Seed-chain-extend alignment is accurate and runs in close to $O(m \log n)$ time for similar sequences: a rigorous average-case analysis

Jim Shaw and Yun William Yu

University of Toronto, Canada

RECOMB 2023

UNIVERSITY OF TORONTO

Introduction

Consider two strings of length n and m, $n \geq m$.

- Sequence alignment – optimally solved in worst-case $O(mn)$ time

Introduction

Consider two strings of length n and m, $n \geq m$.

- Sequence alignment – optimally solved in worst-case $O(mn)$ time
- Dynamic programming (Smith and Waterman, 1981; Needleman and Wunsch, 1970)

Problems

Consider *read alignment* – $n \approx 3,000,000,000$, $m \approx 150 - 1,000,000$

- Optimal methods are slow, so heuristics are used – BWA (Li and Durbin, 2009), minimap2 (Li, 2018), bowtie2 (Langmead and Salzberg, 2012), etc

Problems

Consider *read alignment* – $n \approx 3,000,000,000$, $m \approx 150 - 1,000,000$

- Optimal methods are slow, so heuristics are used – BWA (Li and Durbin, 2009), minimap2 (Li, 2018), bowtie2 (Langmead and Salzberg, 2012), etc
- But heuristics lack theoretical guarantees

Question

Can we put non-trivial theoretical guarantees on heuristic alignment methods?

Seed-chain-extend

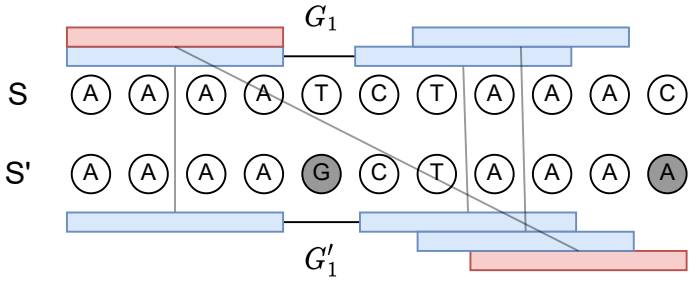
- We analyze the *seed-chain-extend* alignment heuristic

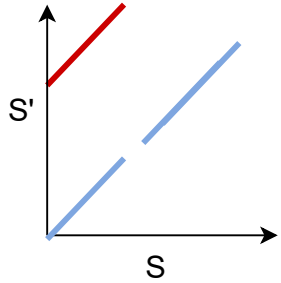
Seed-chain-extend steps

Align two strings S, S' of length n and $m \leq n$.

1. **Seeding and matching k-mers**

Seeding and matching k-mers

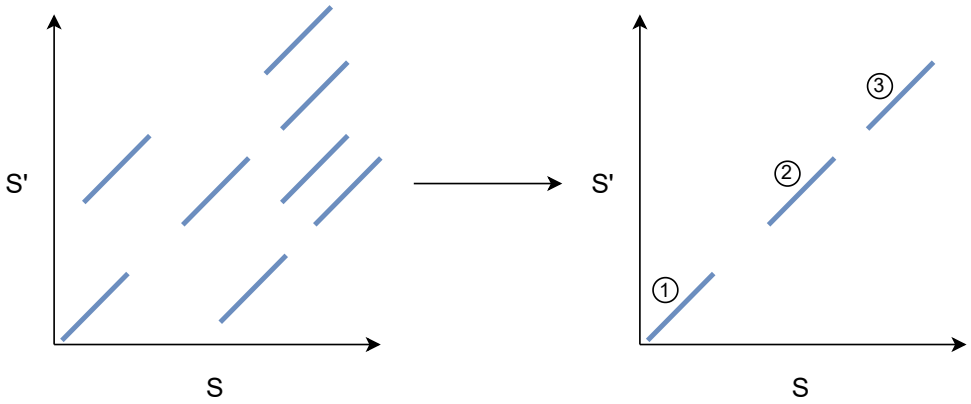




Seed-chain-extend steps

1. Seeding and matching k-mers
2. **Obtain chain (sequence) of k-mer matches (anchors)**

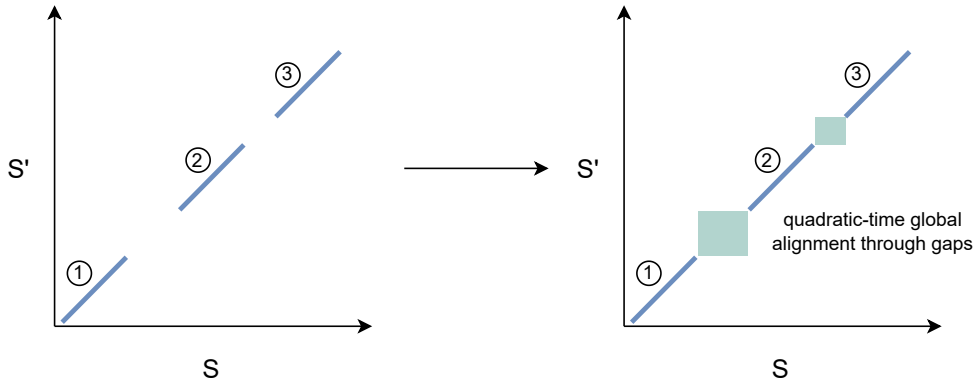
Obtain chain (sequence) of k-mer matches (anchors)



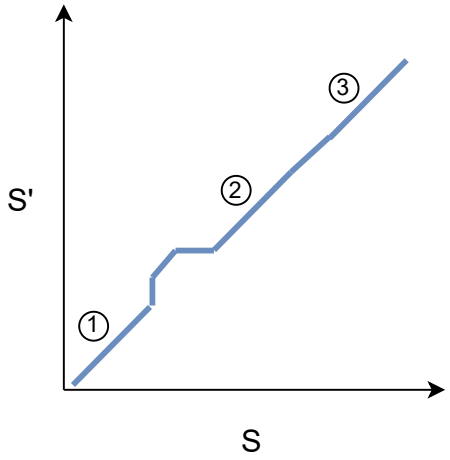
Seed-chain-extend steps

1. Seeding and matching k-mers
2. Obtain chain (sequence) of k-mer matches (anchors)
3. **Dynamic programming extension through gaps**

Dynamic programming extension through gaps



Dynamic programming extension through gaps



Seed-chain-extend use-cases

- Used in read-to-genome (e.g. minimap2) or genome-to-genome alignments (e.g. MUMmer (Marçais et al., 2018))

Seed-chain-extend use-cases

- Used in read-to-genome (e.g. minimap2) or genome-to-genome alignments (e.g. MUMmer (Marçais et al., 2018))
- Worst-case still $O(mn)$ – how do we explain the better than $O(mn)$ behavior? (Medvedev, 2022)

Seed-chain-extend use-cases

- Used in read-to-genome (e.g. minimap2) or genome-to-genome alignments (e.g. MUMmer (Marçais et al., 2018))
- Worst-case still $O(mn)$ – how do we explain the better than $O(mn)$ behavior? (Medvedev, 2022)
- **Average-case analysis**

Random sequence model

1. "reference genome" S - uniformly random string of nucleotides, length n

Random sequence model

1. "reference genome" S - uniformly random string of nucleotides, length n
2. "read" S' - substring of S, length $m \leq n$, i.i.d substitutions with probability θ

Random sequence model

1. "reference genome" S - uniformly random string of nucleotides, length n
2. "read" S' - substring of S, length $m \leq n$, i.i.d substitutions with probability θ
3. k-mer length $k = C \log n$ for some $C(\theta) \approx 2$.

Runtime

- Runtime $= T_{Seed} + T_{Chain} + T_{Extension}$
- $T_{Seed}, T_{Chain}, T_{Extension}$ are **random variables**

Runtime

- Runtime $= T_{Seed} + T_{Chain} + T_{Extension}$
- $T_{Seed}, T_{Chain}, T_{Extension}$ are **random variables**
- Want $\mathbb{E}[T_{Seed} + T_{Chain} + T_{Extension}]$
- Focus on $T_{Chain}, T_{Extension}$.

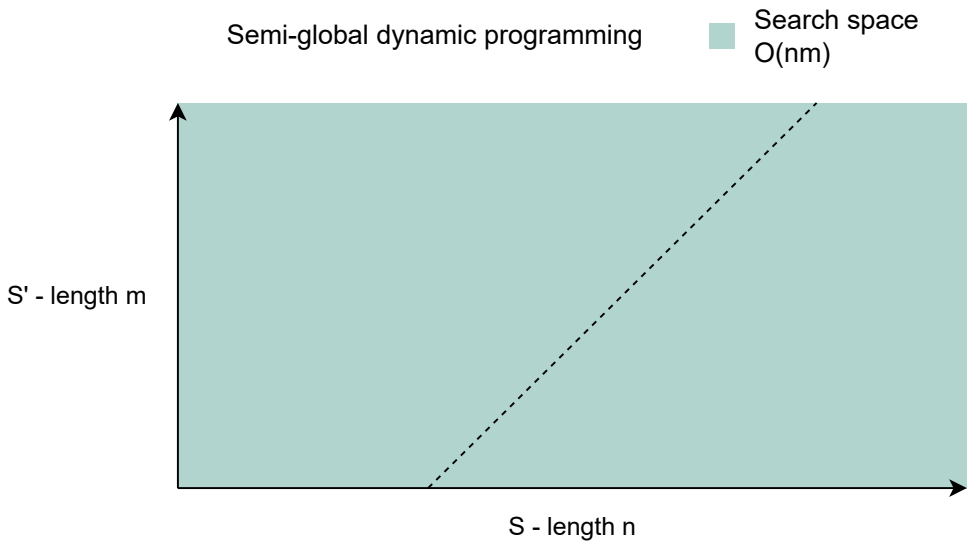
Extension runtime and recoverability

Two questions:

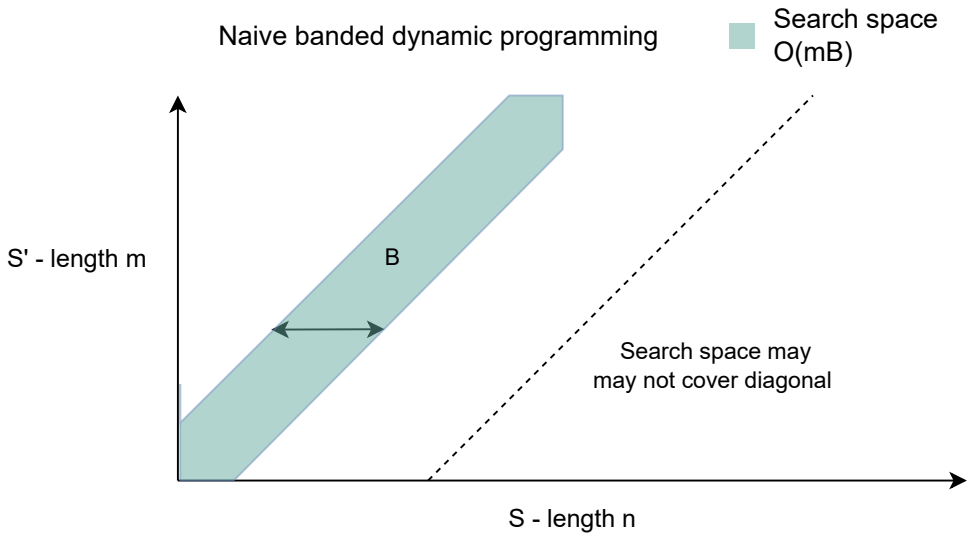
- What is $\mathbb{E}[T_{Extension}]$?
- How good is the resulting alignment?

Random sequence and alignment

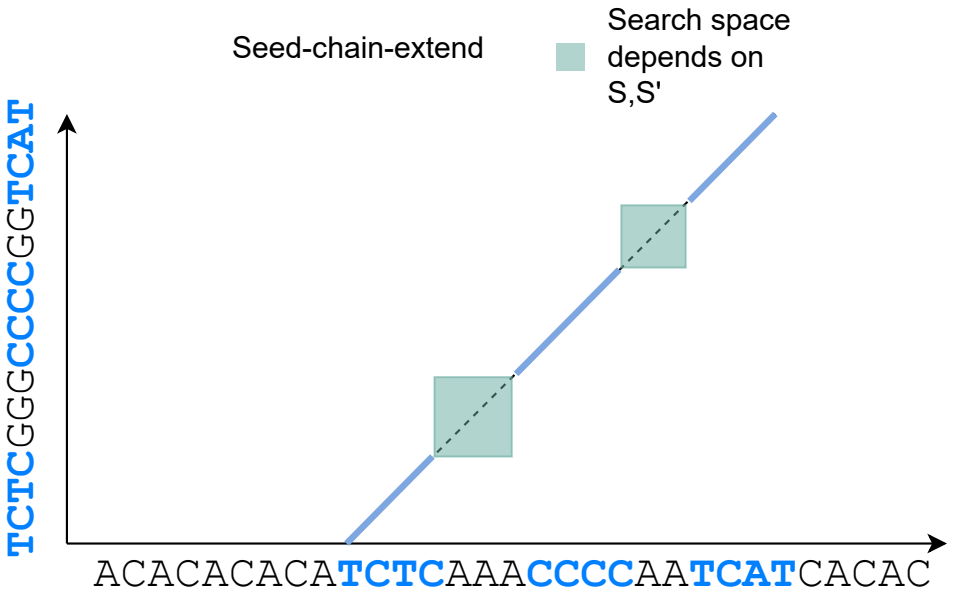
DP search space



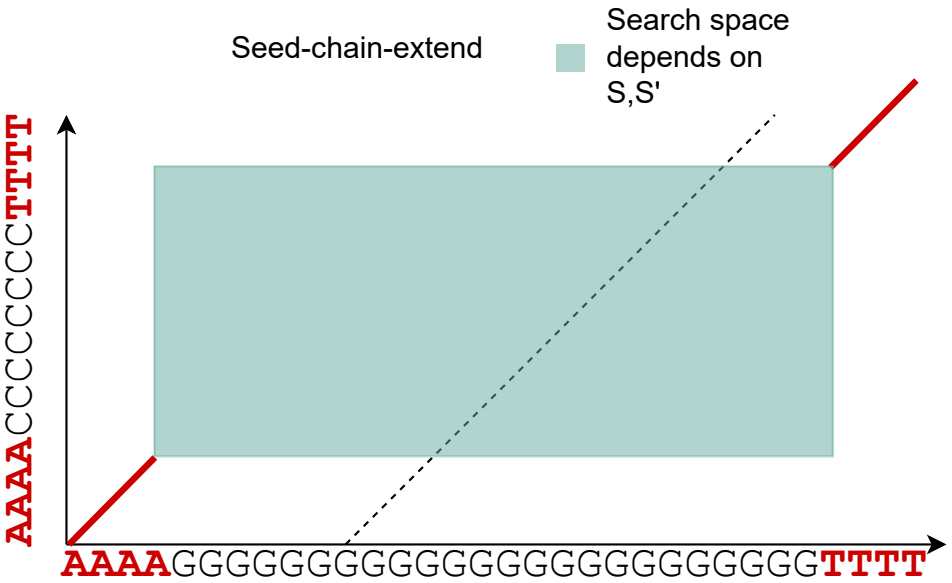
Banded DP search space



Seed-chain-extend search space



Seed-chain-extend search space



Extension runtime and recoverability

1. $\mathbb{E}[T_{Extension}]$ – **expected size search space**

Extension runtime and recoverability

1. $\mathbb{E}[T_{Extension}]$ – **expected size search space**
2. Recoverability – **fraction of diagonal covered by search space and k-mers**

Extension runtime and recoverability

1. $\mathbb{E}[T_{Extension}]$ – **expected size search space**
2. Recoverability – **fraction of diagonal covered by search space and k-mers**

Theorem (Theorem 1 simplified from Shaw and Yu (2023))

$\mathbb{E}[T_{Extension}]$ is $O(mn^{f(\theta)} \log n)$, where $f(\theta) < 2.43 \cdot \theta$. The expected recoverability is $\geq 1 - O(\frac{1}{\sqrt{m}})$.

Proof idea

1. Sums of *almost independent* k-mer random variables $\implies$ no degenerate chainings with high probabililty (Janson, 2004; Ganesh and Sy, 2020)

Proof idea

1. Sums of *almost independent* k-mer random variables $\implies$ no degenerate chainings with high probabililty (Janson, 2004; Ganesh and Sy, 2020)
2. $\mathbb{E}[T_{Extension}] \approx \mathbb{E}[\text{gap size}^2] \leq O(m(1-\theta)^{-k} \log n) = O(mn^{f(\theta)} \log n)$

Runtime of T_{Chain}

- Let N be the number of k-mer matches or anchors (N is a random variable)

Runtime of T_{Chain}

- Let N be the number of k-mer matches or anchors (N is a random variable)
- **Linear gap cost** objective – optimally solved in $T_{Chain} = O(N \log N)$ time (Abouelhoda and Ohlebusch, 2005; Jain et al., 2021)

Runtime of T_{Chain}

- Let N be the number of k-mer matches or anchors (N is a random variable)
- **Linear gap cost** objective – optimally solved in $T_{Chain} = O(N \log N)$ time (Abouelhoda and Ohlebusch, 2005; Jain et al., 2021)

Theorem (Theorem 6 simplified from Shaw and Yu (2023))

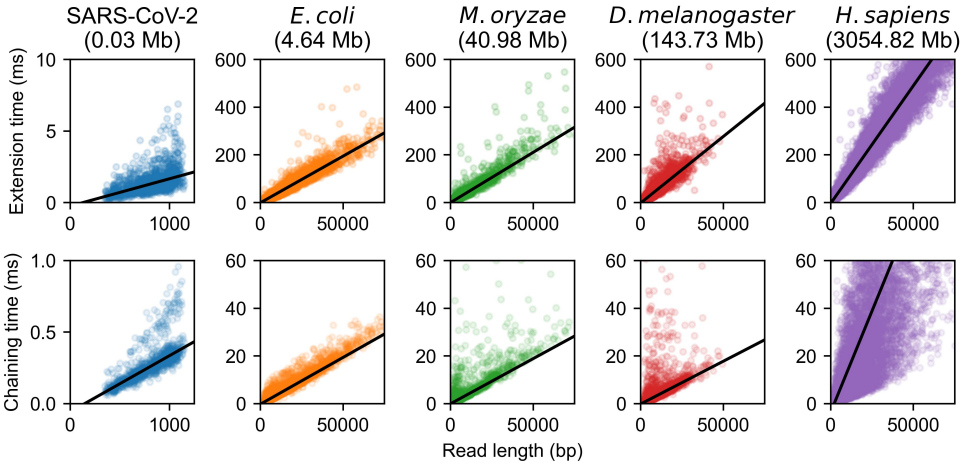
Under our random model with S length n, S' length m, $k = C \log n$,

$$\mathbb{E}[T_{Chain}] = O(m \log m)$$

Real nanopore runtimes

- Are our asymptotic results accurate?
- Took nanopore reads of $\sim 95\%$ accuracy from various species, so $\theta = 0.05$
- Aligned with custom seed-chain-extend aligner, $k = C \log n \approx 2 \log n$

Chaining + Extension runtimes



Extension runtime

- $\mathbb{E}[T_{Extension}] = O(m\ n^{f(\theta)} \log n\ ),\ f(0.05) \approx 0.08$

Extension runtime

- $\mathbb{E}[T_{Extension}] = O(m\, n^{f(\theta)} \log n\ ),\ f(0.05) \approx 0.08$
- Slope of extension runtime $\propto n^{0.08} \log n$ (approximately)

Extension runtime predictions are informative

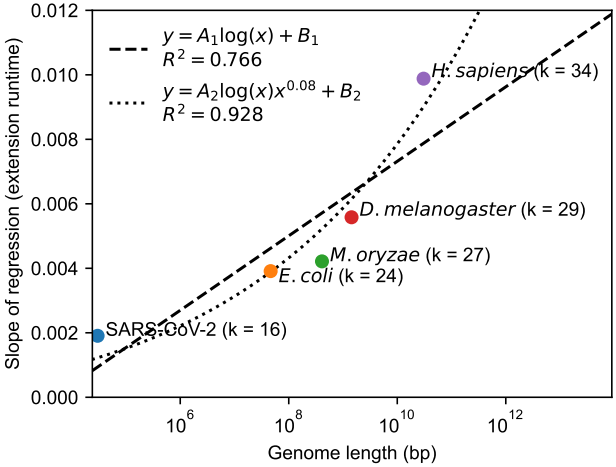


Figure: The $x^{0.08} \log x$ fitted line is better than just $\log x$ as predicted by theory – even with indels!

Sketching result

- Real long-read aligners don't use all k-mers – subsample via *sketching*
- Minimizers (Roberts et al., 2004), syncmers (Edgar, 2021), etc

Sketching

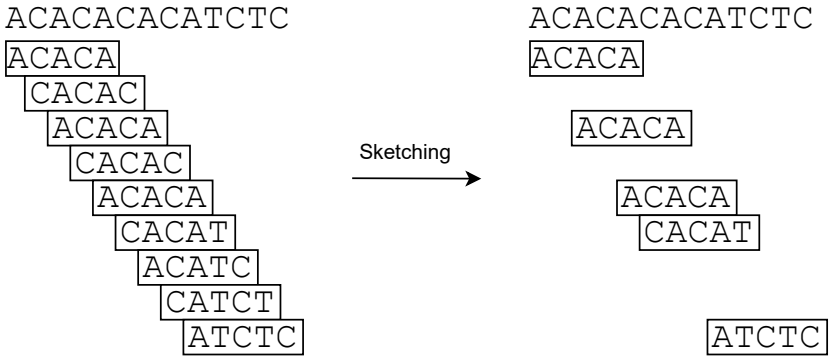


Figure: Sketching – subsampling k-mers

Sketching result

Let $c = \Theta(\log n) < k$.

Theorem (Theorem 2 from Shaw and Yu (2023) simplified)

We can subsample to $\frac{1}{c}$ *of the k-mers using the open syncmer method, and*

- *the* **same extension runtime/recoverability results hold**
- *but* **chaining only takes** $\frac{1}{c}$ **as long**

Sketching result

Let $c = \Theta(\log n) < k$.

Theorem (Theorem 2 from Shaw and Yu (2023) simplified)

We can subsample to $\frac{1}{c}$ of the k-mers using the open syncmer method, and

- *the* **same extension runtime/recoverability results hold**
- *but* **chaining only takes $\frac{1}{c}$ as long**

Sketching provably **reduces** chaining time **without increasing** extension time too much!

Simulation – sketching vs no sketching

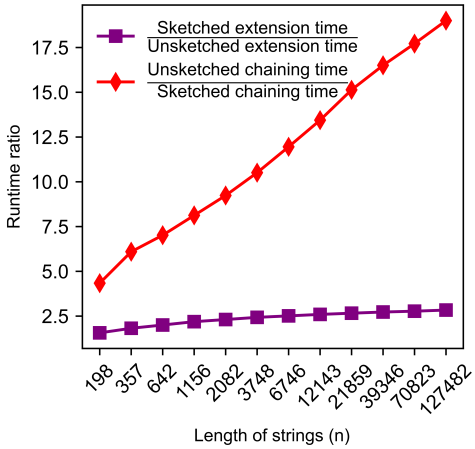


Figure: Align two length n strings with $\theta = 0.10$. Sketching with fraction $c \approx 2 \log n$

Conclusion

Motivation

- Optimal sequence alignment is slow – so we use heuristics

What we did

Conclusion

Motivation

- Optimal sequence alignment is slow – so we use heuristics
- Heuristics have bad worst-case guarantees; not representative of real times

What we did

Conclusion

Motivation

- Optimal sequence alignment is slow – so we use heuristics
- Heuristics have bad worst-case guarantees; not representative of real times

What we did

- Proved $O(mn^{f(\theta)} \log n) \ll O(mn)$ runtime and good accuracy in *expectation* for seed-chain-extend.

Conclusion

Motivation

- Optimal sequence alignment is slow – so we use heuristics
- Heuristics have bad worst-case guarantees; not representative of real times

What we did

- Proved $O(mn^{f(\theta)} \log n) \ll O(mn)$ runtime and good accuracy in *expectation* for seed-chain-extend.
- Bounds also work for sketching and real data; provides justification usage of sketching in long-read aligners

Funding and acknowledgements

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

published in **Genome Research** (advance). Thank you to the anonymous reviewers.

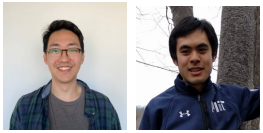


Figure: Jim Shaw, Yun William Yu

Natural Sciences and Engineering Research Council of Canada

Conseil de recherches en sciences naturelles et en génie du Canada

Canada

UNIVERSITY OF TORONTO

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

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

Ganesh A and Sy A. 2020. Near-Linear Time Edit Distance for Indel Channels. *arXiv:2007.03040 [cs, q-bio]* ArXiv: 2007.03040.

Jain C, Gibney D, and Thankachan SV. 2021. Co-linear chaining with overlaps and gap costs. preprint, Bioinformatics.

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

Langmead B and Salzberg SL. 2012. Fast gapped-read alignment with Bowtie 2. *Nature Methods* **9**: 357–359. Number: 4 Publisher: Nature Publishing Group.

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

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

Marçais G, Delcher AL, Phillippy AM, Coston R, Salzberg SL, and Zimin A. 2018. MUMmer4: A fast and versatile genome alignment system. *PLOS Computational Biology* **14**: e1005944. Publisher: Public Library of Science.

Medvedev P. 2022. The limitations of the theoretical analysis of applied algorithms. ArXiv:2205.01785 [cs].

Needleman SB and Wunsch CD. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology* **48**: 443–453.

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

Smith T and Waterman M. 1981. Identification of common molecular subsequences. *Journal of Molecular Biology* **147**: 195–197.