Homomorphic Fingerprints under Misalignments: Sketching Edit and Shift Distances

Alexandr Andoni
Microsoft Research SVC
andoni@microsoft.com

Assaf Goldberger
Tel Aviv University
assafg@post.tau.ac.il

Andrew McGregor*
University of Massachusetts
mcgregor@cs.umass.edu

Ely Porat†
Bar-Ilan University and
University of Michigan
porately@cs.biu.ac.il

ABSTRACT

Fingerprinting is a widely-used technique for efficiently verifying that two files are identical. More generally, *linear sketching* is a form of lossy compression (based on random projections) that also enables the "dissimilarity" of non-identical files to be estimated. Many sketches have been proposed for dissimilarity measures that decompose coordinate-wise such as the Hamming distance between alphanumeric strings, or the Euclidean distance between vectors. However, virtually nothing is known on sketches that would accommodate alignment errors. With such errors, Hamming or Euclidean distances are rendered useless: a small misalignment may result in a file that looks very dissimilar to the original file according such measures.

In this paper, we present the first linear sketch that is robust to a small number of alignment errors. Specifically, the sketch can be used to determine whether two files are within a small Hamming distance of being a cyclic shift of each other. Furthermore, the sketch is homomorphic with respect to rotations: it is possible to construct the sketch of a cyclic shift of a file given only the sketch of the original file. The relevant dissimilarity measure, known as the *shift distance*, arises in the context of embedding edit distance and our result addressed an open problem [26, Question 13] with a rather surprising outcome. Our sketch projects a length n file into $D(n) \cdot \operatorname{polylog} n$ dimensions where $D(n) \ll n$ is the number of divisors of n. The striking fact is that this is near-optimal, i.e., the $D(n)$ dependence is inherent to a problem that is ostensibly about lossy compression.

In contrast, we then show that any sketch for estimating the *edit distance* between two files, even when small, requires sketches whose size is nearly linear in n. This lower bound addresses a long-standing open problem on the low distortion embeddings of edit distance [36, Question 2.15], [24], for the case of *linear* embeddings.

Categories and Subject Descriptors

F.2.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

Keywords

fingerprinting; sketching; edit distance; lower bounds

1. INTRODUCTION

Fingerprinting is a widely-used technique for efficiently verifying that two files are identical. The idea is to map the files to short bit strings such that with high probability, the files are identical if and only if the short strings are identical. Examples include the "rolling" fingerprint of Karp and Rabin [31] and cryptographic hash functions such as MD5 [46]. More generally, *linear sketching* is a form of lossy compression based on random linear projections that also enables one to estimate the "dissimilarity" of two non-identical files. Concretely, we represent the two files as vectors $x, y \in \mathbb{R}^n$ and consider a matrix $A \in \mathbb{R}^{k \times n}$ chosen from some carefully chosen distribution. The distance between x and y is then estimated from the sketches $Ax, Ay \in \mathbb{R}^k$. The goal is to ensure a given approximation guarantee while minimizing $k \ll n$, the *dimension* or *size* of the sketch.

The linearity of sketches confers numerous advantages. For one, the technique is applicable in distributed, parallel, streaming environments, and in network coding [45] since the sketch is homomorphic with respect to linear operations, i.e., $\alpha A(x) + \beta A(y) = A(\alpha x + \beta y)$. Furthermore, linearity often allows for identifying the actual difference $x - y$, assuming it is sparse. Thus, for example, a lot of attention has been devoted to developping sketches for estimating the Hamming distance between two strings [14, 30, 41, 42], the ℓ_p norm of their difference [29], as well as determining the position of mismatches, e.g., heavy hitters or sparse recovery [12, 13, 16, 19, 21, 28, 37, 39, 44]. Note that the problem of finding mismatches has (re)appeared in various contexts such as set reconciliation problem; invertible bloom filters; group testing [38, 43]; and others.

However, nearly all the distances considered to date are not robust to *alignment errors*. For example, a single insertion at the start of a file will result in a file that looks very

*Supported by NSF CAREER Award CCF-0953754.
†This work was supported by a Google award.

STOC'13, June 1–4, 2013, Palo Alto, California, USA.
Copyright 2013 ACM 978-1-4503-2029-0/13/06 ...$15.00.

dissimilar to the original file according to the Hamming measure.

Edit Distance and Shift Distance. To address the issue of misalignment, researchers have considered distances that are more robust to misalignments, in particular the *edit distance* and the *shift distance.* The edit distance (also known as the *Levenshtein* distance) between two strings $x, y \in \Sigma^n$, denoted $\mathrm{ed}(x, y)$, is defined as the minimum number of character insertions, deletions, and substitutions needed to transform one string into the other. Although it has been heavily studied, edit distance still resists efficient algorithms, including good linear sketches. The shift distance $\mathrm{sh}(x, y)$ between $x, y \in \Sigma^n$ is defined as the minimal Hamming distance between x and some cyclic shift (rotation) of y. While edit distance is the canonical distance between strings, shift distance is also natural in signal processing applications. For example, two similar periodic signals recorded at different phases would result in two files that are close under some cyclic shift.

Numerous connections between edit distance and shift distance have arisen in the context of metric embeddings. Shift distance has emerged as a useful distance that is "alignment-friendly" and simple, yet still captures a core hardness structure present in the edit distance. As such, shift distance has already proven instrumental for the discovery of both lower and upper bounds for the edit distance. For example, shift distance has a $\Omega(\log n)$ lower bound on distortion (approximation) for embedding into ℓ_1 [32], which already demonstrates the increased difficulty of dealing with misalignments over, say, the simpler Hamming distance. This lower bound essentially leads to the same bound for edit distance as well [33]. The shift distance is also the core gadget for the lower bound for asymmetric query complexity of edit distance from [5] (the full construction uses a recursive shift distance). Considering these hard instances has then led to a (near-tight) upper bound, which also yields a sub-quadratic time edit distance estimation algorithm with a polylogarithmic approximation [5].

Sketching shift distance has been raised as an open question in [26]. The only previous result is a $1+\epsilon$ approximation sketch of size $O(\sqrt{n/\epsilon})$ [18]. Even when allowing an arbitrary (non-linear) sketches, the only other result achieves $O(\log^2 n)$-approximation in polylogarithmic space [4]. The similar question for edit distance has been studied, and the best approximation for small space is $2^{\tilde{O}(\sqrt{\log n})}$ (again, by a non-linear sketch) [9–11, 40].

We note that the question becomes somewhat easier when only one of the strings is being compressed or sketched. For example, [17, 27] show how to compute the distance and difference under the edit distance and other related distances efficiently if one of the strings is known in full. There is also a body of work on designing fingerprints that measure the similarity between sets or item rankings [6–8, 20, 34]. However, these results are not directly relevant since two sequences may have large edit distance or shift distance even if they are composed of the same set of elements.

Homomorphic Compression. A larger context for our work is the study of lossy compression schemes that support a range of operations on the compressed data without access to the original data. For example, rather than just having a sketch L that is homomorphic with respect to linear operations, one may want to be homomorphic with respect to cyclic shifts. For example, given sketches $L(x)$ and $L(y)$ of two strings $x = x_1 x_2 \ldots x_n$ and $y = y_1 y_2 \ldots y_n$, we want to be able to compute

$$L(\sigma^s(x + y)) \text{ where } \sigma^s(x + y)_i = x_{i-s \bmod n} + y_{i-s \bmod n}$$

for any $s \in \{0, 1, 2, \ldots, n - 1\}$.

We are interested in exploring the limits of the linear sketching technique. Specifically, we are interested in developing a theory of homomorphic compression based on the fact that the linear property of the sketch ensures that the projection preserves certain structural properties. For example, given $x, y, z \in \mathbb{R}^n$, it is not difficult to show that it is possible to determine whether $\{x, y, z\}$ is linearly independent given $Ax, Ay, Az \in \mathbb{R}^k$ where $A \in \mathbb{R}^{k\times n}$ is a random projection with $k = O(\log n)$. A perhaps more surprising result is that it is possible to determine the connectivity properties of a graph on n nodes given only a $O(\mathrm{polylog}\, n)$-dimensional projection of each of the n adjacency lists [2, 3]. While other approximate data structures are "mergable" [1], the linearity of sketches offers the promise of supporting a much richer set of algorithmic operations.

1.1 Our Results and Discussion

In this paper we present three main results on homomorphic sketches for the alignment distances described above.

1. *Shift Distance Sketch:* We present a linear sketch of size $k = O(D(n) \cdot \mathrm{polylog}\, n)$ for the shift distance where $D(n)$ is the number of divisors of n. The sketch also supports homomorphic cyclic shifts. Our sketches can be used for distinguishing between $\mathrm{sh}(x, y) > 0$ and $\mathrm{sh}(x, y) = 0$ where $x, y \in \mathbb{Z}_m^n$ and $m = \mathrm{poly}(n)$. More generally, our sketches can distinguish between the cases $\mathrm{sh}(x, y) = 0, 1, \ldots, t$ and $\mathrm{sh}(x, y) > t$, using space $O((t + D(n)) \cdot \mathrm{polylog}\, n)$. Since the number of divisors of n satisfies $D(n) = n^{O(1/\log\log n)}$, the size of the sketch is always significantly sub-linear in n and typically $D(n) = O(\log n)$. We also show efficient algorithms for computing the shift distance using our sketches, as well as for identifying the shift and any differing positions.

2. *Shift Distance Lower Bound:* We present a near matching lower bound for testing whether $\mathrm{sh}(x, y) = 0$ where $x, y \in \mathbb{Z}_m^n$. Specifically, we show that any linear homomorphic sketch has dimension $\Omega(D(n) \log m/ \log nm)$.

These first two results address an open question posed in [26, Question 13] with a rather surprising outcome. When viewing sketching as a form a compression, it is natural to view the size of a sketch in terms of the amount information that needs to be preserved by the sketch. As such, it would seem natural to assume that the size of an optimal sketch size would be monotonically increasing in n, the size of the data being sketched. Instead, our results show that the optimal sketch size is proportional to the number of divisors of the n, and is thus far from being monotonic.

3. *Edit Distance Lower Bound:* We show that any linear sketch that distinguishes the cases $\mathrm{ed}(x, y) = 2$ and $\mathrm{ed}(x, y) = 1$ has size $\Omega(n)$. More generally, we show that sketch that distinguishes $\mathrm{ed}(x, y) \geq 2\alpha$ and

ed$(x, y) \leq 2$ has size $\Omega(n/\alpha)$. This result on edit distance is in stark constrast to the upper bound for shift distance from above.

We note also that edit distance lower bound has implications for a long-standing open question on the best distortion embeddings for the edit distance [36, Question 2.15], [24]. Namely our result implies that any *linear embedding* of edit distance in ℓ_1 has $\Omega(n)$ distortion.

2. TECHNIQUES AND PRELIMINARIES

In this section, we describe our general approach to the sketch construction, as well as the required number theoretic preliminaries.

We will use $[n]$ to denote the set $\{0, 1, 2, \ldots, n-1\}$. For a string $x \in \Sigma^n$ and index set $I \subset \{1, \ldots n\}$, $x|_I$ is the projection of x onto coordinates in I. Let $D(n)$ denote the number of divisors of n.

2.1 Our Techniques

We describe the ideas behind the construction of the sketch for the shift distance as well as fast algorithms for using the sketches. The ideas for our lower bounds are described in Section 4.

The departing point of our sketch is the Karp–Rabin fingerprint [31]. The latter maps a string $a \in \mathbb{Z}_m^n$ into a fingerprint (hash value)

$$f_a = \sum_{i=0}^{n-1} r^i a_i \mod p$$

for a random prime p and some constant $r \in \mathbb{Z}_p \setminus \{0, 1\}$. Then, for a different string $b \in \mathbb{Z}_m^n$, we have that $f_a \neq f_b$ with good probability (over the choice of the prime p). The advantage of this fingerprint is that, using f_a, one can very efficiently compute the fingerprint of a related string, namely the string $a' = (t, a_0, a_1, \ldots a_{n-2})$ where $a = (a_0, \ldots a_{n-1})$ and $t \in \mathbb{Z}_m$. Specifically, one computes the new fingerprint as $f_{a'} = rf_a + t - r^n a_{n-1}$. Note that for $t = a_{n-1}$, we obtain the fingerprint of the cyclic shift of a, namely $f_{a'} = f_{\sigma(a)}$. With such fingerprints, one can verify whether $\mathrm{sh}(a, b) = 0$ as follows: just check whether $f_{\sigma^s(a)} - f_b = 0$ for any $s \in [n]$. However, in our setting, we cannot use this fingerprint because, to compute $f_{\sigma^s(a)}$, one has to also know $a_{n-1}, a_{n-2}, \ldots a_{n-s}$.

We want to be able to compute the fingerprint $f_{\sigma(a)}$ of the shift of a from f_a alone, without the knowledge of a_{n-1}. A natural attack is to choose p and r such that $r^n = 1 \mod p$ since then $f_{\sigma(a)} = rf_a$. This leads us to consider using Fermat's little theorem, which says that, at least when $p = n+1$ is a prime, we have $r^n = 1 \mod p$ for all $r \in \mathbb{Z}_p^\times$, the multiplicative group of integers modulo p. Leaving aside the issue of a number-theoretic restriction on n, we cannot choose the prime p randomly anymore (and it can be shown that there is no deterministic fingerprint with the desired properties). It is now tempting to choose r randomly and hope that, for a non-zero polynomial

$$f_a(r) = \sum_{i=0}^{n-1} a_i r^i,$$

we have $f_a(r) \neq 0$ for a random r with good probability. In order to bound the number of roots r such that $f_a(r) = 0$, one could use, say, Schwartz-Zippel lemma. However, since $f_a(r)$ may potentially have degree as large as nearly $p = n+1$, one obtains a success probability as small as $\Theta(1/n)$.[1]

Instead, our approach is to choose the base r very carefully, while continuing to use a random p. In particular, we choose r such that it is a root of the equation $r^n = 1 \mod p$ (i.e., r is an nth root of unity). Implemented naively, this approach also runs into trouble, stemming from the fact that no fixed r is sufficient. For example, $r = 1$ seems like a bad choice (even in the original Karp-Rabin fingerprint) since any permutation of a gives precisely the same fingerprint. But, any other root $r \neq 1$ is also root of the polynomial $(r^{n-1} + r^{n-2} + \ldots + r + 1)$ since

$$r^n - 1 = 0 \bmod p \implies (r-1)(r^{n-1} + \ldots + r + 1) = 0 \bmod p.$$

In other words, the polynomial $f_a(r) = r^{n-1} + \ldots + r + 1$ for $a = (1, 1, \ldots 1)$ is always zero modulo p for any $r \neq 1$.

To address this difficulty, our sketch will use several roots $r_1, r_2, \ldots, r_k$ so that for a non-zero polynomial $f_a(r)$, there will be at least one root such that $f_a(r_i) \neq 0$ with good enough probability (over the choice of prime p). In particular, we show that it is enough to (carefully) choose $D(n)$ such roots r_i to satisfy the above property. To construct the roots $r_1, r_2, \ldots$, we look at the possible factorizations of the polynomial $r^n - 1$, which are well characterized by the *cyclotomic polynomials* (see below for a definition and the relevant properties). Furthermore, to prove that our polynomials f_a do not have r_i's as roots, we use the theory of resultants (again, see below).

We also show efficient algorithms for verifying whether $f_a(r_i) = f_{\sigma^s(b)}(r_i)$ for some cyclic shift s. Note that this amounts to computing the shift s such that $f_a(r_i) = r_i^s f_b(r_i)$. We show how to do this quickly (faster than enumerating over all possible shifts s). We note that it is not always possible to determine s accurately — for example when the strings are periodic and hence s is not even well defined. In this case we show that we can determine $s \bmod d_i$ for enough moduli d_i so that, using the Chinese Remainder Theorem, one can determine the shift s modulo the period of the strings.

Finally, to determine the actual difference between a and $\sigma^s(b)$ whenever $a - \sigma^s(b)$ is sparse, we further use a fast syndrome decoding algorithm [15].

2.2 Number Theory Preliminaries

We now summarize the main definitions and results in number theory that we will use in this paper. The interested reader may want to consult Hardy and Wright [23] for proofs and further background.

Lemma 1 (Cyclotomic Polynomials). *The d-th* cyclotomic polynomial *is defined as:*

$$\Phi_d(X) = \prod_{1 \leq k \leq d, gcd(k,d)=1} (X - e^{2i\pi k/d}) \ .$$

Properties of cyclotomic polynomials include:

1. *$\Phi_d(X) \in \mathbb{Q}(X)$ and is the minimal polynomial in $\mathbb{Q}$ for any primitive root of $X^d - 1$. In particular it is irreducible, i.e., has no non-constant factors in $\mathbb{Q}[X]$.*

[1] One might hope to get more flexibility by instead appealing to Euler's theorem: $r^{\varphi(q)} = 1 \bmod q$ if r and p are co-prime and $\varphi(q)$ is Euler's totient function. However this approach seems to run into the exact same difficulty.

2. *$X^n - 1$ factorizes into cyclotomic polynomials:*

$$X^n - 1 = \prod_{d|n} \Phi_d(X)$$

and therefore the number of factors of $X^n - 1$ is $D(n)$, the number of divisors of n, including 1 and n.

For example,

$$\begin{aligned} X^6 - 1 &= \Phi_1(X)\Phi_2(X)\Phi_3(X)\Phi_6(X) \\ &= (X-1)(X+1)(X^2+X+1)(X^2-X+1) \ . \end{aligned}$$

THEOREM 2 (ABEL'S IRREDUCIBILITY THEOREM). *Let $g(X) \in \mathbb{Q}[X]$ be irreducible over $\mathbb{Q}$ and $f(X) \in \mathbb{Q}[X]$ share a root with $g(X)$. Then $f(X)$ shares all the roots of $g(X)$.*

Therefore, if $f(X)$ and $X^6 - 1$ share a root, there exists $d \in \{1, 2, 3, 6\}$ such that $\Phi_d(X)$ is a factor of $f(X)$.

LEMMA 3 (RESULTANT). *The* resultant *of two monic polynomials $f(X), g(X) \in \mathbb{F}[X]$ is defined as*

$$\begin{aligned} \text{Res}(f, g) &= \prod_{x,y \in \mathbb{G}: f(x)=0, g(y)=0} (x - y) \\ &= \prod_{x \in \mathbb{G}: f(x)=0} g(x) = \prod_{x \in \mathbb{G}: g(x)=0} f(x) \ . \end{aligned}$$

where $\mathbb{G}$ is the algebraic closure of $\mathbb{F}$. Note that $\text{Res}(f, g) = 0$ iff f and g have a common root. Furthermore, if f and g have degree $n - 1$ and coefficients in the set $\{-m, -m + 1, \ldots, m\}$, then

$$|\text{Res}(f, g)| \in \{0, 1, \ldots, (2nm)^{2n}\} \ .$$

PROOF. This follows because the resultant can be expressed as the determinant of the Sylvester matrix associated with f and g [47]. Specifically, if $f(x) = \sum_{0 \le i \le n-1} a_i x^i$ and $g(x) = \sum_{0 \le i \le n-1} b_i x^i$ then $\text{Res}(f, g) = \det A$ where A is the $(2n - 2) \times (2n - 2)$ matrix of the form:

$$A = \begin{pmatrix} a_{n-1} & a_{n-2} & \ldots & 0 & 0 \\ 0 & a_{n-1} & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & a_0 & 0 \\ 0 & 0 & \ldots & a_1 & a_0 \\ b_{n-1} & b_{n-2} & \ldots & 0 & 0 \\ 0 & b_{n-1} & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 \\ 0 & 0 & \ldots & b_0 & 0 \\ 0 & 0 & \ldots & b_1 & b_0 \end{pmatrix}$$

and so $|\text{Res}(f, g)| \le (2nm)^{2m}$ if the magnitude of every coefficient if at most m. □

LEMMA 4 (DENSITY OF PRIMES). *For any $n, k \in \mathbb{N}$, define the following set of primes:*

$$C_{n,k} = \{p \text{ prime} : p \le k, \ X^n - 1 \text{ has } n \text{ distinct roots in } \mathbb{Z}_p\} \ .$$

Then, $|C_{n,k}| = \Omega(k/(n \ln k))$.

PROOF. For every prime p of the form $mn + 1$, $X^n - 1$ has n distinct roots. The number of primes less than k, of the form $mn + 1$ is $\Omega(k/(n \ln k))$. □

3. UPPER BOUNDS

In this section we present the basic linear sketch for determining whether two strings are identical up to a shift. We then extend the result to handle the case when the two strings are within small Hamming distance of being identical after some shift.

3.1 The Basic Sketch

We prove the following theorem.

THEOREM 5 (CYCLIC SHIFT SKETCH). *Let $n, m \ge 1$. There exists a randomized function $L : \mathbb{Z}_m^n \to \{0, 1\}^s$ where $s = O(D(n) \cdot \text{polylog}\, n)$ that satisfies the following conditions:*

1. *Sketch: There is an algorithm such that, given sketches $L(a), L(b)$ of any two strings $a, b \in \mathbb{Z}_m^n$, declares whether $\text{sh}(a, b) = 0$ with probability of success at least $2/3$;*

2. *Shift homomorphism: Given $L(a)$ for some string $a \in \mathbb{Z}_m^n$ and cyclic shift σ, one can compute $L(\sigma(a))$;*

3. *Linearity: Given $L(a)$ and $L(b)$ for some $a, b \in \mathbb{Z}_m^n$, we can compute the sketch $L(a + b)$.*

Note that the existence of such a sketch naturally leads to an algorithm for checking whether a is a cyclic shift of b: first we boost the probability of success to $1 - 1/\text{poly}(n)$ by taking the median result of $O(\log n)$ independent sketches and then compare $L(a)$ to $L(\sigma^s(b))$ for each of the n cyclic shifts σ^s. However, this verification algorithm would take $\tilde{O}(D(n) \cdot n)$ time. In Section 3.2, we show how extend the algorithm to support $\tilde{O}(D(n))$ query time. We prove Theorem 5 below.

Sketch Outline. Before presenting the outline of the basic sketch algorithm, we introduce some further notation. Given a vector $a, b \in [m]^n$ we start by considering the natural encoding of a as degree $n - 1$ polynomials $f_a \in \mathbb{Q}[X]$:

$$f_a(x) = \sum_{i=0}^{n-1} a_i x^i.$$

Then, given another vector $b \in [m]^n$ with associated polynomial f_b, for each shift $s \in [n]$, we define $g_s \in \mathbb{Q}[X]$

$$g_s(x) = f_a(x) - x^s f_b(x).$$

Also let $h_s(x)$ be the degree $n - 1$ polynomial formed by replacing all terms x^i in $g_s(x)$ by $x^{i \bmod n}$. For example, with $n = 3$ and $s = 2$,

$$g_s(x) = a_0 + a_1 x + (a_2 - b_0)x^2 - b_1 x^3 - b_2 x^4$$

and $h_s(x) = (a_0 - b_1) + (a_1 - b_2)x + (a_2 - b_0)x^2$.

The components of the sketch are as follows:

1. *Initialization.* Let p be a random prime with certain properties to be determined and let $r_1, r_2, \ldots, r_{D(n)}$ be $D(n)$ specific roots of the equation $x^n - 1 = 0 \mod p$.

2. *Sketch construction.* For string $a \in \mathbb{Z}_m^n$, we compute $f_a(r_i) \mod p$ for all $i \in \{1, 2, \ldots, D(n)\}$. Their concatenation is our sketch $L(a)$.

3. *Shift homomorphism.* For $L(a) = (f_a(r_1), \ldots f_a(r_{D(n)}))$, we have $L(\sigma(a)) = (r_1 f_a(r_1), \ldots r_{D(n)} f_a(r_{D(n)}))$.

4. *Identity checking (query algorithm).* For two strings a, b, if there exists $s \in [n]$ with $g_s(r_i) = 0 \mod p$ for all $i \in D(n)$, claim strings a and b are identical up to a cyclic shift, and in particular the shift is s.

Details and Analysis. First, note that "linearity" and "shift homomorphism" properties of Theorem 5 are automatically satisfied because the sketch is linear and all the roots r_i are such that $r_i^n = 1$. We now argue that the "sketch" property holds.

It is straightforward to argue that the algorithm has no false negatives. Suppose that b is a cyclic shift of a with shift s, i.e., for $i = 0, \ldots, n-1$, $b_i = a_{i+s \mod n}$. Then, for any r with $r^n - 1 = 0 \mod p$, we have

$$\begin{aligned} g_s(r) &\equiv_p h_s(r) \\ &\equiv_p (a_0 - b_{n-s}) + (a_1 - b_{1+n-s})r + \ldots \\ &\quad + (a_{n-1} - b_{n-s-1})r^{n-1} \\ &\equiv_p 0. \end{aligned}$$

The challenge is therefore to ensure that the probability of a false positive is small. To do this we will carefully chose p, $r_1, r_2, \ldots$, and $r_{D(n)}$.

Let p be a prime such that $X^n - 1 = 0 \mod p$ has n distinct roots, e.g., a prime $p = tn+1$ for some $t \in \{1, 2, \ldots\}$. Consider the factorization of $X^n - 1 \in \mathbb{Q}[X]$ into irreducible polynomials

$$X^n - 1 = \prod_{d|n} \Phi_d(X) = c_1(X)c_2(X)\ldots c_{D(n)}(X) \ .$$

and define $r_i \in \mathbb{Z}_p$ to be a root of $c_i(X) = 0 \mod p$ (chosen arbitrarily when $c_i(X)$ has multiple roots).

Suppose that a and b do not satisfy $b_i = a_{i+s \bmod n}$ for all $i = 0, \ldots, n-1$ and therefore $h_s(x)$ is a non-zero polynomial of degree $n-1$. Given that $a, b \in \mathbb{Z}_m^n$, the coefficients of $h_s(x)$ are in $\{-m, -m+1, \ldots, m-1, m\}$.

Lemma 6. *There exists i such that* $\mathrm{Res}(c_i, h_s) \neq 0$.

Proof. Let $c(X) = X^n - 1$. Since $h_s(X)$ has degree $n-1$ and $c(X)$ has degree n, there exists a root r of c such that $h_s(r) \neq 0$. Given the factorization of $c(X)$, there exists $c_i(X)$ such that $c_i(r) \neq 0$ and by Lemma 1, c_i is irreducible. Therefore, by appealing to Theorem 2, we know that every root of c_i is not a root of h_s. Hence, $\mathrm{Res}(c_i, h_s) \neq 0$ as required. □

We next need to argue that c_i and h_s also have no shared roots (with high probability) when viewed as polynomials over $\mathbb{Z}_p$.

Lemma 7. *Let $p \in_R C_{n,k}$ where $k = n^5$. Then,*

$$\Pr[c_i, h_s \textit{ share a root over } \mathbb{Z}_p] \leq \frac{2n\log(2nm)}{|C_{n,k}|} = O(1/n^2) \ .$$

Proof. Consider the resultant of c_i and h_s when viewed as polynomials over $\mathbb{Q}$:

$$\mathrm{Res}(c_i, h_s) = \prod_{x,y \in \mathbb{C}: c_i(x)=0, h_s(y)=0} (x - y) = \prod_{y \in \mathbb{C}: c_i(y)=0} h_s(y)$$

The resultant of c_i and h_s when viewed as polynomials over $\mathbb{Z}_p$ equals $\prod_{y \in \mathbb{C}: c_i(y)=0} h_s(y) \mod p$. By Lemma 3, we know that $\prod_{y \in \mathbb{C}: c_i(y)=0} h_s(y)$ is an integer with maximum absolute value $(2nm)^{2n}$ and there has at most $2n\log(2nm)$ prime divisors. Since p is chosen randomly from $C_{n,k}$ with probability at least $1 - \frac{2n\log(2nm)}{|C_{n,k}|}$, the resultant of c_i and h_s when viewed as polynomials over $\mathbb{Z}_p$ is non-zero and hence they do not share a root. The bound follows since for $k = n^5$, $|C_{n,k}| = \Omega(n^4/\log n)$. □

By applying the union bound over all choices of s, we ensure that for $a, b \in \mathbb{Z}_m^n$ that are not cyclic shifts,

$$\Pr[\forall s \in [n];\ \exists i;\ g_s(r_i) \neq 0 \bmod p] \geq 1 - O(1/n) \ .$$

This completes the proof of Theorem 5.

3.2 Reducing the Query Time

We now show how improve the query time of the sketch. Note that the naive method from the preceding section would use $\tilde{O}(nD(n))$ time since it required verifying n possible cyclic shifts. We now show how to reduce this time to $D(n)\,\mathrm{polylog}\, n + 2^{O(\log^{1/3} n \log\log^{2/3} n)}$. In fact, our algorithm will find the shift j under which the two strings are supposed to be equal, and then verify whether the strings are indeed equal under shift j using the result from the previous section.

To accomplish this we choose the roots r_i of $c_i(X)$ more carefully. For $c_i(X) = \Phi_{d_i}$, we take r_i to be a root that has exact order d_i, i.e., $r_i^{d_i} = 1$ and there is no $\ell < d_i$ such that $r_i^\ell = 1$. In Section 3.2.1 we prove that r_i is indeed a root of Φ_{d_i}, as well as show how to construct r_i's.

It is clear that if the strings are not equal under any shift j, the algorithm will report it so with high probability, due the result from the previous section. The question then boils down to finding a shift j under which the strings are equal, if such exists.

To find the shift j, we consider the equations $f_a(r_i) = r_i^j f_b(r_i) = \mod p$ that must be satisfied by the shift j. Given that $f_b(r_i) \neq 0$ we calculate $j \bmod d_i$, using a discrete-log algorithm in the equation $r_i^j = f_a(r_i)/f_b(r_i) \bmod p$. Applying the Chinese Remainder Theorem on the answers for each r_i, we thus determine j.

Next we prove the correctness of the above algorithm. We will use the following notation: for $d' \mid n$, the index $i_{d'} \in \{1, \ldots, D(n)\}$ will stand for the index i such that $d_i = d'$.

Lemma 8. *Let l be the period of the string where $l = n$ if the string is aperiodic. Assuming b is not identically zero, the least common multiple (lcm) of all d_i for which $f_b(r_i) \neq 0$ will be l with high probability. Therefore we can determine the shift $j \bmod l$.*

Proof. Assume $b = u^{n/l}$ for some string $u \in \mathbb{Z}_m^l$, then

$$f_b(X) = (1 + X^l + X^{2l} + \ldots + X^{n-l})f_u(X) \ .$$

Notice that

$$X^n - 1 = (1 + X^l + X^{2l} + \ldots + X^{n-l})(X^l - 1) \ ,$$

and hence

$$(1 + X^l + X^{2l} + \ldots + X^{n-l}) = \prod_{d'|n \wedge d' \nmid l} \Phi_{d'}(X) \ .$$

For all $d'|n \wedge d' \nmid l$, we have $\Phi_{d'}(X)|f_b(x)$ and therefore we will get $f_b(r_{i_{d'}}) = 0 \bmod p$. Therefore the lcm can be at most l.

Now we will prove that the lcm is l with high probability. We assume that u is the minimal period and $0 <$

$deg(f_u(X)) \le l-1$ (which happens whenever b is not identically zero). Therefore for every $d'|l$ we have

$$(1+X^{d'}+X^{2d'}+\dots X^{l-d'}) \nmid f_u(X) .$$

Assume the lcm is $\ell' \mid l$, in which case also

$$(1+X^{\ell'}+X^{2\ell'}+\ldots+X^{l-\ell'}) \nmid f_u(X) .$$

Therefore there exist $d'|l \wedge d' \nmid \ell'$ such that $\Phi_{d'} \nmid f_u(X)$. We proved before that choosing p randomly in this case imply that $f_u(r_{i_{d'}}) \bmod p \neq 0$ which also implies that $f_b(r_{i_{d'}}) \bmod p \neq 0$. □

Notice that we do not need to run discrete-log $D(n)$ times. It is enough to choose $i_1, i_2, \ldots, i_\ell$ such that $f_b(r_{i_j}) \neq 0$ and the lcm of d_{i_j} is equal to the lcm of all d_i. This means that in the worst cast we run discrete-log only $\log D(n)$ times.

The time complexity of discrete-log [22] is

$$2^{O((\log n)^{1/3}(\log\log n)^{2/3})}$$

and therefore the resulting time to find the shift is

$$D(n) \operatorname{polylog} n + 2^{O((\log n)^{\frac{1}{3}}(\log\log n)^{\frac{2}{3}})} .$$

3.2.1 Constructing the appropriate roots of unity

We now describe how to construct the appropriate roots r_i. First we show that the r_i which has exact order d_i is indeed a root of Φ_{d_i}.

Lemma 9. *If r_i has exact order d_i, it is a root of Φ_{d_i}.*

Proof. Since r_i has exact order d_i, r_i is a root of the polynomial

$$X^{d_i} - 1 = \prod_{d'|d_i} \Phi_{d'}(X) .$$

On the other hand it is not a root of $X^{d'}-1$ for every proper divisor $d'|d$ and hence not a root of $\Phi_{d'}(X)$. We conclude that r_i is a root of Φ_{d_i}. □

Fix an integer n, and a prime p with $p \equiv 1 \mod n$. Let $X^n - 1 = \prod_{d|n} \Phi_d(X)$ be the decomposition into cyclotomic polynomial. We want to choose elements $\{r_{i_d}\}_{d|n}$ in $\mathbb{F}_p^\times$ such that r_{i_d} is a root of Φ_d. To do this, consider the prime factorization of $n = \prod_{i=1}^{r} p_i^{e_i}$. Choose a random element $r_i \in \mathbb{F}_p^\times$ and compute $t_i = r_i^{(p-1)/p_i}$. Now, the element

$$s_i := r_i^{(p-1)/p_i^{e_i}}$$

has order dividing $p_i^{e_i}$, and with probability $(1-1/p_i)$ it will have the exact order $p_i^{e_i}$. This will happen if and only if $t_i \neq 1$. Once we find t_i with $t_i \neq 1$, we compute s_i. Having collected the s_i for all i, we form the product $r = \prod s_i$ which is an element of exact order n. For every $d|n$ compute $r_{i_d} = r^{n/d}$.

3.3 Extension to Small Number of Errors

In the section, we generalize Theorem 5 to the situation when a and a cyclic shift of b are at a small Hamming distance k. In this case we show we can reconstruct the actual difference using a sketch of size $O((t+D(n)) \operatorname{polylog} n)$.

Theorem 10 (Cyclic Shift Sketch with Errors). *Let $n, m \ge 1$. There exists a randomized function $L : \mathbb{Z}_m^n \to \{0,1\}^s$ where $s = O((t+D(n)) \cdot \operatorname{polylog} n)$ that satisfies the following conditions:*

1. *Sketch: There is an algorithm such that given sketches $L(a), L(b)$ of any two strings $a, b \in \mathbb{Z}_m^n$, declares whether $\operatorname{sh}(a,b) = 0, 1, \ldots, t$ or $\operatorname{sh}(a,b) > t$ with probability of success at least $2/3$. Furthermore, we can reconstruct the shift s and $a - \sigma^s(b)$ if $H(a, \sigma^s(b)) \le t$;*
2. *Shift homomorphism: Given $L(a)$ for some string $a \in \mathbb{Z}_m^n$ and cyclic shift σ, one can compute $L(\sigma(a))$;*
3. *Linearity: Given $L(a)$ and $L(b)$ for some $a, b \in \mathbb{Z}_m^n$, we can compute the sketch $L(a+b)$.*

We assume that $t < n/2$ since otherwise we could consider the trivial sketch $L(a) = a$ and still satisfy the required size bound.

The sketch builds upon the basic sketch in Theorem 5. In addition to the sketch described earlier we also evaluate $f_a(X)$ and $f_b(X)$ on $r, r^2, \ldots, r^{2t}$ where r is an nth root of unity with exact order n. Note that $r, r^2, \ldots, r^{2t}$ are distinct because $2t < n$.

We are going to try each rotation separately. Assuming that s is the rotation that minimizes $\operatorname{sh}(a,b)$, let b' is the string formed by rotating b by s positions. Let

$$g(X) = f_a(X) - f_{b'}(X) = \sum_{i=0}^{n-1} \Delta_i X^i$$

where Δ_i is the difference between the symbols at position i. Notice that $g(r^j) = f_a(r^j) - (r^j)^s f_b(r^j)$ due to the fact that r^j also satisfies the equation $x^n = 1$. We know that $g(X)$ is a t-sparse polynomial and we have $2t$ different assignments therefore we can run Reed Solomon syndrome decoding (see, e.g., [35]) in order to find Δ_i.

Running over all possible values of s, we get n sets of candidates. We use the second sketch in order to verify which is the correct (if there is one). Due to the fact that the sketch is linear then we can easily update the sketch and correct the mismatch and therefore our algorithm will return that it equal. Due to the fact that we run it n times then the error probability will be higher by at most factor n therefore we will choose bigger prime.

The Reed-Solomon syndrome decoding can be run in time $O(t \log^2 t)$ [15]. Therefore the running time is bounded by $O(nt \log^2 t + nD(n))$.

4. LOWER BOUNDS

In this section, we first prove a matching lower bound for the shift distance sketch presented in the previous section. We then prove a bound that establishes that no similar sketch is possible for the edit distance. Both lower bounds are based on careful encodings of long strings into the shift/edit distances so that, using the linearity (and shift homomorhism in the case of shift distance), one can do efficient decoding.

4.1 Cyclic Shift Lower Bound

We start by stating the lower bound theorem.

Theorem 11. *Let $n, m \ge 1$, and let $\Sigma = \mathbb{Z}_m$ be the alphabet. Suppose there is a randomized function (sketch) $L : \Sigma^n \to \{0,1\}^s$ that satisfies the following conditions:*

1. *Sketch: There is an algorithm such that given sketches $L(a), L(b)$ of any two strings $a, b \in \mathbb{Z}_m^n$, declares whether $\operatorname{sh}(a,b) = 0$ with probability of success at least $2/3$;*

2. *Shift homomorphism: Given $L(x)$ for some string $x \in \Sigma^n$, one can compute $L(\sigma(x))$;*

3. *Linearity: Given $L(x)$ and $L(y)$ for some $x, y \in \Sigma^n$, we can compute the sketch $L(x+y)$.*

Then, the sketch size s has to be at least $s = \Omega(D(n) \cdot \log|\Sigma| / \log n|\Sigma|)$.

Before proceeding to the proof, we outline the intution that makes the lower bound possible. One of the ideas is to exploit symmetries in the group $\mathbb{Z}_n$. In particular, suppose n is a product of k primes $p_1, \ldots, p_k$, in which case we have $\mathbb{Z}_n \cong \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2} \times \ldots \times \mathbb{Z}_{p_k}$, that is we can represent the string as a function on this k-dimensional rectangle. Furthermore, a shift by n/p_i on the oringinal string corresponds to a shift by 1 in the i coordinate of this rectangle — thus the string can be seen a product of k strings which we can shift cyclically independently of each other.

We show how to exploit this product structure in order to embed a string x of length 2^k into a string $\bar{x}$ so that one can decode the entire x from the sketch $L(\bar{x})$ only. To start, we show how we can embed and decode a string $x = (x_0, x_1)$ when $n = p_1$ is just a prime. In particular, the embedding will be $\bar{x} = (x_0, x_1, 0, 0, \ldots, 0)$. At decoding, one can learn the *sum* of the two characters using rotations as follows:

$$L((x_0 + x_1, x_0 + x_1, \ldots x_0 + x_1)) = \sum_{j=0}^{n-1} L(\sigma^j(\bar{x}))) \ .$$

Hence we can obtain a sketch of the string $(x_0 + x_1, \ldots, x_0 + x_1)$. At this moment, the decoder can just enumerate all possible guesses for the sum $x_0 + x_1$, and construct a sketch to compare against the sketch $L((x_0 + x_1, x_0 + x_1, \ldots x_0 + x_1))$. Once we learn $x_0 + x_1$, we can learn the values of x_0 and x_1 by, guessing all possible strings $(x_0, (x_0 + x_1) - x_0, 0, 0, \ldots, 0)$ and checking its sketch against the sketch $L((x_0, x_1, 0, 0, \ldots, 0))$.

In general, because of the product structure of $\bar{x}$, we will be able to do this successive guessing of certain sums of characters of x in a structured way. We will first learn sums with more terms, and then proceed to sums with fewer and fewer terms, until we decode the individual coordinates of x. In general, the number of "guesses" (sketch comparisons) will be bounded by $2^k \cdot |\Sigma| \le O(nm)$. Hence, as long as the sketch works with high probability, all the guesses will succeed to give the right answer.

PROOF OF THEOREM 11. First of all, note that we can amplify the probability of success of the sketch L to be $1 - (n|\Sigma|)^{-\Omega(1)}$. Hence we will assume that "sketch" property always succeeds. This increases the sketch size by $O(\log n|\Sigma|)$. We now show that the resulting sketch size must be $s = \Omega(D(n) \log|\Sigma|)$.

To prove this, we show how to encode an arbitrary string $x \in \Sigma^{D(n)}$ as a string $\bar{x} \in \Sigma^n$ such that, given only $L(\bar{x})$ it is possible to decode the entire string x with high probability. This implies that the sketch size of L has to be at least $\Omega(D(n) \log|\Sigma|)$.

Distinct Primes. For now, we assume $n = p_1 p_2 \cdot \ldots p_k$, i.e., is a product of k primes. We show how to embed a string $x \in \Sigma^{\{0,1\}^k}$, which has length $2^k = D(n)$. We use the fact that $\mathbb{Z}_n \cong \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2} \times \ldots \times \mathbb{Z}_{p_k}$. In other words, we can organize $\{0, 1, 2, \ldots n-1\}$ in a k-dimensional rectangle with side-lengths $p_1, p_2, \ldots p_k$, where a number $q \in [n]$ is mapped into the vector $(q \bmod p_1, \ldots q \bmod p_k)$, and this mapping is bijective. Let this rectangle be $N = [p_1] \times [p_2] \times \ldots [p_k]$. Thus, from now on we index the output of the embedding as $\bar{x}_u$, where $u \in N$. In this notation, the embedding is just $\bar{x}_u = x_u$, where u ranges over $\{0,1\}^k$, and $\bar{x}_u = 0$ otherwise.

EXAMPLE 1. *For $n = 6$ and $[6] \equiv [2] \times [3]$, the string*

$$x = \begin{pmatrix} x_{00} & x_{01} \\ x_{10} & x_{11} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

becomes

$$\bar{x} = \begin{pmatrix} \bar{x}_{00} & \bar{x}_{01} & \bar{x}_{02} \\ \bar{x}_{10} & \bar{x}_{11} & \bar{x}_{12} \end{pmatrix} = \begin{pmatrix} a & b & 0 \\ c & d & 0 \end{pmatrix}$$

or back in length-n vector notation:

$$\bar{x} = (\bar{x}_{00}, \bar{x}_{11}, \bar{x}_{02}, \bar{x}_{10}, \bar{x}_{01}, \bar{x}_{12}) = (a, d, 0, c, b, 0).$$

Now we introduce a bit more notation. For $u \in \{0, 1, \star\}^k$, let S_u be the sum of the values

$$\{\bar{x}_v | v_i = u_i \text{ for all } i \text{ with } u_i \neq \star\} \ ,$$

i.e., the set formed by "collapsing" dimension i if $u_i = \star$.

EXAMPLE 2. *Given the string,*

$$\bar{x} = (\bar{x}_{00}, \bar{x}_{11}, \bar{x}_{02}, \bar{x}_{10}, \bar{x}_{01}, \bar{x}_{12}) = (a, d, 0, c, b, 0) \ ,$$

then $S_{0,\star} = a+b$, $S_{1,\star} = c+d$, $S_{\star,0} = a+c$, and $S_{\star,1} = b+d$.

Fix some $m|n$ of the form $m = p_{i_1} p_{i_2} \ldots p_{i_t}$. We call $F = \{i_1, \ldots i_t\}$ to be the *set of free coordinates* and $C = \{1, \ldots k\} \setminus F$ to be the *set of collapsed coordinates.* Also, let

$$U_F = \{u \in \{0, 1, \star\}^k \mid u|_F \in \{0,1\}^t, u|_C = \star^{k-t}\}$$

be the set of vectors with coordinates in C collapsed.

We prove the following claim.

CLAIM 12. *Given $L(x)$ for some x, we can compute the sketch $L(y)$ of the string y, defined as: for $u \in N$, we have $y_u = S_v$ where v is such that $v|_F = u|_F$ and $v|_C = \star^{k-t}$ (and y_u is zero whenever $u|_F$ has at least a coordinate outside $\{0, 1, \star\}$). In other words, y is the string composed of sums where the coordinates $i_1, \ldots i_t$ have been collapsed. This sketch $L(y)$ will be called L_F.*

PROOF. We note that $y = \sum_{j=0}^{n/m-1} \sigma^{j \cdot m}(x)$. Hence we can compute $L(y)$ from $L(x)$ using the "linearity" and "shift" property of L. □

EXAMPLE 3. *If $n = 6$ and $m = 2$, and $\bar{x} = (a, d, 0, c, b, 0)$ then,*

$$\begin{aligned} \bar{y} &= (S_{0,\star}, S_{1,\star}, S_{0,\star}, S_{1,\star}, S_{0,\star}, S_{1,\star}) \\ &= (a+b, c+d, a+b, c+d, a+b, c+d) \ . \end{aligned}$$

The decoding algorithm will proceed in stages, computing sums S_v for v with an increasing number of free coordinates. Specifically, in the first stage we learn all sums S_u where u is all $\star$'s (in fact, one sum); in the second stage we learn all S_u which have all but one $\star$; in the third, all but two $\star$'s and so on. In general, to learn S_u for some $u \in \{0, 1, \star\}^k$, we use L_F, where F is the set of coordinates where $u_i \in \{0, 1\}$.

To learn $S_{\star^k}$, we use $L_\emptyset$ as follows. Enumerate over all possible guesses $g \in \Sigma$, and for each construct $L' = L(g^n)$. If $L' = L_\emptyset$, then $g = S_{\star^k}$ whp.

In general, we show how to learn the set of S_v where $v \in U_F$ for a fixed set of free coordinates $F \subseteq \{1, \dots k\}$ of size t. Let $C = \{1, \dots k\} \setminus F$ be the set of collapsed coordinates. We guess $g = S_{u_0}$ where $u_0|_F = 0^t$ and $u_0|_C = \star^{k-t}$. Assuming the guess is correct, (i.e., it was the case that $g = S_{u_0}$) we show how we can compute all other S_u for $u \in U_F$ using only the sums we already learned. Suppose we already know what is S_u for some other $u \in U_F$. Then, if $v \in U_F$ is such that u, v differ in exactly one coordinate $i \in F$, we can compute S_v as $S_v = S_{w(u,v)} - S_u$, where $w(u, v) \in U_F \setminus \{i\}$ is a vector which coincides with u and v on all coordinates except the coordinate i where it is $\star$. Note that such sums S_w for $w \in U_{F \setminus \{i\}}$ have already been computed in the previous stages. Thus, following, say, a Gray code, we can deduce all S_u for $u \in U_F$ from the guess $g = S_{u_0}$.

Example 4. *If $\bar{x} = (a, d, 0, c, b, 0)$ then:*

$$\begin{aligned} F = \emptyset \quad &\Rightarrow \quad \bar{y} = (a+b+c+d, \dots, a+b+c+d) \ , \\ F = \{1\} \quad &\Rightarrow \quad \bar{y} = (a+b, c+d, a+b, c+d, a+b, c+d) \ , \\ F = \{2\} \quad &\Rightarrow \quad \bar{y} = (a+c, b+d, 0, a+c, b+d, 0) \ , \\ F = \{1, 2\} \quad &\Rightarrow \quad \bar{y} = (a, d, 0, c, b, 0) \ . \end{aligned}$$

For each such guess g, we compute the string y' which satisfies: $y'_u = S_v$ where v is such that $v|_F = u|_F$ and $v|_C = \star^{k-t}$ (and zero whenever $u|_F$ has a coordinate outside $\{0, 1, \star\}$). Then we compute $L(y')$ and compare against L_F. With high probability, they are equal iff $y' = y$ from which one obtained L_F, i.e., the guess g is correct.

At the end we will have reconstructed the sums S_v for v where all coordinates are free. Specifically, for $v \in \{0, 1\}^k$, we have $S_v = x_v$. Note that, in total, there have been $2^k \cdot |\Sigma| \le n|\Sigma|$ guesses. Hence, they all succeed with high probability and we can reconstruct x. This completes the proof of the theorem when n is the product of k primes.

General Case. We now show the general case, when $n = p_1^{r_1} \cdots p_k^{r_k}$, which is just a mild generalization of the above reduction. We embed a string $x \in \Sigma^{(1+r_1)\cdot(1+r_2)\cdots(1+r_k)}$ into $\bar{x}$, indexed by the set $N = [p_1^{r_1}] \times \cdots [p_k^{r_k}]$, which is isomorphic to $\mathbb{Z}_n$. Specifically, for $u \in [1 + r_1] \times [1 + r_2] \times \cdots \times [1 + r_k]$, we construct u' as $u'_i = {p_i}^{u_i} - 1$ for all i and set $\bar{x}_{u'} = x_u$. For the remaining vectors u' we have $\bar{x}_{u'} = 0$.

The decoding algorithm is similar to above and proceeds as follows. In general, we replace sets F by vectors $\phi \in N$, with the following meaning. For a coordinate i, ϕ_i will mean i is collapsed, and $\phi_i = r_i$ means the variable is free. More generally, $0 < \phi_i < r_i$ means partial collapse, namely the coordinate values $\phi_i \dots r_i$ are collapsed together and the rest are free. We will denote such partial collapse with $\star_{\phi_i}$. Thus, we define the set of possible sum indexes as the following set of k-dimensional vectors

$$U_\phi = \{u \mid \forall i : u_i \in \{0, 1, \dots \phi_i - 1, \star_{\phi_i}\}\}.$$

Finally, for some vector $u \in U_\phi$, we have the sum S_u to be equal to the sum $S_u = \sum_{E(u)} x_{j_1,\dots,j_k}$, where the extended set $E(u)$ is composed of vectors v such that $v_i = u_i$ whenever $u_i < \phi_i$ and $v_i \in \{\phi_i \dots r_i\}$ when $u_i = \star_{\phi_i}$.

Again, we will compute the sums S_ϕ in a certain lexicographic order over ϕ's, where on each coordinate we have $\star_i$ comes before $\star_j$ iff $i < j$.[2]

Fix some ϕ and suppose we want to compute all the sums S_u for $u \in U_\phi$. Note that we can obtain a sketch $L_\phi = L(y)$ where the vector y (in tensor notation) is as follows: for all $v \in N$, we have $y_v = S_u$ where $u_i = v_i$ whenever $v_i < \phi_i$ and $u_i = \star_{\phi_i}$ whenever $v_i \ge \phi_i$. In particular, take $m = \prod_i p_i^{\phi_i}$ and then $y = \sum_{j=0}^{n/m} \sigma^{jm}(x)$.

We will again construct "guess sketches" for L_ϕ using sums already known. First note that some sums are already known: for $u \in U_F$ if some $u_i \notin \{\phi_i - 1, \star_{\phi_i}\}$, then the sum S_u is known from a previous ϕ. We show how to guess all the rest at the same time. Fix some representative $u_0 \in U_\phi$ for which we do not yet know the sum S_{u_0}. We guess $g = S_{u_0}$ and then we decide the consistent sums S_u for all $u \in U_\phi$. Consider two vectors $u, v \in U_F$, for which we still don't know S_u, S_v and which differ in one coordinate i: say, $u_i = \phi_i - 1$ and $v_i = \star_{\phi_i}$. Then $S_u + S_v = S_w$ where w is equal to u on all coordinates except i and $w_i = \star_{\phi_i - 1}$. Note that S_w has been computed in a previous stage (as it corresponds to a lexicographically smaller ϕ). Hence we can guess the entire string y and compare against L_ϕ. Once we have completed computed the sums for $\phi = (r_1, r_2, \dots r_k)$ we are done as we have reconstructed x. In total this takes

$$O(|\Sigma| \cdot \prod_{j=1}^{k} (r_i + 1)) < O(n|\Sigma|)$$

guesses. This completes the proof of the theorem.

4.2 Edit Distance Lower Bound

We now present our lower bound for edit distance, which essentially states that there is no good linear sketch for the edit distance.

Theorem 13. *Let $n \ge 1$, and let the alphabet be $\Sigma = \mathbb{Z}_m$ where $m = n^3$. Assume that, for some given approximation $\alpha \ge 2$, there is a randomized function (sketch) $L : \Sigma^n \to \{0, 1\}^s$ that satisfies the following conditions:*

1. *Sketch: For any two distinct strings $x, y \in \Sigma^n$, we distinguish the case $\mathrm{ed}(x, y) \ge 2\alpha$ from $\mathrm{ed}(x, y) \le 2$ with probability at least $2/3$;*
2. *Linearity: Given $L(x)$ and $L(y)$ for some $x, y \in \Sigma^n$, we can compute the sketch $L(x + y)$.*

Then, the sketch size s has to be at least $s = \Omega(n/\alpha)$.

Proof. The proof is a reduction from the indexing problem: Alice has a string $x_1, x_2, \dots, x_k \in \Sigma^k$, where $k = n/l$ and $l = 4\alpha$, and Bob has an index $j \in \{1, 2, \dots, k\}$ and wants to determine x_j with probability at least $2/3$. Suppose there exists a sketch L with the properties outlined in the theorem statement. As in the previous proof we can amplify the probability of success of the sketch L to be $1 - (nm)^{-\Omega(1)}$. Hence we will subsequently assume that "sketch" property always succeeds. This increases the sketch size by $O(\log n)$. We now show that the resulting sketch size must be $\Omega(n \log n)$.

Alice generates an encoded string of length n from the input string $x_1, \dots x_k$, using the following function. Let $A : [m] \to [m]^l$ be an encoding function satisfying the following property:

[2] This defines a partial order only, but any linearization of it is sufficient.

- For any distinct $i, j \in \mathbb{Z}_m$, let A' be such that $A'_1 = A(i)_1 + 1$ and $A'_p = A(i)_p - A(j)_p + A(j)_{p-1}$ for $2 \leq p \leq |A(i)|$. Then $\mathrm{ed}(A', A(i)) \geq 2\alpha$.

We note that a random function A works with non-zero probability (at least) as long as $l \geq 4\alpha$ and $m \geq n^3$, hence there exists such a function A. In particular, if $A(i)$ and $A(j)$ are random and independent, then $\mathrm{ed}(A', A(i))$ is at least the distance between two random strings of length $l-1$ over alphabet of size $n^3 > (l-1)^3$. It is now a standard fact that the edit distance in this case will be $\geq l/2 = 2\alpha$.

Given this function, Alice constructs her string

$$\bar{x} = (A(x_1), A(x_2), \dots, A(x_k)) \ .$$

Then Alice communicates $L(\bar{x})$ to Bob. Bob will use $L(\bar{x})$ to successively deduce all entries of x.

We now show how Bob decodes characters $x_k, x_{k-1}, \dots, x_1$ one by one. Suppose Bob already knows $x_{i+1}, \dots x_k$, and wants to decode x_i. Bob will guess the values of x_i and will check whether the guess is right.

Specifically, for a guess $g \in \mathbb{Z}_m$ of x_i, Bob constructs a sketch of a string y defined as follows. y is equal to the string $\bar{x}$ everywhere except in the i-th block, i.e.,

$$y = (A(x_1), \dots A(x_{i-1}), A', A(x_{i+1}), \dots A(x_k)) \ ,$$

where A' is obtained from $A(x_i)$ as follows. $A'_1 = A(x_i)_1 + 1$ and $A'_p = A(x_i)_p - A(g)_p + A(g)_{p-1}$. Note that we can obtain the sketch $L(y)$ from the sketch $L(\bar{x})$ by a linear operation:

$$\begin{aligned} L(y) = L(\bar{x}) + L(0^{(i-1)\cdot l}, 1, -A(g)_2 + A(g)_1, \dots \\ -A(g)_l + A(g)_{l-1}, 0^{(k-i)\cdot l}) \end{aligned} .$$

Now, observe that, if $g = x_i$ then

$$\mathrm{ed}(\bar{x}, y) = \mathrm{ed}(A(x_i), A') \leq 2 \ .$$

Furthermore, if $g \neq x_i$, then, using the special property of A, we also have that $\mathrm{ed}(\bar{x}, y) = \mathrm{ed}(A(x_i), A') \geq 2\alpha$. Hence, Bob can distinguish between the two cases by the "sketch" property of L.

In summary, Bob determines x via $O(mn)$ queries to the sketch. Hence, he can determine x_j as required. Now the indexing lower bound says that the length of the sketch must be at least $\Omega(n \log m) = \Omega(n \log n)$. □

We can also deduce the following corollary using the fact that there exist constant-sized (linear) sketches for vectors under the ℓ_1 distance, for say $(1+\epsilon)$-approximation [25].

Corollary 14. *Any linear embedding of edit distance into ℓ_1 must incur distortion $\Omega(n)$. Same is true also for any fixed power $\alpha < 1$ of the edit distance.*

We also note that our lower bound is close to being tight in at least one case: that of one-way communication complexity. Specifically, there exists a linear, one-way communication protocol that achieves $n^{1+o(1)}/\alpha$ dimension for any approximation $\alpha \geq 1$ [5].

5. CONCLUSIONS AND OPEN QUESTIONS

Our sketch for the shift metric could be used to compute $\mathrm{sh}(\cdot, \cdot)$ exactly if the value was small. It would also be interesting to design a sketch that returned a multiplicative approximation when the the shift metric was large. Another direction for further research is to consider translations in higher dimensions. For example, given two matrices $x, y \in \Sigma^{n \times n}$, does there exist $s_1, s_2 \in [n]$ such that for all

$$\forall i, j \in [n], \quad x_{i,j} = y_{i \bmod n, j \bmod n} \ .$$

Acknowledgements. The authors would like to thank Robi Krauthgamer, Hossein Jowhari, and Atri Rudra for early conversations about the problem, as well as the anonymous referees for their valueable comments.

6. REFERENCES

[1] P. K. Agarwal, G. Cormode, Z. Huang, J. M. Phillips, Z. Wei, and K. Yi. Mergeable summaries. In *PODS*, pages 23–34, 2012.

[2] K. J. Ahn, S. Guha, and A. McGregor. Analyzing graph structure via linear measurements. In *SODA*, pages 459–467, 2012.

[3] K. J. Ahn, S. Guha, and A. McGregor. Graph sketches: sparsification, spanners, and subgraphs. In *PODS*, pages 5–14, 2012.

[4] A. Andoni, P. Indyk, and R. Krauthgamer. Earth mover distance over high-dimensional spaces. In *SODA*, pages 343–352, 2008.

[5] A. Andoni, R. Krauthgamer, and K. Onak. Polylogarithmic approximation for edit distance and the asymmetric query complexity. In *FOCS*, 2010. A full version is available at `arxiv.org/abs/1005.4033`.

[6] Y. Bachrach and R. Herbrich. Fingerprinting ratings for collaborative filtering — theoretical and empirical analysis. In *String Processing and Information Retrieval*, pages 25–36. Springer, 2010.

[7] Y. Bachrach and E. Porat. Fast pseudo-random fingerprints. *arXiv preprint arXiv:1009.5791*, 2010.

[8] Y. Bachrach, E. Porat, and J. S. Rosenschein. Sketching techniques for collaborative filtering. In *Proceedings of the 21st international joint conference on Artificial intelligence*, pages 2016–2021. Morgan Kaufmann Publishers Inc., 2009.

[9] Z. Bar-Yossef, T. S. Jayram, R. Krauthgamer, and R. Kumar. Approximating edit distance efficiently. In *FOCS*, pages 550–559, 2004.

[10] T. Batu, F. Ergün, J. Kilian, A. Magen, S. Raskhodnikova, R. Rubinfeld, and R. Sami. A sublinear algorithm for weakly approximating edit distance. In *STOC*, pages 316–324, 2003.

[11] T. Batu, F. Ergün, and C. Sahinalp. Oblivious string embeddings and edit distance approximations. In *SODA*, pages 792–801, 2006.

[12] E. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52:489 – 509, 2006.

[13] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Proc. of ICALP*, 2002.

[14] R. Clifford, K. Efremenko, E. Porat, and A. Rothschild. k-mismatch with don't cares. In *ESA*, pages 151–162, 2007.

[15] R. Clifford, K. Efremenko, E. Porat, and A. Rothschild. From coding theory to efficient pattern matching. In *SODA*, pages 778–784, 2009.

[16] G. Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms*, 55(1):58–75, 2005.

[17] G. Cormode, M. Paterson, S. C. Sahinalp, and U. Vishkin. Communication complexity of document exchange. In *SODA*, pages 197–206, 2000.

[18] M. S. Crouch and A. McGregor. Periodicity and cyclic shifts via linear sketches. In *APPROX-RANDOM*, pages 158–170, 2011.

[19] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[20] G. Feigenblat, E. Porat, and A. Shiftan. Exponential time improvement for min-wise based algorithms. *Information and Computation*, 209(4):737–747, 2011.

[21] A. C. Gilbert, Y. Li, E. Porat, and M. J. Strauss. Approximate sparse recovery: optimizing time and measurements. In *STOC*, pages 475–484, 2010.

[22] D. M. Gordon. Discrete logarithms in $GF(p)$ using the number field sieve. *SIAM J. Discrete Math.*, 1(6):124–138, 1993.

[23] G. Hardy and E. Wright. *An Introduction to the Theory of Numbers*. Oxford Science Publications. Oxford University Press, USA, 1980.

[24] P. Indyk. Algorithmic aspects of geometric embeddings (tutorial). In *FOCS*, pages 10–33, 2001.

[25] P. Indyk. Stable distributions, pseudorandom generators, embeddings and data stream computation. *J. ACM*, 53(3):307–323, 2006.

[26] P. Indyk, A. McGregor, I. Newman, and K. Onak, editors. *Open Problems in Data Streams, Property Testing, and Related Topics*, 2011. Available at: `people.cs.umass.edu/∼mcgregor/papers/11-openproblems.pdf`.

[27] H. Jowhari. Efficient communication protocols for deciding edit distance. In *ESA*, pages 648–658, 2012.

[28] H. Jowhari, M. Saglam, and G. Tardos. Tight bounds for l_p samplers, finding duplicates in streams, and related problems. In *PODS*, pages 49–58, 2011.

[29] D. M. Kane, J. Nelson, E. Porat, and D. P. Woodruff. Fast moment estimation in data streams in optimal space. In *STOC*, pages 745–754, 2011.

[30] D. M. Kane, J. Nelson, and D. P. Woodruff. An optimal algorithm for the distinct elements problem. In *PODS*, pages 41–52, 2010.

[31] R. M. Karp and M. O. Rabin. Efficient randomized pattern-matching algorithms. *IBM Journal of Research and Development*, 31(2):249–260, 1987.

[32] S. Khot and A. Naor. Nonembeddability theorems via fourier analysis. *Mathematische Annalen*, 334:821–852, 2006.

[33] R. Krauthgamer and Y. Rabani. Improved lower bounds for embeddings into L_1. In *SODA*, pages 1010–1017, 2006.

[34] P. Li and A. Christian König. Theory and applications of b-bit minwise hashing. *Communications of the ACM-Association for Computing Machinery-CACM*, 54(8):101, 2011.

[35] J. H. v. Lint. *Introduction to Coding Theory*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1982.

[36] A. Naor and J. Matoušek. Open problems on embeddings of finite metric spaces. 2011.

[37] H. Q. Ngo, E. Porat, C. Ré, and A. Rudra. Worst-case optimal join algorithms: [extended abstract]. In *PODS*, pages 37–48, 2012.

[38] H. Q. Ngo, E. Porat, and A. Rudra. Efficiently decodable error-correcting list disjunct matrices and applications - (extended abstract). In *ICALP (1)*, pages 557–568, 2011.

[39] H. Q. Ngo, E. Porat, and A. Rudra. Efficiently decodable compressed sensing by list-recoverable codes and recursion. In *STACS*, pages 230–241, 2012.

[40] R. Ostrovsky and Y. Rabani. Low distortion embedding for edit distance. *J. ACM*, 54(5), 2007. Preliminary version appeared in STOC'05.

[41] B. Porat and E. Porat. Exact and approximate pattern matching in the streaming model. In *FOCS*, pages 315–323, 2009.

[42] E. Porat and O. Lipsky. Improved sketching of hamming distance with error correcting. In *CPM*, pages 173–182, 2007.

[43] E. Porat and A. Rothschild. Explicit non-adaptive combinatorial group testing schemes. In *ICALP (1)*, pages 748–759, 2008.

[44] E. Porat and M. J. Strauss. Sublinear time, measurement-optimal, sparse recovery for all. In *SODA*, pages 1215–1227, 2012.

[45] E. Porat and E. Waisbard. Efficient signature scheme for network coding. In *ISIT*, pages 1987–1991, 2012.

[46] R. Rivest. The md5 message-digest algorithm, 1992.

[47] J. Sylvester. A method of determining by mere inspection the derivatives from two equations of any degree. *Phil. Mag.*, 16:132–135, 1840.