# Principles and Applications of Modern DNA Sequencing

EEEB GU4055

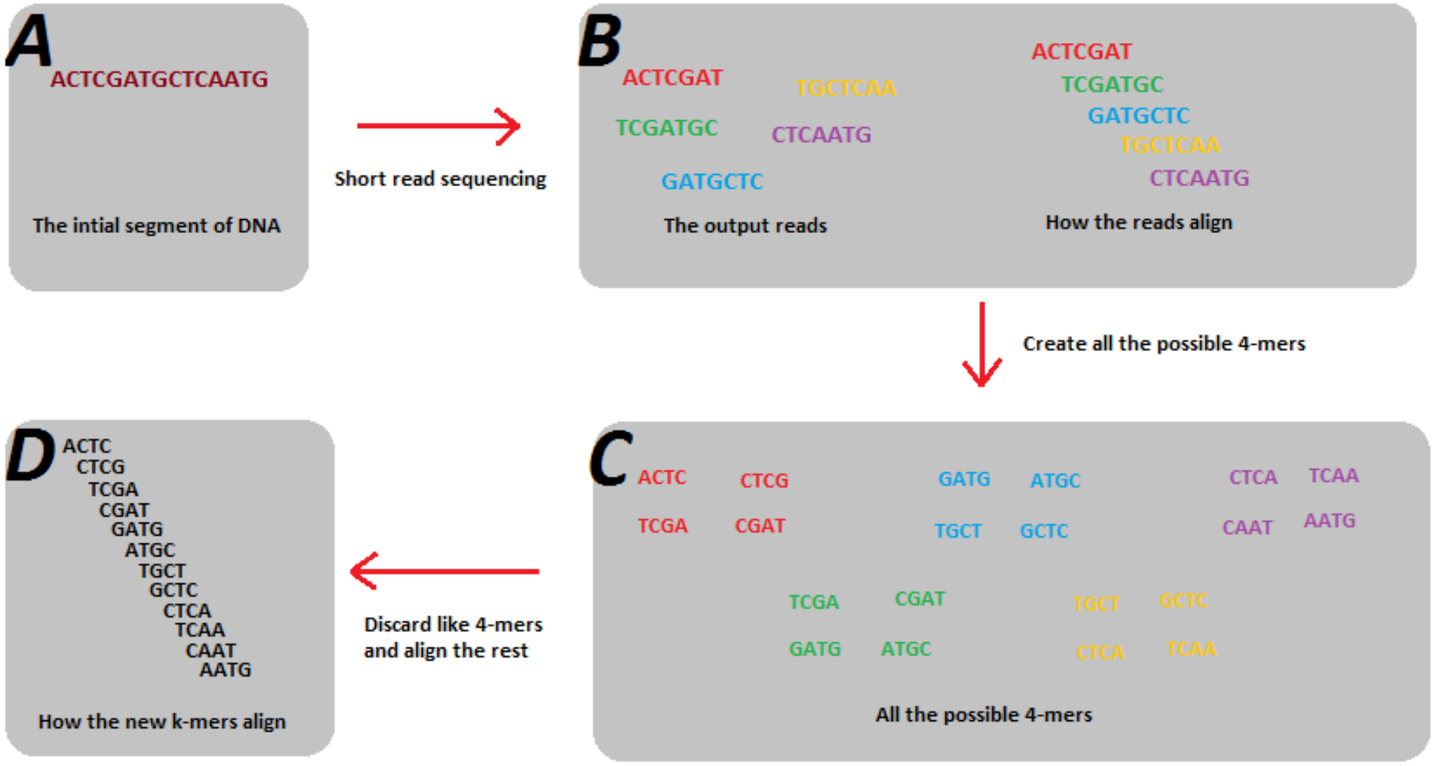
## Session 11: Genome Assembly

# Today's topics

1. de Bruijn Graphs and Euler.
2. Kmers.
3. Challenges in Genome Assembly.
4. Empirical Example.

# Kmers and de Bruijn graphs

Reads start and end at different positions covering all or nearly all of the genome. Decomposing reads into smaller kmers makes it more likely that we have uniformly sized bits covering the entire genome. This is useful for building a graph.



# Kmers and de Bruijn graphs

Shortest possible superstring that contains all substrings of length k.

**Box 1 Origin of de Bruijn graphs**

In 1946, the Dutch mathematician Nicolaas de Bruijn became interested in the 'superstring problem'[12]: find a shortest circular 'superstring' that contains all possible 'substrings' of length $k$ ($k$-mers) over a given alphabet. There exist $n^k$ $k$-mers in an alphabet containing $n$ symbols: for example, given the alphabet comprising A, T, G and C, there are $4^3 = 64$ trinucleotides. If our alphabet is instead 0 and 1, then all possible 3-mers are simply given by all eight 3-digit binary numbers: 000, 001, 010, 011, 100, 101, 110, 111. The circular superstring 0001110100 not only contains all 3-mers but also is as short as possible, as it contains each 3-mer exactly once. But how can one construct such a superstring for all $k$-mers in the case of an arbitrary value of $k$ and an arbitrary alphabet? De Bruijn answered this question by borrowing Euler's solution of the Bridges of Königsberg problem. Briefly, construct a graph $B$ (the original graph called a de Bruijn graph) for which every possible $(k-1)$-mer is assigned to a node; connect one $(k-1)$-mer by a directed edge to a second $(k-1)$-mer if there is some $k$-mer whose prefix is the former and whose suffix is the latter (**Fig. 2**). Edges of the de Bruijn graph represent all possible $k$-mers, and thus an Eulerian cycle in $B$ represents a shortest (cyclic) superstring that contains each $k$-mer exactly once. By checking that the indegree and outdegree of every node in $B$ equals the size of the alphabet, we can verify that $B$ contains an Eulerian cycle. In turn, we can construct an Eulerian cycle using Euler's algorithm, therefore solving the superstring problem. It should now be apparent why the 'de Bruijn graph' construction described in the main text, which does not use all possible $k$-mers as edges but rather only those generated from our reads, is also named in honor of de Bruijn.

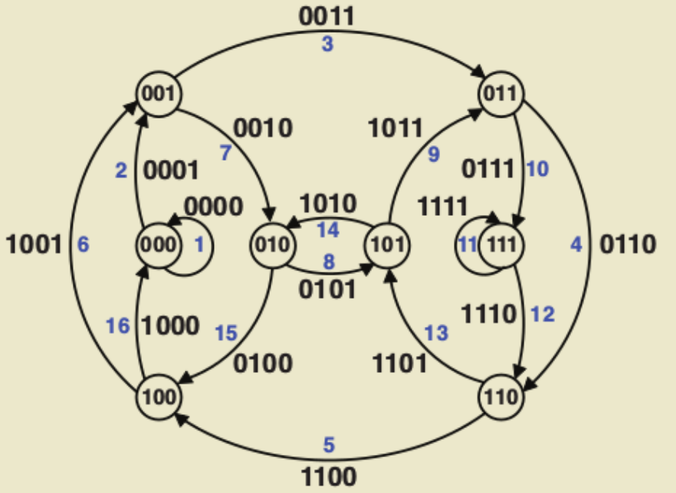
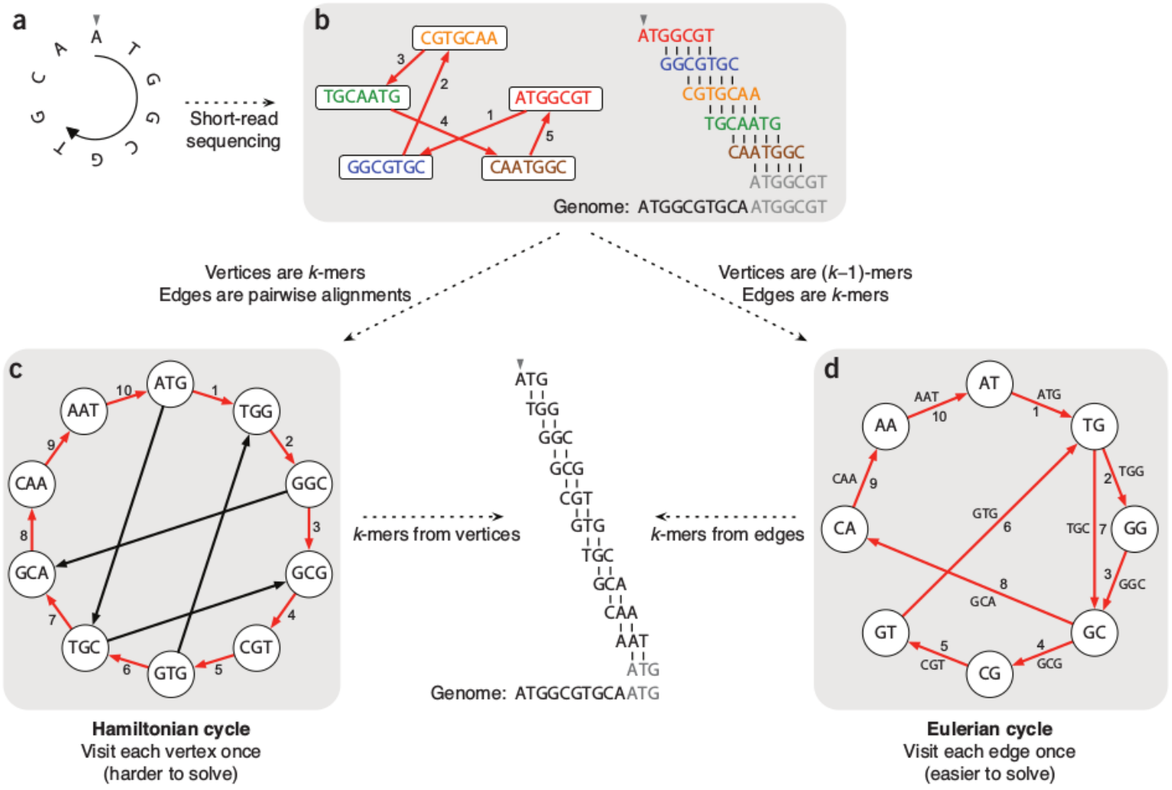



**Figure 2** De Bruijn graph. The de Bruijn graph $B$ for $k = 4$ and a two-character alphabet composed of the digits 0 and 1. This graph has an Eulerian cycle because each node has indegree and outdegree equal to 2. Following the blue numbered edges in order from 1 to 16 traces an Eulerian cycle **0**000, **0**001, **0**011, **0**110, **1**100, **1**001, **0**010, **0**101, **1**011, **0**111, **1**111, **1**110, **1**101, **1**010, **0**100, **1**000. Recording the first character (in boldface) of each edge label spells the cyclic superstring **0000110010111101**.

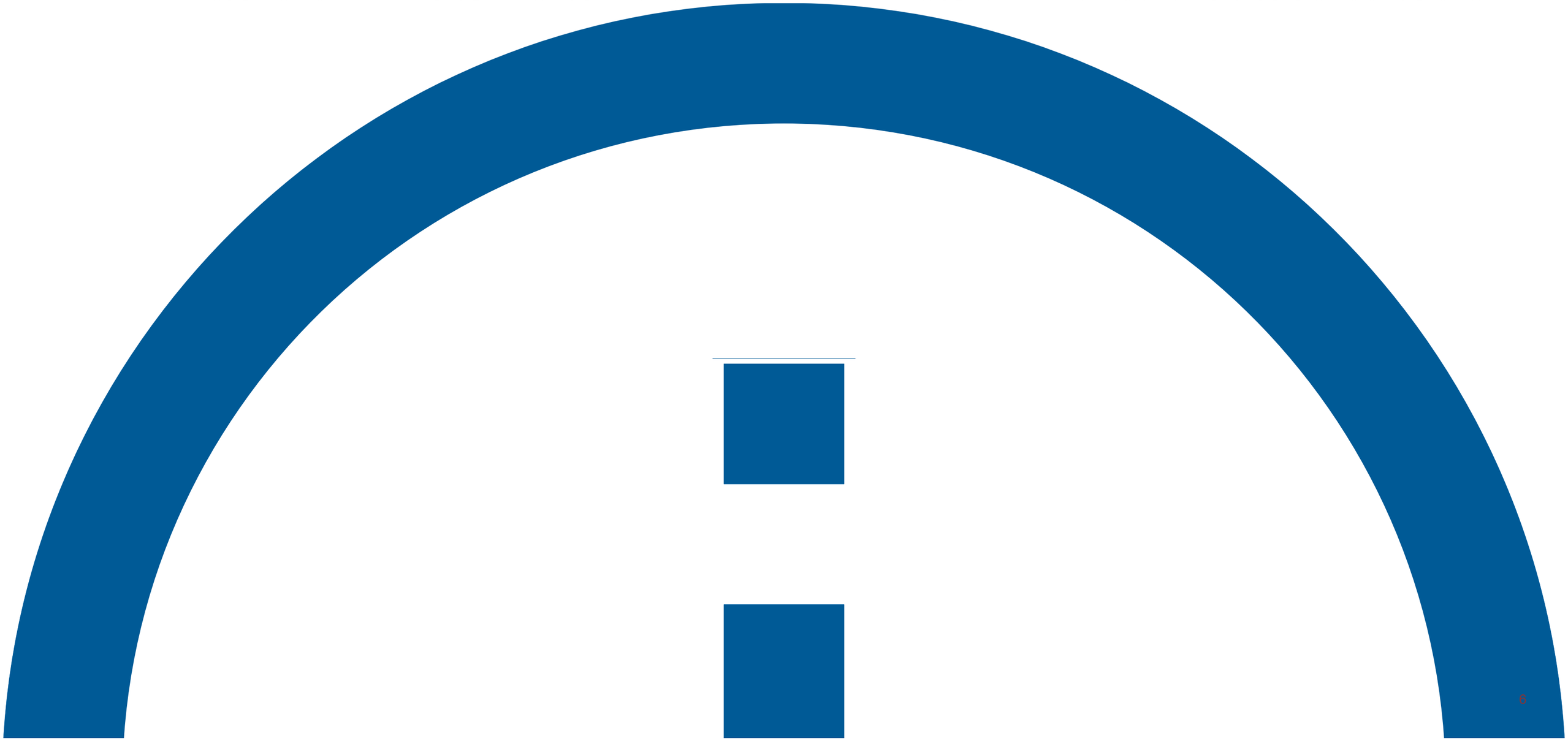
# Kmers and de Bruijn graphs

Hamiltonian graph requires comparing/aligning kmers, which is hard when the number and size of kmers is large. de Bruijn graphs join identical matching (k-1)mers, such that kmers form the edges of the graph -- a much simpler computation.



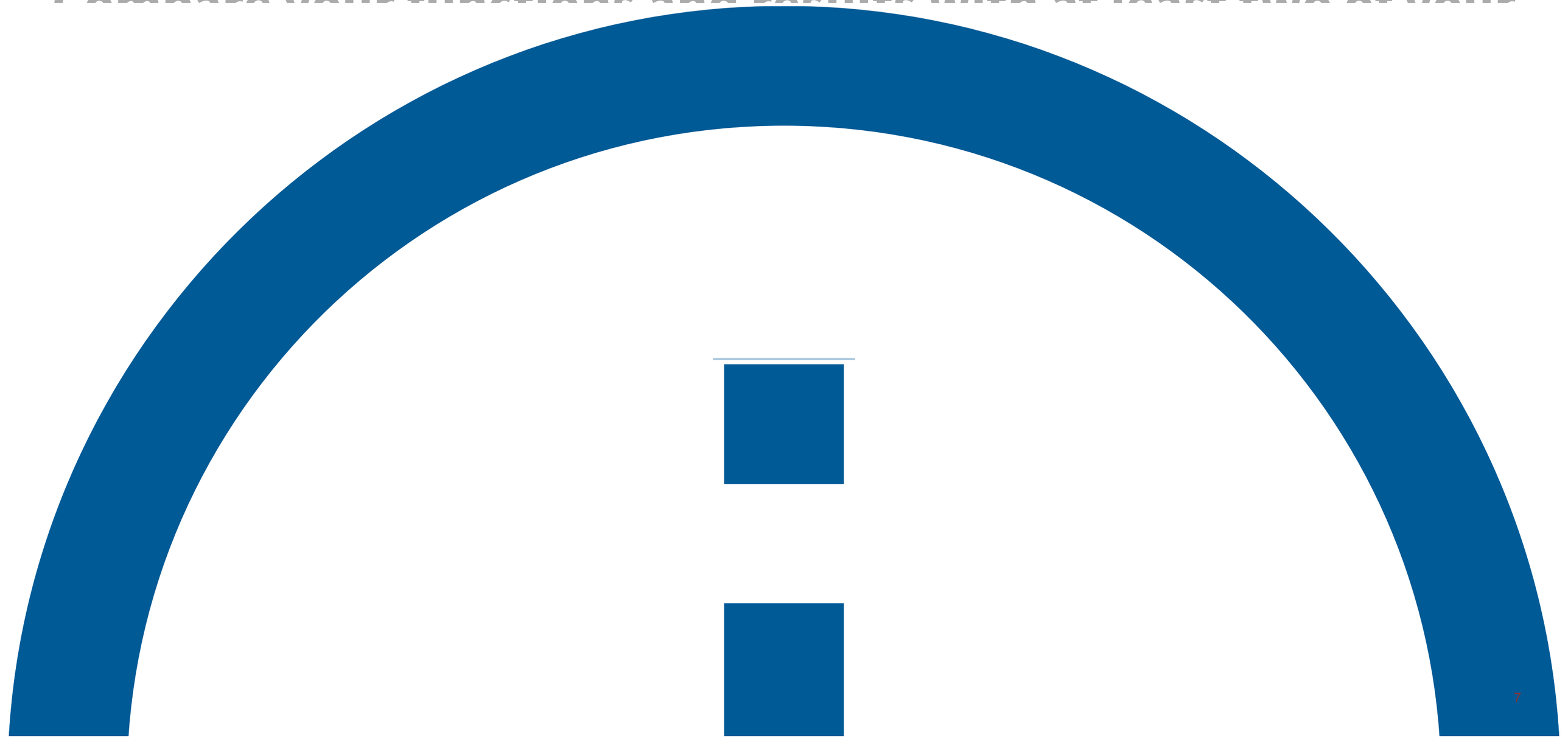
When poll is active, respond at **PollEv.com/dereneaton004**

# [3] Action: Write a function to get all 5-mers from the

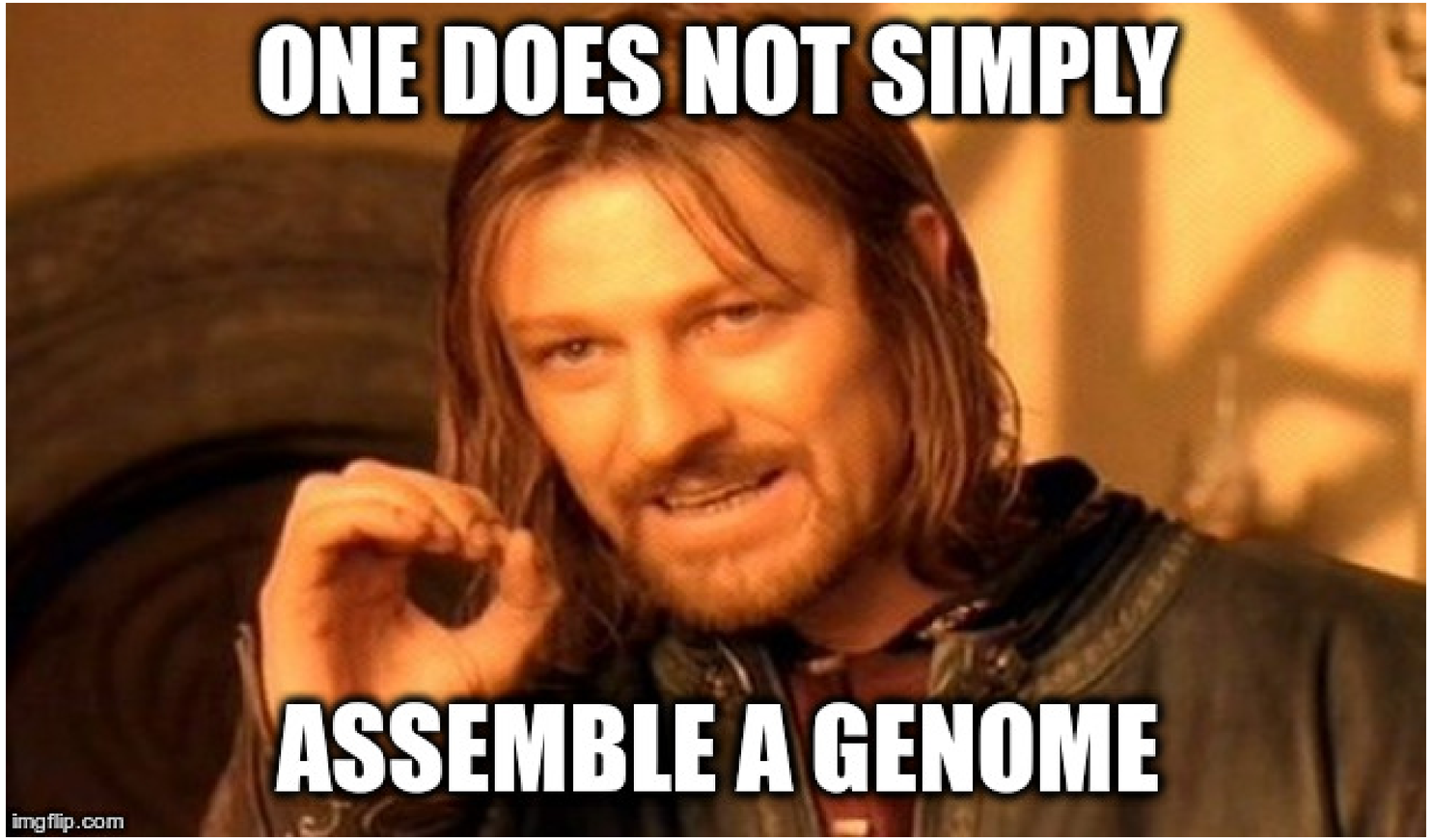


# [6,7,8] Use functions to accomplish the designated tasks...

Compare your functions and results with at least two of your

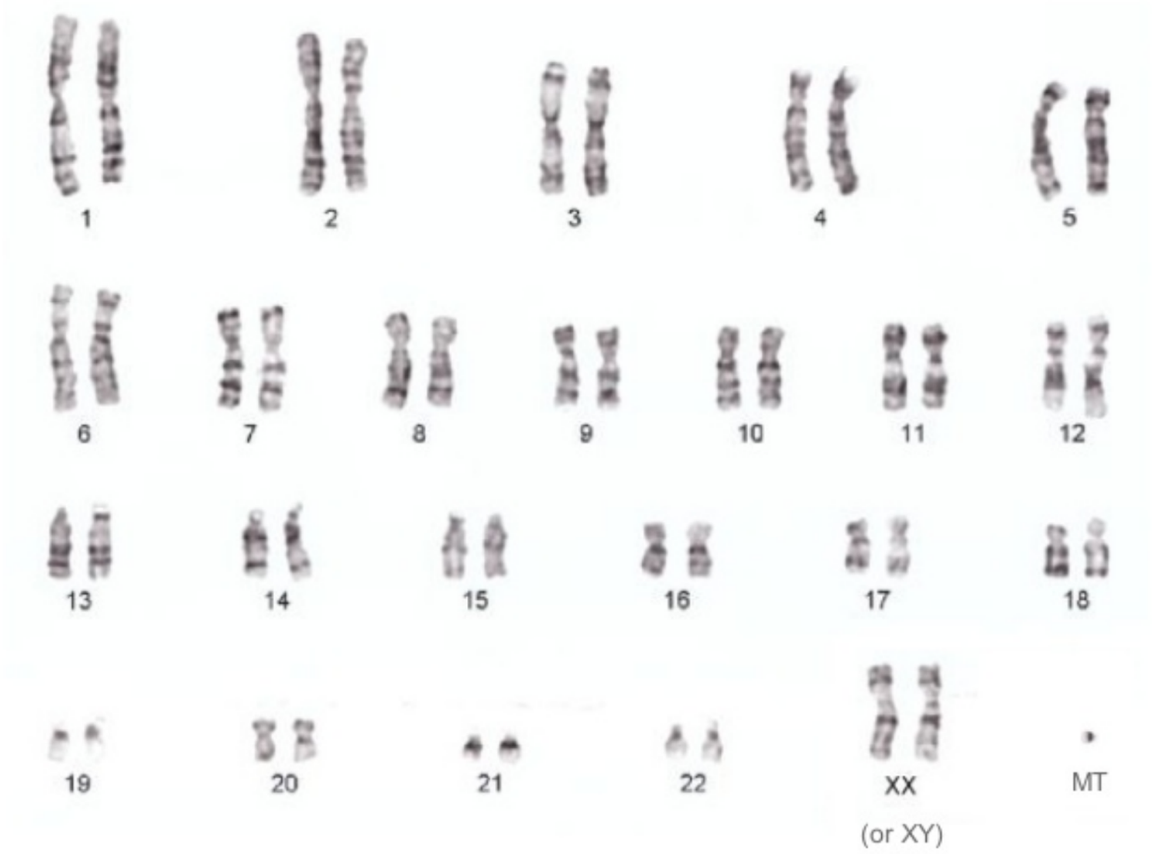


# Genome Assembly



# denovo Genome Assembly

denovo genome assembly is computationally demanding. Requires reads that cover the full genome many times (e.g., 50X). The end goal is to assemble scaffolds that match to chromosomes -- the real *bits* of the genome.

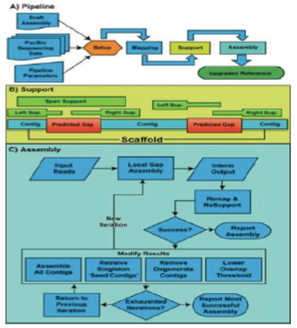


# Combining short and long-read technologies

Short read assemblies are highly fragmented. Long read technologies are highly error prone. Combining the two technologies -- while obtaining high-coverage of both -- is *currently* the gold standard.

## PacBio Assembly Algorithms

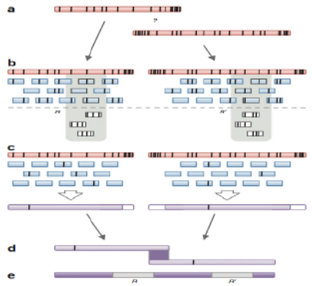
### PBJelly




**Gap Filling and Assembly Upgrade**

English *et al* (2012)
*PLOS One*. 7(11): e47768

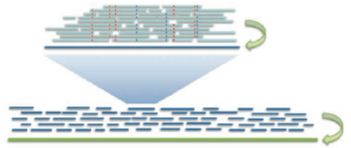
### PacBioToCA & ECTools




**Hybrid/PB-only Error Correction**

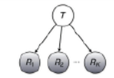
Koren, Schatz, *et al* (2012)
*Nature Biotechnology*. 30:693–700

### HGAP & Quiver



$\Pr(\mathbf{R} \mid T)$

$\Pr(\mathbf{R} \mid T) = \prod_L \Pr(R_L \mid T)$



**Quiver Performance Results**
*Comparison to Reference Genome*
*M. ruber* ; 3.1 MB ; SMRT® Cells)

| | Initial Assembly | Quiver Consensus |
|---|---|---|
| QV | 43.4 | 54.5 |
| Accuracy | 99.99540% | 99.99964% |
| Differences | 141 | 11 |

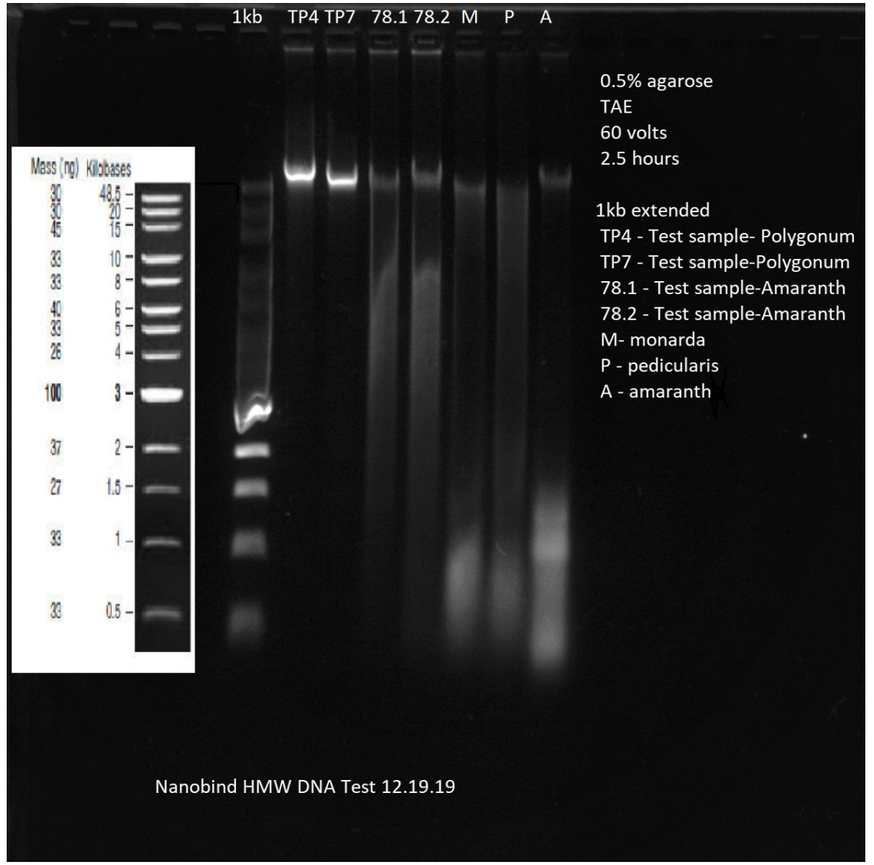
**PB-only Correction & Polishing**

Chin *et al* (2013)
*Nature Methods*. 10:563–569

< 5x PacBio Coverage > 50x

# Caveats: Long reads require HMW DNA, sometimes a lot.

Specialized DNA extraction kits and protocols are used to isolate long (unbroken) DNA fragment lengths. More expensive and time-consuming, but worth it.



# Eucalypus: (500Mb size, 170X ONT; 200X Illumina)

**The draft nuclear genome assembly of *Eucalyptus pauciflora*: a pipeline for comparing *de novo* assemblies**

Weiwen Wang ✉, Ashutosh Das, David Kainer, Miriam Schalamun, Alejandro Morales-Suarez, Benjamin Schwessinger, Robert Lanfear ✉ Author Notes

**Table 1:** Raw (before polish and haplotig removal) assembly statistics

| Assembly | Long-read^ | Short-read | Assembler | Assembly time (CPU hours)* | Length (bp) | contigs | Largest contig (bp) | N50 (bp) | L50 | GC (%) | Ns (%) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Canu_1kb | ≥1 kb (~174 ×) | X | Canu | ~300,000 | 871,577,052 | 2,867 | 7,123,373 | 629,835 | 259 | 39.18 | 0 |
| Canu_35kb | ≥35 kb (~40 ×) | X | Canu | ~50,000 | 825,916,527 | 2,550 | 10,153,603 | 962,598 | 158 | 39.18 | 0 |
| SMARTdenovo_1kb | ≥1 kb (~174 ×) | X | SMARTdenovo | ~8,000 | 610,858,639 | 729 | 6,287,341 | 1,711,661 | 107 | 39.29 | 0 |
| SMARTdenovo_35kb | ≥35 kb (~40 ×) | X | SMARTdenovo | ~4,000 | 586,903,502 | 704 | 9,494,401 | 1,868,532 | 91 | 39.27 | 0 |
| Flye_1kb | ≥1 kb (~174 ×) | X | Flye | ~700 | 596,007,484 | 5,930 | 2,755,662 | 255,434 | 652 | 39.12 | 0 |
| Flye_35kb | ≥35 kb (~40 ×) | X | Flye | ~500 | 561,349,738 | 4,145 | 2,407,003 | 352,050 | 448 | 39.17 | 0 |
| Marvel_35kb | ≥35 kb (~40 ×) | X | Marvel | ~28,000 | 649,061,435 | 1,181 | 6,453,759 | 795,971 | 182 | 39.07 | 0 |
| MaSuRCA_1kb | ≥1,kb (~174 ×) | ~228× | MaSuRCA | ~23,000 | 778,288,575 | 1,311 | 12,224,271 | 1,885,174 | 95 | 39.35 | 0.04 |
| MaSuRCA_35kb | ≥35 kb (~40 ×) | ~228× | MaSuRCA | ~21,000 | 773,035,614 | 1,703 | 8,684,546 | 1,304,720 | 146 | 39.39 | 0.09 |

^All long reads were corrected by Canu before assembly. The Canu correction step took ~200,000 CPU hours, which has not been included in the assembly runtime.
*With ~1 TB of RAM.

**Table 3:** The comparison of final assemblies

| Assembly | Length (bp) | Contig No. | Contig N50 (bp) | BUSCO score (1,440 genes in total) | | | | | | LAI score | Assembly ploidy | Short-read mapping | | Long-read mapping | | CGAL score | Structural variants |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Complete genes | | Duplicated genes | | Fragmented genes | | | | Mapping rate | Error rate | Mapping rate | Error rate | | |
| Canu_1kb | 622,218,742 | 895 | 1,502,325 | 1,346 | 93.47% | 183 | 12.71% | 23 | 1.60% | 7.04 | 1.24 | 96.02% | 0.0061 | 91.73% | 0.1661 | −1.959E+06 | 4,243 |
| Canu_35kb | 585,785,283 | 655 | 2,258,674 | 1,345 | 93.40% | 138 | 9.58% | 29 | 2.01% | 5.34 | 1.17 | 95.52% | 0.0066 | 92.64% | 0.1677 | −2.226E+06 | 5,043 |
| SMARTdenovo_1kb | 514,714,831 | 364 | 2,092,790 | 1,342 | 93.19% | 100 | 6.94% | 27 | 1.88% | 7.02 | 1.03 | **98.42%** | 0.0080 | 92.38% | 0.1678 | −4.275E+06 | 5,940 |
| SMARTdenovo_35kb | 504,515,539 | 370 | 2,178,079 | 1,341 | 93.13% | 100 | 6.94% | 30 | 2.08% | 6.73 | **1.01** | 98.35% | 0.0082 | 92.20% | 0.1679 | −5.869E+06 | 6,024 |
| Flye_1kb | 528,563,896 | 2947 | 295,613 | 1,344 | 93.33% | 100 | 6.94% | 31 | 2.15% | 5.70 | 1.06 | 94.86% | 0.0077 | **93.04%** | 0.1694 | −2.536E+06 | 7,137 |
| Flye_35kb | 516,992,152 | 2548 | 385,290 | 1,336 | 92.78% | **90** | **6.25%** | 31 | 2.15% | 6.50 | 1.03 | 94.24% | 0.0080 | 92.34% | 0.1699 | −2.726E+06 | 7,458 |
| Marvel_35kb | 537,615,613 | 730 | 1,202,845 | 1,180 | 81.94% | 153 | 10.63% | 32 | 2.22% | 3.77 | 1.08 | 87.37% | 0.0075 | 85.18% | 0.1689 | −4.451E+06 | 5,162 |
| MaSuRCA_1kb | 594,528,099 | 415 | 3,234,447 | **1,362** | **94.58%** | 201 | 13.96% | **21** | **1.46%** | 9.27 | 1.19 | 94.91% | **0.0060** | 91.57% | 0.1656 | **−1.774E+06** | 4,020 |
| MaSuRCA_35kb | 594,871,467 | 416 | **3,234,549** | **1,362** | **94.58%** | 200 | 13.89% | **21** | **1.46%** | **9.31** | 1.19 | 94.92% | **0.0060** | 91.49% | **0.1655** | −1.790E+06 | **4,017** |

Note: The best value of each assessment is highlighted in boldface.

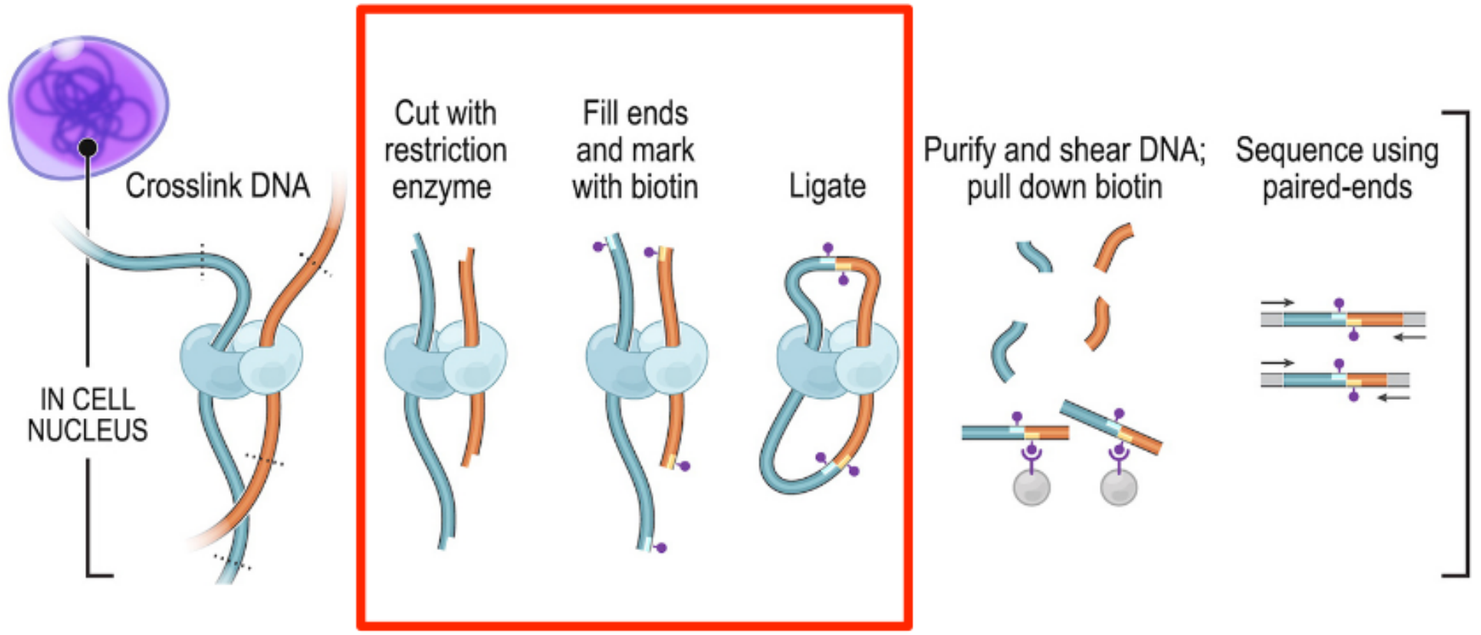
# Scaffolding: Hi-C Proximity Ligation

Chromosome conformation capture (3C) describes the structure of the genome within a cell; it's organization and structure. Better than microscopy, can tell us how close together (potentially interacting) some regions of the genome are (such as promoters and enhancers).

Hi-C: A highthroughput version of 3C is based a library preparation to build chimeric reads followed by short-read sequencing of paired-end reads. Creates a contact map of interactions correlated to spatial distance.

# Scaffolding: Hi-C Proximity Ligation

Restriction digestion; streptavidin bead extraction; paired-seq.



# Scaffolding: Amaranthus Hi-C Assembly