Rank-Modulation Codes for DNA Storage

Netanel Raviv[⋆,†], **Moshe Schwartz**[†], and **Eitan Yaakobi**[⋆]
[⋆]Computer Science Department, Technion – Israel Institute of Technology, Haifa 3200003, Israel
[†]Electrical and Computer Engineering, Ben-Gurion University of the Negev, Beer Sheva 8410501, Israel
netanel.raviv@gmail.com, schwartz@ee.bgu.ac.il, yaakobi@cs.technion.ac.il

Abstract

Synthesis of DNA molecules offers unprecedented advances in storage technology. Yet, the microscopic world in which these molecules reside induces error patterns that are fundamentally different from their digital counterparts. Hence, to maintain reliability in reading and writing, new coding schemes must be developed.

In a reading technique called shotgun sequencing, a long DNA string is read in a sliding window fashion, and a profile vector is produced. It was recently suggested by Kiah et al. that such a vector can represent the permutation which is induced by its entries, and hence a rank-modulation scheme arises. Although this interpretation suggests high error tolerance, it is unclear which permutations are feasible, and how to produce a DNA string whose profile vector induces a given permutation.

In this paper, by observing some necessary conditions, an upper bound for the number of feasible permutations is given. Further, a technique for deciding the feasibility of a permutation is devised. By using insights from this technique, an algorithm for producing a considerable number of feasible permutations is given, which applies to any alphabet size and any window length.

Index Terms

DNA storage, permutations codes, DeBruijn graphs.

I. Introduction

Coding for DNA storage devices has gained increasing attention lately, following a proof of concept by several promising prototypes [1], [2], [4]. Due to the high cost and technical limitations of the synthesis (i.e., writing) process, these works focused on producing short strings, but as the cost of synthesis declines, longer strings can be produced. In turn, long strings are read more accurately by a technique called *shotgun sequencing* [11], in which short substrings are read separately and reassembled together to form the original string.

The shotgun-sequencing technique has motivated the definition of the DNA storage channel [7] (see Figure 1). In a channel with alphabet Σ of size q, and window length ℓ, data is stored as a (possibly circular) string over Σ; the output of the channel is a (possibly erroneous) *histogram*, or a *profile vector* with q^ℓ entries, each containing the number of times that the corresponding ℓ-substring was observed. Errors in this channel might occur as a result of substitutions in the synthesis or sequencing processes, or imperfect coverage of reads.

In order to cope with different error patterns, several code constructions were recently suggested [7], however, much is yet to be done to obtain high error resilience and high rate. It was also suggested in [7, Sec. VIII.B] to employ a rank-modulation scheme, which has the potential for coping with *any* error pattern that does not revert the order among the entries of the profile vector.

In this scheme, the absolute values of the profile vector are ignored, and only their ranks in relation to each other are considered. That is, a given profile vector represents the permutation which is induced by its entries. Clearly, to induce a permutation the entries of the profile vector must be distinct, which is a reasonable assumption for long strings and short window length. For example, the (actual) stored string in Figure 1 represents the permutation

$$\pi = (\mathtt{AA}, \mathtt{AC}, \mathtt{CA}, \mathtt{GA}, \mathtt{AG}, \mathtt{CC}, \mathtt{CG}, \mathtt{GC}, \mathtt{GG}), \tag{1}$$

since AA has the lowest frequency, followed by AC, and so on, until reaching GG with the highest frequency. Furthermore, it is evident that the noisy output in Figure 1, due perhaps to the shotgun-sequencing process, still represents the same permutation π, and hence, this error pattern in the profile vector is correctable by using a rank-modulation scheme.

A well known tool in the analysis of strings is the DeBruijn graph G_q^ℓ, whose set of nodes is Σ^ℓ, and two nodes are connected by a directed edge if the $(\ell-1)$-suffix of the former is the $(\ell-1)$-prefix of the latter. This graph is a useful tool in the analysis of the DNA storage channel since any string over Σ induces a path in the graph, and since a (normalized) profile vector can be seen as measure on its node set. Further, we may restrict our attention to closed strings only (i.e., strings that correspond to closed paths), since in our context, they are asymptotically equivalent to their ordinary counterparts.

The material in this paper was presented in part at the IEEE International Symposium on Information Theory (ISIT 2017), Aachen, Germany, June 2017. This work was supported in part by the Israel Science Foundation under grant no. 130/14 and grant no. 1624/14.

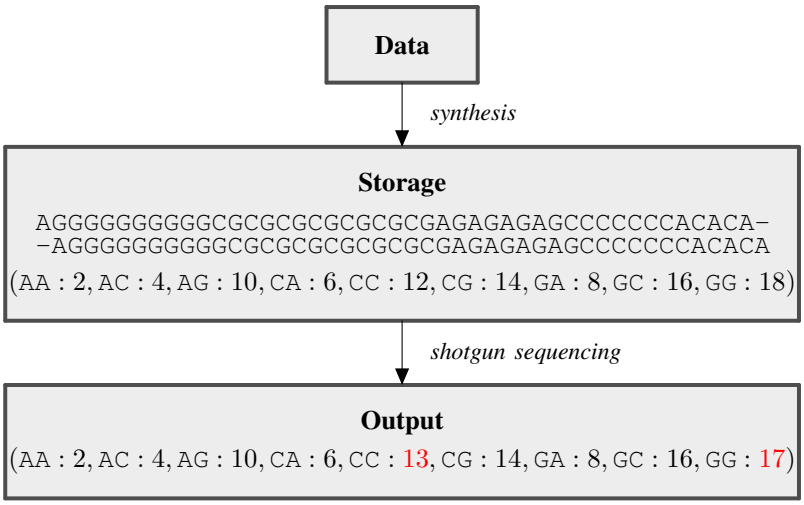

Fig. 1. An example of a transmission in the DNA storage channel with the parameters $q = 3$, $\ell = 2$, and $n = 90$. The data is encoded into a circular DNA string using a process called *synthesis*. The stored string is read using a process called *shotgun sequencing*, whose output is a (possibly erroneous) profile vector.

Due to flow conservation constraints in the DeBruijn graph [6], it is evident that not every permutation is feasible, i.e., there exist permutations that are not induced by any profile vector, and hence by any string. Consequently, [7] suggested to disregard a certain subset S of the entries in the profile vector, encode any permutation on the complement of S, and complete the entries of S to obtain flow conservation.

In this paper the feasibility question is studied in a more restrictive setting, where *all* the entries of the profile vector are considered. By formulating several necessary conditions, an upper bound is given on the number of feasible permutations, out of all permutations on q^ℓ elements. In addition, a linear-programming technique is devised to decide the feasibility of a given permutation. Using insights from this technique, an encoding algorithm for producing a large number of feasible permutations for any q and any ℓ is given. Interestingly, some of the above results rely on an interpretation of the encoding process as a *Markov chain* on the DeBruijn graph.

Finally, this problem may also be seen in a more general setting, beyond any applications for DNA storage. For example, it may be seen as a highly-restrictive variant of *constraint coding*, an area of coding theory that concerns the construction of strings with or without some prescribed substrings. This resemblance is apparent by observing that in our setting, *every* ℓ-substring is required to be more or less frequent than any other ℓ-substring.

The paper is organized as follows. In Section II we provide notation as well as formal definitions which are used throughout the paper. In Section III we prove an upper bound on the number of feasible permutations. An efficient algorithm for deciding whether a permutation is feasible is described in Section IV. We turn in Section V to devise a construction for a large set of feasible permutations, thus also providing a lower bound on their total number. An important parameter of interest is the length of the shortest string for a given feasible permutation. An upper bound on this parameter is given in Section VI. The paper is concluded in Section VII with a short discussion.

II. Preliminaries

For an alphabet $\Sigma \triangleq \{\sigma_1, \ldots, \sigma_q\}$ and a window size $\ell \in \mathbb{N}$, let G_q^ℓ be the DeBruijn graph of order ℓ over Σ. That is, the node set $V(G_q^\ell)$ is Σ^ℓ, and for any two nodes u and v in $V(G_q^\ell)$, the edge set $E(G_q^\ell)$ contains (u, v) if the $(\ell-1)$-suffix of u equals the $(\ell-1)$-prefix of v. An edge (u, v) is labeled by a string $w \in \Sigma^{\ell+1}$ whose ℓ-prefix is u and whose ℓ-suffix is v.

For a string $x \in \Sigma^n$, $x = x_0 x_1 \ldots x_{n-1}$, with $x_i \in \Sigma$, the ℓ-profile vector $p_x \in \mathbb{Z}^{q^\ell}$ (or simply, the profile vector) is a vector with q^ℓ non-negative integer entries $(p_x(w))_{w \in \Sigma^\ell}$, each of which contains the number of occurrences of the corresponding ℓ-substring in x. That is, $p_x(w) = |\{0 \le i \le n-1 \mid x_i, \ldots, x_{i+\ell-1} = w\}|$ for all $w \in \Sigma^\ell$, where indices are taken modulo $|x| = n$. The entries of p_x may be identified by either the set of nodes of G_q^ℓ or the set of edges of $G_q^{\ell-1}$, and the subscript x is omitted if clear from context.

For a given set A, let S_A be the *set of permutations on* A, i.e., the set of vectors in $A^{|A|}$ in which every element of A appears exactly once. For brevity, let $S_{q,\ell} \triangleq S_{\Sigma^\ell}$, and $S_i \triangleq S_{[i]}$ for any positive integer i, where $[i] \triangleq \{1, 2, \ldots, i\}$. Let $\pi \in S_{q,\ell}$ and $p \in \mathbb{R}^{q^\ell}$, where the coordinates of $\mathbb{R}^{q^\ell}$ are identified by the elements of Σ^ℓ, ordered lexicographically. We say that p *satisfies* π, and denote $p \vDash \pi$, if the entries of p are distinct, and their ascending order matches π, i.e., $p(w) < p(w')$ if and only if $\pi(w) < \pi(w')$, for all $w, w' \in \Sigma^\ell$. Similarly, for a string $x \in \Sigma^*$ (where Σ^* is the set of all closed strings) and a permutation $\pi \in S_{q,\ell}$ we say that x satisfies π, and denote $x \vDash \pi$, if $p_x \vDash \pi$, i.e., if the profile vector of x satisfies π.

For example, in Figure 1, the vector at the output of the channel is $p \triangleq (2,4,10,6,13,14,8,16,17) \in \mathbb{R}^9$; and since the entries of $\mathbb{R}^9$ are indexed by the elements of $\{A,C,G\}^2$, ordered lexicographically, it follows that $p \vDash \pi$ for the permutation π that was given in (1). Consequently, the string x in the **Storage** phase of Figure 1 satisfies π.

A permutation $\pi \in S_{q,\ell}$ is called *feasible* if there exists a string $x \in \Sigma^*$ such that $x \vDash \pi$, and infeasible otherwise. Similarly, for any subset $U \subseteq \Sigma^\ell$, a permutation on U is called *infeasible* if it cannot be extended to a feasible permutation in $S_{q,\ell}$. Clearly, only vectors with distinct entries can satisfy a permutation, and hence, not every string satisfies a permutation.

Another constraint on feasibility is *flow conservation*. For every $\ell \geq 2$, and any $x \in \Sigma^*$, we must have

$$\sum_{\sigma\in\Sigma} p_x(\sigma w) = \sum_{\sigma\in\Sigma} p_x(w\sigma),$$

for all $w \in \Sigma^{\ell-1}$. We call these the flow-conservation constraints, which easily follow by noting that each side of the equation equals the number of occurrences of w in x. Another view of these constraints follows by noting that any string $x \in \Sigma^*$ may be scanned using a sliding window of length ℓ, inducing cycle in the DeBruijn graph $G_q^{\ell-1}$. The flow-conservation constraints simply state that, along the cycle, the number of times we enter vertex $w \in \Sigma^{\ell-1}$ is $\sum_{\sigma\in\Sigma} p_x(\sigma w)$, must equal the number of times we exit this vertex, i.e., $\sum_{\sigma\in\Sigma} p_x(w\sigma)$. Due to flow conservation constraints in the DeBruijn graph, some permutations are infeasible, as illustrated in Figure 2.

We note that, given a profile vector p with flow conservation of order ℓ, there exists a string $x \in \Sigma^*$ whose profile vector is p provided another condition is met: Construct the DeBruijn graph $G_q^{\ell-1}$, and remove all edges $w \in \Sigma^\ell$ such that $p(w) = 0$. Then remove all isolated vertices (i.e., vertices with no incoming edges and no outgoing edges). If the resulting graph is strongly connected, such a string x exists. To see that, take $p(w)$ parallel copies of each edge w. Then, each vertex has in-degree that equals its out-degree (due to flow conservation of p), and since the graph is strongly connected, there exists an Eulerian cycle. The string x associated with the Eulerian cycle (i.e., whose sequence of sliding windows of length ℓ equals the sequence of edges in the cycle) has a profile vector p.

Given a (flow conserving) profile vector p such that $p \vDash \pi$ for some permutation π, the above implies a deterministic algorithm for generating a string x such that $x \vDash \pi$. Alternatively, given any vector $r \in \mathbb{R}^{q^\ell}$ such that $r \vDash \pi$ for some π, it is possible to produce the corresponding string x by either turning it to an integer vector[1] which satisfies the same permutation and repeating the above, or by the following randomized algorithm.

Given such a vector r, find α and β in $\mathbb{R}$ such that $s \triangleq \alpha r + \beta \mathbf{1}$ is a positive vector whose sum of entries is 1 (which clearly satisfies the same permutation as r), and define $M_s \in \mathbb{R}^{q^\ell \times q^\ell}$ such that

$$(M_s)_{a,b} = \begin{cases} \frac{s(v\sigma)}{\sum_{\tau\in\Sigma} s(v\tau)} & \text{if } (a,b) \text{ is an edge and } b = v\sigma, \\ 0 & \text{otherwise,} \end{cases}. \tag{2}$$

It is proved in Lemma 18 in Appendix A that M_s is a transition matrix of a Markov chain on G_q^ℓ, and its stationary distribution is s. Hence, it follows from the law of large numbers for Markov chains that following this chain for long enough produces a string whose profile vector satisfies the same permutation as s and r.

The main goals of this paper are to characterize, bound, and construct sets of feasible permutations in $S_{q,\ell}$. To this end, given q and ℓ, let

$$\begin{aligned} \mathcal{F}_{q,\ell} &\triangleq \left|\{\pi \in S_{q,\ell} \mid \exists x \in \Sigma^*,\ x \vDash \pi\}\right|, \text{ and} \\ R_{q,\ell} &\triangleq \frac{\log_2 \mathcal{F}_{q,\ell}}{\log_2(q^\ell!)}. \end{aligned} \tag{3}$$

It is evident that $\mathcal{F}_{q,1} = q!$ for any q. In addition, $\mathcal{F}_{2,\ell} = 0$ for every $\ell \geq 2$, since (assuming $\Sigma = \{0,1\}$) the flow-conservation constraint $p_x(01^{\ell-1}) + p_x(1^\ell) = p_x(1^{\ell-1}0) + p_x(1^\ell)$ implies $p_x(01^{\ell-1}) = p_x(1^{\ell-1}0)$ contradicting the requirement that profile-vector entries be distinct. Therefore, the simplest set of parameters, that will be prominent in the sequel, is $\ell = 2$ and $q = 3$.

III. Upper Bound

For a given permutation $\pi \in S_{q,\ell}$, a necessary condition for π to be feasible is given, and later used to obtain a bound on the number of feasible permutations. To formulate this condition, color an edge (a,b) of G_q^ℓ in green if $\pi(a) < \pi(b)$, and in red if $\pi(a) > \pi(b)$. In addition, for a *non-constant* string[2] $v \in \Sigma^{\ell-1}$, let $G(v)$ be an induced subgraph of G_q^ℓ on the set $T(v) \triangleq \{\sigma_i v\}_{i=1}^q \cup \{v\sigma_i\}_{i=1}^q$, whose edges are colored as in G_q^ℓ. For the next lemma, recall that a perfect matching in a graph is a vertex-disjoint set of edges which covers the entire vertex set of the graph.

[1]This is possible by finding a close enough rational approximation, and multiplying by the least common multiple of its entries' denominators. Alternatively, one may multiply it by a large enough constant such that the absolute difference between any two distinct entries is at least 3, and apply the algorithm of [3, Thm. 39].

[2]That is, a string v that contains at least two distinct symbols.

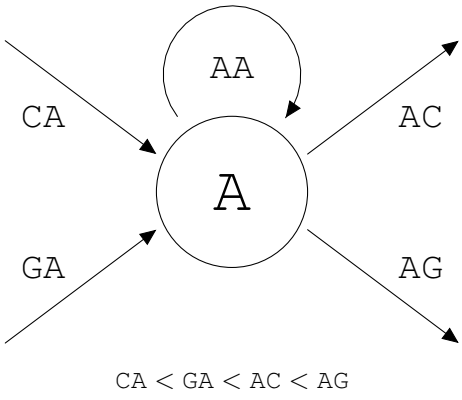

Fig. 2. An infeasible permutation. Since only closed strings are considered, any profile vector p must satisfy *flow conservation constraints* at any node of $G_q^{\ell-1}$. For example, if $q = 3$ and $\ell = 2$, the constraint that corresponds to the node $\mathtt{A}$ is $p(\mathtt{AA}) + p(\mathtt{CA}) + p(\mathtt{GA}) = p(\mathtt{AA}) + p(\mathtt{AC}) + p(\mathtt{AG})$. Hence, any permutation in which $(\mathtt{CA}, \mathtt{GA}, \mathtt{AC}, \mathtt{AG})$ can be attained by deletion of entries is infeasible.

Lemma 1. *If there exists $v \in \Sigma^{\ell-1}$ such that $G(v)$ contains an all-red perfect matching or an all-green perfect matching, then π is infeasible.*

Proof: Assume that there exists a subgraph $G(v)$ with an all-red perfect matching $\{(\sigma_i v, v\sigma_{\kappa(i)})\}_{i=1}^{q}$ for some permutation κ on $[q] \triangleq \{1, \ldots, q\}$. If there exists a string x over Σ which satisfies π, then since x is a closed string, the profile vector p of x satisfies that $\sum_{i=1}^{q} p(\sigma_i v) = \sum_{i=1}^{q} p(v\sigma_i)$. However, since $p(\sigma_i v) > p(v\sigma_{\kappa(i)})$, we have that $\sum_{i=1}^{q} p(\sigma_i v) > \sum_{i=1}^{q} p(v\sigma_i)$, a contradiction. If $G(v)$ contains an all-green matching, the proof is similar. ∎

Let $\pi|_{T(v)}$ be the result of *deleting* from π any element not in $T(v)$. For example, if we take $q = 3$, $\ell = 2$, and $\pi = (\mathtt{AA}, \mathtt{AC}, \mathtt{CA}, \mathtt{GA}, \mathtt{AG}, \mathtt{CC}, \mathtt{CG}, \mathtt{GC}, \mathtt{GG})$, then $\pi|_{T(\mathtt{A})} = (\mathtt{AA}, \mathtt{AC}, \mathtt{CA}, \mathtt{GA}, \mathtt{AG})$. The given bound is derived by counting the number of *infeasible* permutations on mutually disjoint sets $\{T(u_i)\}_{i=1}^{k}$, and estimating the number of permutations in $S_{q,\ell}$ which contain an infeasible permutation on at least one $T(u_i)$. To this end, the following lemmas are given.

Lemma 2. *If $\{u_i\}_{i=1}^{k} \subseteq \Sigma^{\ell-1}$ is an independent set of vertices in $G_q^{\ell-1}$, then the sets $T(u_i)$ are mutually disjoint.*

Proof: If there exist i and j in $[k]$ such that $T(u_i) \cap T(u_j) \neq \varnothing$, then since $u_i \neq u_j$ it follows that there exist σ and τ in Σ such that either $\sigma u_i = u_j \tau$ or $u_i \sigma = \tau u_j$. Without loss of generality assume that $\sigma u_i = u_j \tau$ and notice that in $G_q^{\ell-1}$, σu_i is an edge entering node u_i, and $u_j \tau$ is a edge leaving node u_j. Thus, u_i and u_j are connected by an edge, a contradiction. ∎

Lemma 3. *Let $\{u_i\}_{i=1}^{k} \subseteq \Sigma^{\ell-1}$ be an independent set of non-constant vertices in $G_q^{\ell-1}$. For any set of permutations $\{\pi_i\}_{i=1}^{k}$, where π_i is a permutation on $T(u_i)$, there are $\frac{q^\ell!}{(2q)!^k}$ permutations π on Σ^ℓ such that $\pi|_{T(u_i)} = \pi_i$ for all $i \in [k]$.*

Proof: According to Lemma 2, the sets $T(u_i)$ are mutually disjoint. Hence, the elements of $\cup_{i=1}^{k} T(u_i)$ may be arbitrarily interleaved such that for all i, the relative order π_i of $T(u_i)$ is maintained. It is readily verified that the number of ways to interleave the elements of $\{T(u_i)\}_{i=1}^{k}$, while maintaining the relative orders $\{\pi_i\}_{i=1}^{k}$, equals the number of multi-permutations on the multi-set

$$\{1, \ldots, 1, 2, \ldots, 2, \ldots, k, \ldots, k\},$$

where each element appears exactly $2q$ times. Since the number of multi-permutations on this multi-set is given by the multinomial coefficient $(2q, \ldots, 2q)! \triangleq \frac{(2qk)!}{(2q)!^k}$, and since the remaining $q^\ell - k \cdot 2q$ elements may by inserted consecutively and arbitrarily into the resulting permutation, we have that the number of permutations $\pi \in S_{q,\ell}$ such that $\pi|_{T(u_i)} = \pi_i$ is

$$\frac{(2qk)!}{(2q)!^k} \cdot \prod_{i=2qk+1}^{q^\ell} i = \frac{q^\ell!}{(2q)!^k}$$

∎

For any non-constant $v \in \Sigma^{\ell-1}$, we now show that the set of monochromatic perfect matchings in $G(v)$ is in one-to-one correspondence with the set $\mathcal{S}_q \subseteq \{\pm 1\}^{2q}$, where $s \in \mathcal{S}_q$ if and only if $\sum_{i=1}^{2q} s_i = 0$ and $\sum_{i=1}^{j} s_i \geq 0$ for all $j \in [2q]$. Consequently, the number of monochromatic perfect matchings is given by the q-th Catalan number $C_q = \frac{1}{q+1} \cdot \binom{2q}{q}$. For $s \in \mathcal{S}_q$, let $s^+ \triangleq \{i \in [2q] \mid s_i = 1\}$, and $s^- \triangleq \{i \in [2q] \mid s_i = -1\}$.

Lemma 4. *For $s \in \mathcal{S}_q$ there exists a bijection $\phi : s^+ \to s^-$ such that $\phi(t) > t$ for all $t \in s^+$.*

Proof: The function ϕ is defined in a recursive manner. This definition relies on the recursive structure of $\mathcal{S}_q$, which is as follows. First we observe that the first entry of s must be $s_1 = 1$. Next, denote the index of the left-most (-1)-entry of s by m, and let s' be the vector which results from s by replacing both $s_1 = 1$ and $s_m = -1$ by 0. For any $j \in [2q]$ we have that

$$\sum_{i=1}^{j} s'_i = \begin{cases} j-1 & 1 \le j < m \\ \sum_{i=1}^{j} s_i & m \le j \end{cases}.$$

Hence, it is readily verified that by omitting the first and m-th entry of s we obtain a vector in $\mathcal{S}_{q-1}$. Therefore, the function ϕ may be defined by setting $\phi(1) = m$, omitting the first and m-th entry, and applying the same rule recursively. ∎

Lemma 5. *For $v \in \Sigma^{\ell-1}$, the number of infeasible permutations on $T(v)$ is at least $\frac{2}{q+1} \cdot (2q)!$.*

Proof: We count the number of permutations on $T(v)$ that induce a monochromatic matching in $G(v)$. Clearly, the existence of a monochromatic matching is oblivious to the *internal* permutation on each of the sets $\{\sigma_i v\}_{i=1}^{q}$ and $\{v\sigma_i\}_{i=1}^{q}$. Further, the color of a monochromatic matching, if it exists, is uniquely determined by the lowest-ranking element of $T(v)$. That is, if the lowest-ranking element belongs to $\{\sigma_i v\}_{i=1}^{q}$, the matching is green, and otherwise it is red. Therefore, the number of permutations on $T(v)$ that induce a monochromatic matching is

$$(q!)^2 \cdot 2 \cdot \kappa,$$

where κ is the number of ways to interleave two fixed permutations on $\{\sigma_i v\}_{i=1}^{q}$ and on $\{v\sigma_i\}_{i=1}^{q}$, such that the identity of the lowest-ranking element (i.e., if it belongs to $\{\sigma_i v\}_{i=1}^{q}$ or to $\{v\sigma_i\}_{i=1}^{q}$) is determined, and such that the resulting permutation induces a monochromatic matching. In what follows we show that κ equals the q-th Catalan number, by employing the aforementioned set $\mathcal{S}_q$.

For a permutation π_1 on $\{\sigma_i v\}_{i=1}^{q}$ and a permutation π_2 on $\{v\sigma_i\}_{i=1}^{q}$, let $C(\pi_1, \pi_2)$ be the number of permutations π on $T(v)$ such that reducing π to $\{\sigma_i v\}_{i=1}^{q}$ results in π_1, reducing π to $\{v\sigma_i\}_{i=1}^{q}$ results in π_2, and there exists a monochromatic green matching in $G(v)$. By using a bijection to $\mathcal{S}_q$, it is shown that $|C(\pi_1, \pi_2)| = C_q$.

Given a vector $s \in \mathcal{S}_q$, let $f(s)$ be the permutation of $T(v)$ which results from replacing the 1 entries of s by the elements of $\{\sigma_i v\}_{i=1}^{q}$, sorted by π_1, and replacing the -1 entries by the elements of $\{v\sigma_i\}_{i=1}^{q}$, sorted by π_2. According to Lemma 4, in the resulting permutation $f(s)$ any element in $\{\sigma_i v\}_{i=1}^{q}$ has a unique element in $\{v\sigma_i\}_{i=1}^{q}$ that is higher ranked than it, and hence, $f(s)$ induces a green matching.

Conversely, let π be a permutation on $T(v)$ that induces a green matching M, and hence, the smallest ranking element belongs to $\{\sigma_i v\}_{i=1}^{k}$. Let $g(\pi)$ be the vector in $\{\pm 1\}^{2q}$ that results from replacing all entries of π which contain an element from $\{\sigma_i v\}_{i=1}^{q}$ by 1, and replacing all entries which contain an element from $\{v\sigma_i\}_{i=1}^{q}$ by -1 (this is well defined since v has no self loops, and thus $\{\sigma_i v\}_{i=1}^{q} \cap \{v\sigma_i\}_{i=1}^{q} = \varnothing$).

Assume for contradiction that $g(\pi)$ induces a negative partial sum $\sum_{i=1}^{j} s_i < 0$ for some $j \in [2q]$. Since $s_1 = 1$, we may choose a *minimal* such j, where $\sum_{i=1}^{j-1} s_i = 0$, $\sum_{i=1}^{j} s_i = -1$, and $s_j = -1$. Consider the edges of M as connecting between the respective indices of s, and note that since the matching is green, all edges $(i, j) \in M$ satisfy that $s_i = 1$, $s_j = -1$, and $i < j$. If there is no edge in the set $[j-1] \times \{j, \ldots, 2q\}$, we have that M contains an edge (j, t) with $t > j$ and $s_j = -1$, a contradiction. Similarly, if there exists an edge of M in $[j-1] \times \{j, \ldots, 2q\}$, since the number of 1 and -1 entries among $s_1, \ldots, s_{j-1}$ is equal, we have that there exists a (-1)-entry s_t, for $t \le j-1$, such that $(t, m) \in M$ for some $m > t$. Once again, a contradiction.

Since the above mappings g and f are injective, we have that $|C(\pi_1, \pi_2)| = |\mathcal{S}_q| = C_q$ [12, p. 116]. Note that the proof of this equality is identical for any π_1 and π_2, and that the proof is symmetric if the induced matching is red. Note also that a permutation π on $T(v)$ cannot induce both a red and a green matching, since this would imply that $\sum_{\sigma\in\Sigma} \pi(\sigma v) < \sum_{\sigma\in\Sigma} \pi(v\sigma)$ and $\sum_{\sigma\in\Sigma} \pi(\sigma v) > \sum_{\sigma\in\Sigma} \pi(v\sigma)$, a contradiction. Therefore, we have that the number of permutations on $T(v)$ that induce a monochromatic matching is at least

$$(q!)^2 \cdot 2 \cdot C_q = \frac{2}{q+1} \cdot (2q)!.$$

∎

To state the main result of this section, we introduce the *loopless independence number* $\alpha^*(q, \ell)$ of G_q^ℓ [9], which is the size of the largest subset of nodes that contains no internal edges and no self-loops (i.e., the nodes are non-constant). A lower bound on α^* may be easily derived from [9].

Lemma 6 ([9])**.** $\alpha^*(q, \ell) \ge \frac{q^\ell - q^{\ell-2}}{4}$*, for all $q \ge 2$ and $\ell \ge 2$.*

Proof: According to [9, Proposition 5.2] we have that if $q \ge 2$ and $\ell \ge 3$, then $\alpha^*(q, \ell) \ge q \cdot \alpha^*(q, \ell - 1)$. By applying this claim recursively, we have that $\alpha^*(q, \ell) \ge q^{\ell-2} \cdot \alpha^*(q, 2)$. According to [9, Sec. 5] we have that if $q \ge 2$ then $\alpha^*(q, 2) \ge \frac{q^2-1}{4}$. The result follows from combining these claims. ∎

Theorem 1. *For all $q \geq 3$ and $\ell \geq 3$, the number of feasible permutations is at most*

$$\mathcal{F}_{q,\ell} \leq q^\ell! \cdot \left(\frac{q-1}{q+1}\right)^{\frac{1}{4}(q^{\ell-1}-q^{\ell-3})}.$$

Proof: Let $\alpha^* = \alpha^*(q, \ell-1)$, and let $\{u_i\}_{i=1}^{\alpha^*} \subseteq \Sigma^{\ell-1}$ be a maximum loopless independent set in $G_q^{\ell-1}$. Since it is loopless, it follows that none of $\{u_i\}_{i=1}^{\alpha^*}$ is constant. Hence, Lemma 3 implies that for every set $\{\pi_i\}_{i=1}^{\alpha^*}$ of permutations, where π_i is a permutation on $T(u_i)$, there are $\frac{q^\ell!}{(2q)!^{\alpha^*}}$ permutations $\pi \in S_{q,\ell}$, such that $\pi|_{T(u_i)} = \pi_i$ for all $i \in [\alpha^*]$.

Since by Lemma 5 the number of infeasible permutations on each $T(u_i)$ is at least $\frac{2}{q+1}(2q)!$, it follows that there are at least

$$(2q)!^{\alpha^*} - \left(\frac{q-1}{q+1} \cdot (2q)!\right)^{\alpha^*} = \left(1 - \left(\frac{q-1}{q+1}\right)^{\alpha^*}\right)(2q)!^{\alpha^*}$$

sets $\{\pi_i\}_{i=1}^{\alpha^*}$ which contain at least one infeasible permutation. This implies, together with Lemma 6, that the number of infeasible permutations $\pi \in S_{q,\ell}$ is at least

$$q^\ell! - \mathcal{F}_{q,\ell} \geq \left(1 - \left(\frac{q-1}{q+1}\right)^{\alpha^*}\right) \cdot (2q)!^{\alpha^*} \cdot \frac{q^\ell!}{(2q)!^{\alpha^*}} = \left(1 - \left(\frac{q-1}{q+1}\right)^{\alpha^*}\right) \cdot q^\ell! \geq \left(1 - \left(\frac{q-1}{q+1}\right)^{\frac{q^{\ell-1}-q^{\ell-3}}{4}}\right) \cdot q^\ell!$$

■

For $\ell = 2$, similar techniques do not hold since $\alpha^* = \alpha(q,1) = 0$. However, we may apply Lemma 5 directly without enhancing it by the loopless independence number of $G_q^{\ell-1}$. This results in the following weaker bound, whose proof is similar to that of Theorem 1, and is therefore omitted.

Theorem 2. *For all $q \geq 3$ we have $\mathcal{F}_{q,2} \leq (q^2)! \cdot \frac{q-1}{q+1}$.*

Theorem 1 readily implies that if either q or ℓ goes to infinity, the fraction of feasible permutations goes to zero. However, in terms of rate, both $R_{q,\ell} \xrightarrow[q\to\infty]{} 1$ and $R_{q,\ell} \xrightarrow[\ell\to\infty]{} 1$, leaving the possibility for high-rate schemes wide open. In the sequel, a set of feasible permutations is constructed, whose rate is bounded from below by a constant when ℓ is fixed and q tends to infinity (Lemma 10 in Section V).

IV. A Polynomial Algorithm for Deciding Feasibility

In this section it is shown that a given permutation $\pi \in S_{q,\ell}$ can be decided to be feasible or not in time polynomial in q^ℓ (i.e., polynomial in the length of the permutation). If it is feasible, the time complexity of producing a string which satisfies it depends on its minimal length, a topic which is studied in Section VI. The given algorithm relies on the following simple claim.

Lemma 7. *A permutation $\pi \in S_{q,\ell}$ is feasible if and only if there exists a nonnegative vector $\chi \in \mathbb{R}^{q^\ell}$ such that $\pi \vDash \chi$ and for all $v \in \Sigma^{\ell-1}$, $\sum_{\sigma\in\Sigma} \chi(v\sigma) = \sum_{\sigma\in\Sigma} \chi(\sigma v)$.*

Proof: If π is feasible, then by the definition of feasibility, there exists $x \in \Sigma^*$ such that its ℓ-profile vector p_x satisfies π, and hence, setting $\chi = p_x$ suffices. Conversely, given a non-negative vector χ that satisfies π, as well as flow conservation, we make the following observation: for any $\alpha, \beta \in \mathbb{R}$, $\alpha, \beta \geq 0$, we have that $\alpha\chi + \beta\mathbf{1}$ is a non-negative vector that satisfies π as well as flow conservation. Choose α and β such that $\chi' \triangleq \alpha\chi + \beta\mathbf{1}$ has $\chi(v) \geq 1$ and $|\chi'(v) - \chi'(v')| \geq 3$ for all $v, v' \in \Sigma^\ell$, $v \neq v'$.

By [3, Appendix], there exists an *integer* flow-conserving vector χ'' where $\lfloor \chi'(v) \rfloor \leq \chi''(v) \leq \lceil \chi'(v) \rceil + 1$. By our choice of α and β, all the entries of χ'' are positive, distinct, and retain their rank, i.e., χ'' satisfies π. By Section II, this suffices for the existence of a string $x \in \Sigma^*$ whose profile vector is $p_x = \chi''$, hence x satisfies π. ■

Given a permutation π, Lemma 7 gives rise to the following linear programming algorithm for deciding its feasibility.

- **Variables:** $\{\chi(v) \mid v \in \Sigma^\ell\}$.
- **Objective:** None.
- **Constriants:**
  - For all $v \in \Sigma^\ell$, $\chi(v) \geq 1$.
  - For all distinct u and v in Σ^ℓ such that $\pi(u) > \pi(v)$, $\chi(u) \geq \chi(v) + 1$.
  - For all $v \in \Sigma^{\ell-1}$, $\sum_{\sigma\in\Sigma} \chi(v\sigma) = \sum_{\sigma\in\Sigma} \chi(\sigma v)$.

According to Lemma 7, determining the feasibility of this system is equivalent to determining the feasibility of the given permutation π.

Note that the number of variables is q^ℓ, and the number of constraints is $q^\ell + \binom{q^\ell}{2} + q^{\ell-1}$, which is polynomial in q^ℓ. Hence, the feasibility question may be solved in polynomial time (in q^ℓ).

Since the coefficients in the linear program are all rational, the feasible solutions contain a solution χ that is rational in all of its entries. One may then find a corresponding string by the techniques that are mentioned in Section II.

V. Encoding Algorithms

In this section encoding algorithms are given for any ℓ and any q. These algorithms provide a lower bound on the number of feasible permutations for the respective parameters. Since an additive structure of the alphabet is required, it is assumed in this section that $\Sigma = \mathbb{Z}_q$, the set of integers modulo q.

According to Lemma 7 and its subsequent discussion, to come up with a feasible permutation it suffices to provide the corresponding vector χ. From this vector the corresponding permutation and a suitable string may be computed by either the randomized or the deterministic algorithms that are mentioned after Lemma 7. Hence, the algorithms that are detailed in this section focus on providing the vector χ.

To clearly describe the constraints on the vector χ, by abuse of notation it will be considered either as a vector in $\mathbb{R}^{q^\ell}$ or as a *matrix* in $\mathbb{R}^{q^{\ell-1}\times q^{\ell-1}}$. That is, for a given string $u \in \mathbb{Z}_q^\ell$, the notation $\chi(u)$ stands for the u-th entry of χ when seen as a vector, and for two strings $w, w' \in \Sigma^{\ell-1}$, the notation $\chi(w,w')$ stands for the (w,w')-entry when seen as a matrix, where

$$\chi(w,w') \triangleq \begin{cases} \chi(u) & \text{if } (w,w') \text{ is a } u\text{-labeled edge in } E(G_q^{\ell-1}), \\ 0 & \text{else.} \end{cases} \tag{4}$$

Using this notation, for $v \in \mathbb{Z}_q^{\ell-1}$ the constraint $\sum_{\sigma\in\mathbb{Z}_q}\chi(v\sigma) = \sum_{\sigma\in\mathbb{Z}_q}\chi(\sigma v)$ may be written as $\sum_{u\in\mathbb{Z}_q^{\ell-1}}\chi(v,u) = \sum_{u\in\mathbb{Z}_q^{\ell-1}}\chi(u,v)$, i.e., the v-th row sum equals the v-th column sum. A vector (matrix) χ which satisfies these constraints and satisfies a permutation, is called a *feasible vector (matrix)*. Note that if a feasible vector χ has no zero entries, then the support of the corresponding matrix is identical to the support of the adjacency matrix of $G_q^{\ell-1}$.

We present two algorithms in this section. The first fixes a window size of $\ell = 2$, and recursively increases q. The second algorithm builds on the first one, and extends the window size ℓ.

A. A recursive encoding algorithm for $\ell = 2$ and any q

This algorithm operates recursively on q, where the base case is $q = 3$. To resolve the base case, a repository of all feasible permutations (or their corresponding feasible matrices) for $q = 3$ and $\ell = 2$ should be maintained. Using a computer program which is based on Lemma 7, this repository was constructed within a few minutes on a laptop computer, and its size was discovered to be $f_{3,2} \triangleq 30240$. For future use, it is convenient to assume that all matrices in this repository contain strictly positive integer entries, which is always possible in light of the proof of Lemma 7 and the fact that G_q^1 is the complete graph on q nodes. In Algorithm 1 which follows, the information vector is taken from

$$I_q \triangleq [f_{3,2}] \times (S_4 \times K_4) \times \cdots \times (S_q \times K_q), \tag{5}$$

where for any positive integer m, the set K_m consists of all (m^2-m+1)-bit vectors with Hamming weight m. Given $t \in K_m$, a vector x of length $(m-1)^2$ and a vector y of length m, let $t(x,y)$ be the vector which results from replacing the 1-entries in t by the entries of x and the 0-entries by y, while maintaining their original order.

Algorithm 1: $A_q(v)$, an encoding algorithm for $\ell = 2$ and any q.

Data: A repositpry R of $f_{3,2}$ integer feasible matrices for all feasible permutations in $S_{3,2}$.
Input : An information vector $v \in I_q$.
Output: A feasible matrix $\chi \in \mathbb{N}^{q\times q}$ (which represents a feasible vector $\chi \in \mathbb{N}^{q^2}$).

if $q = 3$ **then** return the v-th feasible matrix in R;
Denote $v = (v', (\pi_q, t_q))$ for $v' \in I_{q-1}$, $\pi_q \in S_q$, and $t_q \in K_q$.
Apply $A_{q-1}(v')$ to get a matrix χ', and for $i,j \in \mathbb{Z}_{q-1}$ denote $x_{i,j} \triangleq (q+1)\cdot(\chi')_{i,j}$.
Choose $y \triangleq (y_0,\ldots,y_{q-1}) \in \mathbb{N}^q$ that is ordered by π_q, such that $\sum_{i=0}^{q-1} y_i$ is minimal, and such that $t_q(\text{sort}(y), \text{sort}((q+1)\cdot\chi')$ is sorted[3].
For $\varepsilon \triangleq \frac{1}{q}$, let

$$\chi \triangleq \begin{pmatrix} x_{0,0} & x_{0,1} & \cdots & x_{0,q-2} & y_0 \\ x_{1,0}+\varepsilon & x_{1,1} & \cdots & x_{1,q-2} & y_1-\varepsilon \\ x_{2,0}+\varepsilon & x_{2,1} & \cdots & x_{2,q-2} & y_2-\varepsilon \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{q-2,0}+\varepsilon & x_{q-2,1} & \cdots & x_{q-2,q-2} & y_{q-2}-\varepsilon \\ y_0-(q-2)\varepsilon & y_1 & \cdots & y_{q-2} & y_{q-1} \end{pmatrix}.$$

return $q\cdot\chi$.

[3]For a real vector v, let sort(v) be the output of applying a sorting algorithm to v.

In a nutshell, each recursive step of Algorithm 1 assigns values for entries in χ that correspond to strings which contain the newly added symbol of the alphabet. These new strings are ordered according to π_q, which is taken from the information vector, and are interleaved with the entries of χ' according to t_q. Following the next example, the correctness of Algorithm 1 is proved in detail.

Example 1. *Assume that for $q = 4$, the given information vector is $v = (v', (\pi_4, t_4))$ where*

$$\pi_4 = (2, 3, 4, 1),$$
$$t_4 = (0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0),$$

and v' is the index in $[f_{3,2}]$ of the permutation $(00, 01, 10, 20, 02, 11, 12, 21, 22)$ (which results from (1) by substituting 0 for A*, 1 for* C*, and 2 for* G*). Assume that by applying $A_3(v')$, we obtain the feasible matrix*

$$\chi' = \begin{pmatrix} 1 & 2 & 5 \\ 3 & 6 & 7 \\ 4 & 8 & 9 \end{pmatrix}, \tag{6}$$

where the rows and columns are identified lexicographically by $\mathbb{Z}_3$. Clearly, choosing $y = (12, 13, 21, 11)$ suffices since it is ordered by π_4, since

$$t_4(\mathit{sort}(y), \mathit{sort}(5 \cdot \chi')) = (5, 10, 11, 12, 13, 15, 20, 21, 25, 30, 35, 40, 45),$$

and since $\sum_{i=0}^{3} y_i$ is clearly minimal. Thus, it follows that

$$\chi = \begin{pmatrix} 5 & 10 & 25 & 12 \\ 15 + \frac{1}{4} & 30 & 35 & 13 - \frac{1}{4} \\ 20 + \frac{1}{4} & 40 & 45 & 21 - \frac{1}{4} \\ 12 - 2 \cdot \frac{1}{4} & 13 & 21 & 11 \end{pmatrix},$$

and hence the output matrix is

$$4 \cdot \chi = \begin{pmatrix} 20 & 40 & 100 & 48 \\ 61 & 120 & 140 & 51 \\ 81 & 160 & 180 & 83 \\ 46 & 52 & 84 & 44 \end{pmatrix},$$

which is an integer feasible matrix for

$$(00, 01, 33, 30, 03, 13, 31, 10, 20, 23, 32, 02, 11, 12, 21, 22).$$

First, notice that by the choice of ε and by the assumption on R, the matrix which is returned in either Line 1 or Line 6 has positive integer entries. Hence, the multiplication of χ' by $q+1$ in Line 3 provides q distinct integers between every two entries of χ'. In turn, this enables the choice of $y_0, \ldots, y_{q-1}$, and their interleaving with the entries of χ', in any possible way.

Second, to verify that the entries of χ are distinct, recall that $\{y_0, \ldots, y_{q-1}\} \cup \{x_{i,j} | i, j \in \mathbb{Z}_{q-1}\}$ is a set of distinct integers. Hence, since $\frac{1}{q} \leq \frac{1}{3}$ and $1 - (q-2)\varepsilon > \varepsilon$, it follows that the entries of χ are distinct. To prove the correctness of the algorithm, it ought to be shown that the row and column sum constraint from Lemma 7 is satisfied, and that different information vectors result in different permutations.

Lemma 8. *Let $q \cdot \chi$ be the output of Algorithm 1. Then for all $\tau \in \mathbb{Z}_q$, $\sum_{\sigma \in \mathbb{Z}_q} \chi(\tau, \sigma) = \sum_{\sigma \in \mathbb{Z}_q} \chi(\sigma, \tau)$.*

Proof: First, it is evident that

$$\text{sum of column } q-1 = \sum_{i \in \mathbb{Z}_q} y_i - \sum_{i=1}^{q-2} \varepsilon = \sum_{i \in \mathbb{Z}_q} y_i - (q-2)\varepsilon = \text{sum of row } q-1,$$

and since χ' it a feasible matrix, it follows that

$$\text{sum of column } 0 = y_0 - (q-2)\varepsilon + x_{0,0} + \sum_{i=1}^{q-2} (x_{i,0} + \varepsilon) = y_0 + \sum_{i=0}^{q-2} x_{0,i} = \text{sum of row } 0.$$

Further, for $1 \leq i \leq q-2$,

$$\text{sum of row } i = (x_{i,0} + \varepsilon) + \sum_{j=1}^{q-2} x_{i,j} + (y_i - \varepsilon) = \sum_{j=0}^{q-2} x_{j,i} + y_i = \text{sum of column } i.$$

■

Lemma 9. *If u and v are distinct information vectors in I_q such that $A_q(u) = \chi_u, A_q(v) = \chi_v$ and $\chi_u \vDash \pi_u, \chi_v \vDash \pi_v$ for some permutations π_u and π_v, then $\pi_u \neq \pi_v$.*

Proof: Note that according to the choice of ε, the relative order among the entries of χ' is preserved. Hence, if $u_i \neq v_i$ for some entry $i \geq 2$, then either the respective permutations between the y_j-s in stage i of the algorithm are distinct, or the interleaving of the y_j-s in the entries of χ' are distinct. In either case, it follows that there exists a pair of corresponding strings (either both new, or one is new and the other is old) on whom π_v and π_u disagree; and since this disagreement persists throughout the entire algorithm, the result follows. If u and v disagree only on their first (leftmost) entry, the proof is similar. ■

Corollary 1. *For any $q \geq 3$,*

$$\mathcal{F}_{q,2} \geq f_{3,2} \cdot \prod_{j=4}^{q} \left(j! \cdot \binom{j^2 - j + 1}{j} \right).$$

Even though taking q to infinity is artificial when DNA storage is discussed, it is inevitable if one wishes to estimate the contribution of Algorithm 1. Hence, the following lemma is given, where the proof is deferred to Appendix A.

Lemma 10. $\lim_{q\to\infty} R_{q,2} \geq \frac{1}{2}$.

B. A recursive encoding algorithm for any ℓ and any q

In this section, the recursive algorithm from Subsection V-A is used to obtain an encoding algorithm for any ℓ and any q. Inspired by [8] and [10], the recursion at stage ℓ relies on embedding the feasible matrix from stage $\ell - 1$ in *homomoprhic pre-images* of $G_q^{\ell-1}$ in G_q^{ℓ}, and breaking ties that emerge according to the information vector.

Definition 1. *[5] For graphs G and H, a function $f : V(G) \to V(H)$ is a (graph) homomorphism if for each pair of vertices u, v in $V(G)$, if (u, v) is an edge in G then $(f(u), f(v))$ is an edge in H.*

The algorithm which follows relies on the following homomorphism, and yet, many other homomorphisms exist, and either of them may be used similarly.

$$\begin{aligned}
&D_\ell : \mathbb{Z}_q^\ell \to \mathbb{Z}_q^{\ell-1} \\
&D_\ell(v) \triangleq (v_0 + v_1, v_1 + v_2, \ldots, v_{\ell-2} + v_{\ell-1}), \text{ and} \\
&D : \mathbb{Z}_q^* \to \mathbb{Z}_q^* \\
&D(v) \triangleq D_{|v|}(v).
\end{aligned}$$

It is an easy exercise to prove that for any ℓ, the function D is a q to 1 surjective homomorphism from G_q^ℓ to $G_q^{\ell-1}$. That is, for every $u \in \mathbb{Z}_q^{\ell-1}$, the set $D^{-1}(u)$ contains exactly q elements. Moreover, it is readily verified that this set may be written as

$$D^{-1}(u) = \{v_0, \ldots, v_{q-1}\}, \text{ such that } v_{i,0} = i \text{ for all } i \in \mathbb{Z}_q. \tag{7}$$

Similarly, every $u \in \mathbb{Z}_q^{\ell-1}$ satisfies that

$$\begin{aligned}
&\{D(\sigma u)\}_{\sigma\in\mathbb{Z}_q} = \{\sigma D(u)\}_{\sigma\in\mathbb{Z}_q}, \text{ and} \\
&\{D(u\sigma)\}_{\sigma\in\mathbb{Z}_q} = \{D(u)\sigma\}_{\sigma\in\mathbb{Z}_q}.
\end{aligned} \tag{8}$$

The information vector is taken from the set J_q^ℓ, which is defined as follows.

$$\begin{aligned}
&J_q^\ell \triangleq I_q \times \mathcal{P}_3 \times \mathcal{P}_4 \times \ldots \times \mathcal{P}_\ell, \text{ where} \\
&\mathcal{P}_i \triangleq \{P \mid P : \mathcal{A}_i \to S_{\mathbb{Z}_q}\}, \text{ and} \\
&\mathcal{A}_i \triangleq \{u \in \mathbb{Z}_q^{i-1} \mid 0 \notin \{u_0, u_{i-2}\}\} \text{ for all } i \in \{3, \ldots, \ell\}.
\end{aligned} \tag{9}$$

That is, an information vector $v \in J_q^\ell$ is of the form $v = (v', P_3, \ldots, P_\ell)$, where P_i is a function from $\mathcal{A}_i$ to $S_{\mathbb{Z}_q}$, and $v' \in I_q$ (see Subsection V-A). In addition, for i and j in $\mathbb{Z}_q$, let $\langle i, j\rangle_q \triangleq i + j \cdot q + 1$.

In stage ℓ, Algorithm 2 relies on embedding the matrix T, which results from stage $\ell - 1$, in entries of χ_ℓ that correspond to homomorphic pre-images of $G_q^{\ell-1}$ in G_q^ℓ. Since the homomorphism D is q to 1, this results in $q^{\ell-1}$ sets of entries in χ_ℓ, of q elements each, that contain identical entries. These equalities are broken by adding small constants to each set, where these constants are ordered according to the permutations in S_q that appear in the current entry of the information vector. To maintain the row and column sum constraint (Lemma 7), the additions will be excluded from a single entry in each row

and each column, where in turn, these excluded entries will be adjusted to cancel out the additions in their respective row or column.

Algorithm 2: $B_q^\ell(v)$, an encoding algorithm for any ℓ and any q.

Input : An information vector $v \in J_q^\ell$.

Output: A feasible matrix $\chi \in \mathbb{N}^{q^{\ell-1}\times q^{\ell-1}}$ (which represents a feasible vector $\chi \in \mathbb{N}^{q^\ell}$).

if $\ell = 2$ **then** return $A_q(v)$;

Denote $v = (v', P)$, where $P \in \mathcal{P}_\ell$ and $v' \in J_q^{\ell-1}$.

Set $T = 2 \cdot B_q^{\ell-1}(v')$.

foreach $v \in \mathbb{Z}_q^\ell$ **do**

Denote $v = v_0 v'$, where $v_0 \in \mathbb{Z}_q$ and $v' \in \mathbb{Z}_q^{\ell-1}$.

Denote $D(v) \triangleq w = (w_0, w', w_{\ell-2})$, where $w_0, w_{\ell-2} \in \mathbb{Z}_q$ and $w' \in \mathbb{Z}_q^{\ell-3}$.

Define

$$\chi_\ell(v) = \begin{cases} T(D(v)) + \frac{P(D(v))(v_0)}{q^{\langle w_0, w_{\ell-2}\rangle}} & (1) \text{ if } D(v) \in \mathcal{A}_\ell \\ T(D(v)) - \sum_{\mu\neq 0} \frac{P(\mu, w', w_{\ell-2})(\mu+v_0)}{q^{\langle \mu, w_{\ell-2}\rangle}} & (2) \text{ if } w_0 = 0, w_{\ell-2} \neq 0 \\ T(D(v)) - \sum_{\tau\neq 0} \frac{P(w_0, w', \tau)(v_0)}{q^{\langle w_0, \tau\rangle}} & (3) \text{ if } w_0 \neq 0, w_{\ell-2} = 0 \\ T(D(v)) + \sum_{\mu\neq 0}\sum_{\tau\neq 0} \frac{P(\mu, w', \tau)(\mu+v_0)}{q^{\langle \mu, \tau\rangle}} & (4) \text{ if } w_0 = w_{\ell-2} = 0 \end{cases}$$

end

return $q^{q^2} \cdot \chi_\ell$.

Example 2. *For $q = 3$ and $\ell = 3$, notice that $\mathcal{A}_3 = \{(11), (12), (21), (22)\}$ and that $\mathcal{P}_3 = \{P | P : \mathcal{A}_3 \to S_{\mathbb{Z}_3}\}$. The output matrix in this case is $\chi = q^{q^2}\chi_3$, where χ_3 is defined by using $T = 2B_3^2(v')$. The three leftmost columns of χ_3 are as follows.*

	00	*01*	*02*
00	$T(00) + \sum_{\mu\neq 0}\sum_{\tau\neq 0} \frac{P(\mu\tau)(\mu)}{q^{\langle \mu,\tau\rangle}}$	$T(01) - \sum_{\mu\neq 0} \frac{P(\mu 1)(\mu)}{q^{\langle \mu,1\rangle}}$	$T(02) - \sum_{\mu\neq 0} \frac{P(\mu 2)(\mu)}{q^{\langle \mu,2\rangle}}$
01	-	-	-
02	-	-	-
10	$T(10) - \sum_{\tau\neq 0} \frac{P(1\tau)(1)}{q^{\langle 1,\tau\rangle}}$	$T(11) + \frac{P(11)(1)}{q^{\langle 1,1\rangle}}$	$T(12) + \frac{P(12)(1)}{q^{\langle 1,2\rangle}}$
11	-	-	-
12	-	-	-
20	$T(20) - \sum_{\tau\neq 0} \frac{P(2\tau)(2)}{q^{\langle 2,\tau\rangle}}$	$T(21) + \frac{P(21)(2)}{q^{\langle 2,1\rangle}}$	$T(22) + \frac{P(22)(2)}{q^{\langle 2,2\rangle}}$
21	-	-	-
22	-	-	-

Notice that for each nonzero entry in the above table, the entries of χ_3 that share the same row or column and are not listed above are zero. Hence, one can verify that the row and column sums coincide with those of τ. To witness the distinctness of entries in a homomorphic preimage of D, consider for example $D^{-1}(11) = \{010, 101, 222\}$, and notice that

$$\chi_3(010) = T(11) + \frac{P(11)(0)}{q^{q+2}}$$
$$\chi_3(101) = T(11) + \frac{P(11)(1)}{q^{q+2}}$$
$$\chi_3(222) = T(11) + \frac{P(11)(2)}{q^{q+2}},$$

which are clearly distinct since $P(11)$ is a permutation. Yet another example is given by considering $D^{-1}(20) = \{021, 112, 200\}$, for which

$$\chi_3(021) = T(20) - \sum_{\tau\neq 0} \frac{P(2\tau)(0)}{q^{\langle 2,\tau\rangle}} = T(20) - \frac{P(21)(0)}{q^{q+3}} - \frac{P(22)(0)}{q^{2q+3}}$$
$$\chi_3(112) = T(20) - \sum_{\tau\neq 0} \frac{P(2\tau)(1)}{q^{\langle 2,\tau\rangle}} = T(20) - \frac{P(21)(1)}{q^{q+3}} - \frac{P(22)(1)}{q^{2q+3}}$$
$$\chi_3(200) = T(20) - \sum_{\tau\neq 0} \frac{P(2\tau)(2)}{q^{\langle 2,\tau\rangle}} = T(20) - \frac{P(21)(2)}{q^{q+3}} - \frac{P(22)(2)}{q^{2q+3}}.$$

Similarly, the above entries are distinct, e.g., on the $(q+3)$-rd digit of the fractional q-ary expansion, since $P(21)$ is a permutation.

To show the correctness of Algorithm 2, it suffices to show that χ_ℓ complies with the row and column sum constraint of Lemma 7, and that its entries are distinct.

Lemma 11. *In Algorithm 2 we have that $\sum_{\sigma\in\mathbb{Z}_q}\chi_\ell(\sigma u)=\sum_{\sigma\in\mathbb{Z}_q}\chi_\ell(u\sigma)$ for every $u\in\mathbb{Z}_q^{\ell-1}$ and every integer $\ell\geq 3$.*

Proof: The correctness in stage ℓ follows from the correctness in stage $\ell-1$, and the following four technical facts. These facts show that the additions to the entries of T cancel out, and hence the row and column sums of χ_ℓ are equal to those of T, which satisfies the row and column sum constraint. First, note that any $u\in\mathbb{Z}_q^{\ell-1}$ satisfies exactly one of the following two cases.

Case A1. $D(u)_{\ell-3}\neq 0$. For any $\sigma\in\Sigma$, the string $v=\sigma u$ satisfies that $w_{\ell-2}\neq 0$ and that $w_0=0$ if and only if $\sigma=-u_1$. Therefore, $\chi_\ell(v)$ is computed by either (1) or (2). Notice that

$$\begin{aligned}\sum_{\sigma\in\mathbb{Z}_q}\chi_\ell(\sigma u)&=\sum_{\sigma\neq -u_1}\left(T(D(\sigma u))+\frac{P(D(\sigma u))(\sigma)}{q^{\langle\sigma+u_1,u_{\ell-3}+u_{\ell-2}\rangle}}\right)\\&\quad+T\left(D(-u_1,u)\right)-\sum_{\mu\neq 0}\frac{P(\mu,D(u))(\mu+(-u_1))}{q^{\langle\mu,u_{\ell-3}+u_{\ell-2}\rangle}}\\&=\sum_{\sigma\neq -u_1}\left(T(D(\sigma u))+\frac{P(\sigma+u_1,D(u))(\sigma)}{q^{\langle\sigma+u_1,u_{\ell-3}+u_{\ell-2}\rangle}}\right)\\&\quad+T\left(D(-u_1,u)\right)-\sum_{\mu\neq -u_1}\frac{P(\mu+u_1,D(u))(\mu)}{q^{\langle\mu+u_1,u_{\ell-3}+u_{\ell-2}\rangle}}\\&=\sum_{\sigma\in\mathbb{Z}_q}T(D(\sigma u))\overset{(8)}{=}\sum_{\sigma\in\mathbb{Z}_q}T(\sigma D(u))=\sum_{\sigma\in\mathbb{Z}_q}T(D(u)\sigma).\end{aligned}$$

Case A2. $D(u)_{\ell-3}=0$. For any $\sigma\in\Sigma$, the string $v=\sigma u$ satisfies that $w_{\ell-2}=0$, and that $w_0=0$ if and only if $\sigma=-u_1$. Therefore, $\chi_\ell(v)$ is computed by either (3) or (4). Hence, let $D(u)\triangleq(r,0)$ for $r\in\mathbb{Z}_q^{\ell-3}$, and notice that

$$\begin{aligned}\sum_{\sigma\in\mathbb{Z}_q}\chi_\ell(\sigma u)&=\sum_{\sigma\neq -u_1}\left(T(D(\sigma u))-\sum_{\tau\neq 0}\frac{P(\sigma+u_1,r,\tau)(\sigma)}{q^{\langle\sigma+u_1,\tau\rangle}}\right)\\&\quad+T(D(-u_1,u))+\sum_{\mu\neq 0}\sum_{\tau\neq 0}\frac{P(\mu,r,\tau)(\mu+(-u_1))}{q^{\langle\mu,\tau\rangle}}\\&=\sum_{\sigma\neq -u_1}T(D(\sigma u))-\sum_{\sigma\neq -u_1}\sum_{\tau\neq 0}\frac{P(\sigma+u_1,r,\tau)(\sigma)}{q^{\langle\sigma+u_1,\tau\rangle}}\\&\quad+T(D(-u_1,u))+\sum_{\mu\neq -u_1}\sum_{\tau\neq 0}\frac{P(\mu+u_1,r,\tau)(\mu)}{q^{\langle\mu+u_1,\tau\rangle}}\\&=\sum_{\sigma\in\mathbb{Z}_q}T(D(\sigma u))\overset{(8)}{=}\sum_{\sigma\in\mathbb{Z}_q}T(\sigma D(u))=\sum_{\sigma\in\mathbb{Z}_q}T(D(u)\sigma).\end{aligned}$$

Further, any $u\in\mathbb{Z}_q^{\ell}$ also satisfies exactly one of the following two cases.

Case B1. $D(u)_0=0$. For any $\sigma\in\Sigma$, the string $v=u\sigma$ satisfies that $w_0=0$, and that $w_{\ell-2}=0$ if and only if $\sigma=-u_{\ell-2}$.

Therefore, $\chi_\ell(v)$ is computed by either (2) or (4). Hence, let $D(u) \triangleq (0, r)$ for $r \in \mathbb{Z}_q^{\ell-3}$, and notice that

$$\begin{aligned}\sum_{\sigma\in\mathbb{Z}_q}\chi_\ell(u\sigma) &= \sum_{\sigma\neq -u_{\ell-2}}\left(T(D(u\sigma)) - \sum_{\mu\neq 0}\frac{P(\mu, r, u_{\ell-2}+\sigma)(\mu+u_0)}{q^{\langle \mu, u_{\ell-2}+\sigma\rangle}}\right)\\ &\quad + T(D(u,-u_{\ell-2})) + \sum_{\mu\neq 0}\sum_{\tau\neq 0}\frac{P(\mu,r,\tau)(\mu+u_0)}{q^{\langle\mu,\tau\rangle}}\\ &= \sum_{\sigma\neq -u_{\ell-2}} T(D(u\sigma)) - \sum_{\mu\neq 0}\sum_{\sigma\neq -u_{\ell-2}}\frac{P(\mu, r, u_{\ell-2}+\sigma)(\mu+u_0)}{q^{\langle \mu, u_{\ell-2}+\sigma\rangle}}\\ &\quad + T(D(u,-u_{\ell-2})) + \sum_{\mu\neq 0}\sum_{\tau\neq -u_{\ell-2}}\frac{P(\mu, r, u_{\ell-2}+\tau)(\mu+u_0)}{q^{\langle \mu, u_{\ell-2}+\tau\rangle}}\\ &= \sum_{\sigma\in\mathbb{Z}_q} T(D(u\sigma)) \overset{(8)}{=} \sum_{\sigma\in\mathbb{Z}_q} T(D(u)\sigma) = \sum_{\sigma\in\mathbb{Z}_q} T(\sigma D(u)).\end{aligned}$$

Case B2. $D(u)_0 \neq 0$. For any $\sigma \in \Sigma$, the string $v = u\sigma$ satisfies that $w_0 \neq 0$, and that $w_{\ell-2} = 0$ if and only if $\sigma = -u_{\ell-2}$. Therefore, $\chi_\ell(v)$ is computed by either (1) or (3). Hence, it follows that

$$\begin{aligned}\sum_{\sigma\in\mathbb{Z}_q}\chi_\ell(u\sigma) &= \sum_{\sigma\neq -u_{\ell-2}}\left(T(D(u\sigma)) + \frac{P(D(u), u_{\ell-2}+\sigma))(u_0)}{q^{\langle u_0+u_1, u_{\ell-2}+\sigma\rangle}}\right)\\ &\quad + T(D(u,-u_{\ell-2})) - \sum_{\tau\neq 0}\frac{P(D(u),\tau)(u_0)}{q^{\langle u_0+u_1,\tau\rangle}}\\ &= \sum_{\sigma\neq -u_{\ell-2}} T(D(u\sigma)) + \sum_{\sigma\neq -u_{\ell-2}}\frac{P(D(u), u_{\ell-2}+\sigma))(u_0)}{q^{\langle u_0+u_1, u_{\ell-2}+\sigma\rangle}}\\ &\quad + T(D(u,-u_{\ell-2})) - \sum_{\tau\neq -u_{\ell-2}}\frac{P(D(u), u_{\ell-2}+\tau)(u_0)}{q^{\langle u_0+u_1, u_{\ell-2}+\tau\rangle}}\\ &= \sum_{\sigma\in\mathbb{Z}_q} T(D(u\sigma)) \overset{(8)}{=} \sum_{\sigma\in\mathbb{Z}_q} T(D(u)\sigma) = \sum_{\sigma\in\mathbb{Z}_q} T(\sigma D(u)).\end{aligned}$$

Since any $u \in \mathbb{Z}_q^{\ell-1}$ satisfies exactly one of the cases A1, A2, and exactly one of the cases B1, B2, the claim follows from the correctness of the recursive step. ∎

Lemma 12. *All entries of χ_ℓ are distinct.*

Proof: Since the output of Algorithm 2 is an integer vector, it follows that the minimum absolute difference between the entries of T is 2. Hence, since the additions to the entries of T are less than 1 in absolute value, it follows that all entries with distinct D-values are distinct (i.e., $\chi_\ell(u) \neq \chi_\ell(v)$ for all $u, v \in \mathbb{Z}_q^\ell$ such that $D(u) \neq D(v)$). It remains to prove that entries with an identical D-value are distinct. To this end, recall that entries with an identical D-value are of the form $D^{-1}(w) = \{v_0, \ldots, v_{q-1}\}$ for some D-value w, where v_i begins with i for all $i \in \mathbb{Z}_q$ (7).

Notice that the additions to the entries of T are of the form $\sum_{i=1}^{q^2}\frac{c_i}{q^i}$ where $c_i < q$ for all i. Hence, it is convenient to consider these additions in their fractional q-ary expansion. In the sequel, the claim is proven separately for the sets $D^{-1}(w)$ according to the types (1), (2), (3), or (4) of w, as noted in Algorithm 2. For example, $D^{-1}(w)$ is of type (1) if $w \in \mathcal{A}_\ell$, and of type (3) if $w_0 \neq 0$ and $w_{\ell-2} = 0$. Since the elements in $D^{-1}(w) = \{v_0, \ldots, v_{q-1}\}$ have distinct leftmost entries $\{v_{0,0}, \ldots, v_{q-1,0}\}$, it follows that –

(1) the additions $\frac{P(D(v_i))(v_{i,0})}{q^{\langle w_0, w_{\ell-2}\rangle}}$ to $T(D(v_i))$ are distinct for any $i \in \mathbb{Z}_q$ since $P(D(v_i)) = P(w)$ is a permutation.

(2) the additions $-\sum_{\mu\neq 0}\frac{P(\mu, w'w_{\ell-2})(\mu+v_{i,0})}{q^{\langle\mu, w_{\ell-2}\rangle}}$ to $T(D(v_i))$ contain the digit $P(\mu, w'w_{\ell-2})(v_{i,0}-\rho)$ in position $\langle\mu, w_{\ell-2}\rangle$ of the q-ary expansion for any $\mu \in \mathbb{Z}_q \setminus \{0\}$, and thus, for any fixed value of μ, different additions are distinct on this digit.

(3) the additions $-\sum_{\tau\neq 0}\frac{P(w_0, w', \tau)(v_{i,0})}{q^{\langle w_0,\tau\rangle}}$ to $T(D(v_i))$ contain the digit $P(w_0, w', \tau)(v_{i,0})$ in position $\langle w_0, \tau\rangle$ of the q-ary expansion for any $\tau \in \mathbb{Z}_q \setminus \{0\}$, and thus, for any fixed value of τ, different additions are distinct on this digit.

(4) the additions $\sum_{\mu\neq 0}\sum_{\tau\neq 0}\frac{P(\mu, w', \tau)(\mu+v_{i,0})}{q^{\langle\mu,\tau\rangle}}$ to $T(D(v_i))$ are distinct on the $\langle\mu,\tau\rangle$-th digit for any fixed values of ρ and τ.

Hence, any two additions to $T(D(v_i))$ and $T(D(v_j))$ for $i, j \in \mathbb{Z}_q$ such that $i \neq j$ are distinct, which implies that all entries of χ_ℓ are distinct. ∎

Lemma 11 and Lemma 12 imply that the output of Algorithm 2 is indeed a feasible matrix. It remains to prove that the outputs of Algorithm 2 for two distinct information vectors correspond to distinct feasible permutations, i.e., that Algorithm 2 is injective.

Lemma 13. *If s and t are distinct information vectors in J_q^ℓ such that $B_q^\ell(s) = q^{q^2}\chi_\ell^s, B_q^\ell(t) = q^{q^2}\chi_\ell^t$ and $\chi_\ell^s \vDash \pi_s, \chi_\ell^t \vDash \pi_t$ for some permutations π_s and π_t, then $\pi_s \neq \pi_t$.*

Proof: Since in every stage i of the algorithm, the additions to the entries of the respective T are less than half of the minimum absolute distance between the entries of T, it follows that Algorithm 2 preserves the order among the distinct homomorphic pre-images of D. That is, for any positive integers i and j, and any $u, v \in \mathbb{Z}_q^i$, if $\chi_i(u) > \chi_i(v)$, then for all $u', v' \in \mathbb{Z}_q^{i+j}$ such that $D^j(u') = u$ and $D^j(v') = v$, we have that $\chi_{i+j}(u') > \chi_{i+j}(v')$.

If $\ell = 2$, the claim follows from Lemma 9. Otherwise, denote $s = (s', P_3^s, \ldots, P_\ell^s)$ and $t = (t', P_3^t, \ldots, P_\ell^t)$. If $s' \neq t'$ then in stage $\ell = 3$ of the algorithm, by Lemma 9 there exist distinct u and v in $\mathbb{Z}_q^2$ on whom χ_2^s and χ_2^t disagree, i.e., $\chi_2^s(u) > \chi_2^s(v)$ and $\chi_2^t(u) < \chi_2^t(v)$. It follows that χ_3^s and χ_3^t disagree on any $u', v' \in \mathbb{Z}_q^3$ such that $D(u') = u$ and $D(v') = v$. If $s' = t'$, then assume that $P_i^s \neq P_i^t$ for some $i \in \{3, \ldots, \ell\}$, which implies that there exists $u \in \mathcal{A}_i$ that is mapped by P_i^s and P_i^t to distinct permutations in S_q. Hence, the pre-images $D^{-1}(u)$ are ordered differently in χ_i. Further, according to the above discussion, all pre-images of $D^{-1}(u)$ are ordered differently in all χ_j for $j > i$, and the claim follows. ∎

The above discussion, together with Corollary 1, implies the following lower bound on the number of feasible permutations. By the definition of $\mathcal{A}_i$ (9) we have that its size is $q^{i-1} - 2q^{i-2} + q^{i-3}$. Hence, the size of $\mathcal{P}_i$ is $(q!)^{q^{i-1}-2q^{i-2}+q^{i-3}}$, and the following corollary is immediate.

Corollary 2. *For any $q \geq 3$ and $\ell \geq 2$,*

$$\mathcal{F}_{q,\ell} \geq f_{3,2} \cdot \prod_{j=4}^{q} \left(j! \cdot \binom{j^2 - j + 1}{j} \right) \cdot \left(\prod_{i=3}^{\ell} (q!)^{q^{i-1}-2q^{i-2}+q^{i-3}} \right).$$

To estimate the contribution of Algorithm 2, one may take either q or ℓ to infinity. The proof of the following claim is given in Appendix A.

Lemma 14. *For any constant $\ell \geq 3$, $\lim_{q\to\infty} R_{q,\ell} \geq \frac{1}{\ell}$.*

When applications in DNA storage are discussed, it is natural to keep q a constant. However, the rate of the set of permutations which is given by Algorithm 2 goes to zero as ℓ tends to infinity. Hence, for finite values of q and ℓ, lower bounds for the corresponding the rates are given in the following table. Due to computational restrictions, the value of entry (q, ℓ) is $\frac{\log(q!)(q-1)(q^{\ell-2}-1)}{\ell q^\ell \log(q)}$, which is a lower bound on $R_{q,\ell}$.

q \ ℓ	3	4	5	6	7	8	9	10
3	0.0805	0.0805	0.0698	0.0597	0.0516	0.0452	0.0403	0.0362
4	0.1075	0.1007	0.0846	0.0714	0.0613	0.0537	0.0478	0.0430
5	0.1269	0.1142	0.0944	0.0792	0.0680	0.0595	0.0529	0.0476
6	0.1417	0.124	0.1015	0.0849	0.0728	0.0637	0.0567	0.0510
7	0.1533	0.1314	0.1070	0.0894	0.0766	0.0671	0.0596	0.0536
8	0.1627	0.1373	0.1113	0.0929	0.0797	0.0697	0.062	0.0558
9	0.1705	0.1421	0.1149	0.0959	0.0822	0.0719	0.0639	0.0575
10	0.1771	0.1461	0.1180	0.0984	0.0843	0.0738	0.0656	0.0590

VI. String Length for Feasible Permutations

In order to obtain a practical rank-modulation scheme from a given set of feasible permutations, it is essential to estimate the minimal lengths of the corresponding strings. To this end, for a feasible permutation $\pi \in S_{q,\ell}$, let $\mathrm{len}(\pi) \triangleq \min\{n \mid \exists x \in \Sigma^n, \pi \vDash x\}$, and for a set $S \subseteq S_{q,\ell}$ of feasible permutations, let $\mathrm{len}(S) \triangleq \max\{\mathrm{len}(\pi)\}_{\pi\in S}$. In this subsection an upper bound for $\mathrm{len}(T_{q,\ell})$ is given, where $T_{q,\ell}$ is the set of permutations which are given by Algorithm 2 for the parameters q and ℓ. It will be evident that for a constant q, the permutations in $T_{q,\ell}$ have corresponding strings whose lengths are polynomial in q^ℓ. The proofs in this subsection rely on the following simple lemma.

Lemma 15. *If $\pi \in S_{q,\ell}$ is a feasible permutation and $\chi \in \mathbb{N}^{q^\ell}$ is a feasible vector such that $\pi \vDash \chi$, then* $\mathrm{len}(\pi) \leq \sum_{v\in\Sigma^{q^\ell}} \chi(v)$.

Proof: Since χ is feasible, it follows that $\sum_{\sigma\in\Sigma} \chi(u\sigma) = \sum_{\sigma\in\Sigma} \chi(\sigma u)$ for all $u \in \mathbb{Z}_{q^{\ell-1}}$. Hence, there exists a closed path in G_q^ℓ that traverses edge v for $\chi(v)$ times for all $v \in \Sigma^\ell$. Clearly, the length of the corresponding string x is $\sum_{v\in\Sigma^\ell} \chi(v)$, and $x \vDash \pi$. ∎

For the set $T_{q,2}$ let $c_q \triangleq \min_{\Phi}\{\max(\chi)|\chi \in \Phi\}$, where Φ ranges over all sets of feasible vectors in $\mathbb{N}^{q^2}$ that contain a unique feasible vector for every permutation in $T_{q,2}$. The following lemma provides a bound for c_q as a function of c_3. Using a computer program, it was discovered that $c_3 \leq 16$.

Lemma 16. $c_q \leq 2^{q-3} \cdot \frac{q!}{6} \cdot \frac{(q+1)!}{24} \cdot c_3$.

Proof: It is evident from Algorithm 1 that an entry in χ is at most $c_{q-1}(q+1)+1$ if it is not in the top row or leftmost column, and $c_{q-1}(q+1)+q+1$ if it is. Hence,

$$\begin{aligned} c_q &\leq q(c_{q-1}(q+1)+q+1) = q(q+1)(c_{q-1}+1) \\ &\leq 2q(q+1)c_{q-1} \leq \ldots \leq 2^{q-3} \cdot \frac{q!}{6} \cdot \frac{(q+1)!}{24} \cdot c_3. \end{aligned}$$

■

In turn, Lemma 15 and Lemma 16 provide the following.

Corollary 3. $\mathrm{len}(T_{q,2}) \leq q^2 \cdot 2^{q-3} \cdot \frac{q!}{6} \cdot \frac{(q+1)!}{24} \cdot c_3$.

Let $c_{q,\ell} \triangleq \min_{\Phi}\{\max(\chi)|\chi \in \Phi\}$, where Φ ranges over all sets of vectors in $\mathbb{N}^{q^\ell}$ that contain a unique feasible vector for every permutation in $T_{q,\ell}$. For the next lemma, notice that $c_{q,2} = c_q$ by definition.

Lemma 17. $c_{q,\ell} \leq c_q(3q^{q^2})^{\ell-2}$.

Proof: Since the additions to the entries of T in Algorithm 2 are of absolute value at most 1, it follows from Line 3, Line 7 and Line 9 of Algorithm 2 that $c_{q,\ell} \leq (2c_{q,\ell-1}+1)q^{q^2} \leq c_{q,\ell-1}(3q^{q^2})$. Solving this recursion relation proves the claim. ■

The proof of the following corollary is immediate from Lemma 17.

Corollary 4. $\mathrm{len}(T_{q,\ell}) \leq c_3 \cdot \frac{2^{q-3} \cdot \frac{q!}{6} \cdot \frac{(q+1)!}{24}}{2q^2} \cdot 3^{\ell-2} \left(q^\ell\right)^{q^2+1}$.

It is evident from Corollary 4 that Algorithm 2 provides permutations whose corresponding strings are of lengths that is polynomial in q^ℓ for a fixed value of q. However, the constants that are involves in this bound, including the constant in the exponent, are rather large even for small values of q.

VII. Discussion

In this paper the question of feasibility of permutations was addressed. Our contributions include an upper bound on the number of feasible permutations, a linear programming algorithm for deciding the feasibility of a permutation, and a recursive algorithm for explicit construction of a large feasible set. Further, the length of the strings which correspond to permutations in this set was shown to be polynomial in q^ℓ for a fixed q. In addition, in Appendix B it is shown how feasible permutations of minimum Kendall-τ distance can be produced by pre-coding the information vector in Algorithm 2. However, the resulting distance is rather low.

The most prominent directions for future research seem to be studying the values of $R_{q,\ell}$ and $\mathrm{len}(\mathcal{F}_{q,\ell})$, and providing better constructions in terms of rate, distance, and length. Furthermore, alternative combinatorial models for DNA storage should be studied. For example, one may group the entries of the profile vector by sum or by cyclic equivalence, and study the achievable rates of applying rank modulation schemes on the resulting sets of vectors.

References

[1] J. Bornholt, R. Lopez, D. M. Carmean, L. Ceze, G. Seelig, and K. Strauss, "A DNA-based archival storage system," *ACM SIGOPS Operating Systems Review*, vol. 50, no. 2, pp. 637–649, 2016.

[2] G. M. Church, Y. Gao, and S. Kosuri, "Next-generation digital information storage in DNA," *Science*, vol. 337, p. 1628, 2012.

[3] O. Elishco, T. Meyerovitch, and M. Schwartz, "Semiconstrained systems," *IEEE Trans. Inform. Theory*, vol. 62, no. 4, pp. 811–824, Apr. 2016.

[4] N. Goldman, P. Bertone, S. Chen, C. Dessimoz, E. M. LeProust, B. Sipos, and E. Birney, "Towards practical, high-capacity, low-maintenance information storage in synthesized DNA," *Nature*, vol. 494, no. 7435, pp. 77–80, 2013.

[5] J. L. Gross and J. Yellen, *Handbook of Graph Theory*. CRC Press, 2004.

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

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

[8] A. Lempel, "On a homomorphism of the de Bruijn graph and its applications to the design of feedback shift registers," *IEEE Trans. Comput.*, vol. 100, no. 12, pp. 1204–1209, 1970.

[9] N. Lichiardopol, "Independence number of de Bruijn graphs," *Discrete Math.*, vol. 306, no. 12, pp. 1145–1160, 2006.

[10] M. Liu, "Homomorphisms and automorphisms of 2-D de Bruijn-Good graphs," *Discrete Math.*, vol. 85, no. 1, pp. 105–109, 1990.

[11] A. S. Motahari, G. Bresler, and D. N. C. Tse, "Information theory of DNA shotgun sequencing," *IEEE Trans. Inform. Theory*, vol. 59, no. 10, pp. 6273–6289, Oct. 2013.

[12] J. H. van Lint and R. M. Wilson, *A Course in Combinatorics, 2nd Edition*. Cambridge Univ. Press, 2001.

APPENDIX A
OMITTED PROOFS

Proof: (Proof of Lemma 10) According to Eq. (3) and Corollary 1, it follows that for any $q \geq 4$, and any real number $\alpha > 1$,

$$\begin{aligned} R_{q,2} &\geq \frac{1}{\log(q^2!)} \log\left(f_{3,2} \cdot \prod_{j=4}^{q} j! \cdot \binom{j^2-j+1}{j} \right) \\ &\geq \frac{1}{\log(q^2!)} \left(\sum_{i\geq 1} \sum_{j=\lfloor q/\alpha^i \rfloor + 1}^{\lfloor q/\alpha^{i-1} \rfloor} \left(\log(j!) + \log\binom{j^2-j+1}{j} \right) \right) \\ &\geq \frac{1}{\log(q^2!)} \sum_{i\geq 1} \left(\left(\left\lfloor \frac{q}{\alpha^{i-1}} \right\rfloor - \left\lfloor \frac{q}{\alpha^i} \right\rfloor \right) \left(\log\left(\left\lfloor \frac{q}{\alpha^i} \right\rfloor ! \right) + \log\binom{\lfloor \frac{q}{\alpha^i} \rfloor^2 - \lfloor \frac{q}{\alpha^i} \rfloor + 1}{\lfloor \frac{q}{\alpha^i} \rfloor} \right) \right), \end{aligned}$$

By using the simple lower bound $\binom{s}{t} \geq (\frac{s}{t})^t$, the identity $\lim_{m\to\infty} \frac{\log(m!)}{m \log m} = 1$, and by omitting the rounding operation when q is large enough, it follows that

$$\begin{aligned} \lim_{q\to\infty} R_{q,2} &\geq \lim_{q\to\infty} \sum_{i\geq 1} \frac{1}{2q^2 \log q} \left(\frac{2q^2(\alpha-1)}{\alpha^{2i}} \log \frac{q}{\alpha^i} \right) \\ &= \sum_{i\geq 1} \lim_{q\to\infty} \frac{1}{2q^2 \log q} \left(\frac{2q^2(\alpha-1)}{\alpha^{2i}} \log \frac{q}{\alpha^i} \right) \\ &= \sum_{i\geq 1} \frac{\alpha-1}{\alpha^{2i}} = \frac{1}{1+\alpha}, \end{aligned}$$

with a simple application of dominated convergence. Since this holds for every $\alpha > 1$,

$$\lim_{q\to\infty} R_{q,2} \geq \lim_{\alpha\to 1} \frac{1}{1+\alpha} = \frac{1}{2}.$$

■

Proof: (of Lemma 14) According to Corollary 2, it follows that for any $q \geq 3$ and $\ell \geq 3$,

$$R_{q,\ell} \geq \frac{\log\left(f_{3,2} \prod_{j=4}^{q} \left(j! \binom{j^2-j+1}{j} \right) \cdot \left(\prod_{i=3}^{\ell} (q!)^{q^{i-1}-2q^{i-2}+q^{i-3}} \right) \right)}{\log(q^\ell!)},$$

and hence,

$$\lim_{q\to\infty} R_{q,\ell} = \lim_{q\to\infty} \frac{\sum_{i=3}^{\ell} (q^{i-1} - 2q^{i-2} + q^{i-3}) q \log q}{\ell q^\ell \log q} = \frac{1}{\ell}$$

■

Lemma 18. *For a given vector positive vector $s \in \mathbb{R}^{q^\ell}$ whose sum of entries is 1, the matrix M_s (2) is a transition matrix of a Markov chain on a DeBruijn graph, whose stationary distribution is s.*

Proof: For given $v \in \Sigma^{\ell-1}$, the sum of row τv of M_s for any $\tau \in \Sigma$ is

$$\sum_{\sigma\in\Sigma} \frac{s(v\sigma)}{\sum_{\rho\in\Sigma} s(v\rho)} = \frac{\sum_{\sigma\in\Sigma} s(v\sigma)}{\sum_{\rho\in\Sigma} s(v\rho)} = 1,$$

and hence M_s is a transition matrix. Further, for any $v \in \Sigma^{\ell-1}$ and $\sigma \in \Sigma$, the $v\sigma$-th entry of sM_s equals

$$\begin{aligned} (sM_s)_{v\sigma} &= \sum_{\rho\in\Sigma} s(\rho v) \cdot (M_s)_{\rho v, v\sigma} \\ &= \sum_{\rho\in\Sigma} \frac{s(\rho v) \cdot s(v\sigma)}{\sum_{\tau\in\Sigma} s(v\tau)} \\ &= s(v\sigma) \cdot \frac{\Sigma_{\rho\in\Sigma} s(\rho v)}{\Sigma_{\tau\in\Sigma} s(v\tau)} = s(v\sigma), \end{aligned}$$

where the last equality follows from the flow-conservation of s. Hence, s is the stationary distribution of M_s. ■

APPENDIX B
MINIMUM DISTANCE BY THE KENDALL τ METRIC

Recall that the purpose of applying a rank-modulation scheme for the DNA storage channel is to endure small errors in the profile vector which is given at the output of the channel. Clearly, any pattern of errors that does not cause two entries in the profile vector to surpass one another is correctable. However, if the channel is restricted to a rank-modulation code by the Kendall-τ distance (see below), error resilience is improved at the cost of lower information rate. In particular, if the channel is restricted to permutations that are of distance at least $2t+1$ apart, then any error pattern of at most t adjacent transpositions is correctable. Fortunately, the structure of the information vectors in Algorithm 1 and Algorithm 2 allows rank-modulation codes to be incorporated conveniently. However, the resulting distances are rather low for small values of q.

First, it is worth noting that each recursive step of Algorithm 1 relies on *interleaving* a new set of strings, which contain the newly added symbol $q-1$, into a permutation on $\mathbb{Z}_{q-1}^2$. The permutation among these added strings, and the particular way by which the interleaving is made, are determined by the information vector. In what follows, this intuition is formalized, and the resulting minimum distance is discussed.

Definition 2. *The Kendall-τ distance d_τ between two strings is the minimum number of adjacent transpositions that can be applied on one to obtain the other.*

Although Definition 2 is usually applied over permutations, i.e., for strings which contain all symbols of the alphabet with no repetitions, it may also be applied over ordinary strings.

Example 3. $d_\tau(10010, 00110) = 2$.

For disjoint sets of symbols A and B let $C_A \subseteq S_A$ and $C_B \subseteq S_B$ be codes of minimum Kendall-τ distances d_A and d_B, respectively, and let $D \subseteq \{0,1\}^{|A|+|B|}$ be of constant Hamming weight $|A|$ and minimum Kendall-τ distance d_D. Define the operator $*_D$ as

$$C_A *_D C_B \triangleq \{\pi \in S_{A\cup B} \ : \ \pi|_A \in C_A, \pi|_B \in C_B, f(\pi) \in D\}, \tag{10}$$

where $\pi|_A$ (resp. $\pi|_B$) denotes the result of deleting all symbols that are not in A (resp. $\pi|_B$) from π, and

$$f : S_{A\cup B} \to \{0,1\}^{|A|+|B|}$$
$$f(\pi)_i = \begin{cases} 1 & \pi_i \in A \\ 0 & \pi_i \in B \end{cases}.$$

Lemma 19. *The minimum distance $d_\tau(\mathcal{C})$ of $\mathcal{C} \triangleq C_A *_D C_B$ is at least* $\min\{d_A, d_B, d_D\}$.

Proof: It is readily verified that the mapping $g : S_{A\cup B} \to S_A \times S_B \times \{0,1\}^{|A|+|B|}$, $\pi \mapsto (\pi|_A, \pi|_B, f(\pi))$ is injective. Further, notice that for an adjacent transposition e, if it affects two elements from A then $g(\pi)$ and $g(e(\pi))$ differ only on the S_A component. Similarly, if e affects two elements from B then $g(\pi)$ and $g(e(\pi))$ differ only on the S_B component, and if it affects one element from A and one from B then $g(\pi)$ and $g(e(\pi))$ differ only on the $\{0,1\}^{|A|+|B|}$ component. Hence, for π_1 and π_2 in $\mathcal{C}$,

$$d_\tau(\pi_1, \pi_2) = d_\tau(\pi_1|_A, \pi_2|_A) + d_\tau(\pi_1|_B, \pi_2|_B) + d_\tau(f(\pi_1), f(\pi_2)),$$

from which the claim follows. ∎

To obtain a minimum distance guarantee at the output of Algorithm 1, let C_3 be a rank-modulation code in $T_{3,2}$ (where the notation $T_{q,\ell}$ is as in Section VI) of minimum Kendall-τ distance d_3, for all $i \in \{4, \ldots, q\}$ let C_i be a rank-modulation code in S_i of minimum Kendall-τ distance d_i, and let $B_i \subseteq \{0,1\}^{i^2-i+1}$ be a binary code of constant Hamming weight i and minimum Kendall-τ distance t_i. Replace the information set I_q (5) by the set

$$I'_q \triangleq C_3 \times (C_4 \times B_4) \times \cdots \times (C_q \times B_q). \tag{11}$$

Lemma 20. *If the information vectors in Algorithm 1 are taken from I'_q rather than from I_q, then the minimum distance of the resulting permutations is at least* $\min(\{d_i\}_{i=3}^q \cup \{t_i\}_{i=4}^q)$.

Proof: Let $T'_{q,2}$ be the permutations which result from Algorithm 1 when applied over information vectors from I'_q. The claim is proved by using induction on q. It follows from the definition of the algorithm that $T_{3,2} = C_3$, which proves the claim for $q = 3$. Assume that $d_\tau(T'_{q-1,2}) \geq \min(\{d_i\}_{i=3}^{q-1} \cup \{t_i\}_{i=4}^{q-1})$. Notice that for $q \geq 4$, the set $T'_{q,2}$ is given by considering $T'_{q-1,2} *_{B_q} C_q$, replacing the elements of $[q]$ by the strings which correspond to $y_0, \ldots, y_{q-1}$, and adding the strings which corresponds to $y_0 - (q-2)\varepsilon$ and to $y_i - \varepsilon$ for $1 \leq i \leq q-2$, at fixed positions. Since the addition of the latter strings may only increase the minimum distance in comparison with $T'_{q-1,2} *_{B_q} C_q$, it follows from Lemma 19 that

$$d_\tau(T'_{q,2}) \geq d_\tau(T'_{q-1,2} *_{B_q} C_q) \geq \min\{d_\tau(T'_{q-1,2}), d_\tau(B_q), d_\tau(C_q)\}.$$

Hence, since the induction hypothesis implies that $d(T'_{q-1,2}) \geq \min(\{d_i\}_{i=3}^{q-1} \cup \{t_i\}_{i=4}^{q-1})$, since $d_\tau(B_q) = t_q$, and since $d_\tau(C_q) = d_q$, the result follows. ∎

If follows from Lemma 20 that by pre-coding the information vectors into I'_q, a non-trivial minimum distance guarantee is obtained. Notice that codes of size one, whose minimum distance is infinite, can be used as either of the C_i-s or B_i-s in (11) to increase the minimum distance of the resulting code.

To incorporate similar approach to Algorithm 2, notice that each recursive step of it relies on *splitting* any string $w \in \mathbb{Z}_q^{\ell-1}$ to q strings of the form $D^{-1}(w) \subseteq \mathbb{Z}_q^\ell$. The order between strings in $\mathbb{Z}_q^\ell$ with different D-preimage is consistent with that of their D-ancestors, whereas the order between strings in $\mathbb{Z}_q^\ell$ with identical D-preimage is determined by information vector. To state this intuition in the spirit of (10), the following definitions are given. Let A and B be disjoint sets, and for $u \in S_A$, $v \in S_B$, and $b \in B$, let $h(u, b, v)$ be the permutation on $A \cup (B \setminus \{b\})$ which results from replacing the occurrence of b in v by u.

Example 4. *If* $A = \{1, 2, 3\}, B = \{4, 5, 6\}$, *and* $u = (3, 1, 2)$, $v = (4, 5, 6)$, *then* $h(u, 4, v) = (3, 1, 2, 5, 6)$.

For $C_A \subseteq S_A$, $C_B \subseteq S_B$, and $b \in B$, let $h(C_A, b, C_B) \triangleq \{h(u, b, v) | u \in C_A, v \in C_B\}$. In an inductive manner, for pairwise disjoint sets $\{A_1, \ldots, A_t, B\}$ and distinct $b_1, \ldots, b_t$, let

$$h((C_{A_i})_{i=1}^t, (b_i)_{i=1}^t, C_B) \triangleq h(C_{A_t}, b_t, h((C_{A_i})_{i=1}^{t-1}, (b_i)_{i=1}^{t-1}, C_B)),$$

which reads as replacing b_i in every codeword of C_B by codewords of C_{A_i}, for every $i \in [t]$.

Lemma 21. *Let* $\{A_1, \ldots, A_{|B|}, B\}$ *be disjoint sets such that* $|A_i| \triangleq q$ *for all* i, *let* $C_{A_i} \subseteq S_{A_i}$ *be a code of minimum distance* d_i *for all* i, *and let* $C_B \subseteq S_B$ *be a code of minimum distance* d_B. *The minimum distance* $d_\tau(\mathcal{H})$ *of* $\mathcal{H} \triangleq h((C_{A_i})_{i=1}^{|B|}, (b_i)_{i=1}^{|B|}, C_B)$ *is at least* $\min(\{d_i\}_{i=1}^{|B|} \cup \{q^2 d_B\})$.

Proof: For $\pi \in \mathcal{H}$ and $A \triangleq \cup_{i=1}^{|B|} A_i$, let $r : S_A \to B^{q|B|}$ be such that $r(\pi)_i$ equals b_j for the unique $j \in \{1, \ldots, |B|\}$ for which $\pi_i \in A_j$; intuitively, the function r identifies the index $j \in \{1, \ldots, |B|\}$ of the set A_j from which each symbol π_i in π is taken, and places b_j instead of π_i. It is readily verified that the function $g : S_A \to S_{A_1} \times \ldots \times S_{A_{|B|}} \times B^{q|B|}$, $\pi \mapsto (\pi|_{A_1}, \ldots, \pi|_{A_{|B|}}, r(\pi))$ is injective. Further, notice that for an adjacent transposition e, if it affects two elements from some A_i, then $g(\pi)$ and $g(e(\pi))$ differ only on the S_{A_i} component. On the other hand, if e affects two elements from distinct A_i and A_j, then $g(\pi)$ and $g(e(\pi))$ differ only on the $B^{q|B|}$ component.

Hence, let π_1 and π_2 be codewords in $\mathcal{H}$, and let c_1 and c_2 be the corresponding codewords from C_B from which π_1 and π_2 were generated. The minimal set of adjacent transpositions which differs π_1 from π_2 contains q^2 transpositions for each transposition which differs c_1 from c_2. Furthermore, it contains a unique transposition on A_i for each transposition which differs $\pi_1|_{A_i}$ from $\pi_2|_{A_i}$, for each i. Therefore,

$$d_\tau(\pi_1, \pi_2) = \sum_{i=1}^{|B|} d_\tau(\pi_1|_{A_i}, \pi_2|_{A_i}) + q^2 d_\tau(c_1, c_2),$$

which proves the claim. ∎

Similar to (11), to obtain a rank-modulation code at the output of Algorithm 2, a different set of information vectors is used. Recall that the information vector J_q^ℓ of Algorithm 2 (9) consists of mappings from sets of strings $\mathcal{A}_i$ into $S_{\mathbb{Z}_q}$. Due to Lemma 21, the set $S_{\mathbb{Z}_q}$ can be replaced in J_q^ℓ by some rank-modulation code. However, due to the q^2 factor in the recursive term of Lemma 21, it suffices to replace $S_{\mathbb{Z}_q}$ only in the rightmost entry of J_q^ℓ. To this end, let $C \subseteq S_{\mathbb{Z}_q}$ be a rank-modulation code of minimum distance d, and let

$$P_\ell^C \triangleq \{P | P : \mathcal{A}_i \to C\}, \text{ and}$$
$$J_{q,C}^\ell \triangleq I_q \times \mathcal{P}_3 \times \ldots \times \mathcal{P}_{\ell-1} \times P_\ell^C.$$

Lemma 22. *For* $\ell \geq 3$, *if the information vectors in Algorithm 2 are taken from* $J_{q,C}^\ell$ *rather than from* J_q^ℓ, *then the minimum distance of the resulting permutations is at least* d.

Proof: Let $T'_{q,\ell}$ be the permutations which result from Algorithm 2 when applied over information vectors from $J_{q,C}^\ell$. The claim is proved using induction on ℓ. For $\ell = 3$, the set $T'_{q,3}$ is given by splitting permutations on $\mathbb{Z}_q^2$. Each string $w \in \mathcal{A}_3 \subseteq \mathbb{Z}_q^2$ is replaced by a permutation $P(w)$ on $D^{-1}(w)$ from[4] the code $C_{D^{-1}(w)}$. Subsequently, strings $u \in \mathbb{Z}_q^2 \setminus \mathcal{A}_3$ are replaced with permutations that depend only on the values of $\{P(w) | w \in \mathcal{A}_3\}$. Hence, since removing elements may only decrease the minimum distance, it suffices to bound the minimum distance of the permutations of $\cup_{w \in \mathcal{A}_3} D^{-1}(w)$. The latter is given as

$$h\left((C_W)_{W \in D^{-1}(\mathcal{A}_3)}, (a)_{a \in \mathcal{A}_3}, T_{q,2}\right), \text{ where}$$
$$D^{-1}(\mathcal{A}_3) \triangleq \{D^{-1}(w) | w \in \mathcal{A}_3\}.$$

[4]Note that the q elements of $D^{-1}(w)$ are arranged according to a codeword the code $C \subseteq S_q$, and hence it may be seen as a code $C_{D^{-1}(w)}$ on $D^{-1}(w)$ of identical minimum distance d.

Therefore, according to Lemma 21, it follows that $d_\tau(T'_{q,3}) \geq \min\{d, q^2 \cdot d_\tau(T_{q,2})\} = d$.

For any $\ell \geq 4$, similar arguments show that it suffices to bound the minimum distance of

$$h\left((C_W)_{W\in D^{-1}(\mathcal{A}_\ell)}, (a)_{a\in\mathcal{A}_\ell}, T'_{q,\ell-1}\right), \text{ where}$$
$$D^{-1}(\mathcal{A}_\ell) \triangleq \{D^{-1}(w) | w \in \mathcal{A}_\ell\},$$

and by again by Lemma 21, it follows that $d_\tau(T'_{q,\ell}) \geq \min\{d, q^2 \cdot d_\tau(T'_{q,\ell-1})\} = d$. ■