EN 601.447/647: Final Project Ideas

Fall 2023
Ben Langmead, TA: Jessica Bonnie, CAs: Molly Li & Sakshi Patil

1 Topic ideas

Here we try to provide *starting points* for developing your project idea. You will still need to **read papers** and **develop ideas** beyond what's suggested here. You can also look in your textbook or use Google, Google Scholar or the like. In some cases, we suggest how a project might be broken into pieces to be handled by different team members.

This is not an exhaustive list. We encourage you to suggest topics not on the list as well.

1. **Small data structures** – some of which can be called "sketches" or "probabilistic structures" – have been proposed for applications where we must represent and compare large genomic datasets. For example:

   - Bloom filter [1, 2]
   - CountMin sketch [3]
   - HeavyKeeper [4]
   - Bloom trees and tries [5, 6]
   - Bloom Comb-Trees [7]
   - Quotient filter [8] and counting quotient filter [9]
   - Cuckoo filter [10]
   - HyperLogLog [11]
   - SetSketch [12]
   - UltraLogLog [13]

   (Note that some of those advances are very recent!)

   Can you propose a novel application of one or more of these data structures? Can you find an interesting way to benchmark and compare them? Can you improve on one of them in some way? A few more examples: [14, 15, 16].

   A few recent review papers might also help: [17, 18, 19].

   Also, here are two YouTube videos where I discuss our recent Dashing and Dashing 2 tools that use sketching:

   - `https://youtu.be/1y9wqQUTvC4?si=LX743Tr_8X7dy7Y2`
   - `https://youtu.be/SixveY1ve-M?si=y3ySrqcgfbIXLW5L`

2. An relatively new idea in text indexing is the **Wheeler Graph** [20]. It generalizes the Burrows-Wheeler Transform to apply to some trees and graphs. It relates to automata as well, since it can be viewed as a way to make certain classes of NFA recognizers more efficient. Possible projects in this area (and different team members could tackle different pieces) would would be to read about and understand the concept of the Wheeler Graph then:

- Apply it to some problem that involves matching patterns against a graph- or tree-shaped text, e.g. many version of the same gene or genomes,
- Build a visualization that allows users to construct graphs, check whether they have the "Wheeler" property, trace the progress of a pattern matching procedure against the graph, etc
- Suggest and test algorithms for determining whether a given graph has the "Wheeler" property.

If you choose this project, we can supply more reading material beyond the reference above. I have posted several videos on Wheeler Graphs toward the end of this playlist: `https://bit.ly/yt_index`.

A current JHU PhD student Kuan-Hao Chao has done some work in this area and you might benefit from reading his paper [21] or viewing a talk of his on YouTube (`https://youtu.be/TkX9S024Dk8?si=S7HqmpUL-STs6zE7`; unfortunately, the audio quality is poor).

3. Sequencing datasets can be **contaminated with unwanted DNA**. In other words, some reads come from the individual/species that the scientist is *trying* to study, but others come from some *other* species the scientist *is not* trying to study. The contaminant could have slipped in when the sample was collected or when it was being processed in the laboratory, for example. This can confuse the analysis, making it hard to distinguish the biological phenomena of interest. Possible contaminants include:

   - Human, bacterial, and fungal DNA from people in the lab and creatures who live on or in them [22, 23]
   - DNA from other experiments in the lab, like mycoplasma [24] and recombinant vector [25]

   Spend time reading these references and get a clear picture of example how contamination arises and how it can be manifested in a sequencing dataset.

   Design a method that systematically scans for contaminants. Particularly useful would be a tool that could easily scan many public sequencing datasets. You might consider both online and offline methods, where online methods can only preprocess the query sequences (the contaminants; see: [26]) and offline methods are allowed to build an index over the public datasets first (like: [5, 6]).

   See also this recent review paper [27].

   Note that the best projects are ones that explore and compare different algorithmic ideas. Applying a machine-learning model to the problem may or may not leave you with much to compare or interpret when the results come back. We suggest sticking to methods whose performance can be predicted and explained.

4. **Refining the reference genome**: The reference genome is never an exact genetic clone of the genome being sequenced. Rather the sequenced genome (sometimes called the "donor") and the reference will differ by mismatches, gaps, or larger differences called structural variationts. Alignments that overlap these differences will be harder to find. In some cases,

especially for the larger differences, or for cases where there are many differences close together, the alignment will be impossible to find.

For each read that does align, though, we get a little more information about how the donor differs from the reference. (Think back to Homework 2, where we did a simplified version of this.) We can use alignments, and the information we have tallied so far at every position, to "update" the reference; i.e. to insert mismatches and gaps to make the reference genome match the genome being sequenced more closely. Try to implement a scheme like this. Some questions you might pursue are: How to efficiently *update* the indexes discussed in class with new alleles? See: [28, 29]. (These are challenging papers.) How would you design an overall system the iteratively refines the reference genome in light of new evidence? See: [30, 31].

For a recent example of why this is a useful thing to do: [32].

5. **SIMD dynamic programming**: For people comfortable with C/C++ and low-level programming: Implement a few SIMD-accelerated dynamic programming schemes and suggest situations in which one might be preferred to others. For instance, is one better for DNA alignment, as opposed to, say, protein alignment? Is one better for longer queries? See: [33, 34, 35, 36].

   Note: This is more like the "seed" of a project, rather than an entire project. If you pick this topic, try to take your ideas well beyond the ones written here.

6. **Indexing large collections of reads**: Design a data structure for storing large collections of sequencing reads in a way that allows search queries, like:

   - Tell me whether this sequence occurs anywhere in the collection
   - Tell me in which sequencing datasets the following sequence occurs
   - List all the places (dataset, chromosome, offset) where this sequence occurs.

   See also: [37, 38, 39, 40]. Start small and work your way up to bigger and bigger collections of sequencing datasets. This is related to the *document listing problem* in information retrieval.

   Note: This is more like the "seed" of a project, rather than an entire project. If you pick this topic, try to take your ideas well beyond the ones written here.

7. **Aligning reads to protein databases**: Implement a tool that takes a collection of sequencing reads (DNA) and aligns them against a database of known proteins (amino acids). To align DNA to amino acids, you will need to translate the DNA into protein in all 6 possible reading frames, 3 on the forward strand and 3 on the reverse complement strand. See also: [41, 42, 43, 44, 45, 46].

   Note: This is more like the "seed" of a project, rather than an entire project. If you pick this topic, try to take your ideas well beyond the ones written here.

8. **Positional Burrows-Wheeler Transform**: We discussed the Burrows-Wheeler Transform and its role in constructing useful text indexes (like the FM Index [47]) that help to solve the read alignment problem. But the BWT is also used to solve other problems in Genomics, like the genotype imputation problem.

- Read about the genotype imputation problem. Some survey articles might help [48, 49].
- Consider how the Positional Burrows-Wheeler Transform is applied to the genotype imputation problem [50], [51]. Is it useful/possible to relate these to the Hidden Markov Model ideas we covered in class?
- Can you implement and compare simple genotype imputation methods based on PBWT and/or HMMs?

Note: This is more like the "seed" of a project, rather than an entire project. If you pick this topic, try to take your ideas well beyond the ones written here.

2 Literature Suggestions

Here are several references that should get you started in your literature survey once you have some idea of what direction you want to go in. If there's a relevant computational topic you want to learn more about that isn't represented here, email us and we'll augment the list.

- Indexing: [52, 53, 54, 55, 56, 57, 47, 20].
- Indeing for "pangenomes" and "pangenome graphs": [**?**, 58, 59]
- Read alignment: Bowtie: [52], Bowtie 2: [60], BWA: [53], BWA-SW: [61], GEM [62], SHRiMP: [63], CloudBurst (MapReduce-based): [64], GASSST [65], Minimap [66], Minimap2 [67], random permutations: [68].
- Assembly: Velvet: [69], SGA: [70], ABySS [71], (unnamed succinct approach) [72], Miniasm [66]. Assembly review articles: [73], [74], [75].
- Error correction: Quake: [76]. Musket: [77]. Lighter: [78].
- Read compression: CRAM: [79], Quip: [16], SCALCE: [80], NGC: [81], QualComp: [82], Bonfield and Mahoney [83], Boiler [84].
- Basic dynamic programming: [54].
- Accelerating dynamic programming with special-purpose hardware. With SIMD: [33, 34, 35]. With GPUs: [85, 86, 87, 88, 89, 90, 91, 92, 93, 94]. With FPGAs: [95]. (Before considering using a CPU or FPGA, recall that we have to be able to easily run your software! Ask us if you are in doubt.)
- Multiple sequence alignment: CLUSTAL: [96], MUSCLE: [97], MGA: [98].
- RNA sequencing and spliced alignment: TopHat: [99], Cufflinks: [100], MapSplice: [101], STAR: [102], Spaln2: [103], HISAT: [104], Rail-RNA: [105].
- Scalable tools for sequencing data analysis: CloudBurst: [64], Crossbow: [106], Myrna: [107], ADAM: [108], eXpress-D: [109], WiggleTools: [110], mosdepth [111].
- Metagenomics: Phymm / PhymmBL [112].

- Search strategies for combinatorial search / tree search problems such as the co-traversal we discussed in class: Beam search [113], BLFS [114], or ILDS [115]. General article on tree search: [116].
- Indexing and querying collections of sequence reads: [38, 39, 40].
- Incrementally updating suffix indexes: [28, 29].
- Iterative refinement of reference genome: [30, 31]
- Streaming algorithms [117], sketch data structures: CountMin sketch [3], which is related to the Bloom Filter [1], Counting Quotient Filter [9], Bloom Sequence Tree [5], Bloom Trie [6], HyperLogLog [11], SetSketch [12]. Review articles: [17, 18].
- Translated search: BLASTX [41] (mentioned briefly in discussion section; hard to find a good citation for this), PADUA [42], DIAMOND [44], Lambda [43].
- Applications of sketch data structures to k-mer counting [14], database search [5], metagenomics assembly [15], pan genomics [6], and compression [16].
- Applications of MinHashing, locality-sensitive hashing, minimizers: orginal paper [118], Kraken [26], Sailfish [119], MHAP [120], Mash [121], Minimap [66], mDBG [122], Dashing [11], Dashing 2 [123].
- Use of FM Index for assembly: [124], [70].
- Genotype imputation using HMMs [48, 49, 125], using PBWT: [50], [51], [126], and using minimal positional substring cover (MPSC) [127]

References

[1] Wikipedia. Bloom filter — wikipedia, the free encyclopedia, 2013. [Online; accessed 2-March-2013].

[2] Andrei Broder and Michael Mitzenmacher. Network applications of bloom filters: A survey. *Internet mathematics*, 1(4):485–509, 2004.

[3] Graham Cormode and S Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, 55(1):58–75, 2005.

[4] Tong Yang, Haowei Zhang, Jinyang Li, Junzhi Gong, Steve Uhlig, Shigang Chen, and Xiaoming Li. Heavykeeper: An accurate algorithm for finding top-k elephant flows. *IEEE/ACM Transactions on Networking*, 27(5):1845–1858, 2019.

[5] Brad Solomon and Carleton Kingsford. Large-scale search of transcriptomic read sets with sequence bloom trees. *bioRxiv*, page 017087, 2015.

[6] Guillaume Holley, Roland Wittler, and Jens Stoye. Bloom filter trie: an alignment-free and reference-free data structure for pan-genome storage. *Algorithms for Molecular Biology*, 11(1):1, 2016.

[7] Camille Marchet and Antoine Limasset. Scalable sequence database search using partitioned aggregated bloom comb-trees. *bioRxiv*, 2022.

[8] Michael A Bender, Martin Farach-Colton, Rob Johnson, Russell Kraner, Bradley C Kuszmaul, Dzejla Medjedovic, Pablo Montes, Pradeep Shetty, Richard P Spillane, and Erez Zadok. Don't thrash: how to cache your hash on flash. *Proceedings of the VLDB Endowment*, 5(11):1627–1637, 2012.

[9] Prashant Pandey, Michael A Bender, Rob Johnson, and Rob Patro. A general-purpose counting filter: Making every bit count. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 775–787. ACM, 2017.

[10] Bin Fan, Dave G Andersen, Michael Kaminsky, and Michael D Mitzenmacher. Cuckoo filter: Practically better than bloom. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, pages 75–88. ACM, 2014.

[11] Daniel N Baker and Benjamin Langmead. Dashing: Fast and accurate genomic distances with hyperloglog. *BioRxiv*, page 501726, 2018.

[12] Otmar Ertl. Setsketch: Filling the gap between minhash and hyperloglog, 2021.

[13] Otmar Ertl. Ultraloglog: A practical and more space-efficient alternative to hyperloglog for approximate distinct counting. *arXiv preprint arXiv:2308.16862*, 2023.

[14] Páll Melsted and Jonathan K Pritchard. Efficient counting of k-mers in dna sequences using a bloom filter. *BMC bioinformatics*, 12(1):333, 2011.

[15] Jason Pell, Arend Hintze, Rosangela Canino-Koning, Adina Howe, James M Tiedje, and C Titus Brown. Scaling metagenome sequence assembly with probabilistic de bruijn graphs. *Proceedings of the National Academy of Sciences*, 109(33):13272–13277, 2012.

[16] Daniel C Jones, Walter L Ruzzo, Xinxia Peng, and Michael G Katze. Compression of next-generation sequencing reads aided by highly efficient de novo assembly. *Nucleic Acids Research*, 2012.

[17] W. P. M. Rowe. When the levee breaks: a practical guide to sketching algorithms for processing the flood of genomic data. *Genome Biol.*, 20(1):199, Sep 2019.

[18] Guillaume Marçais, Brad Solomon, Rob Patro, and Carl Kingsford. Sketching and sublinear data structures in genomics. *Annual Review of Biomedical Data Science*, 2019.

[19] Camille Marchet, Christina Boucher, Simon J Puglisi, Paul Medvedev, Mikaël Salson, and Rayan Chikhi. Data structures based on k-mers for querying large collections of sequencing datasets. *bioRxiv*, page 866756, 2019.

[20] Travis Gagie, Giovanni Manzini, and Jouni Sirén. Wheeler graphs: A framework for bwt-based data structures. *Theoretical computer science*, 698:67–78, 2017.

[21] K. H. Chao, P. W. Chen, S. A. Seshia, and B. Langmead. WGT: Tools and algorithms for recognizing, visualizing, and generating Wheeler graphs. *iScience*, 26(8):107402, Aug 2023.

[22] Richard W Lusk. Diverse and widespread contamination evident in the unmapped depths of high throughput sequencing data. *PloS one*, 9(10):e110808, 2014.

[23] Anthony O Olarerin-George and John B Hogenesch. Assessing the prevalence of mycoplasma contamination in cell culture via a survey of ncbi's rna-seq archive. *Nucleic acids research*, 43(5):2535–2542, 2015.

[24] William B Langdon. Mycoplasma contamination in the 1000 genomes project. *BioData Mining*, 7(1):1, 2014.

[25] Junho Kim, Ju Heon Maeng, Jae Seok Lim, Hyeonju Son, Junehawk Lee, Jeong Ho Lee, and Sangwoo Kim. Vecuum: identification and filtration of false somatic variants caused by recombinant vector contamination. *Bioinformatics*, page btw383, 2016.

[26] Derrick E Wood and Steven L Salzberg. Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biol*, 15(3):R46, 2014.

[27] L. Cornet and D. Baurain. Contamination detection in genomic data: more is not enough. *Genome Biol*, 23(1):60, 02 2022.

[28] Mikaël Salson, Thierry Lecroq, Martine Léonard, and Laurent Mouchard. Dynamic extended suffix arrays. *Journal of Discrete Algorithms*, 8(2):241–257, 2010.

[29] Mikael Salson, Thierry Lecroq, Martine Léonard, and Laurent Mouchard. A four-stage algorithm for updating a burrows–wheeler transform. *Theoretical Computer Science*, 410(43):4350–4359, 2009.

[30] Alaa Ghanayim and Dan Geiger. Iterative referencing for improving the interpretation of dna sequence data. 2013.

[31] Karel Břinda, Valentina Boeva, and Gregory Kucherov. Dynamic read mapping and online consensus calling for better variant detection. *arXiv preprint arXiv:1605.09070*, 2016.

[32] C. Groza, T. Kwan, N. Soranzo, T. Pastinen, and G. Bourque. Personalized and graph genomes reveal missing signal in epigenomic data. *Genome Biol*, 21(1):124, 05 2020.

[33] Torbjørn Rognes and Erling Seeberg. Six-fold speed-up of smith–waterman sequence database searches using parallel processing on common microprocessors. *Bioinformatics*, 16(8):699–706, 2000.

[34] Michael Farrar. Striped smith–waterman speeds database searches six times over other simd implementations. *Bioinformatics*, 23(2):156–161, 2007.

[35] Torbjørn Rognes. Faster smith-waterman database searches with inter-sequence simd parallelisation. *BMC bioinformatics*, 12(1):221, 2011.

[36] Hajime Suzuki and Masahiro Kasahara. Acceleration of nucleotide semi-global alignment with adaptive banded dynamic programming. *bioRxiv*, page 130633, 2017.

[37] Niko Välimäki and Veli Mäkinen. Space-efficient algorithms for document retrieval. In *Annual Symposium on Combinatorial Pattern Matching*, pages 205–215. Springer, 2007.

[38] Anthony J Cox, Markus J Bauer, Tobias Jakobi, and Giovanna Rosone. Large-scale compression of genomic sequence databases with the burrows–wheeler transform. *Bioinformatics*, 28(11):1415–1419, 2012.

[39] Philippe Nicolas, Salson Mikaël, Lecroq Thierry, Léonard Martine, Commes Thérèse, and Rivals Eric. Querying large read collections in main memory: a versatile data structure. *BMC Bioinformatics*, 12.

[40] Dirk-Dominic Dolle, Zhicheng Liu, Matthew L Cotten, Jared T Simpson, Zamin Iqbal, Richard Durbin, Shane McCarthy, and Thomas Keane. Using reference-free compressed data structures to analyse sequencing reads from thousands of human genomes. *bioRxiv*, page 060186, 2016.

[41] 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.

[42] Daniel H Huson and Chao Xie. A poor man's blastx-high-throughput metagenomic protein database search using pauda. *Bioinformatics*, page btt254, 2013.

[43] Hannes Hauswedell, Jochen Singer, and Knut Reinert. Lambda: the local aligner for massive biological data. *Bioinformatics*, 30(17):i349–i355, 2014.

[44] Benjamin Buchfink, Chao Xie, and Daniel H Huson. Fast and sensitive protein alignment using diamond. *Nature methods*, 12(1):59–60, 2015.

[45] A. Westbrook, J. Ramsdell, T. Schuelke, L. Normington, R. D. Bergeron, W. K. Thomas, and M. D. MacManes. PALADIN: protein alignment for functional profiling whole metagenome shotgun data. *Bioinformatics*, 33(10):1473–1478, May 2017.

[46] M. Steinegger and J. Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat Biotechnol*, 35(11):1026–1028, 11 2017.

[47] Paolo Ferragina and Giovanni Manzini. Opportunistic data structures with applications. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 390–398. IEEE, 2000.

[48] S. Das, G. R. Abecasis, and B. L. Browning. Genotype Imputation from Large Reference Panels. *Annu Rev Genomics Hum Genet*, 19:73–96, Aug 2018.

[49] Y. Li, C. Willer, S. Sanna, and G. Abecasis. Genotype imputation. *Annu Rev Genomics Hum Genet*, 10:387–406, 2009.

[50] R. Durbin. Efficient haplotype matching and storage using the positional Burrows-Wheeler transform (PBWT). *Bioinformatics*, 30(9):1266–1272, May 2014.

[51] G. Lunter. Haplotype matching in large cohorts using the Li and Stephens model. *Bioinformatics*, 35(5):798–806, Mar 2019.

[52] Ben Langmead, Cole Trapnell, Mihai Pop, Steven L Salzberg, et al. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome Biol*, 10(3):R25, 2009.

[53] Heng Li and Richard Durbin. Fast and accurate short read alignment with burrows–wheeler transform. *Bioinformatics*, 25(14):1754–1760, 2009.

[54] Dan Gusfield. *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge University Press, 1997.

[55] Udi Manber and Gene Myers. Suffix arrays: a new method for on-line string searches. *siam Journal on Computing*, 22(5):935–948, 1993.

[56] Mohamed Ibrahim Abouelhoda, Stefan Kurtz, and Enno Ohlebusch. Replacing suffix trees with enhanced suffix arrays. *Journal of Discrete Algorithms*, 2(1):53–86, 2004.

[57] Juha Kärkkäinen and Peter Sanders. Simple linear work suffix array construction. *Automata, Languages and Programming*, pages 187–187, 2003.

[58] F. Almodaresi, P. Pandey, M. Ferdman, R. Johnson, and R. Patro. An Efficient, Scalable, and Exact Representation of High-Dimensional Color Information Enabled Using de Bruijn Graph Search. *J Comput Biol*, 27(4):485–499, Apr 2020.

[59] J. Fan, J. Khan, N. P. Singh, G. E. Pibiri, and R. Patro. Fulgor: A fast and compact k-mer index for large-scale matching and color queries. *bioRxiv*, May 2023.

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

[61] Heng Li and Richard Durbin. Fast and accurate long-read alignment with burrows–wheeler transform. *Bioinformatics*, 26(5):589–595, 2010.

[62] Santiago Marco-Sola, Michael Sammeth, Roderic Guigó, and Paolo Ribeca. The gem mapper: fast, accurate and versatile alignment by filtration. *Nature methods*, 9(12):1185–1188, 2012.

[63] Stephen M Rumble, Phil Lacroute, Adrian V Dalca, Marc Fiume, Arend Sidow, and Michael Brudno. Shrimp: accurate mapping of short color-space reads. *PLoS computational biology*, 5(5):e1000386, 2009.

[64] Michael C Schatz. Cloudburst: highly sensitive read mapping with mapreduce. *Bioinformatics*, 25(11):1363–1369, 2009.

[65] Guillaume Rizk and Dominique Lavenier. Gassst: global alignment short sequence search tool. *Bioinformatics*, 26(20):2534–2540, 2010.

[66] Heng Li. Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences. *Bioinformatics*, 32(14):2103–2110, 2016.

[67] Heng Li. Minimap2: fast pairwise alignment for long dna sequences. *arXiv preprint arXiv:1708.01492*, 2017.

[68] Roy Lederman. A random-permutations-based approach to fast read alignment. *BMC bioinformatics*, 14(S-5):S8, 2013.

[69] Daniel R Zerbino and Ewan Birney. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome research*, 18(5):821–829, 2008.

[70] Jared T Simpson and Richard Durbin. Efficient de novo assembly of large genomes using compressed data structures. *Genome research*, 22(3):549–556, 2012.

[71] Jared T Simpson, Kim Wong, Shaun D Jackman, Jacqueline E Schein, Steven JM Jones, and İnanç Birol. Abyss: a parallel assembler for short read sequence data. *Genome research*, 19(6):1117–1123, 2009.

[72] Thomas C Conway and Andrew J Bromage. Succinct data structures for assembling large genomes. *Bioinformatics*, 27(4):479–486, 2011.

[73] Michael C Schatz, Arthur L Delcher, and Steven L Salzberg. Assembly of large genomes using second-generation sequencing. *Genome research*, 20(9):1165–1173, 2010.

[74] Jason R Miller, Sergey Koren, and Granger Sutton. Assembly algorithms for next-generation sequencing data. *Genomics*, 95(6):315, 2010.

[75] Phillip EC Compeau, Pavel A Pevzner, and Glenn Tesler. How to apply de bruijn graphs to genome assembly. *Nature biotechnology*, 29(11):987–991, 2011.

[76] David R Kelley, Michael C Schatz, and Steven L Salzberg. Quake: quality-aware detection and correction of sequencing errors. *Genome Biol*, 11(11):R116, 2010.

[77] Yongchao Liu, Jan Schröder, and Bertil Schmidt. Musket: a multistage k-mer spectrum-based error corrector for illumina sequence data. *Bioinformatics*, 29(3):308–315, 2013.

[78] Li Song, Liliana Florea, and Ben Langmead. Lighter: fast and memory-efficient error correction without counting. *bioRxiv*, 2014.

[79] Markus Hsi-Yang Fritz, Rasko Leinonen, Guy Cochrane, and Ewan Birney. Efficient storage of high throughput dna sequencing data using reference-based compression. *Genome research*, 21(5):734–740, 2011.

[80] Faraz Hach, Ibrahim Numanagić, Can Alkan, and S Cenk Sahinalp. Scalce: boosting sequence compression algorithms using locally consistent encoding. *Bioinformatics*, 28(23):3051–3057, 2012.

[81] Niko Popitsch and Arndt von Haeseler. Ngc: lossless and lossy compression of aligned high-throughput sequencing data. *Nucleic acids research*, 41(1):e27–e27, 2013.

[82] Idoia Ochoa, Himanshu Asnani, Dinesh Bharadia, Mainak Chowdhury, Tsachy Weissman, and Golan Yona. Qualcomp: a new lossy compressor for quality scores based on rate distortion theory. *BMC bioinformatics*, 14(1):187, 2013.

[83] James K Bonfield and Matthew V Mahoney. Compression of fastq and sam format sequencing data. *PloS one*, 8(3):e59190, 2013.

[84] J. Pritt and B. Langmead. Boiler: lossy compression of RNA-seq alignments using coverage vectors. *Nucleic Acids Res.*, Jun 2016.

[85] Michael C Schatz, Cole Trapnell, Arthur L Delcher, and Amitabh Varshney. High-throughput sequence alignment using graphics processing units. *BMC bioinformatics*, 8(1):474, 2007.

[86] Yongchao Liu, Douglas L Maskell, and Bertil Schmidt. Cudasw++: optimizing smith-waterman sequence database searches for cuda-enabled graphics processing units. *BMC research notes*, 2(1):73, 2009.

[87] Yongchao Liu, Bertil Schmidt, and Douglas L Maskell. Cudasw++ 2.0: enhanced smith-waterman protein database search on cuda-enabled gpus based on simt and virtualized simd abstractions. *BMC Research Notes*, 3(1):93, 2010.

[88] Yongchao Liu, Bertil Schmidt, and Douglas L Maskell. Cushaw: a cuda compatible short read aligner to large genomes based on the burrows–wheeler transform. *Bioinformatics*, 28(14):1830–1837, 2012.

[89] Yongchao Liu and Bertil Schmidt. Evaluation of gpu-based seed generation for computational genomics using burrows-wheeler transform. In *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International*, pages 684–690. IEEE, 2012.

[90] Yongchao Liu and Bertil Schmidt. Long read alignment based on maximal exact match seeds. *Bioinformatics*, 28(18):i318–i324, 2012.

[91] Chi-Man Liu, Thomas Wong, Edward Wu, Ruibang Luo, Siu-Ming Yiu, Yingrui Li, Bingqiang Wang, Chang Yu, Xiaowen Chu, Kaiyong Zhao, et al. Soap3: ultra-fast gpu-based parallel alignment tool for short reads. *Bioinformatics*, 28(6):878–879, 2012.

[92] Petr Klus, Simon Lam, Dag Lyberg, Ming S Cheung, Graham Pullan, Ian McFarlane, Giles SH Yeo, and Brian YH Lam. Barracuda-a fast short read sequence aligner using graphics processing units. *BMC research notes*, 5(1):27, 2012.

[93] Ruibang Luo, Thomas Wong, Jianqiao Zhu, Chi-Man Liu, Edward Wu, Lap-Kei Lee, Haoxiang Lin, Wenjuan Zhu, David W Cheung, Hing-Fung Ting, et al. Soap3-dp: Fast, accurate and sensitive gpu-based short read aligner. *arXiv preprint arXiv:1302.5507*, 2013.

[94] Richard Wilton, Tamas Budavari, Ben Langmead, Sarah J Wheelan, Steven L Salzberg, and Alexander S Szalay. Arioc: high-throughput read alignment with gpu-accelerated exploration of the seed-and-extend search space. *PeerJ*, 3:e808, 2015.

[95] Y. Chen, B. Schmidt, and D. L. Maskell. A hybrid short read mapping accelerator. *BMC Bioinformatics*, 14(1):67, Feb 2013.

[96] MA Larkin, G Blackshields, NP Brown, R Chenna, PA McGettigan, H McWilliam, F Valentin, IM Wallace, A Wilm, R Lopez, et al. Clustal w and clustal x version 2.0. *Bioinformatics*, 23(21):2947–2948, 2007.

[97] Robert C Edgar. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, 32(5):1792–1797, 2004.

[98] Michael Höhl, Stefan Kurtz, and Enno Ohlebusch. Efficient multiple genome alignment. *Bioinformatics*, 18(suppl 1):S312–S320, 2002.

[99] Cole Trapnell, Lior Pachter, and Steven L Salzberg. Tophat: discovering splice junctions with rna-seq. *Bioinformatics*, 25(9):1105–1111, 2009.

[100] Cole Trapnell, Brian A Williams, Geo Pertea, Ali Mortazavi, Gordon Kwan, Marijke J Van Baren, Steven L Salzberg, Barbara J Wold, and Lior Pachter. Transcript assembly and quantification by rna-seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature biotechnology*, 28(5):511–515, 2010.

[101] Kai Wang, Darshan Singh, Zheng Zeng, Stephen J Coleman, Yan Huang, Gleb L Savich, Xiaping He, Piotr Mieczkowski, Sara A Grimm, Charles M Perou, et al. Mapsplice: accurate mapping of rna-seq reads for splice junction discovery. *Nucleic acids research*, 38(18):e178–e178, 2010.

[102] A. Dobin, C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, and T. R. Gingeras. STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, 29(1):15–21, Jan 2013.

[103] Hiroaki Iwata and Osamu Gotoh. Benchmarking spliced alignment programs including spaln2, an extended version of spaln that incorporates additional species-specific features. *Nucleic Acids Research*, 2012.

[104] Daehwan Kim, Ben Langmead, and Steven L Salzberg. Hisat: a fast spliced aligner with low memory requirements. *Nature methods*, 12(4):357–360, 2015.

[105] Abhinav Nellore, Leonardo Collado-Torres, Andrew E Jaffe, José Alquicira-Hernández, Christopher Wilks, Jacob Pritt, James Morton, Jeffrey T Leek, and Ben Langmead. Rail-rna: Scalable analysis of rna-seq splicing and coverage. *Bioinformatics*, 2016.

[106] Ben Langmead, Michael C Schatz, Jimmy Lin, Mihai Pop, and Steven L Salzberg. Searching for snps with cloud computing. *Genome Biol*, 10(11):R134, 2009.

[107] Ben Langmead, Kasper D Hansen, Jeffrey T Leek, et al. Cloud-scale rna-sequencing differential expression analysis with myrna. *Genome Biol*, 11(8):R83, 2010.

[108] Matt Massie, Frank Nothaft, Christopher Hartl, Christos Kozanitis, André Schumacher, Anthony D Joseph, and David A Patterson. Adam: Genomics formats and processing patterns for cloud scale computing. Technical report, Technical Report UCB/EECS-2013-207, EECS Department, University of California, Berkeley, 2013.

[109] Adam Roberts, Harvey Feng, and Lior Pachter. Fragment assignment in the cloud with express-d. *BMC bioinformatics*, 14(1):358, 2013.

[110] Daniel R Zerbino, Nathan Johnson, Thomas Juettemann, Steven P Wilder, and Paul Flicek. Wiggletools: parallel processing of large collections of genome-wide datasets for visualization and statistical analysis. *Bioinformatics*, page btt737, 2013.

[111] B. S. Pedersen and A. R. Quinlan. Mosdepth: quick coverage calculation for genomes and exomes. *Bioinformatics*, 34(5):867–868, 03 2018.

[112] Arthur Brady and Steven L Salzberg. Phymm and phymmbl: metagenomic phylogenetic classification with interpolated markov models. *Nature methods*, 6(9):673–676, 2009.

[113] Wikipedia. Beam search — wikipedia, the free encyclopedia, 2013. [Online; accessed 2-March-2013].

[114] Kevin Rose, Ethan Burns, and Wheeler Ruml. Best-first search for bounded-depth trees. In *Fourth Annual Symposium on Combinatorial Search*, 2011.

[115] Richard E Korf et al. Improved limited discrepancy search. In *Proceedings of the National Conference on Artificial Intelligence*, pages 286–291, 1996.

[116] Wikipedia. Tree traversal — wikipedia, the free encyclopedia, 2013. [Online; accessed 2-March-2013].

[117] Wikipedia. Streaming algorithm — wikipedia, the free encyclopedia, 2013. [Online; accessed 2-March-2013].

[118] Michael Roberts, Brian R Hunt, James A Yorke, Randall A Bolanos, and Arthur L Delcher. A preprocessor for shotgun assembly of large genomes. *Journal of computational biology*, 11(4):734–752, 2004.

[119] Rob Patro, Stephen M Mount, and Carl Kingsford. Sailfish enables alignment-free isoform quantification from rna-seq reads using lightweight algorithms. *Nature biotechnology*, 32(5):462–464, 2014.

[120] Konstantin Berlin, Sergey Koren, Chen-Shan Chin, James P Drake, Jane M Landolin, and Adam M Phillippy. Assembling large genomes with single-molecule sequencing and locality-sensitive hashing. *Nature biotechnology*, 2015.

[121] 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):132, 2016.

[122] Barış Ekim, Bonnie Berger, and Rayan Chikhi. Minimizer-space de bruijn graphs. *bioRxiv*, 2021.

[123] D. N. Baker and B. Langmead. Genomic sketching with multiplicities and locality-sensitive hashing using Dashing 2. *Genome Res*, 33(7):1218–1227, Jul 2023.

[124] Jared T Simpson and Richard Durbin. Efficient construction of an assembly string graph using the fm-index. *Bioinformatics*, 26(12):i367–i373, 2010.

[125] B. L. Browning, Y. Zhou, and S. R. Browning. A One-Penny Imputed Genome from Next-Generation Reference Panels. *Am J Hum Genet*, 103(3):338–348, Sep 2018.

[126] S. Rubinacci, D. M. Ribeiro, R. J. Hofmeister, and O. Delaneau. Efficient phasing and imputation of low-coverage sequencing data using large reference panels. *Nat Genet*, 53(1):120–126, Jan 2021.

[127] A. Sanaullah, D. Zhi, and S. Zhang. Minimal positional substring cover is a haplotype threading alternative to Li and Stephens model. *Genome Res*, 33(7):1007–1014, Jul 2023.