The State of Bioinformatics in High Performance Computing in 2017

Philipp Bayer
@PhilippBayer
School of Biological Sciences
University of Western Australia
Basic idea of this talk

• I want to:
  • Give an introduction to a subfield of modern bioinformatics, genome assembly
  • Show you the scale of the work needed
  • Introduce modern concepts to show where the field is going
  • So that the HPC people know more about what the bioinformatics people are doing
Who am I?

- Did my biology undergrad in Germany, started to work around bioinformatics in 2008
- Finished PhD at UQ in 2016 in Dave Edwards’ group
- Since then Postdoctoral researcher at UWA in Dave Edwards’ group
The group has been involved in a few big plant (pan-)genome papers.
• We’re mostly focusing on improving plant breeding using computational methods – SNPs, k-mers, gene presence/absence, graph databases, phenotype linking

• Involved in many plant genome sequencing projects: lentil, pea, clover, chickpea, seagrasses, wine, wheat, soybean, canola, …

• Working with Bayer CropScience on data from breeding programs
• For reference, the human genome is 3.2 giga base pairs = 3,200,000,000 base pairs (3.2 billion letters, ATGC)

• Sequence using sequencing machines: Illumina HiSeq, PacBio RS II, Oxford MinION

• Drawback: current machines can only give you 2x250bp (Illumina, accurate) to 1x10,000 bp (PacBio, less accurate) pieces (reads) of the genome
Sequence costs are going down, down, down...

Source: https://www.genome.gov/sequencingcostsdata/
• Many genome assembly algorithms in use: first, clean noise from data, then link data from reads in a (De Bruijn) graph, then find an Eulerian Path through that graph for a representation of that genome in text format

• We have a read ACTAG, let $k=4$:
  • Vertices ACT, CTA, TAG
  • Edges ACT->CTA->TAG

• Errors in reads, repetitive (identically copied) regions, insufficient coverage means there usually is no Eulerian Path, and several graphs are necessary
Genome assembly end result

>scf7180002107662
CATCAGACAAAGGATCGACCAAAAATGGTTTAGGTTAGTCAGCACACTTCAAGTAG
GGATTGGAGGGCTATACAGCTCAATTTGAGGGCTTTCTCCACATTTACAATGAA
CGGGAAATACCTTGTCTCCACAGGACGGTCAGTTAGTTGACATTACAATTATCT
CAGACCTTCTCAAACACTGATAATGCATTAGAAGCAAGGGTATCATCATCAATTG
CCATAGCCTCGGTCTACGAATCTGATCATCACTCAATAGTGCTTCTTTACACCA
ACTTTCGGGCTTCCGGGACCCTG
AATACACAGTGGGATCCGCGTTAGGTCAAGGTTAGAATTGAGGCTGGATATTCA
AAAGACTGTTGGCTCAAGGATGTTGTTGGAAGCTATTAGGAGCCCGAGCATGGA
ATGTAATTCACCACCTACTAAGCTGAATTCTAGTTATCTTTGAAAGAAATTAT
TTCTATCAACCATTGTCACCAAAAGATTCTTTTCTCAATTTGGCCAAAAAGTCT
TTTTCTGTCGAAAGACTGTTGGCTTCTTTCTGATCTCTGATCATGCTTTCT
AACGTGGATCAGGTATCTCCACTCTTCTGACGCTTCCAGAGTCAGGCTCTAGT
AAGCACCCTGCGCCGACTTCTCTTCTCTTCAAGGGTATCCGCCGACCCTTC
CGAAGTTAGGGCTCTAGTAAAGCCAGCCTTTTGGAGGCCTGATTAGCTAGCTG
ATCATGGATCATCCCTCTCT
GAGGGTGTCCTCCCAGCTTTTCGGTAAAGGTGATCCAGTACAGGGCTTTCCG
GAGGGCTCAGTTCTCGAGGCTCTTTCTGTAATGCAAGCTCAAGTGGACACCC
TTTGGAGGGCTCAGTTCTGGTCTCCTTAAAGGCTACCAAGATGGCTTTCCGG
GAGGGCTCAGTTCTGGTCTCCTTAAAGGCTACCAAGATGGCTTTCCGG
GAGGGCTCAGTTCTGGTCTCCTTAAAGGCTACCAAGATGGCTTTCCGG
Assembly: A matter of size

Area \(\propto\) Genome Size

- Wheat 16Gbp
- Barley 5.1Gbp
- Human 3.2Gbp
- Arabidopsis 135Mbp
- Rice 430Mbp
- Chickpea 740Mbp

Source: Nathan Watson-Haigh, https://figshare.com/articles/Plant_Genome_Size_Comparison/4285340/2
A matter of size

Three things with more genes than you

Total number of genes per organism

- Wheat: 120,000
- Rice: 50,000
- Corn: 32,000
- Human: 20,000

Sources: AAAS, The Gene by Siddhartha Mukherjee

Source: Bill Gates,
https://www.gatesnotes.com/Books/The-Gene
Wheat genealogy

Source: https://www.slideshare.net/WheatInitiative/the-genomes-of-civilization
Current status: it’s hard

The first near-complete assembly of the hexaploid bread wheat genome, *Triticum aestivum*

Aleksey V. Zimin\(^1,2\), Daniela Puiu\(^1\), Richard Hall\(^3\), Sarah Kingan\(^3\), and Steven L. Salzberg\(^1,4,*\)

\(^1\)Center for Computational Biology, McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University School of Medicine, Baltimore, MD

\(^2\)Institute for Physical Sciences and Technology, University of Maryland, College Park, MD

\(^3\)Pacific Biosciences, Menlo Park, CA

\(^4\)Departments of Biomedical Engineering, Computer Science, and Biostatistics, Johns Hopkins University, Baltimore, MD

\(*\)Corresponding author

http://www.biorxiv.org/content/biorxiv/early/2017/07/03/159111
To create the wheat genome assembly, we generated two extremely large primary data sets. The first data set consisted of 7.06 billion Illumina reads containing approximately 1 trillion bases of DNA. The Illumina reads were 150-bp, paired reads from short DNA fragments, averaging 400 bp in length. Using an estimated genome size of 15.3 Gbp, this represented 65-fold coverage of the genome. The second data set used Pacific Biosciences single-molecule (SMRT) technology to generate 55.5 million reads with an average read length just under 10,000 bp, containing a total of 545 billion bases of DNA, representing 36-fold coverage of the genome. All reads were
Current status: it’s hard

Construction of super-reads and mega-reads required approximately 100,000 CPU hours, of which 95% was spent in the mega-reads step. By using large multi-core computers to run these steps in parallel, these steps took 1.5 months of elapsed (wall clock) time. The peak memory (RAM) usage was 1.2 terabytes.

We then assembled the mega-reads and the synthetic pairs using the Celera Assembler [7] (v8.3), which was modified to work with our parallel job scheduling system. The CA assembly process required many iterations of the overlapping, error correction, and contig construction steps, and it was extremely time consuming, even with the many optimizations that have been incorporated in this assembler in recent releases. The total CPU time was ∼470,000 CPU hours (53.7 years), which was only made feasible by running it on a grid with thousands of jobs running in parallel for some of the major steps. The total elapsed time was just over 5 months. When combined with the earlier steps, the entire assembly process took 6.5 months. The resulting assembly, labelled Triticum 1.0, contained 17.046 Gb in 829,839 contigs, with an N50 contig size of 76,267 bp and

Run on MARCC at Johns Hopkins, 676 ‘regular’ nodes (16,224 cores), 50 1TB memory nodes (2,400 cores)
As you can see, assembly is still a hard problem that takes a long time and a lot of computational power to solve.

There are some shortcuts but they are all ‘lossy’ (more on that later).
The current status of genome assemblies

• Genome assembly can be imagined as a kind of lossy compression

• The final assembly does NOT represent the ‘true’ genome: errors are introduced, perfect copies are collapsed, hard-to-sequence regions disappear or get misassembled etc…

• The $1000 genome is here, but that $1000 accounts only for raw sequencing costs – nothing about computation and storage costs
We need even more computers...
We need even more computers…

- It took 53.7 CPU-years for \( \frac{1}{2} \) year to assemble a single, non-representative wheat individual

- We are interested in the genetic differences of >500 individuals

- Cannot assemble those one by one - 250 years of walltime is a bit much

- Usually people align their data with the single reference: much faster (hours), but the approach depends on reference quality and choice of reference individual
Where things are going (I think)

• There are several current approaches I’d like to talk about:
  • Computational normalization
  • Pangenome
  • Population graphs
  • K-mer comparisons
  • Word2vec
Computational normalization

- Software: khmer, bbnorm, etc.

- Iterate over all reads, count k-mers in read, remove read if covered by k-mers that have been seen more than \( c \) times

- In my tests with canola this removes about 50-70% of data while retaining \( \sim 99\% \) of information, needs >500GB of memory

- However: some assemblers use this information to untangle repetitive regions => smaller assembly!
The pangenome of an agronomically important crop plant *Brassica oleracea*


**ARTICLE**

Received 22 Aug 2016 | Accepted 28 Sep 2016 | Published 11 Nov 2016

DOI: 10.1038/ncomms13300

**OPEN**

The *pangenome* of an agronomically important crop plant *Brassica oleracea*

**The pangenome of hexaploid bread wheat**

Juan D. Montenegro, Agnieszka A. Golicz, Philipp E. Bayer, Bhavna Hurgobin, HueyTyng Lee, Chon Kit Kenneth Chan, Paul Visendi, Kaitao Lai, Jaroslav Dolezel, Jacqueline Batley, and David Edwards

1 School of Agriculture and Food Sciences, University of Queensland, Brisbane, Australia,
2 School of Plant Biology, University of Western Australia, Crawley, WA 6009, Australia,
3 CSIRO, North Ryde, NSW 2113, Australia,
4 Institute of Experimental Botany, Centre of the Region Hněva for Biotechnological and Agricultural Research, Šlechtitelů 31, CZ-783 71 Olomouc, Czech Republic, and
5 Institute of Agriculture, University of Western Australia, Crawley, WA 6009, Australia
Pangenome

- Align reads with reference assembly, assemble what does not align, add assembly result to reference, align repeats from other individuals, repeat

- Upsides:
  - Relatively fast, alignments are not computationally hungry (just a few GB of memory)

- Downsides:
  - A badly assembled reference sequence leads to many reads falsely not aligning
  - Reads may not align because the underlying region has diverged, not really new information
Population graph

Source: https://www.sevenbridges.com/graph-old/better-reference/
Population graph

- Make an initial graph, then add additional individuals into the graph while keeping track of where what comes from

- Upsides:
  - Less error-susceptible than pangenome

- Downside:
  - Computationally hungry (unclear how much)
  - Not much software available to assemble population graph
K-mer comparisons

• Data generation has become so cheap that we do not need reference assemblies to find differences

• It only costs a few hundred dollars to sequence an individual to sufficient coverage, could make more sense to compare that data directly

• K-mer counting breaks reads up into pieces and compares counts between groups

• If I see a difference in a gene, do I still need a assembly for a publication? (probably not)
K-mer comparisons - HAWK

word2vec – dna2vec

- word2vec trains a two-layer neural network with sentences to ‘learn’ linguistic context of words
- dna2vec does the same, but with k-mers from sequencing data or genome assemblies

**Upside:**
- Can perhaps better look at the entire k-mer space
- Produced embeddings usable as input for more complex models (deep learning etc.)

**Downside:**
- Current implementation very slow, takes days for one chromosome, best using GPUs
dna2vec: Consistent vector representations of variable-length k-mers

Patrick Ng
ppn3@cs.cornell.edu

Abstract

One of the ubiquitous representation of long DNA sequence is dividing it into shorter k-mer components. Unfortunately, the straightforward vector encoding of k-mer as a one-hot vector is vulnerable to the curse of dimensionality. Worse yet, the distance between any pair of one-hot vectors is equidistant. This is particularly problematic when applying the latest machine learning algorithms to solve problems in biological sequence analysis. In this paper, we propose a novel method to train distributed representations of variable-length k-mers. Our method is based on the popular word embedding model word2vec, which is trained on a shallow two-layer neural network. Our experiments provide evidence that the summing of dna2vec vectors is akin to nucleotides concatenation. We also demonstrate that there is correlation between Needleman-Wunsch similarity score and cosine similarity of dna2vec vectors.

dna2vec + t-SNE

Source: https://github.com/philippbayer/cats_dogs-redux/blob/master/Embeddings%20%2B%20keras.ipynb
Key takeaways

- DNA sequencing has become cheap, but computational and data storage needs are still enormous and require HPC.
- Assembly still needs several hundred CPUs and ~1TB of memory and a lot of time.
- Right now several alternative approaches to assembly are developing, but all still need several hundred CPUs and GB/TBs of memory.
Acknowledgements

Agnieszka Golicz
Kenneth Chan
Jenny Lee
Bhavna Hurgobin
Armin Scheben
Yuxuan (Andy) Yuan
Habib Rijzaani
Clementine Mercer

Jacqueline Batley
Dave Edwards

Perth Machine Learning Group (PMLG.org)
Thanks for listening!

Contact: philipp.bayer@uwa.edu.au
@PhilippBayer

Supervisor: dave.edwards@uwa.edu.au