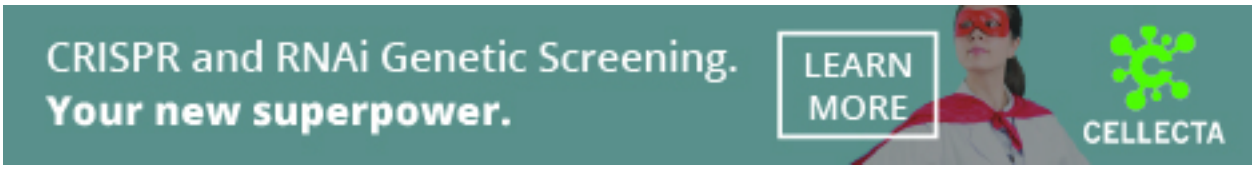
# Proving sequence aligners can guarantee accuracy in almost O(m log n) time through an average-case analysis of the seed-chain-extend heuristic

Jim Shaw and Yun William Yu

*Genome Res.* published online March 29, 2023
Access the most recent version at doi:10.1101/gr.277637.122





# Proving sequence aligners can guarantee accuracy in almost O(m log n) time through an average-case analysis of the seed-chain-extend heuristic

Jim Shaw[1]* and Yun William Yu[1,2]

[1] Department of Mathematics, University of Toronto

[2] Computer and Mathematical Sciences, University of Toronto at Scarborough

* corresponding author: jshaw@math.toronto.edu

Running title: Rigorous analysis of seed-chain-extend alignment

## Abstract

Seed-chain-extend with *k*-mer seeds is a powerful heuristic technique for sequence alignment employed by modern sequence aligners. While effective in practice for both runtime and accuracy, theoretical guarantees on the resulting alignment do not exist for seed-chain-extend. In this work, we give the first rigorous bounds for the efficacy of seed-chain-extend with *k*-mers *in expectation*. Assume we are given a random nucleotide sequence of length $\sim n$ that is indexed (or seeded) and a mutated substring of length $\sim m \leq n$ with mutation rate $\theta < 0.206$. We prove that we can find a $k = \Theta(\log n)$ for the *k*-mer size such that the expected runtime of seed-chain-extend under optimal linear gap cost chaining and quadratic time gap extension is $O\left(mn^{f(\theta)}\ log\ n\right)$ where $f(\theta) < 2.43 \cdot \theta$ holds as a loose bound. The alignment also turns out to be good; we prove that more than $1 - O\left(\sqrt{\frac{1}{m}}\right)$ fraction of the homologous bases are *recoverable* under an optimal chain. We also show that our bounds work when *k*-mers are *sketched*, i.e. only a subset of all *k*-mers is selected, and that sketching reduces chaining time without increasing alignment time or decreasing accuracy too much, justifying the effectiveness of sketching as a practical speedup in sequence alignment. We verify our results in simulation and on real noisy long-read data and

show that our theoretical runtimes can predict real runtimes accurately. We conjecture that our bounds can be improved further, and in particular, $f(\theta)$ can be further reduced.

## Introduction

Since the earliest years of bioinformatics, one primitive task has been sequence alignment (Altschul et al. 1990; Smith and Waterman 1981), which plays a major role in genomic sequencing and phylogenetics. Intuitively, alignment matches together similar parts of two strings, inserting gaps where needed. More formally, alignment is defined by a set of allowed edit operations (e.g. single-character substitutions, insertions, deletions) with associated costs; an alignment is a sequence of those operations transforming one string into another string (or for local alignment, transforming one string into a substring of the other), and the alignment score is the summed cost of those operations (Berger et al. 2021).

Unfortunately, the best guaranteed algorithms for computing alignment (Smith and Waterman 1981; Needleman and Wunsch 1970)are quadratic in time-complexity; worse, this bound appears to be tight (Backurs and Indyk 2018). There exist optimal algorithms with parameterized runtimes based on sequence similarity (Marco-Sola et al. 2021) that are faster and used in practice, but these still have worst-case quadratic time complexity as a function of the sequence length. Thus, to deal with large volumes of sequencing data (Marçais et al. 2019; Berger et al. 2016), sequence alignment programs employ heuristics (Altschul et al. 1990; Li 2018; Marçais et al. 2018; Kiełbasa et al. 2011; Langmead and Salzberg 2012; Lipman and Pearson 1985; Li and Durbin 2009) without performance guarantees(Medvedev 2022a)for computational efficiency. All these heuristic algorithms are by necessity fast, achieving empirically sub-quadratic runtimes on real problems. They fail for adversarial examples, but aligners perform well in practice because the sequences being aligned are similar and not pathological examples (Ivanov et al. 2022; Koerkamp and Ivanov 2022; Ukkonen 1983; Myers 1986).

Phylogenetics often makes use of comparative genomics, where aligning together multiple whole genomes allows annotating the ways in which two species have diverged over evolutionary time(Koonin et al. 2000). In genomic sequencing, the alignment task manifests because the sequencing machines are only able to read a small portion of a chromosome at a time, producing short snippets known as *reads* (Alser et al. 2021). It is then incumbent on read-mapping software to determine the likely origin location of that read in the genome, and for already sequenced species, this is usually performed by aligning the read to a known reference genome (Nurk et al. 2022).

Historically, different types of heuristics were used for the two tasks, because aligning a very small substring to a longer string is easier than aligning comparable-size strings. Indeed, the seed-and-extend heuristic, as seen in BWA and Bowtie 2, is preferred for aligning NGS short-reads to genomes instead of aligning genomes to genomes. However, as 3rd-generation long-read sequencing becomes more prominent, the two tasks become more similar and the same heuristics can be used for both(Sahlin et al. 2022)—in this manuscript, we address the *seed-chain-extend* heuristic (distinct from seed-and-extend) used in modern software for both read mapping and whole genome alignment.

**Our contribution**

Our goal in this manuscript is to close the gap between theory and practice, rigorously justifying the heuristics used in some of the most widely used alignment software. To this end, we turn to the methods of average-case analysis (Szpankowski 2001), which gives us a way of breaking through the quadratic barrier of alignment (Medvedev 2022b). Given a random string, we define a substitution model giving rise to a distribution on pairs of inputs and then average our analysis on pairs of strings over this distribution.

Recently, Ganesh and Sy (Ganesh and Sy 2020) also used probabilistic analysis to show that a heuristic algorithm based on banded alignment can run in $O(n \log n)$ time for two length n sequences and is optimal with high probability. However, their specific method has not yet seen any usage in practical software, and their analysis is invalid in the case of read mapping as it only pertains to two nearly equal-length strings. Thus, we turn instead to the analysis of an empirically battle-tested heuristic for sequence alignment: seed-chain-extend. Seed-chain-extend is a well-established technique for comparative genomics (Myers and Miller 1995; Abouelhoda and Ohlebusch 2005), and recently the addition of *sketching* (or subsampling) has made it popular for long-read aligners (Ren and Chaisson 2021; Chaisson and Tesler 2012; Sović et al. 2016) including minimap2(Li 2018), the primary algorithm our model of seed-chain-extend is based on.

We provide, to the best of our knowledge, the first average-case bounds on runtime and optimality for the *sketched k-mer seed-chain-extend* alignment heuristic under a pairwise mutation model. Our optimality result shows that for large enough *k*-mer size $k$, the alignment is mostly constrained to be near the correct diagonal of the alignment matrix and that runtime is close to linear when the mutation or error rate is reasonably small. We also show that subsampling $\Theta\left(\frac{1}{\log n}\right)$ of *k*-mers asymptotically reduces our bounds on chaining time but not for extension time. Our results give a theoretical justification for both the empirical accuracy and sub-quadratic runtime of seed-chain-extend.

## Results

We first give a specific, simplified version of our main theorem.

**Simplified Theorem 1 (Informal main result; no sketching).** *Suppose we are given a uniformly random DNA string of length $n$ and a mutated substring of length $m$ where each base is substituted with probability $\theta$. If $\theta < 0.206$ and the longer string is already seeded, then we can choose $k = \Theta(\log n)$ such that the expected runtime of k-mer seed-chain-extend is*

$O\left(mn^{f(\theta)}\log(n)\right) = O\left(mn^{2.43\cdot\theta}\log(n)\right)$ , *and in expectation* $\geq 1 - O\left(\frac{1}{\sqrt{m}}\right)$ *fraction of the homologous bases can be recovered from this alignment.*

We will state our models and definitions precisely below. Our main result, Theorem 1, precisely defines the function $f(\theta) < 2.43 \cdot \theta$ in the exponent, but $2.43 \cdot \theta$ is a convenient upper bound. We can quickly see from this bound that for modest mutation rates, $n^{2.43\cdot\theta}$ is not too large.

**Mutation model**

Let $S = x_1 x_2 \ldots x_{n+k-1}$ be a random uniform string with $n + k - 1$ i.i.d letters on an alphabet of size $\sigma$ for some $\mathrm{k} = \Theta(log\ \mathrm{n})$. Let $S' = y_{p+1} y_{p+2} \ldots y_{p+m+k-1}$ be a substring of $S$ of length $m + k - 1$ starting at a fixed position $p + 1$ with each character independently mutated to a different letter with probability $\theta$. Although notationally a little confusing at first, the $k - 1$ term ensures $S, S'$ contain exactly $n$ and $m$ *k*-mers respectively—*k*-mers are in many ways the natural unit of measurement, rather than individual characters. We model only point substitutions here and not indels. Independent substitution models have been considered in theoretical work (Blanca et al. 2022; Shaw and Yu 2022; Ganesh and Sy 2020), and importantly, also demonstrated to be useful empirically (Chaisson and Tesler 2012; Ondov et al. 2016; Sarmashghi et al. 2019). Also, while genomes can be repetitive, on the level of *k*-mers a random model has been shown to be reasonable (Fofanov et al. 2004). We discuss other possible random models in the Discussion section.

**Modelling seed-chain-extend**

A brief overview of seed-chain-extend based alignment is given as follows: first a subset of *k*-mers in both $S, S'$ are taken as *seeds* and exact seed matches between $S, S'$ called *anchors* are obtained. We only use *k*-mer seeds in this study, although other types of seeds are possible (Keich et al. 2004). An optimal increasing subsequence of possibly overlapping anchors based

on some score is then collected into a *chain*, where increasing is defined with the standard precedence relationship (Jain et al. 2022) between *k*-mer anchors (See Fig. 5A and subsection "Chaining" below). The chain is *extended* into a full alignment by aligning between anchor gaps in the chain.

**Model overview.** Our model of seed-chain-extend is primarily inspired by minimap2 with a few key differences. It captures the following steps: seeding the *query* $S'$, matching the *k*-mers to obtain anchors, sorting the anchors, chaining, and extending. We assume the *reference* $S$ has already been seeded and only the query needs seeding, as in the case of read alignment. For comparing two similar length genomes, the seeding time of either genome is comparable, so the asymptotics will be the same for comparative genomics.

**Runtimes.** Non-sketched seeding runtime with a hash table is $O(m)$, whereas sketched seeding runtime is $O(mk)$ (discussed in subsection "Sketching and local *k*-mer selection"). Letting $N$ be a random variable for the number of anchors, matching is $O(N + m)$ by iterating through a hash table, sorting is $O(N \log N)$, and (optimal) chaining is $O(N \log N)$ (see subsection "Chaining"). Extension is the only step with unknown time complexity. It will turn out that extension and $N \log N$ are the dominating asymptotic terms, so our goal is to bound these terms in expectation. Empirically, it has been shown that chaining and alignment usually take the most time (Kalikar et al. 2022) for read mapping.

**Chaining.** A chain is a sequence of tuples $\mathcal{C} = \big((i_1, j_1), \ldots, (i_u, j_u)\big)$ where $i_\ell$ and $j_\ell$ are the starting positions of the anchoring *k*-mers on $S$ and $S'$ respectively, under the convention that $S' = y_{p+1}$ so the *k*-mer labelled $(p + 1)$ on $S'$ is actually the first *k*-mer. The precedence relation $i_\ell > i_{\ell-1}$ and $j_{l\ell} > j_{\ell-1}$ must hold for all $\ell$, and *k*-mers can overlap. Our chaining score is the L1 or linear gap cost (Abouelhoda and Ohlebusch 2005; Li et al. 2020) of the form $u - \zeta[(i_u - i_1) + (j_u - j_1)]$ for $\zeta > 0$, which penalizes long chains from distant spurious anchors and is necessary for our

proofs when $n > m$. The score is sometimes defined equivalently as $u - \sum_{l=2}^{u} \zeta[(i_\ell - i_{\ell-1}) + (j_\ell - j_{\ell-1})]$. In the language of (Abouelhoda and Ohlebusch 2005), we let our anchor fragments have length $1$, so the *k*-mers can overlap. While minimap2 v2.22's default chaining score is different and uses a heuristic banded chaining approach, it does use a linear gap cost (without overlaps) in certain situations, e.g. mapping long contigs(Li 2021).

**Extension.** We use quadratic time extension between gaps based on any alignment score (e.g. edit distance, affine gap costs (Durbin et al. 1998)) as our optimality criterion only depends on the chain (see subsection “Homology and recoverability”). We do not extend past the ends of the chain and *do not* use banded alignment (Chao et al. 1992) in this step, unlike minimap2.

## Extension and chaining runtimes

Given sorted anchors, let $T_{Chain}$ be the time spent finding an optimal chain. $T_{Chain}$ depends on the objective function (Mäkinen and Sahlin 2020; Jain et al. 2022; Abouelhoda and Ohlebusch 2005; Otto et al. 2011). Since our gap costs are linear, $T_{Chain} = O(N \log N)$ where $N$ is the number of anchors(Abouelhoda and Ohlebusch 2005). For extension time $T_{Ext}$, let $(G_1, G_1'), \ldots, (G_{u-1}, G_{u-1}')$ be the size of the gaps for an optimal chain. $G_\ell$ indicates the length of the substring between the *k*-mers $i_\ell, i_{\ell+1}$ on $S$ and $G_\ell'$ similarly for $S'$; $G_\ell, G_\ell'$ can be zero. The extension time is $T_{Ext} = \sum_{\ell=1}^{u-1} O(G_\ell G_\ell')$. Using the fact that $\sum G_\ell \leq n$ and $G_\ell' \leq \sum G_\ell' \leq m$, one can get that $T_{Ext} = \sum_{\ell=1}^{u-1} O(G_\ell m) = O(nm)$. We will show that the expected runtime is better than this upper bound, but it serves as a useful worst case. Since $S, S'$ are random strings, $T_{Chain}, T_{Ext}, N$, and the alignment itself all become random variables. Our goal will be to bound $\mathbb{E}[T_{Chain}]$ and $\mathbb{E}[T_{Ext}]$.

## Theoretical results

First, a few definitions are in order. Recall that $|S| = n + k - 1 \sim n$, and $|S'| = m + k - 1 \sim m$ are our string lengths. We define $\sigma > 1$ to be the size of the alphabet and $0 < \theta < 1$ to be the probability a base mutates. Our theory holds for any alphabet size, but we use $\sigma = 4$ for

specifying numerical constants. Let $log = \log_\sigma$ with base $\sigma$ and $ln$ be the natural logarithm. Let $k = C \log n$ for a fixed $C > 0$ so that key quantities can be expressed interchangeably as $\sigma^k = n^C$ and $(1-\theta)^k = n^{-C\alpha}$ where $\alpha = -\log_\sigma(1-\theta) > 0$. We define the actual goodness of the chain in terms of *recoverability* in subsection "Homology and recoverability"; it measures the fraction of homologous bases under our mutation model that could potentially be recovered by extending through an optimal chain and *only depends on the chain*, not the actual alignment.

**Theorem 1 (Main result; no sketching).** *Under our model of seed-chain-extend in subsection "Modelling seed-chain-extend", assume* $\theta < 0.206$*, let* $\alpha = -\log_4(1-\theta)$*, and pick any* $C > \frac{2}{1-2\alpha}$*. If* $m = \Omega(n^{2C\alpha+\epsilon}) < n$ *for some arbitrary* $\epsilon > 0$*, letting* $k = C \log n$ *and* $\zeta = \frac{1}{6g(n)}$ *where* $g(n) = C\frac{50}{8}\log(n)\ln(n)\, n^{C\alpha}$*, the expected running time is* $O(mn^{C\alpha}\log(n))$ *for extension and* $O(mn^{-C\alpha}\log m)$ *for (optimal) chaining.* $C\alpha$ *can always be chosen to be* $< \frac{1}{2}$ *and the expected recoverability of any optimal chain is* $\geq 1 - O\left(\frac{1}{\sqrt{m}}\right)$.

The above condition on $C$ is equivalent to $C\alpha = \frac{2\alpha}{1-2\alpha} + \delta$ for any $\delta > 0$. In practice, we will make $\delta$ very small so $k$ is not too large. $\frac{2\alpha}{1-2\alpha}$ is plotted in Supplemental Fig. S7; it is convex and thus leads to the bound $C\alpha < 2.43 \cdot \theta$. E.g., if $|S| = |S'|$ and $\theta = 0.05$, which is approximately the level of divergence seen between human and chimpanzee genomes (Britten 2002), $\frac{-2\log_4(0.95)}{1+2\log_4(0.95)} < 0.08$, so for $C\alpha = 0.08$ the running time is $O(n^{1.08}\log(n))$. Even for relatively large values of $n$, say the entire human genome which has size of approximately 3 billion, $n^{1.08}$ is essentially linear as $(3{,}000{,}000{,}000)^{0.08} \sim 5.73$.

Our proof follows in three main steps. First, we bound the first and second moments of the random variable $N$, which denotes the number of anchors, implying that chaining is fast. Second, we use concentration inequalities for sums of *dependent* random variables and exploit the

structure of chaining to show that with high probability, an optimal chain does not deviate much from the chain with only "homologous anchors" (Fig. 5A). The failure probabilities will be on the order of $\Theta\left(\frac{1}{n}\right)$, allowing us to also bound everything in expectation. Lastly, we bound the expected runtime of extension through gaps between homologous anchors.

Asymptotically, the chaining runtime is smaller than the extension runtime. However, the implied constants can be much smaller for the extension term (Kalikar et al. 2022), so it is practically useful to reduce the runtime of chaining via sketching. Let $0 < 1/c \leq 1$ be the expected fraction of selected *k*-mers for a sketching method. We use the *open syncmer k*-mer seeding method (Edgar 2021) which has a useful mathematical property (Theorem 7), giving:

**Theorem 2 (Sketched result).** *In addition to the hypotheses outlined in Theorem 1, let* $c = O(\log n) < k$ *and* $\zeta = \frac{1}{6g'(n)}$ *instead where* $g'(n) = C\frac{200}{8}log(n)\,ln(n)\left(\frac{3}{2}n^{C\alpha} + 2\right)$. *For open syncmer sketched seed-chain-extend, the expected running time is* $O\left(min\left(\frac{1}{c^2}mn^{C\alpha}\,log^3(n), c \cdot mn^{C\alpha}\,log\,n\right)\right)$ *for extension and* $O\left(\frac{1}{c}mn^{-C\alpha}\,log\,m\right)$ *for chaining. The expected recoverability of any optimal chain is* $\geq 1 - O\left(\frac{1}{\sqrt{m}}\right)$.

This shows that the asymptotic upper bound on extension runtime is the same even if we let $c$ grow with $n$ like $c = \Theta(\log n) < k$, leading to the following conclusion: *sketching can reduce chaining time without increasing extension time much*. Other seeding schemes used, e.g. minimizers (Sirén et al. 2021; Rautiainen and Marschall 2020; Li 2018; Colquhoun et al. 2021) or FracMinHash (Irber et al. 2022) behave differently, but our techniques provide intuition and in some cases can be extended.

## Simulated genome alignment experiments

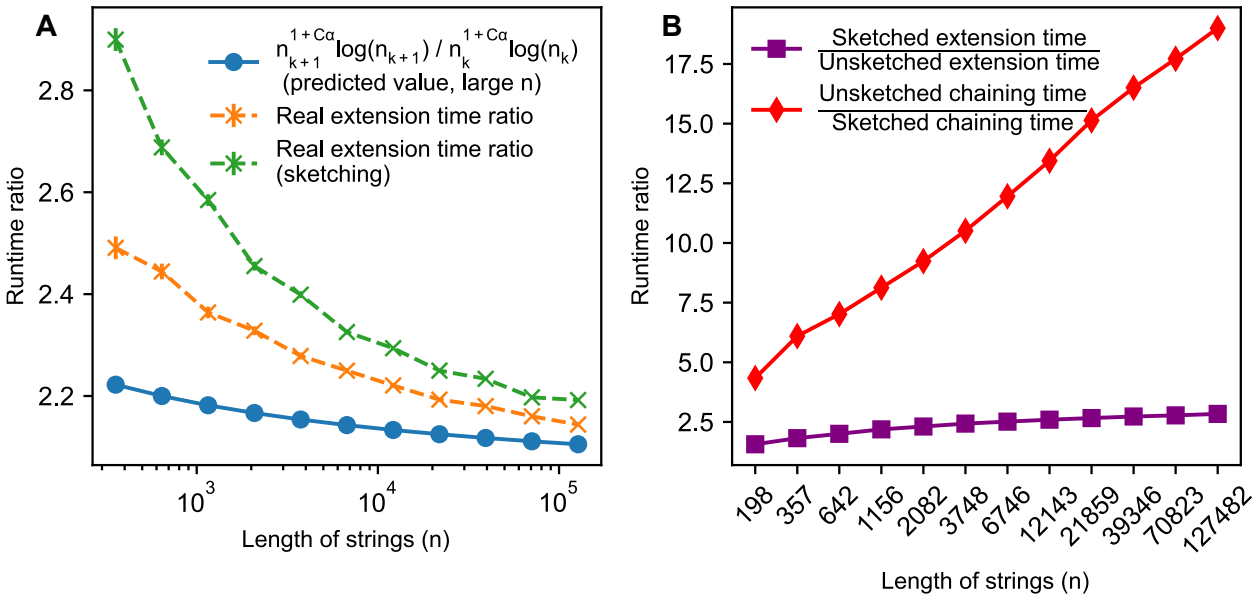
Although the model of seed-chain-extend we consider in the theory is based off of real aligners (e.g. minimap2), real aligners implement many additional tricks to make things work better. As such, to empirically validate our theory, we implemented a basic version of sketched seed-chain-extend and tested it on simulated random sequences with independent point substitutions. For the chaining step, we implemented an AVL tree based max-range-query method (Mäkinen et al. 2015; Li et al. 2020). For the extension step, we used a standard dynamic programming (DP) algorithm implemented in rust-bio (Köster 2016).

We let $k = C \log n$ where $C = \frac{2}{1-2\alpha}$, and do seed-chain-extend on two length $n$ sequences in this section, but we do an additional experiment for a substrings of length $m = n^{0.7} + 100$ in Supplemental Fig. S10. As $k$ cannot be fractional, we let $n_k = 4^{\frac{k}{C}}$ for varying integer values of $k$. We will set $c = k - 7 = C \log(n) - 7$ which *grows with n*. The constant 7 is arbitrary but chosen sufficiently large that so *s*-mers in the open syncmer method are not so small as to make the method degenerate. We found that recoverability was always quite high and that breaks did not occur very often in actual simulations; we show this in Supplemental Fig. S9. Thus, the primary focus of the empirical results will be runtime.

**Accuracy of asymptotic extension runtime predictions.** We first empirically investigate our upper bound on expected extension runtime, which is $O(n^{1+C\alpha} \log(n))$ for both sketched and non-sketched extension when $c = \Theta(\log n)$. Assuming that the runtime is simply $\lambda n_k^{1+C\alpha} \log(n_k)$ for some fixed constant $\lambda$, we can predict the ratio of the runtimes as $\frac{n_{k+1}^{1+C\alpha} \log(n_{k+1})}{n_k^{1+C\alpha} \log(n_k)}$. Of course, this is incorrect for small $n$, as smaller terms may dominate runtime, but we expect it to be reasonably accurate for large $n$. We plot the empirical and predicted ratios of extension runtimes in Fig. 1A.

Fig. 1A shows that our upper bound looks reasonable for both sketched and non-sketched cases for $\theta = 0.10$. In Supplemental Fig. S8, we show the same plots for $\theta = 0.05$. The empirical

231 results never cross the predicted ratio, but this is not too surprising as the predicted ratio is only
232 approximate. Importantly, the empirical extension runtime ratios slope downwards, which agrees
233 with the prediction.




234

235 **Figure 1.** Runtime results on seed-chain-extend between for two sequences of length $n$ and $k = C\log n = \frac{2}{1-2\alpha}\log(n) =$
236 $9{,}10,\ldots,19$ where $\theta = 0.10$ and $\alpha = -\log_4(1-\theta)$. (A) The y-axis shows the runtime ratios (with $95\%$ confidence
237 intervals) for iteration $k+1$ divided by runtime for iteration $k$ where the sequence length $n_k$ is plotted on the x-axis. As
238 $n$ grows, both the sketched and non-sketched extension ratios asymptotically approach the predicted ratio of
239 $n^{1+\frac{2}{1-2\alpha}\alpha}\log(n)$ runtimes. (B) Multiplicative speed-up for sketched versus non-sketched alignment when sketching with
240 density $\frac{1}{c} = \frac{1}{k-7}$ where $k = C\log n$. The slow-down for extension flattens out, whereas the chaining speed-up is almost
241 linear on the log scaled x-axis; chaining speed up dominates extension slow down as predicted by our theory.

242 **Sketching with $\Theta\left(\frac{1}{\log n}\right)$ density gives favourable runtime tradeoffs.** One of our key results in
243 Theorem 2 is that the upper bound on asymptotic runtime of extension does not depend on the
244 density as long as $1/c < k$, but chaining speed-up scales multiplicatively with $c$. In Fig. 1B, we
245 plot the multiplicative speed-up and slow-down of chaining and extension runtimes. This figure
246 shows why sketching is so effective in practice with respect to runtimes; the maximal extension
247 slow-down plateaus to $\leq 3$ times in this case, whereas the chaining speed up is linear as the

string grows exponentially. Therefore, it is worth sketching aggressively to reduce runtime if chaining is slow. One should however still be careful of the sensitivity loss due to sketching in practice(Shaw and Yu 2022).

In practice, because extension is heavily optimized (Marco-Sola et al. 2021; Farrar 2007; Suzuki and Kasahara 2018) and repetitive *k*-mers lead to more anchors, chaining can be a bottleneck even though it is asymptotically faster in runtime. In fact, we tried using WFA(Marco-Sola et al. 2021), a highly optimized algorithm for extension instead of the generic DP algorithm and found it was $\sim 60$ times faster than chaining without sketching for $n \sim 2{,}800{,}000, k = 23, \theta = 0.05$ in our implementation.

**Real nanopore read alignment experiments**

To test the applicability and generalizability of our theorems to real, non-simulated data, we performed an experiment by aligning real long-read data from Oxford Nanopore Technologies (ONT) using seed-chain-extend. We use the sketched seed-chain-extend aligner in the simulated experiments with a slight modification, which is explained in the next section. Importantly, our two main assumptions with respect to uniform random strings and point substitutions are violated in this setup: real genomes are not uniformly random strings, and ONT reads contain a significant amount of *indel errors* (Delahaye and Nicolas 2021).

Our data set consists of five reference genomes and corresponding long-read data sets. The references consist of a viral genome (SARS-CoV-2), a bacterial genome *(E. coli*), a fungi genome (*M. oryzae*), an insect genome (*D. melanogaster*), and a human genome (*H. sapiens*). For each genome, we aligned a corresponding set of publicly available ONT long-reads which may have different length and error distributions. The data sets can be found in Supplemental Table S1.

We are interested in testing the predicted results as a function of $m$ (the read length) and $n$ (the genome size). We let $k = C \log n = \frac{2}{1-2\alpha} \log n$ as before. Because $\alpha$ and $C$ in our results depend on the error rate $\theta$, we first aligned each read set to the genomes with minimap2 and retained only reads with gap-compressed identity $> 94.5\%$ and $< 95.5\%$ so that $\theta \approx 0.05$ is fixed. In addition, we only used reads for which minimap2 aligned $> 90\%$ of the read, as to avoid benchmarking on unalignable reads.

**Practical linear-gap cost consideration.** We use almost the same seed-chain-extend algorithm as before and fix the density to be $1/7$ (i.e. not scaling with $n$), except we apply one change: we found that the predicted linear-gap cost $\zeta(n)$ given by our theorems, while asymptotically appropriate in theory for large $n$, was too small for reasonable $n$. Therefore, we multiplied $\zeta$ by 1000.

This discrepancy is because the bounds we use in our proofs do not have optimized constants. The exact (non-asymptotic) value of $\zeta$ is only used in proving the recoverability result, and our change does not modify $\zeta$'s asymptotic behavior, which is used in other parts of our proofs. Thus, our runtime results are still technically valid, and our experiments below will also suggest that the recoverability result is also still valid.

**Evaluating model assumptions**. We first introduced the following filters to our basic seed-chain-extend alignment algorithm for computational reasons:

1. We filtered reads where the number of anchors is $> 10$ times the number of bases. This occurs due to extremely repetitive *k*-mers and can cause the chaining step to stall.
2. We did not consider reads where the chains had gaps of length $> 10000$ bp, which could occur due to structural variations or failure of chaining and can cause extension to stall.

The two filters also measure how badly our model assumptions, with respect to independent substitutions and uniform random strings, are violated. We report the number of

reads that fail the above filters in Table 1. Only a small fraction of reads fails the 10000 bp gap filter, but the repetitive *k*-mer filter is violated more frequently as the genomes become more complex. Still, $> 80\%$ of the human reads and $> 97\%$ of the *D. melanogaster* reads are not deemed too repetitive. Mapping to repetitive regions is an active area of research (Jain et al. 2020) which is out of the scope of this article, but effective heuristics such as repetitive *k*-mer masking (Sahlin 2022; Li 2018) exist.

| Genome | # reads with > 10 kb gap in chain | # reads with number of anchors > 10 times number of bases | Total # of reads |
|---|---|---|---|
| SARS-CoV-2 | 0 (0%) | 0 (0%) | 1692 |
| *E. coli* | 0 (0%) | 0 (0%) | 8223 |
| *M. oryzae* | 21 (0.39%) | 3 (0.06%) | 5424 |
| *D. melanogaster* | 25 (0.56%) | 116 (2.61%) | 4439 |
| *H. sapiens* | 11 (0.07%) | 3004 (19.48%) | 15419 |

**Table 1.** Counting the number of reads which did not pass our gap filter and repetitive *k*-mer filter. These two statistics measure how badly our model breaks down with respect to repetitiveness, large variation, and chaining failure. The total number of reads only counts a portion of the original reads that pass our initial filters, i.e. 90% of the read is aligned by minimap2 and the sequence divergence is $\mathbf{5}\% \pm \mathbf{0.5}\%$.

**Extension and chaining runtimes are well predicted by theory.** In Fig. 2, we plot the times of chaining and extension and perform a robust linear regression using the Siegel estimator (Siegel 1982) as a function of the read length $m$. For this plot specifically, we only timed alignments where the read was $> 90\%$ aligned by our seed-chain-extend implementation. For each plot, the genome size $n$ is held fixed. For fixed $n$, all runtimes are well-approximated by a linear function in $m$ as predicted by our theory, although chaining times for the human genome have higher variance due to the presence of repetitive *k*-mers.

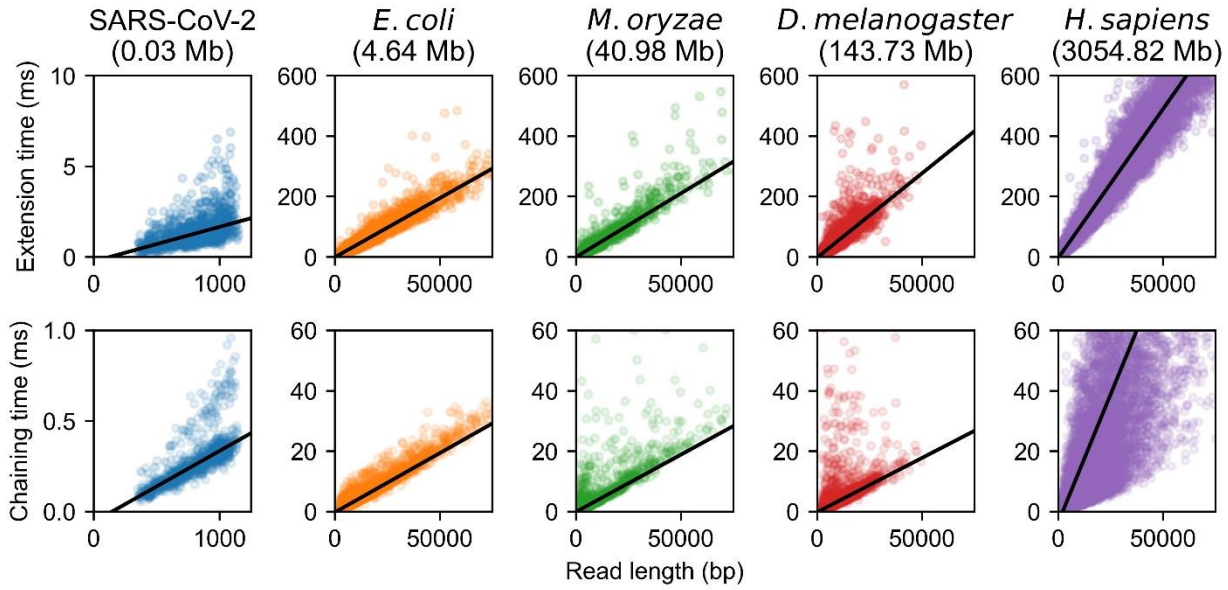
In Fig. 3, we plot the slopes of the regressions for extension time obtained from the first row of Fig. 2, which measures the dependence on $n$ given fixed $m$. This dependence is reasonably fit by a $\log n$ function ($R^2 = 0.766$), so $O(m \log n)$ is not a bad approximation for the runtime in practice. However, our theory leads to a $n^{c\alpha} \log n$ dependence, which comes from the

big O extension runtime. $C\alpha \sim 0.08$ when $\theta = 0.05$ in our setup, so by using a $n^{0.08} \log n$ function instead, we end up getting a better fit ($R^2 = 0.928$), as predicted by our theory.

**Recoverability upper bound by aligned fraction.** Recoverability is not measurable on real, non-simulated reads because true sequence homology is not known. However, we can upper bound recoverability by simply measuring the length of the chain relative to the read length, as any homologous bases outside of the chain (i.e. not within the first and last anchor) can not be recovered under our model. We call this the aligned fraction, which our theory also predicts should be $> 1 - O\left(\frac{1}{\sqrt{m}}\right)$ in expectation (with implied constants depending on $\theta$ and $n$). Practically, the aligned fraction is easily interpretable, and the ability to predict the aligned fraction is useful for ensuring that reads with a given error rate and *k*-mer size are long enough to align to a reference.

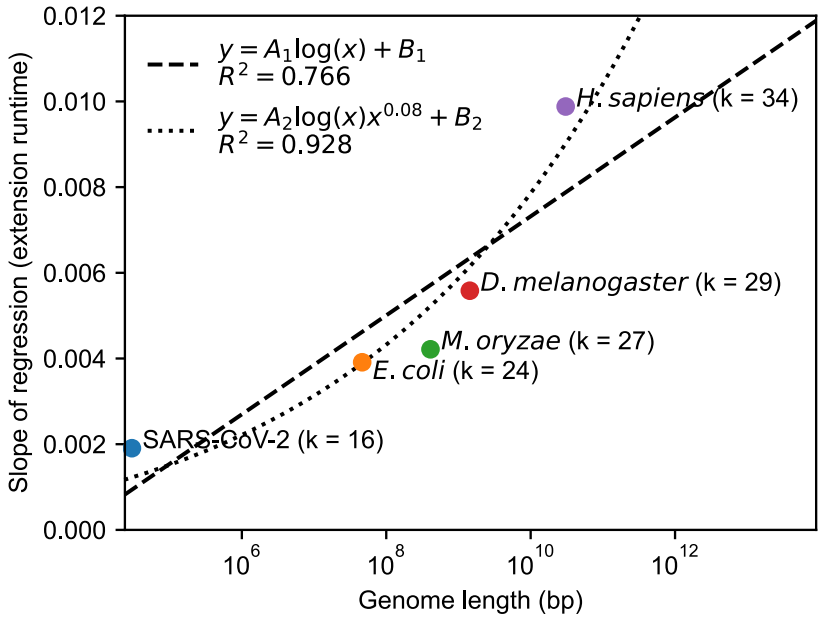
We also observed that the aligned fraction was the main cause of recoverability loss in the simulated experiments (see Supplemental Fig. S9), and it turns out that the main $O\left(\frac{1}{\sqrt{m}}\right)$ term in the theoretical recoverability result exactly comes from bounding the aligned fraction in the proof

329 (Supplemental Lemma S7), so this quantity is theoretically relevant as well.




330

331 **Figure 2.** Extension and chaining time for real ONT reads with approximately 5% sequence divergence. *k*-mer size
332 was chosen to be $k = C \log n$ for reference length $n$ and $C = \frac{2}{1-2\alpha} \approx 2.16$ with error parameter $\theta = 0.05$. Exact data
333 sets are described in Supplemental Table S1. Note the scaled axes for the SARS-CoV-2 reads, which were much
334 smaller than the other data sets. The well-fit linear regression lines indicate essentially linear runtime in read length
335 with fixed reference length $n$ (i.e., fixed $k = C \log n$ and constant $C$), although larger human chaining time variance is
336 due to repetitive *k*-mers.




337

338 **Figure 3.** Data points represent the slopes of the linear regressions in Fig. 2 for extension, with the corresponding value
339 of $k$ (which is $k = C \log n$ for a constant $C = \frac{2}{1-2\alpha}$). This dependence of the genome size, $n$ (x-axis), is decently
340 approximated by a naive $A_1 \log(n) + B_1$ fit where $A_1$ and $B_1$ are paraeters. However, our theory states that the
341 dependence should be $\log(n)\, n^{C\alpha}$ with $C\alpha \sim 0.08$ when $\theta = 0.05$. Fitting $A_2 \log(n)\, n^{0.08} + B_2$ gives a better $R^2$ value
342 (0.928 vs 0.766) with the same number of parameters (2 parameters for both fits), indicating the goodness of our
343 theoretical predictions.

344 In Fig. 4, we plot the aligned fraction for each data set. We perform a least-squares fit of
345 the function $1 - \frac{a}{m^b}$ where $a$ and $b$ are parameters. We exclude points with $< 0.25$ aligned
346 fraction if the read length is $> 1000$ when fitting because these points appear to be outliers
347 caused by poor chaining and can bias the least-squares fit. The resulting fits for all plots are
348 reasonable, with $R^2 > 0.5$ except for the *D. melanogaster* dataset. The seemingly small value of
349 $R^2$ can be explained by the fact that most reads (passing our filters) are well-aligned, so the data
350 is almost constant and $\approx 1$. Constant data will have $R^2 = 0$, so we can not do much better in this
351 case. Nevertheless, $b > 0.85$ on all data sets, suggesting that our $\sqrt{m}$ term may be too

conservative. However, the exact parameter values in the fit are highly sensitive to the fitting methodology, so the exact values displayed are only suggestive and not to be overinterpreted.

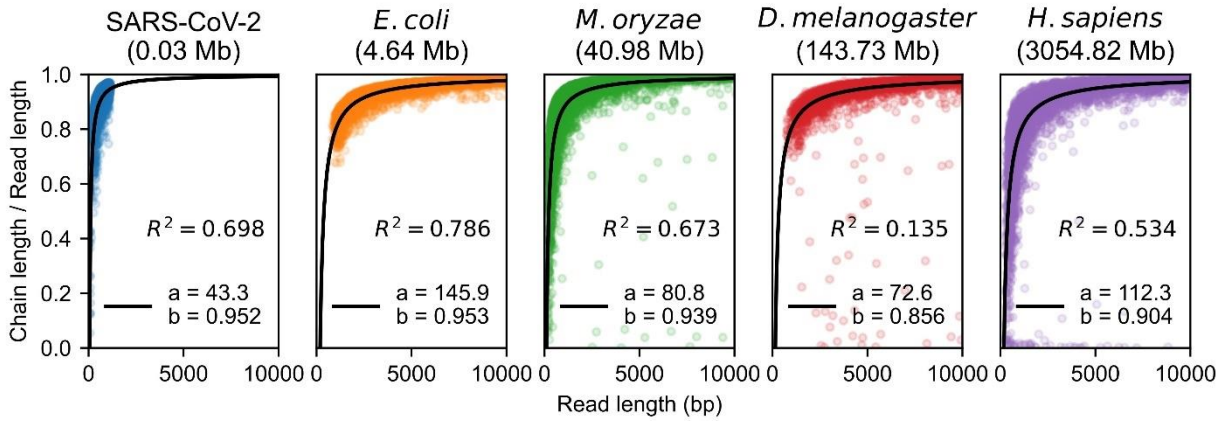



**Figure 4.** The aligned fraction of the aligned reads, which is chain length divided by read length. We fit a function of the form $1 - \frac{a}{m^b}$ where $m$ is the read length (x-axis) and display the resulting $R^2$ values. Aligned fraction is an upper bound for recoverability, and our lower bound on recoverability (and also aligned fraction) is $1 - O\left(\frac{1}{\sqrt{m}}\right)$, which the data suggests may be too conservative.

## Discussion

In this work, we are able to *rigorously* justify the empirical results seen by the seed-chain-extend heuristic through average case analysis under a simple mutation model. We showed the alignment is both accurate and fast: $\geq 1 - O\left(\frac{1}{\sqrt{m}}\right)$ fraction of the sequence homology is recoverable from the chain while only running in $O(mn^{C\alpha}\log(n))$ time, where for even a moderate mutation rate $\theta = 0.05$, $C\alpha < 0.08$. A recent aligner (Koerkamp and Ivanov 2022) empirically achieved this predicted $n^{1.08}$ runtime for $\theta = 0.05$ on two length $n$ sequences using similar but distinct techniques. In addition, we also proved that one can sketch to arbitrary densities $1/c$ where $c < k = O(\log n)$ while asymptotically decreasing runtime and without asymptotically decreasing recoverability, justifying the effectiveness of sketching. Because our set-up is modeled

by techniques used by practical software such as minimap2, our results provide a theoretical backing for why modern sequence alignment software actually performs so well in practice.

We verified our theoretical results in a synthetic experiment, which showed that our main theoretical predictions were valid for our random model. Specifically, our strongly sub-quadratic bounds predict runtime well, sketching can decrease chaining time without increasing extension time too much, and recoverability is high. Importantly, our results generalize past our uniform string and point substitution assumptions; we show that the runtimes for aligning real nanopore reads to references are still well-approximated by our theory. However, on more complex, repetitive genomes, our runtime predictions for chaining break down due to real *k*-mers being much more repetitive than under a uniform random model.

In terms of further work, we propose two possible directions. The first is to optimize the bounds obtained for the current random model proposed in this work, which assumes uniformly random strings and point substitutions. The second is to generalize the random model to model complex genomes more accurately, e.g. repetitive *k*-mers.

**Improving bounds for the current random model**

For runtime, it seems unlikely that the expected runtime is truly quasilinear due to the fundamental quantity $(1-\theta)^k = (1-\theta)^{C\log n} = n^{-C\alpha}$, the likelihood of a *k*-mer match, decreasing faster than logarithmically in $n$ when $k = \Theta(\log n)$; this turns out to be the cause of the $n^{C\alpha}$ term in the runtime (see Methods). In spite of this, we believe there can be significant improvements for our bounds in Theorems 1 and 2. The most significant aspect which we believe can be improved is the restriction on the constant $C$. $C$ is required to be $> \frac{2}{1-2\alpha}$, which seems unsatisfactory because this leads to high *k*-mer sizes, e.g. $k = 34$ for $\theta = 0.05$ on the human genome. This is much larger than the *k*-mer sizes used by noisy long-read aligners in practice (minimap2 defaults to $k = 15$).

The restricted values of $C$ is due to our analysis of spurious anchors relying on a relatively weak variance bound (Lemmas 3 and 4). To use the bound effectively, $C$ must be quite large. The variance bound is also the cause for the $1 - O\left(\frac{1}{\sqrt{m}}\right)$ bound for recoverability, which we also believe can be tightened based on both the simulated and real experiments. To surpass these bounds, a deeper understanding of spurious anchor subsequences must be developed. Spurious subsequences of anchors is a very similar problem to common subsequences in random strings for extremely large alphabets, a topic that has had much theoretical attention (Chvátal and Sankoff 1975; Kiwi et al. 2005; Lember and Matzinger 2009; Navarro 2001).

**Generalizing the random model**

Our current random model does not model complex genomes properly. It is well known that genomes can be extremely repetitive (Koning et al. 2011), which violates the uniform random string assumption. This is why the expected runtime of chaining deviates from our predictions in Fig. 2 for the human genome. One idea for modelling repeats is to use a *k*-mer Markov model for the random strings instead of an independent uniform string. However, the analysis of such models is highly non-trivial (Reinert et al. 2000), and our techniques for analyzing *k*-mers in this work may not apply. Regardless of how repeats are modelled, any sort of non-trivial analysis would still require a better handling of spurious anchor subsequences, which is exactly the issue that needs to be tackled in order to improve the bounds for our current model.

Another direction is to incorporate indels into the random model. The technical reason we opted for modelling point substitutions instead of indels is that indels complicate the statistics of *k*-mer matching. For example, an indel of the nucleotide A may not “change” the 4-mers in the sequence AAAAAA, and sequence homology becomes ambiguous. A formulation of an “indel channel” as a model of sequence mutation was used in (Ganesh and Sy 2020) for analysis of edit

distance, but the resulting analysis for this model was more complex than in the independent substitution case.

Generalizing our model to small indels seems less pressing as our theory was successful for real nanopore data, which has many small indel errors, showing that the substitution model already generalizes well to small indels. However, modelling large structural variations could be a relevant problem.

## Methods

### Homology and recoverability

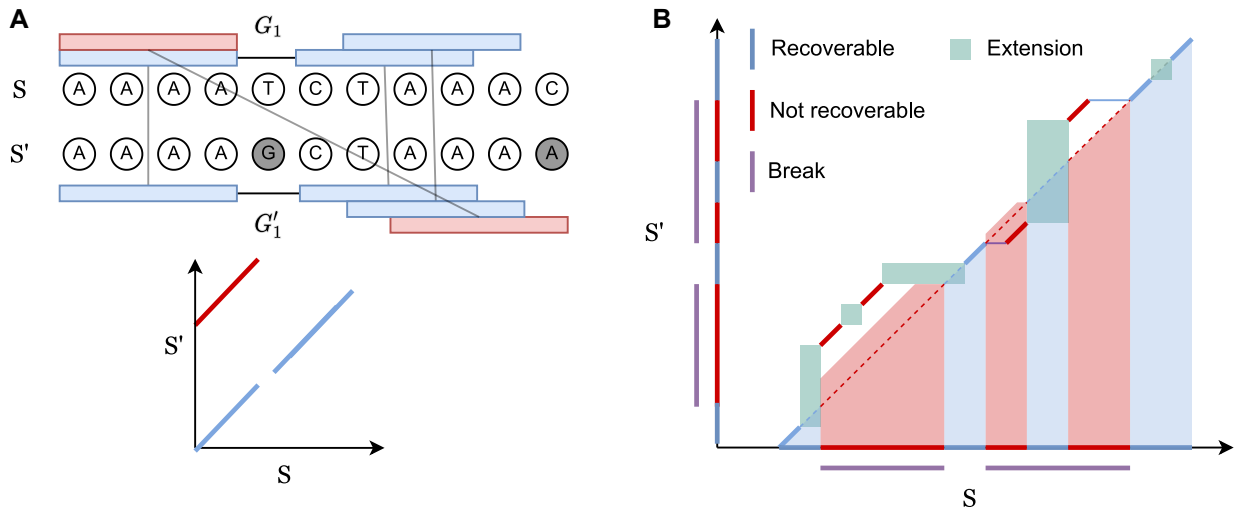
Let the interval $[a..b]$ be the discrete interval $\{a, a+1, \dots, b\}$ and $S[a..b]$ be the substring indexed by $[a..b]$. $S'$ is a mutated version of the substring $S[p+1..p+m]$ of length $m$ starting at $p+1$, so the optimal alignment should map $S'$ to the homologous indices $[p+1..p+m]$. We now define *recoverability* as the number of homologous bases one can possibly recover by seed-chain-extend; a visual example is shown in Fig. 5.

**Definition 1.** *Let an alignment matrix be given as* $[1..|S|] \times [p+1..p+|S'|]$. *Define the homologous diagonal* $D_H = \{(p+1, p+1), \dots, (p+|S'|, p+|S'|)\}$. *Given a chain* $\mathcal{C}$, *let* $Align(\mathcal{C})$ *be the set of possible alignments obtained by extending through gaps and k-mer matching, where* $(x, y) \in Align(\mathcal{C})$ *indicates* $x$ *and* $y$ *can be aligned. Then the set of recoverable bases is* $Align(\mathcal{C}) \cap D_H$ *and the recoverability is* $R(\mathcal{C}) = \frac{|Align(\mathcal{C}) \cap D_H|}{|S'|}$.

We define $Align(\mathcal{C})$ carefully in Section B in the Supplemental Materials and give a visual explanation in Fig. 5B. Note that an optimal chain with respect to our linear gap cost objective

436 $u - \zeta[(i_u - i_1) + (j_u - j_1)]$ does not directly optimize for recoverability. We provably find the chain
437 with the optimal linear gap cost, and then we will argue that it leads to good recoverability.

438 Traditionally, alignment optimality is defined using a generalized edit distance (Vingron
439 and Waterman 1994). However, these distances are only used as a proxy for detecting sequence
440 homology (Batzoglou 2005; Thorne et al. 1991; Lunter et al. 2005). We know the true underlying
441 sequence ancestry in our model, so defining optimality with respect to sequence homology suits
442 the actual goal of sequence alignment. The reason for the name "recoverability" is that extension
443 could *potentially* recover all recoverable bases, but this depends on the extension algorithm and
444 is not guaranteed (Lunter et al. 2008).




445

446 **Figure 5.** Seed-chain-extend visualized on an alignment matrix. Anchors are short diagonal matches in the matrix.
447 Extension corresponds to performing dynamic programming (DP) on the sub-matrix between anchors. (A) *k*-mer
448 anchors under mutations and their corresponding alignment matrix. Blue anchors are *homologous* anchors while red
449 are *spurious* anchors. (B) $Align(\mathcal{C})$ consists of anchors and DP matrices. Recoverable bases correspond to the bases
450 on the homologous diagonal where a *k*-mer lies or is accessible by the green DP matrix between gaps. Breaks cover
451 non-recoverable sections.

452 Under our model, the trivial $O(1)$ alignment that aligns $S'$ back to the originating substring
453 of $S$ without indels is the most homologous. This may seem to make our results superfluous;

however, remember that the algorithm does not know where $S'$ "begins" if $m < n$. Also, while we do not attempt the case with indels, seed-chain-extend is still valid when indels are present whereas the trivial solution does not allow for indels.

We will lower bound $\mathbb{E}[R(\mathcal{C})]$. To do this, we will work with a more natural object called a *break*.

**Definition 2 (inspired by** (Ganesh and Sy 2020)**).** *We call matching bases and anchors of the form* $(x, x)$ *homologous and spurious otherwise. Given a chain* $\big((i_1, j_1), \ldots, (i_u, j_u)\big)$ *and a maximal interval* $[p..q]$ *such that* $(i_p, j_p), \ldots, (i_q, j_q)$ *are all spurious anchors, define the break* $B$ *as* $B = \big[min(i_p, j_p)..max(i_q, j_q) + k - 1\big]$. *Let the length or size of a break be* $L(B) = max(i_q, j_q) - min(i_p, j_p) + k$ *and* $L(\mathcal{C}) = \sum_B L(B)$ *be the total length over all breaks.*

**Lemma 1.** *Given any chain* $\mathcal{C} = \big((i_1, j_1), \ldots, (i_u, j_u)\big)$*, we have that* $\big(j_u - j_1 - L(\mathcal{C})\big)/|S'| \leq R(\mathcal{C})$.

We prove Lemma 1 in the Supplemental Materials. The concept of a break is illustrated in Fig. 5B. Breaks cover non-recoverable regions, so subtracting the breaks from the span of the anchors lower bounds the recoverability.

## Fundamental tools and bounds

In this section, we describe some fundamental tools for dealing with pairs of random mutating strings. We first need to give a careful probabilistic exploration of random *k*-mer anchors on $S = x_1 x_2 \ldots$ and $S' = y_{p+1} y_{p+2} \ldots$. This requires a bit of work due to the dependence between the random strings $S, S'$. For the rest of the paper, missing proofs can be found in the Supplemental

Materials. Supplemental Figures S2-S6 are visual aids for our proofs and can also be found in the Supplemental Materials.

**Definition 3.** *Let* $M(i,j)$ *be random variables such that* $M(i,j) = 1$ *if* $x_i = y_j$ *and* $0$ *otherwise. Let* $A(i,j)$ *be a random variable* $\prod_{l=0}^{k-1} M(i+\ell, j+\ell)$. $A(i,j)$ *is an indicator random variable for the presence of a k-mer anchor at positions* $(i,j)$*. We will also refer to* $A(i,j)$ *variables as "anchors".*

We would like for $M(i,j), M(i+1,j+1)$, to be independent, so finding the probability of *k*-mer matches is easy. However, in our model, it is not actually obvious *a priori* that $M(i,j), M(i+1,j+1)$ are independent when $j \neq i$. Consider $M(1,2)$ and $M(2,3)$. The former is a function of $x_1, y_2$, and the latter $x_2, y_3$. However, $x_2$ and $y_2$ are dependent in our model, so more

work is needed. A convenient graphical representation of independence for the $M$ random variables is the *match graph*, which we define below.

**Definition 4.** *The match graph of a set of random variables of the form* $\mathcal{M} = \{M(i_1, j_1), M(i_2, j_2), \dots\}$ *is a graph* $G(\mathcal{M}) = (V, E)$ *where the vertices* $V$ *are the letters* $x_1, \dots, x_{n+k-1}, y_{p+1}, \dots, y_{p+m+k-1}$, *and the edges*

$$E = \{(x_i, y_i) : i \in [p+1 \,..\, p+m+k-1]\} \cup \{(x_h, y_l) : M(h, l) \in \mathcal{M}\}.$$

*In particular, the match graph is bipartite for the sets* $\{x_1, \dots, x_{n+k-1}\}$ *and* $\{y_{p+1}, \dots, y_{p+m+k-1}\}$.

A match graph is shown in Supplemental Fig. S1. The main reason for defining the match graph is the following theorem, which allows us to graphically determine independence of the $A$ variables.

**Theorem 3.** *The random variables* $\mathcal{M} = \{M(i_1, j_1), M(i_2, j_2), \dots\}$ *where* $i_\ell \neq j_\ell$ *for all* $M(i_\ell, j_\ell) \in \mathcal{M}$ *are independent if the induced match graph has no cycles.*

**Corollary 1.** *If* $i \neq j$, $\Pr(A(i,j) = 1) = \frac{1}{\sigma^k}$. *Otherwise,* $\Pr(A(i,i) = 1) = (1 - \theta)^k$.

We will denote the random variables $\sum_{i=p+1}^{p+m} A(i,i) = \sum_i A(i,i) = N_H(n,m) = N_H$ and $\sum_{i \neq j} A(i,j) = N_S(n,m) = N_S$ respectively as the number of homologous anchors and spurious anchors; we will drop the dependence on $n, m$ to simplify notation. These are key random

variables that we wish to bound later on. The below theorem follows directly from Corollary 1 by linearity of expectation.

**Theorem 4.** $\mathbb{E}[\sum_i A(i,i)] = \mathbb{E}[N_H] = m(1-\theta)^k$, and $\mathbb{E}\left[\sum_{i,j,i\neq j} A(i,j)\right] = \mathbb{E}[N_S] = m(n-1)\frac{1}{\sigma^k}$. In particular, $\mathbb{E}[N] = \mathbb{E}[N_H + N_S] = m(1-\theta)^k + m(n-1)\frac{1}{\sigma^k}$.

## Bounding sums of *k*-mer random variables

We proceed to bound the *distribution* of the random variables $N_S$ and $N_H$, which are sums of *dependent* random variables. We first bound $N_S$ by computing the second moments and using variance-based bounds. To do this, we need to examine the independence structure of the $A(i,j)$ random variables.

**Lemma 2.** *For $A(i,j)$ and $A(h,l)$, if both of the following conditions hold:*

1. $|i-h| \geq k$ *or* $|j-l| \geq k$ *and*
2. $|i-l| \geq k$ *or* $|j-h| \geq k$,

*then the induced match graph on the $M$ variables for $A(i,j)$ and $A(h,l)$ has no cycles.*

Intuitively, the first condition states that two anchors do not overlap too much, e.g. $A(1,1)$ and $A(2,2)$ are not independent when $k = 3$. The intuition behind the second condition can be illustrated by the following situation where $k = 1$ and $\theta \sim 0$: consider the anchors $A(1,5), A(5,1)$. Since it is likely that $x_1 = y_1$ and $x_5 = y_5$, if $x_1 = y_5$ then $x_5 = y_1$ with high probability, so $x_1 = y_5$ is not independent of $x_5 = y_1$.

**Corollary 2.** *If $A(i,j)$ and $A(h,l)$ satisfy Lemma 2, they are independent.*

*Proof.* By condition (1) in Lemma 2, $A(i,j)$ and $A(h,l)$ do not share any $M$ variables i.e. $A(i,j) = M(i,j)M(i+1,j+1)$ and similarly for $A(h,l)$, but no product shares a variable with the others.

Since the match graph has no cycles under these conditions, by Theorem 3, all $M$ variables are independent so $\Pr(A(i,j)A(h,l)=1)=\frac{1}{\sigma^{2k}}=\Pr(A(i,j)=1)\Pr(A(h,l)=1)$ as desired.

We can also bound expected values of $N_H^2, N_S^2$, and $N_S N_H$, giving us variance estimates.

**Lemma 3.** $\mathbb{E}[N_S^2] \leq 8k^2 mn\frac{1}{\sigma^k}+\mathbb{E}[N_S]^2$. Thus the variance $Var(N_S)$ can be upper bounded by $8k^2\frac{mn}{\sigma^k}$. Furthermore, $\mathbb{E}[N_H^2] \leq 2mk(1-\theta)^k + m^2(1-\theta)^{2k}$ and $\mathbb{E}[N_H N_S] \leq 4k\frac{mn}{\sigma^k}+m^2 n(1-\theta)^k\frac{1}{\sigma^k}$.

Now we can use the variance bound and Chebyshev's inequality to get the result below. Note the bound uses $k = C\log n$; we will prefer this form for quantities directly used for proving the main result. The label F1 in the theorem refers to the particular event space for which the bound always holds. We will label each proposition that holds with high probability with the event

space that we are operating in. We will continue this convention for the rest of the paper as this will be useful when computing our final bounds.

**Lemma 4 (F1).** *With probability* $\geq 1 - \frac{1}{n}$, *the number of spurious anchors is* $\leq n^{2-C} + \sqrt{8mC}\log(n)\, n^{1-C/2}$. *That is,*

$$\Pr\left(N_S \geq n^{2-C} + \sqrt{8mC}\log(n)\, n^{1-C/2}\right) \leq \frac{1}{n}.$$

*Proof.* Use Chebyshev's inequality with Lemma 3's bound on $Var(N_S)$ along with the inequality $\frac{m(n-1)}{\sigma^k} < n^{2-C}$.

If $C > 3$, then for large $n$, $N_S = 0$ with high probability, and our analysis would be easy. However, we want $C$ as small as possible. It turns out we can make $C \sim 2$ for reasonable $\theta$, significantly tightening our bounds.

For $N_H$ we can get a stronger exponential bound because of its independence structure. $A(i,i)$s, which we call homologous anchors, are only dependent in a small neighbourhood around $i$ of size $k$ because *k*-mers on non-overlapping substrings are independent. This is called $k$ *–dependence* (not to be confused with *k-independence*) and is used in (Blanca et al. 2022) to show $N_H$ is asymptotically normal. Concentration bounds can also be translated in the $k$ –dependent scenario (Janson 2004).

**Theorem 5. (Dependent Chernoff-Hoeffding bound – Corollary 2.4 from** (Janson 2004) **reworded and simplified).** *Suppose we have* $X = \sum_{a\in\mathcal{A}} Bernoulli_a(q)$ *for some* $0 < q < 1$. *A proper cover of* $\mathcal{A}$ *is a family of subsets* $\{\mathcal{A}_i\}_{i\in I}$ *such that all random variables in* $\mathcal{A}_i \subset \mathcal{A}$ *are*

*independent and* $\bigcup_{i\in I}\mathcal{A}_i = \mathcal{A}$. *Let* $\chi(\mathcal{A})$ *be the minimum size of the cover,* $|I|$, *over all possible proper covers. Then for* $t \geq 0$,

$$\Pr(X \leq \mathbb{E}X - t) \leq \exp\left(-\frac{8t^2}{25|\mathcal{A}|\chi(\mathcal{A})q}\right).$$

**Lemma 5.**

$$\Pr\left(N_H \leq m(1-\theta)^k - t\right) \leq \exp\left(-\frac{8t^2}{25mk(1-\theta)^k}\right)$$

*Proof.* We simply use Theorem 5 with $q = (1-\theta)^k$. By *k*-dependence, we can easily see that $\mathcal{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$ where $\mathcal{A}_j = \{A(j,j), A(j+k, j+k), A(j+2k, j+2k) \ldots\}$ is a partition satisfying the independence condition, and we will have at most $k$ sets. Thus $\chi(\mathcal{A}) \leq k$, and we're done.

**Proof of non-sketched main result**

To prove the main result on seed-chain-extend without sketching, we will need to bound three quantities in expectation: the runtime of chaining, the recoverability of chaining, and the runtime of extension.

We first bound $O(\mathbb{E}[N \log N])$, the expected runtime of chaining (and also anchor sorting). Note that $\mathbb{E}[N] \leq \frac{1}{n^{C-2}} + mn^{-C\alpha}$. We would like $\mathbb{E}[N \log N] = O(\mathbb{E}[N] \log \mathbb{E}[N]) = O(mn^{-C\alpha} \log m)$ to hold. Unfortunately, Jensen's inequality only gives $\mathbb{E}[N \log N] \geq \mathbb{E}[N] \log \mathbb{E}[N]$ because $x \log x$ is convex. However, with a bit more work:

**Theorem 6.** *Assume* $m = \Omega(n^{2C\alpha+\epsilon})$ *for some* $\epsilon > 0$, *and* $C > \frac{1}{1-\alpha}$. *Letting* $N$ *be the total number of k-mer anchors,* $\mathbb{E}[N \log N] = O(\mathbb{E}[N] \log \mathbb{E}[N]) = O(mn^{-C\alpha} \log m)$. *Thus, the runtime of chaining is* $O(mn^{-C\alpha} \log m)$.

Now we bound the expected recoverability of the chain. Given $S$ and $S'$, let a *homologous gap* of size $\ell + k - 1$ bases be an interval of $\ell$ consecutive *k*-mers for which no homologous

anchors exist (i.e. the *k*-mers are mutated). In the context of a chain, a homologous gap will refer to a gap flanked by two homologous anchors. Technically, if $\ell$ consecutive *k*-mers are uncovered but are flanked by two homologous anchors, this gives $\ell - k + 1$ uncovered bases. We will ignore these factors of $k$ as we will show that they are asymptotically small. It turns out homologous gaps grow relatively slowly in $n$ with high probability.

**Lemma 6.** *With probability* $\geq 1 - 1/n$*, no homologous gap has size greater than*

$$g(n) = \frac{50k}{8(1-\theta)^k}\ln(n) = \frac{C \cdot 50}{8}\log(n)\ln(n) \cdot n^{C\alpha}$$

*plus a small* $C \log n$ *term we will ignore because it is small asymptotically.*

In a chain, gaps may also be flanked by one or two spurious anchors. We call these *non-homologous gaps*. We first bound break lengths, which will imply good recoverability, and bound non-homologous gaps later on.

**Lemma 7 (F1 + F2).** *Take any* $C > \min\left(3, \frac{2}{1-2\alpha}\right)$ *and let* $\zeta = \frac{1}{6g(n)}$ *where* $g(n) = C\frac{50}{8}\log(n)\ln(n)\,n^{C\alpha}$*. Assume* $m = \Omega(n^{2C\alpha+\epsilon})$ *for some* $\epsilon > 0$*. Then for large enough* $n$*, there are no breaks of length* $\geq m^{1/2}$ *with probability greater than* $1 - 2/n$ *in an optimal chain.*

**Corollary 3.** *Under the same assumptions as in Lemma 7}, the expected recoverability of any optimal chain is* $\geq 1 - O\left(\frac{1}{\sqrt{m}}\right)$.

The idea behind proving the above propositions is to work in a space of events F1 ∩ F2 where “bad events” do not occur, and any optimal chain has good recoverability. Because this

space of bad events is small, they do not contribute to our expected value too much. This finishes the recoverability result for the main theorem.

The last step is to bound extension running time, and this comes down to separately bounding the size of the homologous and non-homologous gaps in any optimal chain. We bound the runtime of extension through homologous gaps by directly calculating the expectation through *all possible* homologous gaps. We then show that the runtime through non-homologous gaps is small and does not contribute to the asymptotic term.

**Lemma 8.** *Let $T_{Ext}^{H}$ be the time of extension over only the homologous gaps of any optimal chain.* $\mathbb{E}[T_{Ext}^{H}] = O(mn^{C\alpha}\log n)$.

**Lemma 9.** *Let $T_{Ext}^{S}$ be the runtime of extension through only the non-homologous gaps of an optimal chain. Under the same assumptions as in Lemma 7,* $\mathbb{E}\left[T_{Ext}^{S}\right] = O(m)$.

Now we have enough results to prove the Theorem 1.

*Proof (Theorem 1).* The expected runtime of chaining follows from Theorem 6. The recoverability result follows from Corollary 3. The expected runtime of extension is $\mathbb{E}[T_{Ext}] = \mathbb{E}[T_{Ext}^{H}] + \mathbb{E}\left[T_{Ext}^{S}\right]$, and $\mathbb{E}[T_{Ext}^{H}], \mathbb{E}\left[T_{Ext}^{S}\right]$ are both $O(mn^{C\alpha}\log(n))$ by Lemmas 8 and 9. This completes the proof as long as we satisfy the assumptions of Lemma 7 and Theorem 6 on $C, \alpha$. To satisfy the assumptions, we require $C > \frac{1}{1-\alpha}$, $C > \min\left(3, \frac{2}{1-2\alpha}\right)$, and $C\ \alpha < 1/2$ otherwise $m = \Omega(n^{2C\alpha+\epsilon}) > n$ for large enough $n$. It's not hard to check that the limiting condition is $\alpha < 1/6$, so we require

$-\log_4(1-\theta) < 1/6$. This works out to be $\theta < 1 - 4^{-\frac{1}{6}} < 0.2063$. We can also remove the minimum condition on $C$ because $\alpha < 1/6$ implies $\frac{2}{1-2\alpha} < 3$.

## Sketching and local *k*-mer selection

Now consider not selecting all of the *k*-mers in a string, but only a subset of them during the initial seeding step. This allows one to chain only a subset of the *k*-mers, potentially providing runtime savings.

We use the *open syncmer* method (Edgar 2021). Given a string, we take all *k*-mers of the string and break the *k*-mer into *s*-mers with $s < k$. There are $k - s + 1$ *s*-mers in the *k*-mer. We select or *seed* the *k*-mer if the smallest *s*-mer (subject to some ordering, which we choose as uniform random) is in the $\lceil \frac{k-s+1}{2} \rceil$-th 1-indexed position; we call a selected *k*-mer an *open syncmer*. Given repeated smallest *s*-mers, we take the rightmost one to be the smallest. Finding the smallest *s*-mer among the $k - s + 1$ *s*-mers in a *k*-mer takes $k - s + 1$ iterations, so finding all open syncmer seeds in $S'$ takes $O\big((k - s + 1)m\big) = O(mk) = O(m \log n)$ time.

The expected fraction of selected *k*-mers over a string with i.i.d uniform letters is called the $density$, and it is $\frac{1}{k-s+1}$ for the open syncmer method (up to a small error term $O\left(\frac{(k-s+1)^2}{\sigma^s}\right)$ which we will ignore; see (Zheng et al. 2020)). We will let $c$ be the reciprocal of the density, so $c = (k - s + 1)$.

The original open syncmer definition had a parameter $t$ where a *k*-mer was selected if the smallest *s*-mer was in the $t$ –th position; we proved in (Shaw and Yu 2022) that the optimal $t$ is $\lceil \frac{k-s+1}{2} \rceil$ with respect to maximizing the number of conserved bases from *k*-mer matching. The

reason we choose open syncmers is primarily to the following fact which was shown in (Edgar 2021):

**Theorem 7.** *Define* $t = \left\lceil \frac{c}{2} \right\rceil = \left\lceil \frac{k-s+1}{2} \right\rceil$. *If* $k - s + 1$ *is odd, two consecutive open syncmers must have starting positions* $\geq t$ *bases apart. If even, they must have starting positions* $\geq t - 1$ *bases apart.*

Theorem 7 follows by examining the smallest *s*-mer in a *k*-mer and noticing that in the next overlapping *k*-mer, the locations for the new smallest *s*-mer are restricted. This theorem is the reason we use open syncmers and is crucial to our proofs. The spacing property makes selected open syncmers a polar set (Zheng et al. 2021); other methods also give rise to polar sets (Frith et al. 2020, 2023) but open syncmers seem to perform well empirically (Dutta et al. 2022; Shaw and Yu 2022; Frith et al. 2023) and are easy to describe. For the rest of the section, we will assume $c = k - s + 1$ is odd, so $\left\lceil \frac{c}{2} \right\rceil = \frac{c+1}{2}$.

Let $A(i,j)$ be the random variables as defined before. Let $J(i)$ be the indicator random variable for if the $i$th *k*-mer is selected on $S$ as an open syncmer, and $J'(j)$ similarly for the $j$-th *k*-mer on $S'$. We now wish to calculate $\mathbb{E}[J(i)J'(j)A(i,j)]$, the probability that a *k*-mer match exists and the *k*-mer is an open syncmer.

**Lemma 10.** *If* $i = j$, *then* $\mathbb{E}[J(i)J'(j)A(i,j)] = \frac{(1-\theta)^k}{c}$. *Otherwise,* $\mathbb{E}[J(i)J'(j)A(i,j)] = \frac{1}{c\sigma^k}$.

*Proof.* If $A(i,j) = 1$, then the $i$-th *k*-mer and $j$ −th *k*-mer are the same. If a *k*-mer is selected as an open syncmer on $S$, it must also be selected on $S'$, so $\mathbb{E}[J(i)J'(j)A(i,j)|A(i,j) = 1] = \mathbb{E}[J(i)] = \frac{1}{c}$. $A(i,j)$ and $J(i)$ are independent because we assume the random ordering for the *s*-mers is

independent of the random mutations. Using the law of total expectation and Theorem 4 gives the result for both cases.

**Definition 5.** *We will replace all random variables involving anchors and matches with a superscript $*$ to indicate sketched seeds, e.g. $A(i,j)^* = A(i,j)J(i)J'(j)$, $M^*$, $N_S^*, N_H^*$, etc.*

**Corollary 4.** The expected total number of anchors after applying open syncmer seeds with density $\frac{1}{c}$ is

$$\mathbb{E}\left[\sum_{i,j} A(i,j)^*\right] = \frac{1}{c}\left(n(1-\theta)^k + m(n-1)\frac{1}{\sigma^k}\right).$$

The above follows directly from Lemma 10. As expected, subsampling to a $\frac{1}{c}$ fraction of the *k*-mers gives $\frac{1}{c}$ expected hits. Note the important property of *context independence* used in

the proofs: if $A(i,j) = 1$, then $J(i) = J'(j)$. This property is not satisfied if one were sampling *k*-mers randomly or using minimizers (Roberts et al. 2004).

We now deduce the sketched moment bounds on $N^*$:

**Lemma 11.** *The variance* $Var(N_S^*)$ *can be upper bounded by* $\frac{1}{c}8k^2mn^{1-C}$. *Furthermore,* $\mathbb{E}[N_H^{*2}] \leq \frac{1}{c}2mk(1-\theta)^k + \frac{1}{c^2}m^2(1-\theta)^{2k}$ *and* $\mathbb{E}[N_H^* N_S^*] \leq \frac{1}{c}4k\frac{mn}{\sigma^k} + \frac{1}{c^2}m^2n(1-\theta)^k\frac{1}{\sigma^k}$.

**Lemma 12 (F1*).** $\Pr\left(N_S^* \geq \frac{1}{c}mn^{1-C} + \sqrt{\frac{8}{c}}km^{1/2}n^{1-C/2}\right) \leq \frac{1}{n}$

The sketched moment bounds allow us to start re-analyzing the crucial propositions in subsection “Proof of non-sketched main result” but in the context of sketching. The first main result is that sketching reduces the chaining time as one would expect.

**Theorem 8.** Under the same assumptions as in Theorem 6, letting $N^*$ be the total number of sketched *k*-mer anchors, the expected chaining time is $O(\mathbb{E}[N^* \log N^*]) = O\left(\frac{1}{c}mn^{-C\alpha}\log m\right)$.

The second result we wish to highlight is the new bound on extension runtime through homologous gaps.

**Lemma 13.** The expected runtime of sketched extension through the homologous gaps in an optimal chain is $\mathbb{E}[T_{Ext}^{H*}] = O\left(min\left(\frac{1}{c^2}mn^{C\alpha}\log^3(n), c\cdot mn^{C\alpha}\log \mathrm{n}\right)\right)$. If $c = \Theta(\log n)$ then $\mathbb{E}[T_{Ext}^{H*}] = O(mn^{C\alpha}\log n)$.

It will turn out that in the sketched case, extension over homologous gaps also dominates runtime. Since we sketch with density $\frac{1}{c}$, this result states that we can sketch with *decreasing density* and have the *same* asymptotic extension time as without sketching, which is surprising. The crucial fact we use in the proof of Lemma 13 is Theorem 7, which shows that open syncmer seeding weakens the *k*-dependence of the seeds since they are now at least $(c+1)/2$ bases

apart. This tightens the bound in Theorem 7 and also shows that the sketched maximum homologous gap size is $\Theta\big(g(n)\big)$, with $g(n)$ as in Lemma 6.

If we used other methods such as closed syncmers (Edgar 2021) or FracMinHash (Irber et al. 2022; Hera et al. 2022), then we would recover the $O(c \cdot mn^{C\alpha} \log n)$ result in Lemma 13 but not the $O\left(\frac{1}{c^2} mn^{C\alpha} \log^3(n)\right)$ result. Thus, we have saved a $log$ factor by using open syncmers when we let $c = \Theta(\log n)$. The bulk of Section E in the Supplemental Materials is dedicated to proving Lemma 13.

The key quantities used for the rest of the proofs are $N_S^*$ and $g'(n)$, the new maximum homologous gap size. We will prove $g'(n) = \Theta\big(g(n)\big)$ with $g(n)$ as in Lemma 6, so the only asymptotic difference in these bounds is a factor of $\frac{1}{\sqrt{c}}$ in $N_S^*$. Since we want $N_S^*$ to be small, this does not affect downstream analysis. It follows that the proofs of the rest of the lemmas in subsection “Proof of non-sketched main result” can be replicated almost verbatim. We give a sketch of these proofs in the Supplemental Materials.

## Software availability

Scripts for regenerating the simulated experiments are available at https://github.com/bluenote-1577/basic_seed_chainer/. The aligner used for the real nanopore

691 experiments is available at https://github.com/bluenote-1577/sce-aligner/ along with scripts for
692 regenerating figures. All scripts are also available as Supplemenal Code.

693 **Competing interest statement**

694 The authors declare no competing interests.

695 **Acknowledgements**

696 J.S. was supported by an NSERC CGS-D scholarship. This work was supported by Natural
697 Sciences and Engineering Research Council of Canada (NSERC) grant RGPIN-2022-03074.

698 **Author contributions**

699 Both authors contributed to the development of the methods and proofs. J.S implemented the
700 software and performed the experiments.

701 **References**

702 Abouelhoda MI, Ohlebusch E. 2005. Chaining algorithms for multiple genome comparison.
703 *Journal of Discrete Algorithms* **3**: 321–341.

704 Alser M, Rotman J, Deshpande D, Taraszka K, Shi H, Baykal PI, Yang HT, Xue V, Knyazev S,
705 Singer BD, et al. 2021. Technology dictates algorithms: recent developments in read
706 alignment. *Genome Biol* **22**: 249.

707 Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. 1990. Basic local alignment search tool. *J*
708 *Mol Biol* **215**: 403–410.

709 Backurs A, Indyk P. 2018. Edit Distance Cannot Be Computed in Strongly Subquadratic Time
710 (Unless SETH is False). *SIAM Journal on Computing* **47**: 1087–1097.

711 Batzoglou S. 2005. The many faces of sequence alignment. *Brief Bioinform* **6**: 6–22.

712 Berger B, Daniels NM, Yu YW. 2016. Computational Biology in the 21st Century: Scaling with
713 Compressive Algorithms. *Commun ACM* **59**: 72–80.

714 Berger B, Waterman MS, Yu YW. 2021. Levenshtein Distance, Sequence Comparison and
715 Biological Database Search. *IEEE Trans Inf Theory* **67**: 3287–3294.

716 Blanca A, Harris RS, Koslicki D, Medvedev P. 2022. The Statistics of k-mers from a Sequence
717 Undergoing a Simple Mutation Process Without Spurious Matches. *Journal of*
718 *Computational Biology* **29**: 155–168.

719 Britten RJ. 2002. Divergence between samples of chimpanzee and human DNA sequences is
720 5%, counting indels. *Proceedings of the National Academy of Sciences* **99**: 13633–13635.

721 Chaisson MJ, Tesler G. 2012. Mapping single molecule sequencing reads using basic local
722 alignment with successive refinement (BLASR): application and theory. *BMC*
723 *Bioinformatics* **13**: 238.

724 Chao K-M, Pearson WR, Miller W. 1992. Aligning two sequences within a specified diagonal
725 band. *Bioinformatics* **8**: 481–487.

726 Chvátal V, Sankoff D. 1975. Longest Common Subsequences of Two Random Sequences. *J*
727 *Appl Probab* **12**: 306–315.

728 Colquhoun RM, Hall MB, Lima L, Roberts LW, Malone KM, Hunt M, Letcher B, Hawkey J,
729 George S, Pankhurst L, et al. 2021. Pandora: nucleotide-resolution bacterial pan-genomics
730 with reference graphs. *Genome Biol* **22**: 267.

731 Delahaye C, Nicolas J. 2021. Sequencing DNA with nanopores: Troubles and biases. *PLoS*
732 *One* **16**: e0257521.

733 Durbin R, Eddy SR, Krogh A, Mitchison G. 1998. *Biological Sequence Analysis: Probabilistic*
734 *Models of Proteins and Nucleic Acids*. Cambridge University Press.

735 Dutta A, Pellow D, Shamir R. 2022. Parameterized syncmer schemes improve long-read
736 mapping. *PLoS Comput Biol* **18**: e1010638.

737 Edgar R. 2021. Syncmers are more sensitive than minimizers for selecting conserved k-mers in
738 biological sequences. *PeerJ* **9**: e10805.

739 Farrar M. 2007. Striped Smith–Waterman speeds database searches six times over other SIMD
740 implementations. *Bioinformatics* **23**: 156–161.

741 Fofanov Y, Luo Y, Katili C, Wang J, Belosludtsev Y, Powdrill T, Belapurkar C, Fofanov V, Li T-B,
742 Chumakov S, et al. 2004. How independent are the appearances of n-mers in different
743 genomes? *Bioinformatics* **20**: 2421–2428.

744 Frith MC, Noé L, Kucherov G. 2020. Minimally-overlapping words for sequence similarity
745 search. *Bioinformatics*.

746 Frith MC, Shaw J, Spouge JL. 2023. How to optimally sample a sequence for rapid analysis.
747 *Bioinformatics* **39**.

748 Ganesh A, Sy A. 2020. Near-Linear Time Edit Distance for Indel Channels. *arXiv:200703040*.

749 Hera MR, Pierce-Ward NT, Koslicki D. 2022. Debiasing FracMinHash and deriving confidence
750 intervals for mutation rates across a wide range of evolutionary distances. *bioRxiv*.

751 Irber LC, Brooks PT, Reiter TE, Pierce-Ward NT, Hera MR, Koslicki D, Brown CT. 2022.
752 Lightweight compositional analysis of metagenomes with FracMinHash and minimum
753 metagenome covers. *bioRxiv*.

754 Ivanov P, Bichsel B, Vechev M. 2022. Fast and optimal sequence-to-graph alignment guided by
755 seeds. In *International Conference on Research in Computational Molecular Biology*, pp.
756 306–325.

757 Jain C, Gibney D, Thankachan S v. 2022. Co-linear chaining with overlaps and gap costs. In
758 *International Conference on Research in Computational Molecular Biology*, pp. 246–262.

759 Jain C, Rhie A, Zhang H, Chu C, Walenz BP, Koren S, Phillippy AM. 2020. Weighted minimizer
760 sampling improves long read mapping. *Bioinformatics* **36**: i111–i118.

761 Janson S. 2004. Large deviations for sums of partly dependent random variables: Large
762 Deviations for Dependent Random Variables. *Random Struct Algorithms* **24**: 234–248.

763 Kalikar S, Jain C, Vasimuddin M, Misra S. 2022. Accelerating minimap2 for long-read
764 sequencing applications on modern CPUs. *Nat Comput Sci* **2**: 78–83.

765 Keich U, Li M, Ma B, Tromp J. 2004. On spaced seeds for similarity search. *Discrete Appl Math*
766 *(1979)* **138**: 253–263.

767 Kiełbasa SM, Wan R, Sato K, Horton P, Frith MC. 2011. Adaptive seeds tame genomic
768 sequence comparison. *Genome Res* **21**: 487–493.

769 Kiwi M, Loebl M, Matoušek J. 2005. Expected length of the longest common subsequence for
770 large alphabets. *Adv Math (N Y)* **197**: 480–498.

771 Koerkamp RG, Ivanov P. 2022. Exact global alignment using A* with seed heuristic and match
772 pruning. *bioRxiv*.

773 Koning APJ de, Gu W, Castoe TA, Batzer MA, Pollock DD. 2011. Repetitive Elements May
774 Comprise Over Two-Thirds of the Human Genome. *PLoS Genet* **7**: e1002384.

775 Koonin E v, Aravind L, Kondrashov AS. 2000. The Impact of Comparative Genomics on Our
776 Understanding of Evolution. *Cell* **101**: 573–576.

777 Köster J. 2016. Rust-Bio: a fast and safe bioinformatics library. *Bioinformatics* **32**: 444–446.

778 Langmead B, Salzberg SL. 2012. Fast gapped-read alignment with Bowtie 2. *Nat Methods* **9**:
779 357–359.

780 Lember J, Matzinger H. 2009. Standard deviation of the longest common subsequence. *The*
781 *Annals of Probability* **37**.

782 Li H. 2018. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics* **34**: 3094–
783 3100.

784 Li H. 2021. New strategies to improve minimap2 alignment accuracy. *Bioinformatics* **37**: 4572–
785 4574.

786 Li H, Durbin R. 2009. Fast and accurate short read alignment with Burrows–Wheeler transform.
787 *Bioinformatics* **25**: 1754–1760.

788 Li H, Feng X, Chu C. 2020. The design and construction of reference pangenome graphs with
789 minigraph. *Genome Biol* **21**: 265.

790 Lipman DJ, Pearson WR. 1985. Rapid and Sensitive Protein Similarity Searches. *Science*
791 *(1979)* **227**: 1435–1441.

792 Lunter G, Drummond AJ, Miklós I, Hein J. 2005. Statistical Alignment: Recent Progress, New
793 Applications, and Challenges. In *Statistical Methods in Molecular Evolution*, pp. 375–405,
794 Springer-Verlag, New York.

795 Lunter G, Rocco A, Mimouni N, Heger A, Caldeira A, Hein J. 2008. Uncertainty in homology
796 inferences: Assessing and improving genomic sequence alignment. *Genome Res* **18**: 298–
797 309.

798 Mäkinen V, Belazzougui D, Cunial F, Tomescu AI. 2015. *Genome-Scale Algorithm Design:*
799 *Biological Sequence Analysis in the Era of High-Throughput Sequencing*. Cambridge
800 University Press, Cambridge.

Mäkinen V, Sahlin K. 2020. Chaining with overlaps revisited. *ArXiv*.

Marçais G, Delcher AL, Phillippy AM, Coston R, Salzberg SL, Zimin A. 2018. MUMmer4: A fast and versatile genome alignment system. *PLoS Comput Biol* **14**: e1005944.

Marçais G, Solomon B, Patro R, Kingsford C. 2019. Sketching and Sublinear Data Structures in Genomics. *Annu Rev Biomed Data Sci* **2**: 93–118.

Marco-Sola S, Moure JC, Moreto M, Espinosa A. 2021. Fast gap-affine pairwise alignment using the wavefront algorithm. *Bioinformatics* **37**: 456–463.

Medvedev P. 2022a. The limitations of the theoretical analysis of applied algorithms. *ArXiv*.

Medvedev P. 2022b. Theoretical analysis of edit distance algorithms: an applied perspective. *ArXiv*.

Myers EW. 1986. AnO(ND) difference algorithm and its variations. *Algorithmica* **1**: 251–266.

Myers G, Miller W. 1995. Chaining multiple-alignment fragments in sub-quadratic time. In *Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, *SODA '95*, pp. 38–47, Society for Industrial and Applied Mathematics, USA.

Navarro G. 2001. A guided tour to approximate string matching. *ACM Comput Surv* **33**: 31–88.

Needleman SB, Wunsch CD. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol Biol* **48**: 443–453.

Nurk S, Koren S, Rhie A, Rautiainen M, Bzikadze A v, Mikheenko A, Vollger MR, Altemose N, Uralsky L, Gershman A, et al. 2022. The complete sequence of a human genome. *Science (1979)* **376**: 44–53.

821 Ondov BD, Treangen TJ, Melsted P, Mallonee AB, Bergman NH, Koren S, Phillippy AM. 2016.
822 Mash: fast genome and metagenome distance estimation using MinHash. *Genome Biol* **17**:
823 132.

824 Otto C, Hoffmann S, Gorodkin J, Stadler PF. 2011. Fast local fragment chaining using sum-of-
825 pair gap costs. *Algorithms for Molecular Biology* **6**: 4.

826 Rautiainen M, Marschall T. 2020. GraphAligner: rapid and versatile sequence-to-graph
827 alignment. *Genome Biol* **21**: 1–28.

828 Reinert G, Schbath S, Waterman MS. 2000. Probabilistic and Statistical Properties of Words: An
829 Overview. *Journal of Computational Biology* **7**: 1–46.

830 Ren J, Chaisson MJP. 2021. lra: A long read aligner for sequences and contigs. *PLoS Comput*
831 *Biol* **17**: e1009078.

832 Roberts M, Hayes W, Hunt BR, Mount SM, Yorke JA. 2004. Reducing storage requirements for
833 biological sequence comparison. *Bioinformatics* **20**: 3363–3369.

834 Sahlin K. 2022. Strobealign: flexible seed size enables ultra-fast and accurate read alignment.
835 *Genome Biol* **23**: 1–27.

836 Sahlin K, Baudeau T, Cazaux B, Marchet C. 2022. A survey of mapping algorithms in the long-
837 reads era. *bioRxiv*.

838 Sarmashghi S, Bohmann K, P. Gilbert MT, Bafna V, Mirarab S. 2019. Skmer: assembly-free and
839 alignment-free sample identification using genome skims. *Genome Biol* **20**: 34.

840 Shaw J, Yu YW. 2022. Theory of local k-mer selection with applications to long-read alignment.
841 *Bioinformatics* **38**: 4659–4669. https://doi.org/10.1093/bioinformatics/btab790.

842 Siegel AF. 1982. Robust regression using repeated medians. *Biometrika* **69**: 242–244.

Sirén J, Monlong J, Chang X, Novak AM, Eizenga JM, Markello C, Sibbesen JA, Hickey G, Chang P-C, Carroll A, et al. 2021. Pangenomics enables genotyping of known structural variants in 5202 diverse genomes. *Science (1979)*.

Smith TF, Waterman MS. 1981. Identification of common molecular subsequences. *J Mol Biol* **147**: 195–197.

Sović I, Šikić M, Wilm A, Fenlon SN, Chen S, Nagarajan N. 2016. Fast and sensitive mapping of nanopore sequencing reads with GraphMap. *Nat Commun* **7**: 11307.

Suzuki H, Kasahara M. 2018. Introducing difference recurrence relations for faster semi-global alignment of long sequences. *BMC Bioinformatics* **19**: 45.

Szpankowski W. 2001. *Average Case Analysis of Algorithms on Sequences: Szpankowski/Average*. John Wiley & Sons, Inc., Hoboken, NJ, USA.

Thorne JL, Kishino H, Felsenstein J. 1991. An evolutionary model for maximum likelihood alignment of DNA sequences. *J Mol Evol* **33**: 114–124.

Ukkonen E. 1983. On approximate string matching. In *Foundations of Computation Theory* (ed. M. Karpinski), *Lecture Notes in Computer Science*, pp. 487–495, Springer, Berlin, Heidelberg.

Vingron M, Waterman MS. 1994. Sequence alignment and penalty choice: Review of concepts, case studies and implications. *J Mol Biol* **235**: 1–12.

Zheng H, Kingsford C, Marçais G. 2020. Improved design and analysis of practical minimizers. *Bioinformatics* **36**: i119–i127.

Zheng H, Kingsford C, Marçais G. 2021. Sequence-specific minimizers via polar sets. *Bioinformatics* **37**: i187–i195.

865