The theoretical analysis of sequencing bioinformatic algorithms

. . . or, what I did during my sabbatical. . .

Paul Medvedev

Pennsylvania State University

PANGAIA Seminar
November 16, 2020

These slides and associated survey paper draft are available via this link.

Theoretical analysis of algorithm (TAA) performance

Performance: runtime, memory use, or accuracy

Examples

- Insertion sort runs in worst-case $\Theta(n^2)$ time
- Expected lookup time in a hash table is $\mathcal{O}(1)$

Goals [Rou19]

- *Predict* the empirical performance of an alg
- *Design* algorithms that perform well in practice (i.e. be a yardstick)

Techniques for the theoretical analysis of algorithms (TAA) [Rou19]

- Traditional worst-case analysis
  - Constants: dropped
  - Inputs: worst-case
  - Parameter: input size
- Parametrized worst-case analysis
- AofA [SF13]
- Average-case analysis
- Semi-random models

The central dogma of computer science

and its relationship to sequencing bioinformatics

Theoretical analysis of algorithms (TAA) achieves its two goals

- Leads to design of new algorithms that perform well in practice
- Predicts the empirical performance of algorithms

Is it true for algorithms for sequencing data?

- TAA led to the development of ILPs, HMMs, machine learning, etc...
- These are applied in but not developed specifically for bioinformatics
- Has TAA achieved its goals when applied to problems whose main application is sequencing bioinformatics?

Plan for talk

Motivating questions for sequencing bioinformatics

- What TAA techniques have been applied?
- To what extent has TAA achieved its goals?
- What can be done to improve the applicability of TAA?

A first attempt to systematically and critically study the application of TAA to sequencing bioinformatics.

Outline

- Comprehensive look at one problem
  - Edit distance
- Highlights
  - Compact data structures
  - Accuracy of genome assembly algorithms
  - Accuracy of structural variation detection algorithms

Plan for talk

Motivating questions for sequencing bioinformatics

- What TAA techniques have been applied?
- To what extent has TAA achieved its goals?
- What can be done to improve the applicability of TAA?

A first attempt to systematically and critically study the application of TAA to sequencing bioinformatics.

Outline

- Comprehensive look at one problem
  - Edit distance
- Highlights
  - Compact data structures
  - Accuracy of genome assembly algorithms
  - Accuracy of structural variation detection algorithms

Edit distance

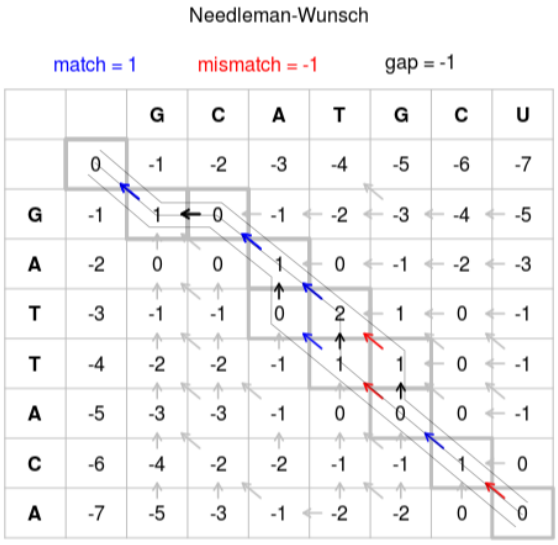
Given two strings A and B of equal length n over a constant sized alphabet, what is the minimum number of substitutions, insertions and deletions needed to transform A into B.

- a.k.a Levenshtein distance or unit-cost edit distance
- Extensions not considered here

In practice

Algorithms

- Needleman-Wunsch
- Banded alignment
- Myers bitparallel technique: optimization to use bitvector operations



Software

- Edlib and SeqAn: banded alignment + Myers
- Parasail: Needleman Wunsch + parallelism
- BGSA: Needleman Wunsch + Myers + parallelism
- Specialized hardware approaches: GPUs, FPGAs, custom processors

Edlib, 100,000nt sequences, single-core

- 1s for divergence of 1%
- 30s for divergence of 40%

Traditional worst-case analysis

Design of alg that works well in practice? Yes

- Needleman-Wunsch alg, by Wagner and Fischer [WF74]
  - Runtime: worst-case $\Theta(n^2)$
  - Can probably credit traditional worst-case analysis with design
  - Basis of other algorithms that work well in practice.

Predict empirical performance of alg? Yes

- Needleman-Wunsch: runtime analysis is actually every-case.

Traditional worst-case analysis led us astray.

- Four-Russians algorithm
  - Runtime: worst-case $\Theta(\frac{n^2}{\log^2 n})$
  - Factor of 5 speedup for $n = 2^{18}$ [Rej]
  - would not exceed 10 even for billion chars
  - not used in practice

Barrier results

- Under strong exponential time hypothesis, there cannot be a strongly sub-quadratic algorithm [BI18]
- We are unlikely to design substantially better algorithms using traditional worst-case analysis as a guide....but we can if you use a different analysis technique

Worst-case analysis parametrized by edit distance (k)

Banded alignment [Ukk85]

- Run time is $\Theta(nk)$ in every case
- No significant implementation constants or data structures

Design of alg that works well in practice? Yes

- Used in edlib, seqan

Predict empirical performance of alg? Yes

- Separates banded alignment from Needleman Wunsch when $k = o(n)$
- Even when $k = dn$, for some constant d, the analysis captures the constant speedup over Needleman Wunsch

For $k = \Theta(n)$, barrier results apply

- Unfortunately this an important case in practice

Worst-case analysis parametrized by entropy and compressibility

Parameter: Entropy of the input h ($0 \leq h \leq 1$)

- Run time: $\mathcal{O}(\frac{hn^2}{\log n})$ for most inputs [CLZU03]

Parameter: Size of compressed input ℓ

- Run time: $\mathcal{O}(\ell n)$ for run-length encoded strings [Mäk03, CLZU03, ALM02]
- Run time: $\mathcal{O}(\ell n \sqrt{\log(n/\ell)})$ for slp-compressed strings [Gaw12]

Achieves design goal? No

- h and ℓ are high in practice

Average-case analysis

a.k.a distributional analysis or random model

Input: Follow a distribution

Performance measured as a function of distribution

- e.g. expected value
- e.g. value "with high probability"

What can we hope to achieve?

- Performance goal? No
- Design goal? Yes, e.g. strongly sub quadratic alg in the model

Uniformly random model

- DNA is not uniformly random
- A and B are related

Indel channel model [GS20]

- A uniformly random
- B mutated from A with subs, dels, and ins at constant probability
- $\mathcal{O}(n \log n)$ algorithm w.h.p.
- Needs to be implemented
- Is the algorithm's runtime and accuracy robust on real data?

Semi-random models

Description

Input: Combination of

- Adversary
- Randomness following a distribution

Performance measure

- Worst-case over choices of adversary
- Average-case over random distribution

Need middle ground between adversary and randomness

- A needs to make sense biologically
- A and B must be dependent on each other

E.g. smoothed analysis [AK12]

- Adversary chooses A and B and a longest common subsequence of them
- Each nucleotide not involved in LCS is perturbed independently
- Each nucleotide involved in LCS is perturbed identically in A and B

Semi-random models

[BSS20]

Model

- Adversary chooses a seed string s
- s is randomly permuted to get A
- Adversary (knowing A and s) constructs B

Algorithm

- Approximation ratio: $1 + o(1)$
- Expected runtime: $\Theta(n^{1.898})$
- $<$ 9x speedup over $\Theta(n^2)$ for $n = 10^9$

Average-case and semi-random model summary

- Promising active direction
- The design goal not achieved so far

Analyzing with advice

[GH17]

Idea: Perhaps the formulation of the computational problem is not general enough.

Suppose that for an input A and B, the alg has access to $\log n$ "correlated" instances.

Alg can find edit distance in $\mathcal{O}(n \log n)$ time with high probability.

What are "correlated instances?"

- Same positional edit sequence
- Not clear where such data might be available

Plan for talk

Motivating questions for sequencing bioinformatics

- What TAA techniques have been applied?
- To what extent has TAA achieved its goals?
- What can be done to improve the applicability of TAA?

A first attempt to systematically and critically study the application of TAA to sequencing bioinformatics.

Outline

- Comprehensive look at one problem
  - Edit distance
- Highlights
  - Compact data structures
  - Accuracy of genome assembly algorithms
  - Accuracy of structural variation detection algorithms

Memory use of compact data structures

A success story

Compact/succinct data structures

- Allow efficient queries using as little space as possible (e.g. BWT)
- Higher-order constants retained during analysis

Successfully developed for sequencing bioinformatics

- BOSS data structure for de Bruijn graph [BOSS12]
  - $4m + o(m)$ bits, where m is the number of edges
  - used in MEGAHIT [LLL+16] assembler and basis for VG toolkit [Sir17, GSN+18]
- Pufferfish k-mer index [ASSP18], used in Salmon [PDL+17]
- Counting quotient filter, used by for k-mer counting [PBJP17]
- Minia, a de Bruijn graph representation in an assembler [CR12]
- Rainbowfish, a representation of colored de Bruijn graph [APP17]
- BBhash, minimum perfect hash function for k-mers [LRCP17]

Summary

- Design of alg that works well in practice? Yes
- Predict empirical performance of alg? Yes

Accuracy of genome assembly algorithms

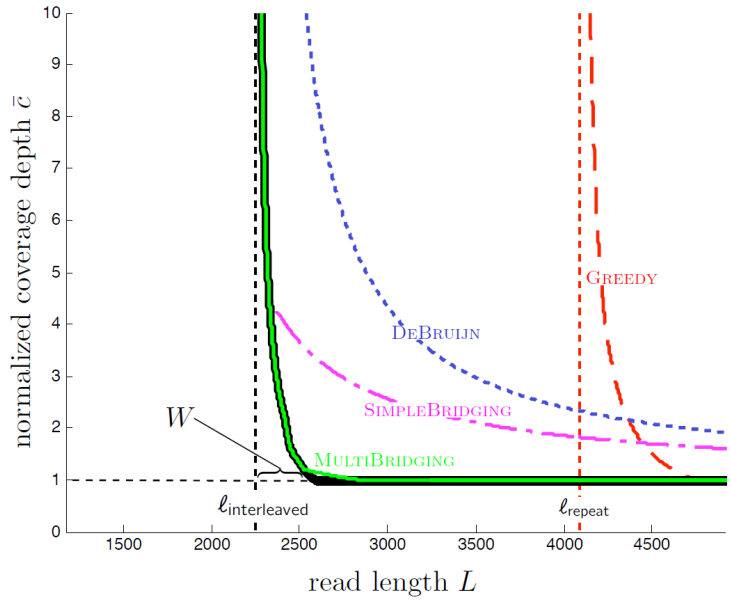
Theoretical analysis attempts

Measure what percentage of all "inferable" contigs are reported [TM17]

- Technically challenging

Measure the conditions under which an algorithm can fully reconstruct the genome [BBT13]

- Shannon assembler [KHM+16]



Accuracy of genome assembly algorithms

Reality

In practice

- Most alg design is driven by accuracy on empirical data (e.g. N50)
- Most algs use complex heuristics that are difficult to analyze
- Plenty use of theory (e.g. combinatorial algorithms)...but not to analyze accuracy

Summary

- Design? Mostly no..
- Performance? No

Assemblathon2 [BFA+13] conclusions for short read assemblers

- Rank of assemblers using experimental evaluation depended on dataset and evaluation metrics/parameters.
- Reviewer: "on any reasonably challenging genome, and with a reasonable amount of sequencing, assemblers neither perform all that well nor do they perform consistently" [TB]

Accuracy of structural variation detection algorithms

Current tools

- Many based on combinatorial algorithms
- Only a few have any theoretical measurement of accuracy
  - e.g. CLEVER [MCC+12] calculates the false discovery rate

What would it take to measure accuracy?

- A model for the distribution of variant type, location, and frequency.
- Our understanding of the biological generative model is incomplete.

Accuracy in practice [KML+19, CDSP19]

- Accuracy on simulated much higher than on real data
- Tools suffer from low recall
- Ranking of tools according to accuracy varied across variant subtypes

Summary

- Design? Mostly no..
- Performance? No
- Open challenge: Come up with generative model for structural variants such that algs designed to maximize expected accuracy on this model outperform existing tools in practice, with an observed accuracy matching the predicted one

Conclusions

What have we learned?

Has TAA been successful in sequencing bioinformatics? Anecdotally:

- Most papers do not perform TAA
- Or perform it but say alg performs much better in practice
- Its impact has been limited in best

Challenges of TAA in sequencing bioinformatics

- Traditional worst-case analysis too pessimistic
- Technology evolves fast and there is a lot of time pressure to {develop, analyze, apply} algorithm
- Researcher background

Is TAA even necessary?

- Empirical evaluation is powerful and essential — but not enough
- Known problems:
  - Benchmarks and critical assessments are not always available
  - Incentives favor overfitting the data
  - Absolute/relative performance does not always generalize

Conclusions

What can be done?

Recognize the Theoretical Analysis of Sequencing Bioinformatics Algorithms (TASBA) as its own research area.

- . . . as opposed to a side note in an algorithm paper

General goals of TASBA research program:

- Develop TAA techniques that are simple yet predictive of empirical performance
- Establish best practices for the application of TAA techniques
- Establish the most effective way to combine empirical and theoretical analysis techniques.

Immediate goals

- Find old success stories
- Look at bioinformatics algorithms more generally
- Develop new success stories. Retrospective is OK.

References I


Alexandr Andoni and Robert Krauthgamer, *The smoothed complexity of edit distance*, ACM Transactions on Algorithms (TALG) 8 (2012), no. 4, 1–25.


Ora Arbell, Gad M Landau, and Joseph SB Mitchell, *Edit distance of run-length encoded strings*, Information Processing Letters 83 (2002), no. 6, 307–314.


Fatemeh Almodaresi, Prashant Pandey, and Rob Patro, *Rainbowfish: A succinct colored de Bruijn graph representation*, WABI 2017: Algorithms in Bioinformatics (Russell Schwartz and Knut Reinert, eds.), LIPIcs-Leibniz International Proceedings in Informatics, vol. 88, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017, pp. 18:1–18:15.


Fatemeh Almodaresi, Hirak Sarkar, Avi Srivastava, and Rob Patro, *A space and time-efficient index for the compacted colored de Bruijn graph*, Bioinformatics 34 (2018), no. 13, i169–i177.


Guy Bresler, Ma'ayan Bresler, and David Tse, *Optimal assembly for high throughput shotgun sequencing*, BMC Bioinformatics 14 (2013), no. S5.


Keith R Bradnam, Joseph N Fass, Anton Alexandrov, Paul Baranay, Michael Bechner, Inanç Birol, Sébastien Boisvert, Jarrod A Chapman, Guillaume Chapuis, Rayan Chikhi, et al., *Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species*, GigaScience 2 (2013), no. 1.


Arturs Backurs and Piotr Indyk, *Edit distance cannot be computed in strongly subquadratic time (unless SETH is false)*, SIAM Journal on Computing 47 (2018), no. 3, 1087–1097.


Alexander Bowe, Taku Onodera, Kunihiko Sadakane, and Tetsuo Shibuya, *Succinct de Bruijn graphs*, WABI, Lecture Notes in Computer Science, vol. 7534, Springer, 2012, pp. 225–235.


Mahdi Boroujeni, Masoud Seddighin, and Saeed Seddighin, *Improved algorithms for edit distance and lcs: Beyond worst case*, Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2020, pp. 1601–1620.

References II


Daniel L Cameron, Leon Di Stefano, and Anthony T Papenfuss, *Comprehensive evaluation and characterisation of short read general-purpose structural variant calling software*, Nat. Commun. 10 (2019), no. 1, 1–11 (en).


Maxime Crochemore, Gad M Landau, and Michal Ziv-Ukelson, *A subquadratic sequence alignment algorithm for unrestricted scoring matrices*, SIAM journal on computing 32 (2003), no. 6, 1654–1673.


Rayan Chikhi and Guillaume Rizk, *Space-efficient and exact de Bruijn graph representation based on a Bloom filter*, WABI 2012: Algorithms in Bioinformatics (Ben Raphael and Jijun Tang, eds.), Lecture Notes in Computer Science, vol. 7534, Springer, 2012, pp. 236–248.


Paweł Gawrychowski, *Faster algorithm for computing the edit distance between slp-compressed strings*, International Symposium on String Processing and Information Retrieval, Springer, 2012, pp. 229–236.


Shafi Goldwasser and Dhiraj Holden, *The complexity of problems in p given correlated instances*, 8th Innovations in Theoretical Computer Science Conference (ITCS 2017), Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.


Arun Ganesh and Aaron Sy, *Near-Linear Time Edit Distance for Indel Channels*, 20th International Workshop on Algorithms in Bioinformatics (WABI 2020), Leibniz International Proceedings in Informatics (LIPIcs), vol. 172, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, pp. 17:1–17:18.


Erik Garrison, Jouni Sirén, Adam M Novak, Glenn Hickey, Jordan M Eizenga, Eric T Dawson, William Jones, Shilpa Garg, Charles Markello, Michael F Lin, et al., *Variation graph toolkit improves read mapping by representing genetic variation in the reference*, Nature biotechnology 36 (2018), no. 9, 875–879.


Sreeram Kannan, Joseph Hui, Kayvon Mazooji, Lior Pachter, and David Tse, *Shannon: An information-optimal de novo RNA-Seq assembler*, bioRxiv (2016), 039230.

References III


Shunichi Kosugi, Yukihide Momozawa, Xiaoxi Liu, Chikashi Terao, Michiaki Kubo, and Yoichiro Kamatani, *Comprehensive evaluation of structural variation detection algorithms for whole genome sequencing*, Genome Biol. **20** (2019), no. 1, 1–18 (en).


Dinghua Li, Ruibang Luo, Chi-Man Liu, Chi-Ming Leung, Hing-Fung Ting, Kunihiko Sadakane, Hiroshi Yamashita, and Tak-Wah Lam, *Megahit v1. 0: a fast and scalable metagenome assembler driven by advanced methodologies and community practices*, Methods **102** (2016), 3–11.


Antoine Limasset, Guillaume Rizk, Rayan Chikhi, and Pierre Peterlongo, *Fast and scalable minimal perfect hashing for massive key sets*, arXiv preprint arXiv:1702.03154 (2017).


Veli Mäkinen, *Approximate matching of run-length compressed strings*, Algorithmica **35** (2003), no. 4, 347–369.


Tobias Marschall, Ivan G Costa, Stefan Canzar, Markus Bauer, Gunnar W Klau, Alexander Schliep, and Alexander Schönhuth, *CLEVER: clique-enumerating variant finder*, Bioinformatics **28** (2012), no. 22, 2875–2882.


Prashant Pandey, Michael A Bender, Rob Johnson, and Rob Patro, *A general-purpose counting filter: making every bit count*, SIGMOD '17: Proceedings of the 2017 ACM International Conference on Management of Data, Association for computing machinery, 2017, pp. 775–787.


Rob Patro, Geet Duggal, Michael I Love, Rafael A Irizarry, and Carl Kingsford, *Salmon provides fast and bias-aware quantification of transcript expression*, Nature methods **14** (2017), no. 4, 417–419.


Martin Rejmon, *Multi-threaded implementation of Four Russians edit distance algorithm*, Bachelor's thesis, Czech Technical University in Prague, Faculty of Information Technology.


Tim Roughgarden, *Beyond worst-case analysis*, Communications of the ACM **62** (2019), no. 3, 88–96.

References IV


Robert Sedgewick and Philippe Flajolet, *An introduction to the analysis of algorithms*, Pearson Education India, 2013.


Jouni Sirén, *Indexing variation graphs*, 2017 Proceedings of the ninteenth workshop on algorithm engineering and experiments (ALENEX), SIAM, 2017, pp. 13–27.


C Titus Brown, *Thoughts on the assemblathon 2 paper*, http://ivory.idyll.org/blog/thoughts-on-assemblathon-2.html, Accessed: 2020-1-9.


Alexandru I Tomescu and Paul Medvedev, *Safe and complete contig assembly through omnitigs*, Journal of Computational Biology 24 (2017), no. 6, 590–602.


Esko Ukkonen, *Algorithms for approximate string matching*, Information and Control 64 (1985), no. 1-3, 100–118.


Robert A Wagner and Michael J Fischer, *The string-to-string correction problem*, Journal of the ACM (JACM) 21 (1974), no. 1, 168–173.