4-string-sketching

Monday, February 2, 2026 10:58 AM

Last week, we discussed ways to analyse a genome by turning it into a bag of k-mers. However, this loses a lot of sequential information. Today, we try to preserve the linear information of strings.

Outline

- Sampling schemes
- Minimizers
- Density
- Frac Min Hash
- Open Syncmers
- ANI & genome distance

Problem: How do we find appropriate anchors in seed + extend for strings S & T.

Sol 1: Check all k-mers in S against all k-mers in T. ($|S| = m$, $|T| = n$)

Analysis: Time: $O(mn)$ naively.

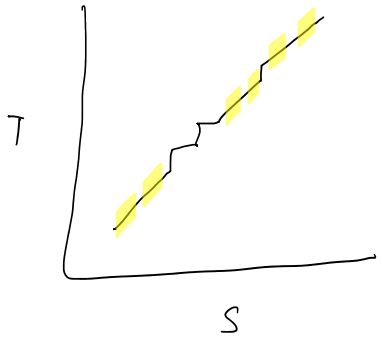
$O(m+n)$ using hash tables or suffix trees

Space: $O(m+n)$ auxiliary space potentially.

This is pretty good asymptotically.

In 02-752, we are interested in more than asymptotics, so a lot of today is going to focus on constants.

How many anchors do we actually need?



} many potential anchor positions
only need 1 for seed + extend
or few more for seed-chain-extend

Sol 2: Sample the k-mers from T, and match them against all possible locations in S.

Sol 2: Sample the k-mers from T, and match them against all possible locations in S.

Analysis: $O(n)$ time in index S.

$O(\#\text{samples})$ time for T. ← how many samples do we need?

Exercise: Given a random string S of length n and a mutated substring T where letters are randomly substituted with probability θ, derive a formula for the number of k-mers you need to sample from T so that in expectation, you have at least one match.

Often, we choose a constant fraction $\frac{1}{c}$ of the k-mers, which would still take $O(n)$ time. ← not asymptotically better, but an order-c speedup in practice

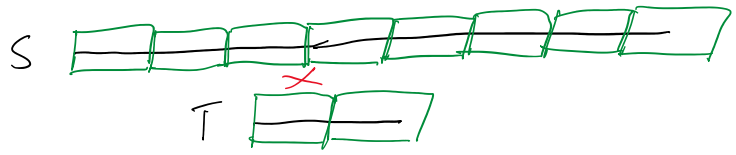
Although sampling may take $O(n)$ time to read through T, space complexity is now $O(m + \frac{n}{c})$.

Sol 3: Can we subsample both S & T in a coordinated fashion?

Recall that we did something similar with MinHash!

Idea: Do a MinHash on chunks of S & T.

Problem: Chunks might not align with each other.



Solution: Overlap the chunks instead. We will call them windows.

Problem: This keeps an awful lot of separate MinHash sketches.

Observation: MinHash sketches of adjacent windows are extremely similar, so they compress together

Put together, this is the basic idea behind minimizers by Schleimer et al [2003] Roberts et al [2004]

Terminology & Definitions

Define: Given parameters w & k, a window is a string containing

Define: Given parameters w & k, a **window** is a string containing exactly w k-mers, of length $\ell = w + k - 1$.

Define: A **local sampling scheme**, for $w \geq 2$ & $k \geq 1$, is a function $f : \Sigma^{\ell} \to [w]$, where Σ is the alphabet & $|\Sigma| = \sigma$

Given a window W, it chooses a single k-mer from W.

(NB: position implies a k-mer string)

Why is locality important?

Consider two strings $S = ATCCTA$, $T = TCCTAG$

Take every other 3-mer

S: TCC, CTA

T: CCT, TAG

} no overlap in sample despite almost perfect overlap of strings.

So we want it to be true that given a window W, no matter what else might come before or after W, the same k-mer is selected.

Possibility: A local sampling scheme might jump around in its sampling backward & forward.

e.g. ATCCGATTACA

window 1: sample ATTA

window 2: sample TCCG

} k-mer sampled jumped backwards if we think of sliding windows along

Define: A local sampling scheme is **forward** if $\forall$ windows W_1, W_2, where $pos(W_1) < pos(W_2)$, the sampled k-mer from W_2 cannot come before the sampled k-mer from W_1.

Intuitively, as we slide windows along, the sampled k-mers also move forward.

Define: An order $\mathcal{O}_k$ on k-mers is a function $\mathcal{O}_k : \Sigma^k \to \mathbb{R}$.

$\forall x, y \in \Sigma^k$, we write $x \leq_{\mathcal{O}_k} y$ iff $\mathcal{O}_k(x) \leq \mathcal{O}_k(y)$.

} It is **total** if $\mathcal{O}_k$ is **injective**. Every k-mer maps to a unique number

Define: A **minimizer scheme** is defined by a total order $\mathcal{O}_k : \Sigma^k \to \mathbb{R}$ and samples the leftmost minimal k-mer in a window W, the **minimizer**

$$f(W) := \text{argmin}_{i \in [w]} \mathcal{O}_k(W[i \ldots i+k])$$

Aside: when sampling, we track both the k-mer & its position as a tuple

Define: The **lexicographic minimizer** is the minimizer that sorts all k-mers lexicographically.

Define: The lexicographic minimizer is the minimizer that sorts all k-mers lexicographically.

AAAA < AACT < TAAA by alphabetical ordering.

Define: The random minimizer uses some uniform random total order O_k.
(maybe from a good hash function or permutation)

Define: The density of a sampling scheme (or any k-mer string sketch) is the expected fraction of k-mers sampled on a random string as string length goes to ∞.

Example: Window size 3

k-mers	hash values	windowed minimums	deduplicated minimizers
AGATTACATTA			
AGAT →	0.451		
GATT →	0.712	0.121	
ATTA →	0.121	0.121	0.121, ATTA, pos 2
TTAC →	0.812	0.121	
TACA →	0.612	0.612	0.612, TACA, pos 4
ACAT →	0.934	0.612	
CATT →	0.771	0.121	0.121, ATTA, pos 7
ATTA →	0.121		

Notice that although you select a (kmer, pos) tuple (minimizer) for each window, there are fewer minimizers than windows because they are often shared between adj windows

Define: For forward schemes, a context is a string of length $c = w+k$, consisting of two overlapping windows. (w+1 k-mers)

Define: A context is charged when two different positions are sampled from the two windows. [Marçais, ..., Kingsford 2017]

Intuitively, charged contexts measure the number of times the minimizer changes as we walk along the string

Thm: The density of a forward scheme is given by the fraction of all possible contexts that are charged.
or equivalently, the probability that a random context is charged.

Exercise: Prove this theorem formally.

Thm: The density of the random minimizer is $< \frac{2}{w+1}$. [Marcais, et al 2017]

Actual proof is fairly involved, but here is a slightly easier theorem.

Thm. If $k > 4\log_\sigma w$, the density of a random minimizer is $< \frac{2}{w+1} + \frac{1}{w^2}$

proof. **Lemma:** For a minimizer scheme, a context is charged if and only if the smallest k-mer is either the very first, or very last position.

pf. Suppose not. The the smallest k-mer is shared between both windows, so the context isn't charged, a contradiction. $\rightarrow\!\!|\leftarrow$

Lemma: If $k > 4\log_\sigma w$, the probability that a random context of w k-mers contains two identical k-mers is $< \frac{1}{w^2}$.

pf. sketch: For any pair of nonoverlapping k-mers, the probability they are equal is $\sigma^{-k} < \sigma^{-4\log_\sigma w} = \frac{1}{w^4}$.

Turns out, the same is true even if they overlap. (so long as the overlap is not the entire k-mer)

There are $\binom{w+1}{2} = \frac{w(w+1)}{2} = \frac{w^2+w}{2} < w^2$ pairs of k-mers

So by a union bound, the probability that any two k-mers are equal is $< \frac{1}{w^2}$. □

Combining the two lemmas, with probability $\geq 1 - \frac{1}{w^2}$, there are no duplicate k-mers in the context, which means that the probability the context is charged is $\frac{2}{w+1}$ (for the smallest k-mer being at front or end).

One the other hand, certainly in the $\frac{1}{w^2}$ chance of a duplicate k-mer, the probability of being charged is ≤ 1 (by def.).

Summing together, we prove the theorem. □

Exercise: A De Bruijn sequence of order c is any circular sequence (sequence loops around, e.g. A A C G C T G) of length σ^c that contains every sequence of length c exactly once.

Prove the following theorem:

The density of any forward scheme is $\frac{D}{\sigma^c}$, where D is the number of charged contexts for that forward scheme on a De Bruijn sequence of order $c = w+k$.

Notice: The density of a sampling scheme is (trivially) lower bounded by $\frac{1}{w}$.

This is because you must select at least one k-mer in every window.

For a while, $\frac{2}{w+1}$ was conjectured to be the lower bound, but this conjecture was proven false by Orenstein, Pellow, Marçais, Shamir, & Stougford in 2016.

One key point of minimizers is that you are always guaranteed to have one in a window. This ensures that for example, every read can be mapped to the reference. So minimizers have become a default for string sketching, much as MinHash has for set sketching.

But what if we don't need the window guarantee?

FracMinHash [Irber, et al 2022] (name given, though idea had been floating around?)

Instead of choosing k-mers based on relative ordering by hash within a window, we can do something simpler & just choose by the hash value itself.

We choose a parameter d & a hash function $h: \Sigma^k \to [0, 1]$.
For every k-mer x, select it if $h(x) < \frac{1}{d}$. (and keep track of position)

Although not a local sampling scheme as we defined earlier, it is still capturing similar properties.

Density? $\frac{1}{d}$ by construction.

No window guarantee.
Also has some sense of locality, in that it doesn't depend on anything except the k-mer itself.

Pro: an unbiased random sample of the k-mer spectrum

Pro: an unbiased random sample of the k-mer spectrum

Paper: Belbasi, et al (2022) show one place where minimizers are biased, messing up a read mapping tool mashmap.

Sometimes, we will sample both a k-mer & its neighbor. This seems inefficient because there's overlapping information. Can this be avoided?

GATTACACCAATTCG

GATT
ATTA
ACCA
CCAA

} sampled k-mers not well spread apart

Open syncmers [Edgar, 2021]

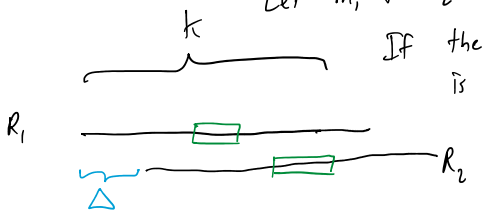
Parameters k, s, t, hash function $h: \Sigma^s \to [0, I]$ and $s < k$, $t \leq k-s+1$

1. For every k-mer in S, look at constituent s-mers.
2. Take all s-mers in a k-mer and hash them.
3. If the smallest hashed s-mer is in position t in the window, select the k-mer.

Density: $\approx \frac{1}{k-s+1}$ Exercise Prove this.

(Spread) Lemma: If $t = \left\lceil \frac{k-s+1}{2} \right\rceil$, the middle s-mer of the k-mer, then Consecutive open syncmers are at least $t-1$ distance apart.

pf. Suppose not. Let R_1 & R_2 be the two selected consecutive open syncmers. Let m_1 & m_2 be the smallest s-mers within R_1 & R_2 respectively.



If the offset b/t R_1 & R_2 is Δ, then m_2 is at position $\Delta + t$ relative to R_1.

If $\Delta < t-1$, then m_2 is at position $< 2t-1$, which is $< k-s$ so m_2 is on R_1

Conversely, m_1 is at position $t > \Delta$, so m_1 is on R_2.

But then both $m_1 < m_2$ & $m_2 < m_1$, a contradiction. →|←

Exercise: Let's define closed syncmers as k-mers where the smallest s-mer is either at the beginning or the end.

[illegible] are guaranteed to be spread apart &

Exercise: ... s-mer is either at the beginning or the end.

Prove that closed syncmers are guaranteed to be spread apart & determine the minimum spread.

This spread property also turns out to be theoretically useful for some proofs.

Genome comparisons

Recall: We can compute divergence of two genomes via Jaccard index.
- divergence: measurement of point mutations differing between two genomes

Caveat: This can capture the evolutionary distance b/t closely related strains of bacteria, but starts failing as evolutionary distance increases.

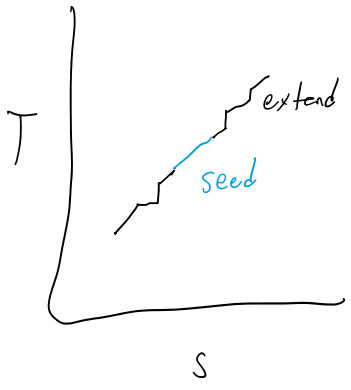
Why? Much larger structural variation.
- e.g. large duplications, deletions, insertions, or inversions
- A single event can require thousands of point mutations to model.

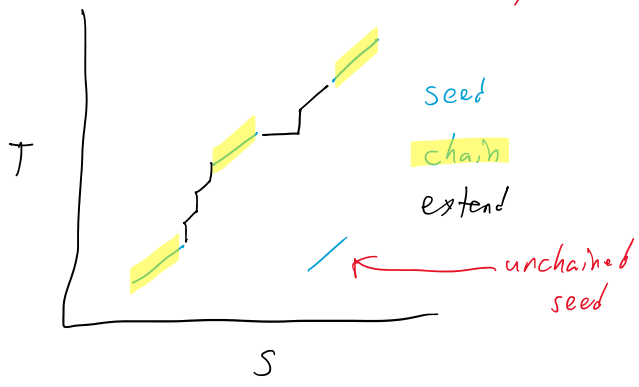
Idea: Let's factor out the big events by only computing divergence in orthologous regions, a metric known as ANI (Average Nucleotide Identity).

Sol 1: align two entire genomes: $O(n^2)$

Sol 2: map large sections of genome to each other.
- Faster, but still requires DP locally

Sol 3: Instead of seed-extend or seed-chain-extend,
- similar to seed & extend but we use many seeds together





we just seed + chain to get an approx alignment, and then look at divergence in similar regions

Fast way to measure bacterial genome similarity and define species boundaries.

Reading: Skani covers this topic in detail, and is how the Genome Taxonomy Database determines species boundaries. [Shaw, Yu, 2023]