Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Near-Optimal Average-Case Approximate Trace Reconstruction from Few Traces

Xi Chen* Anindya De† Chin Ho Lee‡ Rocco A. Servedio§
Sandip Sinha¶

Abstract

In the standard trace reconstruction problem, the goal is to *exactly* reconstruct an unknown source string $\mathsf{x} \in \{0,1\}^n$ from independent "traces", which are copies of x that have been corrupted by a δ-deletion channel which independently deletes each bit of x with probability δ and concatenates the surviving bits. We study the *approximate* trace reconstruction problem, in which the goal is only to obtain a high-accuracy approximation of x rather than an exact reconstruction.

We give an efficient algorithm, and a near-matching lower bound, for approximate reconstruction of a random source string $\mathsf{x} \in \{0,1\}^n$ from few traces. Our main algorithmic result is a polynomial-time algorithm with the following property: for any deletion rate $0 < \delta < 1$ (which may depend on n), for almost every source string $\mathsf{x} \in \{0,1\}^n$, given any number $M \leq \Theta(1/\delta)$ of traces from $\mathrm{Del}_\delta(\mathsf{x})$, the algorithm constructs a hypothesis string $\widehat{\mathsf{x}}$ that has edit distance at most $n \cdot (\delta M)^{\Omega(M)}$ from x. We also prove a near-matching information-theoretic lower bound showing that given $M \leq \Theta(1/\delta)$ traces from $\mathrm{Del}_\delta(\mathsf{x})$ for a random n-bit string x, the smallest possible expected edit distance that any algorithm can achieve, regardless of its running time, is $n \cdot (\delta M)^{O(M)}$.

1 Introduction

1.1 Background and prior work In the *trace reconstruction problem* [18, 21, 20, 3], there is an unknown n-bit source string $\mathsf{x} \in \{0,1\}^n$, and a reconstruction algorithm that has access to independent *traces* of x, where a trace of x is a draw from $\mathrm{Del}_\delta(\mathsf{x})$. Here $\mathrm{Del}_\delta(\cdot)$ is the *deletion channel*, which independently deletes each bit of x with probability δ and outputs the concatenation of the surviving bits. The goal of the reconstruction algorithm is to correctly reconstruct the source string x.

*Columbia University. Supported by NSF grants CCF-1703925, IIS-1838154 and CCF-2106429. Email: xichen@cs.columbia.edu

†University of Pennsylvania. Supported by NSF grants CCF-2045128, CCF-1926872 and CCF-1910534. Email: anindyad@cis.upenn.edu

‡Harvard University. Supported by the Croucher Foundation, the Simons Collaboration on Algorithms and Geometry, NSF grant CCF-1763299 and a Simons Investigator grant to S. Vadhan. Work done while at Columbia University. Email: chlee@seas.harvard.edu

§Columbia University. Supported by NSF grants CCF-1814873, IIS-1838154, CCF-1563155, CCF-2106429, and by the Simons Collaboration on Algorithms and Geometry. Email: rocco@cs.columbia.edu

¶Columbia University. Supported by NSF grants CCF-1714818, CCF-1822809, IIS-1838154, CCF-1617955, CCF-1740833, and by the Simons Collaboration on Algorithms and Geometry. Email: sandip@cs.columbia.edu

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

1.1.1 Exact trace reconstruction Much research effort has been dedicated to different aspects of the trace reconstruction problem in recent years [23, 12, 26, 27, 15, 14, 1, 2, 24, 5, 19, 16, 6, 25, 9, 8]. In the "worst-case" version of trace reconstruction, the source string x may be an arbitrary n-bit string. This is a challenging problem, with the best known information theoretic lower bound on the number of traces required for trace reconstruction (for constant deletion rates δ) being $\tilde{\Omega}(n^{3/2})$ traces [5] and the best known information theoretic upper bound being $\exp\left(\tilde{O}(n^{1/5})\right)$ traces [6] (improving on earlier $\exp\left(\tilde{O}(n^{1/2})\right)$ and $\exp\left(\tilde{O}(n^{1/3})\right)$-time and sample algorithms due to [17] and [26, 12] respectively). In the subconstant deletion rate regime, a poly(n)-time and sample algorithm for worst-case source strings was recently given in [8] for deletion rate $\delta = O(1/n^{1/3+\varepsilon})$, improving on an earlier result for deletion rate $\delta = O(1/n^{1/2+\varepsilon})$ [3]. Turning to the "average-case" variant of trace reconstruction, the goal is to give algorithms (and lower bounds) that hold for most possible source strings x (equivalently, hold with high probability for a uniform random source string $\mathbf{x} \in \{0,1\}^n$). For the average-case problem, at constant deletion rate δ the current best known lower bound is $\tilde{\Omega}((\log n)^{5/2})$ traces [5] and the best known upper bound is $\exp\big(O(\log n)^{1/3}\big)$ traces [15, 16]. Average-case trace reconstruction has been shown to be essentially equivalent to coded trace reconstruction; see [10, 4]. In [3] an $O(\log n)$-trace, poly(n)-time algorithm is given for the average case problem when the deletion rate is $\delta = O(1/\log n)$.

1.1.2 Approximate trace reconstruction Motivation. In this paper we study a relaxation of the exact trace reconstruction problem in which the goal is only to obtain an *approximation* of the unknown source string x. Of course this immediately raises the question of what distance measure to use; throughout this paper we use edit distance as our distance measure between strings. We remark that edit distance is a natural metric to consider in the context of trace reconstruction: in particular, trace reconstruction is motivated by problems such as ancestral DNA reconstruction where the natural corruption process includes synchronization errors such as insertion and deletion. Indeed, edit distance is the distance measure used in all of the works discussed below under "Prior work."

The study of approximate trace reconstruction has several natural motivations; first, in some applications a high-accuracy reconstruction of x may be all that is required rather than exact reconstruction. Second, it is of interest to obtain algorithmic results for trace reconstruction in settings where insufficiently many traces are available for exact reconstruction (because of known lower bounds mentioned above); approximate trace reconstruction offers a potential avenue for obtaining rigorous results in such settings. Finally, as sketched above, there is a frustrating exponential gap between the known upper and lower bounds for exact trace reconstruction in both the worst-case and average-case problem variants. Hence it is natural to wonder whether sharper bounds can be achieved for approximate versions of the problem.

Prior work. Several authors have quite recently considered the approximate trace reconstruction problem and related questions.

Davies, Raćz, Rashtchian, and Schiffer [11] gave several algorithms that use polylog(n) traces and achieve edit distance $n/\text{polylog}(n)$ for certain classes of source strings defined by various run-length assumptions. They also give other algorithms which, under stronger run-length assumptions, succeed in performing approximate reconstruction using only a single trace. In another recent work, Srinivasavaradhan, Du, Diggavi, and Fragouli [29] proposed heuristics for approximate reconstruction based on a few traces.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Sima and Bruck [28] have recently studied exact trace reconstruction under an edit distance constraint. They showed that $n^{O(k)}$ traces suffice to distinguish between two (known) worst-case n-bit strings that are promised to have edit distance at most k from each other. In a related but incomparable result, Grigorescu, Sudan, and Zhu [13] have given lower bounds on "mean-based" algorithms for distinguishing between worst-case pairs of strings that have small edit distance.

Summarizing the prior results on approximate trace reconstruction, we are not aware of either algorithms or lower bounds in the previous literature that apply to typical source strings $\mathbf{x} \sim \{0,1\}^n$ (though see below for a discussion of the recent work of [7] that is simultaneous to ours). It is easy to see that simply outputting a single trace gives expected edit distance δn (for any source string), and also that given M traces no algorithm can achieve expected edit distance better than $\Theta(\delta^M n)$ for random source strings (since in expectation $\delta^M n$ bits of the n-bit source string will have been deleted from all M traces). Other than these simple observations, to the best of our knowledge no prior results were known, either in terms of algorithms or lower bounds, for approximate trace reconstruction of random strings. We describe our algorithms and lower bounds for this setting below.

1.2 Our results Matching upper and lower bounds on approximate reconstruction of random strings from few traces.

Our main contribution is the following algorithmic result:

THEOREM 1.1. (APPROXIMATE AVERAGE-CASE TRACE RECONSTRUCTION ALGORITHM) *There is a* $\mathrm{poly}(n)$ *time algorithm* `Reconstruct` *with the following property: Let* $0 < \delta < 1$*, and let* $\mathbf{x}$ *be an unknown source string that is uniform random over* $\{0,1\}^n$*. Let* $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ *be* $M \leq \Theta(1/\delta)$ *independent traces drawn from* $\mathrm{Del}_\delta(\mathbf{x})$*. Then with probability at least* $1 - 1/\mathrm{poly}(n)$ *over* $\mathbf{x} \sim \{0,1\}^n$ *and* $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)} \sim \mathrm{Del}_\delta(\mathbf{x})$*, the output of* `Reconstruct` *on input* δ *and* $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ *is a string* $\widehat{\mathbf{x}} \in \{0,1\}^*$ *that has* $d_{\mathrm{edit}}(\mathbf{x}, \widehat{\mathbf{x}}) \leq n \cdot (\delta M)^{\Omega(M)}$.

An interesting special case of Theorem 1.1 is obtained when the number of available traces M is $\Theta(1/\delta)$. In this case the `Reconstruct` algorithm achieves edit distance $n/2^{\Omega(1/\delta)}$, which is exponentially better than the benchmark of δn edit distance that is trivially achievable using a single trace.

To complement Theorem 1.1, we prove an information-theoretic lower bound on approximate trace reconstruction of random strings from $M \leq \Theta(1/\delta)$ traces. This lower bound shows that the accuracy achieved by `Reconstruct` is essentially the best possible:

THEOREM 1.2. (LOWER BOUND ON APPROXIMATE AVERAGE-CASE TRACE RECONSTRUCTION) *Let* $0 < \delta < 1$*, and let* $\mathbf{x}$ *be an unknown source string that is uniform random over* $\{0,1\}^n$*. Let* $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ *be* M *independent traces drawn from* $\mathrm{Del}_\delta(\mathbf{x})$*, where* $M \leq \Theta(1/\delta)$*. Let* $\mathtt{A}$ *be any algorithm which, given* δ *and* $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ *as input, outputs a hypothesis string* $\widehat{\mathbf{x}}$ *for* $\mathbf{x}$*. Then the expected edit distance between* $\widehat{\mathbf{x}}$ *and* $\mathbf{x}$ *is at least* $n \cdot (\delta M)^{O(M)}$.

We observe that for natural parameter settings, Theorem 1.2 improves on the simple $\Omega(\delta^M n)$ expected edit distance lower bound mentioned earlier; for example, taking $M = \Theta(1/\delta)$, Theorem 1.2 proves that the best possible accuracy is $n/2^{O(M)}$ rather than $\delta^M n$.

REMARK 1.1. In simultaneous and independent work to ours, Chase and Peres [7] have also considered the problem of approximate trace reconstruction of random source strings $\mathbf{x}$. Their main result is that for any constant deletion rate δ (bounded away from 1) and any constant ε (bounded away from 0), there is an algorithm that uses $O_{\delta,\varepsilon}(1)$ traces and, with high probability

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

over a random source string $\mathbf{x} \sim \{0,1\}^n$, succeeds in reconstructing a hypothesis string $\widehat{x}$ with edit distance at most εn from $\mathbf{x}$.

The work of [7] and the current paper focus on different parameter settings, in particular different regimes for the number of traces available to the algorithm, and establish complementary results. The results of [7] apply in the regime where "many traces" (significantly more than $\Theta(1/\delta)$) are available, and give high-accuracy reconstruction in this regime. In contrast, our results apply in the "few traces" regime where only some number $1 \leq M \leq \Theta(1/\delta)$ of traces are available, and give essentially optimal reconstruction for any such small number of traces.

1.3 Discussion and future work A number of directions suggest themselves for future work on approximate trace reconstruction; we close this introduction by briefly mentioning a few of these.

One natural goal is to obtain results for average-case approximate trace reconstruction which generalize both the results of the current paper and the results of [7], by establishing sharp bounds on approximate average-case trace reconstruction in the regime where more than $\Theta(1/\delta)$ many traces are available. It is clear that the $n \cdot (\delta M)^{\Theta(M)}$ form of our edit distance bound no longer holds once M is $\omega(1/\delta)$; it would be interesting to understand the best achievable edit distance, as a function of δ and M, in this regime.

Another natural goal is to obtain algorithmic results for approximate trace reconstruction of *worst-case* rather than random strings. Here we observe that the current state of the art for worst-case exact trace reconstruction places significant limitations on how much better than edit distance δn (trivially achievable by simply outputting a random trace) it is possible to do for approximate reconstruction of worst-case strings. As noted earlier, until quite recently the best result known for the low deletion rate regime was that of [3], which gave an algorithm using $O(n \log n)$ traces to reconstruct an arbitrary source string x at deletion rate $\delta = n^{-(1/2+\varepsilon)}$. This was recently strengthened to a poly(n)-trace algorithm that reconstructs at rate $\delta = n^{-(1/3+\varepsilon)}$ [8]. For the worst case approximate trace reconstruction problem, achieving edit distance $\delta^3 n$ for all δ, even using poly(n) traces, would require improving the recently established state of the art from [8] for the low-deletion-rate regime of the exact reconstruction problem.

2 Our approach

2.1 Overview of our algorithmic approach (Theorem 1.1)

2.1.1 Some preliminary observations and simplifications We begin by observing that to prove Theorem 1.1 it suffices to prove it under the assumptions that

$$\frac{1}{n^2} \leq \delta < \frac{1}{KM}, \qquad K^2 \leq M \leq \frac{1}{K\delta}, \qquad \text{and} \qquad (\delta M)^{M/K} \geq 1/n^2, \tag{2.1}$$

for a sufficiently large absolute constant K. The upper bounds on δ and M follow directly from our assumption $M \leq \Theta(1/\delta)$ in Theorem 1.1. For the lower bound on δ, note that if $\delta < 1/n^2$, then with probability at least $1 - 1/n$ a single input trace will have no bits deleted and hence will trivially yield a string $\widehat{\mathsf{x}}$ that has $d_{\text{edit}}(\mathbf{x}, \widehat{\mathsf{x}}) = 0$.

For the lower bound on M, we observe that if $M < K^2$, then a single trace would satisfy the claimed edit distance bound in Theorem 1.1. Indeed, since a single trace has edit distance from $\mathbf{x}$ distributed as $\text{Bin}(n, \delta)$, and the probability that a draw from $\text{Bin}(n, \delta)$ exceeds $n \cdot \delta^{0.1}$ is at most $n^{-\Omega(1)}$ (by a standard multiplicative Chernoff bound, using that $1/n^2 \leq \delta \leq 1/K$), the trivial

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

algorithm that simply outputs a single input trace would satisfy edit distance

$$n\delta^{0.1} \leq n(\delta M)^{0.1} \leq n(\delta M)^{0.1M/K^2} = n(\delta M)^{\Omega(M)}.$$

For our final simplifying observation that M and δ jointly satisfy $(\delta M)^{M/K} \geq 1/n^2$, note that if $(\delta M)^{M/K}$ is less than $1/n^2$, then the claimed high-probability edit distance bound $n \cdot (\delta M)^{\Omega(M)}$ of Theorem 1.1 is less than 1 (for a suitable choice of the hidden constants), and hence the claim of Theorem 1.1 is that with high probability the edit distance achieved is zero. In this case since $(\delta M)^{M/K}$ is decreasing for $M \in [0, \delta/e]$, we can simply use $M' < M$ traces so that $1/n^2 \leq (\delta M')^{M'/K} < 1/n$, and achieve edit distance $n \cdot (\delta M')^{M'/K}$ which will also achieve edit distance 0 (which of course suffices to achieve the edit distance required by the theorem statement). Therefore we will assume that the conditions given in (2.1) hold throughout the rest of our proof of Theorem 1.1.

2.1.2 The high-level approach Our main algorithm `Reconstruct` makes essential use of one particular distinguished trace, which we denote $\mathbf{y}^*$ and refer to as the *reference trace*, as well as M other traces $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$. The overall `Reconstruct` algorithm works by repeatedly executing two different subroutines. Below we first give a high level description of what each of these subroutines does and then we present the overall algorithm and explain how it uses these subroutines.

First subroutine: Alignment. The first subroutine is an alignment procedure which we call `Align`. It takes as input the reference trace $\mathbf{y}^*$ and a pointer ℓ^* to a location in the reference trace, as well as the M other traces $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$. It outputs a list of M pointers $(\ell^{(1)}, \ldots, \ell^{(M)})$ where each pointer $\ell^{(m)}$ specifies a location in the m-th trace $\mathbf{y}^{(m)}$. Roughly speaking, `Align` uses the reference trace to "align" the other M traces, i.e. to come up with a pointer into each trace so that most of the pointers agree (`Align` does not change the location ℓ^* of the pointer into the reference trace). In more detail but still at a high level, the main guarantee of the `Align` algorithm is that with high probability, a clear majority of the M pointers all point to locations that came from the same bit $\mathbf{x}_i$ of the source string $\mathbf{x}$. (Another important guarantee is that with high probability this location $i \in [n]$ is "not too far" from the location in $\mathbf{x}$ that $\mathbf{y}^*_{\ell^*}$ came from; we give more details on this below.) Thus a successful run of `Align` results in a clear majority of the $M+1$ pointers (including the reference trace's pointer ℓ^*) all being in agreement. We refer to a specification of the pointer locations $(\ell^{(1)}, \ldots, \ell^{(M)})$ as a *configuration*, and we say that a configuration for which there is a clear majority in agreement as described above is *in consensus*. (We give a fully detailed definition in Section 2.2, along with a detailed statement of the `Align` algorithm's performance guarantee.) We emphasize that the correctness of this subroutine, i.e., Alignment, crucially relies on the source string $\boldsymbol{x}$ being uniform random.

Second subroutine: Bitwise Majority. The second subroutine is a "Bitwise Majority Alignment" procedure, which we call `BMA`. This procedure was first introduced in the work of [3] and was further analyzed in the recent work [8]. (As we explain below, a crucial ingredient in our proof of Theorem 1.1 is a new refined analysis of `BMA` that goes significantly beyond the results of [8].) All of the output bits that our algorithm constructs are produced by `BMA`. The `BMA` procedure takes as input the $M+1$ traces $\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ and the corresponding pointers $\ell^*, \ell^{(1)}, \ldots, \ell^{(M)}$. The `BMA` algorithm is run for $R := (\delta M)^{-\Theta(M)}$ many stages to reconstruct R output bits; in the course of its execution it updates the pointers into all $M+1$ of the traces $\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$.

To explain the performance guarantee of the `BMA` procedure we need the notion of a *k-desert*. Roughly speaking, a binary string $z \in \{0,1\}^*$ is said to be a *k-desert* if (i) it is sufficiently long, and (ii) it is a prefix of s^∞ for some $s \in \{0,1\}^{\leq k}$ (we give a precise definition in Section 2.3). The main

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Algorithm 1: `Reconstruct` (slightly simplified)

Input: A positive integer n and $(M+1)$ traces $\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$ for some $M \leq 1/(K\delta)$

Output: A binary string w

Set $\ell^* = 1$ and $\mathsf{w} = \varepsilon$ (the empty string)

while $\ell^* \leq |\mathsf{y}^*|$ **do**

  Run `Align` $(\ell^*, \mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)})$ to obtain a tuple of locations $(\ell^{(1)}, \ldots, \ell^{(M)})$

  Run `BMA` $(\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}; \ell^*, \ell^{(1)}, \ldots, \ell^{(m)})$ to obtain a binary string (ε or in $\{0,1\}^R$)

  Concatenate the string returned by `BMA` to the end of w

  Set ℓ^* to be the final pointer of y^* in the run of `BMA` above and increment it

return w.

Figure 1: A slightly simplified version of our main algorithm `Reconstruct` (the actual algorithm differs in some small details and is given in Figure 6).

guarantee of the `BMA` procedure is that if it is run on a configuration that is in consensus at some location i in the source string x, then with high probability it produces a R-bit string that agrees with $(x_i, x_{i+1}, \ldots)$ up to the location (if any) where a k-desert of length $L := \Theta(M \log(1/(M\delta)))$ first appears, for some $k \leq L/2$. We note that unlike the first subroutine Alignment, the guarantee of `BMA` procedure is a worst-case guarantee.

The overall `Reconstruct` algorithm. As stated earlier, the overall algorithm repeatedly runs `Align`, then `BMA`, then `Align`, then `BMA`, and so on. We present a slightly simplified version of the algorithm in Figure 1 (see Section 6 for the formal algorithm; the version in Figure 1 differs only in that some parameter settings have been slightly simplified for the sake of readability).

The high level intuition for why the algorithm succeeds is as follows. Each run of `Align` with high probability succeeds in putting the M traces in consensus at a location "not too far" from the location in $\mathbf{x}$ corresponding to $\mathbf{y}^*_{\ell^*}$. Given that this consensus has been achieved by `Align`, then the subsequent run of `BMA` with high probability succeeds in correctly reconstructing the next $R := (\delta M)^{-\Theta(M)}$ many bits of $\mathbf{x}$. In the course of running `BMA` the pointer ℓ^* is with high probability advanced to "approximately the right location" corresponding to the last-reconstructed bit of $\mathbf{x}$, so the next run of `Align` again establishes consensus at approximately the right location. Thus the overall output string w of the algorithm is the concatenation of many length-R strings, most of which correspond to subwords of $\mathbf{x}$ from approximately the right locations. From this it can be shown that the overall reconstructed string is not too far in edit distance from $\mathbf{x}$.

The above high-level explanation sketches an idealized version of the actual scenario and glosses over a number of technical difficulties. In more detail, there are many sources of error from different possible failure events and a careful analysis is required (and is provided in Section 6) to keep the failure probabilities from all of these under control and not "give away too much" in the overall edit distance. The issues that must be handled include the following:

- `Align` may fail to align the traces to a consensus location, or may misalign the traces and achieve consensus at a location that is far away from the location in $\mathbf{x}$ corresponding to $\mathbf{y}^*_{\ell^*}$. Our analysis shows that this happens with small (but non-negligible probability), and bounds the cumulative error (edit distance) incurred by the runs of `Align` for which this

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

happens.

- Even when **Align** aligns the traces at a location that is "not too far" from the correct location, the alignment location in $\mathbf{x}$ may not be exactly the location in $\mathbf{x}$ corresponding to $\mathbf{y}^*_{\ell^*}$. This contributes to the overall edit distance between $\widehat{x}$ and $\mathbf{x}$ even when the **Align** algorithm succeeds.
- On average the random string $\mathbf{x}$ will have a k-desert of length L occuring roughly once every $2^{\Theta(L)}$ positions. When a run of **BMA** encounters such a location the resulting R-bit string that it produces may be badly off from the true corresponding portion of $\mathbf{x}$. This contributes to the overall edit distance between $\widehat{x}$ and $\mathbf{x}$ even when the algorithm succeeds.
- Even when the portion of $\mathbf{x}$ that a given run of **BMA** is operating on does not contain a k-desert of length L, the **BMA** algorithm may fail to correctly reconstruct the relevant portion of **BMA** with small (but non-negligible) probability. Our analysis bounds the overall error in the reconstructed string that comes from such "failed runs" of **BMA**.

2.2 The Align procedure As stated earlier, **Align** takes as input the reference trace $\mathbf{y}^*$, a pointer ℓ^* to a location in the reference trace, and the M other traces $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$, and outputs a list of M pointers $\ell^{(1)}, \ldots, \ell^{(M)}$ into the M traces $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$. In a successful run of **Align**, it generates a list of pointers most of which point to locations that came from the same bit $\mathbf{x}_i$ of the source string $\mathbf{x}$.

At a very high level, **Align** works in two stages. The first stage, which performs an "approximate alignment," consists of a sequence of iterative refinement steps; in each successive step of this stage, for each trace $\mathbf{y}^{(m)}$ **Align** tries to identify successively smaller subwords of $\mathbf{y}^{(m)}$ that fairly closely match (as measured by edit distance) suitable successively smaller subwords, centered at ℓ^*, of the reference trace $\mathbf{y}^*$. At the end of a successful execution of the first stage, for each trace $\mathbf{y}^{(m)}$ a relatively small subword $\mathbf{y}^{(m)}_{Q_1^m}$ has been identified which contains the "right location" in $\mathbf{y}^{(m)}$ (informally, corresponding to the portion of $\mathbf{x}$ that $\mathbf{y}^*_{\ell^*}$ came from). In the second stage, **Align** searches for a suitable subword that appears in at least 95% of $\mathbf{y}^{(1)}_{Q_1^1}, \ldots, \mathbf{y}^{(M)}_{Q_1^M}$ and uses the location of this subword in each $\mathbf{y}^{(m)}$ to determine the exact final pointer location $\ell^{(m)}$.

Correctness of the second stage (given that a successful "approximate alignment" was indeed achieved in the first stage) is established using an elementary but careful analysis that we do not describe here but is given in Section 4.4.2. To gain intuition for the iterative approach employed in the first stage, it is useful to consider the following toy scenario: Fix an a-bit subword w of the reference trace $\mathbf{y}^*$ that is centered at location ℓ^*. Intuitively, the deletion rate δ is relatively low, so the subword w of $\mathbf{y}^*$ should have small edit distance from the corresponding subword of the source string $\mathbf{x}$, and, transitively, should also have small edit distance from the corresponding subwords of each of the M traces $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$. However, since $\mathbf{x}$ is uniform random (and hence each trace $\mathbf{y}^{(m)}$ is also uniform random), if a is a "small" value that is $\ll \log n$, then it is very likely that w will occur as an ℓ-bit subword of each $\mathbf{y}^{(m)}$ in many locations, and thus a simpleminded approach of just scanning all of $\mathbf{y}^{(m)}$ to try to find w (or a close match to it) will not succeed in uniquely identifying the correct location. But if a is a "large" value (actually, being just modestly larger than $\log n$ will do), then it is very likely that only one location in each $\mathbf{y}^{(m)}$ will be a close match to the a-bit string w. This reduces the problem of finding the right location in the $\approx n$-bit string $\mathbf{y}^{(m)}$ to the problem of finding the right location in the $\approx a$-bit subword of $\mathbf{y}^{(m)}$ that was just identified (by virtue of closely matching w); and now we can iterate.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

A more complete overview and explanation of Align is given in Section 4.1. Section 4 gives a detailed proof of Theorem 4.1, which is our main result about Align; since the exact theorem statement is somewhat cumbersome (involving various specific parameter settings), we give an informal version here and defer the fully detailed statement to Section 4. Informally, we say that the Align algorithm *succeeds* on source string x with respect to a tuple of traces $(\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)})$ if the following condition holds for "almost all" locations $\ell^* \in [|\mathsf{y}^*|]$: The output $(\ell^{(1)}, \ldots, \ell^{(M)})$ of $\texttt{Align}(\ell^*, \mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)})$ satisfies (1) At least 90% of $\text{source}^{(m)}(\ell^{(m)})$, $m \in [M]$, agree on the same location $i \in [n]$, and (2) The consensus location i is "quite close" to the location in $\mathbf{x}$ that $\mathbf{y}^*_{\ell^*}$ came from. Now we can state an informal version of Theorem 4.1, which gives a performance guarantee on Align:

THEOREM 2.1. (MAIN RESULT ABOUT Align, INFORMAL STATEMENT) *Let* $\mathbf{x} \sim \{0,1\}^n$ *and let* $\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ *be independent traces drawn from* $\text{Del}_\delta(\mathbf{x})$*, where* δ *and* M *satisfy Equation (2.1). Then* Align *succeeds on* $\mathbf{x}$ *with respect to* $(\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)})$ *with probability at least* $1 - 1/\text{poly}(n)$.

2.3 The BMA procedure The Bitwise Majority Alignment, or BMA, procedure, operates in discrete time steps on a collection of M independent traces. At each time step it outputs one bit of the hypothesis string that it is reconstructing. Throughout its execution, at each time step t, for each $m \in [M]$ the BMA algorithm maintains a pointer into the m-th trace. The idea of BMA is that at each time step t, it should be the case that most of the pointers are correctly aligned, i.e. the majority of the bits that they point to in their respective traces came from the same bit of the source string x. In the t-th time step the majority vote of the M bits that are pointed to in the traces is the output bit BMA produces, and

- For each trace in which the pointer points to a bit that agrees with the majority, the pointer is incremented by one location;
- For each trace in which the pointer points to a bit that disagrees with the majority, the pointer stays in the same location.

A first analysis of BMA, for deletion rate δ slightly less than $n^{-1/2}$, was originally given in [3], and more recently an analysis for deletion rate δ slightly less than $n^{-1/3}$ was given in [8]. We give a significant extension of [8] by providing a much more refined analysis which yields a considerably stronger quantitative result.[1] In more detail, in the current work our analysis of BMA handles deletion rates even as large as a (small) absolute constant independent of n, and indeed handling such deletion rates is essential for our overall results.

To state our main theorem about BMA we require the following terminology: Recall that a string is said to be a *k-desert* for some $k \geq 1$ if it is the prefix of s^∞[2] for some string $\mathsf{s} \in \{0,1\}^k$. We say a string is a *long desert* if it is a k-desert of length L for some $k \leq L/2$.

Our main result about BMA says, roughly speaking, that if the source string does not contain any long desert then with high probability BMA succeeds in exactly reconstructing the source string, and moreover does so with a "clear majority" in each round. Similar to Align, the detailed theorem statement about BMA involves various specific parameter settings, so we defer its exact statement until later (see Theorem 5.1) and here give an informal statement:

[1] This quantitative strengthening plays an essential role in our being able to obtain tight bounds (recall the essentially matching Theorems 1.1 and 1.2) via our approach.

[2] For a string $\mathbf{s}$, we use $\mathbf{s}^\infty$ to denote the string consisting of infinitely many repetitions of $\mathbf{s}$.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

THEOREM 2.2. (MAIN RESULT ABOUT BMA, INFORMAL STATEMENT) *Let* $\tilde{\mathsf{x}} \in \{0,1\}^R$ *be a string that does not contain any long desert. Let* $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(M)} \sim \mathrm{Del}_\delta(\tilde{\mathsf{x}})$ *be independent traces. For suitable settings of* R, M, k, L *and* δ*, with high probability* BMA *returns exactly* $\tilde{\mathsf{x}}$*, and in every round* $t \in [R]$ *the majority is reached by at least* 90% *of* $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(M)}$*.*

2.4 Overview of our lower bound approach (Theorem 1.2) Our lower bound approach is informed by insights arising from the analysis of our algorithm. Given the arguments sketched above for our algorithmic results, it is natural to pursue a lower bound based on the difficulty of reconstructing deserts. The high-level idea of our lower bound is that having access to only a limited number $M \leq \Theta(1/\delta)$ of traces imposes strong limitations on the ability of any reconstruction algorithm to accurately estimate the lengths of deserts, and this inability to accurately reconstruct deserts translates into an inability to perform overall high-accuracy approximate reconstruction. Guided by this general idea, it is natural to consider 1-deserts (runs of all 0's or all 1's) as potential sources of hardness, and indeed this is our approach.

In more detail, our lower bound proceeds in four conceptual stages.

1. We first (Section 7.1) consider the following simple distribution distinguishing problem: an algorithm is given M draws which are guaranteed to come from one of two product distributions over $\mathbb{N} \times \mathbb{N}$: (a) the product distribution $\mathrm{Bin}(M, 1-\delta) \times \mathrm{Bin}(M+1, 1-\delta)$, or (b) the product distribution $\mathrm{Bin}(M+1, 1-\delta) \times \mathrm{Bin}(M, 1-\delta)$, where both (a) and (b) are equally likely to be the target product distribution. We show that any algorithm for determining whether it is (a) or (b) must have failure probability at least $(\delta M)^{O(M)}$.

2. Next, in Section 7.2 we consider the algorithmic task of solving B independent instances of the distinguishing problem described in (1) above; this may be viewed as the problem of inferring an unknown B-bit string that is uniform over $\{0,1\}^B$ given certain partial/noisy information about the string. Building on (1) above, we show that the expected edit distance from the output of any algorithm for this problem to the unknown uniform string in $\{0,1\}^B$ will be at least $B(\delta M)^{O(M)}$.

3. We then (in Section 7.3) observe that a random string $\mathbf{x}$ can be viewed as containing, with high probability, $B = n/2^{\Theta(M)}$ independent instances of the distribution distinguishing problem from (1). Roughly speaking, this is because a random string $\mathbf{x} \sim \{0,1\}^n$ can be viewed as composed of $n/(2M+4)$ blocks of $2M+4$ bits each, and with high probability $\Theta(n/2^{2M+4})$ of these blocks will consist of either the string $\alpha = 0^M 10^{M+1} 11$ or the string $\beta = 0^{M+1} 10^M 11$, and these two strings are equally likely for each block. (The specific structure of these α and β strings is chosen to ensure that they cannot overlap; this is useful for (4) below.)

4. Using (3), in Section 7.3 we show that any algorithm that achieves a certain $n(\delta M)^{\Omega(M)}$ expected edit distance for reconstructing a random string from M traces can be used to give an algorithm that solves $B = n/2^{\Theta(M)}$ independent copies of the distinguishing problem described in (1) with an expected edit distance that is lower than can possibly be obtained, contradicting the lower bound from item (2) above. Establishing this reduction is the most intricate part of our lower bound.

2.5 Organization In Section 3 we set up some preliminaries. In Section 4 we prove Theorem 2.1, our main result about the `Align` algorithm. In Section 5 we prove Theorem 2.2, our main result

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

about BMA. In Section 6 we use Theorems 2.1 and 2.2 to prove Theorem 1.1. Finally, Section 7 proves our lower bound, Theorem 1.2.

3 Preliminaries

Notation. Given a positive integer n, we write $[n]$ to denote $\{1, \ldots, n\}$. Given two integers $a \le b$ we write $[a : b]$ to denote $\{a, \ldots, b\}$. We write ln to denote natural logarithm and log to denote logarithm to the base 2. We denote the set of non-negative integers by $\mathbb{Z}_{\ge 0}$. We write "$a = b \pm c$" to indicate that $b - c \le a \le b + c$. For a string s, we use s^∞ to denote the string consisting of infinitely many repetitions of s.

Subwords. It will be convenient for us to index a binary string $\mathsf{x} \in \{0,1\}^n$ using $[1 : n]$ as $\mathsf{x} = (\mathsf{x}_1, \ldots, \mathsf{x}_n)$. Given such a string $\mathsf{x} \in \{0,1\}^n$ and integers $1 \le i \le j \le n$, we write $\mathsf{x}_{[i:j]}$ to denote the *subword* $(\mathsf{x}_i, \mathsf{x}_{i+1}, \ldots, \mathsf{x}_j)$ of x. An *ℓ-subword* of x is a subword of x of length ℓ, given by $(\mathsf{x}_i, \mathsf{x}_{i+1}, \ldots, \mathsf{x}_{i+\ell-1})$ for some $i \in [1 : n - \ell + 1]$.

Distributions. When we use bold font such as $\mathbf{D}, \mathbf{y}, \boldsymbol{z}$, etc., it indicates that the entity in question is a random variable. We write "$\boldsymbol{r} \sim \mathcal{P}$" to indicate that random variable $\boldsymbol{r}$ is distributed according to probability distribution $\mathcal{P}$. If S is a finite set we write "$\boldsymbol{r} \sim S$" to indicate that $\boldsymbol{r}$ is distributed uniformly over S.

Deletion channel and traces. Throughout this paper the parameter $0 < \delta < 1$ denotes the *deletion probability*. Given a string $\mathsf{x} \in \{0,1\}^n$, we write $\mathrm{Del}_\delta(\mathsf{x})$ to denote the distribution of the string that results from passing x through the δ-deletion channel (so the distribution $\mathrm{Del}_\delta(\mathsf{x})$ is supported on $\{0,1\}^{\le n}$), and we refer to a string in the support of $\mathrm{Del}_\delta(\mathsf{x})$ as a *trace* of x. Recall that a random trace $\mathbf{y} \sim \mathrm{Del}_\delta(\mathsf{x})$ is obtained by independently deleting each bit of x with probability δ and concatenating the surviving bits.[3]

When a trace $\mathbf{y}$ is drawn from $\mathrm{Del}_\delta(\mathsf{x})$ we write $\mathbf{D}$ to denote the set of *locations deleted* when x goes through the deletion channel, i.e., $\mathbf{D}$ is obtained by including each element of $[n]$ independently with probability δ, and $\mathbf{y}$ is set to be $\mathsf{x}_{[n]\setminus\mathbf{D}}$. (When the trace is denoted $\mathbf{y}^*$ or $\mathbf{y}^{(m)}$ we use $\mathbf{D}^*$ or $\mathbf{D}^{(m)}$ to denote the set of locations deleted.)

As discussed earlier, our algorithm uses a special reference trace $\mathbf{y}^*$ and M additional traces $\mathbf{y}^{(m)}$, $m \in [M]$, and maintains pointers into each of these traces. We write ℓ^* to denote the pointer into $\mathbf{y}^*$ and $\ell^{(m)}$ to denote the pointer into $\mathbf{y}^{(m)}$ for $m \in [M]$.

Edit distance and matchings. It will be convenient for us to define the edit distance between two strings $\mathsf{x}, \mathsf{x}' \in \{0,1\}^*$ as

$$d_{\mathrm{edit}}(\mathsf{x}, \mathsf{x}') := |\mathsf{x}| + |\mathsf{x}'| - 2 \cdot |\mathrm{LCS}(\mathsf{x}, \mathsf{x}')|,$$

where $|\mathrm{LCS}(\mathsf{x}, \mathsf{x}')|$ is the length of the longest common subsequence of x and x'. This is equivalent to viewing insertions and deletions of characters as being the only allowable "atomic edits" that can be used to transform x to x', and is easily seen to be equivalent to the standard definition (in which substitutions are also allowed) up to at most a factor of 2, since a substitution can be simulated by a deletion followed by an insertion.

[3] For simplicity in this work we assume that the deletion probability δ is known to the reconstruction algorithm. We note that it is possible to obtain a high-accuracy estimate of δ simply by measuring the average length of traces received from the deletion channel.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

A *matching* μ between two strings $\mathsf{x}, \mathsf{x}' \in \{0,1\}^*$ is a list of pairs $(i_1, j_1), (i_2, j_2), \ldots$ such that $i_1 \leq i_2 \leq \cdots$, $j_1 \leq j_2 \leq \cdots$, and for every t we have $\mathsf{x}_{i_t} = \mathsf{x}'_{j_t}$. The *size* of a matching is the number of pairs. We note that the largest matching between x and x' is of length $|\mathrm{LCS}(\mathsf{x}, \mathsf{x}')|$.

For two intervals $A = [a_1, a_2]$ and $B = [b_1, b_2]$ of equal length, we write "$\mu(A) = B$" to indicate that for every element $a_1 + j \in A$, the pair $(a_1 + j, b_1 + j)$ is in the matching (note that this implies that the subwords x_A and x'_B are identical).

Some notational conventions. To aid the reader we adopt the following conventions:

- **Locations in strings of different types:** The letters i, j are reserved for locations in the source string x, so these variables refer to integers in the range $[1 : n]$. We use capital letters I, J to denote intervals of such locations. The letters p, q are reserved for locations in traces, so if p is a location in a particular trace y then it refers to an integer in the range $[1 : |\mathsf{y}|]$. We use capital letters P, Q to denote intervals of such locations. The letters a, b are reserved for locations in other incidental strings that arise in our analysis, and intervals of such locations are denoted A, B.
- **Indexing multiple strings:** On a number of occasions we deal with collections of multiple strings (such as our M traces). We index such collections with parenthesized superscripts, so for example our M traces are denoted $\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(M)}$.
- **Correspondence between traces and source string x.** Given a location $q \in |\mathbf{y}^*|$ in the reference trace $\mathbf{y}^*$, we write $\mathrm{source}^*(q)$ to denote the location $i \in [n]$ such that bit $\mathbf{x}_i$ gave rise to $\mathbf{y}^*_q$. For $m \in [M]$ we similarly write $\mathrm{source}^{(m)}(q)$ to denote the location $i \in [n]$ such that bit $\mathbf{x}_i$ gave rise to $\mathbf{y}^{(m)}_q$ in the trace $\mathbf{y}^{(m)}$. For an interval $Q = [q_1 : q_2]$ of locations in $\mathsf{y}^{(m)}$, we write $\mathrm{source}^{(m)}(Q)$ to denote the set $\{\mathrm{source}^{(m)}(q) : q \in [q_1 : q_2]\}$. We define $\overline{\mathrm{source}^{(m)}(Q)}$ to be the interval $[\mathrm{source}^{(m)}(a) : \mathrm{source}^{(m)}(b)] \subseteq [n]$.

  Given a location $i \in [n]$, if $i \notin \mathbf{D}^*$ then $\mathrm{image}^*(i)$ denotes the element of $[|\mathbf{y}^*|]$ that $\mathbf{x}_i$ lands in (and if $i \in \mathbf{D}^*$ then we define $\mathrm{image}^*(i)$ to be $\perp$). The notation $\overline{\mathrm{image}^{(m)}(i)}$ is defined similarly with respect to trace $\mathbf{y}^{(m)}$, $m \in [M]$. We observe that if $I \subseteq \overline{\mathrm{source}^{(m)}(Q)}$ then $\mathrm{image}^{(m)}(I) \subseteq Q$.
- **Notation for bitstrings.** To help the reader differentiate between bits and the locations of bits in bitstrings, we use sans serif font to denote "bit-valued objects." Hence the uniform random source string in $\{0,1\}^n$ is $\mathbf{x}$, the traces are $\mathbf{y}^*, \mathbf{y}^{(1)}$, etc., a generic fixed word in $\{0,1\}^*$ which is not a random variable would be denoted w, a generic word in $\{0,1\}^*$ which is a random variable would be denoted $\mathbf{w}$, and so on.

Finally we introduce some useful terminology: We refer to a tuple of pointers $(\ell^{(1)}, \ldots, \ell^{(M)})$ into traces $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ (so each $\ell^{(m)}$ belongs to $[|\mathbf{y}^{(m)}|]$) as a *configuration*. We say the configuration $(\ell^{(1)}, \ldots, \ell^{(M)})$ is *in consensus* if at least $0.9M$ of the values $m \in [M]$ all have $\mathrm{source}^{(m)}(\ell^{(m)})$ equal to the same location $i \in [n]$.

3.1 Useful results

We recall McDiarmid's basic "method of bounded differences" inequality, which we will use repeatedly in our analysis:

THEOREM 3.1. (THEOREM 3.1 OF [22]) *Let $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_n)$ be a family of independent random variables where each $\mathbf{X}_k$ takes values in a set A_k. Suppose that $f : A_1 \times \cdots \times A_n \to \mathbb{R}$ satisfies*

$$|f(x) - f(x')| \leq c_k$$

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

whenever the vectors x and x' differ only in the k-th coordinate. Then for any $t \geq 0$, writing μ for $\mathbf{E}[f(\mathbf{X})]$, we have

$$\mathbf{Pr}[f(\mathbf{X}) - \mu \geq t] \leq \exp\left(-2t^2 / \sum_{k=1}^{n} c_k^2\right).$$

We use standard notation for the binary entropy function $H(p) = -p\log p - (1-p)\log(1-p)$, and we recall the standard upper bound on binomial coefficients in terms of this function, namely that $\binom{n}{pn} \leq 2^{nH(p)}$ for any $0 < p < 1$.

4 The Align algorithm and proof of Theorem 2.1

Recall from Section 2.1.1 that the two parameters δ and M satisfy

$$\frac{1}{n^2} \leq \delta < \frac{1}{KM} < \frac{1}{K} \quad \text{and} \quad K^2 \leq M \leq \frac{1}{K\delta}, \tag{4.2}$$

where K is some sufficiently large absolute constant. Let

$$H := \frac{M}{K} \log\left(\frac{1}{\delta M}\right) \leq 2\log n, \tag{4.3}$$

where the inequality is by Equation (2.1). We observe that Equation (4.2) also gives that $H \geq K$, and that $2^{-\Omega(H)} = (\delta M)^{\Omega(M)}$. Let

$$\gamma = 0.01, \quad \tau = 5/\gamma = 500 \tag{4.4}$$

be two constants that will be used in this section.

In this section we describe the (deterministic) **Align** algorithm and prove Theorem 2.1 about its performance. Let $\mathsf{x} \in \{0,1\}^n$ be the source string and $\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$ be traces of x obtained with corresponding deletion sets $D^*, D^{(1)}, \ldots, D^{(M)} \subseteq [n]$, respectively. The algorithm **Align** takes $\ell^*, \mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$ as inputs, where $\ell^* \in [5\tau\log n : |\mathsf{y}^*| - 5\tau\log n]$, and returns a tuple of locations $(\ell^{(1)}, \ldots, \ell^{(M)})$.

The following terminology will be useful: we say that the **Align** algorithm *succeeds* on source string x with respect to a tuple of traces $(\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)})$ if the following condition holds for all except at most $2^{-0.1H}n$ many locations $\ell^* \in [5\tau\log n : |\mathsf{y}^*| - 5\tau\log n]$: The output $(\ell^{(1)}, \ldots, \ell^{(M)})$ of $\mathbf{Align}(\ell^*, \mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)})$ satisfies

1. The configuration $(\ell^{(1)}, \ldots, \ell^{(M)})$ is in consensus, i.e. at least 90% of $\mathrm{source}^{(m)}(\ell^{(m)})$, $m \in [M]$, agree on the same location $i \in [n]$, and
2. The consensus location i satisfies

$$\mathrm{source}^*(\ell^*) - 2H \leq i \leq \mathrm{source}^*(\ell^*). \tag{4.5}$$

Now we can state the main result of this section which gives a performance guarantee on **Align**:

THEOREM 4.1. (THEOREM 2.1, DETAILED STATEMENT) *Let $\mathbf{x} \sim \{0,1\}^n$ and let $\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)} \sim \mathrm{Del}_\delta(\mathbf{x})$ independently, where δ and M satisfy Equations (4.2) and (4.3). **Align** succeeds on $\mathbf{x}$ with respect to $(\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)})$ with probability at least $1 - 1/n^2$.*

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

4.1 Overview We first give a high-level overview of `Align`. Let ℓ^* be a location in the special reference trace y^* that is not too close to the left and right ends of y^*. Let $Q_S^* \supset \cdots \supset Q_1^*$ be a sequence of nested intervals (of locations of y^*) centered at ℓ^*, with $|Q_s^*| = t_s$ for each $s \in [S]$,

$$t_1 = 2H + 1, \quad t_{s+1} = 3t_s, \quad \text{and } t_S = \Theta(\log n)$$

(note that hence $S = O(\log \log n)$). Let $\mathsf{w}_s^* = \mathsf{y}_{Q_s^*}^*$ for each $s \in [S]$.

The `Align` algorithm consists of two stages. In this subsection we give some intuition behind each stage and its analysis. In the intuitive discussion below, we focus chiefly on understanding the probability that `Align` succeeds at a particular location ℓ^*; in the formal proof we need to apply the bounded difference inequality of McDiarmid to argue that `Align` succeeds on all but except $2^{-\Omega(H)}n$ many locations with high probability.

For the rest of this section, we say that an event is *most likely* to happen if it happens with probability $1 - 2^{-\Omega(H)}$.

First stage — locating a small neighborhood of $\mathrm{source}^*(\ell^*)$ in each trace $\mathsf{y}^{(m)}$.

In the first stage, `Align` works separately on each $\mathsf{y}^{(m)}$, $m \in [M]$. It iteratively uses $\mathsf{w}_S^*, \dots, \mathsf{w}_1^*$ (as templates) to find a sequence of nested intervals $Q_S^m \supset \cdots \supset Q_1^m$ of locations of $\mathsf{y}^{(m)}$ such that

$$d_{\mathrm{edit}}\left(\mathsf{w}_s^*, \mathsf{y}_{Q_s^m}^{(m)}\right) \le 2\gamma t_s, \quad \text{for each } s = S, \dots, 1. \tag{4.6}$$

This is done by first finding $Q_S^m \subset [|\mathsf{y}^{(m)}|]$ that satisfies (4.6) and then repeatedly finding $Q_s^m \subset Q_{s+1}^m$ that satisfies (4.6), for each $s = S-1, \dots, 1$. When multiple Q_s^m satisfy (4.6), we pick one arbitrarily; when no interval Q_s^m exists for some s and some m, `Align` fails and returns $\ell^{(1)} = \cdots = \ell^{(M)} = 1$.

In the analysis we show that when $\mathbf{x} \sim \{0,1\}^n$ and $\mathbf{y}^*, \mathbf{y}^{(1)}, \dots, \mathbf{y}^{(M)} \sim \mathrm{Del}_\delta(\mathbf{x})$, it is most likely that every $m \in [M]$ satisfies

$$\left|\overline{\mathrm{source}^*(Q_s^*)} \triangle \overline{\mathrm{source}^{(m)}(Q_s^m)}\right| \le 4\gamma t_s, \quad \text{for each } s = S, \dots, 1; \tag{4.7}$$

in words, this means that the interval Q_s^m of $\mathbf{y}^{(m)}$ (almost) comes from the subword of $\mathbf{x}$ whose image is Q_s^* in $\mathbf{y}^*$. The proof proceeds by induction on $s = S, \dots, 1$ (see Lemma 4.2). Assume that (4.7) holds for $s+1$:

$$\left|\overline{\mathrm{source}^*(Q_{s+1}^*)} \triangle \overline{\mathrm{source}^{(m)}(Q_{s+1}^m)}\right| \le 4\gamma t_{s+1}. \tag{4.8}$$

Then most likely $I_s^* := \overline{\mathrm{source}^*(Q_s^*)}$ is contained in $\overline{\mathrm{source}^{(m)}(Q_{s+1}^m)}$ given (4.8) and that I_s^* is roughly the middle one-third of $\overline{\mathrm{source}^*(Q_{s+1}^*)}$ (also recall that $\gamma = 0.01$ is a small constant). As a result, $\mathrm{image}^{(m)}(I_s^*)$ would most likely satisfy (4.6) as Q_s^m (using that δ is sufficiently smaller than γ and thus, the number of bits deleted from I_s^* in getting both $\mathbf{y}^*$ and $\mathbf{y}^{(m)}$ is smaller than γt_s). On the other hand, let Q_s^m be the interval actually picked by `Align`. To finish the proof of (4.7), we show that when (4.8) is violated, since $\mathbf{x} \sim \{0,1\}^n$, the two subwords $x_{\overline{\mathrm{source}^*(Q_s^*)}}$ and $x_{\overline{\mathrm{source}^{(m)}(Q_s^m)}}$ most likely have large edit distance (see Claim 4.2), which in turn implies that $\mathbf{y}_{Q_s^*}^*$ (i.e., the string w_s^*) has large edit distance from $\mathbf{y}_{Q_s^m}^{(m)}$, which contradicts (4.6).

Second stage — determining a consensus location close to $\mathrm{source}^*(\ell^*)$.

In the second stage, `Align` uses subwords $\mathsf{y}_{Q_1^m}^{(m)}$, $m \in [M]$, to determine the final locations $\ell^{(1)}, \dots, \ell^{(M)}$. This is done by first identifying a string w that (a) has length at least $0.9t_1$, and (b)

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

appears as a subword in at least 95% of $\mathsf{y}^{(m)}_{Q_1^m}$, $m \in [M]$. (When multiple strings w satisfy the two conditions, `Align` picks one arbitrarily; when no such w exists, `Align` fails and sets $\ell^{(m)} = 1$ for all $m \in [M]$.) Finally `Align` finds w in $\mathsf{y}^{(m)}_{Q_1^m}$ and sets $\ell^{(m)}$ to be the location of the first symbol of w in $\mathsf{y}^{(m)}_{Q_1^m}$, for each $m \in [M]$. (When w appears in $\mathsf{y}^{(m)}_{Q_1^m}$ at multiple locations, `Align` picks one of them arbitrarily as $\ell^{(m)}$; when w does not appear, `Align` sets $\ell^{(m)} = 1$ by default.)

Using $\mathbf{x} \sim \{0,1\}^n$ and $\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)} \sim \mathrm{Del}_\delta(\mathbf{x})$, we show that most likely $\ell^{(1)}, \ldots, \ell^{(M)}$ satisfy the two desired conditions in the definition of "success" (i.e. at least 95% of $\mathrm{source}^{(m)}(\ell^{(m)})$, $m \in [M]$, agree on the same location $i \in [n]$, and this consensus location i satisfies (4.5)). To give some intuition behind the analysis, we first assume that every $m \in [M]$ satisfies (4.7) and in particular,

$$\left| \overline{\mathrm{source}^*(Q_1^*)} \triangle \overline{\mathrm{source}^{(m)}(Q_1^m)} \right| \le 4\gamma t_1. \tag{4.9}$$

Since $|Q_1^*| = t_1$, most likely $I_1^* := \overline{\mathrm{source}^*(Q_1^*)}$ has length close to t_1. Let I_1^{**} denote the interval obtained from I_1^* by extending it in both directions by $4\gamma t_1$ (so I_1^{**} also has length close to t_1 since $\gamma = 0.01$). Using our choice of $t_1 = 2H + 1$, it follows from simple calculations that most likely at least 95% of $\mathbf{y}^{(m)}$, $m \in [M]$, are obtained from $\mathbf{x}$ with no deletions in I_1^{**}. Let $G \subseteq [M]$ be the set of such $m \in [M]$. It follows from (4.9) that $\mathbf{y}^{(m)}_{Q_1^m}$ comes from $\mathbf{x}_{I_1^m}$ with no deletions for some interval I_1^m such that $|I_1^m \triangle I_1^*| \le 4\gamma t_1$, for each $m \in G$.

At this point it is clear that $\mathsf{w} = \mathbf{x}_{\cap_{m \in G} I_1^m}$ would satisfy both conditions (a) and (b) (using $\gamma = 0.01$). On the other hand, if there is a string w that appears in at least 95% of all $x_{I_1^m}$, $m \in [M]$, then for at least 90% of $m \in [M]$, w appears in $\mathbf{y}^{(m)}_{Q_1^m}$ and $m \in G$. Using the randomness of $\mathbf{x} \sim \{0,1\}^n$, one can argue that most likely no string of length at least $0.9t_1$ can appear as a subword more than once in $\mathbf{x}_{I_1^{**}}$. This implies that at least 90% of $\ell^{(m)}$ returned are in consensus. To see that the consensus location $i \in [n]$ satisfies (4.5), we recall $t_1 = 2H + 1$ and observe that $\mathrm{source}^*(\ell^*)$ appears around the middle of I_1^{**} but i (as the unique location where w appears as a subword in $\mathsf{x}_{I_1^{**}}$) lies close to its left end.

4.2 Algorithm `Align` We now describe the algorithm `Align` which takes as input $\ell^*, \mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$ with $\ell^* \in [5\tau \log n : |\mathsf{y}^*| - 5\tau \log n]$. (See Algorithm 2 for a formal presentation of the algorithm.)

`Align` starts by computing a sequence of nested subwords of y^* centered at ℓ^* as follows. Let $t_1 = 2H + 1$ (recall that $H \le 2 \log n$) and $t_s = 3t_{s-1}$ for each $s \ge 1$, and let S be the smallest integer such that

$$t_S \ge \tau \log n \quad (\text{and hence } t_S \le 3\tau \log n)$$

(recall that $\tau = 500$). Given y^* and ℓ^*, we define the sequence of subwords $\mathsf{w}_1^*, \ldots, \mathsf{w}_S^*$, where w_s^* is the t_s-bit subword $\mathsf{y}^*_{Q_s^*}$ with $Q_s^* = [\ell^* - (t_s - 1)/2 : \ell^* + (t_s - 1)/2]$ centered at ℓ^* in y^*. (Given that $t_S \le 3\tau \log n$, we always have $Q_S^* \subset [|\mathsf{y}^*|]$; indeed we have that there are more than t_S elements to the left and to the right of Q_S^* in $[|\mathsf{y}^*|]$, which is the reason why we only consider ℓ^*'s that are at least $5\tau \log n$ away from both ends of y^*.)

We divide the analysis of `Align` into two parts. In the first part (Section 4.3) we begin by describing some good events over the randomness of $\mathbf{D}^*$, $\mathbf{x}$, and $\mathbf{D}^{(m)}$, $m \in [M]$, where $\mathbf{D}^*$ and $\mathbf{D}^{(m)}$ are the sets of deleted locations that gave rise to traces $\mathbf{y}^*$ and $\mathbf{y}^{(m)}$ of $\mathbf{x}$, respectively. We then show that these events happen with probability at least $1 - 1/n^2$. The second part of our analysis (Section 4.4) will be entirely deterministic. We show that `Align` succeeds on x with

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Algorithm 2: `Align`

Input: A location ℓ^* and a tuple of $M+1$ strings $\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$

Output: M locations $\ell^{(1)}, \ldots, \ell^{(M)}$, where $\ell^{(i)} \in [|\mathsf{y}^{(i)}|]$ is a location in $\mathsf{y}^{(i)}$

Compute $\mathsf{w}^*_1, \ldots, \mathsf{w}^*_S$ from y^* as defined in Section 4.2. Let $Q^m_{S+1} = [|\mathsf{y}^{(m)}|]$ for each $m \in [M]$.

for *each* $m \in [M]$ **do** `// First stage`

  for $s = S, \ldots, 1$ **do**

    Find any subword $\mathsf{y}^{(m)}_{Q^m_s}$ in $\mathsf{y}^{(m)}_{Q^m_{s+1}}$ (breaking ties arbitrarily) that has edit distance at most $2\gamma t_s$ from w^*_s; if such a subword does not exist **return** $\ell^{(1)} = \cdots = \ell^{(M)} = 1$.

Find any string w of length at least $0.9t_1$ that appears as a subword in at least 95% of $\mathsf{y}^{(m)}_{Q^m_1}$, $m \in [M]$. If no such w exists, **return** $\ell^{(1)} = \cdots = \ell^{(M)} = 1$. `// Second stage`

for *each* $m \in [M]$ **do**

  If $\mathsf{y}^{(m)}_{Q^m_1}$ has w as a subword, set $\ell^{(m)}$ to be any location such that $\mathsf{y}^{(m)}_{Q^m_1}$ has w as a subword starting at $\ell^{(m)}$; otherwise ($\mathsf{y}^{(m)}_{Q^m_1}$ does not contain w as a subword), set $\ell^{(m)} = 1$.

return $\ell^{(1)}, \ldots, \ell^{(M)}$

Figure 2: The `Align` algorithm.

respect to $(\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)})$ whenever D^*, x and $D^{(m)} : m \in [M]$ satisfy all conditions described in the first part (Section 4.3).

4.3 Probabilistic Analysis Let $\mathcal{D}$ denote the distribution over subsets of $[n]$ where $\mathbf{D} \sim \mathcal{D}$ is drawn by including each integer of $[n]$ independently with probability δ. We prove Theorem 4.1 in two steps. In this subsection we describe an event over $\mathbf{x} \sim \{0,1\}^n$ and $\mathbf{D}^*, \mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(M)} \sim \mathcal{D}$ (as deletions used to obtain $\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ from $\mathbf{x}$) and show that it happens with probability at least $1 - 1/n^2$ (see Corollary 4.2). In Section 4.4, we show that whenever the event occurs, `Align` succeeds on $\mathbf{x}$ with respect to $(\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)})$.

We describe the event by imposing conditions on random variables in the following order: first $\mathbf{D}^* \sim \mathcal{D}$, then $\mathbf{x} \sim \{0,1\}^n$ and finally $\mathbf{D}^{(m)} \sim \mathcal{D}$, $m \in [M]$. We describe conditions on each random variable conditioning on the event that previous ones have already met conditions imposed on them.

We start with some preliminary claims.

CLAIM 4.1. *Let t be a positive integer and let $1 \le i_1 < \cdots < i_t \le n$ and $1 \le j_1 < \cdots < j_t \le n$ with $i_k \ne j_k$ for all $k \in [t]$. For $\mathbf{x} \sim \{0,1\}^n$, we have $\mathbf{Pr}[\mathbf{x}_{i_k} = \mathbf{x}_{j_k}$ for all $k \in [t]] = 2^{-t}$.*

Proof. The proof is by induction on t. The base case when $t = 1$ is trivial. For the inductive step, we assume that the statement holds for $t-1$. Using the induction hypothesis we have

$$\mathbf{Pr}\left[\mathbf{x}_{i_k} = \mathbf{x}_{j_k} \text{ for all } k \in [t]\right] = \mathbf{Pr}\left[\mathbf{x}_{i_t} = \mathbf{x}_{j_t} \mid \mathbf{x}_{i_k} = \mathbf{x}_{j_k} \text{ for all } k \in [t-1]\right] \cdot 2^{-(t-1)}.$$

Without loss of generality we assume that $i_t < j_t$. Note that $\mathbf{x}_{j_t}$ is still uniform when conditioned

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

on values of $\mathbf{x}_{i_k} : k \le t$ and $\mathbf{x}_{j_k} : k < t$. Therefore the conditional probability on the right hand side is $1/2$. This finishes the induction step and the proof of the claim. □

Claim 4.1 has the following corollary which we will use later:

COROLLARY 4.1. *Let t be a positive integer, and let $I \neq I' \subseteq [n]$ be two distinct (but not necessarily disjoint) intervals of length t. For $\mathbf{x} \sim \{0,1\}^n$, we have $\mathbf{Pr}[\mathbf{x}_I = \mathbf{x}_{I'}] = 2^{-t}$.*

Claim 4.1 lets us bound the edit distance between subwords of a random string as follows:

CLAIM 4.2. *Let t be a positive integer. Let $I, I' \subseteq [n]$ be two intervals that satisfy (1) $|I| \ge 25t$ and (2) $|I \triangle I'| \ge t$. Then we have $d_{\text{edit}}(\mathbf{x}_I, \mathbf{x}_{I'}) < t$ with probability at most 2^{-5t} when $\mathbf{x} \sim \{0,1\}^n$.*

Proof. Having $d_{\text{edit}}(\mathbf{x}_I, \mathbf{x}_{I'}) < t$ implies that there exist $J \subseteq I$ and $J' \subseteq I'$ such that $|J| + |J'| < t$, $|I \setminus J| = |I' \setminus J'|$ and $\mathbf{x}_{I \setminus J} = \mathbf{x}_{I' \setminus J'}$. Fixing such a pair (J, J') and writing $I \setminus J$ as $i_1 < i_2 < \cdots$ and $I' \setminus J'$ as $j_1 < j_2 < \cdots$, we claim that $i_k \neq j_k$ for all k. To see this we note that having $i_k = j_k$ for some k implies that we need to delete at least $|I \triangle I'| \ge t$ bits from I and I' even just to match the lengths of I to the left and to the right of k with those of I', a contradiction with $|J| + |J'| < t$.

Therefore, it follows from Claim 4.1 that $\mathbf{x}_{I \setminus J} = \mathbf{x}_{I' \setminus J'}$ with probability at most $2^{-(|I|+|I'|-t)/2}$. It follows by a union bound on all pairs (J, J') that $d_{\text{edit}}(\mathbf{x}_I, \mathbf{x}_{I'}) < t$ with probability at most

$$2^{-(|I|+|I'|-t)/2} \cdot \sum_{k \le t} \binom{|I|}{k} \cdot \sum_{k \le t} \binom{|I'|}{k} \le 2^{-(|I|+|I'|-t)/2} \cdot 2^{H(0.04)(|I|+|I'|)} < 2^{-5t}.$$

This finishes the proof of the claim. □

4.3.1 Conditions on $\mathbf{D}^* \sim \mathcal{D}$ We start with conditions on $\mathbf{D}^* \sim \mathcal{D}$, i.e., the locations of bits deleted in $\mathbf{y}^*$. Given an outcome $D^* \subseteq [n]$ of $\mathbf{D}^*$, we write L^* to denote the interval

$$L^* := [5\tau \log n : n - |D^*| - 5\tau \log n].$$

For each $\ell^* \in L^*$ and $s \in [S]$ we write Q^*_{s,ℓ^*} to denote the interval of length t_s that is centered at ℓ^*. Let L^*_1 denote the set of $\ell^* \in L^*$ such that

$$\left|\overline{\text{source}^*(Q^*_{s,\ell^*})}\right| \le (1+\gamma)t_s, \quad \text{for all } s \in [S].$$

CLAIM 4.3. *With probability at least $1 - \exp(-n^{0.1})$ over $\mathbf{D}^*$, we have $|\boldsymbol{L}^* \setminus \boldsymbol{L}^*_1| \le 2^{-0.2H} n$.*

Proof. Given D^*, for each $\ell^* \in L^* \setminus L^*_1$ there is an $s \in [S]$ such that $|\overline{\text{source}^*(Q^*_{s,\ell^*})}| \ge (1+\gamma)t_s$. We can get from it an interval I with $|I| = (1+\gamma)t_s$ and $|\text{image}^*(I)| \le |Q^*_{s,\ell^*}| = t_s$ by deleting elements from the right end of $\overline{\text{source}^*(Q^*_{s,\ell^*})}$. We note that the intervals I obtained from different $\ell^* \in L^* \setminus L^*_1$ are different. (To obtain the same interval, we must use the same $s \in [S]$ because of the length of I; on the other hand, sharing the same left end and the same s implies that the ℓ^* is the same as well.) Therefore, $|L^* \setminus L^*_1|$ is at most the number of intervals $I \subseteq [n]$ such that $|I| = (1+\gamma)t_s$ for some $s \in [S]$ and $|\text{image}^*(I)| \le t_s$. Below we upperbound the latter when $\mathbf{D}^* \sim \mathcal{D}$.

We apply the McDiarmid inequality (Theorem 3.1). We draw $\mathbf{D}^*$ by drawing n independent random indicator variables $\mathbf{X}_1, \ldots, \mathbf{X}_n$ with $\mathbf{X}_k = 1$ with probability δ (so $k \in \mathbf{D}^*$ if $\mathbf{X}_k = 1$). We

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

use $f(\mathbf{X}_1,\dots,\mathbf{X}_n)$ to denote the number of $I \subseteq [n]$ such that $|I| = (1+\gamma)|t_s|$ for some $s \in [S]$ and $|\text{image}^*(I)| \le t_s$. On the one hand, the probability of an interval I with $|I| = (1+\gamma)t_s$ satisfying $|\text{image}^*(I)| \le t_s$ is at most $2^{|I|} \cdot \delta^{\gamma t_s} \le \delta^{\gamma t_s/2}$, by using $\delta \le 1/K$ and making K sufficiently large. As a result,

$$\mathbf{E}[f] \le n\left(\sum_{s=1}^{S} \delta^{\gamma t_s/2}\right) = \delta^{\Omega(t_1)}n = \delta^{\Omega(H)}n.$$

On the other hand, flipping one variable $\mathbf{X}_k$ can change f by no more than $O(t_S) = O(\log n)$. Thus it follows from the McDiarmid inequality that

$$f(\mathbf{X}_1,\dots,\mathbf{X}_n) \le \delta^{\Omega(H)}n + \tilde{O}(n^{0.55}) \le 2^{-0.2H}n,$$

with probability at least $1-\exp(-n^{0.1})$, where we used that δ is sufficiently small and $H \le 2\log n$ in the last inequality. This finishes the proof of the claim. ∎

4.3.2 Conditions on $\mathbf{x} \sim \{0,1\}^n$ We fix a $D^* \subseteq [n]$ that satisfies Claim 4.3 when describing the conditions for $\mathbf{x}$ and $\mathbf{D}^{(m)}$ below. As D^* is fixed, L^*, L_1^* and $\overline{\text{source}^*(Q^*_{s,\ell^*})}$ for each $\ell^* \in L_1^*$ are all fixed and are no longer random variables. For each $\ell^* \in L_1^*$, for brevity we write I^*_{s,ℓ^*} to denote $\overline{\text{source}^*(Q^*_{s,\ell^*})}$, and we observe that for each $\ell^* \in L_1^*$ we have

$$t_s \le |I^*_{s,\ell^*}| \le (1+\gamma)t_s. \tag{4.10}$$

The conditions for $\mathbf{x} \sim \{0,1\}^n$ are given in the next three claims.

CLAIM 4.4. *With probability at least $1 - 1/n^3$ over $\mathbf{x} \sim \{0,1\}^n$, every $\ell^* \in L_1^*$ and every interval $I \subseteq [n]$ with $|I \triangle I^*_{S,\ell^*}| \ge 4\gamma t_S$ satisfy $d_{\text{edit}}(\mathbf{x}_{I^*_{S,\ell^*}}, \mathbf{x}_I) \ge 4\gamma t_S$.*

Proof. Recall that $|I^*_{S,\ell^*}| \ge t_S$ and $\gamma = 0.01$. Fix an $\ell^* \in L_1^*$ and an interval I with $|I \triangle I^*_{S,\ell^*}| \ge 4\gamma t_S$. By Claim 4.2 we have that $d_{\text{edit}}(\mathbf{x}_{I^*_{S,\ell^*}}, \mathbf{x}_I) < 4\gamma t_S$ occurs with probability at most $2^{-20\gamma t_S} \le 1/n^{100}$, using $t_S \ge \tau \log n$ and $\gamma\tau = 5$. The claim follows by a union bound over no more than n^3 pairs of ℓ^* and I. ∎

For the next claim we need the following notation. Given $\ell^* \in L^*$ and $s \in [S]$, we let $N(I^*_{s,\ell^*})$ denote the interval obtained by adding t_s elements to both ends of I^*_{s,ℓ^*} (so $N(I^*_{s,\ell^*})$ has length $|I^*_{s,\ell^*}| + 2t_s$). Note that $N(I^*_{s,\ell^*})$ is an interval contained in $[n]$ given that $\ell^* \in L^*$ is at least $5\tau \log n$ from both ends of $[n - |D^*|]$ (recall that $t_s \le t_S \le 3\tau \log n$).

CLAIM 4.5. *With probability at least $1-\exp(-n^{0.1})$ over $\mathbf{x} \sim \{0,1\}^n$, all but at most $2^{-0.2H}n$ many locations $\ell^* \in L_1^*$ satisfy the following condition: For any $s \in [2:S]$ and any interval $I \subseteq N(I^*_{s,\ell^*})$ such that $|I \triangle I^*_{s-1,\ell^*}| \ge 4\gamma t_{s-1}$, we have $d_{\text{edit}}(\mathbf{x}_I, \mathbf{x}_{I^*_{s-1,\ell^*}}) \ge 4\gamma t_{s-1}$.*

Proof. Again we use the McDiarmid inequality. Let $f(\mathbf{x}_1,\dots,\mathbf{x}_n)$ denote the number of $\ell^* \in L_1^*$ that violates the condition. We first upperbound the probability over $\mathbf{x}_1,\dots,\mathbf{x}_n$ of a fixed $\ell^* \in L_1^*$ violating the condition.

For each $s \in [2:S]$ and each interval $I \subseteq N(I^*_{s,\ell^*})$ with $|I \triangle I^*_{s-1,\ell^*}| \ge 4\gamma t_{s-1}$ (note that I^*_{s-1,ℓ^*} has length at least t_{s-1}), by Claim 4.2 the probability of $d_{\text{edit}}(\mathbf{x}_I, \mathbf{x}_{I^*_{s-1,\ell^*}}) < 4\gamma t_{s-1}$ is at most $2^{-20\gamma t_{s-1}}$. As

$$\left|N(I^*_{s,\ell^*})\right| \le |I^*_{s,\ell^*}| + 2t_s \le (3+\gamma)t_s,$$

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

by (4.10), it follows from a union bound that each ℓ^* in L_1^* violates the condition with probability at most

$$\sum_{s\in[2:S]} O(t_s^2)\cdot 2^{-20\gamma t_{s-1}} \le \sum_{s\in[2:S]} 2^{-19\gamma t_{s-1}} \le 2^{-18\gamma t_1} \le 2^{-0.3H},$$

using $t_1 = 2H+1$ and H is sufficiently large, and hence $\mathbf{E}[f] \le 2^{-0.3H}n$. Given that each variable $\mathbf{x}_i$ can change f by no more than $O(\log n)$, the lemma follows from arguments similar to the proof of Claim 4.3. ∎

CLAIM 4.6. *With probability at least* $1-\exp(-n^{0.1})$ *over* $\mathbf{x}\sim\{0,1\}^n$, *all but at most* $2^{-0.2H}n$ *many* $\ell^*\in L_1^*$ *are such that no two subwords of* $\mathbf{x}_{N(I^*_{1,\ell^*})}$ *of length* H *are the same.*

Proof. The probability of an $\ell^*\in L_1^*$ violating the above condition is at most $O(t_1^2)\cdot 2^{-H} \le 2^{-0.5H}$ by Corollary 4.1. The proof follows from a similar application of McDiarmid inequality. ∎

4.3.3 Conditions on $\mathbf{D}^{(1)},\dots,\mathbf{D}^{(M)}\sim\mathcal{D}$ We fix an outcome D^* that satisfies Claim 4.3 and a string $\mathsf{x}\in\{0,1\}^n$ that satisfies Claim 4.4, Claim 4.5, and Claim 4.6.

We now describe some useful conditions on $\mathbf{D}^{(1)},\dots,\mathbf{D}^{(M)}$. We start with two conditions for every $\mathbf{D}^{(m)}$.

CLAIM 4.7. *With probability at least* $1-1/n^3$ *over* $\mathbf{D}\sim\mathcal{D}$, *every interval* $I\subseteq[n]$ *of length at most* $(1+3\gamma)t_S$ *satisfies that* $|\mathrm{image}(I)| \ge |I| - \gamma t_S$.

Proof. For each $I\subseteq[n]$ of length at most $(1+3\gamma)t_S$, we have $|\mathrm{image}(I)| < |I|-\gamma t_S$ with probability at most $2^{|I|}\cdot\delta^{\gamma t_S} \le \delta^{\gamma t_S/2}$, where the inequality uses that δ is sufficiently small. Using $t_S \ge \tau\log n$ we have that $\delta^{\gamma t_S/2} \le \delta^{5\log n/2} \le 1/n^5$ (as δ is sufficiently small). The claim then follows from a union bound. ∎

REMARK 4.1. We note that the event described in Claim 4.7 implies that any interval $Q\subseteq\underline{[n-|\mathbf{D}|]}$ with length at most $(1+2\gamma)t_S$ must satisfy $|\overline{\mathrm{source}(Q)}| \le |Q|+\gamma t_S$. To see this, let $I=\overline{\mathrm{source}(Q)}$ and assume for a contradiction that $|I| > |Q|+\gamma t_S$. If $|I|\le(1+3\gamma)t_S$ then I violates the event of Claim 4.7; if $|I|>(1+3\gamma)t_S$ then we can delete bits of I from the beginning to obtain an interval I' with $|I'|=(1+3\gamma)t_S$, which satisfies $|\mathrm{image}(I')| < |\mathrm{image}(I)| \le (1+2\gamma)t_S$ and thus, I' violates the condition of Claim 4.7.

Let L_2^* be the set of all $\ell^*\in L_1^*$ that satisfy the conditions in Claim 4.5 and Claim 4.6.

CLAIM 4.8. *With probability at least* $1-\exp(-n^{0.1})$ *over* $\mathbf{D}\sim\mathcal{D}$, *all but at most* $2^{-0.2H}n$ *many* $\ell^*\in L_2^*$ *satisfy the following condition: For every* $s\in[2:S]$ *and every interval* $I\subseteq N(I^*_{s,\ell^*})$ *of length at most* $(1+3\gamma)t_{s-1}$, *we have* $|\mathrm{image}(I)| \ge |I| - \gamma t_{s-1}$.

Proof. We upper bound the probability of an $\ell^*\in L_2^*$ violating the condition above, and then apply the McDiarmid inequality. Fixing an $\ell^*\in L_2^*$, a value of $s\in[2:S]$, and any interval $I\subseteq N(I^*_{s,\ell^*})$ of length at most $(1+3\gamma)t_{s-1}$, I violates the condition with probability $2^{|I|}\cdot\delta^{\gamma t_{s-1}} \le \delta^{\gamma t_{s-1}/2}$. By a union bound (over all possibilities for s and I), the probability of ℓ^* violating the condition is at most

$$\sum_{s\in[2:S]} O(t_s^2)\cdot\delta^{\gamma t_{s-1}/2},$$

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

and hence the expected number of $\ell^* \in L_2^*$ that violate the condition is at most n times this, which is at most $2^{-0.3H}n$ using that $t_{s-1} \geq t_1 = 2H+1$ and δ is sufficiently small. Finally, given that the outcome of each independent event (of whether an element in $[n]$ is included in $\mathbf{D}$ or not) can change the number of ℓ^* that satisfy the condition by at most $O(\log n)$, the lemma follows from arguments similar to the proof of Claim 4.3. □

REMARK 4.2. Remark 4.1 applies similarly: Whenever the condition holds for $\ell^* \in L_2^*$, any interval $Q \subseteq \text{image}(N(I^*_{s,\ell^*}))$ for any s with length at most $(1+2\gamma)t_{s-1}$ satisfies $|\overline{\text{source}(Q)}| \leq |Q| + \gamma t_{s-1}$.

The last condition considers $\mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(M)}$ together:

CLAIM 4.9. *With probability at least* $1 - \exp(-n^{0.1})$ *over* $\mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(M)} \sim \mathcal{D}$, *all but at most* $2^{-0.2H}n$ *many* $\ell^* \in L_2^*$ *satisfy the following condition: At least* 95% *of* $m \in [M]$ *satisfy*

$$N(I^*_{1,\ell^*}) \cap \mathbf{D}^{(m)} = \varnothing,$$

i.e., no bit of the subword $\mathbf{x}_{N(I^*_{1,\ell^*})}$ *of* $\mathbf{x}$ *is deleted in at least* 95% *of the traces* $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$.

Proof. Consider drawing $\mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(M)}$ by drawing nM independent indicator random variables $\mathbf{X}_k^{(m)}$, $k \in [n]$ and $m \in [M]$, with $k \in \mathbf{D}^{(m)}$ if $\mathbf{X}_k^{(m)} = 1$. We write f to denote the number of $\ell^* \in L_2^*$ such that at least 5% of $m \in [M]$ have $N(I^*_{1,\ell^*}) \cap \mathbf{D}^{(m)} \neq \varnothing$. On the one hand, fixing an outcome of ℓ^*, the probability of $N(I^*_{1,\ell^*}) \cap \mathbf{D}^{(m)} = \varnothing$ is at most $7\delta H$ given that $|N(I^*_{1,\ell^*})| \leq (3+\gamma)t_1 \leq 7H$, and hence the probability of ℓ^* being one of the locations counted in f is at most $2^M \cdot (7\delta H)^{0.05M}$. Recalling the constraint Equation (4.3) on H, we have that

$$\delta H = \frac{1}{K} \cdot \delta M \log\left(\frac{1}{\delta M}\right) \leq \frac{\sqrt{\delta M}}{K} \tag{4.11}$$

where the inequality holds given that δM is sufficiently small (observe from Equation (4.2) that $\delta M \leq 1/K$). Hence the probability is at most

$$2^M \cdot (7\delta H)^{0.05M} \leq (\delta M)^{0.025M} \leq 2^{-0.3H}$$

where the first inequality is by Equation (4.11) and the second uses Equation (4.3) and the fact that K is sufficiently large. Recalling that $H = O(\log n)$, the claim follows from the McDiarmid inequality using similar arguments to those given above and the fact that changing the outcome of any one of the nM independent indicator random variables can only change f by at most $O(H)$. □

4.3.4 Conclusion of Probabilistic Analysis We summarize our probabilistic analysis with the following corollary, which combines all the claims from this subsection.

COROLLARY 4.2. *With probability at least* $1 - 1/n^2$ *over the randomness of* $\mathbf{D}^*, \mathbf{x}, \mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(M)}$, *all of the following hold:*

1. $\mathbf{D}^*$ *satisfies Claim 4.3;*
2. $\mathbf{x}$ *satisfies Claim 4.4, Claim 4.5 and Claim 4.6;*
3. *Every* $\mathbf{D}^{(m)}$, $m \in [M]$, *satisfies Claim 4.7 and Claim 4.8, and*
4. $\mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(M)}$ *together satisfy Claim 4.9.*

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

4.4 Deterministic Analysis The rest of Section 4 is dedicated to proving the following lemma, which finishes the proof of Theorem 4.1 (and hence Theorem 2.1):

LEMMA 4.1. `Align` *succeeds on* x *with respect to* $\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$ *when they satisfy Corollary 4.2.*

Assume that $\mathsf{x}, D^*, D^{(1)}, \ldots, D^{(M)}$ satisfy all conditions of Corollary 4.2. Then we have that all but at most $O(M2^{-0.2H}n) \le 2^{-0.1H}n$ (where the M comes from a union bound in item (vi) and the inequality follows from $H > M/K \ge \sqrt{M}$ using Equation (4.3) and $M \ge K^2$ from Equation (4.2) and thus, $2^{0.1H}$ is enough to cover $O(M)$ when M is sufficient large) many $\ell^* \in L^*$ satisfy the following list of conditions (below we use I_s^* to denote I_{s,ℓ^*}^* for convenience given that ℓ^* is fixed in the rest of the proof):

(i) $|I_s^*| < (1+\gamma)t_s$ for every $s \in [S]$ (Claim 4.3);

(ii) Every interval $I \subseteq [n]$ with $|I \triangle I_S^*| \ge 4\gamma t_S$, has $d_{\text{edit}}(\mathsf{x}_I, \mathsf{x}_{I_S^*}) \ge 4\gamma t_S$ (Claim 4.4);

(iii) For all $s \in [2:S]$ and intervals $I \subseteq N(I_s^*)$ with $|I \triangle I_{s-1}^*| \ge 4\gamma t_{s-1}$, it holds that $d_{\text{edit}}(\mathsf{x}_I, \mathsf{x}_{I_{s-1}^*}) \ge 4\gamma t_{s-1}$ (Claim 4.5);

(iv) No two subwords of $\mathsf{x}_{N(I_1^*)}$ of length H are the same (Claim 4.6);

(v) (Claim 4.7 and Remark 4.1) For all $m \in [M]$, $|\text{image}^{(m)}(I_S^*)| \ge |I_S^*| - \gamma t_S$ and every interval $Q^m \subseteq [|\mathsf{y}^{(m)}|]$ of length at most $(1+2\gamma)t_S$ satisfies

$$\left|\overline{\text{source}^{(m)}(Q^m)}\right| \le |Q^m| + \gamma t_S;$$

(vi) (Claim 4.8 and Remark 4.2) For all $m \in [M]$ and $s \in [2:S]$, $|\text{image}^{(m)}(I_{s-1}^*)| \ge |I_{s-1}^*| - \gamma t_{s-1}$ and every interval $Q^m \subseteq \text{image}(N(I_s^*))$ of length at most $(1+2\gamma)t_{s-1}$ satisfies

$$\left|\overline{\text{source}^{(m)}(Q^m)}\right| \le |Q^m| + \gamma t_{s-1};$$

(vii) At least 95% of $m \in [M]$ satisfy that $N(I_1^*) \cap D^{(m)} = \varnothing$ (Claim 4.9).

4.4.1 First stage: Locating a small neighborhood of $\text{source}^*(\ell^*)$ in each trace $\mathsf{y}^{(m)}$ We prove the following lemma for the first stage of $\texttt{Align}(\ell^*, \mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)})$ (lines 2 to 4):

LEMMA 4.2. *For every* $m \in [M]$*, the final interval* Q_1^m *found by* `Align` *in the first stage satisfies*

$$\left|\overline{\text{source}^{(m)}(Q_1^m)} \triangle I_1^*\right| \le 4\gamma t_1. \tag{4.12}$$

Proof. We prove by induction on $s = S, \ldots, 1$ that

$$\left|\overline{\text{source}^{(m)}(Q_s^m)} \triangle I_s^*\right| \le 4\gamma t_s. \tag{4.13}$$

We start with the base case $s = S$. First we establish completeness by showing that there is an interval $Q^m \subset [|\mathsf{y}^{(m)}|]$ such that $d_{\text{edit}}(\mathsf{w}_S^*, \mathsf{y}_{Q^m}^{(m)}) \le 2\gamma t_S$. To this end, let $Q^m = \text{image}^{(m)}(I_S^*)$, and observe that this is an interval in $[|\mathsf{y}^{(m)}|]$. We have

$$d_{\text{edit}}(\mathsf{w}_S^*, \mathsf{y}_{Q^m}^{(m)}) \le d_{\text{edit}}(\mathsf{w}_S^*, \mathsf{x}_{I_S^*}) + d_{\text{edit}}(\mathsf{y}_{Q^m}^{(m)}, \mathsf{x}_{I_S^*}) < 2\gamma t_S,$$

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

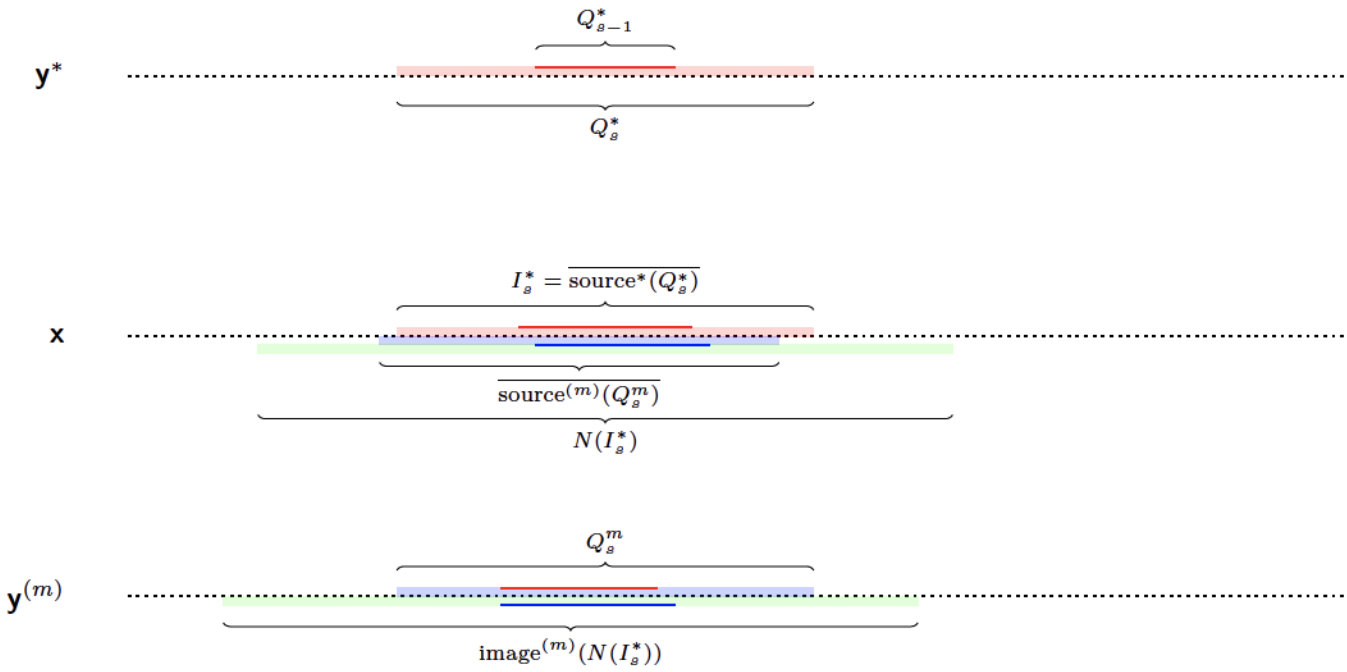

Figure 3: First stage of **Align**: The 3 red lines are the subwords $\mathsf{y}^*_{Q^*_{s-1}}$, $\mathsf{x}_{I^*_{s-1}=\overline{\text{source}^*(Q^*_{s-1})}}$, and $\mathsf{y}^{(m)}_{Q^m=\text{image}^{(m)}(I^*_{s-1})}$. The blue lines are the subwords $\mathsf{x}_{\overline{\text{source}^{(m)}(Q^m_{s-1})}}$, and $\mathsf{y}^{(m)}_{Q^m_{s-1}}$ found by **Align**. In the completeness argument, we have $Q^m = \text{image}^{(m)}(I^*_{s-1}) \subseteq Q^m_s$ because $I^*_{s-1} \subseteq I^*_s$ and $\left|\overline{\text{source}^{(m)}(Q^m_s)} \triangle I^*_s\right|$ is small. In the soundness argument, as $Q^m_{s-1} \subseteq Q^m_s$ and $\left|\overline{\text{source}^{(m)}(Q^m_s)} \triangle I^*_s\right|$ is small, we have $I := \text{source}^{(m)}(Q^m_{s-1}) \subseteq N(I^*_s)$, which implies $Q^m_{s-1} \subseteq \text{image}^{(m)}(N(I^*_s))$.

where we used (i) $|I^*_S| < (1+\gamma)t_S$ to upper bound the first edit distance by γt_S, and (v) $|Q^m| \geq |I^*_S| - \gamma t_S$ to upper bound the second edit distance by γt_S. Next we establish soundness by showing that any interval Q^m_S picked by **Align** satisfies (4.13). Let $I = \overline{\text{source}^{(m)}(Q^m_S)}$. Then we have

$$d_{\text{edit}}(\mathsf{x}_I, \mathsf{x}_{I^*_S}) \leq d_{\text{edit}}(\mathsf{x}_I, \mathsf{y}^{(m)}_{Q^m_S}) + d_{\text{edit}}(\mathsf{y}^{(m)}_{Q^m_S}, \mathsf{w}^*_S) + d_{\text{edit}}(\mathsf{w}^*_S, \mathsf{x}_{I^*_S}) < \gamma t_S + 2\gamma t_S + \gamma t_S = 4\gamma t_S. \tag{4.14}$$

To see that $d_{\text{edit}}(\mathsf{x}_I, \mathsf{y}^{(m)}_{Q^m_S}) \leq \gamma t_S$ we first notice that $|Q^m_S| \leq (1+2\gamma)t_S$ because $d_{\text{edit}}(\mathsf{y}^{(m)}_{Q^m_S}, \mathsf{w}^*_S) \leq 2\gamma t_S$. It then follows from item (v) that $|I| \leq |Q^m_S| + \gamma t_S$, which gives $d_{\text{edit}}(\mathsf{x}_I, \mathsf{y}^{(m)}_{Q^m_S}) \leq \gamma t_S$. For the last summand, as in the completeness argument (i) gives that $|I^*_S| < (1+\gamma)t_S$ and hence $d_{\text{edit}}(\mathsf{w}^*_S, \mathsf{x}_{I^*_S}) \leq \gamma t_S$. The soundness part then follows from (4.14) and (ii).

With the base case in hand, assume for the inductive step that (4.13) holds for some $s \in [2:S]$. We use this to prove it for $s-1$.

The completeness proof is similar to the base case: let $Q^m = \text{image}^{(m)}(I^*_{s-1})$. It follows from (4.13) on s that $Q^m \subseteq Q^m_s$. Then

$$d_{\text{edit}}(\mathsf{w}^*_{s-1}, \mathsf{y}^{(m)}_{Q^m}) \leq d_{\text{edit}}(\mathsf{w}^*_{s-1}, \mathsf{x}_{I^*_{s-1}}) + d_{\text{edit}}(\mathsf{y}^{(m)}_{Q^m}, \mathsf{x}_{I^*_{s-1}}) \leq 2\gamma t_{s-1},$$

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

where we used (i) and (vi).

The soundness argument is also similar to the base case. Let Q^m_{s-1} be the interval found by `Align` and let $I = \overline{\text{source}^{(m)}(Q^m_{s-1})}$. Then $Q^m_{s-1} \subseteq Q^m_s$ and it follows from (4.13) (and the definition of $N(\cdot)$) that

$$I = \overline{\text{source}^{(m)}(Q^m_{s-1})} \subseteq \overline{\text{source}^{(m)}(Q^m_s)} \subseteq N(I^*_s)$$

and thus, $Q^m_{s-1} \subseteq \text{image}(N(I^*_s))$. We also have $|Q^m_{s-1}| \le (1+2\gamma)t_{s-1}$ given that $y^{(m)}_{Q^m_{s-1}}$ has edit distance at most $2\gamma t_{s-1}$ from w^*_{s-1} (which is of length t_{s-1}). As a result, analogous to (4.14), we have

$$d_{\text{edit}}(\mathsf{x}_I, \mathsf{x}_{I^*_{s-1}}) \le d_{\text{edit}}(\mathsf{x}_I, \mathsf{y}^{(m)}_{Q^m_{s-1}}) + d_{\text{edit}}(\mathsf{y}^{(m)}_{Q^m_{s-1}}, \mathsf{w}^*_{s-1}) + d_{\text{edit}}(\mathsf{w}^*_{s-1}, \mathsf{x}_{I^*_{s-1}}) < 4\gamma t_{s-1}, \tag{4.15}$$

where $d_{\text{edit}}(\mathsf{x}_I, \mathsf{y}^{(m)}_{Q^m_{s-1}}) \le \gamma t_{s-1}$ follows as in the base case but now using (vi) rather than (v). The soundness follows from (4.15) and (iii), and the inductive step is completed. ∎

4.4.2 Second stage: Determining a consensus location close to $\text{source}^*(\ell^*)$. We finish the proof of Lemma 4.1 with the following lemma for the second stage of `Align`:

LEMMA 4.3. *Locations $\ell^{(1)}, \dots, \ell^{(M)}$ returned by* `Align` *satisfy the following two conditions: (A) at least 90% of $\text{source}^{(m)}(\ell^{(m)})$, $m \in [M]$, agree on the same location $i \in [n]$ and (B) the consensus location i satisfies*

$$\text{source}^*(\ell^*) - 2H \le i \le \text{source}^*(\ell^*). \tag{4.16}$$

Proof. It follows from Lemma 4.2 that for every $m \in [M]$, the interval Q^m_1 satisfies (4.12). Let G be the set of $m \in [M]$ such that $N(I^*_1) \cap D^{(m)} = \varnothing$; by (vii) we have that $|G| \ge 0.95M$. Let $I^m = \overline{\text{source}^{(m)}(Q^m_1)}$. It follows from (4.12) that every $m \in [M]$ satisfies $|I^m \triangle I^*_1| \le 4\gamma t_1$ and thus $I^m \subseteq N(I^*_1)$. As $N(I^*_1) \cap D^{(m)} = \varnothing$ we have $\mathsf{y}^{(m)}_{Q^m_1} = \mathsf{x}_{I^m}$ (and $\text{source}^{(m)}$ induces a bijection between Q^m_1 and I^m) for each $m \in G$. Moreover,

$$\left| \bigcap_{m \in G} I^m \right| \ge (1 - 2 \cdot 4\gamma) t_1 \ge 0.9 t_1,$$

which implies the completeness part: $\mathsf{w} := \mathsf{x}_{\cap_{m \in G} I^m}$ appears as a subword in at least 95% of $\mathsf{y}^{(m)}_{Q^m_1}$ (i.e., every $m \in G$) and has length at least $0.9t_1$.

Finally, we prove the soundness part: Assume that w is a string of length at least $0.9t_1$ and appears as a subword in at least 95% of $\mathsf{y}^{(m)}_{Q^m_1}$, $m = 1, \dots, M$. Then at least 90% of $m \in [M]$ have $m \in G$ and contain w as a subword. Let $G' \subseteq G$ denote the set of such $m \in G$. It follows from (iv) that $\text{source}^{(m)}(\ell^{(m)})$ are the same for all $m \in G'$, which consists of at least 90% of $[M]$. To prove (4.16), we take any $m \in G'$ and have that $\text{source}^{(m)}(\ell^{(m)})$ is at least $0.9t_1$ away from the right end of I^m; on the other hand, $\text{source}^*(\ell^*)$ is no more than $H + \gamma t_1$ away from the right end of I^*_1, using $|I^*_1| < (1+\gamma)t_1$. Given that the right ends of I^m and I^*_1 differ by no more than $4\gamma t_1$, we have $\text{source}^{(m)}(\ell^{(m)}) \le \text{source}^*(\ell^*)$. Similarly, $\text{source}^{(m)}(\ell^{(m)})$ is at least as large as the left end of I^m but $\text{source}^*(\ell^*)$ is similarly no more than $H + \gamma t_1$ away from the left end of I^*_1. Given that their left ends differ by no more than $4\gamma t_1$, we have that $\text{source}^{(m)}(\ell^{(m)}) - \text{source}^*(\ell^*) \le 4\gamma t_1 + H + \gamma t_1$, which is at most $2H$ by (4.4) and the definition of $t_1 = 2H + 1$. ∎

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

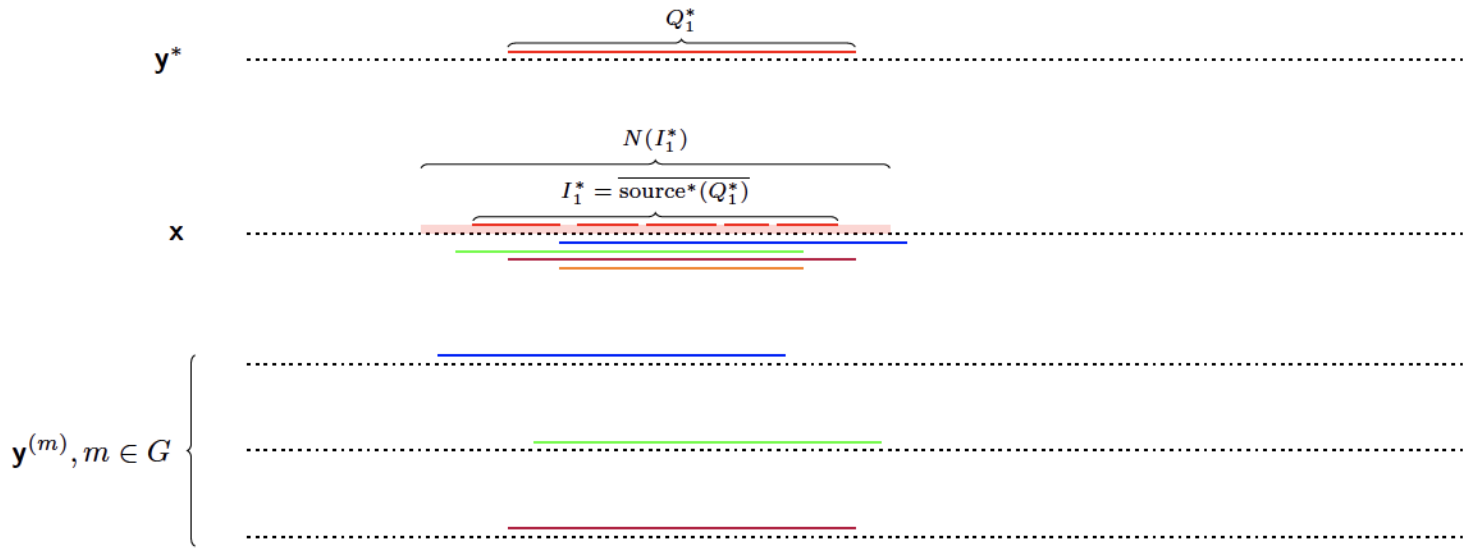

Figure 4: Second stage of **Align**: The red line is the subword $\mathsf{y}^*_{Q_1^*} = \mathsf{w}_1^*$ and it appears as the disconnected red segments in $\mathsf{x}_{\text{source}^*(Q_1^*)}$. The blue, green and purple lines are the subwords $\mathsf{y}^{(m)}_{Q_1^m}, m \in G$ and $\mathsf{x}_{I^m = \overline{\text{source}^{(m)}(Q_1^m)}}, m \in G$. Since $m \in G$, we have that $N(I_1^*) \cap D^{(m)} = \emptyset$, and so $\mathsf{y}^{(m)}_{Q_1^m} = \mathsf{x}_{I^m}$. The orange line is the common subword $\mathsf{x}_{\cap_{m\in G} I^m}$ that appears in the three subwords $\mathsf{y}^{(m)}_{Q_1^m}$.

5 The BMA algorithm and proof of Theorem 2.2

The goal of this section is to prove Theorem 2.2 which is restated below in full detail. As mentioned in the introduction, the **BMA** algorithm (which stands for Bitwise Majority Alignment) was first described and analyzed in [3]. Recall the two parameters δ (deletion rate) and M (number of traces) that satisfy (4.2) for some sufficiently large constant K, and the positive integer $H \le 2\log n$ given in (4.3). Let us set some parameters: define

$$L := 8H, \quad G := L/2, \quad \text{and } R := L2^{0.01L}. \tag{5.17}$$

We prove in this section that **BMA** reconstructs any source string $\tilde{\mathsf{x}} \in \{0,1\}^R$ exactly with M traces from $\text{Del}_\delta(\tilde{\mathsf{x}})$ with high probability[4], when $\tilde{\mathsf{x}}$ does not contain any "*long deserts.*" We now recall the definition of deserts from [8].

DEFINITION 5.1. *A string is said to be a* k-desert *for some* $k \ge 1$ *if it is the prefix of* s^∞ *for some string* $\mathsf{s} \in \{0,1\}^k$. *We say a string is a* long desert *if it is a* k*-desert of length* L *for some* $k \le G$.

The algorithm **BMA** is described in Algorithm 3, and its input consists of M traces $\mathsf{z}^{(1)}, \ldots, \mathsf{z}^{(M)}$ of $\tilde{\mathsf{x}}$.[5] We restate the main theorem of this section:

[4]We use $\tilde{\mathsf{x}}$ instead of x for the source string because later in the next section the role of $\tilde{\mathsf{x}}$ will be played by various different substrings of x of length R.

[5]Note that in **Reconstruct**, we run **BMA** on $M+1$ strings obtained from $\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$ instead of M strings. In its analysis, however, we pretend the string from y^* is not present and focus on what happens when running **BMA** on the other M strings only. This is why we focus on analyzing **BMA** running on M strings in this section.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

THEOREM 5.1. (DETAILED STATEMENT OF THEOREM 2.2) *Let* $\tilde{\mathsf{x}} \in \{0,1\}^R$ *be a string that does not contain any long desert. Let* $\mathsf{z}^{(1)}, \ldots, \mathsf{z}^{(M)} \sim \mathrm{Del}_\delta(\tilde{\mathsf{x}})$ *be independent traces. With probability at least* $1 - 2^{-H}$, `BMA` *returns exactly* $\tilde{\mathsf{x}}$ *and in every round* $t \in [R]$, *the majority is reached by at least* 90% *of the* M *bits that are pointed to in* $\mathsf{z}^{(1)}, \ldots, \mathsf{z}^{(M)}$.

Similar to the previous section, we write $D^{(m)} \subseteq [R]$ to denote the set of positions deleted in $\tilde{\mathsf{x}}$ to obtain $\mathsf{z}^{(m)}$, and use them to define $\mathrm{source}^{(m)}(\cdot)$; the only difference is that we set

$$\mathrm{source}^{(m)}\big(|\mathsf{z}^{(m)}| + \ell\big) = R + \ell$$

for every $m \in [M]$ and $\ell \geq 1$ since the pointer into $\mathsf{z}^{(m)}$ may move beyond $\mathsf{z}^{(m)}$ into the padded $*$'s (this can be viewed as adding $*$'s to the end of the unknown $\tilde{\mathsf{x}}$ which are never deleted and are where the $*$'s at the end of $\mathsf{z}^{(m)}$ come from). As introduced in Algorithm 3, let $\mathrm{current}^{(m)}(t)$ denote the location of the pointer of $\mathsf{z}^{(m)}$ at the start of the t-th step of `BMA`. In addition, let

$$\mathrm{last}^{(m)}(t) := \mathrm{source}^{(m)}\big(\mathrm{current}^{(m)}(t)\big) \quad \text{and} \quad \mathrm{dist}^{(m)}(t) := \mathrm{last}^{(m)}(t) - t.$$

Informally, $\mathrm{dist}^{(m)}(t)$ captures the number of positions in $\tilde{\mathsf{x}}$ that the pointer into $\mathsf{z}^{(m)}$ has gotten "ahead of where it should be."

To prove Theorem 5.1, it suffices to show that with high probability, for every $t^* \in [R]$ it holds that $\mathrm{dist}^{(m)}(t^*) = 0$ for at least 90% of $m \in [M]$. We will analyze the behavior of $\{\mathrm{dist}^{(m)}(\cdot)\}_{m\in[M]}$ over random traces $\mathsf{z}^{(1)}, \ldots, \mathsf{z}^{(M)} \sim \mathrm{Del}_\delta(\tilde{\mathsf{x}})$. The high level goal of the analysis is to show that for each $m \in [M]$, the sequence of random variables $\mathrm{dist}^{(m)}(1), \ldots, \mathrm{dist}^{(m)}(R)$ are nonnegative and *negatively drifted* (i.e., $\mathrm{dist}^{(m)}(t)$ tends to decrease as t grows).

Note that $\{\mathrm{dist}^{(m)}(\cdot)\}_{m\in[M]}$ are not independent over m due to correlations from the consensus $\mathbf{w}$ they produce together and thus this ensemble of random variables can be difficult to analyze. To ease the analysis we introduce a new set of random variables denoted by $\mathrm{dist}^{(m)}_{\mathsf{ideal}}(\cdot)$ for each $m \in [M]$. They are identical to $\mathrm{dist}^{(m)}(\cdot)$ with one key difference: in Step 4 of `BMA`, we set $\mathsf{w}_t = \tilde{\mathsf{x}}_t$ instead of the majority of the bits. Note that this makes $\{\mathrm{dist}^{(m)}_{\mathsf{ideal}}(\cdot)\}_{m\in[M]}$ independent over m and in fact, identically distributed. Hence, it suffices to analyze any one of them which we denote by $\mathrm{dist}_{\mathsf{ideal}}(\cdot)$ over the draw of $\mathbf{z} \sim \mathrm{Del}_\delta(\tilde{\mathsf{x}})$; we let $\mathbf{D}$ denote the set of positions deleted in $\mathbf{z}$. We define $\mathrm{current}_{\mathsf{ideal}}(\cdot)$ and $\mathrm{last}_{\mathsf{ideal}}(\cdot)$ similarly.

The following is a key technical lemma.

LEMMA 5.1. *For every* $t^* \in [R]$, $\mathrm{dist}_{\mathsf{ideal}}(t^*) = 0$ *with probability at least* $1 - 2\delta L$ *over* $\mathbf{z} \sim \mathrm{Del}_\delta(\tilde{\mathsf{x}})$.

We first use Lemma 5.1 to prove Theorem 5.1.

Proof. [Proof of Theorem 5.1 using Lemma 5.1.] Let $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(M)} \sim \mathrm{Del}_\delta(\tilde{\mathsf{x}})$. For each $t \in [R]$, let E_t be the event that

$$\sum_{m\in[M]} \mathbf{1}\left[\mathrm{dist}^{(m)}_{\mathsf{ideal}}(t) = 0\right] \geq 0.9M$$

for each $t \in [R]$. When the event E_t holds for every $t \in [R]$, we have $\mathbf{w} = \tilde{\mathsf{x}}$ by an induction on t. This implies that the two sets of random variables ($\mathrm{dist}^{(m)}(\cdot)$ and $\mathrm{dist}^{(m)}_{\mathsf{ideal}}(\cdot)$) are indeed identical, which in turn implies for every $t \in [R]$, $\mathrm{dist}^{(m)}(t) = 0$ for at least 90% of $m \in [M]$.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Algorithm 3: Algorithm BMA

Input: A multiset $\{\mathsf{z}^{(1)}, \ldots, \mathsf{z}^{(M)}\}$ of M strings

Output: Either ε, the empty string, or a string $\mathsf{w} \in \{0,1\}^R$

For each $m \in [M]$, concatenate R many $*$'s to the end of $\mathsf{z}^{(m)}$

Set $t = 1$ and $\text{current}^{(m)}(t) = 1$ for each $m \in [M]$

while $t \le R$ **do**

  Set $\mathsf{w}_t \in \{0,1,*\}$ to be the majority of the M symbols $\mathsf{z}^{(m)}_{\text{current}^{(m)}(t)}$, $m \in [M]$

  For each $m \in [M]$, set

$$\text{current}^{(m)}(t+1) = \begin{cases} \text{current}^{(m)}(t) + 1 & \text{if } \mathsf{z}^{(m)}_{\text{current}^{(m)}(t)} = \mathsf{w}_t \\ \text{current}^{(m)}(t) & \text{otherwise} \end{cases}$$

  Increment t

return w if w does not contain any $*$ (so $\mathsf{w} \in \{0,1\}^R$) or ε if w contains at least one $*$

Figure 5: The Algorithm BMA

As a result it suffices to understand the probability of E_t. Given that these random variables are independent, it follows from Lemma 5.1 and Equation (4.3) that for every $t \in [R]$:

$$\mathbf{Pr}[E_t] \ge 1 - 2^M \cdot (2\delta L)^{0.9M} \ge 1 - \left(\frac{48\delta M}{K} \log \frac{1}{\delta M}\right)^{\frac{0.9HK}{\log(1/(\delta M))}} \ge 1 - 2^{-0.45HK}.$$

where the last inequality uses

$$\frac{48\delta M}{K} \log\left(\frac{1}{\delta M}\right) \le \sqrt{\delta M}$$

which holds when K is sufficiently large. It follows from a union bound (using $R = L2^{0.01L}$) that

$$\mathbf{Pr}[E_t \text{ holds for all } t \in [R]] \ge 1 - R \cdot 2^{-0.45HK} \ge 1 - 2^{-H}$$

by setting K to be sufficiently large. This finishes the proof of the theorem. □

5.1 Proof of Lemma 5.1 In the rest of the section we prove Lemma 5.1. We start with three simple claims (which may also be found in [8]); these claims hold for any trace z (and deletions D):

CLAIM 5.1. *For each $t \in [R]$, letting b be the $(\text{last}_{\text{ideal}}(t))$-th bit of $\tilde{\mathsf{x}}$, we have*

1. *If $\tilde{\mathsf{x}}_t \ne b$, then $\text{dist}_{\text{ideal}}(t+1) = \text{dist}_{\text{ideal}}(t) - 1$.*
2. *If $\tilde{\mathsf{x}}_t = b$, then $\text{dist}_{\text{ideal}}(t+1) = \text{dist}_{\text{ideal}}(t) + \ell$, where ℓ is the nonnegative integer with*

$$\text{last}_{\text{ideal}}(t)+1, \ldots, \text{last}_{\text{ideal}}(t)+\ell \in D \quad \text{and} \quad \text{last}_{\text{ideal}}(t)+\ell+1 \notin D. \tag{5.18}$$

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Proof. The first item follows from the observation that $\text{current}_{\text{ideal}}(t+1) = \text{current}_{\text{ideal}}(t)$.

For the second item, we have $\text{current}_{\text{ideal}}(t+1) = \text{current}_{\text{ideal}}(t) + 1$. It now points to the next bit in $\mathsf{z}^{(m)}$, which is the bit of $\tilde{\mathsf{x}}$ indexed by $\text{last}_{\text{ideal}}(t) + \ell + 1$ with ℓ defined in (5.18). □

We next have the following two easy observations:

CLAIM 5.2. *We have* $\text{dist}_{\text{ideal}}(t) \geq 0$ *and* $\text{last}_{\text{ideal}}(t) \leq R+1$ *for every* $t \in [R]$.

Proof. The proof of the first part is by induction. Using Claim 5.1, $\text{dist}_{\text{ideal}}(t+1) \geq \text{dist}_{\text{ideal}}(t) - 1$. So if $\text{dist}_{\text{ideal}}(t) > 0$ then $\text{dist}_{\text{ideal}}(t+1) \geq 0$. Otherwise, if $\text{dist}_{\text{ideal}}(t) = 0$, we can apply the second item of Claim 5.1 to conclude that $\text{dist}_{\text{ideal}}(t+1) \geq 0$. For the second part note that once $\text{last}_{\text{ideal}}(t)$ reaches $R+1$ (i.e., $\text{current}_{\text{ideal}}(t)$ reaches $|\mathsf{y}^*|+1$), it cannot move further as x is a string in $\{0,1\}$ but the current bit in y^* is $*$. □

CLAIM 5.3. *If* $\text{dist}_{\text{ideal}}(t) = \cdots = \text{dist}_{\text{ideal}}(t+L) = k$ *for some* $k \in [G]$, *then* $\tilde{\mathsf{x}}$ *has a long desert.*

Proof. We have that $\tilde{\mathsf{x}}_{t+\ell}$ is equal to the $\text{last}_{\text{ideal}}(t+\ell)$-th bit of $\tilde{\mathsf{x}}$ for all $\ell \in [0 : L-1]$, and hence using Claim 5.1, we have $\tilde{\mathsf{x}}_{t+\ell} = \tilde{\mathsf{x}}_{t+k+\ell}$ for all $\ell \in [0 : L-1]$. The claim follows. □

We now start to prove Lemma 5.1. Let $t^* \in [R]$ be the round we consider in Lemma 5.1, with $t^* = t_0 + sL$ such that $t_0 \in \{0, 1, \ldots, L-1\}$ and $s \leq 2^{0.01L}$. We observe that

$$\text{dist}_{\text{ideal}}(t_0),\ \text{dist}_{\text{ideal}}(t_0+L),\ \ldots,\ \text{dist}_{\text{ideal}}(t_0+sL) \tag{5.19}$$

is a Markov process and look at how $\text{dist}_{\text{ideal}}(t_0 + (\ell+1)L)$ changes conditioned on $\text{dist}_{\text{ideal}}(t_0 + \ell L)$.

To this end, let us condition on $\text{dist}_{\text{ideal}}(t_0 + \ell L) = \Delta \geq 0$, so $\text{last}_{\text{ideal}}(t_0 + \ell L) = t_0 + \ell L + \Delta$. Conditioning on this, each bit of $\tilde{\mathsf{x}}$ after $t_0 + \ell L + \Delta$ is deleted independently and added to $\mathbf{D}$ with probability δ. Let $\boldsymbol{\beta}$ be the nonnegative random variable such that *either*

$$t_0 + \ell L + \Delta + \boldsymbol{\beta} \text{ is at most } R \text{ and does not belong to } \mathbf{D} \quad \text{and}$$
$$\{t_0 + \ell L + \Delta + 1, \ldots, t_0 + \ell L + \Delta + \boldsymbol{\beta}\} \cap \overline{\mathbf{D}} = L,$$

i.e., $t_0 + \ell L + \Delta + \boldsymbol{\beta}$ is the unique location in x that is the L-th undeleted position after $t_0 + \ell L + \Delta$, *or* $\boldsymbol{\beta}$ is chosen using $t_0 + \ell L + \Delta + \boldsymbol{\beta} = R+1$ if no such $\boldsymbol{\beta}$ exists. Since $\text{last}_{\text{ideal}}(\cdot)$ can move forward by at most L undeleted positions in steps $t_0 + \ell L + 1, \ldots, t_0 + (\ell+1)L$, it follows from the second part of Claim 5.2 that

$$\text{last}_{\text{ideal}}\big(t_0 + (\ell+1)L\big) \leq t_0 + \ell L + \Delta + \boldsymbol{\beta}$$

and thus, we have the following upper bound:

$$\text{dist}_{\text{ideal}}\big(t_0 + (\ell+1)L\big) \leq \text{dist}_{\text{ideal}}\big(t_0 + \ell L\big) + \boldsymbol{\beta} - L.$$

Moreover, when $\Delta \leq G$ and $\boldsymbol{\beta} \leq L$ (including $\boldsymbol{\beta} = L$), we have

$$\text{dist}_{\text{ideal}}\big(t_0 + (\ell+1)L\big) \leq \max\Big(\text{dist}_{\text{ideal}}\big(t_0 + \ell L\big) - 1, 0\Big).$$

It holds for $\boldsymbol{\beta} = L$ because otherwise by Claim 5.3, the subword of $\tilde{\mathsf{x}}$ in $[t_0 + \ell L : t_0 + (\ell+1)L - 1]$ would be a long desert, contradicting with our assumption that $\tilde{\mathsf{x}}$ has no long deserts.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

In the next portion of the analysis we relate this Markov process to a simpler one for which the transition probabilities are the same for all states. Note that β is distributed as

$$\min\bigl(R+1-(t_0+\ell L+\Delta),\beta^*\bigr)$$

where β^* denotes the sum of L i.i.d. geometric random variables with success probability $1-\delta$.[6] So β is stochastically dominated by β^*.[7] Let $\alpha := \delta L$, which can be made sufficiently small as $\delta L \le 8/K$ and K can be made sufficiently large. We use the following rough estimates for the probability of $\beta^* = L + c$ for $c = 0, 1, \ldots$: When $c = 0$ we have

$$\mathbf{Pr}\left[\beta^* = L\right] = (1-\delta)^L \ge 1 - \delta L = 1 - \alpha > \frac{1-2\alpha}{1-\alpha}. \tag{5.20}$$

(The reason for using the lower bound $(1-2\alpha)/(1-\alpha)$ in the second inequality will become clear soon.) When $c \ge 1$,

$$\mathbf{Pr}\left[\beta^* = L + c\right] = \binom{L+c-1}{c}\delta^c(1-\delta)^L \le \bigl(L\delta\bigr)^c = \alpha^c. \tag{5.21}$$

Inspired by these estimates, we introduce the following simpler Markov chain $\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_s \ge 0$, where (1) $\mathbf{X}_0$ is distributed the same as $\mathrm{dist}_{\mathsf{ideal}}(t_0)$, and (2) for each $\mathbf{X}_{\ell+1}$, if $\mathbf{X}_\ell > G$, then

$$\mathbf{X}_{\ell+1} = \begin{cases} \mathbf{X}_\ell & \text{with probability } (1-2\alpha)/(1-\alpha) \\ \mathbf{X}_\ell + c & \text{with probability } \alpha^c \text{ for each } c \ge 1 \end{cases}; \tag{5.22}$$

if $\mathbf{X}_\ell \le G$, then

$$\mathbf{X}_{\ell+1} = \begin{cases} \max(\mathbf{X}_\ell - 1, 0) & \text{with probability } (1-2\alpha)/(1-\alpha) \\ \mathbf{X}_\ell + c & \text{with probability } \alpha^c \text{ for each } c \ge 1 \end{cases}. \tag{5.23}$$

Note that the use of $(1-2\alpha)/(1-\alpha)$ makes sure that the probabilities sum to 1. Below we will analyze $\mathbf{X}_0, \mathbf{X}_1, ..., \mathbf{X}_s$ in lieu of Equation (5.19).

Lemma 5.1 follows directly by combining the following two claims:

CLAIM 5.4. $\mathbf{Pr}\bigl[\mathrm{dist}_{\mathsf{ideal}}(t^*) \ge c\bigr] \ge \mathbf{Pr}\bigl[\mathbf{X}_s \ge c\bigr]$ *for every* c.

CLAIM 5.5. $\mathbf{Pr}\left[\mathbf{X}_s = 0\right] \ge 1 - 2\alpha$.

Proof. [Proof of Claim 5.4] We prove by induction that for every $\ell \in [0:s]$, $\mathbf{X}_\ell$ stochastically dominates $\mathrm{dist}_{\mathsf{ideal}}(t_0 + \ell L)$. The basis is trivial since $\mathbf{X}_0$ has the same distribution as $\mathrm{dist}_{\mathsf{ideal}}(t_0)$.

To prove the case with $\mathbf{X}_{\ell+1}$ using $\mathbf{X}_\ell$, we make two simple observations:

1. First, for any $a \ge b$, the distribution of $\mathbf{X}_{\ell+1}$ conditioned on $\mathbf{X}_\ell = a$ stochastically dominates the distribution of $\mathbf{X}_{\ell+1}$ conditioned on $\mathbf{X}_\ell = b$.

[6] We use the version of the geometric distribution for which the outcome of a draw is the total number of trials up to and including the first success (hence the support is $\{1, 2, 3, \ldots\}$).

[7] Recall that a random variable $\mathbf{X}$ is said to stochastically dominate $\mathbf{Y}$ if $\mathbf{Pr}[\mathbf{X} \ge a] \ge \mathbf{Pr}[\mathbf{Y} \ge a]$ for all a.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

2. Next for any a, it follows from Equations (5.20) and (5.21) that the distribution of $\mathbf{X}_{\ell+1}$ conditioned on $\mathbf{X}_\ell = a$ stochastically dominates that of $\mathrm{dist}_{\mathsf{ideal}}(t_0 + (\ell+1)L)$ conditioned on $\mathrm{dist}_{\mathsf{ideal}}(t_0 + \ell L) = a$.

It follows from these two observations, as well as the inductive hypothesis on ℓ, that $\mathbf{X}_{\ell+1}$ stochastically dominates $\mathrm{dist}_{\mathsf{ideal}}(t_0 + (\ell+1)H)$. This finishes the proof of the claim. □

Proof. [Proof of Claim 5.5] We prove by induction on $\ell = 0, 1, \ldots, s$ that the distribution of $\mathbf{X}_\ell$ satisfies

$$\mathbf{Pr}[\mathbf{X}_\ell = c] \le \frac{\alpha}{1-2\alpha} \cdot \left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{c-1}, \qquad \text{for every } c \in [G]; \tag{5.24}$$

$$\mathbf{Pr}[\mathbf{X}_\ell > G] \le (\ell+1) \cdot \frac{\alpha}{1-\alpha}\left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{G}. \tag{5.25}$$

Before working on the induction, we have from these two items that

$$\begin{aligned}
\mathbf{Pr}[\mathbf{X}_s = 0] &\ge 1 - \frac{\alpha}{1-2\alpha} \cdot \sum_{a \ge 0}\left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{a} - (s+1)\cdot\frac{\alpha}{1-\alpha}\left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{G} \\
&\ge 1 - \frac{\alpha}{1-4\alpha+2\alpha^2} - \frac{\alpha}{2} \\
&\ge 1 - 2\alpha.
\end{aligned}$$

The second inequality used $s \le 2^{0.01L}$, $G = L/2$ and the fact that α can be made sufficiently small. The last inequality also used that α is sufficiently small. We work on the induction below.

For the base case $\mathbf{X}_0$, recall that this random variable has the same distribution as $\mathrm{dist}_{\mathsf{ideal}}(t_0)$, and hence we have for each $c \ge 1$,

$$\mathbf{Pr}[\mathrm{dist}_{\mathsf{ideal}}(t_0) = c] \le \sum_{a \ge c}\binom{t_0 + c - 1}{a}\delta^a \le \sum_{a \ge c}(L\delta)^a \le \frac{\alpha^c}{1-2\alpha}\cdot\left(\frac{2(1-\alpha)}{1-2\alpha}\right)^{c-1},$$

(where the second inequality above uses $t_0 \le L - 1$ and the third uses that α is sufficiently small) and also

$$\mathbf{Pr}[\mathrm{dist}_{\mathsf{ideal}}(t_0) > G] \le \sum_{c \ge G+1}\sum_{a \ge c}\binom{t_0 + c - 1}{a}\delta^a \le \frac{\alpha^{G+1}}{(1-\alpha)^2} \le \frac{\alpha}{1-\alpha}\left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{G}.$$

For the induction, we assume the statement holds for ℓ and use it to prove the case with $\ell + 1$.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Using Equation (5.24), for every $c \in [G-1]$, we have

$$\begin{aligned}
\mathbf{Pr}[\mathbf{X}_{\ell+1} = c] &\le \sum_{a=0}^{c-1} \alpha^{c-a} \cdot \mathbf{Pr}[\mathbf{X}_\ell = a] + \frac{1-2\alpha}{1-\alpha} \cdot \mathbf{Pr}[\mathbf{X}_\ell = c+1] && \text{(by Equation (5.23))} \\
&\le \alpha^c + \sum_{a=1}^{c-1} \alpha^{c-a} \frac{\alpha}{1-2\alpha} \left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{a-1} + \frac{1-2\alpha}{1-\alpha} \frac{\alpha}{1-2\alpha} \left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{c} \\
&= \alpha^c + \frac{\alpha^c}{1-2\alpha} \sum_{a=1}^{c-1} \left(\frac{2(1-\alpha)}{1-2\alpha}\right)^{a-1} + \frac{\alpha}{1-\alpha} \left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{c} && (5.26) \\
&= \alpha^c \left(1 + \frac{1}{1-2\alpha} \frac{\left(\frac{2(1-\alpha)}{1-2\alpha}\right)^{c-1} - 1}{\frac{2(1-\alpha)}{1-2\alpha} - 1} + \frac{\alpha}{1-\alpha} \left(\frac{2(1-\alpha)}{1-2\alpha}\right)^{c}\right) \\
&= \alpha^c \left(\left(\frac{2(1-\alpha)}{1-2\alpha}\right)^{c-1} + \frac{\alpha}{1-\alpha} \left(\frac{2(1-\alpha)}{1-2\alpha}\right)^{c}\right) && (5.27) \\
&= \frac{\alpha}{1-2\alpha} \left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{c-1}.
\end{aligned}$$

The proof for $c = G$ is similar except that we do not have the term for $\mathbf{X}_\ell = G+1$.

Finally using Equations (5.24) and (5.25) we have

$$\begin{aligned}
\text{(by definition of } \mathbf{X}.) \quad \mathbf{Pr}[\mathbf{X}_{\ell+1} > G] &\le \mathbf{Pr}[\mathbf{X}_\ell > G] + \sum_{b>G} \sum_{a=0}^{G} \alpha^{b-a} \cdot \mathbf{Pr}[\mathbf{X}_\ell = a] \\
&\le \mathbf{Pr}[\mathbf{X}_\ell > G] + \frac{1}{1-\alpha} \sum_{a=0}^{G} \alpha^{G+1-a} \cdot \mathbf{Pr}[\mathbf{X}_\ell = a] \\
\text{(by Equation (5.24))} \quad &\le \mathbf{Pr}[\mathbf{X}_\ell > G] + \frac{\alpha^{G+1}}{1-\alpha} \left(1 + \frac{1}{1-2\alpha} \sum_{a=1}^{G} \left(\frac{2(1-\alpha)}{1-2\alpha}\right)^{a-1}\right) \\
&= \mathbf{Pr}[\mathbf{X}_\ell > G] + \frac{\alpha}{1-\alpha} \left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{G} && (5.28) \\
\text{(by Equation (5.25))} \quad &\le (\ell+2) \cdot \frac{\alpha}{1-\alpha} \left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{G},
\end{aligned}$$

where Equation (5.28) used similar derivation between Equations (5.26) and (5.27) earlier. This finishes the induction and the proof of the claim. □

6 Main Algorithm

Let δ and M be two parameters that satisfy Equations (4.2) and (4.3). Recall the following parameters used in our analysis of `Align` and `BMA`:

$$\tau = 500, \quad H = \frac{M}{K} \log\left(\frac{1}{\delta M}\right) \le 2 \log n, \quad L = 8H, \quad G = L/2 = 4H \quad \text{and} \quad R = L 2^{0.01L}.$$

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Algorithm 4: `Reconstruct`

Input: A positive integer n and a tuple of $(M+1)$ strings $\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$

Output: A binary string w

Set $\ell^* = 5\tau \log n$ and $\mathsf{w} = \varepsilon$

while $\ell^* \le |\mathsf{y}^*| - R$ *and* $\ell^* \le |\mathsf{y}^*| - 5\tau \log n$ **do**

  Run $\texttt{Align}(\ell^*, \mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)})$ to obtain a tuple of locations $(\ell^{(1)}, \ldots, \ell^{(M)})$

  Run $\texttt{BMA}(\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}; \ell^*, \ell^{(1)}, \ldots, \ell^{(m)})$ to obtain a binary string (ε or in $\{0,1\}^R$)

  Concatenate the string returned by `BMA` to the end of w

  Set ℓ^* to be the final pointer of y^* in the run of `BMA` above and increment it

return w.

Figure 6: The Algorithm `Reconstruct`

(Note that $L \le 16 \log n$, $G \le 8 \log n$ and $R \le O(n^{0.16} \log n)$.) Our main (deterministic) algorithm `Reconstruct` is described in Figure 6, where we use

$$\texttt{BMA}\big(\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}; \ell^*, \ell^{(1)}, \ldots, \ell^{(M)}\big)$$

to denote running `BMA` on the suffix of y^* starting at location ℓ^* and the suffix of each $\mathsf{y}^{(m)}$ starting at location $\ell^{(m)}$. Recall that `BMA` either returns the empty string ε or a string $\mathsf{w} \in \{0,1\}^R$.

We prove the following theorem about the performance of `Reconstruct`:

THEOREM 6.1. *Let* $\mathbf{x} \sim \{0,1\}^n$ *and* $\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)} \sim \mathrm{Del}_\delta(x)$. *With probability at least* $1 - 1/n$, `Reconstruct` *on* $\mathbf{y}^*, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ *returns a string* $\mathbf{w}$ *with edit distance at most* $2^{-0.01H} n$ *from* $\mathbf{x}$.

Similar to the analysis of `Align` in Section 4, we divide the analysis of `Reconstruct` into two parts: In Section 6.1, we begin by describing some good events over the randomness of $\mathbf{x}$, $\mathbf{D}^*$ and $\mathbf{D}^{(m)} : m \in [M]$, and show that these events happen with probability at least $1 - 1/n$. The rest of the analysis in Section 6.2 will be entirely deterministic. We show that `Reconstruct` on $\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$ must return a string w with small edit distance from x when all the events described in Section 6.1 hold.

6.1 Probabilistic Analysis We start by showing that for $\mathbf{x} \sim \{0,1\}^n$, the number of length-$(2R)$ subwords of $\mathbf{x}$ that contain at least one long desert is small.

LEMMA 6.1. *With probability at least* $1 - \exp(-n^{0.1})$ *over* $\mathbf{x} \sim \{0,1\}^n$, *the number of* $i \in [n - 2R + 1]$ *such that* $\mathbf{x}_{[i:i+2R-1]}$ *has at least one long desert is at most* $2^{-0.13H} n$.

Proof. Fix any $k \le G$. The probability of a random length-L string being a k-desert is at most

$$2^k \cdot \frac{1}{2^L}.$$

As a result, the probability of a random length-L string being a long desert is at most

$$\sum_{k=1}^{G} \frac{2^k}{2^L} \le 2^{-0.4L}.$$

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Consider the number of length-L subwords in $\mathbf{x} \sim \{0,1\}^n$ that are long deserts. Given that changing each bit can only change the number by no more than L, it follows from McDiarmid's inequality (Theorem 3.1) that with probability at least $1 - \exp(-n^{0.1})$, this number is at most

$$2^{-0.4L}n + O(n^{0.55}L) \le 2^{-0.22H}n$$

using $H \le 2\log n$. When this happens, the number of indices i we care about in the statement of the lemma is at most $(4R - L) \cdot 2^{-0.22H}n \le 2^{-0.13H}n$ using $R = L2^{0.01L}$ and $L = 8H$. ∎

We modify the definition of $\text{image}^{(m)}$ so that it is well-defined for every $i \in [n]$: $\text{image}^{(m)}(i)$ is the smallest location $\ell \in [|\mathsf{y}^{(m)}|]$ such that $\text{source}^{(m)}(\ell) \ge i$, or set $\text{image}^{(m)}(i) = |\mathsf{y}^{(m)}| + 1$ if no such ℓ exists. (Note that when the latter happens, the suffix of $\mathsf{y}^{(m)}$ starting at $|\mathsf{y}^{(m)}| + 1$ is the empty string ε.) We say `BMA` *succeeds on* $\mathsf{y}^{(1)}, \dots, \mathsf{y}^{(M)}$ *at location* $i \in [n-R+1]$ *of* x if running `BMA` on $\mathsf{y}^{(1)}, \dots, \mathsf{y}^{(M)}$ starting at $\text{image}^{(1)}(i), \dots, \text{image}^{(M)}(i)$ returns exactly the R-bit string $\mathsf{x}_{[i:i+R-1]}$ and moreover, the consensus is achieved by at least 90% of strings in every round of `BMA`'s execution.

The following lemma is a direct corollary of Lemma 6.1 and Theorem 5.1:

LEMMA 6.2. *Let* $\mathbf{x} \sim \{0,1\}^n$ *and* $\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(M)} \sim \text{Del}_\delta(\mathbf{x})$. *With probability at least* $1 - 2\exp(-n^{0.1})$, `BMA` *succeeds on all but at most* $2^{-0.1H}n$ *locations* $i \in [n - R + 1]$ *in* $\mathbf{x}$.

Proof. It follows from Lemma 6.1 that with probability at least $1-\exp(-n^{0.1})$, a random $\mathbf{x} \sim \{0,1\}^n$ has no more than $2^{-0.13H}n$ length-$(2R)$ subwords that contain at least one long desert. Let x be such a string and fix any location i such that $x_{[i:i+R-1]}$ contains no long deserts. If the conclusion of Theorem 5.1 holds on $\tilde{\mathsf{x}} := \mathsf{x}_{[i:i+R-1]}$ over subwords of $\mathbf{y}^{(m)}$ that originate from $\mathsf{x}_{[i:i+R-1]}$, then `BMA` must succeed at location i. It follows from Theorem 5.1 that this happens with probability at least $1 - 2^{-H}$.

We will apply McDiarmid's inequality. Note that each of the nM independent random variables (each of which indicates whether or not a bit of x is included in $\mathbf{D}^{(m)}$) can only change the number we care about (i.e. the number of locations i such that the conclusion of Theorem 5.1 holds on $\tilde{\mathsf{x}} := \mathsf{x}_{[i:i+R-1]}$ over subwords of $\mathbf{y}^{(m)}$ that originate from $\mathsf{x}_{[i:i+R-1]}$) by no more than R. It follows that with probability at least $1 - \exp(-n^{0.1})$, `BMA` succeeds on all except $2^{-0.1H}n$ many locations i in x such that $\mathsf{x}_{[i:i+R-1]}$ has no long deserts. The lemma follows. ∎

LEMMA 6.3. $\mathbf{D}^* \sim \mathcal{D}$ *satisfies the following two properties with probability at least* $1 - 1/n^2$*: (note that* $|\mathbf{y}^*| = n - |\mathbf{D}^*|$)

1. $\text{source}^*(5\tau \log n) = O(\log n)$ *and*

$$\min\left(\text{source}^*(|\mathbf{y}^*| - R), \text{source}^*(|\mathbf{y}^*| - 5\tau\log n)\right) \ge n - (2R + O(\log n));$$

2. *There are at most* $2^{-0.2H}n$ *values of* $i \in [n]$ *such that* $\text{source}^*(\ell) < i < \text{source}^*(\ell + 1)$ *for some* $\ell \in [|\mathbf{y}^*| - 1]$ *for which* $\text{source}^*(\ell + 1) - \text{source}^*(\ell) \ge 2H$.

Proof. The first part follows from a Chernoff bound.

For the second part, note that for i to be counted, it must be the case that either $[i - H + 1 : i] \subseteq \mathbf{D}^*$ or $[i : i + H - 1] \subseteq \mathbf{D}^*$, which occurs with probability at most $2 \cdot \delta^H \le 2^{-H}$. The second part then follows from an application of McDiarmid's inequality. ∎

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

To describe our final condition on $\mathbf{x} \sim \{0,1\}^n$ and $\mathbf{D}^* \sim \mathcal{D}$, we introduce a procedure which we call $\texttt{BMA}^*$. $\texttt{BMA}^*$ takes as input a string y^*, a location ℓ^* such that $|\mathsf{y}^*| \geq \ell^* + R$, and a reference string $\mathsf{z} \in \{0,1\}^R$. Let $\text{current}^*(1) = \ell^*$. The $\texttt{BMA}^*$ procedure repeats the following for R rounds: In the t-th round, $t \in [R]$, $\texttt{BMA}^*$ compares z_t with $\mathsf{y}^*_{\text{current}^*(t)}$ and set $\text{current}^*(t+1) = \text{current}^*(t) + 1$ if they match and $\text{current}^*(t+1) = \text{current}^*(t)$ if they do not match. For each t we also define $\text{last}^*(t) = \text{source}^*\big(\text{current}^*(t)\big)$. After the final ($R$-th) round $\texttt{BMA}^*$ outputs

$$\text{last}^*(R+1) := \text{source}^*\big(\text{current}^*(R+1)\big).$$

Intuitively, $\texttt{BMA}^*$ outputs the final location of the pointer ℓ^* after a successful run of $\texttt{BMA}$. Now we state the final condition on $(\mathbf{x}, \mathbf{D}^*)$. We say $\texttt{BMA}^*$ *succeeds on a source string* $\mathsf{x} \in \{0,1\}^n$ *with respect to* y^* if for all but at most $2^{-0.1H}n$ many $\ell^* \in [|\mathsf{y}^*| - R]$, we have

$$\texttt{BMA}^*\big(\mathsf{y}^*, \ell^*, \mathsf{x}_{[i:i+R-1]}\big) \leq i + R + G, \quad \text{for every } i \in [\text{source}^*(\ell^*) - 2H : \text{source}^*(\ell^*)]. \tag{6.29}$$

Note that from an argument similar to the proof of Claim 5.2 we always have

$$\texttt{BMA}^*\big(\mathsf{y}^*, \ell^*, \mathsf{x}_{[i:i+R-1]}\big) \geq i + R, \tag{6.30}$$

so intuitively, $\texttt{BMA}^*$ succeeds on x with respect to y^* if for almost every ℓ^*, the output of $\texttt{BMA}^*$ on input $(\mathsf{y}^*, \ell^*, \mathsf{x}_{[i:i+R-1]})$ is close to the "right value" $i + R$.

LEMMA 6.4. *With probability at least* $1 - 2\exp\big(-n^{0.1}\big)$ *over* $\mathbf{x} \sim \{0,1\}^n$ *and* $\mathbf{D}^* \sim \mathcal{D}$, $\texttt{BMA}^*$ *succeeds on* $\mathbf{x}$ *with respect to* $\mathbf{y}^*$.

Proof. We will show that with high probability, the number of pairs ℓ^* and $i \in [\text{source}^*(\ell^*) - 2H : \text{source}^*(\ell^*)]$ that violate Equation (6.29) is at most $2^{-0.1H}n$. First, note that it follows from Lemma 6.1 that with probability at least $1 - \exp\big(-n^{0.1}\big)$, the number of length-$(2R)$ subwords of $\mathbf{x} \sim \{0,1\}^n$ that have at least one long desert is at most $2^{-0.13H}n$. Fixing such an $\mathsf{x} \in \{0,1\}^n$ in the rest of the proof, we show that with probability at least $1 - \exp\big(-n^{0.1}\big)$ over $\mathbf{D}^* \sim \mathcal{D}$, $\texttt{BMA}^*$ succeeds on x with respect to $\mathbf{y}^*$, from which the lemma follows.

To this end, we consider an ℓ^* and an i in the window such that $\mathsf{x}_{[i:i+2R-1]}$ has no long deserts. (By doing this we skipped no more than $2H \cdot 2^{-0.13H}n$ many pairs, which is much smaller than our target of $2^{-0.1H}n$.) The idea of the argument is to upper-bound the probability that Equation (6.29) is violated at ℓ^* and i over $\mathbf{D}^* \sim \mathcal{D}$, and then apply McDiarmid's inequality to finish the proof.

We note that whether Equation (6.29) holds or not only depends on the $[i : i + R + G]$ window of x, because if $\text{last}^*(t)$ ever becomes larger than $i + R + G$ then Equation (6.29) is already violated. Since $R + G < 2R$, this subword $\mathsf{x}_{[i:i+R+G]}$ of x has no long deserts. Similar to the analysis of $\texttt{BMA}$, we define $\text{dist}^*(t)$ for each $t = 1, \ldots, R, R+1$ as

$$\text{dist}^*(t) := \text{last}^*(t) - t - (i-1),$$

where $\text{last}^*(t)$ is from the execution of $\texttt{BMA}^*$. Note that

$$\text{dist}^*(1) = \text{last}^*(1) - i = \text{source}^*(\ell^*) - i \in [0, 2H].$$

So the condition (6.29) can be restated as $\text{dist}^*(R+1) \leq G$.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Recall the random variables $\{\mathrm{dist}_{\mathsf{ideal}}(t)\}_{t\in[R+1]}$ defined in Section 5. We claim that the random variable $\mathrm{dist}_{\mathsf{ideal}}(t)+(\mathrm{source}^*(\ell^*)-i)$ stochastically dominates $\mathrm{dist}^*(t)$ for every $t\in[R+1]$. To see this, observe that for every $c\geq 0$,

$$\mathbf{Pr}\big[\mathrm{dist}_{\mathsf{ideal}}(1)\leq c\big]\leq \mathbf{Pr}\big[\mathrm{dist}^*(1)\leq c+(\mathrm{source}^*(\ell^*)-i)\big]=1.$$

Moreover, conditioned on $\mathrm{dist}_{\mathsf{ideal}}(t)+(\mathrm{source}^*(\ell^*)-i)=\mathrm{dist}^*(t)$, the random variable $\mathrm{dist}_{\mathsf{ideal}}(t+1)+(\mathrm{source}^*(\ell^*)-i)$ stochastically dominates $\mathrm{dist}^*(t+1)$. (They are identical when $\mathrm{dist}_{\mathsf{ideal}}(t)\neq 0$.) Also, for any $a\geq b$, we have that $\mathrm{dist}^*(t+1)$ conditioned on $\mathrm{dist}^*(t)=a$ stochastically dominates $\mathrm{dist}^*(t+1)$ conditioned on $\mathrm{dist}^*(t)=b$.

It follows from Claim 5.4 and Equation (5.25) that

$$\begin{aligned}\mathbf{Pr}[\mathrm{dist}^*(R+1)>G]&\leq \mathbf{Pr}[\mathrm{dist}_{\mathsf{ideal}}(R+1)>G-(\mathrm{source}^*(\ell^*)-i)]\\&\leq \mathbf{Pr}[\mathrm{dist}_{\mathsf{ideal}}(R+1)>G-2H]\\&\leq (s+1)\cdot\frac{\alpha}{1-\alpha}\left(\frac{2\alpha(1-\alpha)}{1-2\alpha}\right)^{2H}\\&\leq 2^{-H},\end{aligned}$$

where we used $s=R/L=2^{0.01L}$, and $G=4H$, and α sufficiently small.

We now apply McDiarmid's inequality. The expected number of pairs ℓ^* and i such that x in $[i:i+2R-1]$ has no long deserts and Equation (6.29) is violated is at most $2^{-H}\cdot 2Hn$. Since Equation (6.29) only depends on deletions in the window of $[i:i+R+4H]$, each random variable can only change the number we care about above by $O(R)$. It follows from McDiarmid's inequality that with probability at least $1-\exp(-n^{0.1})$, the number of such pairs is at most $2^{-0.15H}n$. When this happens, the total number of pairs of ℓ^* and i that violate Equation (6.29) (including those i's in Lemma 6.1 that have at least one long desert in $[i:i+2R]$ of x) is at most

$$(2H+1)\cdot 2^{-0.13H}n+2^{-0.15H}n\leq 2^{-0.1H}n.$$

This finishes the proof of the lemma. □

We are now ready to present the list of conditions on $\mathbf{x},\mathbf{y}^*,\mathbf{y}^{(1)},\ldots,\mathbf{y}^{(M)}$:

COROLLARY 6.1. *With probability at least* $1-1/n$, $\mathbf{x}\sim\{0,1\}^n$ *and* $\mathbf{y}^*,\mathbf{y}^{(1)},\ldots,\mathbf{y}^{(M)}\sim\mathrm{Del}_\delta(\mathbf{x})$ *satisfy all conditions stated in Theorem 4.1, Lemma 6.2, Lemma 6.3 and Lemma 6.4.*

6.2 Deterministic Analysis We prove that when $\mathsf{x},\mathsf{y}^*,\mathsf{y}^{(1)},\ldots,\mathsf{y}^{(M)}$ satisfy all conditions in Theorem 4.1, Lemma 6.2, Lemma 6.3, and Lemma 6.4, the string w that `Reconstruct` returns on $\mathsf{y}^*,\mathsf{y}^{(1)},\ldots,\mathsf{y}^{(M)}$ must have edit distance at most $2^{-0.01H}n$ from x. This, together with Corollary 6.1, finishes the proof of Theorem 6.1.

We start the proof with some notation. Let P be the following interval of locations of y^*:

$$P=\Big[5\tau\log n:|\mathsf{y}^*|-\max\big(5\tau\log n,R\big)\Big].$$

Let Q be the set of locations $\ell^*\in P$ such that all three conditions below hold:

(i) The output $(\ell^{(1)},\ldots,\ell^{(M)})$ of $\mathtt{Align}(\ell^*,\mathsf{y}^*,\mathsf{y}^{(1)},\ldots,\mathsf{y}^{(M)})$ satisfies

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

(a) At least 90% of $\text{source}^{(m)}(\ell^{(m)})$, $m \in [M]$, agree on the same location $i^* \in [n]$, and
(b) Their consensus location i^* satisfies $i^* \in [\text{source}^*(\ell^*) - 2H : \text{source}^*(\ell^*)]$;

(ii) Running $\texttt{BMA}$ on $\mathsf{y}^*, \mathsf{y}^{(1)}, \ldots, \mathsf{y}^{(M)}$ starting at $\text{image}^*(\mathsf{y}^*), \text{image}^{(1)}(i^*), \ldots, \text{image}^{(M)}(i^*)$ returns $\mathsf{x}_{[i^*:i^*+R-1]}$ and the consensus is achieved by at least 90% of strings in every round;

(iii) The pair ℓ^* and i^* satisfies $\texttt{BMA}^*(\mathsf{y}^*, \ell^*, \mathsf{x}_{[i^*:i^*+R-1]}) \le i^* + R + G$.

Using conditions from Theorem 4.1 (for (i)) Lemma 6.2 (for (ii)), and Lemma 6.4 (for (iii)), we have $|P \setminus Q| \le O(H2^{-0.1H}n)$. If $\texttt{Reconstruct}$ uses a value ℓ^* that belongs to Q in an execution of the main loop, the string concatenated to w during this execution of the loop must be $\mathsf{x}_{[i^*:i^*+R-1]}$ for some $i^* \in [\text{source}^*(\ell^*) - 2H : \text{source}^*(\ell^*)]$. Furthermore, before incrementing ℓ^* at the end of this loop, we have from (iii) that $\text{source}^*(\ell^*) \le i^* + R + G$. In the rest of the proof, we write $\ell_1^*, \ell_2^*, \ldots \in P$ to denote the locations of y^* that are used in each execution of the main loop of $\texttt{Reconstruct}$. We use Good to denote the set of k such that $\ell_k^* \in Q$ and Bad to denote the set of k such that $\ell_k^* \notin Q$. For each k, we write w^k to denote the binary string concatenated to the end of w in Step 5 of the k-th execution of the loop, so the output string of $\texttt{Reconstruct}$ is $\mathsf{w} = \mathsf{w}^1\mathsf{w}^2\cdots$. Our analysis bounding the edit distance between w and x will proceed in three steps.

First step: In the first step we delete from w every w^k with $\ell_k^* \in \mathsf{Bad}$. Let w' denote the concatenation of all and only the w^k for which $\ell_k^* \in \mathsf{Good}$. We have that the edit distance between w and w' is at most

$$|\mathsf{Bad}| \cdot R \le |P \setminus Q| \cdot R \le O(H2^{-0.1H}n) \cdot R \le 2^{-0.011H}n,$$

where for the last step we recall that $R = 8H2^{0.08H}$.

Second step: After the first step w' is a concatenation of subwords of x but because these subwords are not necessarily disjoint w' is not necessarily a subsequence of x yet. In the second step we delete some bits of w' to obtain a subsequence of x. For each $k \in \mathsf{Good}$, recall that $i_k^* \in [\text{source}^*(\ell_k^*) - 2H : \text{source}^*(\ell_k^*)]$ and $\mathsf{w}^k = \mathsf{x}_{[i_k^*:i_k^*+R-1]}$, and that w' is the concatenation of w^k across all $k \in \mathsf{Good}$. For any two consecutive $k' < k$ in Good, we also have $\text{source}^*(\ell_k^*) > i_{k'}^* + R$ using Equation (6.30) and so the windows $[\text{source}^*(\ell_k^*) : i_k^* + R - 1]$ are disjoint and thus, defining w'' to be the concatenation of the subwords $\mathsf{x}_{[\text{source}^*(\ell_k^*):i_k^*+R-1]}$ of x, we get that w'' is a subsequence of x. To bound the edit distance between w' and w'' we note that each new window $[\text{source}^*(\ell_k^*) : i_k^* + R - 1]$ can be obtained from $[i_k^* : i_k^* + R - 1]$ by deleting no more than $2H$ indices at the beginning. Using

$$n = |\mathsf{x}| \ge |\mathsf{w}''| \ge |\mathsf{w}'| - |\mathsf{Good}| \cdot 2H = |\mathsf{Good}|\,(R - 2H) \quad \text{and} \quad 2H < R/2,$$

we have that $|\mathsf{Good}| \le 2n/R$ and thus, the edit distance between w' and w'' is at most

$$|\mathsf{Good}| \cdot 2H \le \frac{2n}{R} \cdot 2H \le 2^{-0.07H}n.$$

Third step: Given that w'' (the concatenation of subwords of x in $[i_k^{**} : i_k^* + R - 1]$ for each $k \in \mathsf{Good}$) is a subsequence of x, to bound its edit distance from x it suffices to bound the number of $j \in [n]$ such that $j \notin [i_k^{**} : i_k^* + R - 1]$ for any $k \in \mathsf{Good}$. The following two cases cover every such $j \in [n]$:

1. $j < \text{source}^*(5\tau \log n)$ or $j > \text{source}^*(|\mathsf{y}^*| - \max(5\tau \log n, R))$. There are only $O(R + \log n)$ many such j by the first part of Lemma 6.3.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

2. Otherwise, there is a unique k-th loop such that $\text{source}^*(\ell_k^*) \le j < \text{source}^*(\ell_{k+1}^*)$.

We split the second case further into two cases: $k \in \mathsf{Good}$ or $k \in \mathsf{Bad}$.

We start with the case when $k \in \mathsf{Bad}$ and bound the total number of x_i skipped. In this loop, BMA starts with location ℓ_k^* of y^* and ends at location $\ell_{k+1}^* - 1$. Given that BMA only has R rounds we have $\ell_{k+1}^* - 1 \le \ell_k^* + R$. The number of j's skipped because of these loops is thus captured by

$$\sum_{k\in\mathsf{Bad}} \left(\text{source}^*(\ell_{k+1}^*) - \text{source}^*(\ell_k^*)\right) = \sum_{k\in\mathsf{Bad}} \sum_{\ell=\ell_k^*}^{\ell_{k+1}^*-1} \left(\text{source}^*(\ell+1) - \text{source}^*(\ell)\right)$$
$$\le |\mathsf{Bad}|R + \sum_{k\in\mathsf{Bad}} \sum_{\ell=\ell_k^*}^{\ell_{k+1}^*-1} \left(\text{source}^*(\ell+1) - \text{source}^*(\ell) - 1\right).$$

Using the second part of Lemma 6.3, the above can be upperbounded by

$$|\mathsf{Bad}|R + |\mathsf{Bad}|R \cdot 2H + 2^{-0.2H}n \le 2^{-0.011H}n.$$

We finish with the case when $k \in \mathsf{Good}$ and bound the total number of j's skipped because of some $k \in \mathsf{Good}$. Given that every bit of x in the window of $[\text{source}^*(\ell_k^*) : i_k^* + R - 1]$ is included, the number of j's skipped is captured by

$$\text{source}^*(\ell_{k+1}^*) - (i_k^* + R)$$
$$= \text{source}^*(\ell_{k+1}^*) - \text{source}^*(\ell_{k+1}^* - 1) + \text{source}^*(\ell_{k+1}^* - 1) - (i_k^* + R).$$

Note that from (iii) we have $\text{source}^*(\ell_{k+1}^* - 1) - (i_k^* + R) \le G$. As a result, the total number of j's skipped is at most (again using the second part of Lemma 6.3 and a similar argument as above)

$$|\mathsf{Good}|G + \sum_{k\in\mathsf{Good}} \left(\text{source}^*(\ell_{k+1}^*) - \text{source}^*(\ell_{k+1}^* - 1) - 1\right) \le |\mathsf{Good}|G + 2^{-0.2H}n + |\mathsf{Good}| \cdot 2H.$$

Using $|\mathsf{Good}| \le 2n/R$, the above is at most $2^{-0.07H}n$.

To summarize, the edit distance between w and x is at most

$$\overbrace{2^{-0.011H}n}^{\text{first step}} + \overbrace{2^{-0.07H}n}^{\text{second step}} + \overbrace{O(R+\log n)}^{\text{third step, case 1}} + \overbrace{2^{-0.011H}n}^{\text{third step, case 2, } k\in\mathsf{Good}} + \overbrace{2^{-0.07H}n}^{\text{third step, case 2, } k\in\mathsf{Bad}} \le 2^{-0.01H}n.$$

This finishes the proof of Theorem 6.1.

7 Proof of Theorem 1.2: Lower bound on expected edit distance from few traces

In this section we prove Theorem 1.2. Recall that $\mathbf{x} \sim \{0,1\}^n$, $M \le \Theta(1/\delta)$, $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)} \sim \text{Del}_\delta(\mathbf{x})$, and $\mathtt{A}$ is an arbitrary algorithm which, on input δ and $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$, outputs a hypothesis string $\widehat{\mathsf{x}}$ for $\mathbf{x}$. We prove Theorem 1.2 by showing that $\mathbf{E}[d_{\text{edit}}(\widehat{\mathsf{x}}, \mathbf{x})] \ge n \cdot (\delta M)^{O(M)}$.

The main idea is to reduce the approximate trace reconstruction problem to the problem of computing the exact length of many runs of 0's (1-deserts) that are either of length M or of length $M+1$. We consider many instances of (a slight variation of) the following atomic problem: distinguish between a run of length M and a run of length $M+1$, given M "traces" of the run at

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

deletion rate δ. This problem is equivalent to that of distinguishing between $\mathbf{X} \sim \mathrm{Bin}(M, 1-\delta)$ and $\mathbf{X}' \sim \mathrm{Bin}(M+1, 1-\delta)$ given samples from a distribution that is either $\mathbf{X}$ or $\mathbf{X}'$. (For technical reasons the actual atomic problem we work with, described in the next subsection, is the problem of distinguishing between two product distributions over non-negative integers which are closely related to these binomial distributions.)

7.1 The atomic problem Consider the following two product distributions over pairs of non-negative integers:

$$\mathcal{D}_0 := \mathrm{Bin}(M, 1-\delta) \times \mathrm{Bin}(M+1, 1-\delta); \quad \mathcal{D}_1 := \mathrm{Bin}(M+1, 1-\delta) \times \mathrm{Bin}(M, 1-\delta).$$

We consider a uniform prior distribution $\mathcal{P}$ over $\{\mathcal{D}_0, \mathcal{D}_1\}$. In this subsection we prove the following lemma:

LEMMA 7.1. *Let $\mathcal{D}_{\boldsymbol{b}} \sim \mathcal{P}$, and let p be the optimal (minimal) failure probability of any algorithm which is given a sample $\mathbf{S}_M$ consisting of M independent draws from $\mathcal{D}_{\boldsymbol{b}}$ and aims to identify whether $\boldsymbol{b} = 0$ or $\boldsymbol{b} = 1$. Then $p \geq (\delta M)^{cM}$ for some absolute constant $c > 0$.*

Proof. The optimal failure probability is achieved by the Bayes optimal predictor, which outputs 0 if $\mathbf{Pr}[\mathcal{D}_0|\mathbf{S}_M] \geq \mathbf{Pr}[\mathcal{D}_1|\mathbf{S}_M]$ and outputs 1 if $\mathbf{Pr}[\mathcal{D}_0|\mathbf{S}_M] < \mathbf{Pr}[\mathcal{D}_1|\mathbf{S}_M]$. By Bayes' theorem, for $b \in \{0,1\}$ we have

$$\mathbf{Pr}[\mathcal{D}_b|\mathbf{S}_M] = \frac{\mathbf{Pr}[\mathbf{S}_M|\mathcal{D}_b]\,\mathbf{Pr}[\mathcal{D}_b]}{\mathbf{Pr}[\mathbf{S}_M]} = \frac{\mathbf{Pr}[\mathbf{S}_M|\mathcal{D}_b]}{2\,\mathbf{Pr}[\mathbf{S}_M]}$$

so for any fixed outcome S_M of the random variable $\mathbf{S}_M$, the Bayes optimal predictor outputs 0 on S_M if and only if

$$\mathbf{Pr}_{\mathbf{S}_M \sim (\mathcal{D}_0)^M}[\mathbf{S}_M = S_M] \geq \mathbf{Pr}_{\mathbf{S}_M \sim (\mathcal{D}_1)^M}[\mathbf{S}_M = S_M].$$

Consider the particular outcome of the M draws which is

$$\overbrace{(M, M-1), \ldots, (M, M-1)}^{M \text{ pairs}},$$

i.e., in each draw the outcome of the first coordinate is M and the outcome of the second coordinate is $M-1$. It is clear that the Bayes optimal predictor, on this input, will output 1; to see this rigorously, the probability of this outcome under $\mathcal{D}_1$ is

$$\left(\binom{M+1}{M}(1-\delta)^M\delta \cdot \binom{M}{M-1}(1-\delta)^{M-1}\delta\right)^M = \left((M+1)M(1-\delta)^{2M-1}\delta^2\right)^M \tag{7.31}$$

while its probability under $\mathcal{D}_0$ is

$$\left(\binom{M}{M}(1-\delta)^M \cdot \binom{M+1}{M-1}(1-\delta)^{M-1}\delta^2\right)^M = \frac{1}{2^M} \cdot (7.31) \tag{7.32}$$

But the probability of this outcome when the source distribution is $\mathcal{D}_0$ is (recalling that $M \leq \Theta(1/\delta)$)

$$(7.32) = \frac{1}{2^M} \cdot (7.31) = \left(\Theta(M^2\delta^2)\right)^M = \left(\Theta(M\delta)\right)^{\Theta(M)}.$$

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Since the probability that the source distribution is $\mathcal{D}_0$ is $1/2$, it follows that the optimal failure probability for any M-sample algorithm for this distinguishing problem is at least

$$\frac{1}{2} \cdot (\Theta(M\delta))^{\Theta(M)} = (M\delta)^{\Theta(M)}.$$

□

7.2 Direct sum (Paired Run Length Problem) We define the **Paired Run Length Problem (PRLP)** as follows. Fix $B \in \mathbb{N}$. An instance of the PRLP is specified by a binary vector $\mathsf{z} = (\mathsf{z}_1, \mathsf{z}_2, \ldots, \mathsf{z}_B) \in \{0,1\}^B$. For an instance z of the PRLP, an algorithm is given as input samples of B-tuples of M pairs from the product distribution $\mathcal{D}_{\mathsf{z}} = \mathcal{D}_{\mathsf{z}_1} \times \mathcal{D}_{\mathsf{z}_2} \times \cdots \times \mathcal{D}_{\mathsf{z}_B}$. It then must return some $\widehat{\mathsf{z}} \in \{0,1\}^*$, with the objective of minimizing $\mathbf{E}[d_{\text{edit}}(\mathsf{z}, \widehat{\mathsf{z}})]$.

We begin by recording a warmup lemma which states that the PRLP cannot be solved *exactly* with success probability better than that obtained by solving each instance independently.

LEMMA 7.2. *Let $p = p(M,\delta) < 1/2$ be the optimal failure probability of any algorithm for the atomic problem from Lemma 7.1. For a uniform $\mathbf{z} \sim \{0,1\}^B$, let $\mathtt{A}_{\mathrm{PRLP}}$ be any algorithm for the PRLP that is given M samples from $\mathcal{D}_{\mathbf{z}}$, and let $\widehat{\mathbf{z}}$ be its output. Then $\mathbf{Pr}[\widehat{\mathbf{z}} = \mathbf{z}] \le (1-p)^B$.*

Proof. This is a consequence of the independence of the distributions $\mathcal{D}_{\mathbf{z}_i}, i \in [B]$; we now give details. We have

$$\mathbf{Pr}[\widehat{\mathbf{z}} = \mathbf{z}] = \prod_{i=1}^{B} \mathbf{Pr}\left[\mathbf{z}_i = \widehat{\mathbf{z}}_i \mid \mathbf{z}_k = \widehat{\mathbf{z}}_k \text{ for all } k \in [i-1]\right].$$

Suppose there exists an algorithm $\mathtt{A}_{\mathrm{PRLP}}$ for the PRLP which outputs $\widehat{\mathsf{z}}$ such that $\mathbf{Pr}[\widehat{\mathbf{z}} = \mathbf{z}] > (1-p)^B$. Then there exists an $i^* \in [B]$ such that

$$\mathbf{Pr}\left[\mathbf{z}_{i^*} = \widehat{\mathbf{z}}_{i^*} | \mathbf{z}_k = \widehat{\mathbf{z}}_k \text{ for all } k \in [i^*-1]\right] > 1-p. \tag{7.33}$$

We use this to construct a "too good to be true" algorithm $\mathtt{A}$ for the atomic problem. Given M samples of the distribution $\mathcal{D}_{\boldsymbol{b}}$ for a uniform (unknown) $\boldsymbol{b} \sim \{0,1\}$, $\mathtt{A}$ "embeds" the problem into the PRLP problem. Specifically, it draws $\mathbf{z}' \sim \{0,1\}^{B-1}$ and simulates M samples of $\mathcal{D}_{\mathbf{z}'_i}, i \in [B-1]$. Let $\mathbf{z} \in \{0,1\}^B$ be defined by

$$\mathbf{z}_i = \begin{cases} \mathbf{z}'_i, & i < i^* \\ \boldsymbol{b}, & i = i^* \\ \mathbf{z}'_{i-1}, & i > i^*. \end{cases}$$

Clearly, $\mathbf{z}$ is uniformly random. $\mathtt{A}$ generates M samples of $\mathcal{D}_{\mathbf{z}}$ by appropriately concatenating the M samples of $\mathcal{D}_{\boldsymbol{b}}$ and the simulated samples of $\mathcal{D}_{\mathbf{z}'_i}, i \in [B-1]$. It then invokes $\mathtt{A}_{\mathrm{PRLP}}$ on the generated samples, receives output $\widehat{\mathbf{z}} \in \{0,1\}^B$, and checks whether $\mathbf{z}_k = \widehat{\mathbf{z}}_k$ for all $k \in [i^*-1]$; if this is the case then it returns $\widehat{\boldsymbol{b}} := \widehat{\mathbf{z}}_{i^*}$, and if it is not the case then it tries again by drawing a fresh independent $\mathbf{z}' \sim \{0,1\}^{B-1}$, repeating this until it is the case that $\mathbf{z}_k = \widehat{\mathbf{z}}_k$. By Equation (7.33), $\widehat{\boldsymbol{b}} = \boldsymbol{b}$ with probability more than $1-p$, which contradicts the definition of p. □

We use Lemma 7.2 to give a lower bound on the expected edit distance for any algorithm for the PRLP:

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

LEMMA 7.3. *Let $p = p(M, \delta) < 1/2$ be the optimal failure probability of any algorithm for the atomic problem from Lemma 7.1. For a uniform $\mathbf{z} \sim \{0,1\}^B$, let $\mathtt{A}_{\mathrm{PRLP}}$ be any algorithm for the PRLP that is given M samples from $\mathcal{D}_{\mathbf{z}}$, and let $\widehat{\mathbf{z}}$ be its output. Then $\mathbf{E}[d_{\mathrm{edit}}(\mathbf{z}, \widehat{\mathbf{z}})] \geq c'B \cdot p/\log(1/p)$, where $c' > 0$ is an absolute constant.*

Proof. Let $p' < p$ be a parameter to be specified later, and let $\mathcal{E}$ be the event that $d_{\mathrm{edit}}(\mathbf{z}, \widehat{\mathbf{z}}) \leq Bp'$. (Since $\mathtt{A}_{\mathrm{PRLP}}$ can without loss of generality be taken to be deterministic, $\mathcal{E}$ is over the uniform random draw of $\mathbf{z}$ and the M samples from $\mathcal{D}_{\mathbf{z}}$.) Fix a potential matching $\mu = (i_t, j_t)_{t \in [(1-p')B]}$ between $\mathbf{z}$ and $\widehat{\mathbf{z}}$ that is of size $(1-p')B$. Let $\mathcal{E}_\mu$ be the event that $\mathbf{z}_{i_t}$ is actually equal to $\widehat{\mathbf{z}}_{j_t}$ for all t (i.e., the potential matching actually is a matching between $\mathbf{z}$ and $\widehat{\mathbf{z}}$). By Lemma 7.2, we have that

$$\mathbf{Pr}[\mathcal{E}_\mu] \leq (1-p)^{(1-p')B} \leq \exp(-pB/2),$$

where we used $1 - p' > 1/2$ for the second inequality.

Now, by a union bound over all potential matchings of size $(1-p')B$, we get that

$$\mathbf{Pr}[\mathcal{E}] \leq \sum_{\mu} \mathbf{Pr}[\mathcal{E}_\mu] \leq \binom{B}{Bp'}^2 \cdot \exp\left(-\frac{pB}{2}\right) \leq \exp\left(B\left(2p' \log \frac{e}{p'} - \frac{p}{2}\right)\right).$$

Choosing $p' = c_1 p/\log(1/p)$ for some small enough $c_1 > 0$, we have $\mathbf{Pr}[\mathcal{E}] \leq 2^{-c_2 Bp}$. Hence we have $\mathbf{E}[d_{\mathrm{edit}}(\mathbf{z}, \widehat{\mathbf{z}})] \geq (1 - 2^{-c_2 Bp}) \cdot (Bp') = c'B \cdot p/\log(1/p)$ for some absolute constant $c' > 0$, and the lemma is proved. ∎

7.3 Embedding and proof of Theorem 1.2 In this subsection we relate the PRLP to the average-case approximate trace reconstruction problem and prove Theorem 1.2. To explain the connection between average-case approximate trace reconstruction and the PRLP, let us define two subwords

$$\alpha := 0^M 1 0^{M+1} 1 1, \qquad \beta := 0^{M+1} 1 0^M 1 1.$$

Observe that for any string $\mathtt{x} \in \{0,1\}^n$ and any pair of distinct intervals $I, J \subset [n]$ such that $\mathtt{x}_I, \mathtt{x}_J \in \{\alpha, \beta\}$, I and J must be disjoint. Note that in a uniform random string $\mathbf{x}$, each of these subwords occurs with expected frequency $q = 2^{-N}$, where $N := |\alpha| = |\beta| = 2M + 4$. Intuitively, given M traces from $\mathrm{Del}_\delta(\mathbf{x})$, determining whether a segment of $\mathbf{x}$ is in fact α or β corresponds to a single instance of the atomic problem from Section 7.1, and determining this for B disjoint segments corresponds to an instance of the PRLP. In the rest of this subsection we make this correspondence precise and show how a high-accuracy algorithm for M-sample average-case approximate trace reconstruction yields a high-accuracy algorithm for the PRLP; combining this with the lower bound on the PRLP from Section 7.2 gives Theorem 1.2.

Fix a sufficiently small absolute constant $c < 1$, and let $\mathtt{A}$ be an algorithm for average-case approximate trace reconstruction which, given $M \leq c/\delta$ traces of a random string $\mathbf{x} \in \{0,1\}^n$ (and the value of δ), returns $\widehat{\mathbf{x}} \in \{0,1\}^*$ such that $\mathbf{E}[d_{\mathrm{edit}}(\mathbf{x}, \widehat{\mathbf{x}})] \leq n(\delta M)^{CM}$ for some constant C. Building on $\mathtt{A}$, in Figure 7 we provide a "too good to be true" (given Lemma 7.3) algorithm $\mathtt{A}_{\mathrm{PRLP}}$ for the Paired Run Length Problem.

We give a description of the algorithm $\mathtt{A}_{\mathrm{PRLP}}$. Consider a uniformly random string $\mathbf{x}' \in \{0,1\}^n$, where $n := N \cdot 2^N \cdot B$ is chosen such that $\mathbf{x}'$ contains at least B occurrences of α or β with high probability. Given $\mathbf{x}'$ and $\mathbf{z} \sim \{0,1\}^B$, we define another string $\mathbf{x} \in \{0,1\}^n$ by replacing the b-th

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Algorithm 5: $\mathtt{A}_{\mathrm{PRLP}}$

Input: A list of M samples $s^{(1)}, \ldots, s^{(M)}$ from the product distribution $\mathcal{D}_{\mathbf{z}}$ for an (unknown) uniformly random $\mathbf{z} \sim \{0,1\}^B$ (and the value of $\delta \in (0,1)$).

Output: A string $\widehat{\mathbf{z}} \in \{0,1\}^{\leq B}$.

1 Set $N = 2M + 4$ and $n = N \cdot 2^N \cdot B$.

2 Generate a uniformly random string $\mathbf{x}' \in \{0,1\}^n$.

3 Let $B' = \min\{B$, number of occurrences of α or β in $\mathbf{x}'\}$.

4 For $b \in [B']$, let $I^{(b)} = [i^{(b)}, i^{(b)} + N - 1]$ be the b-th interval in $\mathbf{x}'$ such that $\mathbf{x}'_{I^{(b)}} \in \{\alpha, \beta\}$. Also set $i^{(B'+1)} = n + 1$.

5 **for** $m \in [M]$ **do**

6 Set $\mathbf{y}^{(m)} \sim \mathrm{Del}_\delta\left(\mathbf{x}'_{[1, i^{(1)}-1]}\right)$.

7 Let $s^{(m)} = \left(s^{(m)}_{b,1}, s^{(m)}_{b,2}\right)_{b \in [B']}$, where $\left(s^{(m)}_{b,1}, s^{(m)}_{b,2}\right) \sim \mathcal{D}_{\mathbf{z}_b}$.

8 **for** $b \in [B']$ **do**

9 Set $\mathbf{y}'^{(m)}_b \sim 0^{s^{(m)}_{b,1}} \circ \mathrm{Del}_\delta(1) \circ 0^{s^{(m)}_{b,2}} \circ \mathrm{Del}_\delta(11) \in \{0,1\}^{\leq N}$.

10 Set $\bar{\mathbf{y}}^{(m)}_b \sim \mathrm{Del}_\delta\left(\mathbf{x}'_{[i^{(b)}+N, i^{(b+1)}-1]}\right)$.

11 Append $\mathbf{y}'^{(m)}_b \circ \bar{\mathbf{y}}^{(m)}_b$ to the end of $\mathbf{y}^{(m)}$.

12 Run $\mathtt{A}$ on δ and the M strings $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ to obtain $\widehat{\mathbf{x}} \in \{0,1\}^*$.

13 Let $\widehat{B} = \min\{B$, number of occurrences of α or β in $\widehat{\mathbf{x}}\}$.

14 **for** $b \in [\widehat{B}]$ **do**

15 Let $J^{(b)}$ be the b-th interval in $\widehat{\mathbf{x}}$ such that $\widehat{\mathbf{x}}_{J^{(b)}} \in \{\alpha, \beta\}$.

16 Set $\widehat{\mathbf{z}}_b = 0$ if $\widehat{\mathbf{x}}_{J^{(b)}} = \alpha$, and $\widehat{\mathbf{z}}_b = 1$ if $\widehat{\mathbf{x}}_{J^{(b)}} = \beta$.

17 **return** $\widehat{\mathbf{z}} := \left(\widehat{\mathbf{z}}_1, \widehat{\mathbf{z}}_2, \ldots, \widehat{\mathbf{z}}_{\widehat{B}}\right)$.

Figure 7: Algorithm $\mathtt{A}_{\mathrm{PRLP}}$ for the PRLP problem, given an algorithm $\mathtt{A}$ for average-case approximate trace reconstruction.

occurrence of α or β in $\mathbf{x}'$ with α if $\mathbf{z}_b = 0$ and with β if $\mathbf{z}_b = 1$ as long as $b \leq B$. As any pair of occurrences of α or β are disjoint, the above procedure is well-defined. We show in Lemma 7.4 that $\mathbf{x}$ is uniformly random, and hence a set of M traces from $\mathbf{x}$ is a legitimate input to $\mathtt{A}$.

The algorithm $\mathtt{A}_{\mathrm{PRLP}}$ for the PRLP of course does not have access to $\mathbf{z}$, but only to samples $s^{(1)}, \ldots, s^{(M)} \sim \mathcal{D}_{\mathbf{z}}$. Hence, it cannot generate $\mathbf{x}$ explicitly. However, we show that it can *simulate* M independent traces from $\mathbf{x}$ by generating $\mathbf{x}' \sim \{0,1\}^n$, followed by generating traces from the segments in $\mathbf{x}'$ that are disjoint from the occurrences of α or β, and then concatenating them appropriately with the traces of α or β that are generated using the samples $s^{(1)}, \ldots, s^{(M)}$ (see the loop spanning lines 5–11 of the algorithm). It then invokes $\mathtt{A}$ on these traces to obtain a string $\widehat{\mathbf{x}} \in \{0,1\}^*$, extracts from $\widehat{\mathbf{x}}$ a binary vector $\widehat{\mathbf{z}} \in \{0,1\}^*$ based on the first (at most) B occurrences of α or β in $\widehat{\mathbf{x}}$, and returns it.

We note that the different intervals I in line 4 are disjoint from each other, and likewise for the

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

different intervals J in line 15.

LEMMA 7.4. *Let $\mathbf{z} \in \{0,1\}^B$ be uniformly random, let $\mathbf{x}'$ be uniform random over $\{0,1\}^n$. Let B' be the minimum of B and the number of occurrences of α or β in $\mathbf{x}'$, and let $\mathbf{x}$ be obtained from $\mathbf{x}'$ by replacing the b-th occurrence of α or β in $\mathbf{x}'$ with α if $\mathbf{z}_b = 0$ and with β if $\mathbf{z}_b = 1$ for all $b \in [B']$. Then $\mathbf{x}$ is uniformly random over $\{0,1\}^n$.*

Proof. Fix any possible outcome $\mathsf{x} \in \{0,1\}^n$ of $\mathbf{x}$ and let $j = \min\{B$, number of occurrences of α or β in $\mathsf{x}\}$. There are precisely 2^j outcomes $\mathsf{x}' \in \{0,1\}^n$ of $\mathbf{x}'$ for which it is possible that $\mathbf{x}' = \mathsf{x}'$ could give rise to $\mathbf{x} = \mathsf{x}$ (these are precisely the 2^j strings obtained by replacing the first j occurrences of α or β in x by α or β in all possible ways). Each of these outcomes has probability $1/2^n$ under $\mathbf{x}'$ because $\mathbf{x}'$ is uniform random and for each such outcome there is a $1/2^j$ chance that the replacement yields $\mathbf{x} = \mathsf{x}$ from $\mathbf{x}' = \mathsf{x}'$. Hence $\mathbf{Pr}[\mathbf{x} = \mathsf{x}] = 2^j \cdot (1/2^n) \cdot (1/2^j) = 1/2^n$. ∎

LEMMA 7.5. *$\mathbf{x}'$ (and hence $\mathbf{x}$) contains at least B disjoint occurrences of α or β with probability at least $1 - \exp(-\Omega(B))$.*

Proof. As $\mathbf{x}'$ is a uniformly random string, for any position $i \in [n - N + 1]$, we have that

$$\mathbf{Pr}\left[\mathbf{x}'_{[i:i+N-1]} \in \{\alpha, \beta\}\right] = \frac{2}{2^N} = \frac{1}{2^{N-1}}.$$

Let $q = 2^{-(N-1)}$. We divide $\mathbf{x}'$ into $s := 2^N \cdot B$ disjoint segments $\mathbf{x}'^{(j)}$ of length N. For $j \in [s]$, let $\mathcal{F}_j$ be the indicator of the event that $\mathbf{x}'^{(j)} \in \{\alpha, \beta\}$. Then $\mathcal{F}_j \sim \mathrm{Ber}(q)$, and $\mathcal{F}_j, j \in [s]$ are independent.

Let $\mathcal{F} = \sum_j \mathcal{F}_j$ denote the number of segments $\mathbf{x}'^{(j)}$ that are α or β. Note that $\mathcal{F}$ is a lower bound on the overall number of disjoint occurrences of α or β, because we are only considering segments ending at positions which are integral multiples of N. Clearly,

$$\mathbf{E}[\mathcal{F}] = \sum_{j \in [s]} \mathbf{E}[\mathcal{F}_j] = sq = 2B.$$

By the Chernoff Bound, we have $\mathcal{F} < B$ with probability at most $\exp(-\Omega(B))$. Along with the observations above and the fact that $\mathbf{x}_I \in \{\alpha, \beta\}$ if and only if $\mathbf{x}'_I \in \{\alpha, \beta\}$ for an interval I of length N, this concludes the proof. ∎

Next we state and prove a crucial lemma which implies that if $\mathtt{A}$ is a good algorithm for average-case approximate trace reconstruction, then $\mathtt{A}_{\mathrm{PRLP}}$ is a good algorithm for the PRLP problem:

LEMMA 7.6. *If $\mathbf{x}'$ (and hence $\mathbf{x}$) contains at least B disjoint occurrences of α or β, then $d_{\mathrm{edit}}(\mathbf{z}, \widehat{\mathbf{z}}) \leq 2 \cdot d_{\mathrm{edit}}(\mathbf{x}, \widehat{\mathbf{x}})$.*

Proof. Let $I^{(1)}, I^{(2)}, \ldots, I^{(S)}$ be intervals of length N corresponding to the occurrences of α or β in $\mathbf{x}$ (so by the assumption of the lemma, we have $S \geq B$). Similarly, let $J^{(1)}, J^{(2)}, \ldots, J^{(T)}$ be intervals of length N corresponding to the occurrences of α or β in $\widehat{\mathbf{x}}$. Fix an optimal matching μ between $\mathbf{x}$ and $\widehat{\mathbf{x}}$ corresponding to any longest common subsequence between those strings (if there is more than one choice for μ it can be selected from the optimal matchings arbitrarily).

Consider the matching τ on $[S] \times [T]$, where $(s,t) \in \tau$ if and only if $\mu(I^{(s)}) = J^{(t)}$. Let τ' be the induced matching obtained by restricting τ to $[B] \times [|\widehat{\mathbf{z}}|]$. Note that τ' corresponds to a longest common subsequence of $\mathbf{z}$ and $\widehat{\mathbf{z}}$. We consider two cases.

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

1. Suppose $(s,t) \in \tau$ for some $s \le B$ and $t > B$. Note that this implies there are at least $t > B$ occurences of α or β in $\widehat{\mathbf{x}}$, and so we have $|\widehat{\mathbf{z}}| = |\mathbf{z}|$. We claim that for every $t' \le B$, if $(s',t') \in \tau$ for some s' then $s' \le B$; this is because otherwise we have $s, t' \le B$ and $s', t > B$, but $(s,t), (s',t') \in \tau$, contradicting our definition of matching.

   Therefore, for every $t \in [T], t \le B$ that is not matched to an element of $[S]$ in τ', it is also not matched to an element of $[S]$ in τ, and hence either (1) some element in $J^{(t)}$ is not matched in μ, or (2) the indices in $\mathbf{x}$ that are matched to $J^{(t)}$ do not form an interval. Either case contributes at least 1 deletion in $\mathbf{x}$ or $\widehat{\mathbf{x}}$ to $d_{\text{edit}}(\mathbf{x}, \widehat{\mathbf{x}})$. Since there are $d_{\text{edit}}(\mathbf{z}, \widehat{\mathbf{z}})/2$ such t's, we have $d_{\text{edit}}(\mathbf{z}, \widehat{\mathbf{z}}) \le 2\, d_{\text{edit}}(\mathbf{x}, \widehat{\mathbf{x}})$.

2. Otherwise, for every $s \le B$, if $(s,t) \in \tau$ then it must be that $t \le B$. Moreover, for every $s \le B$ that is not in τ', it is also not in τ, and therefore either (1) some element in $I^{(s)}$ is not in μ, or (2) the indices in $\widehat{\mathbf{x}}$ that are matched to $I^{(s)}$ do not form an interval. Either case contributes at least 1 deletion in $\mathbf{x}$ or $\widehat{\mathbf{x}}$ to $d_{\text{edit}}(\mathbf{x}, \widehat{\mathbf{x}})$. Since $|\mathbf{z}| \ge |\widehat{\mathbf{z}}|$, we have at least $d_{\text{edit}}(\mathbf{z}, \widehat{\mathbf{z}})/2$ such s', and so $d_{\text{edit}}(\mathbf{z}, \widehat{\mathbf{z}}) \le 2\, d_{\text{edit}}(\mathbf{x}, \widehat{\mathbf{x}})$.

□

Proof. [Proof of Theorem 1.2] Lemma 7.1 and Lemma 7.3 imply that for any algorithm $\mathtt{A}_{\text{PRLP}}$ that solves the PRLP given M samples from $\mathcal{D}_{\mathbf{z}}$, its output $\widehat{\mathbf{z}}$ satisfies

$$\mathbf{E}\big[d_{\text{edit}}(\mathbf{z}, \widehat{\mathbf{z}})\big] \ge B \cdot (\delta M)^{cM} \tag{7.34}$$

for some absolute constant $c > 0$.

Now, let $\mathtt{A}$ be any algorithm which, given δ and traces $\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(M)}$ from a random string $\mathbf{x} \in \{0,1\}^n$ as input, outputs a hypothesis string $\widehat{\mathbf{x}}$ for $\mathbf{x}$ such that $\mathbf{E}[d_{\text{edit}}(\mathbf{x}, \widehat{\mathbf{x}})] < n \cdot (\delta M)^{CM}$. Consider the algorithm $\mathtt{A}_{\text{PRLP}}$ described in Figure 7. By Lemma 7.5, $\mathbf{x}'$ (and hence $\mathbf{x}$) has at least B disjoint occurrences of α or β with probability at least $1 - e^{-\Omega(B)}$, in which case we have $d_{\text{edit}}(\mathbf{z}, \widehat{\mathbf{z}}) \le 2 \cdot d_{\text{edit}}(\mathbf{x}, \widehat{\mathbf{x}})$ by Lemma 7.6. If $\mathbf{x}$ has fewer than B disjoint occurrences of α or β, we have $d_{\text{edit}}(\mathbf{z}, \widehat{\mathbf{z}}) \le 2B$ as $\mathbf{z} \in \{0,1\}^B$ and $\widehat{\mathbf{z}} \in \{0,1\}^{\le B}$. So, we obtain

$$\begin{aligned} \mathbf{E}[d_{\text{edit}}(\mathbf{z}, \widehat{\mathbf{z}})] &\le e^{-\Omega(B)} \cdot 2B + \big(1 - e^{-\Omega(B)}\big) \cdot 2\, \mathbf{E}\big[d_{\text{edit}}(\mathbf{x}, \widehat{\mathbf{x}})\big] \\ &\le 4n \cdot (\delta M)^{CM} \\ &\le B \cdot (\delta M)^{C'M} \end{aligned} \tag{7.35}$$

for some suitable constant $C' > 0$. Equations (7.34) and (7.35) lead to the desired contradiction for $C' > c$, which concludes the proof of Theorem 1.2. □

References

[1] Frank Ban, Xi Chen, Adam Freilich, Rocco A. Servedio, and Sandip Sinha. Beyond trace reconstruction: Population recovery from the deletion channel. In *60th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 745–768. IEEE Computer Society, 2019. 1.1.1

[2] Frank Ban, Xi Chen, Rocco A. Servedio, and Sandip Sinha. Efficient average-case population recovery in the presence of insertions and deletions. In *APPROX/RANDOM 2019*, volume 145 of *LIPIcs*, pages 44:1–44:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. 1.1.1

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

[3] Tuğkan Batu, Sampath Kannan, Sanjeev Khanna, and Andrew McGregor. Reconstructing strings from random traces. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004*, pages 910–918, 2004. 1.1, 1.1.1, 1.3, 2.1.2, 2.3, 5

[4] Joshua Brakensiek, Ray Li, and Bruce Spang. Coded trace reconstruction in a constant number of traces. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 482–493, 2020. 1.1.1

[5] Zachary Chase. New lower bounds for trace reconstruction. *Ann. Inst. H. Poincaré Probab. Statist.*, 57(2):627–643, 2021. 1.1.1

[6] Zachary Chase. Separating words and trace reconstruction. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 21–31. ACM, 2021. 1.1.1

[7] Zachary Chase and Yuval Peres. Personal communication. Manuscript, 2021. 1.1.2, 1.1, 1.3

[8] Xi Chen, Anindya De, Chin Ho Lee, Rocco A. Servedio, and Sandip Sinha. Polynomial-time trace reconstruction in the low deletion rate regime. In *12th Innovations in Theoretical Computer Science Conference*, volume 185 of *LIPIcs*, pages 20:1–20:20, 2021. 1.1.1, 1.3, 2.1.2, 2.3, 5, 5.1

[9] Xi Chen, Anindya De, Chin Ho Lee, Rocco A. Servedio, and Sandip Sinha. Polynomial-time trace reconstruction in the smoothed complexity model. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pages 54–73, 2021. 1.1.1

[10] Mahdi Cheraghchi, João Ribeiro, Ryan Gabrys, and Olgica Milenkovic. Coded trace reconstruction. In *2019 IEEE Information Theory Workshop (ITW)*, page 1–5. IEEE Press, 2019. 1.1.1

[11] Sami Davies, Miklos Z. Rácz, Cyrus Rashtchian, and Benjamin G. Schiffer. Approximate trace reconstruction: Algorithms. In *IEEE International Symposium on Information Theory*, 2021. 1.1.2

[12] Anindya De, Ryan O'Donnell, and Rocco A. Servedio. Optimal mean-based algorithms for trace reconstruction. In *Proceedings of the 49th ACM Symposium on Theory of Computing (STOC)*, pages 1047–1056, 2017. 1.1.1

[13] Elena Grigorescu, Madhu Sudan, and Minshen Zhu. Limitations of mean-based algorithms for trace reconstruction at small distance. In *IEEE International Symposium on Information Theory*, 2021. 1.1.2

[14] Lisa Hartung, Nina Holden, and Yuval Peres. Trace reconstruction with varying deletion probabilities. In *Proceedings of the Fifteenth Workshop on Analytic Algorithmics and Combinatorics, ANALCO 2018, New Orleans, LA, USA, January 8-9, 2018.*, pages 54–61, 2018. 1.1.1

[15] Nina Holden, Robin Pemantle, and Yuval Peres. Subpolynomial trace reconstruction for random strings and arbitrary deletion probability. In *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018*, volume 75 of *Proceedings of Machine Learning Research*, pages 1799–1840. PMLR, 2018. 1.1.1

[16] Nina Holden, Robin Pemantle, Yuval Peres, and Alex Zhai. Subpolynomial trace reconstruction for random strings and arbitrary deletion probability. *Mathematical Statistics and Learning*, 2(3/4):275–309, 2019. 1.1.1

[17] Thomas Holenstein, Michael Mitzenmacher, Rina Panigrahy, and Udi Wieder. Trace reconstruction with constant deletion probability and related results. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008*, pages 389–398, 2008. 1.1.1

[18] V. V. Kalashnik. Reconstruction of a word from its fragments. *Computational Mathematics and Computer Science (Vychislitel'naya matematika i vychislitel'naya tekhnika), Kharkov*, 4:56–57, 1973. 1.1

[19] Akshay Krishnamurthy, Arya Mazumdar, Andrew McGregor, and Soumyabrata Pal. Trace reconstruction: Generalized and parameterized. In *27th Annual European Symposium on Algorithms, ESA 2019*, volume 144 of *LIPIcs*, pages 68:1–68:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. 1.1.1

[20] Vladimir Levenshtein. Efficient reconstruction of sequences. *IEEE Transactions on Information Theory*, 47(1):2–22, 2001. 1.1

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited

[21] Vladimir Levenshtein. Efficient reconstruction of sequences from their subsequences or supersequences. *Journal of Combinatorial Theory Series A*, 93(2):310–332, 2001. 1.1

[22] Colin McDiarmid. *Concentration*, pages 195–248. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. 3.1

[23] Andrew McGregor, Eric Price, and Sofya Vorotnikova. Trace reconstruction revisited. In *Proceedings of the 22nd Annual European Symposium on Algorithms*, pages 689–700, 2014. 1.1.1

[24] Shyam Narayanan. Improved algorithms for population recovery from the deletion channel. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1259–1278. SIAM, 2021. 1.1.1

[25] Shyam Narayanan and Michael Ren. Circular Trace Reconstruction. In *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*, pages 18:1–18:18, 2021. 1.1.1

[26] Fedor Nazarov and Yuval Peres. Trace reconstruction with $\exp\left(O(n^{1/3})\right)$ samples. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 1042–1046, 2017. 1.1.1

[27] Yuval Peres and Alex Zhai. Average-case reconstruction for the deletion channel: Subpolynomially many traces suffice. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 228–239. IEEE Computer Society, 2017. 1.1.1

[28] Jin Sima and Jehoshua Bruck. Trace reconstruction with bounded edit distance. In *IEEE International Symposium on Information Theory*, 2021. Manuscript, available at https://arxiv.org/abs/2102.05372. 1.1.2

[29] Sundara Rajan Srinivasavaradhan, Michelle Du, Suhas Diggavi, and Christina Fragouli. On maximum likelihood reconstruction over multiple deletion channels. In *IEEE International Symposium on Information Theory, ISIT 2018*, pages 436–440, 2018. 1.1.2

Downloaded 08/01/22 to 165.123.34.86 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Copyright © 2022 by SIAM
Unauthorized reproduction of this article is prohibited