Substring Density Estimation from Traces

Kayvon Mazooji, Ilan Shomorony
Department of Electrical and Computer Engineering
University of Illinois, Urbana-Champaign
mazooji2@illinois.edu, ilans@illinois.edu

Abstract

In the trace reconstruction problem, one seeks to reconstruct a binary string s from a collection of traces, each of which is obtained by passing s through a deletion channel. It is known that $\exp(\tilde{O}(n^{1/5}))$ traces suffice to reconstruct any length-n string with high probability. We consider a variant of the trace reconstruction problem where the goal is to recover a "density map" that indicates the locations of each length-k substring throughout s. We show that $\epsilon^{-2} \cdot \mathrm{poly}(n)$ traces suffice to recover the density map with error at most ϵ. As a result, when restricted to a set of source strings whose minimum "density map distance" is at least $1/\mathrm{poly}(n)$, the trace reconstruction problem can be solved with polynomially many traces.

1 Introduction

In the trace reconstruction problem, there is an unknown binary string $s \in \{0,1\}^n$, which we wish to reconstruct based on T subsequences (or traces) of s. Each trace is obtained by passing the *source string* s through a deletion channel, which deletes each bit of s independently with probability p. The main question of interest is how many traces are needed in order to reconstruct s correctly.

This problem was originally proposed by Batu et al. [1], motivated by problems in sequence alignment, phylogeny, and computational biology [2]. Most of the work on the trace reconstruction problem has focused on characterizing the minimum number of traces needed for reconstructing the source string s exactly. The most common formulation of the problem, known as *worst-case trace reconstruction* [3], requires the reconstruction algorithm to recover $s \in \{0,1\}^n$ exactly with high probability as $n \to \infty$ for *any* string $s \in \{0,1\}^n$. While this problem has received considerable attention, there is still a significant gap between upper and lower bounds on the number of traces needed. Currently, the best lower bound is $\tilde{\Omega}(n^{3/2})$, while the best upper bound is $\exp(\tilde{O}(n^{1/5}))$, both due to Chase [4, 5].

The exponential gap between the best known lower and upper bounds has motivated the formulation of several variants of the trace reconstruction problem where tighter bounds can hopefully be obtained. For example, in the *average-case trace reconstruction* problem, s is assumed to be drawn uniformly at random from all $\{0,1\}^n$ strings. In this case, it is known that only $T = \exp(O(\log^{1/3}(n)))$ traces are sufficient [6]. An *approximate trace reconstruction* problem, where a fraction of the recovered bits is allowed to be incorrect, has also been formulated [7], and the problem of finding the maximum likelihood sequence s from a small number of traces (possibly insufficient for exact reconstruction) has been recently studied [8]. We can also consider a more modest goal than the reconstruction of the source sequence s itself. One example that is particularly relevant to our discussion is the reconstruction of the k-subword deck of s [9, 10].

The k-subword deck of a binary sequence $s \in \{0,1\}^n$ is the multiset of all length-k substrings, i.e., $\{s[i : i+k-1] : i = 1, \dots, n-k+1\}$. Equivalently, the k-subword deck can be defined by the counts $N_{s,x}$ of the number of times x appears in s as a substring:

$$\mathcal{S}_k(s) = [N_{s,x} : x \in \{0,1\}^k]. \tag{1}$$

(a)

$s = 0\,0\,0\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,0\,0\,0\,1\,1\,1$

$\mathcal{S}_4(s) = \{$0000:6, 0001:3, 0011:3,
0111:3, 1110:2, 1100:2, 1000:2$\}$

(b)

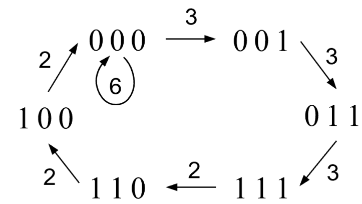


Fig. 1: (a) Example of a source binary string s and its k-subword deck (or k-spectrum) $\mathcal{S}_k(s)$, for $k = 4$. (b) Given $\mathcal{S}_4(s)$ in (a), one can build a de Bruijn graph where the elements in $\mathcal{S}_4(s)$ correspond to edges (with multiplicities) and the nodes correspond to 3-mers. Notice that s corresponds to an Eulerian path on the de Bruijn graph, but such a path is not unique; for example, $s' = 000111000111000000000111$ corresponds to another Eulerian path.

As shown in [10], for $k = O(\log n)$, the k-subword deck $\mathcal{S}_k(s)$ can be recovered with poly(n) traces. The k-subword deck of a sequence is an important object in bioinformatics, with applications in error correction [11, 12], sequence assembly [13, 14], and genomic complexity analysis [15, 16]. In these contexts, the k-subword deck $\mathcal{S}_k(s)$ is often referred to as the k-spectrum, and each length-k substring is called a k-mer. Intuitively, as long as k is large enough, the k-subword deck can uniquely determine the source sequence s. In fact, a classical result by Ukkonen [17] provides a necessary and sufficient condition for $\mathcal{S}_k(s)$ to uniquely determine s based on the length of the "interleaved repeats" in s [18]. In particular, if there are no repeats of length $k-1$ in s, one can reconstruct s from $\mathcal{S}_k(s)$ by simply merging k-mers with a prefix-suffix match of length $k-1$. More generally, given $\mathcal{S}_k(s)$, one can build the *de Bruijn graph*, where nodes correspond to $(k-1)$-mers and edges correspond to k-mers, and s is guaranteed to be an *Eulerian path* in the graph [13, 19]. This is illustrated in Figure 1.

While the k-subword deck is a natural intermediate goal towards the reconstruction of s (and can be recovered with only poly(n) traces), it does not capture all the information present in the traces. For example, the k-subword deck $\mathcal{S}_k(s)$ in Figure 1 also admits the reconstruction $s' =$ 000111000111000000000111, even though s' should be easy to distinguish from s based on traces (by estimating the length of the second and third runs of zeros). Motivated by this shortcoming of the k-subword deck, we propose the idea of a k-mer density map, as a kind of localized k-subword deck where, in addition to knowing the number of times a given k-mer appears in s, we have some information about where it occurs.

For a k-mer $x \in \{0,1\}^k$, let $\mathcal{I}_{s,x} \in \{0,1\}^{n-k+1}$ be the indicator vector of the occurrences of x in s; i.e., $\mathcal{I}_{s,x}[j] = \mathbb{1}\{s_{j:j+k-1} = x\}$, as illustrated in Figure 2. Notice that recovering the k-subword deck can be seen as recovering $\sum_j \mathcal{I}_{s,x}[j]$ for each $x \in \{0,1\}^k$. Also notice that recovering s is equivalent to recovering $\mathcal{I}_{s,x}$ for all $x \in \{0,1\}^k$. A k-mer density map can be obtained by computing

$$K_{s,x}[i] = \sum_{j=1}^{n-k+1} h(i,j)\mathcal{I}_{s,x}[j] \tag{2}$$

for some "smoothing kernel" $h(i,j)$, as illustrated in Figure 2. Intuitively, for a given x, $K_{s,x}$ gives a coarse indication of the occurrences of x in s. Moreover, if h is such that $\sum_i h(i,j) = 1$ for each j, it holds that

$$\sum_i K_{s,x}[i] = \sum_i \sum_j h(i,j)\mathcal{I}_{s,x}[j] = \sum_j \mathcal{I}_{s,x}[j] \sum_i h(i,j) = \sum_j \mathcal{I}_{s,x}[j],$$

which means that the k-subword deck $\mathcal{S}_k(s)$ is a function of $K_{s,x}$, and the density map $K_{s,x}$ can be thought of as a generalization of the k-subword deck that provides information about k-mer location.

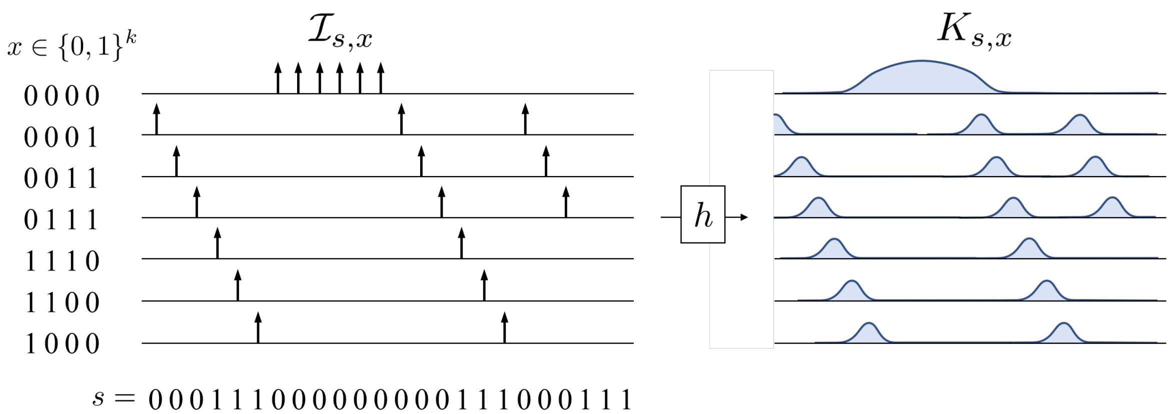


Fig. 2: For each $x \in \{0,1\}^k$, $\mathcal{I}_{s,x}$ indicates the occurrences of x in s. The density map $K_{s,x}$ can be obtained via $K_{s,x}[i] = \sum_{j=1}^{n-k+1} h(i,j)\mathcal{I}_{s,x}[j]$.

We will focus on a specific choice of $h(i,j)$ that will render $K_{s,x}$ easier to estimate from the traces. We will let $h(i,j)$ be the probability that a binomial random variable with $j-1$ trials and probability parameter $1-p$ is equal to $i-1$; i.e., $h(i,j) = \binom{j-1}{i-1}(1-p)^{i-1}p^{j-i}$. This is also the probability that the jth bit of s (if not deleted) ends up as the ith bit of a trace. Hence we have

$$K_{s,x}[i] = \sum_{j=1}^{n-k+1} \binom{j-1}{i-1}(1-p)^{i-1}p^{j-i}\mathcal{I}_{s,x}[j]. \tag{3}$$

for $i \in \{1,\dots,n-k+1\}$. Notice that the maximum value of $h(i,j)$ for a fixed j occurs when $i \approx j(1-p)$ so the kernel $h(\cdot,j)$ has its peak shifted to the left and $K_{s,x}$ is a density map of occurrences of x in s shifted to the left. Operationally, $(1-p)^k K_{s,x}[i]$ is the probability that a fully preserved copy of x in s appears in position i on a given trace of s.

We define the k-mer density map of s as $K_s = [K_{s,x} : x \in \{0,1\}^k]$ (the concatenation of all vectors $K_{s,x}$). If the k-mer density map K_s is known *exactly*, s can also be recovered exactly. This can be seen by noticing the invertibility of the upper-triangular matrix that transforms the binary vector $\mathcal{I}_{s,x}$ into the vector $K_{s,x}$ (for a fixed x). While invertible, this matrix is ill-conditioned, making the transformation from $K_{s,x}$ to $\mathcal{I}_{s,x}$ sensitive to noise in $K_{s,x}$.

We present an algorithm that, given T traces, constructs an estimate $\hat{K}_s$ for the k-mer density map. Our main result establishes that we can achieve estimation error

$$\|\hat{K}_s - K_s\|_\infty = \max_{x,i} |\hat{K}_{s,x}[i] - K_{s,x}[i]| < \epsilon$$

using $T = \epsilon^{-2} \cdot \text{poly}(n)$ traces. Hence, the density map K_s can be estimated with maximum error $\epsilon_n = 1/g(n)$ for $g(n) \in \text{poly}(n)$ using polynomially many traces. In particular, given a set of candidate source strings $\mathcal{A} \subset \{0,1\}^n$ such that, for any $s, s' \in \mathcal{A}$,

$$\|K_s - K_{s'}\|_\infty \geq 2\epsilon,$$

the true source sequence $s \in \mathcal{A}$ can be recovered with $\epsilon^{-2} \cdot \text{poly}(n)$ traces. This adds to the existing literature on classes of strings recoverable/distinguishable with polynomially many traces [20–22].

Since $\mathcal{I}_{s,x}$ and $K_{s,x}$ are related through an invertible (albeit ill-conditioned) linear transformation $K_{s,x} = F\mathcal{I}_{s,x}$, the approximate recovery of the k-mer density map $\hat{K}_{s,x}$ suggests natural reconstruction algorithms for $\mathcal{I}_{s,x}$, e.g., based on a regularized least squares problem

$$\min_{\hat{\mathcal{I}}_{s,x}} \|\hat{K}_{s,x} - F\hat{\mathcal{I}}_{s,x}\|_2^2 + \delta\|\hat{\mathcal{I}}_{s,x}\|_2^2,$$

which is a convex program if $\hat{\mathcal{I}}_{s,x}$ is allowed to be real-valued. The solution $\hat{\mathcal{I}}_{s,x}$ can then be converted into a reconstructed string $\hat{s} \in \{0,1\}^n$ through a majority voting across candidate k-mers for each position. Hence, in contrast to much of the theoretical literature on the trace reconstruction problem, the k-mer density map leads to new reconstruction approaches.

Our main result relies on a nontrivial estimator for $K_{s,x}$ that simultaneously uses count information for all binary strings y that are supersequences of x. The analysis of the estimator is based on the application of a known result in the combinatorics of strings, and an application of McDiarmid's inequality to prove the estimator is successful with high probability. This is different in flavor from recent results on the trace reconstruction problem based on complex analysis [3–6, 23]. Our techniques also lead to an improvement on a previously known upper bound [10] on the number of traces needed for reconstructing the k-subword deck of s for $p < 0.5$.

A. Related work

The current best upper bounds for worst-case trace reconstruction [3, 5, 23] are obtained using algorithms that consider each pair of possible source strings $y, z \in \{0,1\}^n$, and decide whether the set of traces looks more like a set of traces from y, or more like a set of traces from z (we formalize this shortly). Then if there are enough traces, with high probability the true source string s will be the unique string such that the set of traces looks more like it came from s than from any other string in $\{0,1\}^n$. Therefore, the trace reconstruction problem is closely related to the trace-distinguishing problem [21, 22], where we want to decide from the set of traces whether the source string is either y or z with high probability. The best existing upper bound of $\exp(O(n^{1/5}))$ for worst-case trace reconstruction [5] is proved using the fact that any two strings can be distinguished using $\exp(O(n^{1/5}))$ traces. It has also been shown that string pairs at constant Hamming distance can be distinguished using poly(n) traces by McGregor, Price and Vorotnikova [20], and separately by Grigorescu, Sudan, and Zhu using different techniques [21]. It was recently shown that strings at constant Levenshtein distance can be distinguished using poly(n) traces by Sima and Bruck [22].

To distinguish between two strings y and z from a set of traces, current state of the art algorithms identify a function $f_{y,z}$ such that $|E[f_{y,z}(\tilde{Y})] - E[f_{y,z}(\tilde{Z})]|$ is sufficiently large where $\tilde{Y}$ denotes a trace of y. Given T traces $\tilde{S}_1, \dots, \tilde{S}_T$ of a source string s, $E[f_{y,z}(\tilde{S})]$ can be estimated as $\frac{1}{T}\sum_{i=1}^T f_{y,z}(\tilde{S}_i)$ and we say that y *beats* z if $\frac{1}{T}\sum_{i=1}^T f_{y,z}(\tilde{S}_i)$ is closer to $E[f_{y,z}(\tilde{Y})]$ than to $E[f_{y,z}(\tilde{Z})]$. Observe that if $f_{y,z}$ is such that $|E[f_{y,z}(\tilde{Y})] - E[f_{y,z}(\tilde{Z})]|$ is large enough, then assuming the source string is y or z, we can distinguish between the two cases given a reasonable number of traces. If there is a unique string u such that u beats all other strings, then u is output as the reconstruction.

The first such function $f_{y,z}$ introduced by Nazarov and Peres [23], and independently by De, O'Donnell, and Servedio [3], is simply the value of a single bit in the trace, i.e., $f_{y,z}(\tilde{S}) = \tilde{S}[i]$ for source string s and some index i that depends on y and z. An argument based on an existing result on complex-valued polynomials was used to show that for any pair of strings y, z, there is some index $i_{y,z}$ such that $|E[\tilde{Y}[i_{y,z}]] - E[\tilde{Z}[i_{y,z}]]| \geq \exp\left(-O(n^{1/3})\right)$. Using this result along with a standard concentration inequality, the upper bound of $\exp\left(O(n^{1/3})\right)$ traces is obtained. This choice of $f_{y,z}$ is known in the literature as a single bit statistic. The current best upper bound by Chase [5] picks a more complicated function $f_{y,z}$ that involves multiple bits, and proves a novel result on complex polynomials in order to prove a gap of $|E[f_{y,z}(\tilde{Y})] - E[f_{y,z}(\tilde{Z})]| \geq \exp\left(-\tilde{O}(n^{1/5})\right)$, which yields an upper bound of $\exp\left(\tilde{O}(n^{1/5})\right)$ traces. A choice of $f_{y,z}$ that uses multiple bits in combination (e.g. that of Chase [5]) is known in the literature as a multi-bit statistic. There are similarities between the approach used in [5] and the approach used in this paper, and it may be possible to obtain results similar to those in this paper based on the complex analysis techniques explored in [3, 5, 23].

Obtaining lower bounds for the number of traces needed in the trace reconstruction problem amounts to proving a lower bound on the number of traces required to distinguish two strings (trace-distinguishing problem) since any algorithm to solve the trace reconstruction problem can be used to solve the trace-distinguishing problem. For a string a, let a^i denote the string where a is repeated i times. The current best lower bound of $\tilde{\Omega}(n^{3/2})$ for trace reconstruction discovered by Chase [4] is obtained by analyzing the string pair $(01)^m 1(01)^{m+1}$ and $(01)^{m+1}1(01)^m$ where $n = 2m + 3$.

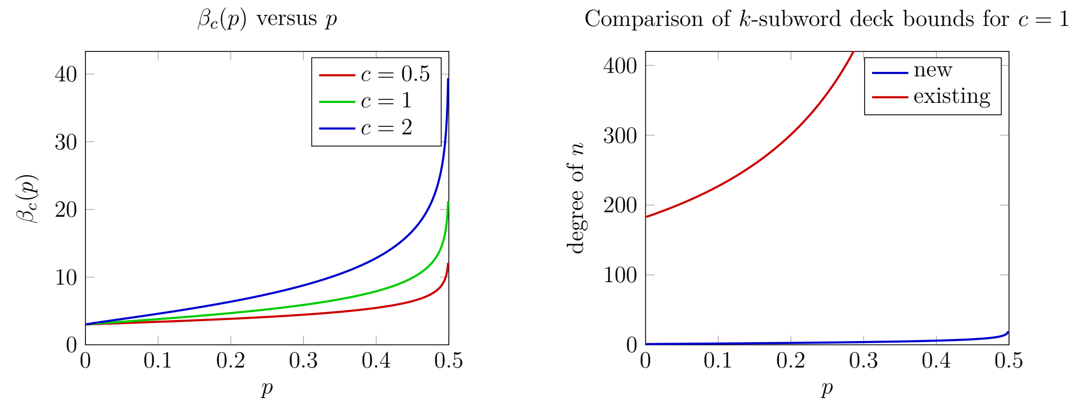
The trace reconstruction problem has also been considered in the smoothed complexity model by Chen, De, Lee, Servedio, and Sinha [10], in which a worst-case string is passed through a binary symmetric channel before trace generation, and the noise-corrupted string needs to be reconstructed with high probability (recall that a binary symmetric channel flips each bit in the string independently with some fixed probability). Chen et al. proved that in the smoothed complexity model, trace reconstruction requires $\text{poly}(n)$ traces. This result relies on the simple fact that if there are no repeated substrings of length $k-1$ or greater in the source string s, then the k-subword deck uniquely determines s. The authors prove that for an arbitrary string, the $(\log n)$-subword deck can be reconstructed with high probability using $\text{poly}(n)$ traces, and prove there will not be any repeats in the source string of length at least $(\log n - 1)$ with high probability after it is passed through a binary symmetric channel, thereby proving that $\text{poly}(n)$ traces suffice for trace reconstruction in the smoothed complexity model.

The bulk of the work done in [10] is proving that the $(\log n)$-subword deck can be reconstructed with high probability using $\text{poly}(n)$ traces for any $p < 1$. In order to prove this, a formula for the number of times a substring x is present in the source string s is derived in terms of the expected number of times a trace contains x, and the expected number of times a trace contains supersequences of x. The expected values of these statistics are then estimated from the set of traces in order to estimate the number of times x appears in s. Concentration inequalities are used to prove that the number of times x appears is estimated correctly with high probability. Narayanan and Ren [24] provide a similar result on reconstructing the $(100 \log n)$-subword deck for circular strings in $\text{poly}(n)$ time. Our result showing that the $(\log n)$-subword deck can be computed using $\text{poly}(n)$ traces with high probability for $p < 0.5$ was originally proved independently, without knowledge of [10] and [24].

B. *Notation*

Strings in this paper are binary and indexed starting from 1. If the index i is negative, $x[i]$ is the $(-i)$th element starting from the right end of x. For example, if $s = 1001$, then $s[1] = 1$, $s[2] = 0$, $s[-1] = 1$, and $s[-2] = 0$. Let $s \in \{0,1\}^n$ be the length-n string we are trying to recover. The string s will be called the *source string*. A *trace* of s is denoted by $\tilde{S}$, and is generated by deleting each bit of s independently with probability p.

For a given string x, we let $|x|$ denote the length of x. For a string a and integer r, a^r denotes the string formed by concatenating r copies of a. A *subsequence* of x is a string that can be formed by deleting elements from x, and a *supersequence* of x is a string that can be formed by inserting elements into x. This is in contrast to a *substring* of x, which is a string that appears in x. We let $x[i,j] = (x[i], x[i+1], \ldots, x[j])$ be the substring of x the begins at position i and ends at position j. For example, if $s = 10101$, then $s[1:3] = 101$ and $s[2:-2] = 010$. For two stings x and y, the number of ways we can make $|y| - |x|$ deletions on y to form x is denoted by $\binom{y}{x}$ [8, 25]. This is a generalization of the binomial coefficient for strings. Observe that $\binom{y}{x} \le \binom{|y|}{|x|}$. To simplify our notation, for strings x, y, we will also let $\binom{y}{x}' = \binom{y[2:-2]}{x[2:-2]}$.



(a) (b)

Fig. 3: (a) Plot of $\beta_c(p)$ for various c. Observe that as p increases, the algorithm requires more traces to achieve the same level of performance. Similarly, as c increases, more traces are needed. For all values of c, the limit of $\beta_c(p)$ as $p \to 0$ is equal to 3. (b) Plot of exponent in (6) versus plot of exponent in (7) for $c = 1$. Observe that our bound is significantly tighter than the existing bound.

2 Main Results

Let $s \in \{0,1\}^n$ denote the source string and $x \in \{0,1\}^k$ denote the target k-mer, whose density $K_{s,x}$ we wish to estimate. To simplify the notation, we fix a constant $c > 0$ and define

$$f_c(n) = \frac{\left(1 + 2n^{\alpha_c(p)}\right)^2}{2n^{2c\log(1-p)-1}} \quad \text{and} \quad \alpha_c(p) = 1 + c\log\left(\frac{1-p}{p}\right) + \frac{cH(1-\frac{p}{1-p}) + c\log\left(\frac{p}{1-p}\right)}{1-\frac{p}{1-p}} \tag{4}$$

where $H(\cdot)$ is the binary entropy function. The function $f_c(n)$ can be upper bounded by a polynomial of degree $\beta_c(p) = 2\alpha_c(p) - 2c\log(1-p) + 1$, which can be numerically computed as shown in Figure 3(a). The following theorem is proved in Section 3.

Theorem 1. *Suppose $p < 0.5$ and $k = c\log n$. Given $\log\left(\frac{2}{\delta}\right) \cdot \epsilon_n^{-2} \cdot f_c(n)$ traces, an estimator $\hat{K}_{s,x}[i]$ for the ith entry of $K_{s,x}$ can be constructed so that $|\hat{K}_{s,x}[i] - K_{s,x}[i]| < \epsilon_n$ with probability $1-\delta$. Moreover, given $\log\left(\frac{2n^{1+c\log 2}}{\delta}\right) \cdot \epsilon_n^{-2} \cdot f_c(n)$ traces, an estimator for the entire density map $\hat{K}_s$ can be constructed so that $\|\hat{K}_s - K_s\|_\infty < \epsilon_n$ with probability $1-\delta$.*

In particular, Theorem 1 implies that for $p < 0.5$ and $\epsilon_n = 1/g(n)$ where $g(n) \in \text{poly}(n)$, all entries of the $(c\log n)$-mer density map $K_{s,x}$ can be estimated with error at most ϵ_n using poly(n) traces. Theorem 1 also implies that the trace reconstruction problem restricted to a set of binary strings with a bounded minimum density map distance can be solved with poly(n) traces.

Corollary 1. *Suppose $p < 0.5$ and $k = c\log n$, and let $\mathcal{A} \subset \{0,1\}^n$ be such that, for any $s, s' \in \mathcal{A}$,*

$$\|K_s - K_{s'}\|_\infty \geq 2\epsilon_n.$$

Given $\log\left(\frac{2n^{1+c\log 2}}{\delta}\right) \cdot \epsilon_n^{-2} \cdot f_c(n)$ traces from some source string $s \in \mathcal{A}$, s can be correctly identified with probability $1-\delta$.

Consequently, the trace reconstruction problem restricted to a set of binary strings with minimum density map distance $1/g(n)$ where $g(n) \in \text{poly}(n)$ can be solved with poly(n) traces. We point out that it is fairly nontrivial to find a pair of strings s and s' so that $\|K_s - K_{s'}\|_\infty$ is exponentially small in n and, to the best of our knowledge, no explicit example of s and s' is known.

Recall that, from [10], one can recover the $(c\log n)$-subword deck using poly(n) traces with high probability. A pair of strings s, s' can be distinguished based on their $(c\log n)$-subword deck alone if and only if their $(c\log n)$-subword decks are distinct, which is equivalent to requiring

$$|N_{s,x} - N_{s',x}| = |\|K_{s,x}\|_1 - \|K_{s',x}\|_1| \geq 1$$

for some $x \in \{0,1\}^{c\log n}$, since $N_{s,x} = \|K_{s,x}\|_1$. In contrast, our main result implies that as long as $|\|K_{s,x}\|_1 - \|K_{s',x}\|_1| \geq 1/\text{poly}(n)$ for some x, s and s' can be distinguished with poly(n) traces, since due to the equivalence of ℓ_∞ and ℓ_1 norms and the reverse triangle inequality,

$$\|K_{s,x} - K_{s',x}\|_\infty \geq \frac{1}{n}\|K_{s,x} - K_{s',x}\|_1 \geq \frac{1}{n}|\|K_{s,x}\|_1 - \|K_{s',x}\|_1|. \tag{5}$$

This further establishes the k-mer density map as a generalization of the k-subword deck.

A special case of Corollary 1 with an explicit condition is given below and proved in the appendix.

Corollary 2. *Suppose $p < 0.5$ and $k = c\log n$. If the strings s, s' are such that $x \in \{0,1\}^k$ begins at position i in s and x does not appear in s' at an index in the range $[i - f(n), i + f(n)]$ for $f(n) = \Omega(n^a)$ where $a > 0.5$, then s can be distinguished from s' with high probability using poly(n) traces.*

Corollaries 1 and 2 are an addition to the literature on conditions for distinguishing strings from traces, which includes the facts that strings at constant Hamming distance can be distinguished using poly(n) traces [20, 21], and strings at constant Levenshtein distance can be distinguished using poly(n) traces [22].

Improved upper bound for k-subword deck reconstruction: Using our proof technique for estimating $K_{s,x}$, we also give a novel proof[1] that the $(c\log n)$-subword deck can be reconstructed using poly(n) traces for $p < 0.5$, which yields an improved upper bound on the required number of traces compared to the analysis of the algorithm for $p < 0.5$ in [10].

Theorem 2. *For $p < 0.5$, we can reconstruct the $(c\log n)$-subword deck of any source string $s \in \{0,1\}^n$ from*

$$\tilde{O}\left(n^{1+c\left(\frac{(1-p)H(1-p/(1-p))+p\log(p/(1-p))}{1/2-p}+2\log\left(\frac{1}{1-p}\right)\right)}\right) \tag{6}$$

traces with high probability.

In contrast, the analysis in [10] proves that

$$\tilde{O}\left(n^{4+12c\left(\frac{e^2}{1/2-p}\right)+c\log(4)}\right) \tag{7}$$

traces are sufficient for this task. Comparing the exponents for $p < 0.5$, we prove in the appendix that

$$\begin{aligned}1 + c\left(\frac{(1-p)H(1-p/(1-p))+p\log(p/(1-p))}{1/2-p}+2\log\left(\frac{1}{1-p}\right)\right) \\ < 4 + 12c\left(\frac{e^2}{1/2-p}\right) + c\log(4)\end{aligned} \tag{8}$$

which shows that asymptotically, our upper bound is tighter for any c and any $p < 0.5$. See Figure 3(b) for a plot showing the comparison. In particular, for c close to zero, (6) is close to linear in n, nearly matching the following lower bound.

[1] This proof was discovered independently of [10] in 2021.

Theorem 3. *For deletion probability p and any source string $s \in \{0,1\}^n$, we have that $\Omega(np(1-p))$ traces are necessary for recovering the $g(n)$-subword deck for any function g such that $g(n) \geq 2$.*

Proof: Suppose we want to decide whether the k-subword deck of the source string is the k-subword deck of $s_1 = 1^{n/2}0^{n/2}$, or whether it is the k-subword deck of $s_2 = 1^{(n/2)+1}0^{(n/2)-1}$. Observe that for any $k \geq 2$, the only length n string that has the k-subword deck of s_1 is s_1 itself, and likewise, the only length n string that has the k-subword deck of s_2 is s_2 itself. Thus, distinguishing between these two k-subword decks is equivalent to distinguishing between s_1 and s_2. From Section 4.2 of [1], we know that $\Omega(np(1-p))$ traces are necessary for distinguishing between s_1 and s_2. ■

We did not compare our upper bound on trace complexity to the analysis of the algorithm in [10] that reconstructs the $(\log n)$-subword deck using poly(n) traces for $p < 1$ because an explicit upper bound for this algorithm has not appeared in the literature to the best of our knowledge. As discussed in Section 4, we also prove in the appendix that the initial algorithm for k-subword deck reconstruction presented in [10] only needs poly(n) traces for reconstructing the $(\log n)$-subword deck of a string for $p < 0.5$, which was previously unknown.

3 An Estimator for the k-mer Density Map

In this section, we describe our estimator for the k-mer density map and prove Theorem 1. We first introduce some additional notation. For a string x, let $Y_i(x)$ be the set of length-i supersequences of x that have the same first and last bit of x. For example, if $x = 101$, then $Y_4(x) = \{1011, 1101, 1001\}$.

For source string s and k-mer x, let $P_{s,x}[i] = \Pr(\tilde{S}[i : i + |x| - 1] = x)$, i.e., the probability that x appears at position i in a trace $\tilde{S}$ of s. Notice that it is straightforward to estimate $P_{s,x}[i]$ from the set of traces as $\hat{P}_{s,x}[i] = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}\{\tilde{S}_t[i : i + |x| - 1] = x\}$. Recall that the entry in the k-mer density map K_s corresponding to the substring x at index i of s is defined as

$$K_{s,x}[i] = \sum_{j=i}^{n} \binom{j-1}{i-1} (1-p)^{i-1} p^{j-i} \mathbb{1}\{s[j : j + \ell - 1] = x\}.$$

In order to estimate $K_{s,x}[i]$, we first write it in terms of $P_{s,x}[i]$ and $P_{s,y}[i]$ for all y of length greater than k. This will then allow us to estimate $K_{s,x}[i]$ using the estimates of $P_{s,x}[i]$ and $P_{s,y}[i]$, which can be obtained directly from the set of traces.

Lemma 1. *For source string s and k-mer x, we have*

$$K_{s,x}[i] = \frac{1}{(1-p)^k}\left(P_{s,x}[i] - \sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)} (-1)^{|y|-|x|+1} P_{s,y}[i] \binom{y}{x}' \left(\frac{p}{1-p}\right)^{\ell-k}\right). \tag{9}$$

Proof: We begin by deriving a recursive formula for $K_{s,x}[i]$. We first notice that

$$\begin{aligned} P_{s,x}[i] &= \sum_{\ell=k}^{n}\sum_{y\in Y_\ell(x)} \binom{y}{x}' p^{\ell-k}(1-p)^k \sum_{j=i}^{n}\binom{j-1}{i-1}(1-p)^{i-1}p^{j-i}\mathbb{1}\{s[j : j+\ell-1] = y\} \\ &= \sum_{\ell=k}^{n}\sum_{y\in Y_\ell(x)} \binom{y}{x}' p^{\ell-k}(1-p)^k K_{s,y}[i]. \end{aligned} \tag{10}$$

This follows because, in order for x to appear at position i in a trace, a superstring $y \in Y_\ell(x)$ must appear at position $j \geq i$ in s, $\binom{y}{x}'$ bits from y must be deleted, and $\binom{j-1}{i-1}$ bits in front of y must be deleted. Notice that $\binom{y}{x}' p^{\ell-k}(1-p)^k$ is the probability that a copy of y in s becomes x in $\tilde{S}$, and $K_{s,y}[i]$ is the probability that the beginning of a copy of y in s is shifted to position i in $\tilde{S}$.

Notice that, for $\ell = k$, the only term in the summation in (10) is $(1-p)^k K_{s,x}[i]$. This allows us to rewrite (10) as

$$K_{s,x}[i] = \frac{1}{(1-p)^k}\left(P_{s,x}[i] - \sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)}\binom{y}{x}' p^{\ell-k}(1-p)^k K_{s,y}[i]\right)$$
$$= \frac{1}{(1-p)^k}\left(P_{s,x}[i] - \sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)}(1-p)^\ell\binom{y}{x}' K_{s,y}[i]\left(\frac{p}{1-p}\right)^{\ell-k}\right). \tag{11}$$

By recursively applying (11) into itself, we write $K_{s,x}[i]$ in terms of $P_{s,x}[i]$ terms. This yields

$$K_{s,x}[i] = \frac{1}{(1-p)^k}\left(P_{s,x}[i] - \sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)} P_{s,y}[i]a_{s,x,y}\left(\frac{p}{1-p}\right)^{\ell-k}\right) \tag{12}$$

where $a_{s,x,y} \in \mathbb{Z}$ is a constant that depends on s, x, y. Observe that $a_{s,x,y}$ obeys the following recursion: for $y \in Y_\ell(x)$, we have that

$$a_{s,x,y} = \binom{y}{x}' - \sum_{k+1\le j<\ell}\ \sum_{z\in Y_j(x)} a_{s,x,z}\binom{y}{z}'. \tag{13}$$

This is because as we expand (11) one step at a time to eventually obtain (12), we observe that every time we obtain a new term involving $P_{s,y}[i]$ in the expansion with coefficient $c_y\left(\frac{p}{1-p}\right)^{|y|-k}$, in the next step of the expansion we obtain a term involving $P_{s,z}[i]$ with coefficient $-c_y\binom{z}{y}'\left(\frac{p}{1-p}\right)^{|z|-k}$ for every $z \in \cup_{\ell=|y|+1}^{n} Y_\ell(y)$. One step of the expansion is shown in the appendix to illuminate this argument. We proceed to prove that for any s, x, y, we have that

$$a_{s,x,y} = (-1)^{|y|-|x|+1}\binom{y}{x}'. \tag{14}$$

We will use the following lemma, which appears as Corollary (6.3.9) in [25].

Lemma 2. *For any two strings f, g over an alphabet Σ,*

$$\sum_h (-1)^{|g|+|h|}\binom{f}{h}\binom{h}{g} = \delta_{f,g} \tag{15}$$

where $\delta_{f,g} = 0$ if $f \neq g$, and $\delta_{f,g} = 1$ if $f = g$.

If $|y| - |x| = 1$, (13) implies that $a_{s,x,y} = \binom{y}{x}'$, and (14) clearly holds. Suppose (14) holds for $|y| - |x| < m$. Then if $|y| - |x| = m$, we have that

$$a_{s,x,y} = \binom{y}{x}' - \sum_{k<j<\ell}\ \sum_{z\in Y_j(x),} a_{s,x,z}\binom{y}{z}' \tag{16}$$
$$= \binom{y}{x}' - \sum_{k<j<\ell}\ \sum_{z\in Y_j(x),} (-1)^{|z|-|x|+1}\binom{z}{x}'\binom{y}{z}' \tag{17}$$
$$= \binom{y}{x}' + \sum_{k<j<\ell}\ \sum_{z\in Y_j(x),} (-1)^{|z|+|x|}\binom{z}{x}'\binom{y}{z}' \tag{18}$$
$$= \binom{y}{x}' + \left(\sum_{k\le j\le\ell}\ \sum_{z\in Y_j(x),} (-1)^{|z|+|x|}\binom{z}{x}'\binom{y}{z}'\right) - \binom{x}{x}'\binom{y}{x}' - (-1)^{|y|+|x|}\binom{y}{x}'\binom{y}{y}' \tag{19}$$

$$= \binom{y}{x}' - \binom{y}{x}' - (-1)^{|y|+|x|}\binom{y}{x}' \tag{20}$$

$$= (-1)^{|y|-|x|+1}\binom{y}{x}'. \tag{21}$$

where (20) follows from Lemma 2. By plugging in this formula for $a_{s,x,y}$ into (12), we obtain

$$K_{s,x}[i] = \frac{1}{(1-p)^k}\left(P_{s,x}[i] - \sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)}(-1)^{|y|-|x|+1}P_{s,y}[i]\binom{y}{x}'\left(\frac{p}{1-p}\right)^{\ell-k}\right).$$

■

Lemma 1 allows us to obtain an unbiased estimator for $K_{s,x}[i]$ given by

$$\hat{K}_{s,x}[i] = \frac{1}{(1-p)^k}\left(\hat{P}_{s,x}[i] - \sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)}(-1)^{|y|-|x|+1}\hat{P}_{s,y}[i]\binom{y}{x}'\left(\frac{p}{1-p}\right)^{\ell-k}\right) \tag{22}$$

where $\hat{P}_{s,x}[i] = \frac{1}{T}\sum_{t=1}^{T}\mathbb{1}\{\tilde{S}_t[i : i+|x|-1] = x\}$ and $\hat{P}_{s,y}[i]$ is defined analogously.

One way to analyze the performance of our estimator $\hat{K}_{s,x}[i]$ would be to apply a standard concentration inequality such as the Chernoff bound to each of the terms $\hat{P}_{s,y}[i]$ and use that to bound the error of $\hat{K}_{s,x}[i]$. However, this yields a suboptimal analysis as we do not need to guarantee the accuracy of each $\hat{P}_{s,y}[i]$ term. Directly analyzing the accuracy of $\hat{K}_{s,x}[i]$ is more subtle, as $\hat{K}_{s,x}[i]$ is not a sum of independent random variables. To that end, we apply McDiarmid's inequality to analyze the deviation of $\hat{K}_{s,x}[i]$ from $K_{s,x}[i]$ directly.

Lemma 3. *For source string s, a k-mer x with $|x| = c\log n$, and a set of T traces, we have*

$$\Pr\left(|\hat{K}_{s,x}[i] - K_{s,x}[i]| \geq \epsilon\right) \leq 2\exp\left(-\frac{2T\epsilon^2}{n\left(\frac{1}{n^{c\log(1-p)}}\left(1+2n^{\alpha_c(p)}\right)\right)^2}\right). \tag{23}$$

Proof: In order to apply McDiarmid's inequality, we will view the estimator $\hat{K}_{s,x}[i]$ as a function of Tn independent random variables corresponding to the indicator random variables that indicate whether a particular bit is deleted in a particular trace. To apply McDiarmid's inequality, we have to upper bound how much the estimator can change by changing the value of one of these indicator random variables. Changing the value of the indicator random variable for a particular bit in the tth trace $\tilde{S}_t$ changes the estimator by at most

$$b \leq \frac{1}{T}\frac{1}{(1-p)^k}\left(1 + \sum_{\ell=k+1}^{n} 2\max_{y\in Y_\ell(x)}\binom{y}{x}'\left(\frac{p}{1-p}\right)^{\ell-k}\right) \tag{24}$$

$$\leq \frac{1}{T}\frac{1}{(1-p)^k}\left(1 + \sum_{\ell=k+1}^{n} 2\binom{\ell-2}{k-2}\left(\frac{p}{1-p}\right)^{\ell-k}\right) \tag{25}$$

$$\leq \frac{1}{T}\frac{1}{(1-p)^k}\left(1 + 2n\max_{\ell\in[k+1,n]}\binom{\ell-2}{k-2}\left(\frac{p}{1-p}\right)^{\ell-k}\right). \tag{26}$$

To analyze b when $k = c\log(n)$ for c constant, and $p < 0.5$ we have that

$$b \leq \frac{1}{T}\frac{1}{(1-p)^{c\log(n)}}\left(1 + 2n\max_{\ell\in[c\log(n)+1,n]}\binom{\ell-2}{c\log(n)-2}\left(\frac{p}{1-p}\right)^{\ell-c\log(n)}\right) \tag{27}$$

$$\leq \frac{1}{T}\frac{1}{(1-p)^{c\log(n)}}\left(1 + 2n\max_{\ell\in[c\log(n)+1,n]}\binom{\ell}{c\log(n)}\left(\frac{p}{1-p}\right)^{\ell-c\log(n)}\right) \tag{28}$$

$$= \frac{1}{T}\frac{1}{n^{c\log(1-p)}}\left(1 + 2n^{1+c\log(\frac{1-p}{p})}\max_{\ell\in[c\log(n)+1,n]}\binom{\ell}{c\log(n)}\left(\frac{p}{1-p}\right)^{\ell}\right). \tag{29}$$

We have that

$$\max_{\ell\in[c\log(n)+1,n]}\binom{\ell}{c\log(n)}\left(\frac{p}{1-p}\right)^{\ell} \tag{30}$$

$$\leq \max_{\ell\in[c\log(n)+1,n]} 2^{\ell H(c\log(n)/\ell)}\left(\frac{p}{1-p}\right)^{\ell} \tag{31}$$

$$\leq 2^{\frac{\log(n^c)}{1-p/(1-p)}H(1-p/(1-p))}\left(\frac{p}{1-p}\right)^{\frac{c\log(n)}{1-p/(1-p)}} \tag{32}$$

$$= n^{\frac{cH(1-p/(1-p))}{1-p/(1-p)}+\frac{c\log(p/(1-p))}{1-p/(1-p)}} \tag{33}$$

because the maximizing ℓ is given by $\ell^* = \frac{\log(n^c)}{1-p/(1-p)}$. This is because $x = \frac{\log(n^c)}{1-q}$ is the only zero of

$$\frac{d}{dx}2^{xH(c\log(n)/x)}q^x \tag{34}$$

$$= q^x n^{c\log(1-\log(n^c)/x)-c\log(\log(n^c)/x)}\left(1-\frac{\log(n^c)}{x}\right)^{-x}(\log(q)-\log(1-\log(n^c)/x)) \tag{35}$$

where $q = p/(1-p)$, the function is a differentiable for $x \in (c\log(n)+1, n)$, and the second derivative of the function at $x = \frac{\log(n^c)}{1-q}$ is given by

$$-\frac{(q-1)^2 n^{c\log(q)-c\log(1-q)}}{q\log(n^c)} \tag{36}$$

which is negative. We therefore have that we have that

$$b \leq \frac{1}{T}\frac{1}{n^{c\log(1-p)}}\left(1 + 2n^{1+c\log((1-p)/p)}n^{\frac{cH(1-p/(1-p))}{1-p/(1-p)}+\frac{c\log(p/(1-p))}{1-p/(1-p)}}\right) \tag{37}$$

$$= \frac{1}{T}\frac{1}{n^{c\log(1-p)}}\left(1 + 2n^{1+c\log((1-p)/p)+\frac{cH(1-p/(1-p))}{1-p/(1-p)}+\frac{c\log(p/(1-p))}{1-p/(1-p)}}\right) \tag{38}$$

when $k = c\log(n)$ for c constant and $p < 0.5$. Let

$$\alpha_c(p) = 1 + c\log((1-p)/p) + \frac{cH(1-p/(1-p))}{1-p/(1-p)} + \frac{c\log(p/(1-p))}{1-p/(1-p)}. \tag{39}$$

Plugging this into McDiarmid's inequality, we have

$$\Pr\left(|\hat{K}_{s,x}[i] - K_{s,x}[i]| \geq \epsilon\right) \leq 2\exp\left(-\frac{2\epsilon^2}{nT\left(\frac{1}{T}\frac{1}{n^{c\log(1-p)}}\left(1+2n^{\alpha_c(p)}\right)\right)^2}\right) \tag{40}$$

$$= 2\exp\left(-\frac{2T\epsilon^2}{n\left(\frac{1}{n^{c\log(1-p)}}\left(1+2n^{\alpha_c(p)}\right)\right)^2}\right). \tag{41}$$

■

Setting $\delta = \Pr\left(|\hat{K}_{s,x}[i] - K_{s,x}[i]| \geq \epsilon\right)$, we conclude that

$$T = \log(2/\delta)\frac{n}{2\epsilon^2}\left(\frac{1}{n^{c\log(1-p)}}\left(1+2n^{\alpha_c(p)}\right)\right)^2 \tag{42}$$

traces suffices for recovering $K_{s,x}[i]$ with error less than ϵ with probability at least $1-\delta$.

4 A New Bound for the k-subword deck Reconstruction Problem

In this section, we derive an estimator for the k-subword deck using the same techniques used to derive the estimator for the k-mer density map, and prove Theorem 2. Let $N_{s,x}$ denote the number of times the k-mer x appears in the source string s. When s is clear from context, we write $N_{s,x}$ as N_x. In this section, let $E_{s,x}$ denote the expected number of times that k-mer x appears in a trace $\tilde{S}$ of source string s. When s is clear from context, we write $E_{s,x}$ as E_x. Notice that we can easily estimate E_x from a set of T traces as $\hat{E}_x = \frac{1}{T}\sum_{i=1}^{T}(\#\text{ of times } x \text{ appears as a substring in trace } \tilde{S}_i)$.

Similar to the k-mer density map estimator, we derive a formula for N_x in terms of E_x and E_y for all supersequences y of x. This allows us to estimate N_x directly from the set of traces. Our formula for N_x is given in the following lemma. Refer to the beginning of the previous section for the definition of $Y_i(x)$.

Lemma 4. *For source string $s \in \{0,1\}^n$, deletion probability $p \in [0,1)$, and $x \in \{0,1\}^k$, we have*

$$N_x = \frac{1}{(1-p)^k}\left(E_x - \sum_{i=k+1}^{n}\sum_{y\in Y_i(x)}(-1)^{|y|-|x|+1}E_y\binom{y}{x}'\left(\frac{p}{1-p}\right)^{i-k}\right). \tag{43}$$

Proof: Observe that

$$\begin{aligned} E_x &= \sum_{i=k}^{n}\sum_{y\in Y_i(x)}N_y\binom{y}{x}'p^{i-k}(1-p)^k \\ &= \sum_{i=k}^{n}\sum_{y\in Y_i(x)}(1-p)^i N_y\binom{y}{x}'\left(\frac{p}{1-p}\right)^{i-k} \end{aligned} \tag{44}$$

by linearity of expectation applied to all ways x can be formed from a substring y in s. This can be rewritten as the following recursive formula for N_x

$$N_x = \frac{1}{(1-p)^k}\left(E_x - \sum_{i=k+1}^{n}\sum_{y\in Y_i(x)}(1-p)^i N_y\binom{y}{x}'\left(\frac{p}{1-p}\right)^{i-k}\right). \tag{45}$$

This formula is identical to the recursive formula for the k-mer density map in (11) with $K_{s,x}[i]$ replaced by N_x and $E_{s,y}[i]$ replaced by E_y. Therefore, the same steps can be taken to obtain a non-recursive formula for N_x, which is given by

$$N_x = \frac{1}{(1-p)^k}\left(E_x - \sum_{i=k+1}^{n}\sum_{y\in Y_i(x)}(-1)^{|y|-|x|+1}E_y\binom{y}{x}'\left(\frac{p}{1-p}\right)^{i-k}\right).$$

■

Our estimator is given by

$$\hat{N}_x = \frac{1}{(1-p)^k}\left(\hat{E}_x - \sum_{i=k+1}^{n}\sum_{y\in Y_i(x)}(-1)^{|y|-|x|+1}\hat{E}_y\binom{y}{x}'\left(\frac{p}{1-p}\right)^{i-k}\right). \tag{46}$$

which we then round to the nearest integer. Observe that if the error in the estimate of N_x is less than 0.5, then we will estimate N_x correctly after rounding.

Chen, De, Lee, Servedio, and Sinha [10] derive a formula for N_x that is equivalent to ours, but written in a different form. Their formula is given by

$$N_x = \frac{1}{(1-p)^k}\sum_{\substack{\alpha\in\mathbb{Z}_{\geq 0}^{k-1}\\ |\alpha|\leq n-k}} E\left[\#(x[1]*^{\alpha[1]}x[2]*^{\alpha[2]}x[3]\cdots x[k-1]*^{\alpha[k-1]}x[k],\ \tilde{S})\right]\left(\frac{-p}{1-p}\right)^{|\alpha|} \tag{47}$$

where $*$ is the wildcard symbol and $\#(x[1] *^{\alpha[1]} x[2] *^{\alpha[2]} x[3] \cdots x[k-1] *^{\alpha[k-1]} x[k],\ \tilde{S})$ is the number of times a substring of the form $x[1] *^{\alpha[1]} x[2] *^{\alpha[2]} x[3] \cdots x[k-1] *^{\alpha[k-1]} x[k]$ appears in a trace $\tilde{S}$ of s, and $|\alpha| = \sum_{i=1}^{k-1} |\alpha[i]|$ for $\alpha \in \mathbb{Z}_{\geq 0}^{k-1}$. In order to prove this formula, the authors define a complex polynomial that is a generalization of N_x, perform a Taylor expansion on the polynomial, and relate the partial derivatives of the polynomial to expected values of functions of traces of s. The formula above is then a special case of this more general result. Our significantly shorter proof is very different from theirs in that we first derive a recursive estimator of N_x and then use structural knowledge of $\binom{y}{x}'$ to prove its equivalence to the polynomial above.

The initial algorithm presented in [10] is to use (47) directly for estimation of N_x. For $p < 0.5$, the authors bound the error of the estimate of each $E\left[\#(x[1] *^{\alpha[1]} x[2] *^{\alpha[2]} x[3] \cdots x[k-1] *^{\alpha[k-1]} x[k],\ \tilde{S})\right]$ term using a Chernoff bound and the fact that $\#(x[1] *^{\alpha[1]} x[2] *^{\alpha[2]} x[3] \cdots x[k-1] *^{\alpha[k-1]} x[k],\ \tilde{S}) \in [0, n]$. This analysis of the algorithm yields an upper bound of $n^{O(k)} \cdot \log(1/\delta)$ traces. For $k = c\log(n)$, this is superpolynomial in n. In the appendix, we employ McDiarmid's inequality to prove that $\text{poly}(n)$ traces suffice for reconstructing N_x from (47) with high probability.

In the improved algorithm [10] for $p < 0.5$ that uses $\text{poly}(n)$ traces, the authors estimate N_x using only estimates of $E\left[\#(x[1] *^{\alpha[1]} x[2] *^{\alpha[2]} x[3] \cdots x[k-1] *^{\alpha[k-1]} x[k],\ \tilde{S})\right]$ for $|\alpha| \leq d + k$ where $d = \frac{e^2}{1/2-p}\left(k \log\left(\frac{e^2}{1/2-p}\right) + \log(n)\right)$. The truncated formula they use is

$$N_x = \frac{1}{(1-p)^k} \sum_{\substack{\alpha \in \mathbb{Z}_{\geq 0}^{k-1} \\ |\alpha| \leq d}} E\left[\#(x[1] *^{\alpha[1]} x[2] *^{\alpha[2]} x[3] \cdots x[k-1] *^{\alpha[k-1]} x[k],\ \tilde{S})\right] \left(\frac{-p}{1-p}\right)^{|\alpha|}. \quad (48)$$

The authors then prove that the sum of the terms in (47) for strings y of length $> d + k$ is small, and again bound the error of the estimate of each $E\left[\#(x[1] *^{\alpha[1]} x[2] *^{\alpha[2]} x[3] \cdots x[k-1] *^{\alpha[k-1]} x[k],\ \tilde{S})\right]$ for $|\alpha| \leq d$ using a Chernoff bound and the fact that $\#(x[1] *^{\alpha[1]} x[2] *^{\alpha[2]} x[3] \cdots x[k-1] *^{\alpha[k-1]} x[k],\ \tilde{S}) \in [0, n]$. In specific, they prove Lemma 5 to upper bound the magnitude of the sum of the truncated terms in (47), and prove that the truncated polynomial (48) can be estimated with error at most 0.2 with high probability. This leads to an estimator of (47) with error at most 0.3 with high probability, which is sufficient for estimating N_x correctly after rounding.

Lemma 5. *For $p < 1/2$, and $d = \frac{e^2}{1/2-p}\left(k \log\left(\frac{e^2}{1/2-p}\right) + \log(n)\right)$,*

$$\left| \frac{1}{(1-p)^k} \sum_{\substack{\alpha \in \mathbb{Z}_{\geq 0}^{k-1} \\ |\alpha| > d}} E\left[\#(x[1] *^{\alpha[1]} x[2] *^{\alpha[2]} x[3] \cdots x[k-1] *^{\alpha[k-1]} x[k],\ \tilde{S})\right] \left(\frac{-p}{1-p}\right)^{|\alpha|} \right| \leq 0.1. \quad (49)$$

The number of traces used by the truncated algorithm to reconstruct N_x with probability at least $1-\delta$ for $p < 1/2$ is proved to be $O((M^2 2^k)^2) \log\left(\frac{M}{\delta}\right)$, and it is proved that $M \leq n\left(\frac{e^2}{1/2-p}\right)^{3k}$. It follows that number of traces used in the analysis of [10] is

$$\begin{aligned} &O\left(\left(\left(n\left(\frac{e^2}{1/2-p}\right)^{3k}\right)^2 2^k\right)^2\right) \log\left(\frac{n\left(\frac{e^2}{1/2-p}\right)^{3k}}{\delta}\right) \\ &= O\left(n^4 \left(\frac{e^2}{1/2-p}\right)^{12k} 4^k\right) \log\left(\frac{n\left(\frac{e^2}{1/2-p}\right)^{3k}}{\delta}\right). \end{aligned} \quad (50)$$

For $k = c\log(n)$, the number of traces used is

$$O\left(n^4\left(\frac{e^2}{1/2-p}\right)^{12c\log(n)}4^{c\log(n)}\right)\log\left(\frac{n\left(\frac{e^2}{1/2-p}\right)^{3c\log(n)}}{\delta}\right)$$
$$= O\left(n^4 n^{12c\left(\frac{e^2}{1/2-p}\right)}n^{c\log(4)}\right)\log\left(\frac{n\cdot n^{3c\log\left(\frac{e^2}{1/2-p}\right)}}{\delta}\right)$$
$$= O\left(n^{4+12c\left(\frac{e^2}{1/2-p}\right)+c\log(4)}\right)\log\left(\frac{n^{1+3c\log\left(\frac{e^2}{1/2-p}\right)}}{\delta}\right). \tag{51}$$

We show that by using McDiarmid's to analyze sample complexity, we can significantly improve the upper bound on the number of samples needed by the truncated estimator for reconstructing the k-subword deck. This result immediately yields Theorem 2.

Lemma 6. *For deletion probability $p < 0.5$, and any source string $s \in \{0,1\}^n$, we can reconstruct N_x for any k-mer x where $k = c\log n$ with probability $1-\delta$ using*

$$O\left(\log^4(n)\cdot n^{1+c\left(\frac{(1-p)H(1-p/(1-p))+p\log(p/(1-p))}{1/2-p}+2\log\left(\frac{1}{1-p}\right)\right)}\right)\log(1/\delta) \tag{52}$$

traces.

Proof: Recall that the truncated estimator from [10] paper written in our form is given by

$$\hat{N}_x = \frac{1}{(1-p)^k}\left(\hat{E}_x - \sum_{i=k+1}^{d}\sum_{y\in Y_i(x)}(-1)^{|y|-|x|+1}\hat{E}_y\binom{y}{x}'\left(\frac{p}{1-p}\right)^{i-k}\right)$$

where

$$d = \frac{e^2}{1/2-p}\left(k\ln\frac{e^2}{1/2-p}+\ln n\right).$$

The set of independent random variables we use in McDiarmid's inequality is the set of Tn indicator random variables that indicate whether a particular bit is deleted in a particular trace. To apply McDiarmid's inequality, we have to upper bound how much the estimator can change by changing the value of one of the indicator random variables. Changing the value of the indicator random variable for a particular bit in the tth trace $\tilde{S}_t$ changes the estimator by at most

$$b \le \frac{1}{T}\frac{1}{(1-p)^k}\left(\min(k, n-k+1) + \sum_{i=k+1}^{d}2\min(i, n-i+1)\max_{y\in Y_i(x)}\binom{y}{x}'\left(\frac{p}{1-p}\right)^{i-k}\right)$$
$$\le \frac{1}{T}\frac{1}{(1-p)^k}\left(k + \sum_{i=k+1}^{d}2i\binom{i-2}{k-2}\left(\frac{p}{1-p}\right)^{i-k}\right)$$
$$\le \frac{1}{T}\frac{1}{(1-p)^k}\left(k + 2d\max_{i\in[k+1,d]} i\binom{i-2}{k-2}\left(\frac{p}{1-p}\right)^{i-k}\right)$$
$$\le \frac{1}{T}\frac{1}{(1-p)^k}\left(k + 2d^2\max_{i\in[k+1,d]}\binom{i-2}{k-2}\left(\frac{p}{1-p}\right)^{i-k}\right) \tag{53}$$

To analyze b when $k = c\log(n)$ for c constant, and $p < 0.5$ we have that

$$\begin{aligned} b &\leq \frac{1}{T}\frac{1}{(1-p)^{c\log(n)}}\left(c\log(n) + 2d^2 \max_{i\in[c\log(n)+1,d]}\binom{i-2}{c\log(n)-2}\left(\frac{p}{1-p}\right)^{i-c\log(n)}\right) \\ &\leq \frac{1}{T}\frac{1}{(1-p)^{c\log(n)}}\left(c\log(n) + 2d^2 \max_{i\in[c\log(n)+1,d]}\binom{i}{c\log(n)}\left(\frac{p}{1-p}\right)^{i-c\log(n)}\right) \\ &= \frac{1}{T}\frac{1}{n^{c\log(1-p)}}\left(c\log(n) + 2d^2 n^{c\log(\frac{1-p}{p})} \max_{i\in[c\log(n)+1,d]}\binom{i}{c\log(n)}\left(\frac{p}{1-p}\right)^{i}\right). \end{aligned} \tag{54}$$

We have that

$$\max_{i\in[c\log(n)+1,n]}\binom{i}{c\log(n)}\left(\frac{p}{1-p}\right)^{i} \tag{55}$$

$$\leq \max_{i\in[c\log(n)+1,n]} 2^{iH(c\log(n)/i)}\left(\frac{p}{1-p}\right)^{i} \tag{56}$$

$$\leq 2^{\frac{\log(n^c)}{1-p/(1-p)}H(1-p/(1-p))}\left(\frac{p}{1-p}\right)^{\frac{c\log(n)}{1-p/(1-p)}} \tag{57}$$

$$= n^{\frac{cH(1-p/(1-p))}{1-p/(1-p)}+\frac{c\log(p/(1-p))}{1-p/(1-p)}} \tag{58}$$

where the maximizer in (56) is given by $i = \frac{\log(n^c)}{1-p/(1-p)}$ as proved in Lemma 3. Recall that this is the maximizer because $x = \frac{\log(n^c)}{1-q}$ is the only zero of

$$\begin{aligned} &\frac{d}{dx}2^{xH(c\log(n)/x)}q^x \\ &= q^x n^{c\log(1-\log(n^c)/x)-c\log(\log(n^c)/x)}\left(1-\frac{\log(n^c)}{x}\right)^{-x}(\log(q)-\log(1-\log(n^c)/x)) \end{aligned} \tag{59}$$

where $q = p/(1-p)$, the function is a differentiable for $x \in (c\log(n)+1, n)$, and the second derivative of the function at $x = \frac{\log(n^c)}{1-q}$ is given by

$$-\frac{(q-1)^2 n^{c\log(q)-c\log(1-q)}}{q\log(n^c)}$$

which is negative. We therefore have that

$$\begin{aligned} b &\leq \frac{1}{T}\frac{1}{n^{c\log(1-p)}}\left(c\log(n) + 2d^2 n^{c\log((1-p)/p)} n^{\frac{cH(1-p/(1-p))}{1-p/(1-p)}+\frac{c\log(p/(1-p))}{1-p/(1-p)}}\right) \\ &= \frac{1}{T}\frac{1}{n^{c\log(1-p)}}\left(c\log(n) + 2d^2 n^{c\log((1-p)/p)+\frac{cH(1-p/(1-p))}{1-p/(1-p)}+\frac{c\log(p/(1-p))}{1-p/(1-p)}}\right) \end{aligned} \tag{60}$$

when $k = c\log(n)$ for c constant and $p < 0.5$. Let

$$\gamma_c(p) = c\log((1-p)/p) + \frac{cH(1-p/(1-p))}{1-p/(1-p)} + \frac{c\log(p/(1-p))}{1-p/(1-p)} \tag{61}$$

Due to Lemma 5, it suffices to estimate N_x with error at most 1/5. Using McDiarmid's inequality, and setting $\epsilon = 1/5$, we have

$$\Pr\left(\hat{N}_x - N_x \geq \frac{1}{5}\right) \leq \exp\left(-\frac{2\left(\frac{1}{5}\right)^2}{nT\left(\frac{1}{T}\frac{1}{n^{c\log(1-p)}}\left(c\log(n)+2d^2n^{\gamma_c(p)}\right)\right)^2}\right)$$

$$= \exp\left(-\frac{T}{\frac{25}{2}n\left(\frac{1}{n^{c\log(1-p)}}\left(c\log(n)+2d^2n^{\gamma_c(p)}\right)\right)^2}\right) \tag{62}$$

Setting $\delta = \Pr\left(\hat{N}_x - N_x \geq \frac{1}{5}\right)$, we have

$$T = \log(1/\delta)\frac{25}{2}n\left(\frac{1}{n^{c\log(1-p)}}\left(c\log(n)+2d^2n^{\gamma_c(p)}\right)\right)^2 \tag{63}$$

traces suffices for recovering N_x with probability at least $1-\delta$.

The number of traces needed to recover N_x with probability at least $1-\delta$ for $p < 1/2$ using the truncated estimator is

$$\begin{aligned}&\log(1/\delta)\frac{25}{2}n\left(\frac{1}{n^{c\log(1-p)}}\left(c\log(n)+2d^2n^{\gamma_c(p)}\right)\right)^2\\&= O\left(d^4n^{1+2\left(\gamma_c(p)+c\log\left(\frac{1}{1-p}\right)\right)}\right)\log(1/\delta)\end{aligned} \tag{64}$$

traces where

$$\begin{aligned}\gamma_c(p) &= c\log((1-p)/p) + \frac{cH(1-p/(1-p))}{1-p/(1-p)} + \frac{c\log(p/(1-p))}{1-p/(1-p)}\\&= \frac{c(-1+p/(1-p))\log(p/(1-p))}{1-p/(1-p)} + \frac{cH(1-p/(1-p))}{1-p/(1-p)} + \frac{c\log(p/(1-p))}{1-p/(1-p)}\\&= \frac{c(1-p)H(1-p/(1-p))}{1-2p} + \frac{cp\log(p/(1-p))}{1-2p}\\&= \frac{c(1-p)H(1-p/(1-p))+cp\log(p/(1-p))}{1-2p}.\end{aligned} \tag{65}$$

Thus, the number of traces needed by our algorithm is upper bounded by

$$\begin{aligned}&O\left(d^4n^{1+2\left(\frac{c(1-p)H(1-p/(1-p))+cp\log(p/(1-p))}{1-2p}+c\log\left(\frac{1}{1-p}\right)\right)}\right)\log(1/\delta)\\&= O\left(d^4n^{1+2c\left(\frac{(1-p)H(1-p/(1-p))+p\log(p/(1-p))}{1-2p}+\log\left(\frac{1}{1-p}\right)\right)}\right)\log(1/\delta)\\&= O\left(d^4n^{1+c\left(\frac{(1-p)H(1-p/(1-p))+p\log(p/(1-p))}{1/2-p}+2\log\left(\frac{1}{1-p}\right)\right)}\right)\log(1/\delta)\end{aligned} \tag{66}$$

Plugging in

$$d = \frac{e^2}{1/2-p}\left(k\ln\frac{e^2}{1/2-p}+\ln n\right), \tag{67}$$

the number of traces is upper bounded by

$$O\left(\log^4(n)\cdot n^{1+c\left(\frac{(1-p)H(1-p/(1-p))+p\log(p/(1-p))}{1/2-p}+2\log\left(\frac{1}{1-p}\right)\right)}\right)\log(1/\delta). \tag{68}$$

■

References

[1] T. Batu, S. Kannan, S. Khanna, and A. Mcgregor, "Reconstructing strings from random traces," *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, vol. 15, 01 2004.

[2] V. Bhardwaj, P. A. Pevzner, C. Rashtchian, and Y. Safonova, "Trace reconstruction problems in computational biology," *IEEE Transactions on Information Theory*, vol. 67, no. 6, pp. 3295–3314, 2020.

[3] A. De, R. O'Donnell, and R. A. Servedio, "Optimal mean-based algorithms for trace reconstruction," in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1047–1056, 2017.

[4] Z. Chase, "New lower bounds for trace reconstruction," in *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, vol. 57, pp. 627–643, Institut Henri Poincaré, 2021.

[5] Z. Chase, "Separating words and trace reconstruction," in *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 21–31, 2021.

[6] N. Holden, R. Pemantle, and Y. Peres, "Subpolynomial trace reconstruction for random strings and arbitrary deletion probability," in *Conference On Learning Theory*, pp. 1799–1840, PMLR, 2018.

[7] S. Davies, M. Z. Rácz, B. G. Schiffer, and C. Rashtchian, "Approximate trace reconstruction: Algorithms," in *2021 IEEE International Symposium on Information Theory (ISIT)*, pp. 2525–2530, IEEE, 2021.

[8] S. R. Srinivasavaradhan, M. Du, S. Diggavi, and C. Fragouli, "On maximum likelihood reconstruction over multiple deletion channels," in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 436–440, IEEE, 2018.

[9] T. Holenstein, M. Mitzenmacher, R. Panigrahy, and U. Wieder, "Trace reconstruction with constant deletion probability and related results.," in *SODA*, vol. 8, pp. 389–398, 2008.

[10] X. Chen, A. De, C. H. Lee, R. A. Servedio, and S. Sinha, "Polynomial-time trace reconstruction in the smoothed complexity model," in *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 54–73, SIAM, 2021.

[11] Y. Liu, J. Schröder, and B. Schmidt, "Musket: a multistage k-mer spectrum-based error corrector for illumina sequence data," *Bioinformatics*, vol. 29, no. 3, pp. 308–315, 2013.

[12] I. Shomorony, T. A. Courtade, and D. Tse, "Fundamental limits of genome assembly under an adversarial erasure model," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 2, no. 2, pp. 199–208, 2016.

[13] P. A. Pevzner, H. Tang, and M. S. Waterman, "An eulerian path approach to dna fragment assembly," *Proceedings of the National Academy of Sciences*, vol. 98, no. 17, pp. 9748–9753, 2001.

[14] S. Marcovich and E. Yaakobi, "Reconstruction of strings from their substrings spectrum," *IEEE Transactions on Information Theory*, vol. 67, no. 7, pp. 4369–4384, 2021.

[15] B. Chor, D. Horn, N. Goldman, Y. Levy, and T. Massingham, "Genomic dna k-mer spectra: models and modalities," *Genome biology*, vol. 10, no. 10, pp. 1–10, 2009.

[16] P. A. Noble *et al.*, "Tetranucleotide frequencies in microbial genomes," *Electrophoresis*, vol. 19, Apr 1998.

[17] E. Ukkonen, "Approximate string-matching with q-grams and maximal matches," *Theoretical computer science*, vol. 92, no. 1, pp. 191–211, 1992.

[18] G. Bresler, M. Bresler, and D. Tse, "Optimal assembly for high throughput shotgun sequencing," *BMC Bioinformatics*, 2013.

[19] I. Shomorony, S. H. Kim, T. A. Courtade, and D. N. Tse, "Information-optimal genome assembly via sparse read-overlap graphs," *Bioinformatics*, vol. 32, no. 17, pp. i494–i502, 2016.

[20] A. Mcgregor, E. Price, and S. Vorotnikova, "Trace reconstruction revisited," in *ESA*, 2014.

[21] E. Grigorescu, M. Sudan, and M. Zhu, "Limitations of mean-based algorithms for trace reconstruction at small edit distance," *IEEE Transactions on Information Theory*, 2022.

[22] J. Sima and J. Bruck, "Trace reconstruction with bounded edit distance," in *2021 IEEE International Symposium on Information Theory (ISIT)*, pp. 2519–2524, IEEE, 2021.

[23] F. Nazarov and Y. Peres, "Trace reconstruction with exp(o(n1/3)) samples," in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, (New York,

NY, USA), p. 1042–1046, Association for Computing Machinery, 2017.
[24] S. Narayanan and M. Ren, "Circular trace reconstruction," in *ITCS*, 2021.
[25] *Combinatorics on Words*. Cambridge Mathematical Library, Cambridge University Press, 2 ed., 1997.

5 Appendix

A. Proof of Corollary 2

Without loss of generality, assume that x occurs at position $\kappa(n) = \Theta(n)$ in s (if $\kappa(n) \neq \Theta(n)$ we can apply the following argument where s, s' and x are reversed). Observe that for $i \in \{1, \ldots, n-k+1\}$,

$$\begin{aligned} K_{s,x}[i] &= \sum_{j=1}^{n-k+1} \binom{j-1}{i-1} (1-p)^{i-1} p^{j-i} \mathcal{I}_{s,x}[j] \\ &= \sum_{j=1}^{n-k+1} \Pr(\mathrm{Bin}(j-1, 1-p) = i-1) \mathcal{I}_{s,x}[j] \end{aligned} \tag{69}$$

where $\mathrm{Bin}(n, q)$ is a binomial random variable with n trials and success probability q. In specific, at $i = \lceil (1-p)\kappa(n) \rceil$, we have that

$$\begin{aligned} K_{s,x}[\lceil (1-p)\kappa(n) \rceil] &\geq \Pr(\mathrm{Bin}(\kappa(n)-1, 1-p) = \lceil (1-p)\kappa(n) \rceil - 1) \\ &= \Pr(\mathcal{N}((1-p)(\kappa(n)-1),\ p(1-p)(\kappa(n)-1)) = \lceil (1-p)\kappa(n) \rceil - 1) + o\left(\frac{1}{\sqrt{n}}\right) \\ &= \frac{1}{\sqrt{2\pi p(1-p)(\kappa(n)-1)}} e^{-\frac{((1-p)(\kappa(n)-1) - \lceil (1-p)\kappa(n) \rceil + 1)^2}{p(1-p)(\kappa(n)-1)}} + o\left(\frac{1}{\sqrt{n}}\right) \\ &\geq \frac{1}{\sqrt{2\pi p(1-p)(\kappa(n)-1)}} e^{-\frac{(p+1)^2}{p(1-p)(\kappa(n)-1)}} + o\left(\frac{1}{\sqrt{n}}\right) \\ &= \frac{1}{\sqrt{2\pi p(1-p)(\kappa(n)-1)}} e^{-\frac{(p+1)^2}{p(1-p)(\kappa(n)-1)}} + o\left(\frac{1}{\sqrt{n}}\right) \\ &= \Omega\left(\frac{1}{\sqrt{n}}\right) \end{aligned} \tag{70}$$

by the local limit theorem. On the other hand, we have that for some function g such that $g(n) = \Omega(n^a)$ where $a > 0.5$,

$$\begin{aligned} K_{s',x}[\lceil (1-p)\kappa(n) \rceil] &\leq n \max\Big(\Pr\left(\mathrm{Bin}(\kappa(n) - g(n) - 1, 1-p) = \lceil (1-p)\kappa(n) \rceil - 1\right), \\ &\qquad \Pr(\mathrm{Bin}(\kappa(n) + g(n) - 1, 1-p) = \lceil (1-p)\kappa(n) \rceil - 1) \Big) \\ &\leq n \Pr(|\mathrm{Bin}(\kappa(n) - g(n) - 1, 1-p) - (1-p)(\kappa(n) - g(n) - 1)| > (1-p)g(n) + p) \\ &\leq n 2 \exp\left(-\frac{((1-p)g(n) + p)^2}{n} \right) \\ &= O\left(n \exp\left(-n^{2a-1} \right) \right) \end{aligned} \tag{71}$$

by Hoeffding's inequality. Thus, there is a $1/\mathrm{poly}(n)$ gap between $K_{s,x}[\lceil (1-p)\kappa(n) \rceil]$ and $K_{s',x}[\lceil (1-p)\kappa(n) \rceil]$ so s can be distinguished from s' using $\mathrm{poly}(n)$ traces by Corollary 1.

B. Proof of (8)

For $p < 0.5$,

$$
\begin{aligned}
&1 + c\left(\frac{(1-p)H(1-p/(1-p)) + p\log(p/(1-p))}{1/2-p} + 2\log\left(\frac{1}{1-p}\right)\right) \\
&\leq 1 + c\left(\frac{1-p}{1/2-p}\right) + 2c\log(2) \\
&= 1 + c\left(\frac{1-p}{1/2-p}\right) + c\log(4) \\
&< 4 + 12c\left(\frac{e^2}{1/2-p}\right) + c\log(4). \qquad (72)
\end{aligned}
$$

C. Proof that initial k-subword deck algorithm from [10] requires poly(n) traces

McDiarmid's inequality states the following. Let $X_1, ..., X_m$ are iid random variables. Suppose the function f has the property that for any sets of values $x_1, x_2, ..., x_m$ and $x_1', x_2', ..., x_m'$ that only differ at the ith random variable, it follows that

$$|f(x_1, x_2, ..., x_m) - f(x_1', x_2', ..., x_m')| \leq b_i. \qquad (73)$$

Then, for any $t > 0$, we have that

$$\Pr(|f(X_1, X_2, ..., X_m) - \mathbb{E}[f(X_1, X_2, ..., X_m)]| \geq t) \leq 2\exp\left(-\frac{2t^2}{\sum_{i=1}^m b_i^2}\right). \qquad (74)$$

Applying this to our problem, we set $f = \hat{N}_x$ and we take each random variable X_i to be an indicator of whether a particular bit in s is deleted in a particular trace.

Recall that our estimator is given by

$$\hat{N}_x = \frac{1}{(1-p)^k}\left(\hat{E}_x - \sum_{i=k+1}^{n}\sum_{y\in Y_i(x)}(-1)^{|y|-|x|+1}\hat{E}_y\binom{y}{x}'\left(\frac{p}{1-p}\right)^{i-k}\right). \qquad (75)$$

Changing the indicator random variable corresponding to a single bit in the jth trace $\tilde{S}_j$ changes the estimator by at most

$$
\begin{aligned}
b &\leq \frac{1}{T}\frac{1}{(1-p)^k}\left(\min(k, n-k+1) + \sum_{i=k+1}^{n} 2\min(i, n-i+1)\max_{y\in Y_i(x)}\binom{y}{x}'\left(\frac{p}{1-p}\right)^{i-k}\right) \\
&\leq \frac{1}{T}\frac{1}{(1-p)^k}\left(k + \sum_{i=k+1}^{n} 2i\binom{i-2}{k-2}\left(\frac{p}{1-p}\right)^{i-k}\right) \\
&\leq \frac{1}{T}\frac{1}{(1-p)^k}\left(k + 2n\max_{i\in[k+1,n]} i\binom{i-2}{k-2}\left(\frac{p}{1-p}\right)^{i-k}\right) \\
&\leq \frac{1}{T}\frac{1}{(1-p)^k}\left(k + 2n^2\max_{i\in[k+1,n]}\binom{i-2}{k-2}\left(\frac{p}{1-p}\right)^{i-k}\right). \qquad (76)
\end{aligned}
$$

To analyze b when $k = c\log(n)$ for c constant, and $p < 0.5$ we have that

$$
\begin{aligned}
b &\leq \frac{1}{T}\frac{1}{(1-p)^{c\log(n)}}\left(c\log(n) + 2n^2\max_{i\in[c\log(n)+1,n]}\binom{i-2}{c\log(n)-2}\left(\frac{p}{1-p}\right)^{i-c\log(n)}\right) \\
&\leq \frac{1}{T}\frac{1}{(1-p)^{c\log(n)}}\left(c\log(n) + 2n^2\max_{i\in[c\log(n)+1,n]}\binom{i}{c\log(n)}\left(\frac{p}{1-p}\right)^{i-c\log(n)}\right)
\end{aligned}
$$

$$= \frac{1}{T} \frac{1}{n^{c\log(1-p)}} \left(c\log(n) + 2n^{2+c\log(\frac{1-p}{p})} \max_{i\in[c\log(n)+1,n]} \binom{i}{c\log(n)} \left(\frac{p}{1-p}\right)^i \right). \tag{77}$$

We have that

$$\begin{aligned}
&\max_{i\in[c\log(n)+1,n]} \binom{i}{c\log(n)} \left(\frac{p}{1-p}\right)^i \\
&\le \max_{i\in[c\log(n)+1,n]} 2^{iH(c\log(n)/i)} \left(\frac{p}{1-p}\right)^i \\
&\le 2^{\frac{\log(n^c)}{1-p/(1-p)} H(1-p/(1-p))} \left(\frac{p}{1-p}\right)^{\frac{\log(n)}{1-p/(1-p)}} \\
&= n^{\frac{cH(1-p/(1-p))}{1-p/(1-p)} + \frac{c\log(p/(1-p))}{1-p/(1-p)}}
\end{aligned} \tag{78}$$

The maximizer above is given by $i^* = \frac{\log(n^c)}{1-p/(1-p)}$ as proved in Lemma 3. This is because $x = \frac{\log(n^c)}{1-q}$ is the only zero of

$$\begin{aligned}
&\frac{d}{dx} 2^{xH(c\log(n)/x)} q^x \\
&= q^x n^{c\log(1-\log(n^c)/x) - c\log(\log(n^c)/x)} \left(1 - \frac{\log(n^c)}{x}\right)^{-x} (\log(q) - \log(1-\log(n^c)/x))
\end{aligned} \tag{79}$$

where $q = p/(1-p)$, the function is a differentiable for $x \in (c\log(n)+1, n)$, and the second derivative of the function at $x = \frac{\log(n^c)}{1-q}$ is given by

$$-\frac{(q-1)^2 n^{c\log(q) - c\log(1-q)}}{q\log(n^c)} \tag{80}$$

which is negative. We therefore have that

$$\begin{aligned}
b &\le \frac{1}{T}\frac{1}{n^{c\log(1-p)}} \left(c\log(n) + 2n^{2+c\log((1-p)/p)} n^{\frac{cH(1-p/(1-p))}{1-p/(1-p)} + \frac{c\log(p/(1-p))}{1-p/(1-p)}} \right) \\
&= \frac{1}{T}\frac{1}{n^{c\log(1-p)}} \left(c\log(n) + 2n^{2+c\log((1-p)/p) + \frac{cH(1-p/(1-p))}{1-p/(1-p)} + \frac{c\log(p/(1-p))}{1-p/(1-p)}} \right)
\end{aligned} \tag{81}$$

when $k = c\log(n)$ for c constant and $p < 0.5$. Let

$$\omega_c(p) = 2 + c\log((1-p)/p) + \frac{cH(1-p/(1-p))}{1-p/(1-p)} + \frac{c\log(p/(1-p))}{1-p/(1-p)} \tag{82}$$

Plugging this into McDiarmid's inequality, and setting $\epsilon = 1/2$, we have

$$\begin{aligned}
\Pr\left(\hat{N}_x - N_x \ge \frac{1}{2}\right) &\le \exp\left(-\frac{2\left(\frac{1}{2}\right)^2}{nT\left(\frac{1}{T}\frac{1}{n^{c\log(1-p)}}\left(c\log(n) + 2n^{\omega_c(p)}\right)\right)^2} \right) \\
&= \exp\left(-\frac{T}{2n\left(\frac{1}{n^{c\log(1-p)}}\left(c\log(n) + 2n^{\omega_c(p)}\right)\right)^2} \right)
\end{aligned} \tag{83}$$

Setting $\delta = \Pr\left(\hat{N}_x - N_x \ge \frac{1}{2}\right)$, we have

$$\log(1/\delta) 2n \left(\frac{1}{n^{c\log(1-p)}}\left(c\log(n) + 2n^{\omega_c(p)}\right)\right)^2 \tag{84}$$

traces suffice for recovering N_x with probability at least $1-\delta$.

There are $2^{c\log(n)}$ strings x in total to test, so if we want to upper bound the probability of making making a mistake in estimating any x using the number of traces above, we can use the union bound. Setting $\delta = \frac{1}{n2^{c\log(n)}}$, we have

$$\begin{aligned}
&\log(n2^{c\log(n)})2n\left(\frac{1}{n^{c\log(1-p)}}\left(c\log(n)+2n^{\omega_c(p)}\right)\right)^2 \\
&= \log(e^{log(n)+c\log(n)\log(2)})2n\left(\frac{1}{n^{c\log(1-p)}}\left(c\log(n)+2n^{\omega_c(p)}\right)\right)^2 \\
&= 2n(log(n)+c\log(n)\log(2))\left(\frac{1}{n^{c\log(1-p)}}\left(c\log(n)+2n^{\omega_c(p)}\right)\right)^2 \\
&= O(\text{poly}(n)) \qquad (85)
\end{aligned}$$

traces suffice.

Using the union bound on the probability of error over all $2^{c\log(n)}$ kmers, we get a probability of error upper bounded by

$$2^{c\log(n)}\frac{1}{n2^{c\log(n)}} = \frac{1}{n}. \qquad (86)$$

Thus for $p < 0.5$, we reconstruct the the $c\log(n)$-spectrum of s with probability greater than $1-\frac{1}{n}$ using $O(\text{poly}(n))$ traces.

D. Expansion of (11)

We carry out one recursive step of the expansion of (11) in order to help illuminate the argument:

$$\begin{aligned}
K_{s,x}[i] &= \frac{1}{(1-p)^k}\left(P_{s,x}[i] - \sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)}(1-p)^\ell\binom{y}{x}' K_{s,y}[i]\left(\frac{p}{1-p}\right)^{\ell-k}\right) \\
&= \frac{1}{(1-p)^k}\left(P_{s,x}[i] - \sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)}(1-p)^\ell\binom{y}{x}'\right. \\
&\quad \left.\cdot\frac{1}{(1-p)^\ell}\left(P_{s,y}[i] - \sum_{j=\ell+1}^{n}\sum_{z\in Y_j(y)}(1-p)^j\binom{z}{y}' K_{s,z}[i]\left(\frac{p}{1-p}\right)^{j-\ell}\right)\left(\frac{p}{1-p}\right)^{\ell-k}\right) \\
&= \frac{1}{(1-p)^k}\left(P_{s,x}[i] - \sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)}\binom{y}{x}' P_{s,y}[i]\left(\frac{p}{1-p}\right)^{\ell-k}\right. \\
&\quad \left.+\sum_{\ell=k+1}^{n}\sum_{y\in Y_\ell(x)}\binom{y}{x}'\left(\sum_{j=\ell+1}^{n}\sum_{z\in Y_j(y)}(1-p)^j\binom{z}{y}' K_{s,z}[i]\left(\frac{p}{1-p}\right)^{j-k}\right)\right).
\end{aligned}$$

Observe that as we expand (11) one step at a time to eventually obtain (12), every time we obtain a new term involving $P_{s,y}[i]$ in the expansion with coefficient $c_y\left(\frac{p}{1-p}\right)^{|y|-k}$, in the next step of the expansion we obtain a term involving $P_{s,z}[i]$ with coefficient $-c_y\binom{z}{y}'\left(\frac{p}{1-p}\right)^{|z|-k}$ for every $z \in \cup_{\ell=|y|+1}^{n} Y_\ell(y)$.