to the case when ℓ and δ are fixed constants.

The rest of this paper is organized as follows. In Section II we introduce the definitions and related results. Two coding schemes are introduced in Section III for the strongly locally balanced constraints and another scheme is introduced in Section IV for the locally balanced constraints. As a further attempt to the capacity characterization and new coding strategies, an additional counting result specifically targeted at the $(6, 1)$-locally balanced constraint is presented in Section V. Finally, Section VI concludes the paper.

II. Preliminaries

Let $\Sigma = \{0, 1\}$. For a sequence $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \Sigma^n$, its Hamming weight is denoted as $wt(\boldsymbol{x})$. Given integers ℓ and i, where $1 \leq i \leq n - \ell + 1$, a consecutive subword of length ℓ starting at the i-th coordinate of $\boldsymbol{x}$ is denoted as $\boldsymbol{x}[i; \ell]$, i.e., $x[i; \ell] = (x_i, x_{i+1}, \ldots, x_{i+\ell-1})$.

Definition 1. *Given a positive even integer ℓ and a positive integer δ, a word $\boldsymbol{x} \in \Sigma^n$ is said to be (ℓ, δ)-locally balanced if for all $1 \leq i \leq n - \ell + 1$, it holds that*

$$\frac{\ell}{2} - \delta \leq wt(\boldsymbol{x}[i; \ell]) \leq \frac{\ell}{2} + \delta.$$

Definition 2. *Given a positive even integer ℓ and a positive integer δ, a word $\boldsymbol{x} \in \Sigma^n$ is said to be strongly (ℓ, δ)-locally balanced, if $\boldsymbol{x}$ is (ℓ', δ)-locally balanced for every even integer $\ell' \geq \ell$.*

Definition 3. *Let $\Sigma^n(\ell, \delta)$ be the set of all binary (ℓ, δ)-locally balanced words of length n. Let $\Sigma^n(\geq \ell, \delta)$ be the set*

Research supported by National Key Research and Development Program of China under Grant Nos. 2020YFA0712100, 2021YFA1001000 and 2018YFA0704703, National Natural Science Foundation of China under Grant Nos. 12001323 and 11971325, Shandong Provincial Natural Science Foundation under Grant No. ZR2021YQ46, and Beijing Scholars Program.

of all binary strongly (ℓ,δ)-locally balanced words of length n. The capacity of such two sets are

$$\mathbb{C}(\ell,\delta)=\limsup_{n\to\infty}\frac{\log(|\Sigma^n(\ell,\delta)|)}{n},$$
$$\mathbb{C}(\geq\ell,\delta)=\limsup_{n\to\infty}\frac{\log(|\Sigma^n(\geq\ell,\delta)|)}{n}.$$

In [6], Gabrys et al. proposed the definitions above and they aimed at calculating or giving bounds on the values of $\mathbb{C}(\ell,\delta)$ and $\mathbb{C}(\geq\ell,\delta)$. The calculation of $\mathbb{C}(\ell,\delta)$ can be done by analyzing a subgraph $G_{\ell,\delta}$ of the de Bruijn graph of order ℓ, induced by the (ℓ,δ)-locally balanced words of length ℓ. By the Perron-Frobenius theorem, the value of $\mathbb{C}(\ell,\delta)$ equals $\log\lambda$, where λ is the spectral radius of the adjacency matrix of $G_{\ell,\delta}$. The following table of the capacity $\mathbb{C}(\ell,\delta)$ for $4\leq\ell\leq14$ and $\delta\in\{1,2\}$ is given in [6].

TABLE I

ℓ	4	6	8	10	12	14
$\delta=1$	0.879	0.841	0.824	0.815	0.811	0.807
$\delta=2$	1	0.975	0.958	0.947	0.939	0.933

As for $\mathbb{C}(\geq\ell,\delta)$, we need the following definition.

Definition 4. *For $\boldsymbol{x}=(x_1,x_2,\ldots,x_n)\in\Sigma^n$, its running digital sum sequence $RDS(\boldsymbol{x})=(s_0,s_1,\ldots,s_n)\in\mathbb{Z}^{n+1}$ is defined as follows: $s_0=0$, and for $1\leq i\leq n$,*

$$s_i=\sum_{j=1}^{i}(-1)^{1-x_j}=2wt((x_1,\ldots,x_i))-i.$$

Moreover, let $dis(\boldsymbol{x})=max_{0\leq i\leq n}\{s_i\}-min_{0\leq i\leq n}\{s_i\}$.

In fact, s_i calculates the difference between the appearances of 1 and the appearances of 0 among the first i coordinates of $\boldsymbol{x}$ and $dis(\boldsymbol{x})$ represents the gap between the largest and smallest integers in $RDS(\boldsymbol{x})$. For example, if $\boldsymbol{x}=(1,0,0,0,0,1)$, then we have $RDS(\boldsymbol{x})=(0,1,0,-1,-2,-3,-2)$ and $dis(\boldsymbol{x})=1-(-3)=4$.

Definition 5. *For a given integer δ, a word $\boldsymbol{x}\in\Sigma^n$ is said to be a δ-RDS word if $dis(\boldsymbol{x})\leq\delta$. The set of all δ-RDS words of length n is denoted as $\Sigma^n_{RDS}(\delta)$ and the capacity of the set is*

$$\mathbb{C}_{RDS}(\delta)=\limsup_{n\to\infty}\frac{\log(|\Sigma^n_{RDS}(\delta)|)}{n}.$$

The calculation of $\mathbb{C}_{RDS}(\delta)$ is related to a well-known combinatorial problem of counting Dyck paths of bounded height. It has been completely solved in [6] that $\mathbb{C}(\geq\ell,\delta)=\mathbb{C}_{RDS}(2\delta+1)$. In particular, when $\delta=1$, $\mathbb{C}(\geq\ell,1)=\mathbb{C}_{RDS}(3)\approx0.694$.

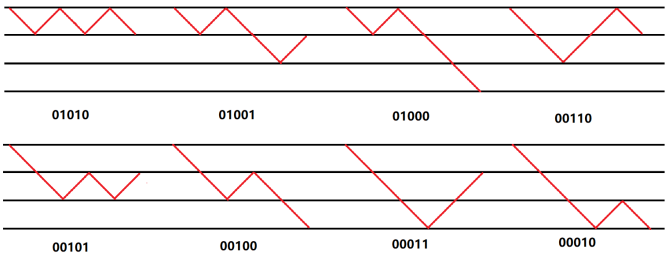
III. A CODING SCHEME FOR STRONGLY LOCALLY BALANCED CONSTRAINTS

In this section we consider coding schemes for strongly locally balanced constraints. That is, we want to find an encoding scheme $\mathcal{E}:\Sigma^k\to\Sigma^n(\geq\ell,\delta)$ with code rate k/n as close to the capacity as possible, together with its corresponding decoding algorithm $\mathcal{D}$. For simplicity, we only consider the case $\ell=4$ and $\delta=1$. The ideas behind the scheme for $\ell=4$ and $\delta=1$ can be naturally generalized to arbitrary ℓ and δ.

We want a coding scheme with rate approaching the capacity $\mathbb{C}(\geq4,1)=0.694$. First, note that by encoding 0 as 01 and 1 as 10, one can get a trivial encoding scheme $\mathcal{E}:\Sigma^k\to\Sigma^{2k}(\geq4,1)$ with code rate 0.5. We explain our idea by the toy example with code rate 0.6 as follows.

Example 1. *We construct $\mathcal{E}:\Sigma^{3k}\to\Sigma^{5k}(\geq4,1)$ and thus its rate is $3/5=0.6$. The encoder divides $\boldsymbol{x}\in\Sigma^{3k}$ into k blocks of length 3 and sequentially encodes each block into a string of length 5. To ensure that the final output $\boldsymbol{y}$ is strongly $(4,1)$-locally balanced, it suffices to make sure that $dis(\boldsymbol{y})\leq3$. Since $\boldsymbol{y}$ and $RDS(\boldsymbol{y})$ are one-to-one correspondence, we can describe $RDS(\boldsymbol{y})=(s_0,s_1,\ldots,s_{5k})$ instead of $\boldsymbol{y}$, and WLOG we fix the range of each entry of $RDS(\boldsymbol{y})$ within the interval $[-1,2]$.*

Suppose that we are in the position of encoding the block $\{x_{3p+1},x_{3p+2},x_{3p+3}\}$, $0\leq p\leq k-1$, and the current RDS sequence has $s_{5p}=2$, i.e., at the top layer of its range. Consider the following 8 diagrams of Dyck paths, which start from layer 2 and are bounded by the interval $[-1,2]$. Each such diagram depicts the trend of the RDS sequence and can be translated into a binary string of length 5.



Thus, we may pick an arbitrary one-to-one correspondence mapping between Σ^3 and these 8 diagrams. Symmetrically, if the current RDS sequence has $s_{5p}=-1$, i.e., at the bottom layer of its range, we also have 8 proper diagrams. It is routine to check that if $s_{5p}\in\{0,1\}$, i.e., the current RDS entry is on the middle two layers, then there are more than 8 diagrams which depict proper trends of the RDS sequence within the interval $[-1,2]$ and we may choose arbitrary 8 of them and build a table-based encoding and decoding algorithm.

Essentially, the scheme above is valid since the number of bounded Dyck paths of length 5 is at least $8=2^3$, no matter in which layer the starting point lies. Motivated by this example, we proceed with the following scheme.

Definition 6. *Given an integer m, let $p(m)$ be the number of Dyck paths bounded by the interval $[-1,2]$ which starts at the layer 2 (symmetrically, the layer -1) and let $q(m)$ be the number of Dyck paths bounded by the interval $[-1,2]$ which starts at the layer 1 (symmetrically, the layer 0).*

Construction 1. *Given an integer s, find the minimum integer m such that $p(m) \geq 2^s$ and $q(m) \geq 2^s$. Then there is an encoding scheme $\mathcal{E} : \Sigma^{sk} \to \Sigma^{mk}(\geq 4, 1)$ for arbitrary k as follows:*

- *Divide $\boldsymbol{x} \in \Sigma^{sk}$ into k disjoint blocks of size s.*
- *Build two tables, each indicating a one-to-one map from Σ^s to the two sets of bounded Dyck paths.*
- *Sequentially encode each block of size s by observing the current RDS entry, and then select the corresponding Dyck path from the proper table.*

The final output $\boldsymbol{y}$ is strongly $(4, 1)$-locally balanced since $dis(\boldsymbol{y}) \leq 3$ always holds. The decoding algorithm is straightforward based on a table-based search. To maximize the code rate, we need to calculate $p(m)$ and $q(m)$.

Lemma 1. *For every positive integer m, $p(m) = F_{m+1}$ and $q(m) = F_{m+2}$, where $\{F_m\}$ is the Fibonacci sequence with $F_1 = 1$, $F_2 = 1$, and $F_k = F_{k-2} + F_{k-1}$ for $k \geq 3$.*

Proof: For a bounded Dyck path starting at layer 2, its next step must go to layer 1, and thus $p(m) = q(m-1)$. For a bounded Dyck path starting at layer 1, its next step goes to either layer 2 or layer 0, and thus $q(m) = p(m-1) + q(m-1)$. Thus we can deduce that $p(m) = q(m-1) = p(m-2) + q(m-2) = p(m-2) + p(m-1)$, with the initial values $p(1) = 1$ and $p(2) = 2$. Therefore, $p(m) = F_{m+1}$ and $q(m) = p(m+1) = F_{m+2}$. ∎

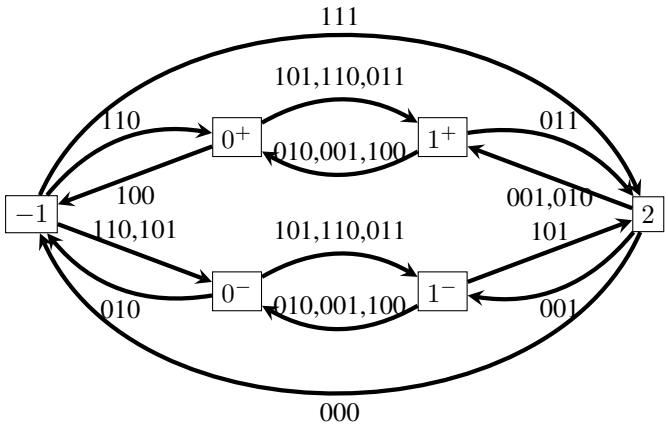
The lemma indicates that $q(m) \geq p(m)$ always holds. Therefore, for every s we only need to find the minimum integer m with $p(m) = F_{m+1} \geq 2^s$ and then we have a coding scheme with rate s/m. The rate for small s is summarized in the following table.

s	2	3	4	5	6	7	8
m	4	5	7	8	10	11	13
s/m	0.5	0.6	0.571	0.625	0.6	0.636	0.615
s	9	10	11	12	13	14	15
m	14	16	17	18	20	21	23
s/m	0.642	0.625	0.647	0.667	0.65	0.667	0.652

When m is sufficiently large, $F_m \approx (\frac{1+\sqrt{5}}{2})^m \approx 2^{0.694m}$, so Construction 1 has rate approaching the capacity 0.694, as $s \to \infty$. However, even to achieve rate 0.667 we already need $(s, m) = (12, 18)$. Since our map from Σ^s to proper Dyck paths is table-based, each table will need to store $2^{12} = 4096$ entries, which is rather impractical.

To solve this issue, we move on to a modified construction with rate 0.667 but only requires a table storing 24 entries. The scheme is illustrated by the following state transition diagram.

Construction 2. *The encoding scheme works as follows. A message $\boldsymbol{x} \in \Sigma^{2k}$ is divided into k blocks of size 2 and we sequentially encode each block into a string of length 3. Initially we have the RDS sequence starting with $s_0 = 0$, and we set our initial status as 0^+. When reading the first block x_1, x_2, the four choices are set one-to-one correspondence with the four arrows leaving the state 0^+, which are $\{101, 110, 011, 100\}$. Then recursively in each step we check the current state and then choose one out of the four arrows leaving each state. After encoding all $2k$ information symbols we arrive at a sequence of length $3k$. Finally, we add a '1' if we end in the state 0^+, 1^+ and -1. Otherwise we add a '0' if we end in the state 0^-, 1^-, and 2. Note that in this way the added bit still guarantees that $s_{3k+1} \in [-2, 1]$ and thus does not violate the strongly locally balanced constraint.*



The decoding scheme works as follows. Given the encoded string $\boldsymbol{y}$ of length $3k+1$, first check the entry s_{3k} in the RDS sequence. Together with the last bit, we uniquely determine which state we end with. Then the decoding can be done by reading every three bits backwards in a table-based way according to the transition diagram.

For example, suppose we pick the following table for our coding scheme:

Status	00	01	10	11
-1	110 (to 0^+)	110 (to 0^-)	101	111
0^+	101	110	011	100
0^-	101	110	011	010
1^+	010	001	100	011
1^-	010	001	100	101
2	000	010	001 (to 1^+)	001 (to 1^-)

Then $\boldsymbol{x} = (10{-}01{-}11{-}01)$ is encoded as $\boldsymbol{y} = 011{-}001{-}100{-}110{-}0$. To decode back, first note that in $RDS(\boldsymbol{y})$ we have $s_{12} = 0$ and the last bit is 0, so the final state is 0^-. The edge labelled '110' into the state 0^- comes from the state -1 and thus $(x_7, x_8) = 01$. The edge labelled '100' into the state -1 comes from the state 0^+ and thus $(x_5, x_6) = 11$. Repeating this process and then we finally decode $\boldsymbol{x} = (10{-}01{-}11{-}01)$.

To sum up, we have proved the following:

Theorem 1. *The coding scheme based on the state transition diagram above is a coding scheme $\mathcal{E} : \Sigma^{2k} \to \Sigma^{3k+1}(\geq 4, 1)$ and thus its rate is $\frac{2k}{3k+1} \to 0.667$ as $k \to \infty$.*

We close this section by discussing the key feature of the state transition diagram. Each state has out-degree 4, which guarantees the encoding process. The incoming arrows into each state have distinct labels, which guarantees the decoding process. In fact, Construction 2 could be further

generalized to $\Sigma^{11k} \to \Sigma^{16k+1}(\geq 4, 1)$ with rate approaching $11/16 = 0.687$ and $\Sigma^{20k} \to \Sigma^{29k+1}(\geq 4, 1)$ with rate approaching $20/29 = 0.690$. Compared with Construction 1, Construction 2 approaches 0.694 faster. Moreover, to achieve the same rate, the table size of Construction 2 will be much smaller than Construction 1. Due to lack of space, we postpone the details to a future journal version of this paper.

IV. A CODING SCHEME FOR LOCALLY BALANCED CONSTRAINTS

Now we consider (ℓ, δ)-locally balanced constraints. Note that the two constructions above for strongly $(4, 1)$-locally balanced constraints also work for $(\ell, 1)$-locally balanced constraints for arbitrary even integer ℓ. However, the gap between the rate 0.667 and the capacity results as shown in Table I is too large to be negligible. In this section, we attempt to propose a general algorithm to find a coding scheme $\Sigma^k \to \Sigma^n(\ell, \delta)$ with rate k/n as close to the capacity as possible.

Given ℓ, define a directed graph G_m as follows, for all $m \geq \ell - 1$. The vertex set of G_m is Σ^m, the set of all binary strings of length m. An arrow from $\boldsymbol{x}$ to $\boldsymbol{y}$ exists if and only if their concatenation sequence $\boldsymbol{xy}$ is an (ℓ, δ)-locally balanced string of length $2m$. The next algorithm aims at finding the largest integer $s = s(m)$, such that there exists a subgraph of G_m where every vertex has out degree at least 2^s.

Algorithm 1

Input: m

Output: $\mathcal{G}_m$, s

Initially: set $s := \lfloor \log \Delta \rfloor$, where Δ is the maximum degree of G_m.

1: Build the graph G_m.
2: Delete all vertices with degree less than 2^s.
3: Check the remaining graph. If it is an empty graph then go to Step 4. If it is nonempty and the minimum degree is less than 2^s, repeat Step 2. Otherwise, go to Step 5.
4: Set $s := s - 1$ and go back to Step 1.
5: Output the current subgraph as $\mathcal{G}_m$, in which every vertex has degree at least 2^s. Output the current value of s.

Building on this algorithm, a coding scheme for (ℓ, δ)-locally balanced constraints is as follows.

Construction 3. *For every $m \geq \ell - 1$, implement Algorithm 1 to find the corresponding $s(m)$. Compare the values of $s(m)/m$ and find the largest one as s'/m'. The output digraph from Algorithm 1 with m' as the input is denoted as $\mathcal{G}_{m'}$.*

Given any message $\boldsymbol{x} \in \Sigma^{ks'}$ for arbitrary k, divide $\boldsymbol{x}$ into k disjoint blocks of size s' and sequentially encode each block of size s' into a string of length m'. The encoding starts with a predetermined one-to-one map between $\Sigma^{s'}$ to a subset of the vertex set of $\mathcal{G}_{m'}$. Then, the encoding of the next block is based on a predetermined one-to-one map between $\Sigma^{s'}$ to a subset of the out-going edges from the current vertex. Recursively do the encoding and finally the output is the concatenation of all the strings length m', altogether a string of length km'.

Via computer search, the rate of Construction 3 for $4 \leq \ell \leq 14$ and $\delta \in \{1, 2\}$ (and the corresponding values of s' and m') is summarized as follows.

ℓ	4	6	8
$\delta = 1$	11/13=0.846	12/15=0.8	10/13=0.769
$\delta = 2$	1	14/15=0.933	13/14=0.929
ℓ	10	12	14
$\delta = 1$	11/15=0.733	11/15=0.733	11/15=0.733
$\delta = 2$	8/9=0.889	12/14=0.857	12/14=0.857

V. MORE ON THE $(6, 1)$-LOCALLY BALANCED CONSTRAINT

In this section we derive the linear recurrence relation on the size of $\Sigma^n(6, 1)$. There are mainly two motivations for analyzing this formula.

Firstly, while theoretically computing the capacity $\mathbb{C}(\ell, \delta)$ can be done by the spectral graph theory approach as explained in [6], the exponentially growing size of the adjacency matrix makes all known algorithms of spectral radius impractical. This is also why Table I ends with $\ell = 14$. For general ℓ and δ, determining $\mathbb{C}(\ell, \delta)$ seems to be a very difficult problem. We want to try to find the linear recurrence relation and thus the exact formula on the size of $\Sigma^n(\ell, \delta)$, or try to find linear recurrence inequalities which can provide upper or lower bounds. The formula for the size of $\Sigma^n(4, 1)$ can be easily solved and thus the next step starts with the case $\Sigma^n(6, 1)$.

Secondly, an explicit formula might lead to a coding scheme based on recursive enumeration techniques, based on a partial order of the set of proper codewords, which is indeed the case for coding schemes under run-length limited constraints, see for example [8]. To be honest, it seems that our formula below for $\Sigma^n(6, 1)$ does not lead to an explicit coding scheme at this moment, but we believe that it might shed light on some new strategies for coding schemes other than Construction 3.

Now we present the main result of this section.

Theorem 2. *Let f_n be the size of $\Sigma^n(6, 1)$, the set of $(6, 1)$-locally balanced binary words of length n. Then we have the following linear recurrence relation*

$$f_{n+12} = f_{n+11} + f_{n+10} + f_{n+9} - f_{n+6} - f_{n+4} - f_{n+3} + f_n.$$

The proof of this theorem is broken into several lemmas. For any binary string $\boldsymbol{z}$ let $f_n(\boldsymbol{z})$ be the number of words in $\Sigma^n(6, 1)$ with $\boldsymbol{z}$ as a prefix. Given integers $0 \leq t \leq s$, let $X_n(6, 1; s, t)$ be the words $\boldsymbol{x} \in \Sigma^n(6, 1)$ such that $wt(\boldsymbol{x}[1; s]) = t$ and let $f_n(s, t) = |X_n(6, 1; s, t)|$.

Lemma 2. $f_n = f_{n+3}(000) + f_{n+3}(111) = f_{n+4}(1000) + f_{n+4}(0111)$.

Proof: For any $\boldsymbol{x} \in \Sigma^n(6, 1)$, consider the weight of its first three entries and then $\Sigma^n(6, 1)$ is a disjoint partition of $X_n(6, 1; 3, 0), X_n(6, 1; 3, 1), X_n(6, 1; 3, 2), X_n(6, 1; 3, 3)$ and thereby we have $f_n = f_n(3, 0) + f_n(3, 1) + f_n(3, 2) + f_n(3, 3)$.

For any $\boldsymbol{x} \in X_n(6,1;3,0)$, $\boldsymbol{x} = (0,0,0,x_4,x_5,x_6,\ldots,x_n)$ and $\{x_4,x_5,x_6\}$ must contain at least two 1's. Let $\boldsymbol{x}' \triangleq 111\boldsymbol{x}$. Then we have $wt(\boldsymbol{x}'[1;6]) = 3$, $wt(\boldsymbol{x}'[2;6]) \in \{2,3\}$ and $wt(\boldsymbol{x}'[3;6]) \in \{2,3\}$. Moreover, $\boldsymbol{x}'[i;6] = \boldsymbol{x}[i-3;6]$ for $i \geq 4$ and is thereby $(6,1)$-locally balanced. Thus we have $\boldsymbol{x}' \in \Sigma^{n+3}(6,1)$. Similarly, it is also routine to check that for any $\boldsymbol{x} \in X_n(6,1;3,1)$, it also holds that $111\boldsymbol{x} \in \Sigma^{n+3}(6,1)$. Meanwhile, for any sequence $111\boldsymbol{x} \in \Sigma^{n+3}(6,1)$, delete the prefix 111 and the remaining $\boldsymbol{x}$ must satisfy $\boldsymbol{x} \in \Sigma^n(6,1)$ and its first three entries has weight either 2 or 3. To sum up, we have proved $f_{n+3}(111) = f_n(3,0) + f_n(3,1)$. Symmetrically we have $f_{n+3}(000) = f_n(3,2) + f_n(3,3)$ and thus $f_n = f_{n+3}(000) + f_{n+3}(111)$. The proof of $f_n = f_{n+4}(1000) + f_{n+4}(0111)$ is similar and thus omitted. ∎

Lemma 3. $f_{n+2}(110) = f_n(0) - f_n(0111)$ *and* $f_{n+2}(001) = f_n(1) - f_n(1000)$.

Proof: For any $\boldsymbol{x} \in \Sigma^n(6,1)$ starting with 0, $11\boldsymbol{x} \notin \Sigma^{n+2}(6,1)$ if and only if $\boldsymbol{x}$ starts with 0111. Thereby $f_{n+2}(110) = f_n(0) - f_n(0111)$. Similarly we have $f_{n+2}(001) = f_n(1) - f_n(1000)$. ∎

Lemma 4. $f_{n+4}(0000) = f_n(11)$ *and* $f_{n+4}(1111) = f_n(00)$.

Proof: Any $\boldsymbol{x} \in \Sigma^{n+4}(6,1)$ starting with 0000 must start with 000011. For any $\boldsymbol{x} \in \Sigma^n(6,1)$ starting with 11, we can add the prefix 0000 and the resultant string is still $(6,1)$-locally balanced. Therefore $f_{n+4}(0000) = f_n(11)$. Similarly we have $f_{n+4}(1111) = f_n(00)$. ∎

Lemma 5. $f_{n+3} - f_{n+2} - f_{n+1} - f_n = -f_{n+1}(0000) - f_{n+1}(1111) - f_n(000) - f_n(111)$.

Proof: First consider the difference $f_{n+3} - f_{n+2}$. For any word $\boldsymbol{x} \in \Sigma^{n+2}(6,1)$, $0\boldsymbol{x} \in \Sigma^{n+3}(6,1)$ if and only if the first five entries of $\boldsymbol{x}$ has weight $2,3,4$, and $1\boldsymbol{x} \in \Sigma^{n+3}(6,1)$ if and only if the first five entries of $\boldsymbol{x}$ has weight $1,2,3$. Therefore, $f_{n+3} - f_{n+2} = f_{n+2}(5,2) + f_{n+2}(5,3)$.

Next, for $\boldsymbol{x} \in X_{n+2}(6,1;5,2)$, by puncturing the first entry we get a string in $\boldsymbol{y} \in X_{n+1}(6,1;4,1) \cup X_{n+1}(6,1;4,2)$. On the other hand, for any $\boldsymbol{y} \in X_{n+1}(6,1;4,1) \cup X_{n+1}(6,1;4,2)$ there is a unique way to add a bit in the beginning to get $\boldsymbol{x} \in X_{n+2}(6,1;5,2)$. Therefore, $f_{n+2}(5,2) = f_{n+1}(4,1) + f_{n+1}(4,2)$. Similarly, $f_{n+2}(5,3) = f_{n+1}(4,3) + f_{n+1}(4,2)$. Following a similar analysis, we also have $f_{n+1}(4,2) = f_n(3,1) + f_n(3,2)$.

Thus we have the following computation:

$$\begin{aligned}
&f_{n+3} - f_{n+2} - f_{n+1} - f_n \\
&= f_{n+2}(5,2) + f_{n+2}(5,3) - f_{n+1} - f_n \\
&= f_{n+1}(4,1) + 2f_{n+1}(4,2) + f_{n+1}(4,3) - f_{n+1} - f_n \\
&= -f_{n+1}(0000) - f_{n+1}(1111) + f_n(3,1) + f_n(3,2) - f_n \\
&= -f_{n+1}(0000) - f_{n+1}(1111) - f_n(000) - f_n(111).
\end{aligned}$$

Thus the lemma follows. ∎

With these preparations, now we are ready for the proof of the linear recurrence relation.

Proof of Theorem 2: We have the following deduction.

$$\begin{aligned}
&f_{n+4}(0111) + f_{n+4}(1000) = f_n && (1)\\
\Rightarrow &f_{n+6}(110) + f_{n+6}(001) = f_{n+4} - f_n && (2)\\
\Rightarrow &f_{n+6}(110) + f_{n+6}(001) + f_{n+6}(111) + f_{n+6}(000) \\
&= f_{n+4} + f_{n+3} - f_n && (3)\\
\Rightarrow &f_{n+9}(000) + f_{n+9}(111) \\
&= f_{n+6}(01) + f_{n+6}(10) + f_{n+4} + f_{n+3} - f_n && (4)\\
\Rightarrow &f_{n+10}(0000) + f_{n+10}(1111) + f_{n+9}(000) + f_{n+9}(111) \\
&= f_{n+6} + f_{n+4} + f_{n+3} - f_n && (5)
\end{aligned}$$

Equation (1) follows Lemma 2.

Equation (2) follows Lemma 2 and Lemma 3, which implies that $f_{n+6}(110) + f_{n+6}(001) = f_{n+4}(1) + f_{n+4}(0) - f_{n+4}(1000) - f_{n+4}(0111) = f_{n+4} - f_n$.

Equation (3) follows Lemma 2, which implies that $f_{n+6}(000) + f_{n+6}(111) = f_{n+3}$.

Equation (4) follows Lemma 2 and Equation (3), which implies that $f_{n+9}(000) + f_{n+9}(111) = f_{n+6} = f_{n+6}(10) + f_{n+6}(01) + f_{n+6}(11) + f_{n+6}(00) = f_{n+6}(10) + f_{n+6}(01) + f_{n+4} + f_{n+3} - f_n$.

Equation (5) follows Lemma 4, which implies that $f_{n+10}(0000) + f_{n+10}(1111) = f_{n+6}(11) + f_{n+6}(00)$.

Finally, using Equation (5) and Lemma 5, we can deduce that $f_{n+12} - f_{n+11} - f_{n+10} - f_{n+9} = -f_{n+10}(0000) - f_{n+10}(1111) - f_{n+9}(000) - f_{n+9}(111) = -f_{n+6} - f_{n+4} - f_{n+3} + f_n$, and thus the recurrence relation holds. ∎

Note that by solving the recurrence relation, one can find that $f(n) \approx 1.791^n$ and thus $\mathbb{C}(6,1) = 0.841$, which is the same as the result from the spectral graph theory method as expected. take the log of f(n), by definition; n >>

A final remark is that unfortunately the recurrence relation does not trivially lead to a coding scheme based on enumerations as in the case of run-length limited constrained codes. However, we believe that the exact formula might shed light on some new coding strategies. Moreover, finding the recurrence relation (or proper forms of inequalities) on the size of $\Sigma^n(\ell,\delta)$ for general (ℓ,δ) is an interesting and challenging direction.

VI. Conclusion

In this paper we further study the locally balanced constraints proposed in [6]. We propose two coding schemes for strongly locally balanced constraints. In particular, for the strongly $(4,1)$-locally balanced constraint we have a scheme with rate 0.667 close to the capacity limit 0.694, based on a simple look-up table. For locally balanced constraints we propose an algorithm to find a table-based coding scheme, with rate as close to the capacity limit as possible. Moreover, we give an additional result on the linear recurrence relation on the size of $\Sigma^n(6,1)$, and new coding strategies based on such formulas are considered for future research.

REFERENCES

[1] Potomac Institute for Policy Studies, "The future of DNA data storage", Arlington, VA, available at https://potomacinstitute.org/reports/43-pips-reports/189-the-future-of-dna-data-storage, 2018.

[2] D. E. Knuth, "Efficient balanced codes", *IEEE. Trans. Inf. Theory*, vol. 32, no. 1, pp. 51-53, 1986.

[3] B. H. Markus, R. M. Roth, and P. H. Siegel, "An introduction to coding for constrained systems", Lecture notes, 2001.

[4] L. R. Varshney, "Transporting information and energy simultaneously", *in Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp. 1612-1616, 2008.

[5] Y. Benita, R. S. Oosting, M. C. Lok, M. J. Wise1 and I. Humphery-Smith, "Regionalized GC content of template DNA as a predictor of PCR success", *Nucleic acids research*, vol. 31, no. 16, pp. e99-e99, 2003.

[6] R. Gabrys, H. M. Kiah, A. Vardy, E. Yaakobi, and Y. Zhang, "Locally balanced constraints", *in Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, pp. 664-669, 2020.

[7] T. T. Nguyen, K. Cai, and K. A. S. Immink, "Efficient design of subblock energy-constrained codes and sliding window-constrained codes", *IEEE. Trans. Inf. Theory*, vol. 67, no. 12, pp. 7914-7924, 2021.

[8] T. T. Nguyen, K. Cai, K. A. S. Immink and H. M. Kiah, "Capacity-Approaching Constrained Codes With Error Correction for DNA-Based Data Storage", *IEEE. Trans. Inf. Theory*, vol. 67, no. 8, pp. 5602-5613, 2021.