Incorporating indel channels into average-case analysis of seed-chain-extend

Spencer Gibson
Department of Biomedical Engineering, Carnegie Mellon University, Pittsburgh, PA, USA

Yun William Yu[1]
Computational Biology Department, Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

Given a sequence s_1 of n letters drawn i.i.d. from an alphabet of size σ and a mutated substring s_2 of length $m < n$, we often want to recover the mutation history that generated s_2 from s_1. Modern sequence aligners are widely used for this task, and many employ the seed-chain-extend heuristic with k-mer seeds. Previously, Shaw and Yu showed that optimal linear-gap cost chaining can produce a chain with $1 - O\left(\sqrt{\frac{1}{m}}\right)$ recoverability, the proportion of the mutation history that is recovered, in $O\left(mn^{2.43\theta}\log n\right)$ expected time, where $\theta < 0.159$ is the mutation rate under a substitution-only channel and s_1 is assumed to be uniformly random. However, a gap remains between theory and practice, since real genomic data includes insertions and deletions (indels), and yet seed-chain-extend remains effective. In this paper, we generalize those prior results under indel channels by introducing mathematical machinery to prove that the expected recoverability of an optimal chain is $\geq 1 - O\big((\log n)^2\, n^{-C\alpha}\big)$, when the total mutation rate given by the sum of the substitution, insertion, and deletion mutation rates ($\theta_T = \theta_i + \theta_d + \theta_s$) is less than 0.159, i.e., $0 < \theta_T < 0.159$, and $\alpha = -\log_\sigma(1-\theta_T)$.

2012 ACM Subject Classification Applied computing → Computational biology; Applied computing → Bioinformatics

Keywords and phrases Sequence alignment, seed-chain-extend, indel channels

Digital Object Identifier 10.4230/LIPIcs.TBD.2025.23

Funding No funding or competing interests are declared.

1 Introduction

String alignment—i.e. determining the best way to match positions of two similar strings s_1 and s_2 under some cost function—has always been one of the central primitives in computational biology, essential for downstream biological analyses like comparing relatedness of genomes or mapping sequenced reads [5, 15, 21, 26]. It is closely related to the edit distance and longest common substring (LCS) problems [4, 8, 14, 23, 29], as the choice of matching positions in the alignment implies a series of insertions, deletions, and substitutions that would be needed to transform s_2 into s_1 or vice versa. To this end, Needleman-Wunsch [20] and Smith-Waterman [27] gave dynamic programming exact solutions to the global and local alignment problems in quadratic $O(mn)$ time, where $|s_1| = n$ and $|s_2| = m$. Although more efficient algorithms exist, e.g., the Four Russians Method has a time complexity of $O(mn/\log(n))$ [1], it turns out that in the worst case, we cannot do polynomially better—Backurs and Indyk showed in 2015 that "edit distance cannot be computed in strongly subquadratic time (unless SETH is false)" [3].

Of course, the landscape of alignment algorithms and heuristics is more complicated. Although in the earliest days of bioinformatics, we had the luxury of algorithms with

[1] Corresponding author

mathematically provable guarantees on accuracy and speed, the rapid growth of biological data necessitated the development of faster heuristics that do not come with those strong guarantees. BLAST [2], one of the most highly cited papers of all time [31], gives a linear-time heuristic for local alignment, at the cost of optimality, and is still to this day one of the primary workhorses of bioinformatics, whereas Smith-Waterman has been relegated to being just a subroutine within heuristic software. More broadly, there are many exact sequence alignment/edit distance algorithms that have subquadratic time complexity under certain conditions—Ukkonen's method [30] for edit distance runs in $O(s * \min(m, n))$ in the worse caste, where s is the edit distance between the two strings, and Myer's algorithm [19] has $O(m + n + d^2)$ average-case time complexity where d is the minimum edit script between the two strings. Other than exact algorithms, there are numerous heuristics that optimize sequence alignment for specific tradeoffs [7, 11, 12, 16].

One heuristic of particular interest is "seed-chain-extend," which is used in modern software such as Minimap 2 [18]. The seed-chain-extend heuristic has three stages. In *seeding*, k-mer 'seeds' are selected on both s_1 and s_2, and shared k-mers are marked as 'anchors' between the two strings. Afterwards, concordant anchors are *chained* together to form the skeleton of an alignment. Lastly, the space between anchors is filled in using standard worst-case quadratic-time dynamic programming in a process known as *extension*. Seed-chain-extend empirically showcases near quasilinear runtime on the similar genomic strings it is typically applied to, but is not guaranteed to find optimal alignments.

For a long time, bioinformaticians have contented ourselves to this gap between theory and practice. In the last five years, though, theoreticians have made several new breakthroughs by defining a generative model of string evolution and revisiting average-case analysis. Analysis of string algorithms [6, 17], particularly average-case analysis [28], historically made extensive use of generating functions; however, more recent approaches instead used tail-bounds to bound bad events, more akin to the analysis of some randomized probabilistic sketches [33]. Ganesh and Sy's 2020 breakthrough was to show that under their random mutation model, a modified dynamic programming algorithm will compute edit distance in $O(n \log n)$ time between a random string s_1 and a mutated string s_2 of near equal length [10] with high probability. However, part of what made their analysis work was the concordance of their DP algorithm with their mutation model, which meant that extending their results to more practical but sophisticated heuristics like seed-chain-extend was nontrivial.

To that end, last year, our research group made substantial progress on proving similar results for seed-chain-extend [25]. Unfortunately, the machinery and techniques we introduced in that paper were insufficiently powerful to address insertions and deletions, so we had to restrict our theoretical results to a substitution-only mutation model. Indels tend to complicate analyses and dependence structures, so most bioinformatics theoreticians either avoid directly working with them [9, 22, 24, 32], or adjust their algorithm and model to directly capture them [10]. We were able to run empirical benchmarks with indels that closely tracked our substitution-only theory (including accurate predictions of exponents), so we believed without proof the theorems were also correct for indels [25]. In this sequel, we make progress toward narrowing that remaining gap between theory and practice, generalizing our prior machinery and techniques to handle indels.

2 Strategy Overview

2.1 Challenges to analysis of seed-chain-extend

There are several difficulties in proving average-case results for seed-chain-extend. First, seed-chain-extend has three stages, and chaining and extension have different optimization objectives. When considering the overall performance of the heuristic, it is in theory possible for failures to happen at any stage. By failure here, we mean anything that leads to bad downstream events, which include most prominently not finding the correct string alignment or taking too long (e.g. quadratic time) to find that alignment. A bad chain can result from a failure in chaining, or just a bad selection of anchors in seeding. Similarly, extension failure can be a result of bad chaining, or because the extension procedure does not itself find the right alignment, despite the chaining being "good". Here we should also note that the alignment problem under a mutation model actually diverges somewhat from the edit distance problem. The best scoring alignment corresponds to some edit distance between the strings, but arguably the "correct" alignment (at least from a biological perspective) is the one that reflects the mutations that happened to transform s_1 into s_2. Furthermore, seed-chain-extend is known to be an approximate heuristic, and it does not guarantee a correct alignment (under either definition of correct).

To resolve these difficulties, in the prequel [25], we introduced the concept of "recoverability", which decoupled chaining accuracy from extension accuracy. Extension is only performed in the gaps between anchors on the chain, so roughly speaking, recoverability measures how much of the correct alignment can possibly be recovered given a chain. By structuring our problem thus, we can focus on just the seeding and chaining—i.e. how good is the chain as a starting point for the extension phase. Importantly, recoverability of a chain is a theoretical measure of the goodness of the chain, as opposed to the optimization criterion used to generate the chain, such as linear-gap cost chaining.

2.2 Difficulty from indels

Unfortunately, our formal definition of recoverability in the prequel relied on the substitution-only error model. In the substitution-only regime, the correct alignment of a mutated substring s_2 to s_1 is always a diagonal line in the alignment matrix (which we termed the "homologous diagonal"). Thus, recoverability could be defined as the proportion of the homologous diagonal covered by anchors on the optimal chain and the dynamic programming (DP) extension blocks between anchors. In the presence of indels though, the correct alignment is no longer a straight diagonal. In this sequel, we thus must generalize the homologous diagonal to a "homologous path".

In the same vein, indels also mess up the matching of indices between s_1 and s_2. This is not only notationally very tricky to reconcile, but also make it hard to discuss the dependence structure of positions in anchors, which is necessary for the limited-dependence Chernoff bounds we used in the prequel.

Finally, having several different types of mutations raises questions of ordering and reversal that do not appear with only substitutions. One such complication is 'no operations' ('no-ops') – mutations that at positions in the original string without leaving behind a different sequence of characters – such as an deletion of a letter and an immediate reinsertion of it.

This however creates a problem with defining recoverability of a homologous path. If a letter is inserted and then deleted, or vice versa, then the alignment specified by the homologous path will have a spurious kink that cannot be found via k-mer matching. For example, if s_1 = ACGT, and it is mutated to s_2 = ACGT by inserting another G before the original G, and then deleting the original G, then the "correct" alignment is

```
AC-GT
||  |
ACG-T
```

which naturally will not be found by any reasonable alignment method. This is not a problem for recoverability in extension regions, as it still could be an alignment produced by the extension block (given some extension algorithm, however uncommon), but it cannot be found as an anchor, which would produce instead the "incorrect" (but lowest edit distance) alignment

```
ACGT
||||
ACGT
```

Still, in the interest of theoretical consistency, we will apply a recoverability penalty in the latter case, despite it not actually being a problem in practice, because the lower-edit-distance alignment that seed-chain-extend will find does not reflect the actual no-op mutation history.

2.3 Proof structure and motivation

The basic intuition behind the prequel [25] is that given reasonably low mutation rates, the optimal chain under linear-gap-cost chaining will be close to correct in the sense that most of the anchors will lie on the homologous diagonal. A gap between anchors can either be homologous if the anchors flanking it are both on the homologous diagonal, or non-homologous if at least one of the two anchors is off the homologous diagonal. Non-homologous gaps can lead to "breaks", which are regions of the string where extension through the gaps does not cover the homologous diagonal, leading to a decrease in recoverability. However, with high probability, each break has size $< \sqrt{m}$ and the number of breaks is small, so the recoverability will be high. Additionally, the runtime can be bounded by extension time through all the homologous and non-homologous gaps, which are small in an optimal chain.

Roughly speaking, the intuitive reason the above strategy works is that substitutions are much more likely to break anchors than they are to create spurious anchors. So long as we remain in the regime where sufficiently many anchors can still be found, the optimal chain is close to the correct chain, and the recoverability will be high. When indel channels are added to the mix, the above logic still basically holds (though we do suddenly have pseudo-spurious anchors from indel reversals we have to deal with), but we have to carefully redefine the model and generalize recoverability. Furthermore, because of the redefinition of the model, many of the theorems and lemmas from [25] need to be updated and proved from scratch. We detail the full proof for each theorem/lemma which requires new techniques and omit the proof only when it is exactly analogous to the corresponding proof in the prequel.

The major difference we handle is a new class of anchors, defined below, termed "clipping" anchors (Fig. 2), which lie partially on the homologous path. Clipping anchors are a strange phenomenon of indels. They contribute to recoverability in a similar way as homologous anchors: extending through gaps flanked by clipping anchors still contains the bulk majority of the path in the gap. However, they behave like spurious anchors in that the anchors themselves may not recover points on the homologous path. Clipping anchors may seem to be an anomaly but, in fact, they are the most likely anchor type with indels. Intuitively, there are many ways in which indels can occur without breaking an anchor and the probability of these events is relatively close to the probability of a homologous anchor occurring.

By generalizing all the theorems in the prequel and bounding the number of missed points in regions of clipping anchors, we conclude that the expected recoverability of an optimal chain is $\geq 1 - O\big((\log n)^2\, n^{-C\alpha}\big)$. The reader will find proofs for mathematically involved lemmas and theorems in the appendix.

2.4 Preliminaries

There are three mutation types used in our analysis, inspired by the mutation models of Shaw and Yu [25], and Ganesh and Sy [10]: substitutions, deletions, and insertions. Each mutation occurs independently at each position.

Before giving a formal definition of the mutation model, we define and bound the constant ρ_i', the failure probability of the geometric distribution from which insertion lengths are sampled. Intuitively, a larger ρ_i' leads to shorter insertions.

▶ **Definition 1.** *Conditional on an insertion occurring at a position, the insertion length is sampled from the distribution given by* $\mathrm{Geom}(1-\rho_i')$. *In this analysis,* ρ_i' *must be bounded away from* 1, *i.e.,* $0 < \rho_i' < \gamma < 1$, *for any arbitrary* $\gamma \in (0,1)$.

▶ **Definition 2** (Mutation Model). *Let* $S = x_1 x_2 \cdots x_{n+k-1}$ *be a string where each letter* x_i *is sampled i.i.d. from an alphabet of size* σ. *The mutated substring S' is obtained by passing the substring* $S[p+1 : p+m']$ *through a mutation channel where for each position* $p+j$, $1 \leq j \leq m'$, *the input symbol* $S[p+j]$ *undergoes the following mutations independently:*

- ***Substitution*** *(probability* θ_s*):* $S[p+j]$ *is replaced by a different letter, chosen uniformly from the other* $\sigma - 1$ *symbols.*
- ***Deletion*** *(probability* θ_d*):* $S[p+j]$ *is deleted.*
- ***Insertion*** *(probability* θ_i*): a random string of length* $L \sim \mathrm{Geom}(1-\rho_i')$ *is inserted at position* $p+j$.
- ***Match*** *(probability* $1-\theta_s-\theta_d$*): S[p+j] is not deleted nor replaced by a different symbol.*

Note that if an insertion occurs then it has length at least 1. In the substitution-only case [25], the optimal alignment is the diagonal matching every position $p+j$ in S to j in S', i.e., the set of points $\{(p+j, j) \mid 1 \leq j \leq m'\}$. When indels occur, the optimal alignment is nonobvious. We choose to define the optimal alignment as the **path history** of mutations from which S' is generated from S. We call this path history the *homologous path*, which generalizes the homologous diagonal from the prequel [25]. The homologous path, formalized below, is adapted from the canonical alignment from Ganesh and Sy [10].

▶ **Definition 3** (Inspired by Ganesh and Sy). *Let* S *be the original length* n *substring with i.i.d. letters sampled from the alphabet of size* σ, *and S' the string resulting from passing* $S[p+1, p+m']$ *through the mutation channel where edits* $\mathcal{E}$ *are applied.*

The homologous path P_H *between* S *and* S' *is represented as a list that initially contains* $(p, 0)$. *Let* (i,j) *be the last element of* P_H. *We iteratively append points to* P_H *based on the mutations that occur at position* $i+1$ *in* S, *assuming* $p+1 \leq i+1 \leq p+m'$, *following* $\mathcal{E}$*:*

- *If no insertion or deletion occurs at position* $i+1$, *i.e., a substitution occurs or there is no mutation, append the point* $(i+1, j+1)$ *to* P_H.
- *If an insertion of* I *letters occurs at position* $i+1$ *and no deletion occurs, append the points* $(i, j+1), \ldots, (i, j+I), (i+1, j+I+1)$ *to* P_H.
- *If a deletion and no insertion occurs at position* $i+1$, *append the point* $(i+1, j)$ *to* P_H.

- *If both an insertion of I letters and a deletion occur at position $i+1$, append the points $(i, j+1)$, ..., $(i, j+I)$, $(i+1, j+I)$ to P_H.*

Figure 1 provides an example of S = TACTTCGC mutating into S' = TACTTTAC, following the mutation model defined above. Figure 2 shows the homologous path given the edit history.

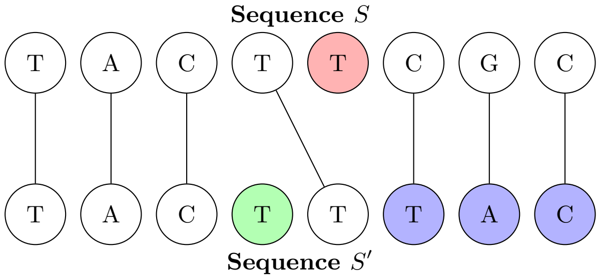

Figure 1 The match graph resulting from the mutation process that gives S' from S = TACTTCGC, including deletion (red), insertion (green), substitutions (blue), and matches (clear in S'). Lines between nodes represent corresponding positions between the sequences. Specifically, from left to right, an insertion of the letter T occurs at position 4, position 5 is deleted and the characters at positions 6, 7, and 8 are mutated.

▶ **Definition 4.** *The total mutation rate is denoted by $\theta_T = \theta_s + \theta_d + \theta_i$. We will assume that $\theta_T < 0.159$.*

We end this section by defining constants and bounds on constants that will be used throughout the paper.

▶ **Definition 5.** *The constant $\alpha = -\log_\sigma(1-\theta_T)$ represents the expected per-base contribution to the matching length between S and S'. The length of the seeds, k, is given by $k = C\log_\sigma(n)$ for any $C > \frac{3}{1-2\alpha}$. We will write $\log = \log_\sigma$ for short. We can write $C\alpha = \frac{3\alpha}{1-2\alpha} + \delta$, and noting that $\alpha < \frac{1}{8}$, we can always choose $\delta > 0$ small enough such that $C\alpha < \frac{1}{2}$.*

Let $c_0 = \max(\frac{1}{2}\ln(\frac{9}{1+8\gamma}), \frac{21}{\beta}) \leq 30$, where $\beta = log_\sigma(e)$. The gap penalty constant in the seed chain extend linear-gap cost will be $\xi = \frac{1}{6(c_0+1)g(n)}$ where $g(n) = \frac{50k}{8(1-\theta_T)^k}\ln(n)$.

The length of the generative region in S is $m' = \Omega(n^{2C\alpha+\epsilon})$, for an arbitrarily small enough $\epsilon > 0$. Note such a choice is possible since $C\alpha < 1/2$ and $n^{2C\alpha+\epsilon} < n$.

By our choice of variables, $\sigma^k = n^C$ and $(1-\theta_T)^k = n^{-C\alpha}$, which we will use repeatedly.

2.5 Tools

In this section, we go through definitions of tools and terms that simplify our analysis. A central concept of seed-chain-extend is an anchor, which we define formally below. An anchor is an exact k-mer match between S and S'.

▶ **Definition 6.** *We say that an anchor of length k occurs at (i, j), i.e., starting at position i in S and position j in S', if $S[i : i+k-1] = S'[j : j+k-1]$. Note that indexing is right-inclusive. All anchors mentioned throughout the paper are of length k.*

We will also find it useful to define the random variable indicating if letters at a position in S and S' match, as well as the random variable indicating if there is an anchor starting at specified positions in S and S'.

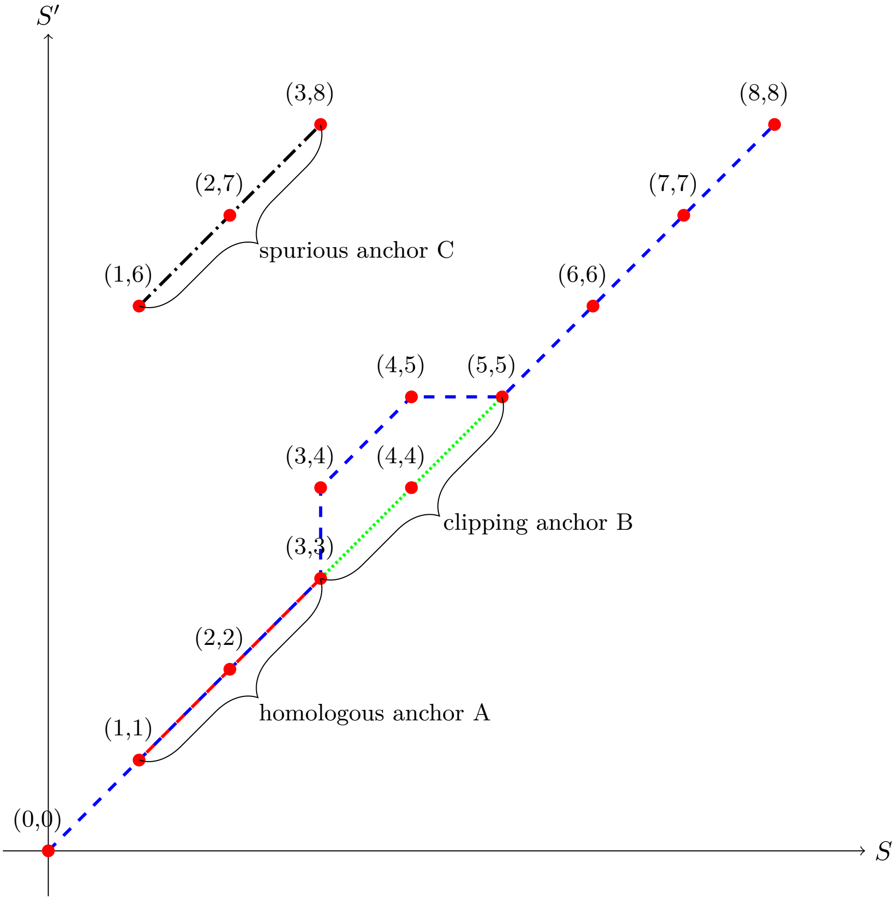

Figure 2 The points along the dashed blue line make up the homologous path given the edits turning S = TACTTCGC into S' = TACTTTAC following figure 1. In this example, anchors are matching seeds of length 3. Anchor A (red dash) is a *homologous* anchor since it lies entirely on the path. Anchor B (green dash) is a *clipping* anchor since it lies partially on the path, namely, the midpoint of the anchor does not belong to the homologous path. Anchor C (black dash) is spurious since it lies entirely off the path.

▶ **Definition 7.** *Let $M(i,j) = \mathbf{1}\{S[i] = S'[j]\}$ be the indicator random variable detecting if S and S' share the same character at positions i and j. Define $A(i,j) = \prod_{\ell=0}^{k-1} M(i+\ell,\ j+\ell)$, the indicator random variable for an anchor occurring at (i,j).*

There are three types of anchors relevant to our analysis: homologous, clipping, and spurious anchors (see figure 2 for examples of each). Visually, anchors always appear as diagonals of length k in the alignment matrix. A homologous anchor lies entirely on the homologous path (anchor A in figure 2), which implies the homologous path contains no mutations at all throughout that anchor. Unlike the substitution-only case, under indel channels, anchors can touch the homologous path at a number of points without lying entirely on it– these are called clipping anchors (anchor B in figure 2). Spurious anchors remain the same as before: anchors that lie entirely off the homologous path (anchor C in figure 2). We now formally define each anchor type.

▶ **Definition 8.** *For notational ease, let $A = \{(i+t, j+t) \mid 0 \le t \le k-1\}$ and $B = \{(x,y) \in P_H \mid i \le x \le i+k-1 \ \wedge\ j \le y \le j+k-1\}$. If there exists an anchor at (i,j), then A*

is the set of points belonging to that anchor, and B is the set of points on the homologous path between (inclusive) those two points. The anchor at (i, j) is ***homologous*** *if $A = B$,* ***spurious*** *if $A \cap B = \emptyset$, and* ***clipping*** *otherwise.*

▶ Remark 9. Clipping anchors arise when insertions and deletions occur under the mutation model. By definition, they miss points on the homologous path inside of their extension boxes. Our definition of recoverability accounts for these points by incorporating the extension boxes around anchors. This allows us to treat clipping and homologous anchors equally.

We now define the notion of a *chain*.

▶ **Definition 10.** *A* ***chain*** *C is a list of anchors $((i_1, j_1), \ldots, (i_u, j_u))$ such that $i_k \leq i_{k+1}$ and $j_k \leq j_{k+1}$ for all $1 \leq k \leq u - 1$.*

Let $x \in (p+1, p+m')$ be an index of the generative region of S. If the character at position x is not deleted in the mutation process, then there is a unique corresponding position in S'. The character at every other position of S' is independent of S$[x]$. It is convenient to define a function f that maps each position i ∈S to its unique corresponding position in S' if one exists. The mapping returns the minimum y such that $(x, y) \in P_H$ and null if no such point exists. To see why this works, consider the following cases: if the character at position x in S is deleted, then there is no corresponding position, so f should return null; otherwise, let $y_0 = \min\{\, y : (x, y) \in P_H \,\}$. If there is a substitution at position x or no mutation, then the corresponding position to x is exactly y_0. If there is an insertion and no deletion at position x, then y_0 will still be the corresponding position in S' to position x in S since insertions occur to the left of mutated positions.

▶ **Definition 11.** *Define the function $f : \{1, \ldots, |S|\} \longrightarrow \{1, \ldots, |S'|\} \cup \{\text{null}\}$ such that*

$$f(x) = \begin{cases} \emptyset, & \textit{if } x \notin (p+1, p+m') \textit{ or } x \textit{ is deleted,} \\ \min\{\, y : (x, y) \in P_H \,\}, & \textit{otherwise.} \end{cases}$$

If in the mutation process, position $x \in (p + 1, p+, m')$ of S is not deleted, then either no mutation occurred at x or another character is substituted for $S[x]$. In either case, there exists a position y in S' such that $(x, y) \in P_H$, meaning that $\min\{\, y : (x, y) \in P_H \,\}$ is well-defined.

3 Methods and Bounds

3.1 Recoverability setup

The recoverability of a chain, $C = ((i_1, j_1), \ldots, (i_u, j_u))$, loosely speaking, is the fraction of the homologous path P_H, that lies in the extension of anchors in the chain and in gap extensions. We define recoverability to account for all *possible* alignments given by the chain, which means that we count any portion of the homologous path in an extension as 'recovered' since, in theory, it could be recovered by some extension algorithm. We formalize this intuition below.

▶ **Definition 12** ((Yu and Shaw) Recoverability)**.** *Given a chain $C = ((i_1, j_1), \ldots, (i_u, j_u))$, we define the union of all possible alignments for the chain C, $Align(C)$, as:*

$$Align(C) = \bigcup_{\ell=1}^{u} \{(i_\ell, j_\ell), \ldots, (i_\ell + k - 1, j_\ell + k - 1)\} \;\cup\; \bigcup_{\ell=1}^{u-1} Ext(\ell).$$

Where $Ext(\ell) = \{i_\ell + k - 1, \dots, i_{\ell+1}\} \times \{j_\ell + k - 1, \dots, j_{\ell+1}\}$*. If* $i_\ell + k > i_{\ell+1} - 1$ *or* $j_\ell + k > j_{\ell+1} - 1$*, then* $Ext(\ell) = \emptyset$*.*

The recoverability of the chain, $R(C)$*, is defined to be:*

$$R(C) = \frac{|Align(C) \cap P_H|}{|P_H|}$$

3.2 Independence lemmas and the match graph

The match graph, defined below, is a bipartite graph that captures the dependence structure between positions on S and S'. The match graph contains edges between letters at positions i in S and $f(i)$ in S', given that $f(i) \neq \emptyset$, i.e., edges occur between letters at corresponding positions in S and S'. When examining the independence structure of a set of random matching variables $\{M(x_1, y_1), \dots, M(x_p, y_p)\}$ where the x_i are positions on S and the y_i are positions on S', we refer to the *induced* match graph of $\mathcal{M}$. Here, all edges in the original match graph remain and additional edges occur between the letters at position x_1 in S and y_1 in S', x_2 in S and y_2 in S', etc. (see Figure 3 for an example). Intuitively, when the induced match graph of $\mathcal{M}$ does not have any cycles, then the random variables $M(i_\ell, j_\ell) \in \mathcal{M}$ are independent. We will use this fact to calculate the probability that different anchor types occur, and when a pair of anchor indicator variables are independent, which is used extensively in Lemma 24.

▶ **Definition 13.** *Let* $\mathcal{M} = \{M(a_1, b_1), \dots, M(a_p, b_p)\}$ *be a set of matching variables where* a_i *are positions in S and* b_i *are positions in S'. The match graph induced by* $\mathcal{M}$ *refers to the graph* $G = (V, E)$ *where the vertices* V *are the letters in S and S', and the edges are given by* $E = \{(x_i, y_{f(x_i)}) \mid i \in (p+1, p+m') \wedge f(x_i) \neq \emptyset\} \cup \{(x_h, y_l) \mid M(h, l) \in \mathcal{M}\}$*. Note that* $G(\mathcal{M})$ *is bipartite.*

▶ **Lemma 14** ((Yu and Shaw) Conditional independence in the match graph)**.** *Consider a set of random variables of the form* $\mathcal{M} = \{\, M(i_1, j_1), \dots M(i_\ell, j_\ell)\,\}$*. If two vertices* u, v *lie in separate connected components of* $G(\mathcal{M})$*, then they are conditionally independent of* $\mathcal{M}$*.*

The theorem below outlines a sufficient condition for match variables $\mathcal{M}$ to be independent: no variable references corresponding positions: $(i_t, j_t) \notin P_H$, for each $t = 1, \dots, \ell$.

▶ **Theorem 15** ((Adapted from Yu and Shaw) Spurious match variables are independent in a cycle free match graph)**.** *The random variables*

$$\mathcal{M} = \{\, M(i_1, j_1),\ M(i_2, j_2),\ \dots \},$$

where $f(i_\ell) \neq j_\ell$ *for all* $M(i_\ell, j_\ell) \in \mathcal{M}$*, are independent if the induced match graph has no cycles.*

▶ Remark 16. The above theorem requires that the match variables $M(i_t, j_t)$ use spurious points, i.e., $(i_t, j_t) \notin P_H$, which is captured by the condition $f(i_t) \neq j_t$. This is equivalent to the condition used in the prequel [25], where it is required that $i_t \neq j_t$, implying $(i_t, j_t) \notin P_H$ under the substitution-only mutation model. Although it is more notationally complex, the new condition can simply replace the old one, and the proof is unchanged.

▶ **Corollary 17.** *The match graph induced by a single anchor,* $A(i, j)$*, has no cycles.*

Proof. First note that any position in S that is deleted and any position in S' that is inserted can have degree at most 1, coming from the anchor $A(i, j)$. Any such position cannot be involved in a cycle, so we can remove all such positions from the graph. Relabel the surviving positions in S and S' as their new indices. The interval $(i, i+k-1)$ becomes $X_{i'}$, possibly empty, and $(j, j+k-1)$ becomes $X_{j'}$, also possibly empty. If $X_{i'}$ is empty then there cannot be any cycle that uses positions on S since each node has degree at most 1, and hence there are no cycles since the graph is bipartite; we conclude the same if $X_{j'}$ is empty. Note that the unconditioned match graph at this point appears as two sets of vertices of equal size with edges between each corresponding pair.

We now assume $X_{i'}$ and $X_{j'}$ are nonempty and that there exists a cycle. Let $i' = \min H_{i'}$ and $j' = \min X_{j'}$. First, suppose that $i' < j'$. Let x_{i+l} be the first point on S belonging to a cycle with $l \geq 0$. Then x_{i+l} has the neighbor y_{i+l}, so let the first edge of the cycle be (x_{i+l}, y_{i+l}). Then y_{i+l} must have degree exactly 2 for there to exist a cycle and its remaining edge must be induced from the anchor. Since $i' < j'$, its neighbor lies to the left of x_{i+l}, it is some x_{i+a} where $0 \leq a < l$, contradicting the minimality of l. Similarly, if $i' > j'$, we can apply the same argument but for the largest l such that x_{i+l} is in a cycle. Thus, no cycle exists.

◀

From the above, we get the following corollary, which states that if $\{(i+l, j+l) \mid 0 \leq l \leq k-1\}$ are spurious positions (off the homologous path), then a spurious anchor occurs at (i,j) with probability $\frac{1}{\sigma^k}$.

▶ **Corollary 18.** *If $f(i+\ell) \neq j+\ell$ for $\ell = 0, \ldots, k-1$, i.e., (i,j) is a possible spurious anchor, then $\Pr\big(A(i,j) = 1\big) = \frac{1}{\sigma^k}$.*

Proof. Corollary 17 shows that the random variables $M(i,j), M(i+1,j+1), \ldots, M(i+k-1, j+k-1)$ are independent, since the induced match graph of $A(i,j)$ has no cycle. Since each position pair does not belong to the homologous path, $Pr(M(i+l, j+l) = 1) = \frac{1}{\sigma}$ for $l = 0, \ldots, k-1$. Thus, $Pr(A(i,j) = 1) = \prod_{\ell=0}^{k-1} \Pr(M(i+\ell, j+\ell) = 1) = \frac{1}{\sigma^k}$. ◀

▶ Remark 19 ((Yu and Shaw) Independence lemma under substitutions). In the substitution-only regime considered by Yu and Shaw [25], the independence lemma is as follows: for $A(i,j)$ and $A(h,\ell)$, if both of the following conditions hold:
1. $|i-h| \geq k$ or $|j-\ell| \geq k$, and
2. $|i-\ell| \geq k$ or $|j-h| \geq k$,

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

Intuitively, the first condition $|i-h| \geq k$ or $|j-l| \geq k$ ensures that the anchors do not overlap too much – the anchors' coverage can overlap on S or S' but not on both. The second condition prevents 'twisting': consider anchors $A(1,3)$ and $A(3,1)$ in the substitution-only mutation model as shown in figure 3. There exists a cycle going from $x_1 \to y_3 \to x_3 \to y_1 \to x_1$ where x_i represents node i on the top set of vertices and y_i represents node i on the bottom set of vertices. The biological interpretation is that under low mutation rates, x_1 and y_1 are likely to be equal, as are x_3 and y_3. This implies that the events $x_1 = y_3$ and $x_1 = y_3$ are not independent. In the generalized lemma below, the first condition is the remains the same while the second condition is changed to account for possible insertions that complicate indices. The intuition, however, is the same. Although the general lemma is more notationally complex, the match graph in the indel case is not any more complicated because inserted and deleted positions do not have neighbors.

▶ **Lemma 20** (General independence lemma). *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*

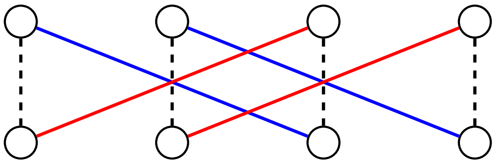

Figure 3 Induced match graph in the substitution-only regime of an initial string of length 4 with anchors $A(1,3)$ and $A(3,1)$. These anchors violate Yu and Shaw's [25] conditions for independence and, as can be seen, there exists a cycle in the graph.

2. $[i : i+k-1] \cap f^{-1}([l : l+k-1]) = \emptyset$ *or* $[h : h+k-1] \cap f^{-1}([j : j+k-1]) = \emptyset$,

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

See the appendix for the full proof.

3.3 Anchor-count concentration bounds

The following lemma shows that any k-mer in $S[p+1 : p+m']$ cannot expand by more than ck when generating S' with high probability. Our approach will be to show that the probability of an arbitrary k-block expanding too much is very low, and then we will take a union bound over all k-blocks in $S[p+1 : p+m']$ to upper bound the global failure rate by $1/n$. We will find it useful to write k in terms of $\ln(n)$ in moment-generating function and Chernoff style concentration inequalities. To that end, since $k = C\log_\sigma(n) = C\ln(n)\log_\sigma(e)$, let $\beta = \log_\sigma(e)$ so that $k = C\beta\ln(n)$. The constant β does not affect the analysis but allows us to be precise.

The two lemmas below establish inequalities that allow us to sum an infinite series and bound probabilities related to the expansion of a k-mer block in the generative region of S.

▶ **Lemma 21.** *For $t_0 = \frac{1}{2}\ln(\frac{9}{1+8\gamma})$, $e^{t_0}\rho'_i < 1$ and $(1-\theta_i)+\theta_i(\frac{(1-\rho'_i)e^t}{1-e^t\rho'_i}) \leq e$ for all $0 < \rho'_i < \gamma$.*

▶ **Lemma 22 (Bounded expansion (E)).** *With probability $\geq 1 - 1/n$, no k-mer in $S[p+1 : p+m']$ has more than $\frac{1}{t_0}(\frac{2}{\beta}+1)k$ inserted base pairs, for $t_0 = \frac{1}{2}\ln(\frac{9}{1+8\gamma})$.*

We now show a similar lemma for contraction below: namely, sufficiently large blocks in $S[p+1 : p+m']$ do not have too many deletions.

▶ **Lemma 23 (Bounded contraction (C)).** *With probability $\geq 1 - 1/n$, no ℓ-block in $S[p+1 : p+m']$ contracts to size $\leq \frac{(1-\theta_d)\ell}{2}$, where $\ell = \frac{21k}{\beta}$.*

We will refer to the space where the bounded contraction and expansion lemmas are jointly satisfied as **EC**.

▶ **Lemma 24 ((EC)).** $\mathbb{E}(N_S^2) \leq \mathbb{E}(N_S)^2 + 2T_0k^2\frac{mn}{\sigma^k}$ *where* $T_0 = \max(\frac{1}{t_0}(\frac{2}{\beta}+1)\frac{2}{1-\theta_d}\frac{21}{\beta}, 4)$. *Thus,* $\mathrm{var}(N_S) \leq T_0k^2\frac{mn}{\sigma^k}$

The previous lemma shows that allowing indels changes the variance of N_S, the number of spurious anchors, by at most a constant compared to the substitution-only case. The below lemma uses the conditional bounded variance of N_S along with the high likelihood of being in the EC space, to bound the number of spurious anchors w.h.p. The spurious anchor bound below is exactly the same as in the prequel up to a constant, which makes no asymptotic difference.

▶ **Lemma 25** (F1). *With probability at least $1 - \frac{3}{n}$, the number of spurious anchors is*

$$\leq n^{2-C} + \sqrt{T_0}\, C \log(n)\, n^{\frac{3-C}{2}}$$

Mathematically,

$$\Pr\Big(N_S \geq n^{2-C} + \sqrt{T_0}\, C \log(n)\, n^{\frac{3-C}{2}}\Big) \ \leq\ \frac{3}{n}$$

▶ Remark 26. Since $C > \frac{3}{1-2\alpha} > 3$, for large enough n there are no spurious anchors at all with probability $\geq 1 - \frac{3}{n}$ in the EC space.

We finish this section by upper bounding the expected number of clipping anchors, which allows us to upper bound the expected number of missed points due to clipping anchors in the chain. We start by providing the general lemma below, which bounds the number of clipping anchors in terms of the cardinality of the homologous path. In the following corollary, we specialize the below lemma to the EC case.

▶ **Lemma 27.** *The expected number of clipping anchors, N_C, is at most $|P_H|k(1-\theta_T)^k$.*

Proof. Consider any point $(i,j) \in P_H$. There are exactly k possible anchors containing that point – anchors that contain it at k different distances from the start of the anchor. Let $N_C(i,j)$ be the number of clipping anchors contributed for the point (i,j). Then $\mathbb{E}(N_C(i,j)) \leq \sum_{l=1}^{k} \Pr(A(i-l+1, j-l+1) = 1) \leq k(1-\theta_T)^k$. Thus, $\mathbb{E}(N_C) = \sum_{(i,j)\in P_H} \mathbb{E}(N_C(i,j)) \leq |P_H|k(1-\theta_T)^k$. ◀

▶ **Corollary 28** ((EC)). *The expected number of clipping anchors is at most $O(mk(1-\theta_T)^k)$.*

Proof. Under EC, each block of k positions in S contributes at most $(\frac{1}{t_0}(\frac{2}{\beta}+1)+1)k$ points to the homologous path P_H by Lemma 22. Then there can be at most $(\frac{1}{t_0}(\frac{2}{\beta}+1)+1)m = O(m)$ points in P_H. Applying Lemma 27 gives the result. ◀

In the next section, we show that there are no long homologous gaps in the generative region of S, and use this fact to bound break lengths with the help of the expansion-contraction lemma.

3.4 Bounding homologous gaps

A homologous gap is defined to be a region $S[p+a, p+b]$, $1 \leq a, b \leq m'$ in the generative portion of S, for which there are no homologous anchors. We begin by bounding the length of any homologous gap by establishing concentration bounds on homologous anchors in this k-dependence case similar to the prequel [13, 25]. The lemmas in this substring make use of the generative region of S, which has length m' but the final inequalities are in terms of $|S'| = m$. This is fine, since under EC, the two lengths are equivalent up to a constant ($m' = cm$ for a constant $c > 0$). We will represent fixed constants with variants of c; the exact values do not make a difference in the analysis.

▶ **Theorem 29.** *(Yu and Shaw) Suppose we have $X = \sum_{a\in A} Bernoulli_a(q)$ for some $0 < q < 1$. A proper cover of A is a family of subsets $\{A_i\}_{i\in I}$ such that all random variables in $A_i \subset A$ are independent and $\bigcup_{i\in I} A_i = A$. Let $\chi(\mathcal{A})$ be the minimum size of the cover, $|I|$, over all possible proper covers. Then for t ≥ 0,*

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

▶ **Lemma 30.** $Pr(N_H \leq m'(1-\theta_T)^k - t) \leq exp(-\frac{8t^2}{25m'k(1-\theta_T)^k})$

Proof. We use the previous theorem with $q = (1-\theta_T)^k$. Let $A(i)$ denote the random variable taking on the value of 1 if $S[i : i+k-1]$ has a homologous k-mer in S'. Then each set $A_j = \{A(j+tk) \mid t \geq 0 \land j+tk \leq p+m'\}$ contains mutually independent random variables. Note $A = \bigcup_{i \in \{1,\ldots,m'\}} A_i$, and thus the A_i form a partition of A. This implies that $\chi(\mathcal{A}) \leq k$ and the result follows. ◀

We can now apply this lemma to bound homologous gaps in the generative region of S exactly as in the prequel.

▶ **Lemma 31.** *For any interval of length ℓ in $S[p+1 : p+m']$, the probability that no homologous anchor occurs is upper bounded by*

$$\exp\left(-\frac{8\ell(1-\theta_T)^k}{25k}\right).$$

Proof. Each interval of length ℓ in $S[p+1 : p+m']$ can be viewed as an identically distributed length ℓ version of it. Applying the previous lemma with $m' = \ell$ and $t = \ell(1-\theta_T)^k$ gives $\Pr(N_H \leq 0) = \Pr(N_H = 0) \leq \exp(-\frac{8\ell(1-\theta_T)^k}{25k})$. ◀

Working in $S[p+1 : p+m']$, we can now apply the previous lemma to bound homologous gaps w.h.p. exactly as in the prequel.

▶ **Lemma 32** (F2)**.** *With probability $\geq 1-\frac{1}{n}$, no homologous gap in $S[p+1 : p+m']$ has size greater than*

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

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

Proof. The proof is identical to Lemma 6 from Yu and Shaw [25]. We will replicate the proof for this crucial lemma.

Let $\ell = g(n) = \frac{50k\ln(n)}{8(1-\theta_T)^k}$. Define $\mathrm{HG}_1, \ldots, \mathrm{HG}_{m'-\ell+1}$ be the random variables indicating if there is a homologous gap of length ℓ at a given position, i.e., $\mathrm{HG}_i = 1$ when no k-mer in $S[p+j : p+j+\ell-1]$ is part of a homologous anchor. Note that $\mathbb{E}[\sum_{i=1}^{m'-\ell+1} HG]_i \leq \frac{m'}{n^2} \leq \frac{1}{n}$. Applying Markov's inequality, we get that $\Pr(\sum_{i=1}^{m'-\ell+1}\mathrm{HG}_i \geq 1) = \Pr(\sum_{i=1}^{m'-\ell+1}\mathrm{HG}_i \geq n * 1/n) \leq \frac{1}{n}$, and the result follows. ◀

4 Recoverability Theorem

We will now move on to the main result: proving that the expected recoverability of seed-chain-extend with indels is $\geq 1 - O\big((\log n)^2\, n^{-C\alpha}\big)$ for large enough n. To show this, all that remains is to prove that there are anchors sufficiently close to the start and end of the homologous path.

To this end, we lower bound the recoverability of any chain under the space $(EC{+}F1{+}F2)$.

▶ **Lemma 33** ((EC+F1+F2))**.** *Let $C = ((i_1, j_1), \ldots, (i_u, j_u))$ be an optimal chain. Let the last point of the homologous path be (i_e, j_e). The recoverability of C can be lower bounded as*

$$R(C) \geq 1 - \frac{i_1 + j_1 + (i_e - i_u) + (j_e - j_u) + kT_0N_C}{|P_H|}$$

.

Proof. Any point $(x, y) \in P_H$ for which $0 \leq x < i_1$ or $0 \leq y < j_1$ or $x \geq i_u + k$ or $y \geq j_u + k$ is clearly not recoverable. There are at most $i_1 + j_1 + (i_e - j_u) + (j_e - j_u)$ such points.

Consider any point $(x, y) \in P_H$ such that $i_1 \leq x \leq i_u$ and $j_1 \leq y \leq j_u$. These points belong to three categories:

1. **Anchor Points**. Since (x, y) lies on an anchor, it is recovered.
2. **Between anchors**. There exist anchors (i_p, j_p) and (i_{p+1}, j_{p+1}) such that $i_p + k - 1 \leq x \leq i_{p+1}$ and $j_p + k - 1 \leq y \leq j_{p+1}$. Then (x, y) lies in an extension box and is recovered.
3. **Clipping anchors or edits**. Here, (x, y) either lies in the extension box of a clipping anchor or results from a sequence of edits overlapping with a clipping anchor of which there can be at most $T_0 k N_C$ points.

Accounting for all cases, we get a total of at most $i_1 + j_1 + (i_e - j_u) + (j_e - j_u) + T_0 k N_C$ unrecovered points on P_H, and the result follows. ◀

Finally, we give the main recoverability result: the expected recoverability of seed-chain-extend with indels is $\geq 1 - O\big((\log n)^2\, n^{-C\alpha}\big)$ for large enough n.

▶ **Theorem 34.** *The expected recoverability of an optimal chain, $C = ((i_1, j_1), \ldots, (i_u, j_u))$, is $\geq 1 - O\big((\log n)^2\, n^{-C\alpha}\big)$ for large enough n.*

Proof. Recall that the recoverability of a chain C is defined as $R(C) = \frac{|Align(C) \cap P_H|}{|P_H|}$. We showed this is $\geq 1 - \frac{i_1+j_1+(i_e-i_u)+(j_e-j_u)+kT_0N_C}{|P_H|}$, in $\mathcal{F} = \mathrm{EC} \cap \mathrm{F1} \cap \mathrm{F2}$ in the previous lemma. Note that $\Pr(\mathcal{F}) \geq 1 - \frac{2}{n} - \frac{3}{n} - \frac{1}{n} = 1 - \frac{6}{n}$.

Working in $\mathcal{F}$: by Lemma 32, there is no homologous gap in $S[p+1 : p+m']$ of length larger than $g(n)$, so $i_1 \leq g(n)$, and since (i_1, j_1) is at least a clipping anchor, $j_1 \leq i_1 + T_0 k$. Thus, $i_1 + j_1 = O(g(n))$. Similarly, $(i_e - j_u) + (j_e - j_u) = O(g(n))$. Note also that $|P_H| \leq T_0 m$. Thus, $\mathbb{E}(\frac{i_1+j_1+(i_e-j_u)+(j_e-j_u)}{|P_H|} \mid \mathcal{F}) = O(\frac{g(n)}{m}) = O(\frac{1}{\sqrt{m}})$.

Using Corollary 28: $\mathbb{E}(\frac{T_0 k N_C}{|P_H|} \mid \mathcal{F}) = O(k^2(1-\theta_T)^k) = O\big((\log n)^2\, n^{-C\alpha}\big)$. Putting the pieces together, we get:

$$\mathbb{E}(R) \geq \mathbb{E}(R \mid \mathcal{F}) \Pr(\mathcal{F}) = \left(1 - O\left(\frac{1}{\sqrt{m}}\right) - O\left((\log n)^2 n^{-C\alpha}\right)\right)(1 - 6/n) = 1 - O\left((\log n)^2 n^{-C\alpha}\right).$$

◀

▶ Remark 35. Under the conditions given in Definition 5, an optimal chain consists exactly of all homologous and clipping anchors. Since clipping anchors always miss at least one point on the path, and this is not fixable through trivial definitional changes of recoverability, the asymptotic recoverability, $1 - O\big((\log n)^2\, n^{-C\alpha}\big)$, is relatively tight. Specifically, one can always expect to miss $O(mk(1-\theta_T)^k)$ points due to clipping anchors. However, it may be possible to prove a recoverability of $1 - O\big(f(m,n)(\log n)\, n^{-C\alpha}\big)$ for some $f(m,n) < \log n$.

5 Conclusion

In this work, we have shown that under the assumptions of Definition 5, the expected recoverability of an optimal chain under indels is $\geq 1 - O\big((\log n)^2\, n^{-C\alpha}\big)$. This result is weaker than the substitution-only case, in which Yu and Shaw [25] proved that the expected recoverability of an optimal chain is $\geq 1 - O(\frac{1}{\sqrt{m}})$. The weaker bound is due to the existence of clipping anchors, which uniquely arise with indels. In the substitution-only case, an anchor either lies entirely on or off the path. However, when the path is kinked, as it is with indels, this is no longer true. In the prequel [25], unrecovered points arise from breaks or at the start/end of the chain – no points are missed in regions covered by homologous anchors or on the ‘sides’ of extension boxes. The main contributions in this work were the development of the bounded expansion and contraction lemmas, which were used to bound the distribution of the spurious anchor count. Additionally, this work introduced a new class of anchors, clipping anchors, which uniquely arise under indels and are fundamental in the sense that trivial algorithmic changes do not recover the points they lose. Lastly, our work gives an exact description of the optimal chain given by seed chain extend under the conditions outlined in the paper: it consists of all homologous and clipping anchors.

References

1 A. V. Aho, M. R. Garey, and J. D. Ullman. The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1(2):131–137, 1972. arXiv:https://doi.org/10.1137/0201008, doi:10.1137/0201008.

2 Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

3 Arturs Backurs and Piotr Indyk. Edit distance cannot be computed in strongly subquadratic time (unless seth is false). In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 51–58, 2015.

4 Bonnie Berger, Michael S Waterman, and Yun William Yu. Levenshtein distance, sequence comparison and biological database search. *IEEE transactions on information theory*, 67(6):3287–3294, 2020.

5 Roy J Britten. Divergence between samples of chimpanzee and human dna sequences is 5%, counting indels. *Proceedings of the National Academy of Sciences*, 99(21):13633–13635, 2002.

6 Boris Bukh and Raymond Hogenson. Length of the longest common subsequence between overlapping words. *SIAM Journal on Discrete Mathematics*, 34(1):721–729, 2020.

7 Mark J Chaisson and Glenn Tesler. Mapping single molecule sequencing reads using basic local alignment with successive refinement (blasr): application and theory. *BMC bioinformatics*, 13:1–18, 2012.

8 Vaclàv Chvátal and David Sankoff. Longest common subsequences of two random sequences. *Journal of Applied Probability*, 12(2):306–315, 1975. doi:10.2307/3212444.

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

10 Arun Ganesh and Aaron Sy. Near-linear time edit distance for indel channels. In *20th International Workshop on Algorithms in Bioinformatics*, page 17, 2020.

11 Ragnar Groot Koerkamp and Pesho Ivanov. Exact global alignment using a* with chaining seed heuristic and match pruning. *bioRxiv*, pages 2022–09, 2022.

12 Pesho Ivanov, Benjamin Bichsel, and Martin Vechev. Fast and optimal sequence-to-graph alignment guided by seeds. In *International Conference on Research in Computational Molecular Biology*, pages 306–325. Springer, 2022.

13 Svante Janson. Large deviations for sums of partly dependent random variables. *Random Structures & Algorithms*, 24(3):234–248, 2004.

14 Marcos Kiwi, Martin Loebl, and Jiří Matoušek. Expected length of the longest common subsequence for large alphabets. *Advances in Mathematics*, 197(2):480–498, 2005.

15 Eugene V Koonin, L Aravind, and Alexey S Kondrashov. The impact of comparative genomics on our understanding of evolution. *Cell*, 101(6):573–576, 2000.

16 Ben Langmead and Steven L Salzberg. Fast gapped-read alignment with bowtie 2. *Nature methods*, 9(4):357–359, 2012.

17 Jüri Lember and Heinrich Matzinger. Standard deviation of the longest common subsequence. *The Annals of Probability*, 37(3):1192 – 1235, 2009. doi:10.1214/08-AOP436.

18 Heng Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, 2018.

19 Eugene W Myers. An o (nd) difference algorithm and its variations. *Algorithmica*, 1(1):251–266, 1986.

20 Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

21 Sergey Nurk, Sergey Koren, Arang Rhie, Mikko Rautiainen, Andrey V Bzikadze, Alla Mikheenko, Mitchell R Vollger, Nicolas Altemose, Lev Uralsky, Ariel Gershman, et al. The complete sequence of a human genome. *Science*, 376(6588):44–53, 2022.

22 Brian D Ondov, Todd J Treangen, Páll Melsted, Adam B Mallonee, Nicholas H Bergman, Sergey Koren, and Adam M Phillippy. Mash: fast genome and metagenome distance estimation using minhash. *Genome biology*, 17:1–14, 2016.

23 Gesine Reinert, Sophie Schbath, and Michael S Waterman. Probabilistic and statistical properties of words: an overview. *Journal of Computational Biology*, 7(1-2):1–46, 2000.

24 Jim Shaw and Yun William Yu. Theory of local k-mer selection with applications to long-read alignment. *Bioinformatics*, 38(20):4659–4669, 2022.

25 Jim Shaw and Yun William Yu. Proving sequence aligners can guarantee accuracy in almost o (m log n) time through an average-case analysis of the seed-chain-extend heuristic. *Genome Research*, 33(7):1175–1187, 2023.

26 Jouni Sirén, Jean Monlong, Xian Chang, Adam M Novak, Jordan M Eizenga, Charles Markello, Jonas A Sibbesen, Glenn Hickey, Pi-Chuan Chang, Andrew Carroll, et al. Pangenomics enables genotyping of known structural variants in 5202 diverse genomes. *Science*, 374(6574):abg8871, 2021.

27 Temple F Smith, Michael S Waterman, et al. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

28 Wojciech Szpankowski. *Average case analysis of algorithms on sequences*. John Wiley & Sons, 2011.

29 Esko Ukkonen. On approximate string matching. In *International Conference on Fundamentals of Computation Theory*, pages 487–495. Springer, 1983.

30 Esko Ukkonen. Algorithms for approximate string matching. *Information and control*, 64(1-3):100–118, 1985.

31 Richard Van Noorden, Brendan Maher, and Regina Nuzzo. The top 100 papers. *Nature News*, 514(7524):550, 2014.

32 Y William Yu, Deniz Yorukoglu, Jian Peng, and Bonnie Berger. Quality score compression improves genotyping accuracy. *Nature biotechnology*, 33(3):240–243, 2015.

33 Yun William Yu and Griffin M Weber. Hyperminhash: Minhash in loglog space. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):328–339, 2022.

6 Appendix

▶ **Lemma 36** (Supplemental Lemma). *For positions $x, y \in [|S|]$, either $f(x) = f(y) = \emptyset$ or $f(x) \neq f(y)$.*

Proof. If x and y are both deleted, then $f(x) = f(y) = \emptyset$. Otherwise, if only one of x or y is deleted, without loss of generality, let it be x, then $f(x) = \emptyset \neq f(y)$. Lastly, let if neither x nor y is deleted then they have unique corresponding positions x', y' on S', implying that $f(x) = x' \neq y' = f(y)$. ◀

Proof of Lemma 20. There are four cases to consider. We proceed with the first case, $|i - h| \geq k$ and $f[i : i + k - 1] \cap [l : l + k - 1] = \emptyset$. The remaining cases follow symmetrically.

Let $X_i = [i : i + k - 1]$. Since $|i - h| \geq k$, we have that $[i : i + k - 1] \cap [j : j + k - 1] = \emptyset$, which implies that $f([i : i + k - 1]) \cap f([h : h + k - 1]) = \emptyset$ by Lemma 36. Furthermore, since $f([i : i + k - 1]) \cap [l : l + k - 1] = \emptyset$, we have that for any $x \in X_i$, $f(x) \notin f([h : h + k - 1]) \cup [l :$

$l+k-1]$. Thus, $f(x)$ can have degree at most 2: one edge from the unconditioned match graph and one due to conditioning on $A(i,j)$. Similarly, all edges touching x must come from the original unconditioned match graph or from conditioning on one of the anchors. Since $|i-h| \geq k$, x has no edge due to conditioning on $A(h,\ell)$, and so x has degree at most 2. With this in place, we continue with the proof.

Assume that there exists a cycle C. Let $C(X_i) \subset X_i$ be the set of all points in X_i in the cycle. If $C(X_i) = \emptyset$, then there still exists a cycle after removing all edges induced by $A(i,j)$. This implies that there is a cycle in the match graph after conditioning on $A(h,\ell)$, contradicting our previous lemma. Thus, $C(X_i)$ is not empty.

We first show that for any $x \in C(X_i)$, x cannot be deleted. This is clear since if x were deleted then it has no edge in the unconditioned match graph and so can only have degree 1 due to some edge coming from conditioning on $A(i,j)$; since a vertex with degree 1 cannot be in a cycle, it follows $x \notin C(X_i)$. This is also establishes that $f(x)$ is not null.

Now, note that each $x_p \in C(X_i)$ has exactly two neighbors in the cycle: $f(x)$ and y_{j+p-i}, the neighbor from conditioning on $A(i,j)$. Specifically, there are $|C(X_i)|$ neighbors due to $A(i,j)$ in $C(X_i)$, each of which must be in the cycle. Each position in $f(C(X_i))$ has a neighbor in the cycle due to $A(i,j)$ and there are exactly $|f(C(X_i))| = |C(X_i)|$ such neighbors. Since each neighbor of $f(C(X_i))$ due to $A(i,j)$ is in X_i and in the cycle, this is exactly $C(X_i)$. This also implies that all neighbors of $C(X_i)$ due to edges from $A(i,j)$ are exactly the points in $f(C(X_i))$.

Considering the subgraph given by $C(X_i) \cup f(C(X_i))$. All vertices in this graph have degree exactly 2. Since all 2-regular graphs contain a cycle, it follows that this subgraph contains a cycle. However, it is a subgraph of the original match graph conditioned on $A(i,j)$, implying that it too contains a cycle, which cannot be true. This completes the proof. ◀

Proof of Lemma 21. From the choice of t_0, it follows that $e^{t_0} \leq \frac{9}{1+8\gamma}$. Note that $f(x) = \frac{x}{1+\gamma(x-1)}$ is increasing for $x > 0$. We have $9 < \frac{e-1}{0.206} + 1 \leq \frac{e-1}{\theta_i} + 1$ because $\theta_i \leq \theta_T < 0.206$. Letting $A = \frac{e-1}{\theta_i} + 1$, the previous two facts give that $e^{t_0} \leq \frac{A}{1+\gamma(A-1)}$, and since $\rho'_i \leq \gamma$, we have additionally that $e^{t_0} \leq \frac{A}{1+\gamma(A-1)} \leq \frac{A}{1+\rho'_i(A-1)}$.

Multiplying both sides of this inequality by ρ'_i gives $e^{t_0}\rho'_i \leq \frac{A\rho'_i}{\rho'_i A+(1-\rho'_i)} < 1$ since $\rho'_i < \gamma < 1$. This final inequality shows why it is necessary to bound ρ'_i away from 1.

The second inequality in the lemma follows by rearrangement and inserting the expression represented by A:

Expanding the inequality gives $(1-\rho'_i)e^{t_0} + e^{t_0}\rho'_i A \leq A$, rearranging we get $(1-\rho'_i)e^{t_0} \leq A(1-\rho'_i e^{t_0})$. Simplifying yields $\frac{(1-\rho'_i)e^{t_0}}{1-\rho'_i e^{t_0}} \leq A$. Plugging in $A = \frac{e-1}{\theta_i} + 1$ and multiplying both sides by θ_i gives $\theta_i(\frac{(1-\rho'_i)e^{t_0}}{1-\rho'_i e^{t_0}}) \leq e - 1 + \theta_i$, from which we immediately get $(1-\theta_i) + \theta_i(\frac{(1-\rho'_i)e^t}{1-e^t\rho'_i}) \leq e$. This proves the second claim. ◀

Proof of Lemma 22. Denote the random variable representing the total insertion length at the $p+j$-th coordinate as I_j where

$$I_j = \begin{cases} 0, & \text{with probability } 1-\theta_i, \\ \mathrm{Geom}(1-\rho'_i), & \text{with probability } \theta_i, \end{cases}$$

In particular, for $\ell > 0$, $\Pr(I_j = \ell) = \theta_i(1-\rho'_i)(\rho'_i)^{\ell-1}$. Define $Z = \sum_{j=1}^{k} I_j$, which represents the total insertion length, the expansion, of the first k-block. A simple Chernoff bound shows that for any $t > 0$,

$$\Pr(Z \geq c) \leq \frac{\mathbb{E}[e^{tZ}]}{e^{tc}} = \frac{\prod_{j=1}^{k} \mathbb{E}[e^{tI_j}]}{e^{tc}} = \frac{\mathbb{E}[e^{tI_1}]^k}{e^{tc}}.$$

Where the first equality follows since the $\{I_j\}_{j=1}^{k}$ are independent and the second from them being identically distributed. Choosing $t = t_0 = \frac{1}{2}\ln(\frac{9}{1+8\gamma})$, as in the previous lemma, we can calculate the moment generating function of I_1 can be directly:

$M_{I_1}(t_0) = \mathbb{E}[e^{t_0 I_1}] = (1-\theta_i)+\sum_{j=1}^{\infty} \theta_i(1-\rho_i')(\rho_i')^{j-1}e^{t_0 j} = (1-\theta_i)+\theta_i(1-\rho_i')e^{t_0}\sum_{j'=0}^{\infty}(\rho_i' e^{t_0})^{j'}$. The last term is a geometric series which converges since $\rho_i' e^t < 1$ by Lemma 21. Thus, $M_{I_1}(t) = 1 - \theta_i + \theta_i \frac{(1-\rho_i')e^{t_0}}{1-\rho_i' e^{t_0}}$, which the previous lemma (Lemma 21) shows is at most e.

Thus, $\Pr(Z \geq c) \leq e^{k - t_0 c}$. Choosing $c = \frac{1}{t_0}(\frac{2}{\beta}+1)$ gives $\Pr(Z_1 \geq \frac{1}{t_0}(\frac{2}{\beta}+1)k) \leq e^{-(t_0\frac{1}{t_0}(\frac{2}{\beta}+1))C\beta\ln(n)} \leq e^{-2\ln(n)} = \frac{1}{n^2}$ since $C > 1$.

Define Z_i to be the random variable denoting the expansion of the $S[p+i : p+i+k-1]$, the i-th block in $S[p+1 : p+m']$, formally, Z_i is the sum of the insertion lengths at each position in the k-mer $S[p+i : p+i+k-1]$. Note that each Z_i has the same distribution as Z.

A simple union bound shows that $\Pr(\exists j : Z_j \geq \frac{1}{t_0}(\frac{2}{\beta}+1)k) \leq (m-k+1)\frac{1}{n^2} \leq n\frac{1}{n^2} = \frac{1}{n}$, and the result follows. ◀

Proof of Lemma 23. The proof follows similarly to the previous lemma. Define

$$X_j = \begin{cases} 0, & \text{if the } j\text{-th index of } S[p+1:p+m'] \text{ is deleted (with probability } \theta_d), \\ 1, & \text{otherwise.} \end{cases}$$

Note if $X_j = 1$, then the j-th index of $S[p+1 : p+m']$ survives. Let $X = \sum_{j=1}^{\ell} X_j$ be the total number of surviving indices in the ℓ-block and set $q_d = 1 - \theta_d$ to be the survival rate of an index. A classic Chernoff bound on the sum of i.i.d. Bernoulli random variables gives for any $0 < \delta < 1$: $\Pr(X \leq (1-\delta)q_d\ell) \leq \exp(-\frac{\delta^2 q_d \ell}{2})$. Since $\ell = ck = c\beta C\ln(n)$ where $c = \frac{21}{\beta}$ gives $\frac{\delta^2}{2} q_d c_0 C\beta \geq \frac{1}{8}(.794)21 > 2$, when $\delta = 1/2$ and $\theta_d \leq \theta_T \leq 0.206$. Thus, $\Pr(X \leq \frac{1}{2}q_d\ell) \leq \exp(-2\ln(n)) = \frac{1}{n^2}$.

A union bound over all ℓ-blocks in $S[p+1 : p+m']$ gives:

$$\Pr(\exists\ell\text{-block shrunk to less than } \frac{(1-\theta_d)\ell}{2}) \leq n/n^2 = \frac{1}{n}.$$

◀

Proof of Lemma 24. Let $S_p = \{(i,j) \in S \times S' \mid [(i,j), \ldots, (i+k-1, j+k-1)] \cap P_H = \emptyset\}$ be the set of all positions where an anchor at that position does not intersect the homologous path.

Define $B_k(i,j) = \{(h,l) \in S_p \mid |h-i| \leq k \ \cap \ |l-j| \leq k\}$, and

$$\begin{aligned} P_k(i,j) = \ \big\{(h,l) \in S_p : f^{-1}\big([l : l+k-1]\big) \cap [i : i+k-1] \neq \emptyset, \\ f^{-1}\big([j : j+k-1]\big) \cap [h : h+k-1] \neq \emptyset\big\}. \end{aligned}$$

By Lemma 20, for any $(h,l) \notin B_k(i,j) \ \cup \ P_k(i,j)$, $A(i,j)$ and $A(h,l)$ are independent. Then, $N_S = \sum_{(i,j)\in S_p} A(i,j)$. We calculate the variance as follows:

$$
\begin{aligned}
N_S^2 &= \sum_{(h,l)\in S_p}\sum_{(i,j)\in S_p} A(i,j)A(h,l)\\
&= \underbrace{\sum_{(h,l)\in S_p\setminus\left(B_k(i,j)\cup P_k(i,j)\right)}\sum_{(i,j)\in S_p} A(i,j)A(h,l)}_{S_1} + \underbrace{\sum_{(h,l)\in S_p\cap B_k(i,j)}\sum_{(i,j)\in S_p} A(i,j)A(h,l)}_{S_2}\\
&\quad + \underbrace{\sum_{(h,l)\in\left(S_p\cap P_k(i,j)\right)\setminus B_k(i,j)}\sum_{(i,j)\in S_p} A(i,j)A(h,l)}_{S_3}.
\end{aligned}
$$

Dealing first with S_1: by the independence lemma (Lemma 20), $A(h,l), A(i,j)$ are independent, so:

$$
\begin{aligned}
\mathbb{E}(S_1) &= \sum_{(h,l)\in S_p\setminus\left(B_k(i,j)\cup P_k(i,j)\right)}\sum_{(i,j)\in S_p} \mathbb{E}\big(A(i,j)\big)\,\mathbb{E}\big(A(h,l)\big)\\
&\le \sum_{(h,l)\in S_p}\sum_{(i,j)\in S_p} \mathbb{E}\big(A(i,j)\big)\,\mathbb{E}\big(A(h,l)\big) = \mathbb{E}(N_S)^2.
\end{aligned}
$$

Note $\mathbb{E}(A(h,l)A(i,j)) \le \mathbb{E}(A(i,j))$ since the anchor random variables take values in $\{0,1\}$. Using that $|B_k(i,j)| \le 4k^2$, $\Pr(A(i,j)=1) = \frac{1}{\sigma^k}$ for $(i,j)\in S_p$ from Corollary 18, and the naive bound $|S_p| \le mn$, we get:

$$
\mathbb{E}(S_2) \le \sum_{(h,l)\in S_p\cap B_k(i,j)}\sum_{(i,j)\in S_p} \mathbb{E}(A(i,j)) \le 4k^2\frac{mn}{\sigma^k}.
$$

Lastly, we handle the S_3 term. Working under EC, the region $S[i:i+k-1]$ can expand to have at most $(\frac{1}{t_0}(\frac{2}{\beta}+1)+1)k+2k$ corresponding positions l on S'. Note the additional $2k$ comes from k-mers that start before and after the corresponding region but still intersect it. The same argument shows that there are at most $(\frac{1}{t_0}(\frac{2}{\beta}+1)+1)k+2k$ positions j that correspond to $S[h:h+k-1]$. Thus, $|P_k(i,j)| \le ((\frac{2}{\beta}+1)+1)k+2k)^2 = (\frac{1}{t_0}(\frac{2}{\beta}+1)+3)^2k^2$. For simplicity, let the constant $T_0 = 2\max((\frac{1}{t_0}(\frac{2}{\beta}+1)+3)^2, 4)$.

This yields,

$$
\mathbb{E}(N_S^2) \le \mathbb{E}(N_S)^2 + T_0k^2\frac{mn}{\sigma^k}.
$$

From which it immediately follows that $\operatorname{var}(N_S) \le T_0k^2\frac{mn}{\sigma^k}$. ◀

Proof of Lemma 25. Call the event $X = \{N_S \ge n^{2-C} + \sqrt{T_0}\,C\log(n)\,n^{\frac{3-C}{2}}$, i.e., the event that there are more spurious anchors than the amount given by the expression. By the law of total probability:

$$
\begin{aligned}
\Pr(X) &= \Pr\big(X \mid EC\big)\ \Pr(EC) + \Pr\big(X \mid EC^c\big)\ \Pr(EC^c)\\
&\le \Pr\big(X \mid EC\big) + \Pr(EC^c).
\end{aligned}
$$

We first bound $\Pr(X \mid EC)\Pr(EC) \le \Pr(X \mid EC)$. To do this, let us first show that $\frac{mn}{\sigma^k} \le n^{2-C}$. Recall that $k = C\log(n)$. This follows from noting that $\log(m) \le \log(n)$, since $m \le n$ under EC, and rearranging, $(C-1)\log(n) + \log(m) \le C\log(n) = k$, from which we get $\log(mn) - k \le (2-C)\log(n)$ and, finally, $\log(\frac{mn}{\sigma^k}) \le \log(n^{2-C})$, which gives the inequality.

Now, by Chebyshev, $\Pr(N_S \ge \mathbb{E}(N_S) + \sqrt{n\mathrm{var}(N_S)}) \le \frac{1}{n}$. We can upper bound $\Pr(X \mid EC)$ by using the variance bound of N_S given by Lemma 24 and the previous inequality. This yields: $\mathrm{var}(N_S) \le T_0 k^2 \frac{mn}{\sigma^k}$, so $\frac{1}{n} \ge \Pr(N_S \ge \mathbb{E}(N_S) + \sqrt{n\mathrm{var}(N_S)}) \ge \Pr(N_S \ge \mathbb{E}(N_S) + \sqrt{T_0} C \log(n) n^{\frac{3-C}{2}})$. Finally, bounding $\mathbb{E}(N_S) \le \frac{mn}{\sigma^k}$ and using our previously shown inequality, we get the full result that $\Pr(X \mid EC) \le \frac{1}{n}$.

For the second term, $\Pr(EC^c) \le \frac{2}{n}$, by a simple union bound from Lemma 22 and Lemma 23.

Combining terms, $\Pr\Big(N_S \ge n^{2-C} + \sqrt{T_0}\, C \log(n)\, n^{\frac{3-C}{2}}\Big) \le \frac{3}{n}$. ◀