The limitations of the theoretical analysis of applied algorithms

PAUL MEDVEDEV, The Pennsylvania State University, USA

The theoretical analysis of performance has been an important tool in the engineering of algorithms in many application domains. Its goals are to predict the empirical performance of an algorithm and to be a yardstick that drives the design of novel algorithms that perform well in practice. While these goals have been achieved in many instances, they have not been achieved in some other crucial application domains. In this paper, I focus on the example of sequencing bioinformatics, an inter-disciplinary field that uses algorithms to extract biological meaning from genome sequencing data. I will demonstrate two concrete examples of how theoretical analysis has failed to achieve its goals but also give one encouraging example of success. I will then catalog some of the challenges of applying theoretical analysis to sequencing bioinformatics, argue why empirical analysis is not enough, and give a vision for improving the relevance of theoretical analysis to sequencing bioinformatics and other application domains. By recognizing the problem, understanding its roots, and providing potential solutions, this work can hopefully be a crucial first step towards making theoretical analysis more relevant in modern application domains.

1 INTRODUCTION

When I ask first year computer science undergraduate students how to quantify the speed of an algorithm, they look at me puzzled and tell me to just run the algorithm and see how long it takes. Such empirical analysis, in fact, is the most direct and natural way to measure algorithm performance. However, it has long been understood to have many shortcomings [40] and, to overcome these shortcomings, computer scientists developed ways to theoretically analyze algorithm performance. The most common technique for this is \textit{traditional worst-case analysis}. For example, we say that merge sort runs in $O(n \log n)$ worst-case time, which formally means that there exists a constant $c$ such that for any large-enough input of $n$ elements, merge sort takes at most $cn \log n$ time. Other more sophisticated techniques, such as parametrized analysis, average-case analysis, or semi-random models, better capture the properties of real data [39]. Additionally, theoretical analysis can be used to measure not just speed but other aspects of algorithm performance like memory usage or accuracy. When undergraduate students take an Algorithms course, they finally learn about the theoretical analysis of algorithms and how to use it to capture general patterns of performance that empirical analysis does not.

The most direct impact of such theoretical analysis is in applied algorithms, i.e. algorithms which are implemented and applied to real data (in contrast to complexity theory, where such analysis serves the purpose of understanding the hierarchy of problem, rather algorithm, complexity). The goals of the theoretical analysis of applied algorithms, which we will denote by $TA^3$, are twofold [39]. One goal is to \textit{predict} the empirical performance of an algorithm, either in an absolute sense or relative to others. The second goal is to be a yardstick that drives the \textit{design} of novel algorithms that perform well in practice. $TA^3$ has achieved its goals with resounding success, being directly responsible for the design and performance prediction of many algorithms used in practice (e.g. Dijkstra’s shortest path algorithm). Because of this success, Algorithms instructors typically jump into theoretical analysis with only a cursory justification of why it is needed. To put it bluntly, the fact that $TA^3$ achieves the two goals has become a dogma of computer science.

Without denying the \textit{overall} success of $TA^3$, I posit that it has failed to achieve the stated goals in \textit{some crucial} application domains. I use the example of \textit{sequencing bioinformatics} (which we will call SeqBio for short), an inter-disciplinary field that uses algorithms to extract biological meaning from sequencing data. Sequencing bioinformatics has revolutionized the life sciences, with algorithms developed by computer scientists (e.g. [3, 21]) enabling projects...
such as the Earth Microbiome Project [14], the Vertebrate Genomes Project [37], and the Cancer Genome Atlas [17]. It is also in the process of revolutionized healthcare, enabling projects such as Obama’s Precision Medicine Initiative.

However, the vast majority of SeqBio papers either do not perform TA3 (e.g. [3]) or perform traditional worst-case analysis only to conclude that it is not a good predictor of the algorithm’s performance in practice (e.g. [27]). In this paper, I will demonstrate two concrete examples of TA3’s failure in SeqBio but also give one encouraging example of success. I will then catalog some of the challenges of applying TA3 to SeqBio, argue why empirical analysis is not enough, and give a vision for improving the relevance of TA3 to SeqBio and other application domains. By recognizing the problem, understanding its roots, and providing potential solutions, this work can hopefully be a crucial first step towards making TA3 more relevant in modern application domains.

2 ACCURACY OF GENOME ASSEMBLERS

Sequencing Bioinformatics algorithms work with data generated by various instruments that, roughly speaking, repeatedly sample short substrings (called reads) from a long genome sequence [26]. The locations are chosen semi-uniformly at random but are not output by the instrument; the output only contains the string sequence of each read. Moreover, the reads may contain errors and hence not be exact substrings of the genome. Such a sequencing experiment can generate billions of reads at ever-decreasing costs. There are many applications of sequencing technology but in this section we will focus on the genome assembly problem, a classical SeqBio problem.

The genome assembly computational problem is to take a sequencing experiment from a single genome and to reconstruct the full DNA sequence of that genome [42]. There are dozens of widely-used assemblers, with hundreds more at the prototype stage. Genome assembly algorithms have enabled genome-wide studies of thousands of species and have a biological impact that is hard to overstate. The most important aspect of an assembler’s performance is, arguably, its accuracy, since the resources required to collect a DNA sample usually outweigh the computational resources of an assembler. In this section, we will describe the various attempts to apply theoretical analysis to 1) predict the accuracy of assemblers and 2) design more accurate assemblers.

One of the earliest theoretical measures of accuracy is the likelihood of the reads given an assembly. The idea of building an assembler to maximize this likelihood was originally proposed in [30] and later pursued in [5, 15, 28, 46]. Much of the work centers on finding an appropriate likelihood function, i.e. one which models the intricacies of the sequencing process. Unfortunately, these formulations have not directly led to any state-of-the-art assemblers, leaving the design goal of TA3 unfulfilled. The reasons for this are not fully clear from the literature, but we will speculate in Section 5.

I am not aware of any work that attempted to use likelihood to theoretically predict algorithm performance. Some works did explore the idea of using a likelihood score to evaluate the accuracy of an assembly [12, 13, 16, 22, 35, 47]. Though these tools have been widely used, they are not designed to give a theoretical accuracy of an assembler but, rather, to be run on a concrete output. As such, they cannot predict in advance how an assembler will empirically perform, leaving the prediction goal of TA3 unfulfilled.

A later approach [7] is to evaluate an assembler by the conditions under which it can fully reconstruct the original genome sequence. Such conditions could, for example, be the number of reads needed or the highest error that could be tolerated. This accuracy framework did lead to the design of a new assembler called Shannon that has been used in practice [18], thus partially fulfilling the design goal. However, these conditions rarely arise in practice (i.e. the genome usually has too many repetitive sequences to be reconstructed completely and unambiguously). Therefore it is not clear if designing algorithms to optimize this would lead to other assemblers that perform well in practice. In terms
of the prediction goal, this framework was also applied to theoretically predict the accuracy of some simple assembly strategies [7]; however, it has not been applied to predict the accuracy of any other assemblers used in practice. The common challenge to all prediction attempts such as this one is that most assembly algorithms rely heavily on ad-hoc, hard-to-analyze heuristics.

One more recent approach is to measure what percentage of all the substrings that could possibly be inferred to exist in the genome are output by the assembler [45]. However, this framework has proven technically challenging to apply to real data, e.g. to account for sequencing errors and gaps in coverage. It has not yet led to the design of a new assembler or to the accuracy prediction of assemblers, though work is ongoing [8].

In summary, TA3 cannot be credited with the design of any of the widely used assemblers, nor has it led to any theoretical analysis that can predict the performance of an assembler on real data. In practice, assemblers are designed heuristically to perform well on a set of empirically observed metrics, such as the lengths of the segments that the assembler reports or the recovery of conserved genomic elements [38]. Additional validation is performed by measuring the agreement with data from an orthogonal sequencing technology [36]. Assembly algorithms are usually developed to perform well on publicly available datasets and often validated by the exact same datasets.

The shortcomings of TA3 have been felt in practice. The assemblathon2 competition [6] performed an empirical evaluation of assemblers and found that the ranking of different assemblers according to their relative accuracy depended on the dataset, on the evaluation metrics being used, and on the parameter choices made in the evaluation scripts. These are exactly the shortcomings of empirical evaluation that TA3 is intended to address. Moreover, the assemblers themselves are simply not as good as they could be, or, as one of the reviewers concluded, "on any reasonably challenging genome, and with a reasonable amount of sequencing, assemblers neither perform all that well nor do they perform consistently" [44]. In the last five years, the practical situation has to some extent improved due to newer sequencing technologies that generate higher quality data. Nevertheless, the experience of the assemblathon2 competition is instructive to appreciate the limitations of solely-empirical analysis in SeqBio.

### 3 ACCURACY OF STRUCTURAL VARIATION DETECTION ALGORITHMS

Once a genome is assembled for a species, it forms what is called a reference genome. Follow up studies then sequence different individuals of the same species but do not perform a de novo genome assembly. Instead, they catalog the variations between the sequenced genome and the reference, under the assumption that the genome is unchanged in places where there is no alternative evidence. Variants that affect large regions of more than 500 nucleotides, called structural variants, are responsible for much genomic diversity and are linked to numerous human diseases, including cancer and a myriad of neurodevelopmental disorders [48]. Algorithms for detecting structural variation started to appear around 2008 (see [24, 29] for surveys) and a recent assessment identified at least 69 usable tools [20]. As with genome assembly, the most important aspect of algorithm performance is, arguably, accuracy.

The vast majority of tools are heuristics with no theoretical analysis of their accuracy. There are some exceptions, when a probabilistic formulation is used to achieve a desired false discovery rate (e.g. [27]). In such cases, TA3 can take some credit for the design of the algorithm. However, accuracy greatly depends on the type of variant (e.g. deletions are easier to detect than duplications) and on the location of the variant (e.g. repetitive sequences make variants harder to detect). Thus, a useful analysis of accuracy requires a statistical model for the distribution of variant type, location, and relative frequency. But coming up with realistic models is challenging, as our understanding of the biological process that generates structural variants is limited. Therefore, even when accuracy is predicted theoretically, it does
not correspond to what is observed in practice because the models are too idealized [27]. Thus, even in the limited cases where TA3 has been applied, it has not achieved its prediction goals.

As with genome assembly, the algorithms used in practice suffer from many of the limitations that TA3 is intended to address. Algorithms are typically evaluated empirically, using simulated data or an established benchmark. Two recent studies assessing the empirical accuracy of algorithms [9, 20] found that the tools suffered from low recall and the ranking of the tools according to accuracy varied greatly across different subtypes of variants. Empirical evaluation is hampered by the same lack of models that hampers theoretical evaluation and [9] warned developers against considering “simulation results representative of real-world performance.” In fact, accuracy on simulated data is typically much higher for most tools than on real data [20]. As with genome assembly, the problems described in [9] are the types inherent to empirical-only evaluation: “But with newly published callers [algorithms] invariably reporting favourable performance, it is difficult to discern whether the results of these studies are representative of robust improvements or due to the choice of validation data, the other callers selected for comparison, or over-optimisation to specific benchmarks.”

4 MEMORY USAGE OF k-MER DATA STRUCTURES

Sequencing data is often reduced to a collection of k-long strings (called k-mers) that are stored in a variety of data structures. The exact data structure depends on the types of queries that need to be supported, the type of associated data that is maintained, and the type of sequencing experiment. Such k-mer data structures have become ubiquitous in SeqBio and form the backbone of hundreds of tools (for surveys, see [10, 25]). The theoretical analysis of their memory is a rare bright light in the theoretical analysis of SeqBio algorithms and I discuss it here to illustrate TA3’s potential for success in SeqBio.

Many of the techniques used to analyse the memory used by k-mer data structures have been borrowed from the field of compact data structures [31]. The analysis of compact data structure memory differs from traditional worst-case analysis in that the higher order terms are often written without asymptotic notation (e.g. $4n + o(n)$ instead of $O(n)$). This helps distinguish algorithms whose memory differs by a constant factor, at the expense of a more technically involved analysis.

Using this type of analysis as a yardstick has led to the design of several k-mer data structures that perform well on real data and are included in broadly used software [1, 2, 4, 11, 32, 43]. In one example, such an analysis led to the design of a compact representation of a popular k-mer data structure called the de Bruijn graph that uses $4n + o(n)$ bits, where $n$ is the number of edges [4]. This data structure uses very little memory in practice [34] and forms the core of the widely used MEGAHIT assembler [23]. Another successful example of a k-mer data structure is the pufferfish index [2], which forms part of the popular Salmon [33] software to quantify levels of gene expression in a sequencing sample. In addition to satisfying the design goal, this type of analysis has been used to correctly predict how much memory k-mer data structures will use in practice and compare the relative performance of different methods (e.g. quantifying the tradeoffs between memory and query time of various indices).

Thus, theoretically analyzing memory by retaining the constant for the higher order terms has satisfied both the design and prediction goals. Such a TA3 success, though rare in SeqBio, demonstrates that it is indeed possible for TA3 to have an impact in SeqBio.
5 THE CHALLENGES OF THEORETICAL ANALYSIS OF SEQUENCING BIOINFORMATIC ALGORITHMS

Before proceeding, it is useful to make the distinction between direct and indirect influence of TA3 in SeqBio. Without question, TA3 can be credited with the development and analysis of methods which have become part of SeqBio’s toolbox, e.g. integer linear programming, hidden Markov models, clustering algorithms, suffix trees/arrays, the Burrows-Wheeler transform, Bloom filters, minhash sketches, machine learning, etc. Combinatorial algorithms for optimization problems with theoretical guarantees form the backbone of many assemblers [3, 41] and structural variant detection tools [29]. However, in this paper we are concerned with direct influence, i.e. situations where theoretical analysis was applied to problems that are specific to SeqBio, or at least for which SeqBio was a major motivation.

What is it about SeqBio that makes this direct application of TA3 so challenging? I can identify at least four reasons. First, traditional worst-case analysis is in most cases too pessimistic when it comes to real biological data and fails to separate high performing algorithms, which take advantage of the structure of real data, from poorly performing ones that do not. This is in fact what many successful bioinformatic algorithms do, as they stem from a deep understanding of the data followed by heuristics to exploit its structure. Such heuristics are difficult to analyze and are unlikely to be invented when the yardstick is traditional worst-case analysis.

Second, because SeqBio researchers require a broad inter-disciplinary skill set, they often lack the technical expertise necessary to apply more sophisticated TA3 techniques. These techniques, sometimes called beyond worst case analysis, do in fact capture some of the complexities of real data [39]. A good example of such a technique is smoothed analysis or, more generally, semi-random models, which make the analysis more realistic by assuming there is random noise forced upon any worst case instance. However, these advanced techniques are rarely taught as part of the core CS curricula, and SeqBio researchers are typically exposed to theory only through introductory courses. This affects the technical complexity of the TA3 that SeqBio researchers can perform. For example, it is a rare case that a SeqBio researcher can do a smoothed analysis of an algorithm. Being able to come up with a novel analysis technique is even more rare.

The third reason is that the throughput of sequencing technology is growing at a rate faster than Moore’s law [19]. The usual justification for ignoring constants in traditional worst-case analysis is that a constant factor improvement in time or memory utilization will quickly become obsolete, as computing capacity grows exponentially. But when the size of the data is growing faster than the computing capacity, a constant factor speedup may in fact be relevant for a long time. This helps explain the success of the theoretical analysis of $k$-mer data structures, where the constant is kept (Section 4).

The fourth reason is that SeqBio algorithms are usually developed, analyzed, and applied under significant time pressure. In many cases, the algorithmic problem that a SeqBio researcher is tasked with solving is only a small part of a more complex biological project. Because the sequencing technology is rapidly evolving and changing, time is of the essence. The researcher must work under time pressure to deliver a method that would analyze the data at hand and cannot afford to dedicate months of time to theoretically analyze an algorithm. There are some notable exceptions of SeqBio subareas where the data is stable enough so that the analysis and development of new algorithms has had enough time for complex TA3 techniques to emerge (e.g. the edit distance problem), but these cases are rare.

Because of these challenges, a TA3 technique that is useful has to not only be predictive of empirical performance but also be easy to apply, easy to explain, and easy to understand. Thus, the theoretical analysis of SeqBio algorithms not only favors but in fact requires simplicity of the analysis technique. A simpler technique will be more trusted by biologists, more broadly understood and applied by bioinformaticians, more easily taught to students, and more...
likely included in bioinformatics curricula. The challenge is to have a simple technique which nevertheless accurately captures empirical performance and is an effective yardstick for the development of empirically better algorithms.

6 EMPIRICAL ANALYSIS IS NOT ENOUGH

In spite of the challenges above, sequencing bioinformatics researchers continuously develop successful tools that have an enormous biological impact. A popular approach that fits both the accelerated development timeline and the appropriate technical level described in Section 5 is to reduce the problem to one from a toolbox of known approaches. A bioinformatician’s toolbox includes black-box solvers for problems like clustering or integer linear programs; it also includes techniques like greedy algorithms or dynamic programming. This toolbox then forms the basis of incrementally designed heuristic algorithms. These are rule based heuristics where the rules are incrementally improved by looking at where simpler heuristics fails on empirical data. This is in contrast to a design driven by a mathematical understanding of the abstract problem with respect to a theoretical yardstick.

Empirical evaluation has its shortcomings but can nevertheless be very beneficial when done right. In particular, benchmark datasets and/or community competitions have been very useful. The Genome In A Bottle consortium for example has released both a benchmark sequencing dataset and a benchmark validation dataset for the problem of structural variant detection. Similarly, competitive assessment contests, such as assemblathon2 [6], are designed to test submitted algorithms on strategically designed datasets. Such benchmarks and competitions aim to achieve the same design and prediction goals as TA3.

Nevertheless, we have seen in Sections 2 and 3 how SeqBio suffers from serious problems that are hard to resolve with empirical analysis. In many cases, benchmarks and competitions have not been developed, or have been developed years later than many of the algorithms (e.g. in structural variation). In these cases, there is a proliferation of algorithms that perform well in the empirical validation of the authors but not in an independent evaluation. Note that the intent of the authors is not usually malicious; it is simply that the empirical validation approaches at their disposal are limited due to the lack of benchmarks. Even after the appearance of good benchmarks and competitions, algorithms that do well on these do not necessarily generalize well to other datasets; in fact, the incentives sometimes favor algorithms that overfit the data. Some assemblers, for example, are known to perform better on Human or E.coli, which are exactly the benchmarks commonly used for design and evaluation. On the other hand, algorithms that are designed to do well on a good (i.e. matching empirical observation) theoretical yardstick have the potential to be more generalizable, and the theoretical yardstick has the potential to predict algorithm performance on different datasets in a way that an empirical benchmark cannot. Thus, TA3 can overcome the problems associated with relying solely on empirical analysis.

7 A VISION FOR THE FUTURE

The first step to tackling the challenges of TA3 in sequencing bioinformatics is to recognize the Theoretical Analysis of Applied Algorithms as its own research area, distinct from the design of the algorithms themselves. In SeqBio journals or conferences, it is currently seen as a side note of algorithm development. Even in more theoretical SeqBio venues, the value of a TA3 contribution is not always appreciated. While a breakthrough result will likely be appreciated, a paper describing incremental progress on an algorithm is much more likely to be seen favorably than a paper describing substantial progress on an analysis technique. Moreover, there is generally an expectation that a SeqBio paper, even a theoretical one, delivers a novel algorithm. In most cases, this is a valid expectation, but it is not always appropriate for TA3 papers. Such publication challenges limit the formulation of TA3 subproblems and are in general detrimental to progress in the TA3 field.
Though we have focused on SeqBio, similar challenges of applying TA3 are present in other fast-paced, big data application domains. The fragmentation of TA3 research across multiple domains has thus been a bottleneck to progress. Recognizing TA3 as a research area will help with the formation of a community of like-minded researchers and all the synergies that go along with it.

The first step of a SeqBio TA3 research program could be to survey the literature for successful applications of TA3 techniques. This paper has surveyed the techniques in three sub-areas, but SeqBio is very broad and a more thorough investigation is likely to turn up some more successful applications. It will be useful to also identify successful TA3 techniques from areas of bioinformatics that are not based solely on sequencing data, such as whole-genome analysis and phylogeny reconstruction. The toolbox of successful TA3 techniques can then become the starting point of further research; furthermore, it can be added to the bioinformatics curriculum in computer science.

Viewing TA3 as its own research field would allow researchers to focus on retrospective prediction of algorithm performance; i.e. to develop TA3 techniques using algorithms where the empirical performance is already known. For example, when the theoretical computer science community tackled the limits of the competitive ratio analysis technique for the online paging problem, the empirically best algorithm was already known; the challenge was to find the right technique to reach the same conclusion [39]. Similarly, a distinct TA3 research program would not be afraid to tackle the question of performance prediction for short-read assembly, even though the empirically best methods have already been established.

The ultimate goal would be to develop TA3 techniques that are simple yet predictive of real-world performance. These techniques could then propel SeqBio forward and enable it to respond more quickly and accurately to the rapid evolution of sequencing technology. These techniques would ultimately generalize to other, similar, application domains. Without investment in TA3 research, SeqBio and other application domains will continue to be hampered by the limitations of solely-empirical analysis of performance.

Acknowledgments: This manuscript was inspired by the online videos for the 2014 class Beyond Worst-Case Analysis by Tim Roughgarden. I would like to thank Rob Patro, Jens Stoye, Alexandru Tomescu, and Fabio Vandin for reading an earlier version of this manuscript and providing great feedback, Michael Brudno for illuminating discussions on SVs, the twitter respondents at https://twitter.com/pashadag/status/1279232810407124992, and Jouni Siren for helping me understand the role of BOSS in VG. This material is based upon work supported by the National Science Foundation under Grants No. 1439057, 1453527, and 1931531.

REFERENCES

[1] Fatemeh Almodaresi, Prashant Pandey, and Rob Patro. Rainbowfish: A succinct colored de Bruijn graph representation. In Russell Schwartz and Knut Reinert, editors, WABI 2017: Algorithms in Bioinformatics, volume 88 of LIPIcs-Leibniz International Proceedings in Informatics, pages 18:1–18:15. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[2] Fatemeh Almodaresi, Hirak Sarkar, Avi Srivastava, and Rob Patro. A space and time-efficient index for the compacted colored de Bruijn graph. Bioinformatics, 34(13):i169–i177, 2018.

[3] Anton Bankevich, Sergey Nurk, Dmitry Antipov, Alexey A Gurevich, Mikhail Dvorkin, Alexander S Kulikov, Valery M Lesin, Sergey I Nikolenko, Son Pham, Andrey D Prijibelski, Alexey V Pyshkin, Alexander V Sirotkin, Nikolay Vyahhi, Glenn Tesler, Max A Alekseyev, and Pavel A Pevzner. SPAdes: A new genome assembly algorithm and its applications to Single-Cell sequencing. Journal of Computational Biology, 19(5):455–477, 2012.

[4] Alexander Bowe, Taku Onodera, Kunihiko Sadakane, and Tetsuo Shibuya. Succinct de Bruijn graphs. In WABI, volume 7534 of Lecture Notes in Computer Science, pages 225–235. Springer, 2012.

[5] Vladimír Boža, Broňa Brejová, and Tomáš Vinař. GamL: genome assembly by maximum likelihood. Algorithms for Molecular Biology, 10(1):1–10, 2015.

[6] Keith R Bradnam, Joseph N Fass, Anton Alexandrov, Paul Baranay, Michael Bechner, Inanç Biró, 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(1), 2013.
Rayan Chikhi, Jan Holub, and Paul Medvedev. Data structures to represent a set of k-long dna sequences. ACM Computing Surveys (CSUR), 54(1):1–22, 2021.

Rayan Chikhi and Guillaume Rizk. Space-efficient and exact de Bruijn graph representation based on a Bloom filter. In Ben Raphael and Jijun Bo Li, Nathanael Fillmore, Yongsheng Bai, Mike Collins, James A Thomson, Ron Stewart, and Colin N Dewey. Evaluation of de novo transcriptome assemblies from rna-seq data. Genome biology, 15(5):1–10, 2013.

Carolyn Hutter and Jean Claude Zenzlusan. The cancer genome atlas: creating lasting value beyond its data. Cell, 173(2):283–285, 2018.

Sreeam Kannan, Joseph Hui, Kayvon Mazooji, Lior Pachter, and David Tse. Shannon: An information-optimal de novo RNA-Seq assembler. Genome Biol., 20(1):1–18, June 2019.

Ben Langmead and Steven L Salzberg. Fast gapped-read alignment with bowtie 2.

Kenneth Katz, Oleg Shutov, Richard Lapoint, Michael Kimelman, J Rodney Brister, and Christopher O’Sullivan. The sequence read archive: a decade of explosive growth. Nucleic acids research, 30(1):D398–D402, 2002.

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(1):1–18, June 2019.

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

Bo Li, Nathanael Fillmore, Yongsheng Bai, Mike Collins, James A Thomson, Ron Stewart, and Colin N Dewey. Evaluation of de novo transcriptome assemblies from rna-seq data. Genome biology, 15(12):1–21, 2014.

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

Medhat Mahmoud, Nastassia Gobet, Diana Ivette Cruz-Dávalos, Ninon Mounier, Christophe Dessimoz, and Fritz J Sedlazeck. Structural variant calling: the long and the short of it. Genome Biol., 20(1):1–14, November 2019.

Camille Marchet, Christina Boucher, Simon J Puglisi, Paul Medvedev, Mikael Salson, and Rayan Chikhi. Data structures based on k-mers for querying large collections of sequencing data sets. Genome Research, 31(1):1–12, 2021.

Elaine R Mardis. DNA sequencing technologies: 2006–2016. Nature protocols, 12(2):213, 2017.

Tobias Marschall, Ivan G Costa, Stefan Canzar, Markus Bauer, Gunnar W Klau, Alexander Schliep, and Alexander Schönhuth. CLEVER: clique-enumerating variant finder. Bioinformatics, 28(22):2875–2882, 2012.

Paul Medvedev and Michael Brudno. Maximum likelihood genome assembly. Journal of computational Biology, 16(8):1101–1116, 2009.

Paul Medvedev, Monca Stanciu, and Michael Brudno. Computational methods for discovering structural variation with next-generation sequencing. Nature Methods, 6(11 Suppl):S13–20, November 2009.

Eugene W Myers. Toward simplifying and accurately formulating fragment assembly. Journal of Computational Biology, 2(2):275–290, 1995.

Gonzalo Navarro. Compact data structures: a practical approach. Cambridge University Press, 2016.

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

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(4):417–419, 2017.

Amatur Rahman and Paul Medvedev. Representation of k-mer sets using spectrum-preserving string sets. In Russell Schwartz, editor, Proceedings of Research in Computational Molecular Biology - 24th Annual International Conference, RECOMB 2020, volume 12074 of Lecture Notes in Computer Science, pages 152–168. Springer, 2020.

Atif Rahman and Lior Pachter. Cgal: computing genome assembly likelihoods. Genome biology, 14(1):1–10, 2013.

Arang Rhee, Ann Mc Cartney, Kishwar Shafin, Michael Alonge, Andrey Bizkadze, Giulio Formenti, Arkarachai Fungtammasan, Kerstin Howe, Chirag Jain, Sergey Koren, et al. Chasing perfection: validation and polishing strategies for telomere-to-telomere genome assemblies. bioRxiv, 2021.

Arang Rhee, Shane A McCarthy, Olivier Fedrigo, Joana Damas, Giulio Formenti, Sergey Koren, Marcela Uliano-Silva, William Chow, Arkarachai Fungtammasan, Juwan Kim, et al. Towards complete and error-free genome assemblies of all vertebrate species. Nature, 592(7856):737–746, 2021.
The limitations of the theoretical analysis of applied algorithms

[38] Arang Rhie, Brian P Walenz, Sergey Koren, and Adam M Phillippy. Merqury: reference-free quality, completeness, and phasing assessment for genome assemblies. Genome biology, 21(1):1–27, 2020.

[39] Tim Roughgarden. Beyond worst-case analysis. Communications of the ACM, 62(3):88–96, 2019.

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

[41] J T Simpson and R Durbin. Efficient construction of an assembly string graph using the FM-index. Bioinformatics, 26(12):i367–i373, 2010.

[42] Jared T Simpson and Mihai Pop. The theory and practice of genome sequence assembly. Annual review of genomics and human genetics, 16:153–172, 2015.

[43] Jouni Sirén. Indexing variation graphs. In 2017 Proceedings of the nineteenth workshop on algorithm engineering and experiments (ALENEX), pages 13–27. SIAM, 2017.

[44] C Titts Brown. Thoughts on the assemblathon 2 paper. http://ivory.idyll.org/blog/thoughts-on-assemblathon-2.html. Accessed: 2020-1-9.

[45] Alexandru I Tomescu and Paul Medvedev. Safe and complete contig assembly through omnitigs. Journal of Computational Biology, 24(6):590–602, 2017.

[46] Aditya Varma, Abhiram Ranade, and Srinivas Aluru. An improved maximum likelihood formulation for accurate genome assembly. In 2011 IEEE 1st International Conference on Computational Advances in Bio and Medical Sciences (ICCABS), pages 165–170. IEEE, 2011.

[47] Francesco Vezzi, Giuseppe Narzisi, and Bud Mishra. Rerevaluating assembly evaluations with feature response curves: Gage and assemblathons. PloS one, 7(12):e52210, 2012.

[48] Joachim Weischenfeldt, Orsolya Symmons, Francois Spitz, and Jan O Korbel. Phenotypic impact of genomic structural variation: insights from and for human disease. Nature Reviews Genetics, 14(2):125–138, 2013.