Reconstruction of Strings from their Substrings Spectrum

Sagi Marcovich
Technion - Israel Institute of Technology
Haifa 3200003, Israel
sagimar@cs.technion.ac.il

Eitan Yaakobi
Technion - Israel Institute of Technology
Haifa 3200003, Israel
yaakobi@cs.technion.ac.il

***Abstract*—This paper studies reconstruction of strings based upon their substrings spectrum. Under this paradigm, it is assumed that all substrings of some fixed length are received and the goal is to reconstruct the sequence. While many existing works assumed that substrings are received error free, we follow in this paper the noisy setup of this problem that was first studied by Gabrys and Milenkovic. The goal of this study is twofold. First we study the setup in which not all substrings in the multispectrum are received, and then we focus on the case where the read substrings are not error free. In each case we provide specific code constructions of strings that their reconstruction is guaranteed even in the presence of failure in either model. We present efficient encoding and decoding maps and analyze the cardinality of the code constructions.**

I. Introduction

In many storage and communication channels it is not possible to receive the transmitted or stored word as one unit, even in its noisy version. Rather, this information can only be provided in other forms such as a list of its subwords, statistics on its symbols, and more. This class of models is usually falls the general framework of the *string reconstruction problems*. There are several instances of this setup, such as the *trace reconstruction problem* [1], the *k-deck problem* [2], [3], [4], the *reconstruction from substring compositions problem* [5], [6] and the *reconstruction problem* by Levenshtein [7].

This paper studies an important setup for this class of problems, where it is assumed that the information about the word is conveyed by the multispectrum of all its substrings of some fixed length. Under this paradigm, the goal is to reconstruct the word and the success of this process usually depends on the length of the read substrings and the stored word. This model of strings reconstruction is motivated by current DNA sequencing technologies and in particular shotgun DNA sequencing [8]. In this method, the DNA strand is broken into multiple fragments, called *reads*, which are then assembled together to reconstruct the strand [9], [10], [11].

Mathematically speaking, for a length-n string $\boldsymbol{w}$ and a positive integer L, its *L-multispectrum*, denoted by $S_L(\boldsymbol{w})$, is the multiset of all its length-L substrings, $S_L(\boldsymbol{w}) = \{\boldsymbol{w}_{1,L}, \boldsymbol{w}_{2,L}, \ldots, \boldsymbol{w}_{n-L+1,L}\}$, where $\boldsymbol{w}_{i,L}$ is the substring $(w_i, w_{i+1}, \ldots, w_{i+L-1})$. Then, the goal is to reconstruct the string $\boldsymbol{w}$ given its multispectrum $S_L(\boldsymbol{w})$. If a string can be uniquely reconstructed from its L-multispectrum, then it is called *L-reconstructible*. It was proved by Ukkonen [12] that if all length-$(L-1)$ substrings of $\boldsymbol{w}$ are different, then the word $\boldsymbol{w}$ is L-reconstructible. A string $\boldsymbol{w}$ that satisfies this constraint is referred as *$(L-1)$-substring unique*. Based upon this property, it was recently proved in [13], [14] that if $L = \lceil a \log(n) \rceil$ for some fixed value of $a > 1$, then the asymptotic rate of all L-reconstructible strings approaches 1.

Several recent papers have taken an information-theoretic point of view to the sequence assembly problem. The goal of these works was to study the fundamental limits of reconstruction of $\boldsymbol{w}$ from $S_L(\boldsymbol{w})$ with a fixed failure probability under different setups and various error models. Arratia et al. [15] studied the limits of any assembly algorithm that recovers $\boldsymbol{w}$ from $S_L(\boldsymbol{w})$ where $\boldsymbol{w}$ is an i.i.d DNA sequence, and later Motahari, Bresler, and Tse [16] studied the case where only a subset of $S_L(\boldsymbol{w})$ is available, while each read begins at a uniformly distributed location of the sequence. They both showed that if the reads are long enough to have no repeats, then reconstruction is possible with high probability. This was then extended in [17] for the case where every read is transferred through a symmetric substitution noisy read channel and in [18] it was assumed that the reads are corrupted by at most some fixed number of edit errors. Moreover, it has been shown that if $\boldsymbol{w}$ satisfies several constraints, which are based on its repeats statistics, then it can be assembled with high probability from $S_L(\boldsymbol{w})$ [19] or a subset of it [20]. Another variation of the sequence assembly problem, which allows to partially reconstruct the sequence $\boldsymbol{w}$, was studied in [21].

In this paper, we follow the recent work by Gabrys and Milenkovic [13] and assume that the L-multispectrum is not received error free, while requiring for reconstruction of $\boldsymbol{w}$ in the *worst case*. We consider two models of this setup. In the first one, it is assumed that not all substrings in the L-multispectrum were read so only a subset of $S_L(\boldsymbol{w})$ is received. The second model assumes that all reads in the L-multispectrum were received, however some of them might be erroneous. An important tool in our constructions uses the set of substring unique strings and we also study its extension. Namely, for fixed L and d, it is said that $\boldsymbol{w}$ is an (L,d)-substring distant string if the Hamming distance between any two of its length-L substrings is at least d. We study the cardinality of this set of words and show an encoding and decoding maps for this constraint.

The rest of the paper is organized as follows. In Section II, we formally define the codes and constraints studied in this paper and review several previous results. In Section III, we study the case where an incomplete multispectrum is received. Section IV studies the setup where some of the read substrings are noisy as well as (L,d)-substring distant strings. Due to the lack of space, some of the proofs in the paper are omitted. These proofs and several more results can be found in the full version of this work in [22].

II. Definitions and Preliminaries

This section formally defines the notations, constraints, and codes studied in the paper. For integers $i, j \in \mathbb{N}$ such that $i \leqslant j$ we denote by $[i,j]$ the set $\{i, i+1, \ldots, j-1, j\}$. We notate by $[i]$ a shorthand for $[1,i]$. For a multiset A, let $|A|$ denote the number of elements in A (with repetitions). For a set A of integers and $a \in A$, $b_A(a)$ denotes the binary representation of the index of a in A using $\lceil \log |A| \rceil$ bits, when the integers are ordered in an increasing order. When A is omitted, it is implied that $A = [n]$, while n will be clear from the context.

Let Σ denote a finite alphabet of size $|\Sigma| = q$, n an integer, and $\boldsymbol{w} \in \Sigma^n$ a string. For two positive integers i and k such that $i + k - 1 \leqslant n$, let $\boldsymbol{w}_{i,k}$ denote the length-k substring

978-1-7281-6432-8/20/$31.00 ©2020 IEEE

of $\boldsymbol{w}$ starting at position i. Additionally, let $\text{Pref}_k(\boldsymbol{w}) = \boldsymbol{w}_{1,k}$, $\text{Suff}_k(\boldsymbol{w}) = \boldsymbol{w}_{n+1-k,k}$ denote the *k-prefix*, *k-suffix* of $\boldsymbol{w}$, respectively. For two strings $\boldsymbol{w}, \boldsymbol{x} \in \Sigma^n$, $d_H(\boldsymbol{w}, \boldsymbol{x})$ is the Hamming distance between $\boldsymbol{w}$ and $\boldsymbol{x}$ and $w_H(\boldsymbol{w})$ is the Hamming weight of $\boldsymbol{w}$. For a multiset $S = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m\} \subseteq \Sigma^n$ of strings, $d_H(S)$ is defined to be the *minimum Hamming of* S, which is the minimum Hamming distance among all pairs of strings in S, i.e., $d_H(S) = \min_{1 \leqslant i < j \leqslant m}\{d_H(\boldsymbol{s}_i, \boldsymbol{s}_j)\}$. For a nonnegative integer $t \leqslant n$, $B_t(\boldsymbol{w})$ denotes the radius-t Hamming ball around $\boldsymbol{w}$, that is, $B_t(\boldsymbol{w}) = \{\boldsymbol{x} \in \Sigma^n \mid d_H(\boldsymbol{w}, \boldsymbol{x}) \leqslant t\}$.

For a string $\boldsymbol{w} \in \Sigma^n$ and a positive integer $L \leqslant n$, the set $S_L(\boldsymbol{w})$ is defined to be the *L-multispectrum of $\boldsymbol{w}$*, which is the *multiset* of all its length-L substrings

$$S_L(\boldsymbol{w}) = \{\boldsymbol{w}_{1,L}, \boldsymbol{w}_{2,L}, \ldots, \boldsymbol{w}_{n-L+1,L}\}.$$

The main family of strings studied in this paper is defined in the next definition.

Definition 1. *A string $\boldsymbol{w} \in \Sigma^n$ is called an (L,d)**-substring distant string** if the Hamming distance of its L-multispectrum is at least d, that is, $d_H(S_L(\boldsymbol{w})) \geqslant d$. For $d = 1$, we refer to an $(L,1)$-substring distance string as an L**-substring unique string**.*

We note that the case of $d = 1$ has also been studied in [14] and was referred as *repeat-free words*.

Example 1. Let $n = 16, L = 8$, and $\boldsymbol{x} = 0100000111011111$, so its L-multispectrum is

$$S_L(\boldsymbol{x}) = \{01000001, 10000011, 00000111, 00001110, \\ 00011101, 00111011, 01110111, 11101111, 11011111\}.$$

The string $\boldsymbol{x}$ is 8-substring unique and in fact is also $(8,2)$-substring distant. However, $\boldsymbol{x}$ is not $(8,3)$-substring distant since $d_H(\boldsymbol{x}_{3,8}, \boldsymbol{x}_{4,8}) = 2$.

The family of (L,d)-substring distant strings and more specifically L-substring unique strings is highly related to the set of reconstructible strings, which is defined next. Namely, a string $\boldsymbol{w} \in \Sigma^n$ is called an ***L*-reconstructible string** if it can be uniquely reconstructed from its L-multispectrum. Hence, $\boldsymbol{w}$ is an L-reconstructible string if for every $\boldsymbol{x} \neq \boldsymbol{w}$ it holds that $S_L(\boldsymbol{w}) \neq S_L(\boldsymbol{x})$. For positive integers n, q, d, L, we denote by $\mathcal{S}_{n,q}(L,d)$ the set of all length-n (L,d)-substring distant strings over Σ^n, where $|\Sigma| = q$. For $d = 1$ we simply denote this value by $\mathcal{S}_{n,q}(L)$. The set of L-reconstructible strings is denoted by $\mathcal{R}_{n,q}(L)$. We also let $S_{n,q}(L,d) = |\mathcal{S}_{n,q}(L,d)|$, $S_{n,q}(L) = |\mathcal{S}_{n,q}(L)|$, and $R_{n,q}(L) = |\mathcal{R}_{n,q}(L)|$. Lastly, in case $q = 2$ we will seldom remove q from these notations.

The following connection between substring unique and reconstructible strings was first established by Ukkonen in [12].

Theorem 2.*[12] If a string $\boldsymbol{x} \in \Sigma^n$ is $(L-1)$-substring unique then it is L-reconstructible.*

According to Theorem 2, it holds that $\mathcal{S}_{n,q}(L-1) \subseteq \mathcal{R}_{n,q}(L)$ and in particular $S_{n,q}(L-1) \leqslant R_{n,q}(L)$.

The opposite direction of Theorem 2 does not always hold. In fact, in [13], an encoding scheme that uses the property from Theorem 2 is used in order to encode L-reconstructible strings that are almost $(L-1)$-substring unique. Recently, two encoding schemes of reconstructible binary strings that are also $(L-1)$-substring unique were proposed in [14]. The first scheme is applied for a window length of $L = 2\lceil\log(n)\rceil + 2$ with a single bit of redundancy, and the second one works for windows of length $L = \lceil a\log(n)\rceil$ for $1 < a \leqslant 2$ and its asymptotic rate approaches 1. According to the first scheme, one can deduce that $S_n(2\lceil\log(n)\rceil + 2) \geqslant 2^{n-1}$ and the second one implies that for all $1 < a \leqslant 2$,

$$\lim_{n\to\infty} \frac{\log_2(S_n(\lceil a\log(n)\rceil))}{n} = 1.$$

This result is also proved directly in [14], by deriving a lower bound on the number of strings in $\mathcal{S}_n(\lceil a\log(n)\rceil)$.

The motivation to study (L,d)-substring distant strings originates from the observation that in many cases the L-multispectrum cannot be read error-free. This translates to a stronger property, such as the one given by (L,d)-substring distant, that strings need to satisfy in order to guarantee unique reconstruction in the presence of errors.

Definition 3. *Let $\boldsymbol{w} \in \Sigma^n$ be a string and $S_L(\boldsymbol{w})$ is its L-multispectrum. A multiset U is called a **t-losses L-multispectrum of** $\boldsymbol{w}$ if $U \subseteq S_L(\boldsymbol{w})$ and $|S_L(\boldsymbol{w})| - |U| \leqslant t$. The **$t$-losses L-multispectrum ball of** $\boldsymbol{w}$, denoted by $\mathcal{B}_{L,t}(\boldsymbol{w})$, is defined to be the multiset*

$$\mathcal{B}_{L,t}(\boldsymbol{w}) = \{U \mid U \text{ is a } t\text{-losses } L\text{-multispectrum of } \boldsymbol{w}\}.$$

Example 2. Let $n, L, \boldsymbol{x}$ be from Example 1. The multiset

$$U_1 = \{10000011, 00000111, 00001110, 01110111, \\ 11101111, 11011111\},$$

which equals to $S_L(\boldsymbol{x}) \setminus \{\boldsymbol{x}_{1,L}, \boldsymbol{x}_{5,L}, \boldsymbol{x}_{6,L}\}$, is a 3-losses L-multispectrum of $\boldsymbol{x}$.

When discussing reconstruction from some lossy multispectrum U of $\boldsymbol{w}$, notice that if successive substrings of $\boldsymbol{w}$ are missing from the start or the end of U then several entries of the input string can be entirely absent from U. Therefore, given the multispectrum U, we define its ***maximal-reconstructible substring***, denoted by $\mathbf{W}_1(U)$, as the largest consecutive substring of $\boldsymbol{w}$ which its data is contained in U. Since there are at most t losses, it is ensured that the length of $\mathbf{W}_1(U)$ is at least $n - t$. Accordingly, the following definition presents the family of strings that will be studied in Section III.

Definition 4. *A string $\boldsymbol{w}$ is called an (L,t)**-reconstructible string** if its maximal-reconstructible substring $\mathbf{W}_1(U)$ can be uniquely reconstructed from any of its t-losses L-multispectrums U in $\mathcal{B}_{L,t}(\boldsymbol{w})$.*

Let $C_{\epsilon,i}$ be such that when reading $M = C_{\epsilon,i}n$ substrings of $\boldsymbol{x}$ uniformly at random with replacement, all but i substrings of $S_L(\boldsymbol{x})$ are read with probability at least $1-\epsilon$. Assume that the complete L-multispectrum $S_L(\boldsymbol{x})$ is required to recover the sequence $\boldsymbol{x}$. The probability that a single length-L substring is not read upon M attempts is

$$P = \left(1 - \frac{1}{n-L+1}\right)^M = \left(1 - \frac{1}{n-L+1}\right)^{C_{\epsilon,0}n} \approx e^{-C_{\epsilon,0}}.$$

Thus, the probability that upon M reads not all substrings in the L-multispectrum $S_L(\boldsymbol{x})$ are read can be upper bounded by the union bound as

$$P_0 \leqslant (n-L+1)P = (n-L+1)e^{-C_{\epsilon,0}} \approx ne^{-C_{\epsilon,0}}.$$

Hence, in order to guarantee success probability of at least $1-\epsilon$, it suffices that $ne^{-C_{\epsilon,0}} \leqslant \epsilon$, i.e., $C_{\epsilon,0} \geqslant \ln(n) + \ln(1/\epsilon)$.

On the other hand, if it is possible to reconstruct the sequence $\boldsymbol{x}$ even in the presence of t losses, it is enough to require $(ne^{-C_{\epsilon,t}})^{t+1} \leqslant \epsilon$ and therefore $C_{\epsilon,t} \geqslant \ln(n) + \frac{\ln(1/\epsilon)}{t+1}$. Hence, if for example $1/\epsilon = \mathcal{O}(n^a)$, then the number of reads can be reduced roughly by a factor of $(t+1)(1 - \frac{t}{a+t+1})$.

III. Reconstructing an Incomplete Multispectrum

In this section, we define constraints for (L,t)-reconstructible strings, propose a reconstruction algorithm for those strings, and analyze the cardinality of such family of strings.

A. Reconstruction Constraints

The goal of this subsection is to construct t-losses L-reconstructible strings. This will be given by strings that satisfy a few constraints, given in the next definition. For simplicity, we consider here only the binary case, so $\Sigma = \{0,1\}$. For the rest of this section, we denote the integers $\ell_1 = L - \lfloor t/3 \rfloor - 1, \ell_2 = L - \lceil 2t/3 \rceil - 1, \ell_3 = L - t - 1$ and the sets $I_2 = [n - \ell_2 - t + 1, n - \ell_2 + 1], I_3 = [n - \ell_3 - t + 1, n - \ell_3 + 1]$.

Definition 5. *A string $\boldsymbol{x} \in \Sigma^n$ is said to satisfy the (n,L,t)-**lossy reconstruction (LREC) constraints** if it fulfills the following three constrains.*

1) *$\boldsymbol{x}$ is a ℓ_1-substring unique string.*
2) *The first and last $t+1$ length-ℓ_2 substrings are not identical to all other length-ℓ_2 substrings. Namely, for all $i \in [t+1], j \in [n-\ell_2+1]$ with $i \neq j$ then $\boldsymbol{x}_{i,\ell_2} \neq \boldsymbol{x}_{j,\ell_2}$ and for all $i \in [n-\ell_2+1], j \in I_2$ with $i \neq j$, then $\boldsymbol{x}_{i,\ell_2} \neq \boldsymbol{x}_{j,\ell_2}$.*
3) *The first $t+1$ length-ℓ_3 substrings are not identical to the last $t+1$ length-ℓ_3 substrings. Namely, for all $i \in [t+1], j \in I_3, \boldsymbol{x}_{i,\ell_3} \neq \boldsymbol{x}_{j,\ell_3}$.*

For n, L, t, denote by $\mathcal{D}_n(L,t)$ the set of all strings that satisfy the (n,L,t)-LREC constraints and let $D_n(L,t) = |\mathcal{D}_n(L,t)|$.

In [13], the authors focused on a type of errors which corresponds to occurrence of bursts of substring losses. They identified a lossy multispectrum $U \subseteq S_L(\boldsymbol{x})$ to have G-maximal coverage gap if G is the maximum number of consecutive substrings that are not included in $S_L(\boldsymbol{x})$. Based on this characterization, they showed that if $\boldsymbol{x}$ is $(L-G-1)$-substring unique it is reconstructible from such a lossy multispectrum U. When applying this constraint to our problem, assume that $U \in \mathcal{B}_{L,t}(\boldsymbol{x})$, then it is necessary that $G = t$ since all the losses can occur consecutively. Based on the results of [14], in order to construct a rate-1 code of (L,t)-reconstructible strings for given n and t, the construction proposed in [13] requires that $L > \lceil a \log(n) \rceil + t$ for some $a > 1$. It will be shown in Section III-C that the (n,L,t)-LREC constraint composes a rate-1 code for values of L that satisfies $L > \lceil a \log(n) \rceil + \lfloor t/3 \rfloor$, where $a > 1 + b/3$ and $t = \lceil b \log(n) \rceil + o(\log(n))$ for $b < 3$. Hence, for these parameters, the construction proposed in this paper imposes a weaker constraint on the value of L than the construction proposed in [13].

B. Reconstruction Algorithm

Our next goal is showing that for every string which satisfies the (n,L,t)-LREC constraint, its maximal-reconstructible substring can be uniquely decoded even if at most some t substrings are not read. Namely, we prove the following theorem.

Theorem 6. *Every string $\boldsymbol{x} \in \mathcal{D}_n(L,t)$ is (L,t)-reconstructible.*

The proof of Theorem 6 is given by an explicit decoding algorithm which receives a multiset $U \in \mathcal{B}_{L,t}(\boldsymbol{x})$ for some $\boldsymbol{x} \in \mathcal{D}_n(L,t)$. First, we present in Algorithm 1, an auxiliary procedure, called the *Stitching Algorithm*, which receives two inputs: 1) A set A of substrings that we aim to stitch, and 2) $\rho \leqslant t$, a parameter that will indicate the minimum overlapping size of two substrings in order to be stitched together. The stitching algorithm is based on iterative stitching steps and is composed of three nested loops. At the most inner loop, two substrings are stitched if the suffix of the first is identical to the prefix of the second. This will later indicate that these substrings originated from the same positions in the input string. The middle loop constructs continuous substrings of U by finding a prefix of such a substring and repeatedly applying the inner loop in order to correctly concatenate to it more bits. The outer loop iterates over $k = 0, \ldots, \rho$ and at every iteration we bridge gaps that were created by losses of k consecutive substrings. The stitching algorithm returns a set of continuous substrings reconstructed from U. We say that an operation of the stitching algorithm is *successful* if the output set size is strictly smaller than the input set size.

Algorithm 1 Stitch(A, ρ)

```
for $k = 0, \ldots, \rho$ do
    $B = \emptyset$
    while $A \neq \emptyset$ do
        pick $\boldsymbol{w} \in A$ such that for every other $\boldsymbol{w}' \in A$, $\mathrm{Pref}_{L-k-1}(\boldsymbol{w}) \neq \mathrm{Suff}_{L-k-1}(\boldsymbol{w}')$
        set $A = A \setminus \{\boldsymbol{w}\}$
        while there exists $\boldsymbol{w}' \in A$ such that $\mathrm{Suff}_{L-k-1}(\boldsymbol{w}) = \mathrm{Pref}_{L-k-1}(\boldsymbol{w}')$ do
            set $\boldsymbol{w} = \boldsymbol{w} \circ \mathrm{Suff}_{|\boldsymbol{w}'|-L+k+1}(\boldsymbol{w}')$
            set $A = A \setminus \{\boldsymbol{w}'\}$
        end while
        set $B = B \cup \{\boldsymbol{w}\}$
    end while
    set $A = B$
end for
return $B$
```

Algorithm 2, called the *Reconstruction Algorithm*, receives a t-losses $L-$multispectrum U for some $\boldsymbol{x} \in \mathcal{D}_n(L,t)$ and uses the stitching algorithm to reconstruct $\mathbf{W}_1(U)$, the maximal reconstructible substring of U. In case the returned set by the reconstruction algorithm consists of a single string we assume that the output is the string itself (i.e. not a set).

Algorithm 2 Reconstruct(U, t)

```
Input: $U \in \mathcal{B}_{L,t}(\boldsymbol{x})$ for some $\boldsymbol{x} \in \mathcal{D}_n(L,t)$
Output: $\mathbf{W}_1(U)$ the maximum reconstructible-substring of $U$
1: Invoke $A_0 = \mathrm{Stitch}(U, \lfloor t/3 \rfloor)$.
2: If $|A_0| = 1$ and $A_0 = \{\boldsymbol{y}\}$: return $\boldsymbol{y}$.
3: If $|A_0| = 2$ and $A_0 = \{\boldsymbol{y}_1, \boldsymbol{y}_2\}$: return $\mathrm{Stitch}(A_0, t)$.
4: If $|A_0| = 3$ and $A_0 = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \boldsymbol{y}_3\}$: for $i = 1,2,3$ invoke $A_i = \mathrm{Stitch}(A_0 \setminus \{\boldsymbol{y}_i\}, \lceil 2t/3 \rceil)$ and if successful invoke $A_i' = \mathrm{Stitch}(A_i \cup \{\boldsymbol{y}_i\}, \lceil 2t/3 \rceil)$. If successful again, return $A_i'$.
```

Let B_k denote the set B after the k-th iteration of the for loop of Algorithm 1. The next example demonstrates how Algorithms 1 and 2 operate.

Example 3. Let $n, L, \boldsymbol{x}, U_1$ from Example 2, so that $U_1 \in \mathcal{B}_{L,t}(\boldsymbol{x})$ and $\boldsymbol{x} \in \mathcal{D}_n(L,t)$ with $t = 3$. Assume that we invoke Reconstruct(U_1, t). First, the algorithm invokes $A_0 = \mathrm{Stitch}(U_1, 1)$. At the first iteration of the for loop where $k = 0$, assume the algorithm picks $\boldsymbol{x}_{2,8} = 10000011$ and stitches to it $\boldsymbol{x}_{3,8} = 00000111$ followed by $\boldsymbol{x}_{4,8} = 00001110$. Next, the algorithm picks $\boldsymbol{x}_{7,8} = 01110111$ and stitches to it $\boldsymbol{x}_{8,8} = 11101111$ followed by $\boldsymbol{x}_{9,8} = 11011111$. Thus, we have at the end of this iteration

$$B_0 = \{\boldsymbol{x}_{2,10}, \boldsymbol{x}_{7,10}\} = \{1000001110, 0111011111\}.$$

No stitching is made at the second iteration for $k = 1$ and thus $A_0 = B_1 = B_0$ is the output of the stitching algorithm. Since $|A_0| = 2$, we execute next in Step 3, Stitch$(A_0, 3)$. Then, the two substrings of A_0 are stitched at iteration $k = 2$, since $\mathrm{Suff}_5(\boldsymbol{x}_{2,10}) = \mathrm{Pref}_5(\boldsymbol{x}_{7,10})$. Eventually, the string $\boldsymbol{x}_{2,15} = 10000011101111 = \mathbf{W}_1(U_1)$ is returned as expected.

C. Cardinality Analysis

Next, we estimate the value of $D_n(L,t)$ for some specific parameters of n, L, t. Our approach is based on the result from [14] which claims that the asymptotic rate of the set $\mathcal{S}_{n,2}(L)$ approaches 1, when $L = \lceil a\log(n)\rceil$ and $a > 1$. Building upon this result, for a given value of t that satisfies $t = \lceil b\log(n)\rceil + o(\log(n))$ for some $0 \leqslant b < 3$, we show how to choose the value of L so the (n,L,t)-LREC constraints hold. This result is proved in the following theorem.

Theorem 7. *If $t = \lceil b\log(n)\rceil + o(\log(n))$ for some $0 \leqslant b < 3$ and $L = \lceil a\log(n)\rceil + \lfloor t/3\rfloor + 1$, where $a > 1 + b/3$, then*

$$\lim_{n\to\infty} \frac{\log_2(D_n(L,t))}{n} = 1.$$

Proof: For t and L stated in the theorem it holds that $L = (a + b/3)\log(n) + o(\log(n))$, $\ell_1 = a\log(n) + o(\log(n))$, and $\ell_2 = (a - b/3)\log(n) + o(\log(n))$. According to [14] it holds that the rate of the set $\mathcal{S}_{n,2}(L')$ when $L' = \lceil a'\log(n)\rceil$ approaches 1 for all $a' > 1$, that is,

$$\lim_{n\to\infty} \frac{\log_2(S_n(L' = \lceil a'\log(n)\rceil))}{n} = 1. \qquad (1)$$

The outline of the proof works as follows. Consider the set $\mathcal{S}_{n',2}(\ell_2)$ for $n' = n - (t + \ell_3)$. By (1), it holds that

$$\lim_{n\to\infty} \frac{\log_2(S_{n-(t+\ell_3)}(\ell_2))}{n-(t+\ell_3)} = \lim_{n\to\infty} \frac{\log_2(S_{n-(t+\ell_3)}(\ell_2))}{n} = 1.$$

Next, we will show that $D_n(L,t) \geqslant S_{n-(t+\ell_3)}(\ell_2)$ and this will conclude the proof. In order to accomplish this result, we will show that every string in $\mathcal{S}_{n',2}(\ell_2)$ can be extended into a length-n string in $\mathcal{D}_n(L,t)$. Let $\boldsymbol{w} \in \mathcal{S}_{n',2}(\ell_2)$, so it is an ℓ_2-substring unique string. We show how to find a string $\boldsymbol{u} \in \Sigma^{t+\ell_3}$ such that $\boldsymbol{w} \circ \boldsymbol{u} \in \mathcal{D}_n(L,t)$, i.e., it satisfies all three (n,L,t)-LREC constraints. In fact we will show how to find $\boldsymbol{u}$ such that $\boldsymbol{w} \circ \boldsymbol{u}$ is ℓ_2-substring unique and it satisfies the third constraint of the (n,L,t)-LREC constraints. First note that the string $\boldsymbol{u}$ has $2^{t+\ell_3}$ optional values. Since the string $\boldsymbol{w} \circ \boldsymbol{u}$ has to be ℓ_2-substring unique, the number of options that are eliminated is at most

$$n \cdot (t + \ell_2 + \ell_3) \cdot 2^{t+\ell_3-\ell_2} = n \cdot (t + \ell_2 + \ell_3) \cdot 2^{\lceil 2t/3\rceil}. \qquad (2)$$

Similarly, the number of eliminated strings by the third constraint is at most

$$(t+1) \cdot (t+1) \cdot 2^t. \qquad (3)$$

Lastly, we have that $2^{t+\ell_3} = 2^{\lceil a\log(n)\rceil + \lfloor t/3\rfloor}$ and by comparing with (2) we get for n large enough

$$2^{\lceil a\log(n)\rceil + \lfloor t/3\rfloor} > n \cdot (t + \ell_2 + \ell_3) \cdot 2^{\lceil 2t/3\rceil}.$$

Moreover, by comparing with (3) it also holds that

$$2^{\lceil a\log(n)\rceil + \lfloor t/3\rfloor} > (t+1) \cdot (t+1) \cdot 2^t$$

since $b < 3$. Thus, it is concluded that such a string $\boldsymbol{u}$ exists. ■

IV. Reconstructing an Erroneous Multispectrum

In this section, we address the problem of reconstructing strings from a multispectrum that suffered substitution errors. This family of multispectrums is formally defined as follows.

Definition 8. *Let $w \in \Sigma^n$ be a string and $S_L(w)$ is its L-multispectrum. A multiset $U = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{n-L+1}\}$ is called a (t,s)**-erroneous** L**-multispectrum of** $\boldsymbol{w}$ if there exists a set of indices $I_e(U) = \{i_1, \ldots, i_m\} \subset [n-L+1]$ where $m \leqslant t$ such that for every $i \in [n-L+1] \setminus I_e(U)$, $\boldsymbol{u}_i = \boldsymbol{w}_{i,L}$ and for every $i \in I_e(U)$, $d_H(\boldsymbol{u}_i, \boldsymbol{w}_{i,L}) \leqslant s$. The (t,s)**-erroneous** L**-multispectrum ball of** $\boldsymbol{w}$, denoted by $\mathcal{B}_{L,t,s}(\boldsymbol{w})$, is defined to be the multiset*

$$\mathcal{B}_{L,t,s}(\boldsymbol{w}) = \{U \mid U \text{ is a } (t,s)\text{-erroneous } L\text{-multispectrum of } \boldsymbol{w}\}.$$

Let $\boldsymbol{w} \in \Sigma^n$ be a string and $U = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{n-L+1}\} \in \mathcal{B}_{L,t,s}(\boldsymbol{w})$ be an erroneous spectrum. Note that if an entry of the input string $\boldsymbol{w}$ can appear in U incorrectly more times than it appears correctly, we are not able to determine its correct value from U. Hence, let $\mathbf{W}_2(U)$ denote the maximum reconstructible-substring of U, a string of length n that takes at every position i the majority value of the occurrences of w_i in U. Namely, for $m \in \mathbb{N}$ we define the function $\text{maj}_m : \Sigma^m \to \Sigma$ that takes a vector $A \in \Sigma^m$ and returns the element $a \in A$ that has the most appearances in A. If there is more than one element of Σ that satisfies this requirement, the function maj_m selects the first element in lexicographic order. For convenience, we omit the parameter m if it is clear from the context, and sometimes refer to A as a multiset instead of as a vector. Thus,

$$\mathbf{W}_2(U) = (w_1, \ldots, w_n) \text{ with } w_j = \text{maj}(U, j),$$

where

$$\text{maj}(U,j) = \text{maj}\{(\boldsymbol{u}_i)_k \mid i \in [n-L+1], k \in [L], i+k-1 = j\}.$$

Definition 9. *A string $\boldsymbol{w}$ is called an (L,t,s)**-reconstructible string** if its maximal-reconstructible substring $\mathbf{W}_2(U)$ can be uniquely reconstructed from any of its (t,s)-erroneous L-multispectrums U in $\mathcal{B}_{L,t,s}(\boldsymbol{w})$.*

In order to have a controlled number of incorrect entries in $\mathbf{W}_2(U)$, we can add the constraint $t < L/2$. This constraint ensures that for every $U \in \mathcal{B}_{L,t,s}(\boldsymbol{w})$, all entries of $\boldsymbol{w}$ besides the first and last $2t$ entries can not appear in U erroneously more times than their appear correctly. Therefore, $\mathbf{W}_2(U)$ satisfies $\mathbf{W}_2(U)_{2t+1,n-4t} = \boldsymbol{w}_{2t+1,n-4t}$.

Our next goal is to establish the following theorem that will be followed by a reconstruction algorithm for erroneous multispectrums of (L,t,s)-reconstructible strings, which is based upon the substring-distant property.

Algorithm 3 Reconstruct(U,t,s)

```
Input: $U \in \mathcal{B}_{L,t,s}(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{S}_n(L-1, 4s+1)$
Output: $\mathbf{W}_2(U)$ the maximum reconstructible-substring of $U$
Initialize $B[1,\ldots,n]$ as an array of $n$ empty vectors, set $i = 1, A = U$
Pick $\boldsymbol{w}_1 \in A$ such that for every other $\boldsymbol{w} \in A$, $d_H(\text{Pref}_{L-1}(\boldsymbol{w}_1), \text{Suff}_{L-1}(\boldsymbol{w})) \geqslant 2s+1$
Set $A = A \setminus \{\boldsymbol{w}_1\}$
For every $j = 1,\ldots,L$, append $(\boldsymbol{w}_1)_j$ to $B[j]$
while $|A| \neq 0$ do
    Pick $\boldsymbol{w}_{i+1} \in A$ such that $d_H(\text{Suff}_{L-1}(\boldsymbol{w}_i), \text{Pref}_{L-1}(\boldsymbol{w}_{i+1})) \leqslant 2s$
    Set $A = A \setminus \{\boldsymbol{w}_{i+1}\}, i = i+1$
    For every $j = 1,\ldots,L$, append $(\boldsymbol{w}_i)_j$ to $B[i+j-1]$
end while
Return $\boldsymbol{y} = (y_1,\ldots,y_n)$ where $y_j = maj(B[j])$
```

Theorem 10. *If a string $\boldsymbol{x} \in \Sigma^n$ is $(L-1, 4s+1)$-substring distant, then it is (L,t,s)-reconstructible.*

The proof of Theorem 10 is given by an explicit reconstruction algorithm, presented in Algorithm 3. The algorithm receives an erroneous multispectrum $U \in \mathcal{B}_{L,t,s}(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{S}_n(L-1, 4s+1)$ and reconstructs the maximum reconstructible substring $\mathbf{W}_2(U)$. The algorithm uses the substring-distant property of $\boldsymbol{x}$ to identify the correct order of the substrings of U.

Algorithm 4 LDEncode($\boldsymbol{w}, L, d$)

```
Input: A string $\boldsymbol{w} \in \Sigma^{n-1}$
Output: A string $\boldsymbol{x} \in S_n(L,d)$
Set $\boldsymbol{x} = \boldsymbol{w} \circ 0 \circ 1^d \circ 0^{\ell-|\boldsymbol{u}_d|} \circ \boldsymbol{u}_d$
Elimination:
while exist indexes $i < j$ such that $d_H(\boldsymbol{x}_{i,L}, \boldsymbol{x}_{j,L}) < d$ or an index $i \leqslant |\boldsymbol{x}| - 2\ell + |\boldsymbol{u}_d|$ where $d_H(\boldsymbol{x}_{i,\ell}, 0^{\ell-|\boldsymbol{u}_d|} \circ \boldsymbol{u}_d) < d$ do
    case 1: violating substrings $\boldsymbol{x}_{i,L}, \boldsymbol{x}_{j,L}$ exists
        if $i, j \in J_1 = [|\boldsymbol{x}| - L - \ell - d, |\boldsymbol{x}| - L + 1]$ ( $\boldsymbol{x}_{i,L}$ contains the suffix $0 \circ 1^d \circ 0^{\ell-|\boldsymbol{u}_d|} \circ \boldsymbol{u}_d$) then
            Remove $\boldsymbol{x}_{i,L-(|\boldsymbol{x}|-L-\ell-d-i)+1}$, append $100 \circ b_{J_1}(i) \circ b_{J_1}(j) \circ EncDist_{L,d-1}(\boldsymbol{x}_{i,L}, \boldsymbol{x}_{j,L})$ to the left of $\boldsymbol{x}$
        else
            Remove $\boldsymbol{x}_{i,L}$, append $101 \circ b(i) \circ b(j) \circ EncDist_{L,d-1}(\boldsymbol{x}_{i,L}, \boldsymbol{x}_{j,L})$ to the left of $\boldsymbol{x}$
        end if
    case 2: a substring $\boldsymbol{x}_{i,\ell}$ with $i < |\boldsymbol{x}| - 2\ell + |\boldsymbol{u}_d|$ such that $d_H(\boldsymbol{x}_{i,\ell}, 0^{\ell-|\boldsymbol{u}_d|} \circ \boldsymbol{u}_d) < d$ exists
        if $i \in J_2 = [|\boldsymbol{x}| - 2\ell - d, |\boldsymbol{x}| - 2\ell + |\boldsymbol{u}_d| - 1]$ ( $\boldsymbol{x}_{i,\ell}$ contains the suffix $0 \circ 1^d \circ 0^{\ell-|\boldsymbol{u}_d|} \circ \boldsymbol{u}_d$) then
            Remove $\boldsymbol{x}_{i,\ell-(|\boldsymbol{x}|-2\ell-d-i)+1}$, append $11 \circ b_{J_2}(i) \circ EncDist_{\ell,d-1}(\boldsymbol{x}_{i,\ell}, 0^{\ell-|\boldsymbol{u}_d|} \circ \boldsymbol{u}_d)$ to the left of $\boldsymbol{x}$
        else
            Remove $\boldsymbol{x}_{i,\ell}$, append $0 \circ b(i) \circ EncDist_{\ell,d-1}(\boldsymbol{x}_{i,\ell}, 0^{\ell-|\boldsymbol{u}_d|} \circ \boldsymbol{u}_d)$ to the left of $\boldsymbol{x}$
        end if
end while
if $|\boldsymbol{x}| \geqslant n$, return $\boldsymbol{x}_{1,n}$
Expansion:
while $|\boldsymbol{x}| < n$ do
    Set
        $$B = \Big( \bigcup_{i \in [1, |\boldsymbol{x}| - \ell]} \mathcal{B}_{d-1}(\boldsymbol{x}_{i,\ell}) \Big) \cup \Big( \bigcup_{i \in [|\boldsymbol{x}| - \ell + 1, |\boldsymbol{x}|]} \mathcal{CB}_{\ell,d-1}(\boldsymbol{x}_{i,|\boldsymbol{x}|-i+1}) \Big)$$
    Pick $\boldsymbol{y} \in \Sigma^\ell \setminus B$ and append $\boldsymbol{x} = \boldsymbol{x} \circ \boldsymbol{y}$
end while
Return $\boldsymbol{x}_{1,n}$
```

Then, it takes for each entry of $\boldsymbol{x}$ the majority vote of its occurrences in U.

The correctness of Algorithm 3, which verifies the correctness of Theorem 10 can be found in [22] as well the proof of the following two lemmas.

Lemma 11. *For* $L = 2\log(n) + (d-1)\log(\log(n)) + \mathcal{O}(1)$ *and* n *large enough, it holds that* $S_n(L,d) \geqslant 2^{n-1}$ *and hence the redundancy of the set* $\mathcal{S}_n(L,d)$ *is at most a single bit.*

Lemma 12. *For fixed* d, $a > 1$, *and* $L = \lceil a\log(n) \rceil$, *it holds that the asymptotic rate of the set* $\mathcal{S}_n(L,d)$ *is 1.*

Lastly in this section, a generic encoding algorithm is presented that uses a single redundancy bit in order to encode length-n strings that are (L,d)-distant, for[1]

$$L = 2\log(n) + 2(d-1)\log(\log(n)) + 4.$$

Note that this value of L is far from the value derived in Lemma 11 only by roughly $(d-1)\log(\log(n))$.

Let $\boldsymbol{w}, \boldsymbol{w}' \in \Sigma^n$ be strings such that $d_H(\boldsymbol{w}, \boldsymbol{w}') \leqslant \rho$ for an integer $\rho \leqslant n$. The construction $EncDist_{n,\rho}(\boldsymbol{w}, \boldsymbol{w}')$ is taken from [23] and encodes the distance between $\boldsymbol{w}, \boldsymbol{w}'$. Let $p_1, \ldots, p_{d_H(\boldsymbol{w},\boldsymbol{w}')}$ denote the indices of the entries which $\boldsymbol{w}, \boldsymbol{w}'$ do not agree upon. For every $i \in [\rho]$ let $\boldsymbol{y}_i \in \Sigma^{\log(n)}$ be set as $b(p_i)$ if $i \leqslant d_H(\boldsymbol{w}, \boldsymbol{w}')$ and $0^{\log(n)}$ otherwise. Thus,

$$EncDist_{n,\rho}(\boldsymbol{w}, \boldsymbol{w}') = \boldsymbol{y}_1 \circ \cdots \circ \boldsymbol{y}_\rho.$$

The output size is independent of $\boldsymbol{w}, \boldsymbol{w}'$ and equals $\rho\log(n)$.

We utilize a marker substring, first introduced in [23], which we notate as a *d-auto cyclic string*. A string $\boldsymbol{u} \in \Sigma^n$ is a d-auto cyclic string, if it satisfies $d_H(\boldsymbol{u}, 0^i \circ \boldsymbol{u}_{1,n-i}) \geqslant d$ for every $1 \leqslant i \leqslant d$. The authors of [23] also presented a construction of such strings of length $d\lceil\log(d)\rceil + 2d$. Let $\boldsymbol{u}_d$ denote a d-auto cyclic string for the rest of this section.

Next, let $\boldsymbol{w} \in \Sigma^k$ for $k \leqslant n$. We want to construct a set of length-n strings, that contains all $\boldsymbol{y} \in \Sigma^n$ that satisfy

$$d_H(\mathrm{Pref}_n(\boldsymbol{w} \circ \boldsymbol{y}), \boldsymbol{y}) \leqslant t, \tag{4}$$

for some $t \leqslant n$. Therefore, let $m = \lceil n/k \rceil$, and set $\mathcal{CB}_{n,t}(\boldsymbol{w}) =$

$$\left\{ \mathrm{Pref}_{1,n}(\boldsymbol{w}_1 \circ \cdots \circ \boldsymbol{w}_m) \;\middle|\; \begin{array}{l} \exists\{t_i\}_{i=1}^m \text{ s.t. } \sum_{i=1}^m t_i \leqslant t, \\ \exists\{\boldsymbol{w}_i\}_{i=0}^m \text{ s.t. } \boldsymbol{w}_0 = \boldsymbol{w} \text{ and} \\ \forall i \in [m], \boldsymbol{w}_i \in \mathcal{B}_{t_i}(\boldsymbol{w}_{i-1}) \end{array} \right\}.$$

The set $\mathcal{CB}_{n,t}(\boldsymbol{w})$ is notated as the *concatenation ball of radius-t* around $\boldsymbol{w}$. One can verify that for every $\boldsymbol{y} \in \Sigma^n$ that satisfies (4), then $\boldsymbol{y} \in \mathcal{CB}_{n,t}(\boldsymbol{w})$.

Algorithm 4 receives a string $\boldsymbol{w} \in \Sigma^{n-1}$, and outputs a string $\boldsymbol{x} \in \mathcal{S}_n(L,d)$. The algorithm shares ideas with the encoding scheme of *repeat-free words* from [14], and consists of two main procedures, elimination and expansion. First, we append to $\boldsymbol{w}$ a marker substring of length $L/2 + d + 1$ that contains the d-auto cyclic string $\boldsymbol{u}_d$, which is used by the decoder to identify the end of the elimination procedure. Then, at the elimination procedure we repeatedly look for substrings of length L that their Hamming distance is less than d. When found, we remove the first of the substrings and encode the occurrence using the function $EncDist_{L,d-1}$. Likewise, we eliminate occurrences of substrings of length $L/2$ that their Hamming distance from the $(L/2)$-suffix of the string is less than d. During this procedure, we ensure that the marker substring located at the suffix of the string remains intact. Later, at the expansion procedure we enlarge the string to length n by inserting substrings of length $L/2$ while making sure that the string remains (L,d)-distant. The value of ℓ in the Algorithm 4 is defined by $\ell = L/2 = \log(n) + (d-1)\log(\log(n)) + 2$.

The correctness of Algorithm 4 as well as explanation for the decoding can also be found in [22]. In [22] we also present another construction for reconstructing from erroneous multi-spectrum, which we had to omit due to the lack of space.

Acknowledgment

This work was partially supported by the United States-Israel BSF grant 2018048.

[1] For simplicity, in this section we drop some of the ceiling notations.

References

[1] T. Batu, S. Kannan, S. Khanna, and A. McGregor, “Reconstructing strings from random traces,” in *Proc. of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, USA*, 2004, pp. 910–918.

[2] M. Dudık and L. J. Schulman, “Reconstruction from subsequences,” *Journal of Combinatorial Theory, Series A*, vol. 103, no. 2, pp. 337–348, 2003.

[3] B. Manvel, A. Meyerowitz, A. Schwenk, K. Smith, and P. Stockmeyer, “Reconstruction of sequences,” *Discrete Mathematics*, vol. 94, no. 3, pp. 209–219, 1991.

[4] A. D. Scott, “Reconstructing sequences,” *Discrete Mathematics*, vol. 175, no. 1-3, pp. 231–238, 1997.

[5] J. Acharya, H. Das, O. Milenkovic, A. Orlitsky, and S. Pan, “On reconstructing a string from its substring compositions,” in *Proc. of the IEEE International Symposium on Information Theory*, Austin, Texas, USA, 2010, pp. 1238–1242.

[6] ——, “String reconstruction from substring compositions,” *SIAM Journal on Discrete Mathematics*, vol. 29, no. 3, pp. 1340–1371, 2015.

[7] V. I. Levenshtein, “Efficient reconstruction of sequences from their subsequences or supersequences,” *Journal of Combinatorial Theory, Series A*, vol. 93, no. 2, pp. 310–332, 2001.

[8] H. Li, “Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences,” *Bioinformatics*, vol. 32, no. 14, pp. 2103–2110, 2015.

[9] C.-S. Chin, D. H. Alexander, P. Marks, A. A. Klammer, J. Drake, C. Heiner, A. Clum, A. Copeland, J. Huddleston, E. E. Eichler, S. W. Turner, and J. Korlach, “Nonhybrid, finished microbial genome assemblies from long-read SMRT sequencing data,” *Nature Methods*, vol. 10, no. 6, pp. 563–569, 2013.

[10] N. Loman, J. Quick, and J. Simpson, “A complete bacterial genome assembled de novo using only nanopore sequencing data,” *Nature Methods*, vol. 12, no. 8, pp. 733–735, 2015.

[11] S. L. Salzberg, “Mind the gaps,” *Nature Methods*, vol. 7, no. 2, pp. 105–106, 2010.

[12] E. Ukkonen, “Approximate string-matching with q-grams and maximal matches,” *Theoretical Computer Science*, vol. 92, no. 1, pp. 191–211, 1992.

[13] R. Gabrys and O. Milenkovic, “Unique reconstruction of coded sequences from multiset substring spectra,” in *Proc. of the IEEE International Symposium on Information Theory, Vail, Colorado, USA*, 2018, pp. 2540–2544.

[14] O. Elishco, R. Gabrys, M. Medard, and E. Yaakobi, “Repeat free codes,” in *Proc. of the IEEE International Symposium of Information Theory*, Paris, France, 2019, pp. 932–936.

[15] R. Arratia, D. Martin, G. Reinert, and M. Waterman, “Poisson process approximation for sequence repeats, and sequencing by hybridization,” *Journal of Computational Biology : a Journal of Computational Molecular Cell Biology*, vol. 3, pp. 425–463, 1996.

[16] A. S. Motahari, G. Bresler, and D. Tse, “Information theory of DNA shotgun sequencing,” *IEEE Transactions on Information Theory*, vol. 59, no. 10, pp. 6273–6289, 2013.

[17] A. Motahari, K. Ramchandran, D. Tse, and N. Ma, “Optimal DNA shotgun sequencing: Noisy reads are as good as noiseless reads,” in *Proc. of the IEEE International Symposium of Information Theory*, Istanbul, Turkey, 2013, pp. 1640–1644.

[18] S. Ganguly, E. Mossel, and M. Racz, “Sequence assembly from corrupted shotgun reads,” in *Proc. of the IEEE International Symposium of Information Theory*, Barcelona, Spain, 2016, pp. 265–269.

[19] G. Bresler, M. Bresler, and D. Tse, “Optimal assembly for high throughput shotgun sequencing,” *BMC Bioinformatics*, vol. 14, 2013.

[20] I. Shomorony, T. Courtade, and D. Tse, “Do read errors matter for genome assembly?” in *Proc. of the IEEE International Symposium of Information Theory*, Hong Kong, 2015, pp. 919–923.

[21] I. Shomorony, G. Kamath, F. Xia, T. Courtade, and D. Tse, “Partial DNA assembly: A rate-distortion perspective,” in *Proc. of the IEEE International Symposium of Information Theory*, Barcelona, Spain, 2016, pp. 1799–1803.

[22] S. Marcovich and E. Yaakobi, “Reconstruction of strings from their substrings spectrum,” *Arxiv*, 2019.

[23] M. Levy and E. Yaakobi, “Mutually uncorrelated codes for DNA storage,” *IEEE Transactions on Information Theory*, vol. 65, no. 6, pp. 3671–3691, 2019.