Genomics and Transcriptomics

Class 05 - Sequence Assembly

INSTRUCTOR:
Aureliano Bombarely
Department of Bioscience
Universita degli Studi di Milano
aureliano.bombarely@unimi.it

Outline of Topics

1. Brief history about genome assembly

2. Basics about sequence assembly

3. Whole genome assembly

    3.1. From reads to contigs and scaffolds.

    3.2. From scaffolds to chromosomes.

4. Transcriptome assembly

Outline of Topics

1. Brief history about genome assembly

2. Basics about sequence assembly

3. Whole genome assembly

   3.1. From reads to contigs and scaffolds.

   3.2. From scaffolds to chromosomes.

4. Transcriptome assembly

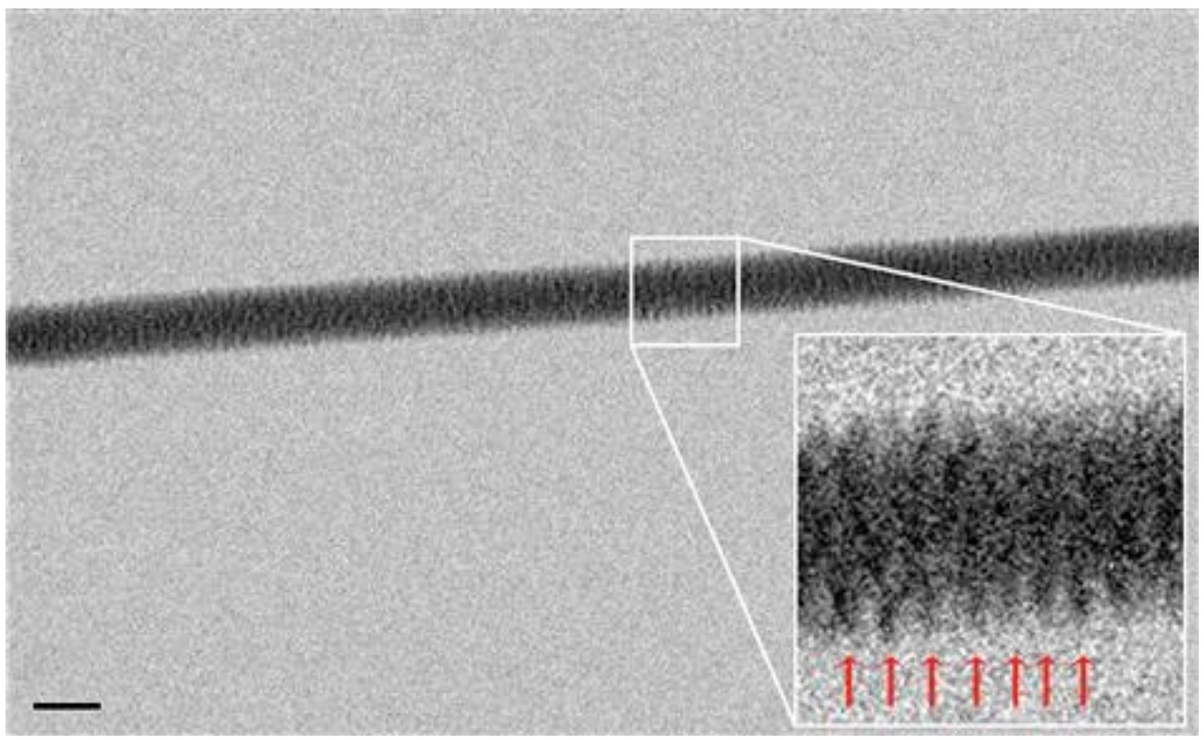
WHAT IS A GENOME?

Life is specified by **genomes**. Every organism, including humans, has a genome that contains all of the biological information needed to build and maintain a living example of that organism. The biological information contained in a genome is encoded in its **deoxyribonucleic acid (DNA)** and is divided into discrete units called **genes**. Genes code for proteins that attach to the genome at the appropriate positions and switch on a series of reactions called gene expression.

In 1909, Danish botanist Wilhelm Johanssen coined the word **gene** for the hereditary unit found on a chromosome. Nearly 50 years earlier, Gregor Mendel had characterized hereditary units as **factors**— observable differences that were passed from parent to offspring. Today we know that a single gene consists of a unique sequence of DNA that provides the complete instructions to make a functional product, called a protein. Genes instruct each cell type—such as skin, brain, and liver—to make discrete sets of proteins at just the right times, and it is through this specificity that unique organisms arise.

http://www.ncbi.nlm.nih.gov/About/primer/genetics_genome.html

Genome = N x Sequence of DNA



Gentile F. et al. *Direct Imaging of DNA Fibers: The Visage of Double Helix*
Nano Lett., 2012, 12 (12), pp 6453–6458

???

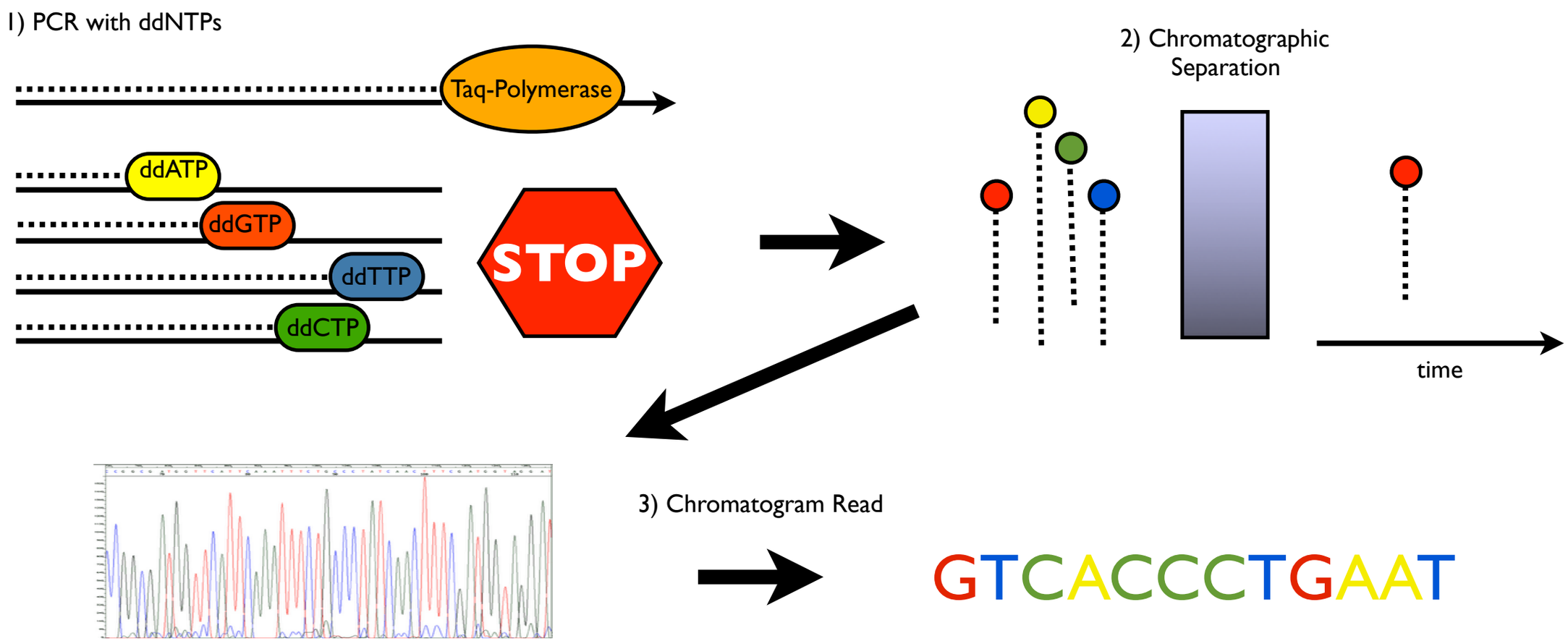
DNA sequencing

1. Brief history about genome assembly

DNA Sequencing:

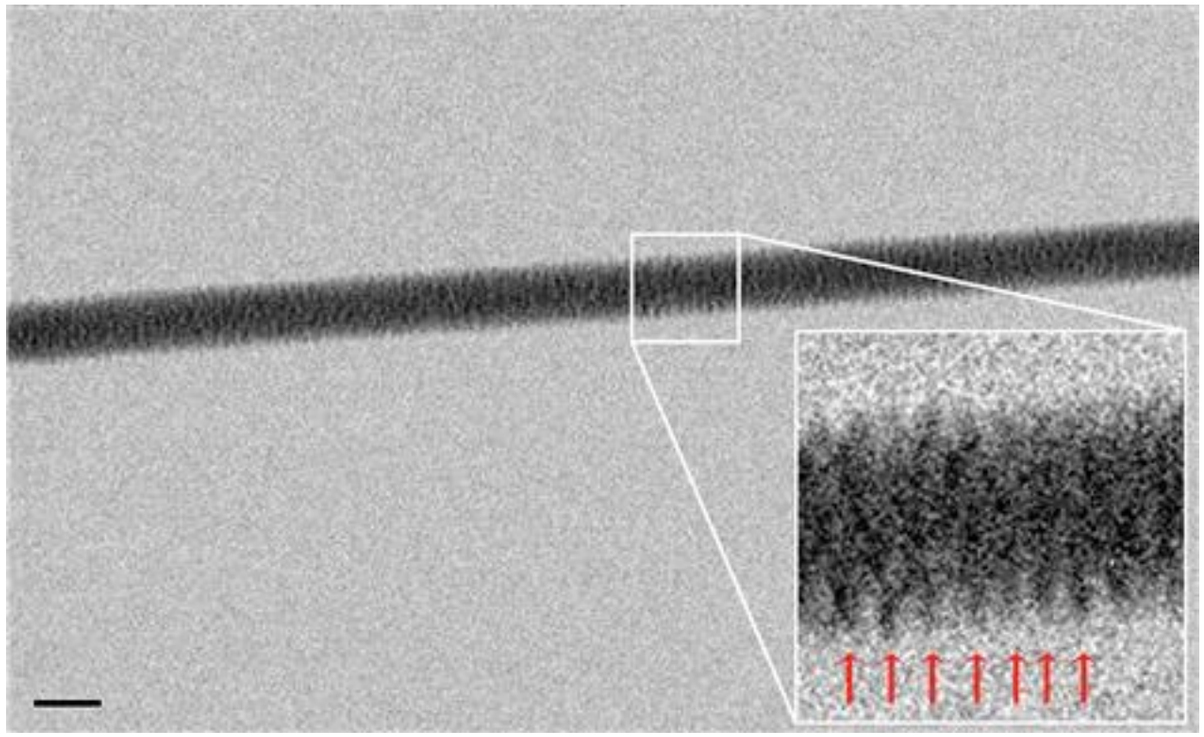
"Process of determining the precise order of nucleotides within a DNA molecule."

-Wikipedia



DNA Sanger Sequencing

1. Brief history about genome assembly



Gentile F. et al. *Direct Imaging of DNA Fibers: The Visage of Double Helix*
Nano Lett., 2012, 12 (12), pp 6453–6458

DNA sequencing

Fragments

ACCCCTGGGGGGGTTGTCGA

ACGCGTTTTGTTGTGG

ACCCCGACGTTGTCGA

ACGCGGTGACGTTGTCGA

ACGATTTAAATGACGTTGTCGA

ACGCGTTTTGTTGTGG

ACGCGTTTTGTTGTGG

ACGCGGTGACGTTGTCGA

Sequence assembly

ACGCGTTTTGTTGTGGTGGCCACACCACGCAGTGACGGAGATAACGGCGAGAGCATGGACGGAGGATGAGGATGG

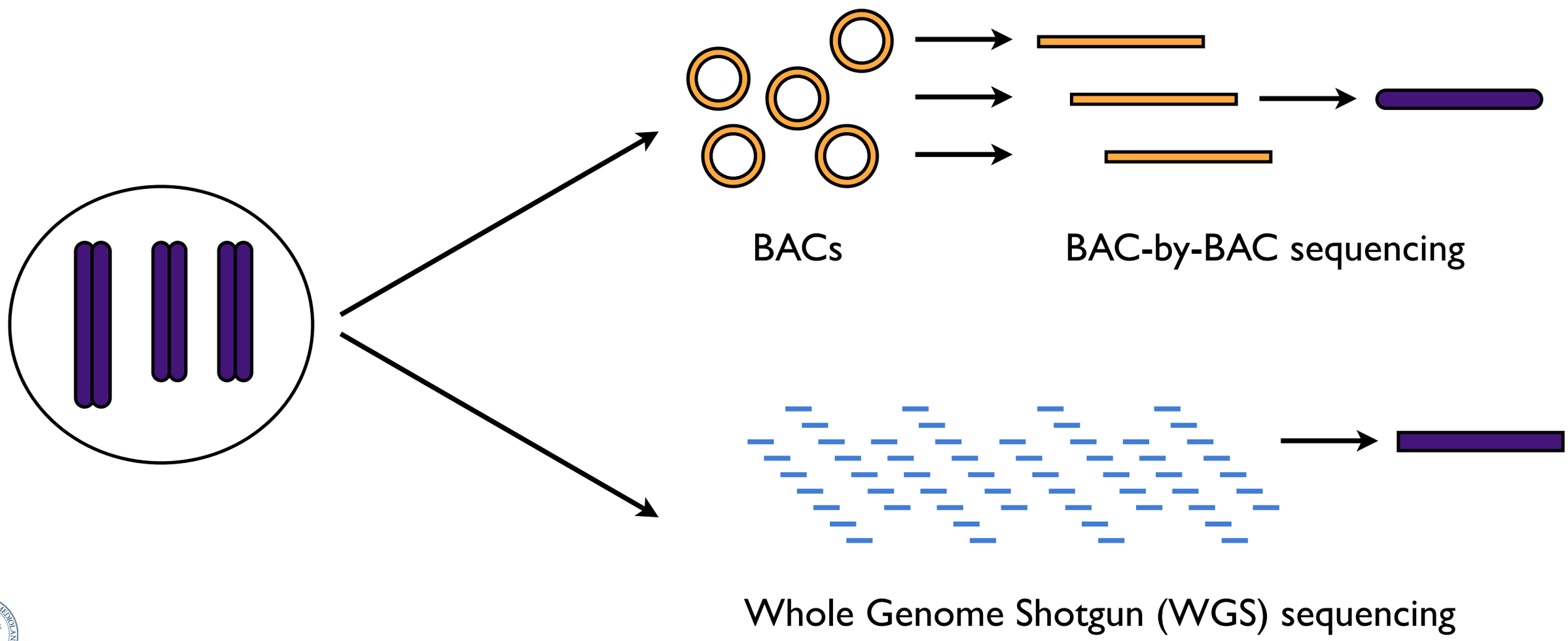
<u>Sequence assembly</u>

=

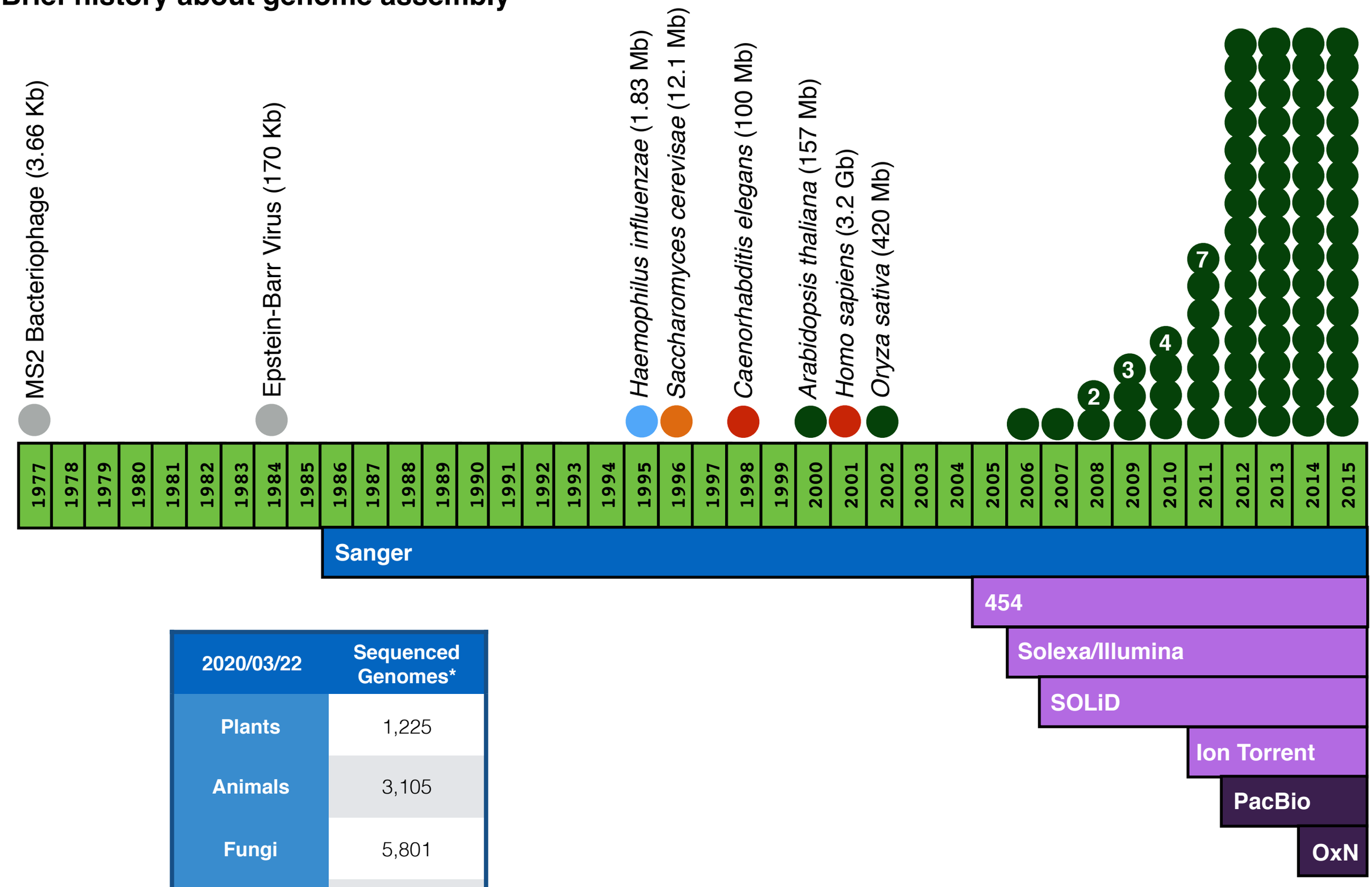
Resolve a puzzle and rebuild a DNA sequence from its pieces (fragments)

Image courtesy of iStock photo

Resolve a puzzle and rebuild a DNA sequence from its pieces (fragments)



1. Brief history about genome assembly



2020/03/22	Sequenced Genomes*
Plants	1,225
Animals	3,105
Fungi	5,801
Bacteria	244,944

https://www.ncbi.nlm.nih.gov/genome/browse#!/eukaryotes/

1. Brief history about genome assembly

MS2 Bacteriophage (3.658 Kb)	1977

Complete nucleotide sequence of bacteriophage MS2 RNA: primary and secondary structure of the replicase gene

W. Fiers, R. Contreras, F. Duerinck, G. Haegeman, D. Iserentant, J. Merregaert, W. Min Jou, F. Molemans, A. Raeymaekers, A. Van den Berghe, G. Volckaert & M. Ysebaert

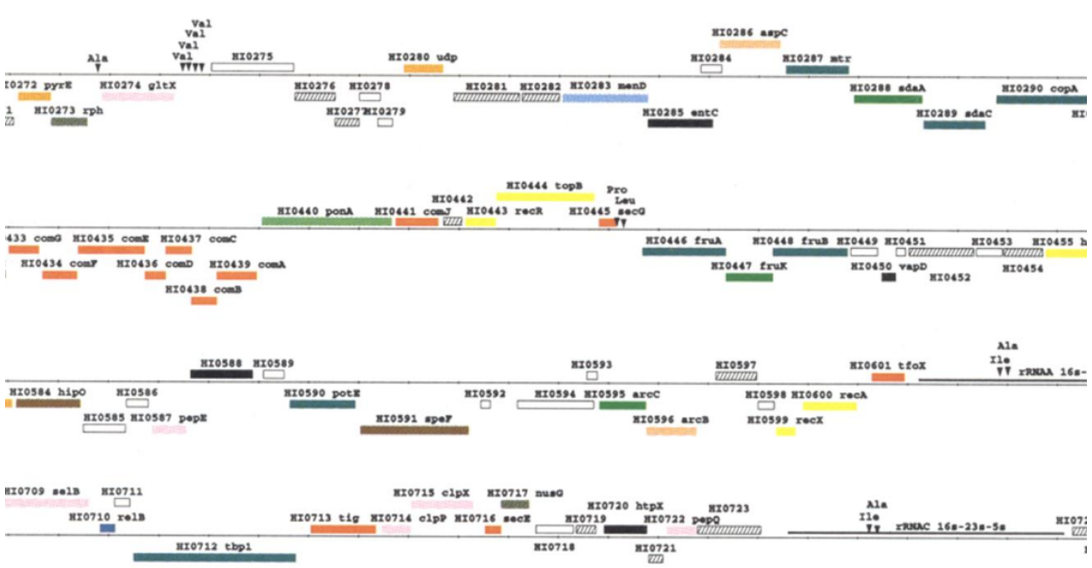
Laboratory of Molecular Biology, University of Ghent, 9000 Ghent, Belgium

1. Brief history about genome assembly

Haemophilus influenzae (1.83 Mb)	1995

Whole-Genome Random Sequencing and Assembly of *Haemophilus influenzae* Rd

Robert D. Fleischmann, Mark D. Adams, Owen White, Rebecca A. Clayton, Ewen F. Kirkness, Anthony R. Kerlavage, Carol J. Bult, Jean-Francois Tomb, Brian A. Dougherty, Joseph M. Merrick, Keith McKenney, Granger Sutton, Will FitzHugh, Chris Fields,* Jeannine D. Gocayne, John Scott, Robert Shirley, Li-Ing Liu, Anna Glodek, Jenny M. Kelley, Janice F. Weidman, Cheryl A. Phillips, Tracy Spriggs, Eva Hedblom, Matthew D. Cotton, Teresa R. Utterback, Michael C. Hanna, David T. Nguyen, Deborah M. Saudek, Rhonda C. Brandon, Leah D. Fine, Janice L. Fritchman, Joyce L. Fuhrmann, N. S. M. Geoghagen, Cheryl L. Gnehm, Lisa A. McDonald, Keith V. Small, Claire M. Fraser, Hamilton O. Smith, J. Craig Venter†

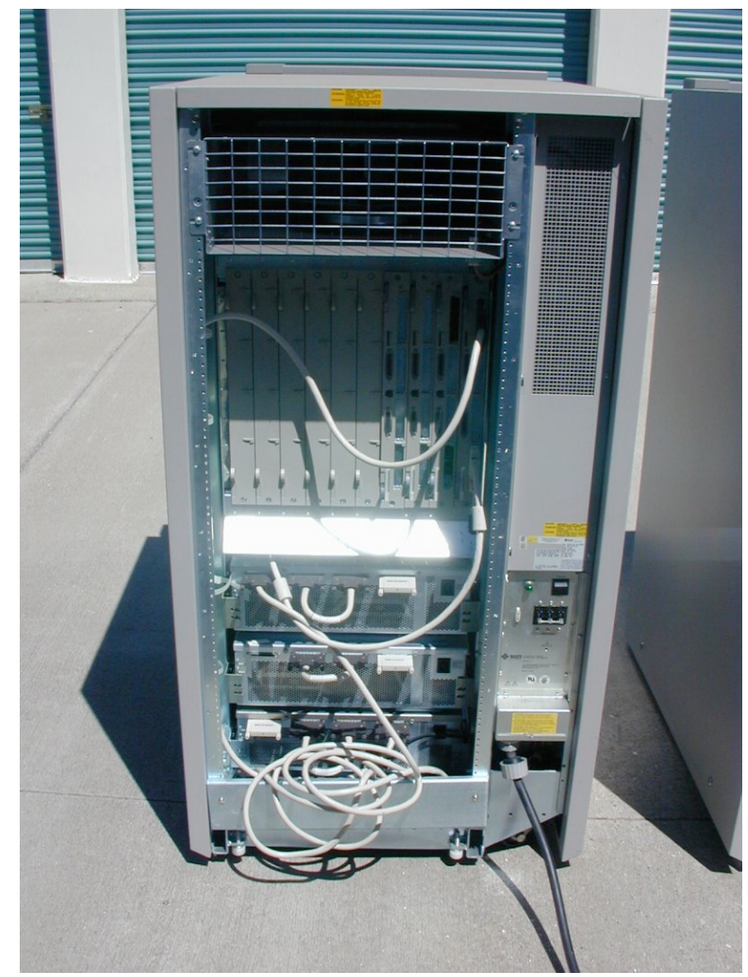


Software:

TIGR ASSEMBLER

Hardware:

SPARCenter 2000 (512 Mb RAM)



1. Brief history about genome assembly

Haemophilus influenzae (1.83 Mb)	1995

Software:

TIGR ASSEMBLER → Smith-Waterman alignments

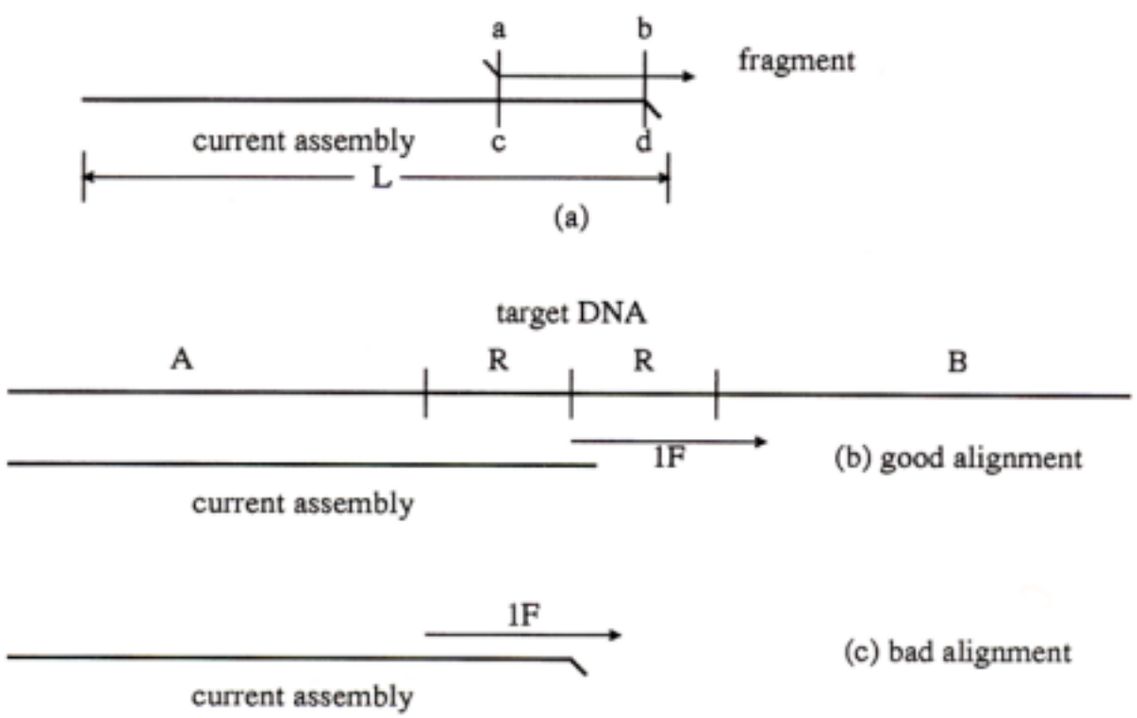


FIG. 1. **a.** A graphic representation of a Smith-Waterman alignment is shown. The overlap is shown between vertical bars, and the diagonal lines represent overhangs where the sequences do not match. The current assembly has a length of L. **b.** A tandem repeat of two copies of repeat region R is shown along with the proper alignment of fragment 1F with the current assembly. **c.** A bad alignment of the current assembly and fragment 1F is produced if the overlap is maximized without regard to the length of the overhang. The bad alignment can result in two outcomes. If the overhang is short, it will be ignored, and the two repeat regions will be compressed into a single region. If the overhang is long, the merge will not be allowed, and the current assembly will not be extended.

OVERLAPPING METHODOLOGY

Sutton GG. et al. TIGR Assembler: A New Tool to Assembly Large Shotgun Sequencing Projects Genome Science and Technology, 1995, 1:9-19

1. Brief history about genome assembly

Homo sapiens (3.2 Gb)	2001

BAC-by-BAC sequencing

articles

Initial sequencing and analysis of the human genome

International Human Genome Sequencing Consortium*

* *A partial list of authors appears on the opposite page. Affiliations are listed at the end of the paper.*

International Human Genome Sequencing Consortium. Initial Sequencing and Analysis of the Human Genome. Nature. 2001. 409:860-921

Whole Genome Shotgun (WGS) sequencing

OVERLAPPING METHODOLOGY

The Sequence of the Human Genome

J. Craig Venter,[1]* Mark D. Adams,[1] Eugene W. Myers,[1] Peter W. Li,[1] Richard J. Mural,[1]
Granger G. Sutton,[1] Hamilton O. Smith,[1] Mark Yandell,[1] Cheryl A. Evans,[1] Robert A. Holt,[1]

Venter JC. et al. The Sequence of the Human Genome. Science. 2001. 291:1304-1351

1. Brief history about genome assembly

Homo sapiens (3.2 Gb)	2001

Whole Genome Shotgun (WGS) sequencing

THE HUMAN GENOME

The Sequence of the Human Genome

J. Craig Venter,[1*] Mark D. Adams,[1] Eugene W. Myers,[1] Peter W. Li,[1] Richard J. Mural,[1]
Granger G. Sutton,[1] Hamilton O. Smith,[1] Mark Yandell,[1] Cheryl A. Evans,[1] Robert A. Holt,[1]

Venter JC. et al. The Sequence of the Human Genome. Science. 2001. 291:1304-1351

Software:

WGA ASSEMBLER (CABOG)

Hardware:

40 machines AlphaSMPs (4 Gb RAM/each and 4 cores/each, total=160 Gb RAM and 160 cores); 5 days.

1. Brief history about genome assembly

Ailuropoda melanoleura (2.3 Gb)	2009

Whole Genome Shotgun (WGS) sequencing

ARTICLES

The sequence and *de novo* assembly of the giant panda genome

Ruiqiang Li[1,2]*, Wei Fan[1]*, Geng Tian[1,3]*, Hongmei Zhu[1]*, Lin He[4,5]*, Jing Cai[3,6]*, Quanfei Huang[1], Qingle Cai[1,7], Bo Li[1], Yinqi Bai[1], Zhihe Zhang[8], Yaping Zhang[6], Wen Wang[6], Jun Li[1], Fuwen Wei[9], Heng Li[10], Min Jian[1], Jianwen Li[1],

Li R. et al. The Sequence and the Novo Assembly of the Giant Panda Genome. Nature. 2009. 463:311-317

BRUIJN GRAPHS METHODOLOGY

Software:

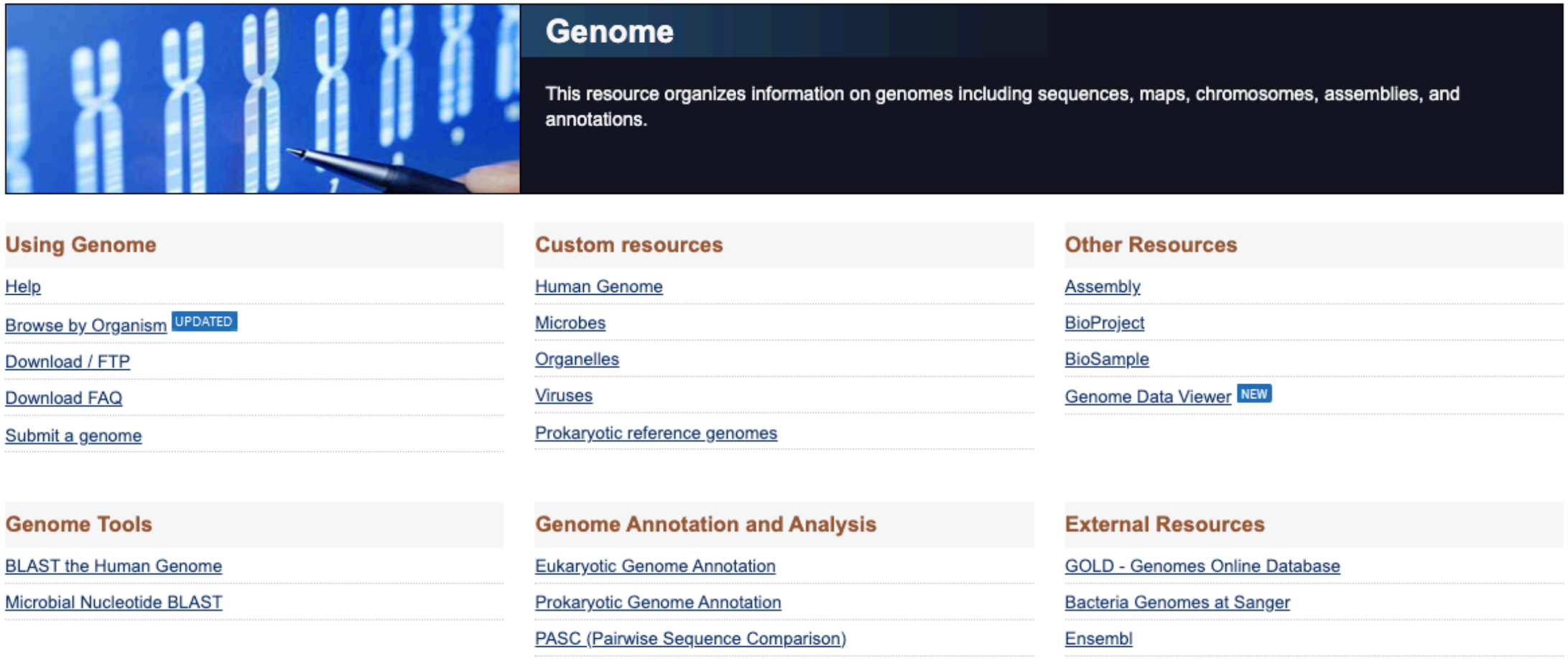
SOAPdenovo

Hardware:

Supercomputer with 32 cores and 512 Gb RAM.

1. Brief history about genome assembly

https://www.ncbi.nlm.nih.gov/genome/



https://www.ncbi.nlm.nih.gov/genome/

1. Brief history about genome assembly

https://www.yourgenome.org/facts/timeline-history-of-genomics

In: FACTS Methods and Technology

Timeline: History of genomics

A timeline depicting the key events in the history of genomics and genetic research alongside those in popular culture. From the discovery of DNA, and the election of Roosevelt, right through to whole genome sequencing and Andy Murray winning Wimbledon for the first time.

Outline of Topics

1. Brief history about genome assembly

2. Basics about sequence assembly

3. Whole genome assembly

   3.1. From reads to contigs and scaffolds.

   3.2. From scaffolds to chromosomes.

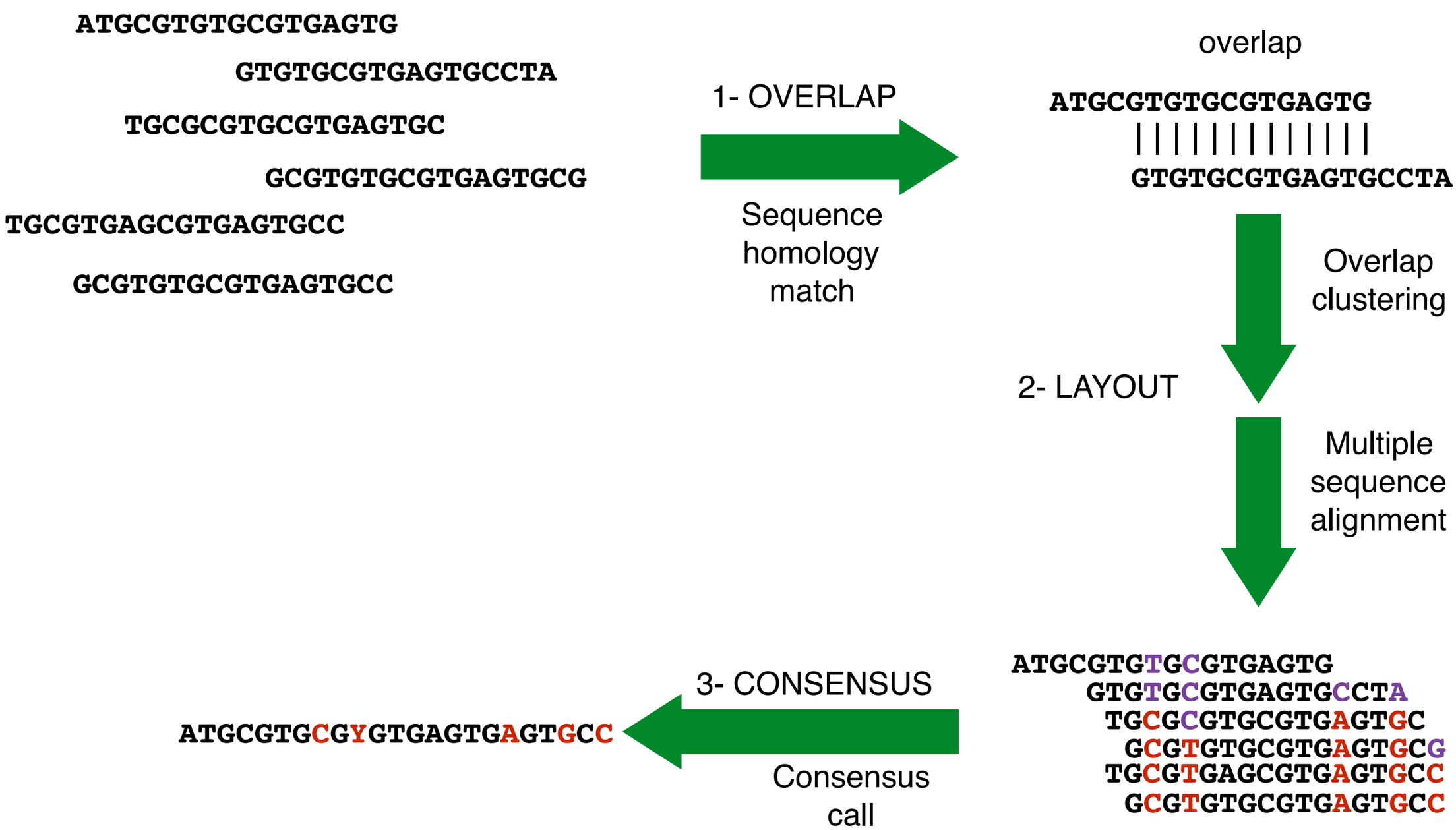
4. Transcriptome assembly

There are two sequence assembly main approaches:

- Overlap-Layout-Consensus (OLC) based in sequence alignments.

- De-Bruijn Graphs (DBG) based in Kmer decomposition and De-Bruijn graphs.

Overlap-Layout-Consensus (OLC)



Overlap-Layout-Consensus (OLC)

1- Finding the Overlap

```
Seq_x:  ATGCGTGTGCGTGAGTG
          ↓
 Pattern of length L=4:
          AGTG
          ↓ Search Pattern
Seq_y:  GTGTGCGTGAGTGCCTA
```

→

```
Seq_x:  ATGCGTGTGCGTGAGTG
          ↓
    Pattern extension
          CGTG
          ↓ Pattern extension
Seq_y:  GTGTGCGTGAGTGCCTA
```

… and repeat with each pair

2 reads — 2 comparisons
4 reads — 16 comparisons
10 reads — 100 comparisons
1000000 reads — 1000000000000 comparisons
…

Overlap-Layout-Consensus (OLC)

1- Finding the Overlap

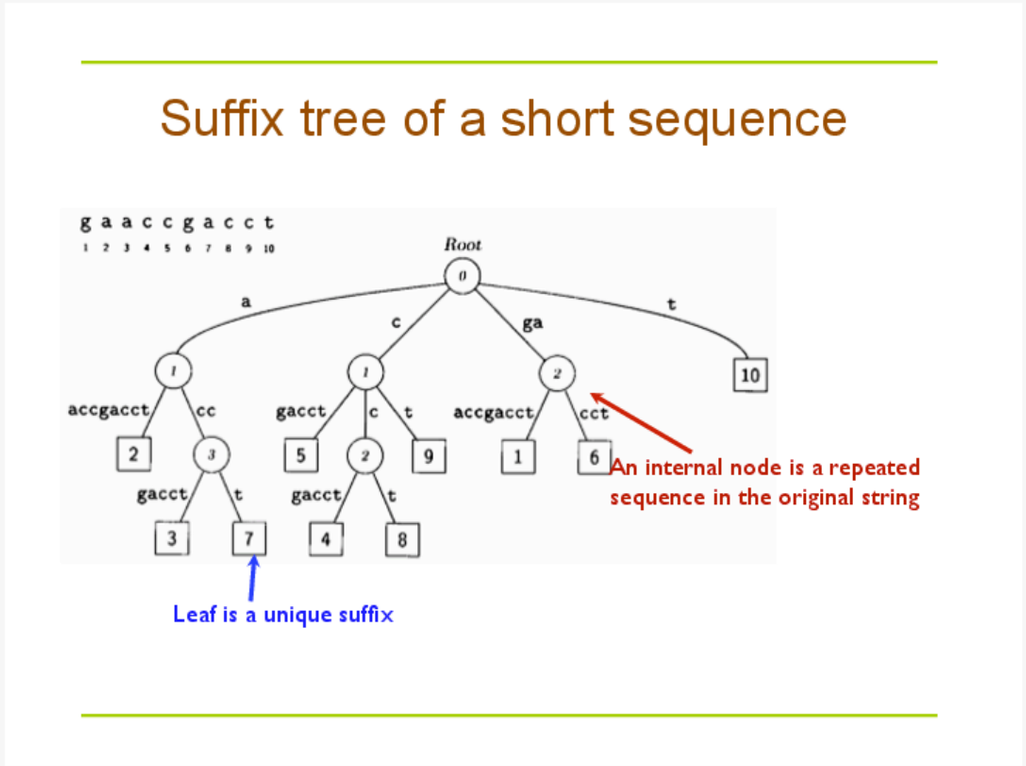
1.1- Suffix tree

Suffix tree allows one to find, extremely efficiently, all distinct subsequences in a given sequence.

Suffix tree and suffix array for string matching

Suffix tree

- Definition: a suffix tree is a compressed trie containing all the suffixes of the given text as their keys and positions in the text as their values. See an example of suffix tree:



http://homes.sice.indiana.edu/yye/lab/teaching/spring2016-C343/suffix.php

Overlap-Layout-Consensus (OLC)

1- Finding the Overlap

1.2- Dynamic Programming: Smith Waterman Algorithms

The Smith–Waterman algorithm performs local sequence alignment; that is, for determining **similar regions between two strings of nucleic acid sequences or protein sequences**. Instead of looking at the entire sequence, the Smith–Waterman algorithm compares segments of all possible lengths and optimizes the similarity measure.

The Smith–Waterman algorithm has four steps:

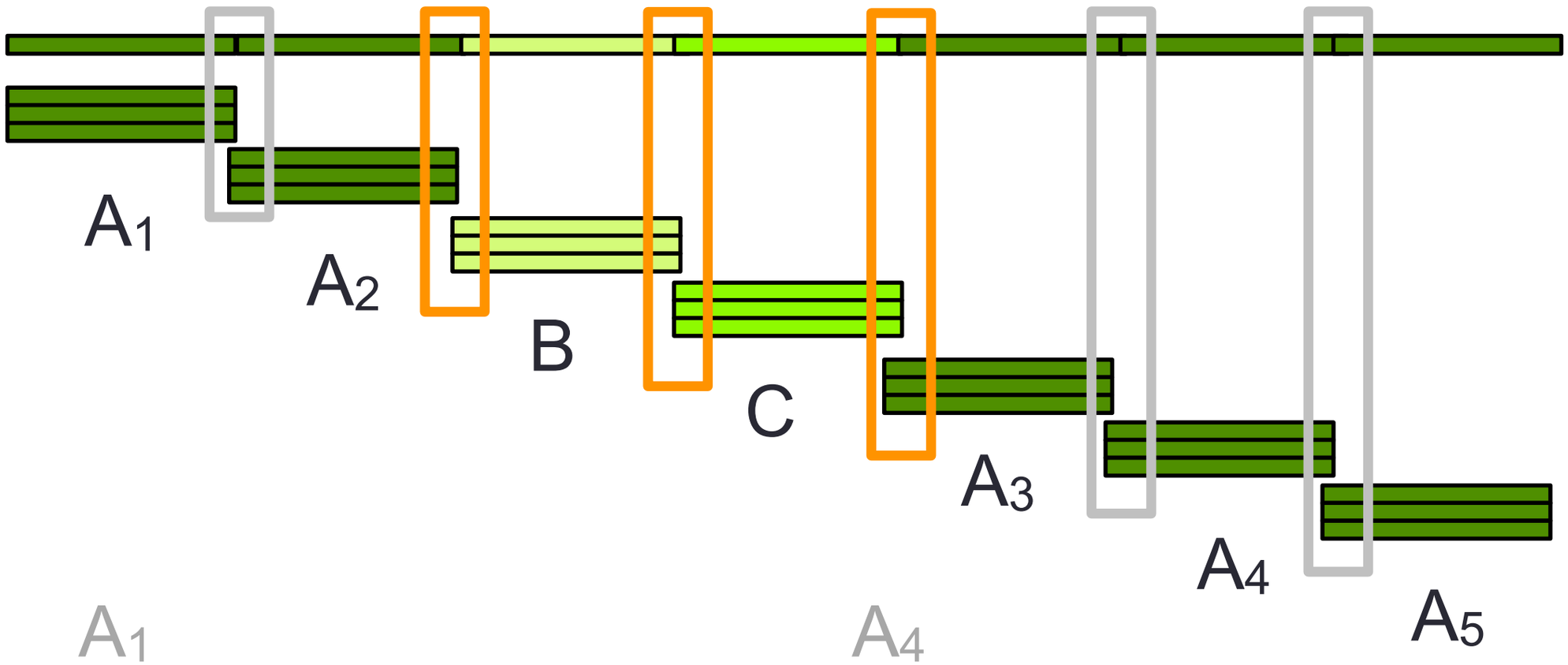
1. Determine the substitution matrix and the gap penalty scheme.
2. Initialize the scoring matrix.
3. Scoring.
4. Traceback.

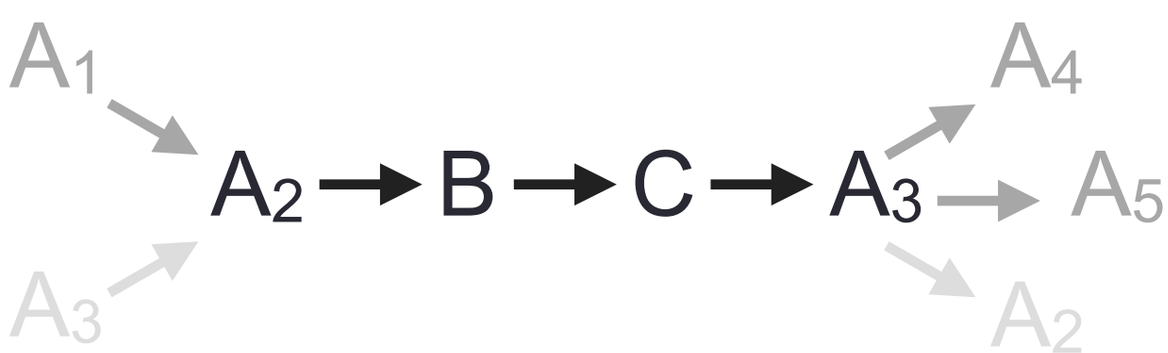
The Smith–Waterman algorithm example: https://www.youtube.com/watch?v=QphFHG9tmOY

https://en.wikipedia.org/wiki/Smith%E2%80%93Waterman_algorithm

Overlap-Layout-Consensus (OLC)

2- Resolving the layout





Overlap-Layout-Consensus (OLC)

3- Calling the consensus

```
ATGCGTGTGCGTGAGTG
     GTGTGCGTGAGTGCCTA
      TGCGCGTGCGTGAGTGC
      TGCGTGTGCGTGAGTGCC
       GCGTGTGCGTGAGTGCG
       GCGTGTGCGTGAGTGCC
```

4 x C 2 x T	3 x C 3 x T	**4 x A** 1 x C	**4 x G** 1 x A	1 x G **2 x C**

Consensus by majority vote

ATGCGTGCGYGTGAGTGAGTGCC

De-Bruijn Graphs (DBG)

What is a Kmer ?

Specific n-tuple or n-gram of nucleic acid or amino acid sequences.

-Wikipedia

ordered list of elements

contiguous sequence of *n* items from a given sequence of text

ATGCGCAGTGGAGAGAGAGCGATG Sequence A with 25 nt

↓ 5 Kmers of 20-mer

ATGCGCAGTGGAGAGAGAGC
TGCGCAGTGGAGAGAGAGCG
GCGCAGTGGAGAGAGAGCGA
CGCAGTGGAGAGAGAGCGAT
GCAGTGGAGAGAGAGCGATG

N_kmers = L_read - Kmer_size

2. Basics about sequence assembly

De-Bruijn Graphs (DBG)

ATGCGTGTGCGTGAGT
GTGTGCGTGAGTGCCT
TGCGTGTGAGTGAGTGC
GCGTGTGAGTGAGTGCG

Kmer transformation (e.g. Kmer-5)

ATGCG
TGCGT
GCGTG
CGTGT
GTGTG
TGTGC
GTGCG
TGCGT
GCGTG
CGTGA
GTGAG
TGAGT

GTGTG
TGTGC
GTGCG
TGCGT
GCGTG
CGTGA
GTGAG
TGAGT
GAGTG
AGTGC
GTGCC
TGCCT

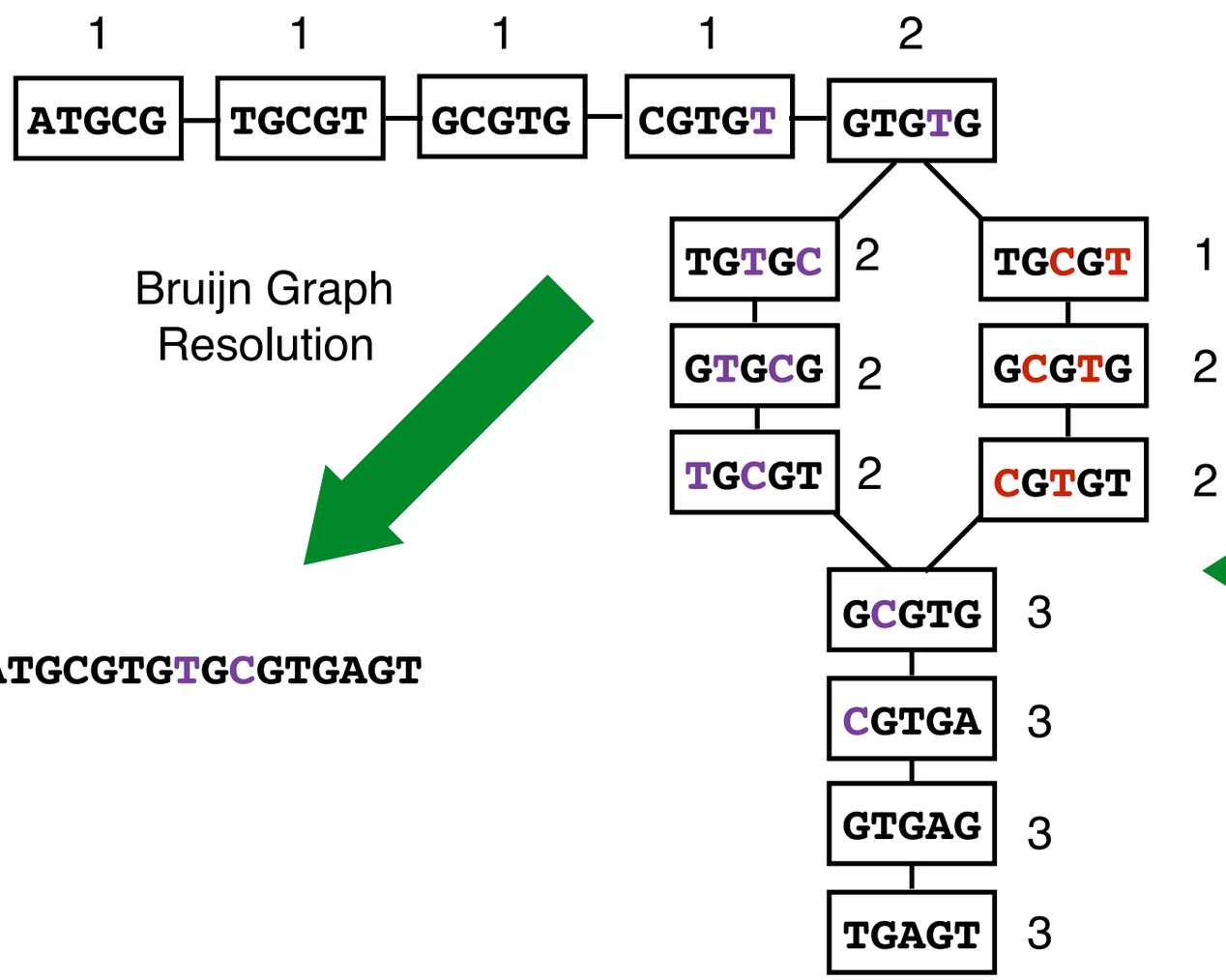
TGCGT
GCGTG
CGTGT
GTGTG
TGTGA
GTGAG
TGAGT
GAGTG
AGTGA
GTGAG
TGAGT
GAGTG
AGTGC

GCGTG
CGTGT
GTGTG
TGTGA
GTGAG
TGAGT
GAGTG
AGTGA
GTGAG
TGAGT
GAGTG
AGTGC
GTGCG

Kmer count

Kmer	Count
ATGCG	1
TGCGT	1
GCGTG	1
CGTGT	1
GTGTG	2
TGTGC	2
GTGCG	2
TGCGT	2
GCGTG	3
CGTGA	3
GTGAG	3
TGAGT	3
TGCGT	1
GCGTG	2

Bruijn Graph Construction



Bruijn Graph Resolution

ATGCGTGTGCGTGAGT

De-Bruijn Graphs (DBG)

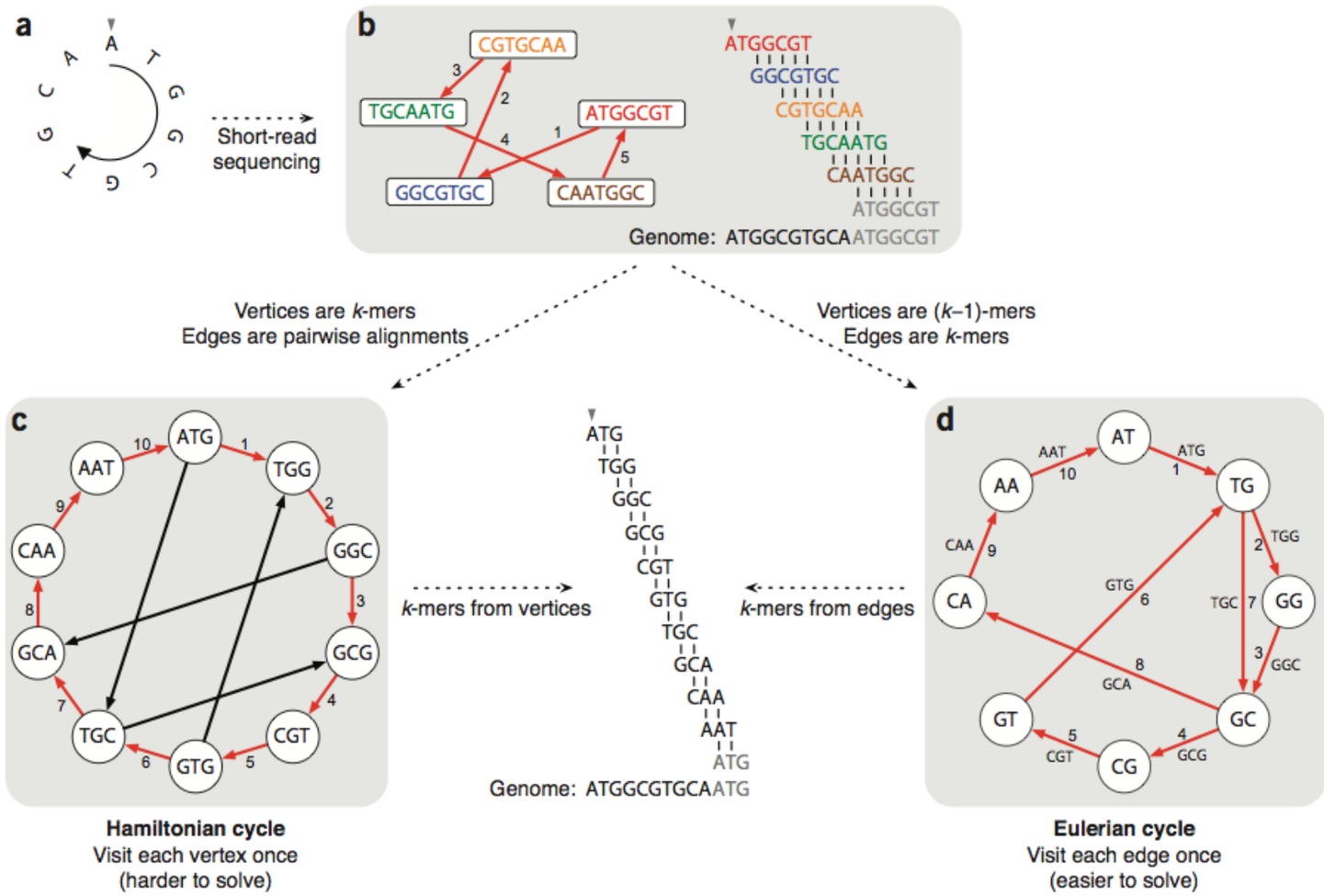


Figure 3 Two strategies for genome assembly: from Hamiltonian cycles to Eulerian cycles. (**a**) An example small circular genome. (**b**) In traditional Sanger sequencing algorithms, reads were represented as nodes in a graph, and edges represented alignments between reads. Walking along a Hamiltonian cycle by following the edges in numerical order allows one to reconstruct the circular genome by combining alignments between successive reads. At the end of the cycle, the sequence wraps around to the start of the genome. The repeated part of the sequence is grayed out in the alignment diagram. (**c**) An alternative assembly technique first splits reads into all possible *k*-mers: with $k = 3$, ATGGCGT comprises ATG, TGG, GGC, GCG and CGT. Following a Hamiltonian cycle (indicated by red edges) allows one to reconstruct the genome by forming an alignment in which each successive *k*-mer (from successive nodes) is shifted by one position. This procedure recovers the genome but does not scale well to large graphs. (**d**) Modern short-read assembly algorithms construct a de Bruijn graph by representing all *k*-mer prefixes and suffixes as nodes and then drawing edges that represent *k*-mers having a particular prefix and suffix. For example, the *k*-mer edge ATG has prefix AT and suffix TG. Finding an Eulerian cycle allows one to reconstruct the genome by forming an alignment in which each successive *k*-mer (from successive edges) is shifted by one position. This generates the same cyclic genome sequence without performing the computationally expensive task of finding a Hamiltonian cycle.

Compeau PEC. et al. How to apply de Bruijn graphs to genome assembly. Nature Biotech. 2011. 29:287-291

BRIEFINGS IN FUNCTIONAL GENOMICS. VOL 11. NO 1. 25–37 doi:10.1093/bfgp/elr035

Comparison of the two major classes of assembly algorithms: overlap–layout–consensus and de-bruijn-graph

Zhenyu Li, Yanxiang Chen*, Desheng Mu*, Jianying Yuan, Yujian Shi, Hao Zhang, Jun Gan, Nan Li, Xuesong Hu, Binghang Liu, Bicheng Yang and Wei Fan*

Advance Access publication date 19 December 2011

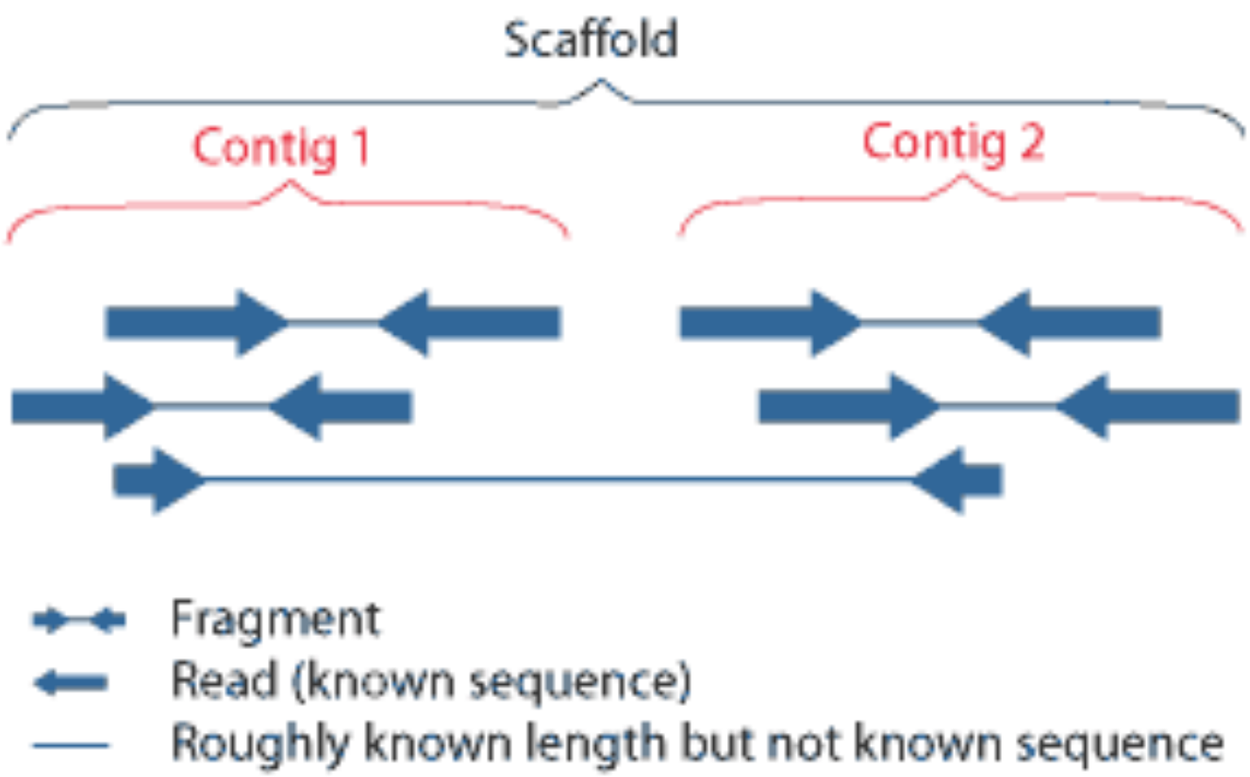
OLC (Overlap-layout-consensus) algorithm is more suitable for the low-coverage long reads, whereas the DBG (De-Bruijn-Graph) algorithm is more suitable for high-coverage short reads and especially for large genome assembly

Key Points

- High-quality genome sequences for many species are still strongly desired by the genomics community. With the rapid development of sequencing technologies and assembly algorithms, we have seen practical improvements and a bright future lies ahead.
- There are two major types of assembly algorithms: OLC and DBG; both of them are in accordance with Lander–Waterman model, but suit the assembly of different read lengths and sequencing depths, and have significant differences in computational efficiency.
- How well a genome can be assembled depends not only on sequencing technologies such as read length and sequencing error rate, but also on the characteristics of the genome, including repeat and the heterozygosity rate of the sequenced sample.

What is a Scaffold?

A scaffold is a portion of the genome sequence reconstructed from end-sequenced whole-genome shotgun clones. Scaffolds are composed of contigs and gaps. A contig is a contiguous length of genomic sequence in which the order of bases is known to a high confidence level. Gaps occur where reads from the two sequenced ends of at least one fragment overlap with other reads in two different contigs (as long as the arrangement is otherwise consistent with the contigs being adjacent). Since the lengths of the fragments are roughly known, the number of bases between contigs can be estimated.

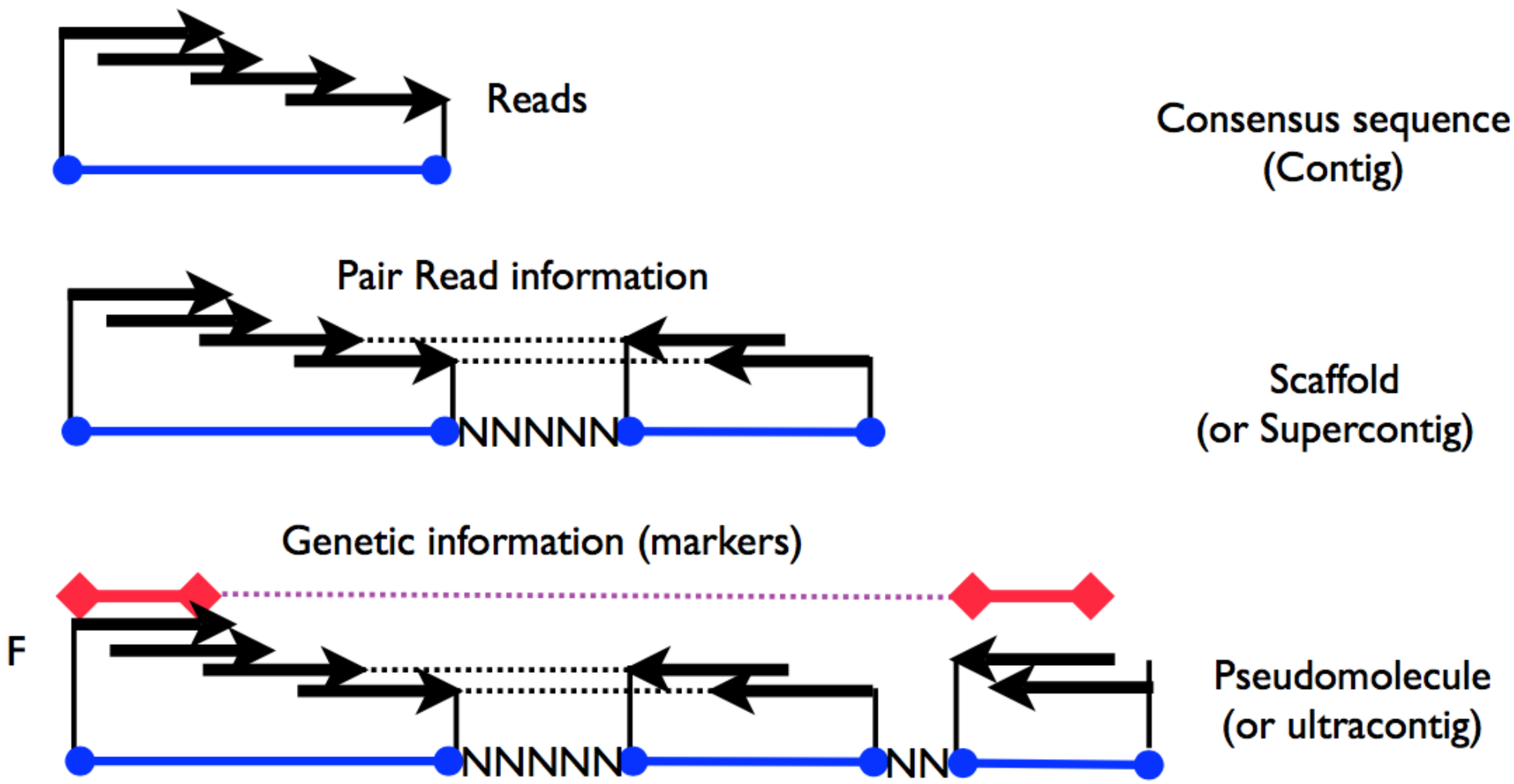


https://genome.jgi.doe.gov/help/scaffolds.jsf

What is a Scaffold?

- Why is important the pair information ?

  - *novo* assembly:



Outline of Topics

1. Brief history about genome assembly

2. Basics about sequence assembly

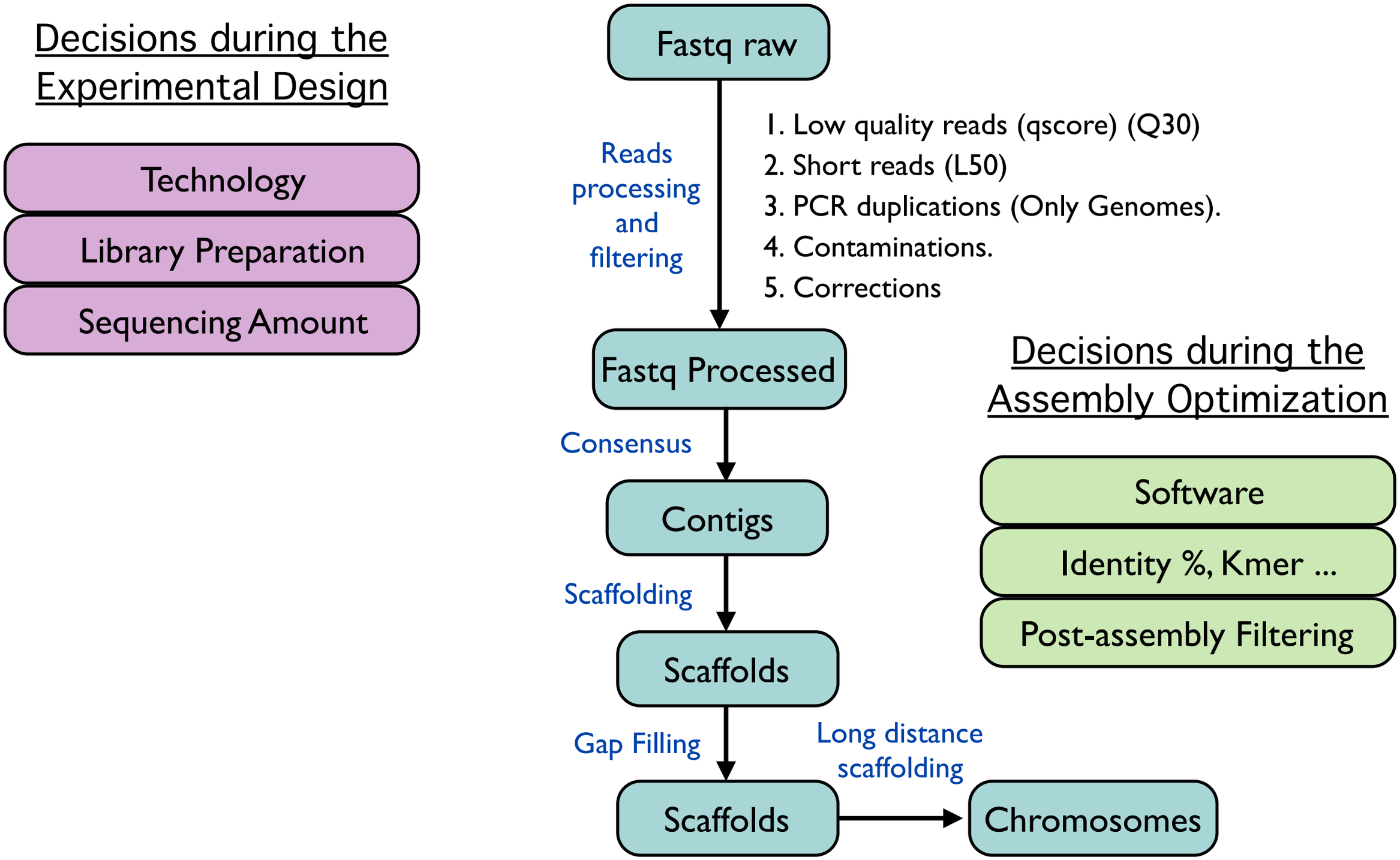
3. Whole genome assembly

   3.1. From reads to contigs and scaffolds.

   3.2. From scaffolds to chromosomes.

4. Transcriptome assembly

3. Whole genome assembly



Decisions during the Experimental Design

- Technology
- Library Preparation
- Sequencing Amount

3. Whole genome assembly

Technologies

Technology	Read length (bp)	Accuracy	Reads/Run	Time/Run	Cost/Mb
Applied Bio 3730XL (Sanger)	400 - 900	99.9%	384	4 h (12 runs/day)	US$2,400
Roche 454 GS FLX (Pyrosequencing)	700 Single/Pairs	99.9%	1,000,000	24h	US$10
Illumina HiSeq2500 (Seq. by synthesis)	50-250 Single/Pairs	99%	4,000,000,000	24 to 120 h	$0.05 to $0.15
Ilumina MiSeq (Seq. by synthesis)	50-300 Single/Pairs	99%	44,000,000	24 to 72 h	US$0.17
SOLiD 4 (Seq. by ligation)	25-50 Single/Pairs	99.9%	1,400,000,000	168 h	US$0.13
ION Torrent (Seq. by semiconductor)	170-400 Single	98%	80,000,000	2 h	US$2
Pacific Biosciences RSII (SMRT)	10,000 Single	85% (99.9%)	750,000	4 h	US$0.6
Oxford N. Minion (Nanopore sequencing)	10,000 Single	62% (96%)	4,400,000	48 h	US$0.02
10X Genomics	20,000 Single	99%	1,000,000,000	NA	NA

Libraries

★ Library types (orientations):

- Single reads

- Pair ends (PE) (150-800 bp insert size)

Illumina

- Mate pairs (MP) (2-40 Kb insert size)

Illumina

454/Roche

3. Whole genome assembly

Sequencing Amount

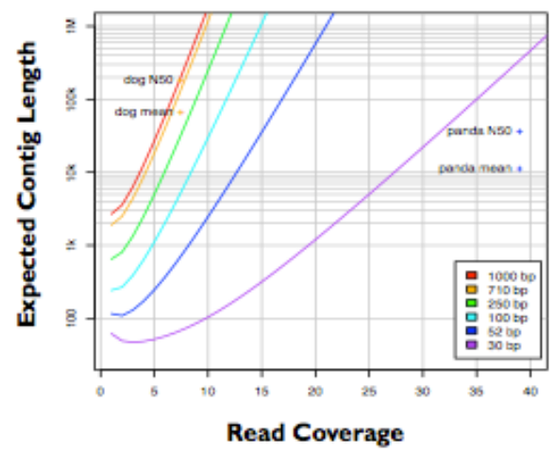
Ingredients for a good assembly

Read Length

Reads & mates must be longer than the repeats

- Short reads will have ***false overlaps*** forming hairball assembly graphs
- With long enough reads, assemble entire chromosomes into contigs

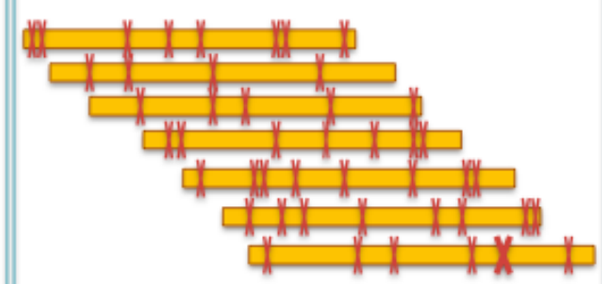
Coverage



High coverage is required

- Oversample the genome to ensure every base is sequenced with long overlaps between reads
- Biased coverage will also fragment assembly

Quality



Errors obscure overlaps

- Reads are assembled by finding kmers shared in pair of reads
- High error rate requires very short seeds, increasing complexity and forming assembly hairballs

Current challenges in *de novo* plant genome sequencing and assembly
Schatz MC, Witkowski, McCombie, WR (2012) *Genome Biology*. 12:243

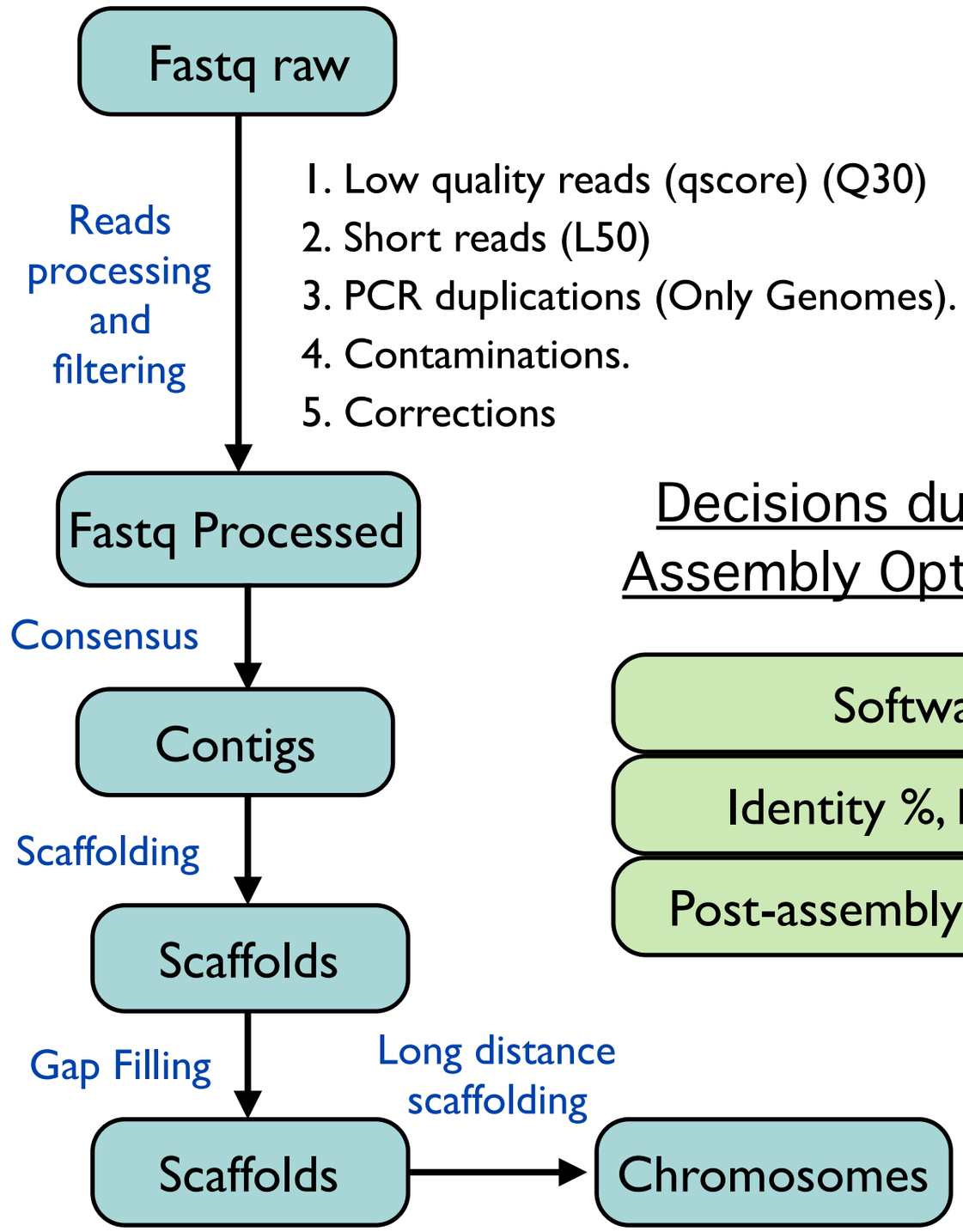
* Slide from MC. Schatz

Sequencing Amount

Depending of the genome complexity, technology used and assembler:

- More is better (if you have enough computational resources).
  - Sanger > 10X (less for BACs-by-BACs approaches).
  - 454 > 20X
  - **Illumina > 100X**
  - **PacBio > 20X (corrected by Illumina) or 50X (corrected PacBio)**
  - **10X Genomics > 20X**
- Polyploidy or high heterozygosity increase the amount of reads needed.
- The use of different library types (pair ends and mate pairs with different insert sizes is essential).
- Longer reads is preferable.

3. Whole genome assembly



Decisions during the Assembly Optimization

- Software
- Identity %, Kmer ...
- Post-assembly Filtering

Outline of Topics

1. Brief history about genome assembly

2. Basics about sequence assembly

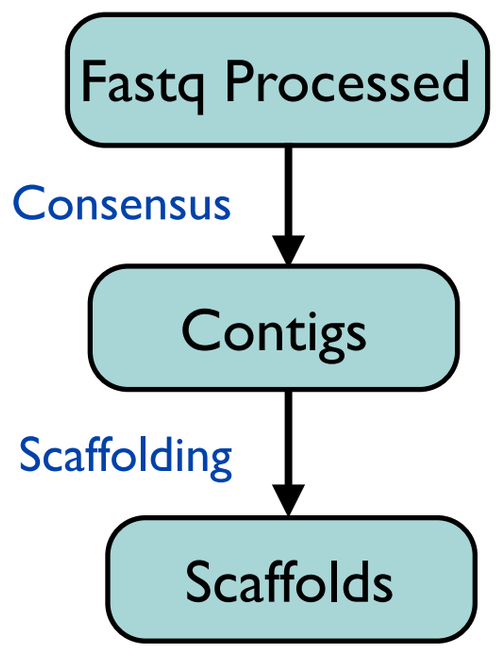
3. Whole genome assembly

   3.1. From reads to contigs and scaffolds.

   3.2. From scaffolds to chromosomes.

4. Transcriptome assembly

3.1. From reads to contigs and scaffolds.



Decisions during the Assembly Optimization

Software

3.1. From reads to contigs and scaffolds.

Tools: Assemblers

	Type	Technology Used	Features	Link
Arachne	Overlap-layout-consensus	Sanger, 454	Highly configurable	http://www.broadinstitute.org/crd/wiki/index.php/Main_Page
CABOG	Overlap-layout-consensus	Sanger, 454, Illumina, PacBio	Highly configurable	http://sourceforge.net/apps/mediawiki/wgs-assembler/index.php?title=Main_Page
MIRA	Overlap-layout-consensus	Sanger, 454	Highly configurable	http://sourceforge.net/apps/mediawiki/mira-assembler
gsAssembler	Overlap-layout-consensus	Sanger, 454	Easy to use	http://454.com/products/analysis-software/index.asp
iAssembler	Overlap-layout-consensus	Sanger, 454	Improves MIRA	http://bioinfo.bti.cornell.edu/tool/iAssembler
ABySS	Bruijn graph	454 or Illumina	Easy to use	http://www.bcgsc.ca/platform/bioinfo/software/abyss
ALLPATH-LG	Bruijn graph	454 or Illumina	Good results	http://www.broadinstitute.org/software/allpaths-lg/blog
Ray	Bruijn graph	454 or Illumina	Slow but use less memory	http://denovoassembler.sf.net/
SOAPdenovo2	Bruijn graph	454 or Illumina	Fastest	http://soap.genomics.org.cn/soapdenovo.html
Velvet	Bruijn graph	454 or Illumina or SOLiD	SOLiD	http://www.ebi.ac.uk/~zerbino/velvet/
Minia	Bloom filter + Brujn graph	Illumina	Really fast Only contigs	https://github.com/GATB/minia

Tools: Assemblers

... but there are more assemblers and information... Take a look to SeqAnswers

Also highly recommendable:

CSH PRESS GENOME RESEARCH

Assemblathon 1: A competitive assessment of de novo short read assembly methods

Dent Earl, Keith Bradnam, John St. John, et al.

Genome Res. 2011 21: 2224-2241 originally published online September 16, 2011
Access the most recent version at doi:10.1101/gr.126599.111

GAGE: A critical evaluation of genome assemblies and assembly algorithms

Steven L. Salzberg, Adam M. Phillippy, Aleksey Zimin, et al.

Genome Res. 2012 22: 557-567 originally published online December 6, 2011
Access the most recent version at doi:10.1101/gr.131383.111

3.1. From reads to contigs and scaffolds.

Tools: Assemblers

	Type	Technology Used	Features	Link
Arachne	Overlap-layout-consensus	Sanger, 454	Highly configurable	http://www.broadinstitute.org/crd/wiki/index.php/Main_Page
CABOG	Overlap-layout-consensus	Sanger, 454, Illumina, PacBio	Highly configurable	http://sourceforge.net/apps/mediawiki/wgs-assembler/index.php?title=Main_Page
MIRA	Overlap-layout-consensus	Sanger, 454	Highly configurable	http://sourceforge.net/apps/mediawiki/mira-assembler
gsAssembler	Overlap-layout-consensus	Sanger, 454	Easy to use	http://454.com/products/analysis-software/index.asp
iAssembler	Overlap-layout-consensus	Sanger, 454	Improves MIRA	http://bioinfo.bti.cornell.edu/tool/iAssembler
ABySS	Bruijn graph	454 or Illumina	Easy to use	http://www.bcgsc.ca/platform/bioinfo/software/abyss
ALLPATH-LG	Bruijn graph	454 or Illumina	Good results	http://www.broadinstitute.org/software/allpaths-lg/blog
Ray	Bruijn graph	454 or Illumina	Slow but use less memory	http://denovoassembler.sf.net/
SOAPdenovo2	Bruijn graph	454 or Illumina	Fastest	http://soap.genomics.org.cn/soapdenovo.html
Velvet	Bruijn graph	454 or Illumina or SOLiD	SOLiD	http://www.ebi.ac.uk/~zerbino/velvet/
Minia	Bloom filter + Brujn graph	Illumina	Really fast Only contigs	https://github.com/GATB/minia

3.1. From reads to contigs and scaffolds.

Tools: Assemblers

	Type	Technology Used	Features	Link
HGAP	Overlap-layout-consensus	PacBio	Recommended by PacBio. Multiple tools. Difficult to install	https://github.com/PacificBiosciences/Bioinformatics-Training/wiki/HGAP
Falcon	Overlap-layout-consensus	PacBio	Faster than HGAP or CABOG	https://github.com/PacificBiosciences/falcon
Sprai	Overlap-layout-consensus	PacBio	Easy to use interface for CABOG	http://zombie.cb.k.u-tokyo.ac.jp/sprai/README.html
Canu	Overlap-layout-consensus	PacBio, Oxford Nanopore	Easy to use	http://canu.readthedocs.org/
Miniasm	Overlap-layout-consensus	PacBio, Oxford Nanopore	Easy to use	https://github.com/lh3/miniasm
MECAT	Overlap-layout-consensus	PacBio, Oxford Nanopore	Fast assembler	https://github.com/xiaochuanle/MECAT
Spades	Brujn graphs	Illumina, Ion Torrent, PacBio, Oxford Nanopore	Diverse type of input	http://cab.spbu.ru/software/spades/
MaSurCa	Hybrid	PacBio + Illumina	Hybrid assembly	http://www.genome.umd.edu/masurca.html
Supernova	Hybrid	10X Genomics	Used with 10X genomics	https://support.10xgenomics.com/de-novo-assembly/software/pipelines/latest/using/running

3.1. From reads to contigs and scaffolds.

Tools: Assemblers

	Type	Technology Used	Features	Link
HGAP	Overlap-layout-consensus	PacBio	Recommended by PacBio. Multiple tools. Difficult to install	https://github.com/PacificBiosciences/Bioinformatics-Training/wiki/HGAP
Falcon	Overlap-layout-consensus	PacBio	Faster than HGAP or CABOG	https://github.com/PacificBiosciences/falcon
Sprai	Overlap-layout-consensus	PacBio	Easy to use interface for CABOG	http://zombie.cb.k.u-tokyo.ac.jp/sprai/README.html
Canu	Overlap-layout-consensus	PacBio, Oxford Nanopore	Easy to use	http://canu.readthedocs.org/
Miniasm	Overlap-layout-consensus	PacBio, Oxford Nanopore	Easy to use	https://github.com/lh3/miniasm
MECAT	Overlap-layout-consensus	PacBio, Oxford Nanopore	Fast assembler	https://github.com/xiaochuanle/MECAT
Spades	Brujn graphs	Illumina, Ion Torrent, PacBio, Oxford Nanopore	Diverse type of input	http://cab.spbu.ru/software/spades/
MaSurCa	Hybrid	PacBio + Illumina	Hybrid assembly	http://www.genome.umd.edu/masurca.html
Supernova	Hybrid	10X Genomics	Used with 10X genomics	https://support.10xgenomics.com/de-novo-assembly/software/pipelines/latest/using/running

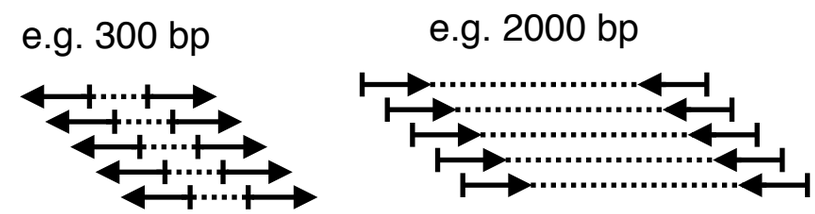
Scaffolding approaches:

1. Pair ends/Mate pair information (Since 2008) - Short Distances
2. Optical mapping data (Since 2012) - Medium-Long Distance
3. HiC (Since 2014) - Long-Very Long Distance
4. Linkage groups anchoring (Since 2001) - Very Long Distance

1. Pair ends/Mate pair information

0. Input

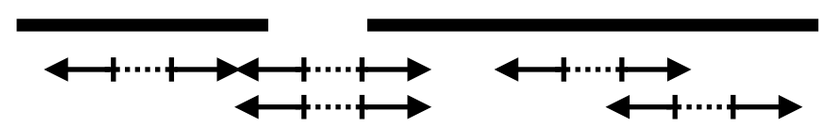
Contigs



Pair ends

Mate pairs

1.1 Small fragments mapping



1.2. Scaffold building

2.1. Long fragments mapping

2.2. Scaffold building

3.1. From reads to contigs and scaffolds.

Tools: Scaffolders

	Technology Used	Link
SSPACE	Short reads, pairs	http://www.baseclear.com/genomics/bioinformatics/basetools/SSPACE
SSPACE-Long	Long reads	http://www.baseclear.com/genomics/bioinformatics/basetools/SSPACE-longread
Bambus2	Pairs	https://www.cbcb.umd.edu/software/bambus2
Ragoo	Reference genome	https://github.com/malonge/RaGOO
PEP_scaffolder	Protein sequence	https://github.com/CAFS-bioinformatics/PEP_scaffolder

3.1. From reads to contigs and scaffolds.

Tools: Scaffolders

Journal List > Genome Biol > v.15(3); 2014 > PMC4053845

BioMed Central The Open Access Publisher | Genome Biology

this article | search | submit a manuscript | register

Genome Biol. 2014; 15(3): R42.
Published online 2014 Mar 3. doi: 10.1186/gb-2014-15-3-r42

PMCID: PMC4053845

A comprehensive evaluation of assembly scaffolding tools

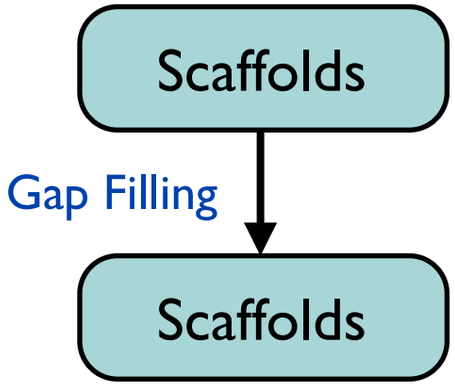
Martin Hunt,[1] Chris Newbold,[2,1] Matthew Berriman,[1] and Thomas D Otto[1]

Author information ▶ Article notes ▶ Copyright and License information ▶

This article has been cited by other articles in PMC.

Genome

3.1. From reads to contigs and scaffolds.



3.1. From reads to contigs and scaffolds.

Tools: Gap fillers

	Type	Technology Used	Features	Link
GapCloser	Overlap-layout-consensus	Illumina	C++ Free	http://soap.genomics.org.cn/soapdenovo.html
GapFiller	Overlap-layout-consensus	Any	Perl Commercial*	http://www.baseclear.com/landingpages/basetools-a-wide-range-of-bioinformatics-solutions/gapfiller/
PBSuite	Overlap-layout-consensus	454, PacBio	Python, Long reads	http://sourceforge.net/p/pb-jelly/wiki/

Outline of Topics

1. Brief history about genome assembly

2. Basics about sequence assembly

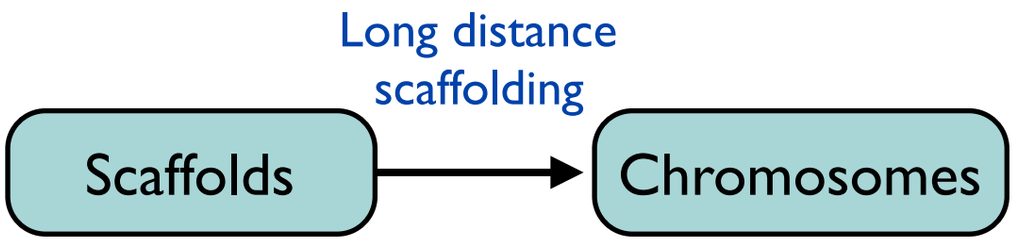
3. Whole genome assembly

   3.1. From reads to contigs and scaffolds.

   3.2. From scaffolds to chromosomes.

4. Transcriptome assembly

3.2. From scaffolds to chromosomes.

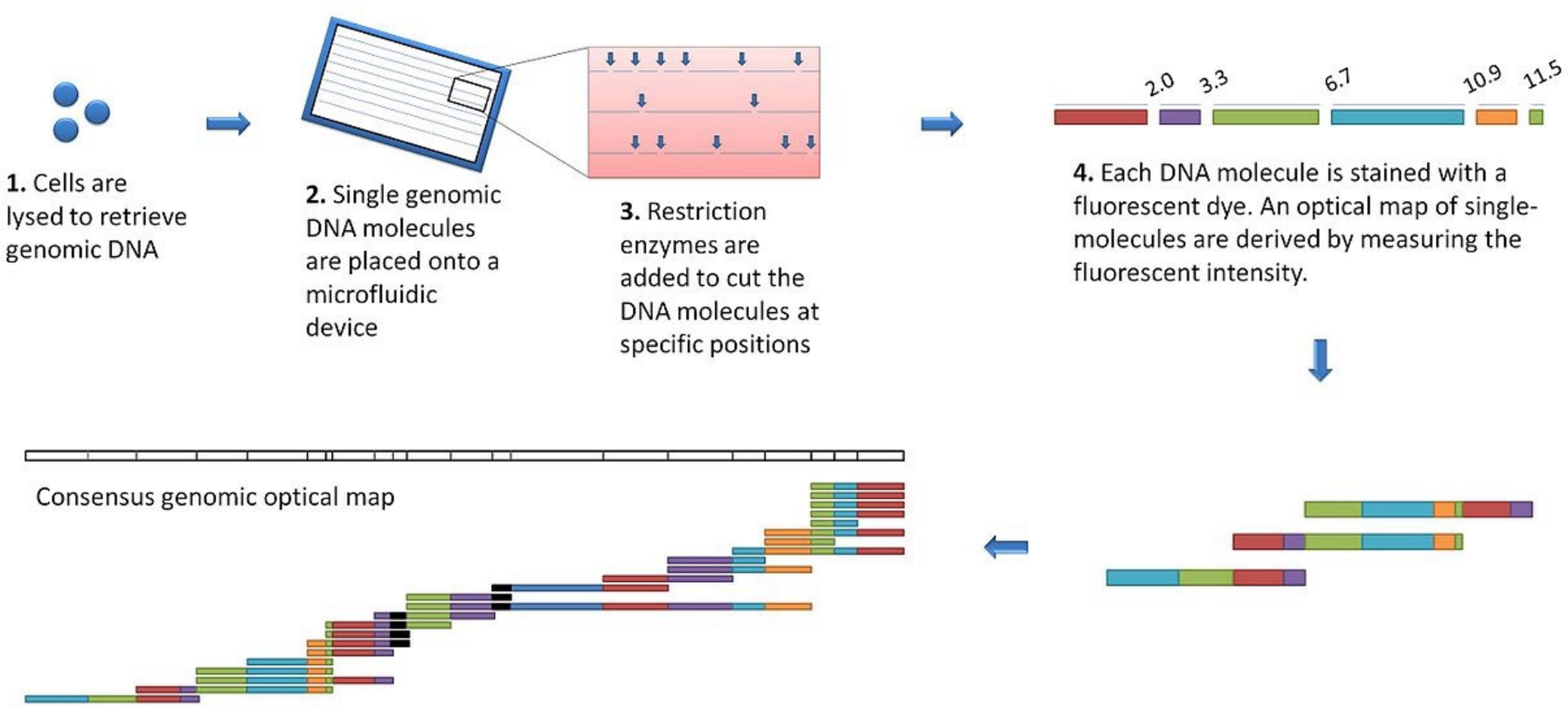


Scaffolding approaches:

1. Pair ends/Mate pair information (Since 2008) - Short Distances
2. Optical mapping data (Since 2012) - Medium-Long Distance
3. HiC (Since 2014) - Long-Very Long Distance
4. Linkage groups anchoring (Since 2001) - Very Long Distance

2. Optical mapping data (Since 2012).

Optical mapping is a technique for constructing ordered, genome-wide, **high-resolution restriction maps from single, stained molecules of DNA**, called "optical maps". By mapping the location of restriction enzyme sites along the unknown DNA of an organism, the spectrum of resulting DNA fragments collectively serves as a unique **"fingerprint" or "barcode"** for that sequence

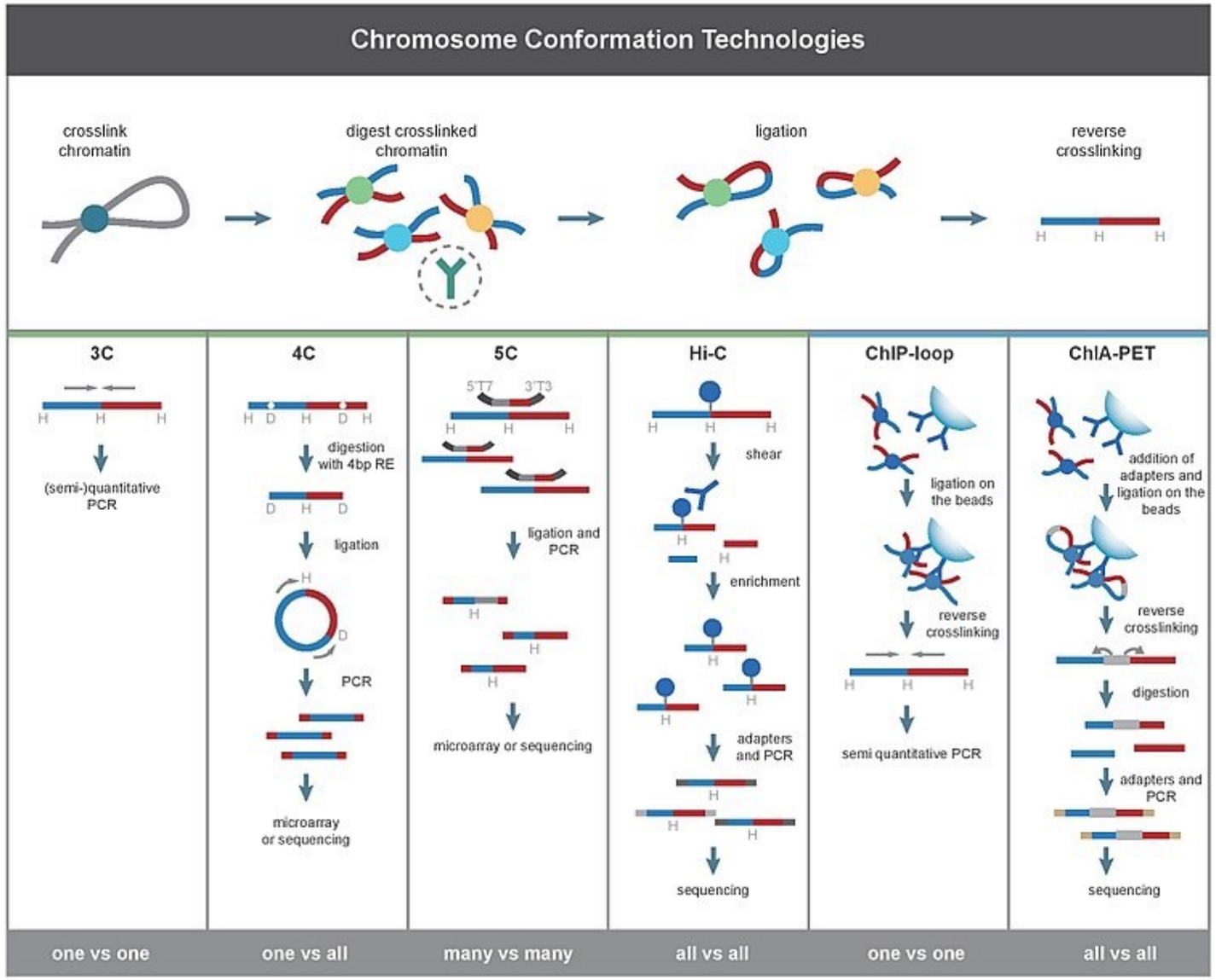


5. Overlapping of the multiple single-molecule maps gives us the consensus genomic optical map

https://en.wikipedia.org/wiki/Optical_mapping

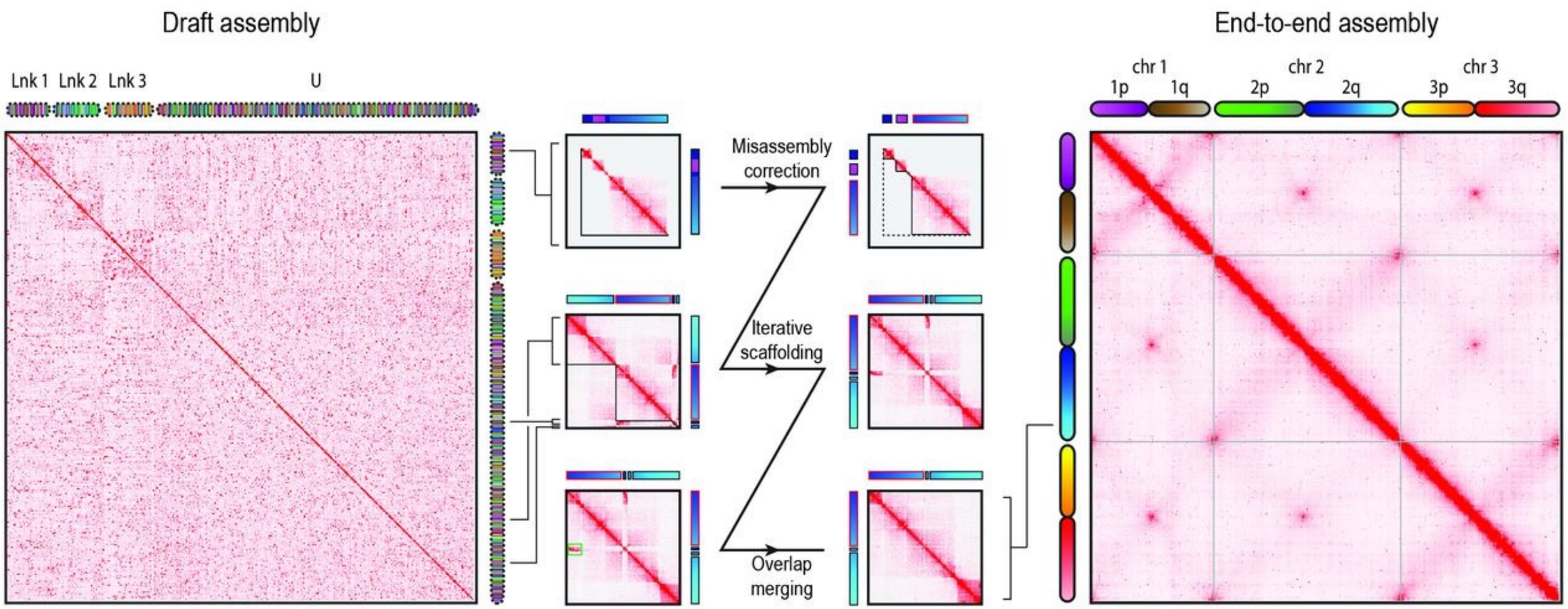
3. HiC, Chromatin Conformation Capture (Since 2014).

Chromosome conformation capture techniques (often abbreviated to 3C technologies or 3C-based methods) are a set of molecular biology methods used to analyze the **spatial organization of chromatin** in a cell. These methods quantify the number of interactions between genomic loci that are nearby in 3-D space, but may be separated by many nucleotides in the linear genome



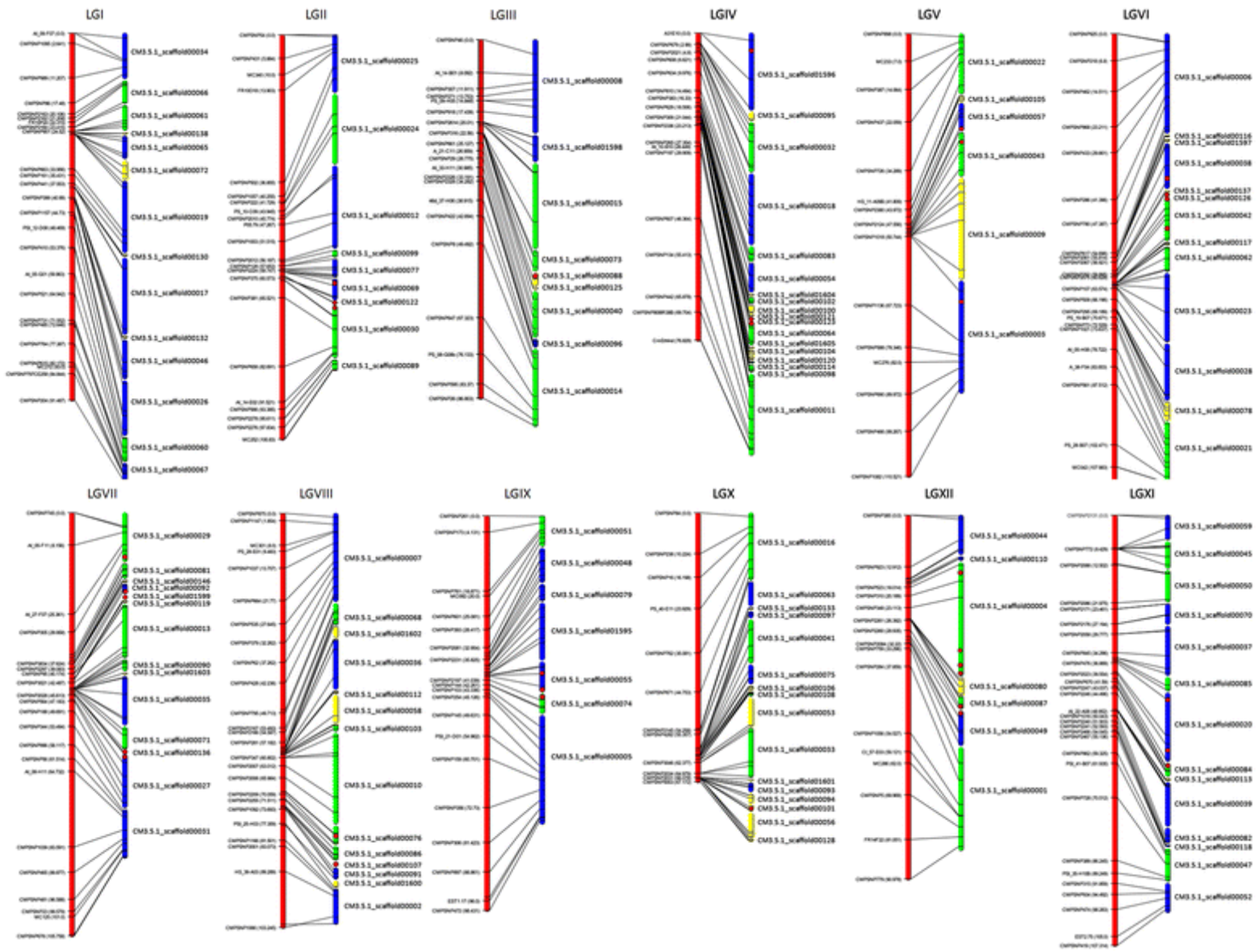
https://en.wikipedia.org/wiki/Chromosome_conformation_capture

3. HiC, Chromatin Conformation Capture (Since 2014).



Dudechenko et al, Science 07 Apr 2017: Vol. 356, Issue 6333, pp. 92-95
DOI: 10.1126/science.aal3327

4. Linkage groups anchoring (Since 2001) - Very Long Distance



Argyris et al, BMC Genomics201516:4

Outline of Topics

1. Brief history about genome assembly

2. Basics about sequence assembly

3. Whole genome assembly

   3.1. From reads to contigs and scaffolds.

   3.2. From scaffolds to chromosomes.

4. Transcriptome assembly

3. Whole genome assembly

Decisions during the Assembly Optimization

- Software
- Identity %, Kmer ...
- Post-assembly Filtering

Computers

Bigger is better:

- How much do you need depends of:
  - ➡ how big is your genome ?
    ~ Human size (3Gb) require ~256 Gb to 1 Tb
  - ➡ how many reads do you have, ?
    More reads, more memory and hard disk.
  - ➡ what software are you going to use ?
    OLC uses more memory and time than DBG.
  - ➡ what parameters are you going to use ?
    Bigger Kmers, more memory.

Four options:

1. Buy your own server (512 Gb, 4.5 Tb, 64 cores; ~ $15,000).
2. Rent a server for ~ 1 month (same specs. $1.5/h; ~$1,000)(CBSU).
3. Use a supercomputing center associated with NSF, NIH, USDA... where they offer reduced prices (iPlant, Indiana University....).
4. Collaborate with some group with a big server.

Assembly evaluation

Different methods to evaluate an assembly:

1. Assembly stats (total assembly size vs expected size; N50/L50...)

2. Read remapping, variant calling and coverage evaluation.

3. Comparison with alternative sequencing methods.

4. Gene Space Completeness (RNASeq & EST mapping, BUSCO).

5. Comparison with genetic information.

Assembly evaluation: Assembly stats

Rationale: Close that a sequence assembly is to the expected genome size, better it will be the genome assembly.

- Ideal case: The final assembly will have as many sequences as chromosomes have the genome.

- Real case: The assembly will be fragmented is sequences with different sizes. Lower number of fragments, better quality.

Assembly evaluation: Assembly stats

During the assembly optimization will be generated several assemblies. The most used parameters to evaluate the assembly are:

1. Total Assembly Size,

How far is this value from the estimated genome size

2. Total Number of Sequences (Scaffold/Contigs)

How far is this value from the number of chromosomes.

3. Longest scaffold/contig

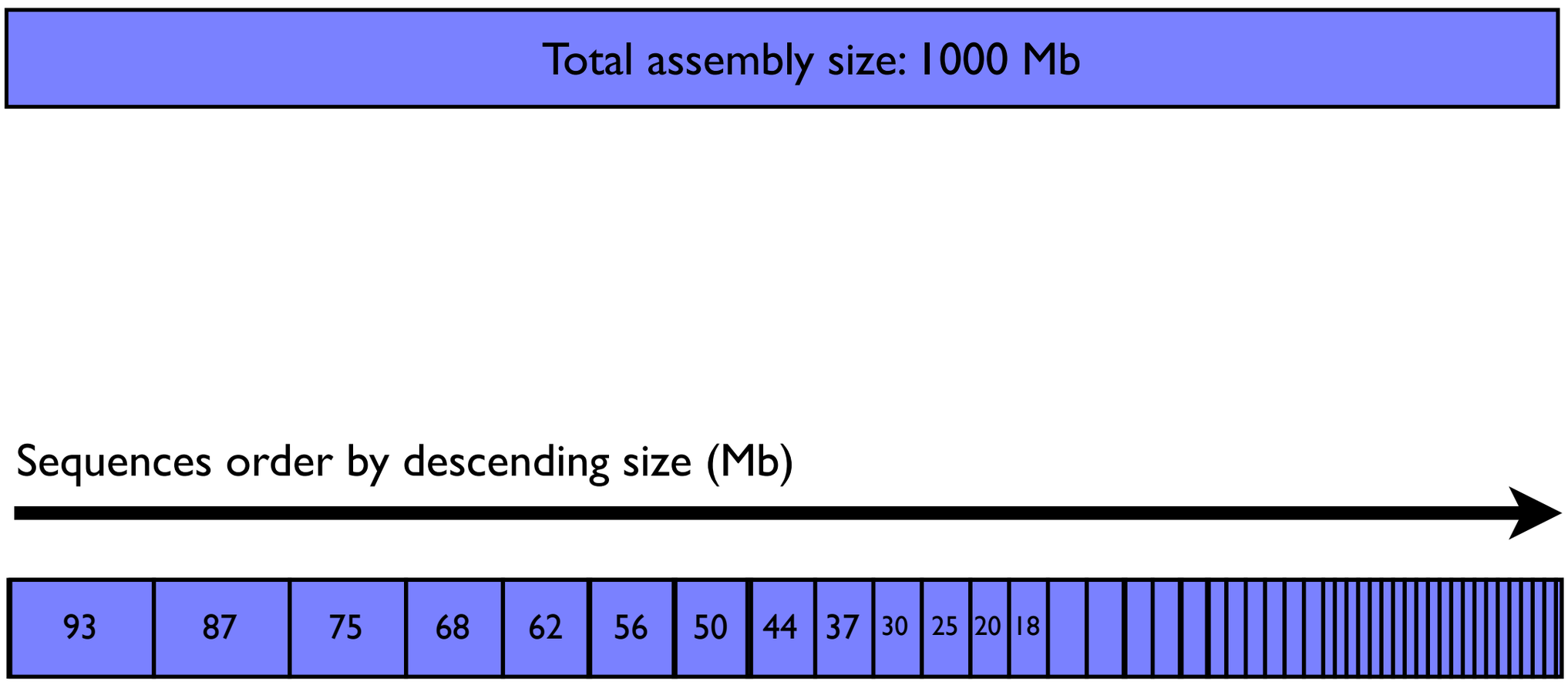
4. Average scaffold/contig size

5. N50/L50 (or any other N/L)

Number sequence (N) and minimum size of them (L) that represents the 50% of the assembly if the sequences are sorted by size, from bigger to smaller.

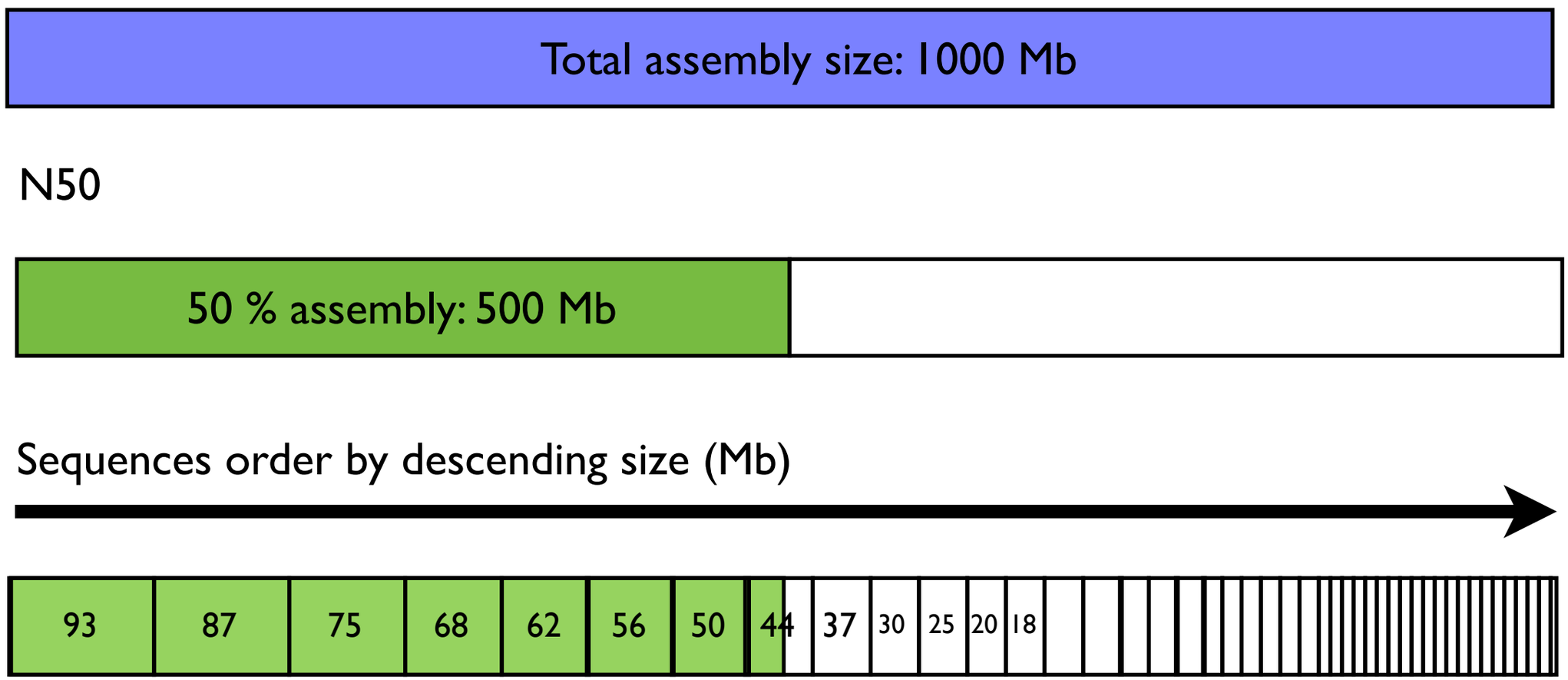
Assembly evaluation: Assembly stats

N50/L50



Assembly evaluation: Assembly stats

N50/L50

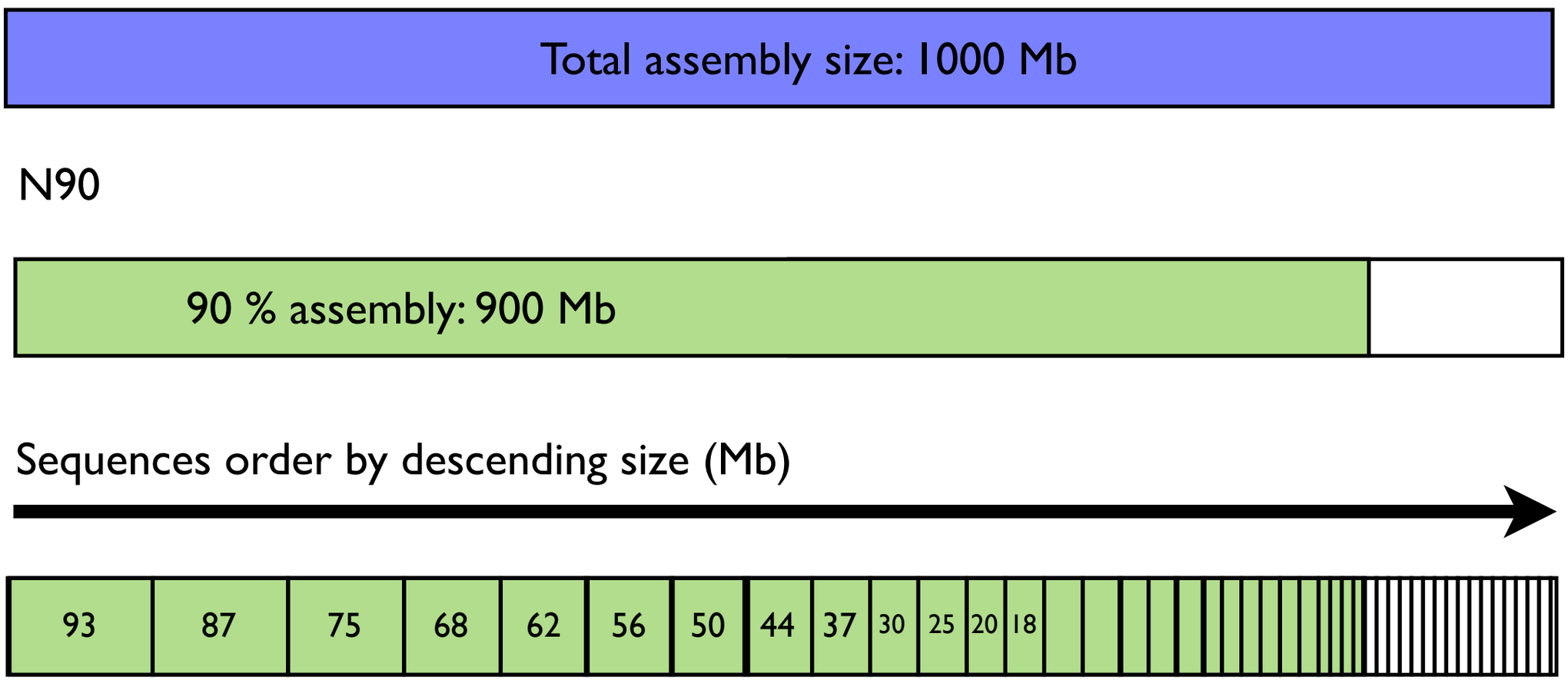


N50 = 7 sequences

L50 = 50 Mb

Assembly evaluation: Assembly stats

N90/L90



N90 = 29 sequences

L90 = 12.5 Mb

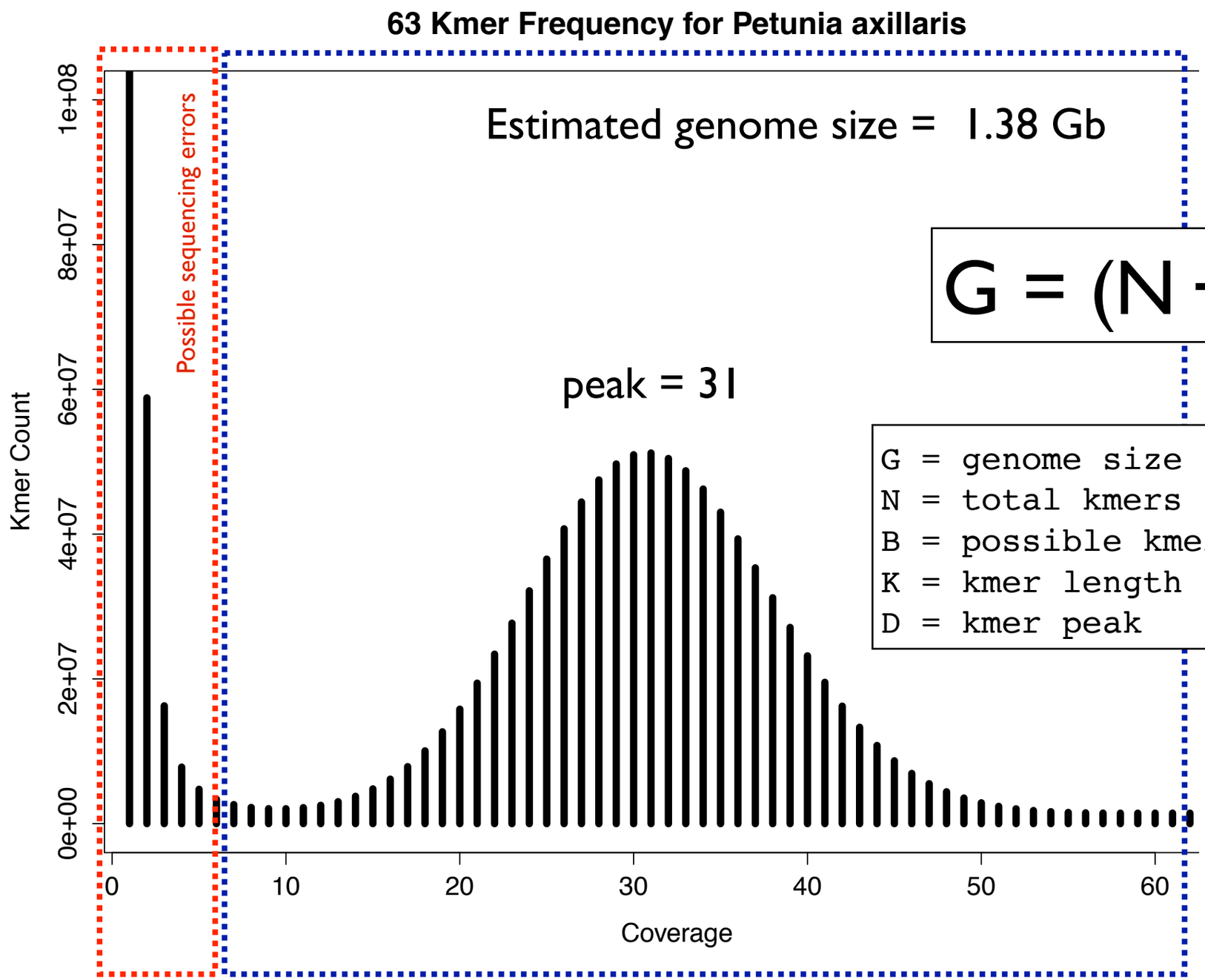
Assembly evaluation: Assembly stats

Example: Sequencing of the *Petunia axillaris* genome.

Current Assembly Dataset	*P. axillaris*  Peaxi v1.6.2	
	Contigs	Scaffolds
Total assembly size (Gb)	1.22	1.26
Total assembly sequences	109,892	83,639
Longest sequence (Mb)	0.57	8.56
Sequence length mean (Kb)	11.08	15.05
N90 (sequences)	13,481	1,051
L90 (Kb)	22.28	295.75
N50 (sequences)	3,943	309
L50 (Kb)	95.17	1,236.73

Assembly evaluation: Assembly stats

Genome size (Kmer count)



$$G = (N - B) * K / D * 2$$

```
G = genome size
N = total kmers
B = possible kmers with errors
K = kmer length
D = kmer peak
```

Assembly evaluation: Assembly stats

Estimated genome size for *P. axillaris*:

- Flow Cytometry (White & Rees,1987)= 1.37 Gb
- Kmer Count* = 1.38 Gb
- Assembly size (scaffolds v1.6.2) = 1.26 Gb
- Assembly size (contigs v1.6.2) = 1.22 Gb

Estimated genome not assembled = 110 - 120 Mb

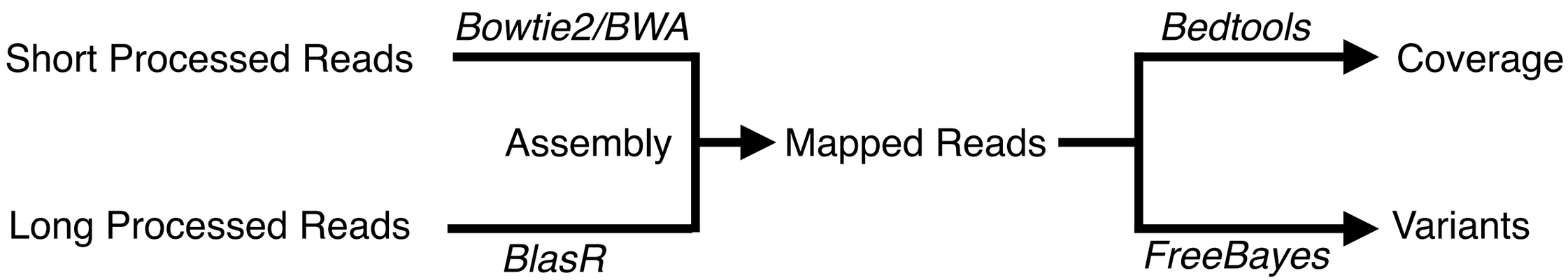
Assembly evaluation: Read mapping

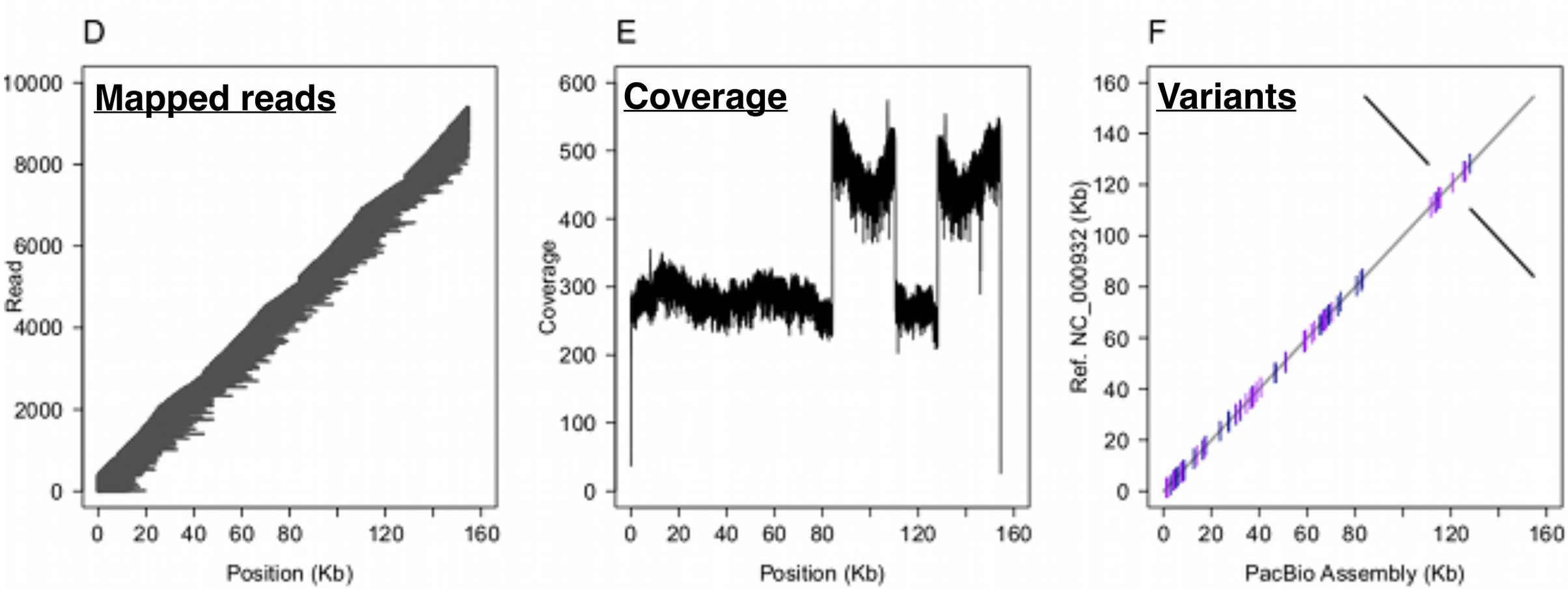
Rationale: Low quality assembled regions will have:

- Low coverage (breaking points)
- High coverage and high number of polymorphisms (due the collapsing of multiple copies.

3. Whole genome assembly

Assembly evaluation: Read mapping



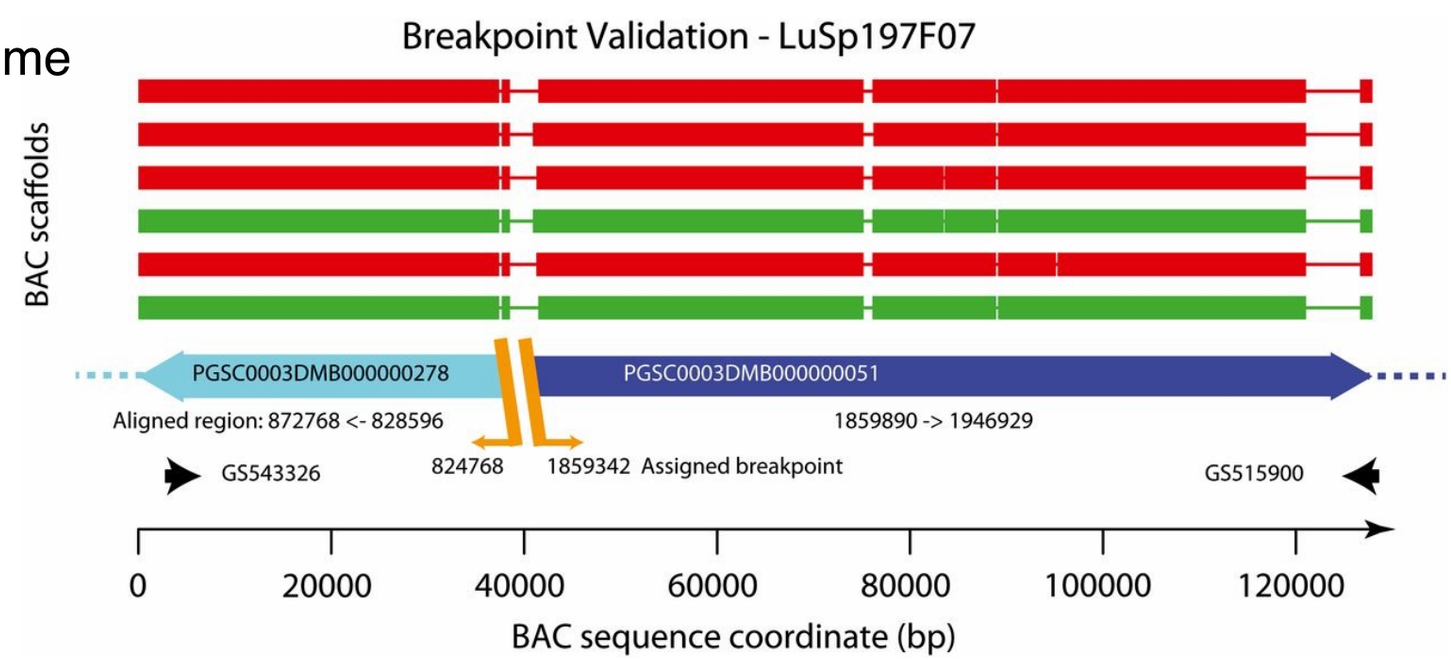


Assembly evaluation: Comparative with alternative sequencing methods

Rationale: A sequence assembly should agree with alternative sequencing methods.

- Illumina based assembly —> PacBio based assembly
- Short distance scaffolding (< 200 Kb) —> Optical Mapping scaffolding.
- Long distance scaffolding (> 200 Kb) —> BACs, YACs and Genetic maps

E.g. Potato genome



Sharma et al. 2013

Assembly evaluation: Gene space completeness

Rationale: The gene space is defined as the whole collection of genes in a genome. A complete genome should have a complete gene space. There are two ways to evaluate the completeness of a gene space.

- Check the completeness of a set of conserved genes (CEGMA/BUSCO).
  - Maybe not all the conserved genes are in the studied genome
- Check the mapping rate of a transcriptome to a genome (RNASeq).
  - The RNASeq may have contaminations of other organisms.

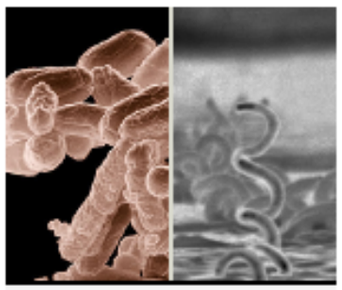
Assembly evaluation: Gene space completeness

Other way to estimate how good is an assembly, it is to run BUSCO over the assembly and evaluate how much of the gene space was captured.

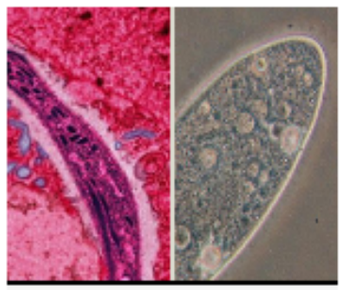
http://busco.ezlab.org/

BUSCO v3

Datasets


Bacteria sets


Eukaryota sets


Protists sets


Metazoa sets


Fungi sets


Plants set

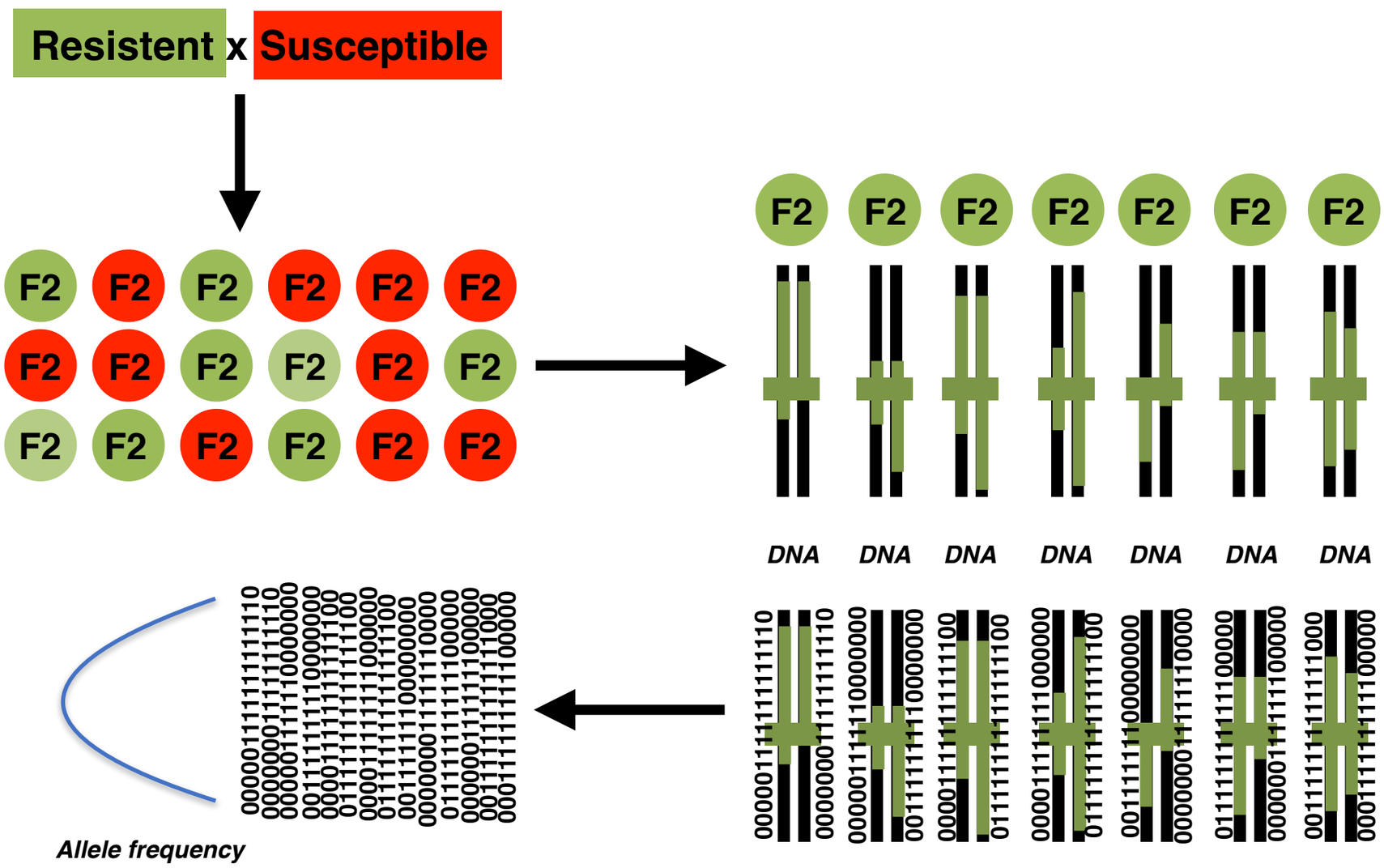
Assembly evaluation: Gene space completeness

Other way to estimate how good is an assembly, it is to run BUSCO over the assembly and evaluate how much of the gene space was captured.

```
# BUSCO version is: 3.0.2
# The lineage dataset is: embryophyta_odb9 (Creation date: 2016-02-13, number of
species: 30, number of BUSCOs: 1440)
# To reproduce this run: python /data/software/busco/scripts/run_BUSCO.py -i Arth
a_TAIR10_Genome.fasta -o TAIR10_GENOME_BUSCO_TEST -l /data/software/busco/dataset
s/embryophyta_odb9/ -m genome -c 60 -sp arabidopsis
#
# Summarized benchmarking in BUSCO notation for file Artha_TAIR10_Genome.fasta
# BUSCO was run in mode: genome

        C:98.2%[S:97.3%,D:0.9%],F:0.5%,M:1.3%,n:1440

        1414    Complete BUSCOs (C)
        1401    Complete and single-copy BUSCOs (S)
        13      Complete and duplicated BUSCOs (D)
        7       Fragmented BUSCOs (F)
        19      Missing BUSCOs (M)
        1440    Total BUSCO groups searched
```

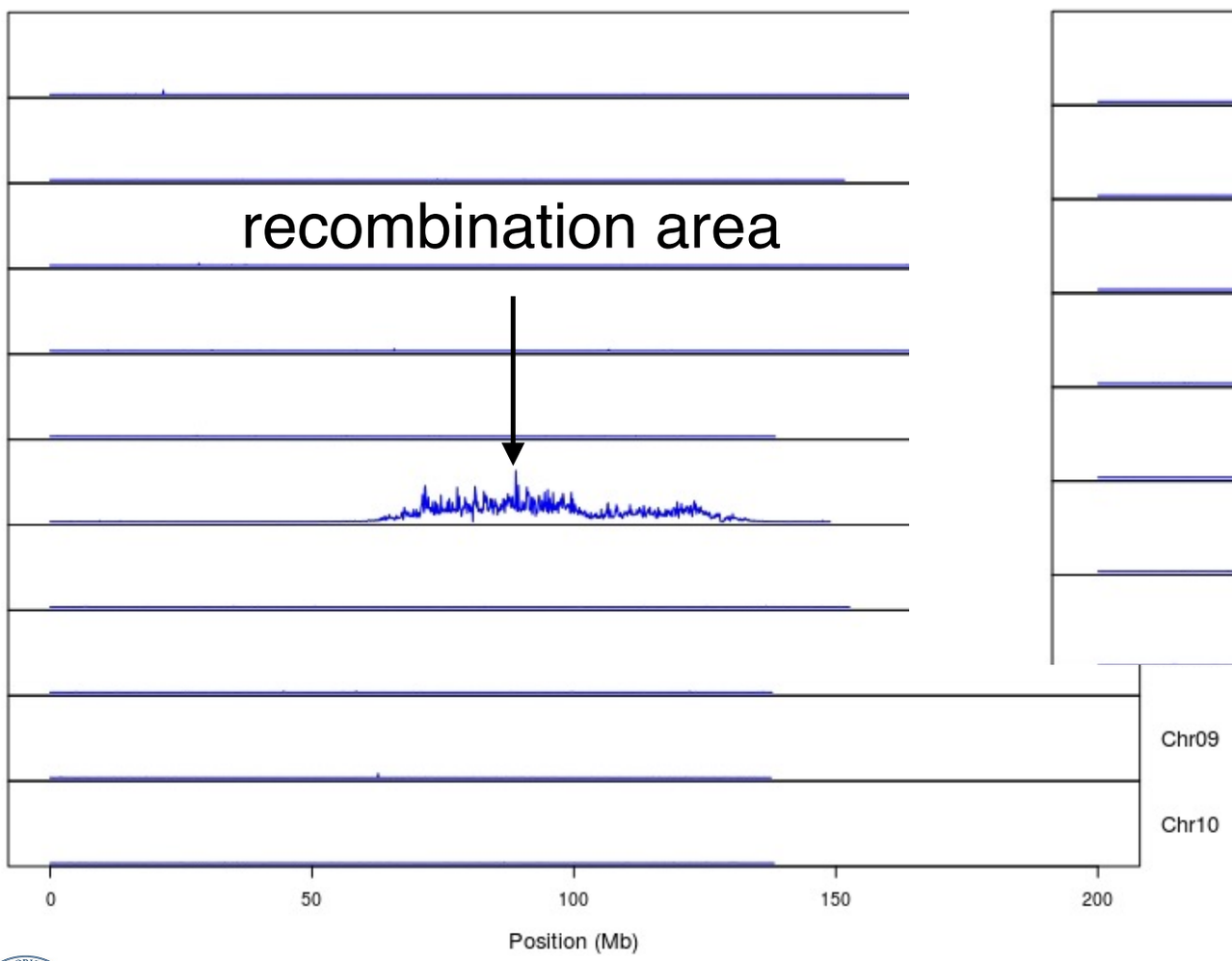
Assembly evaluation: Comparison with Genetic Information

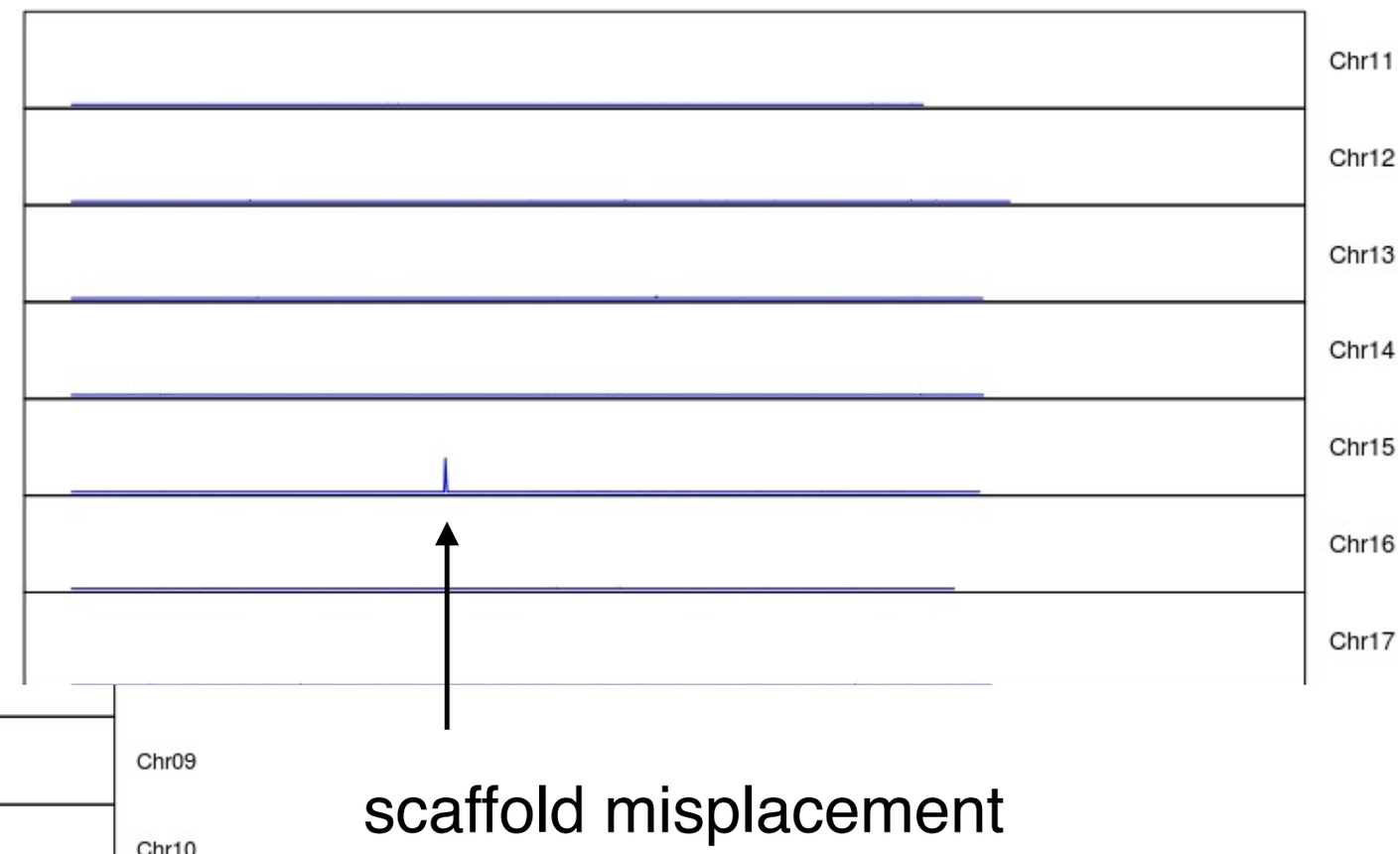
Rationale: The assembly information should be in agreement with genetic maps and recombination studies.



Assembly evaluation: Comparison with Genetic Information

Rationale: The assembly information should be in agreement with genetic maps and recombination studies.





3. Whole genome assembly

	Whole Genome Representation	Sequence Status	Genes	Usability
1	Incomplete for non-repetitive regions	Small scaffolds and contigs	Incomplete genes	Markers development
2	Complete for non-repetitive regions	Medium scaffolds and contigs	Complete but 1-2 genes/contig	Gene mining
3	Complete for non-repetitive regions	Large scaffolds and contigs	Several dozens of genes/contig	Microsynteny
4	Complete for almost the whole genome	Pseudomolecules	Hundreds of genes/ contig	Any (Synteny, Candidate gene by QTLs)
5	Complete genome	Pseudomolecules	Thousands of genes/ contig	

Outline of Topics

1. Brief history about genome assembly

2. Basics about sequence assembly

3. Whole genome assembly

   3.1. From reads to contigs and scaffolds.

   3.2. From scaffolds to chromosomes.

4. Transcriptome assembly

4. Transcriptome assembly

https://github.com/trinityrnaseq/trinityrnaseq/wiki

Home

Brian Haas edited this page on 10 Apr 2018 · 37 revisions

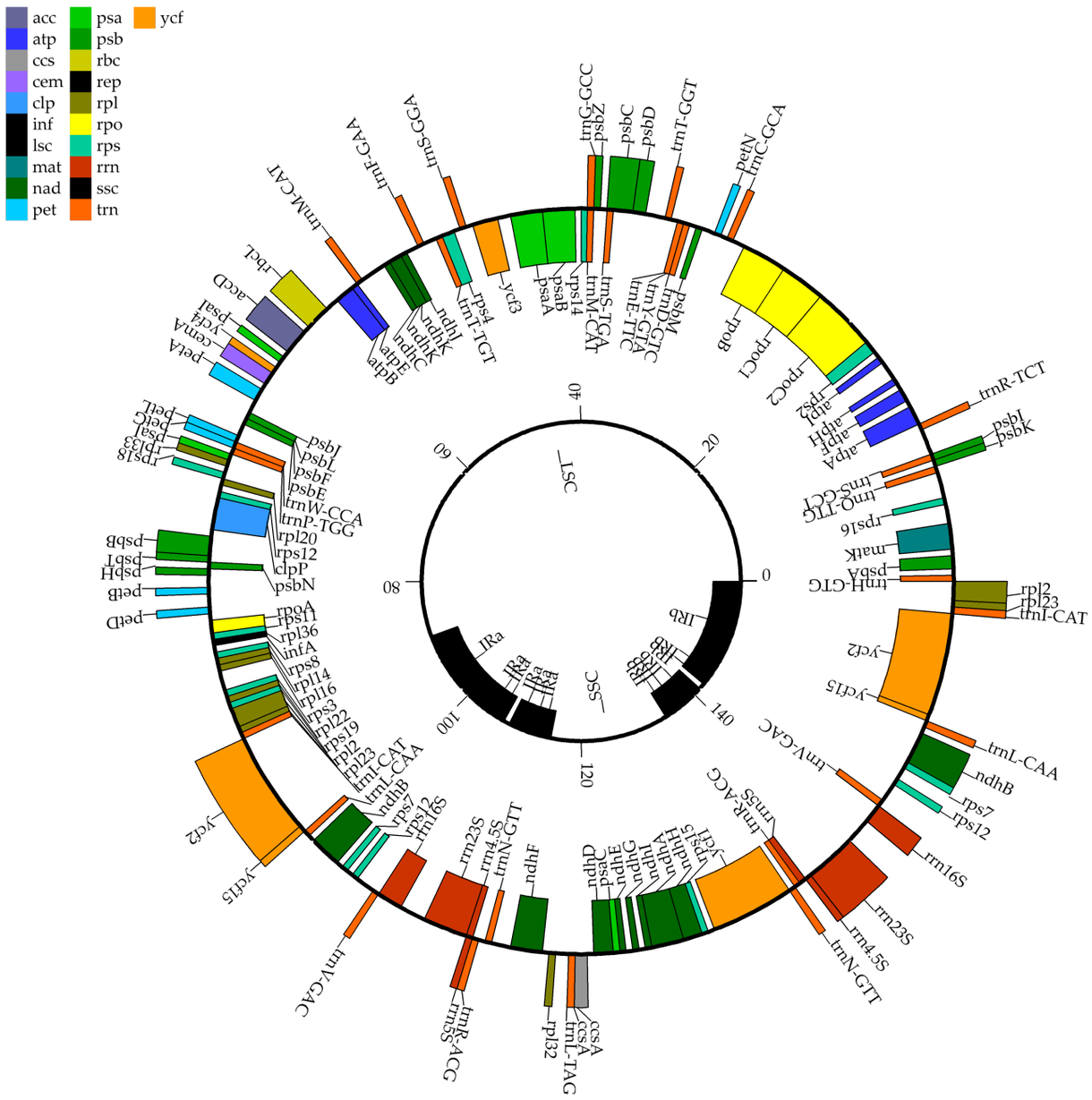
RNA-Seq De novo Assembly Using Trinity

Exercises 5

Chloroplast genome
assembly

- Exercise 5.1: Get read dataset stats.
- Exercise 5.2: Short single read assembly - Coverage.
- Exercise 5.3: Short single read assembly - Kmer.
- Exercise 5.4: Short pair end read assembly - Scaffolding.
- Exercise 5.5: Long read assembly.
- Exercise 5.6: Assembly evaluation - Coverage.
- Exercise 5.7: Assembly evaluation - Variants.
- Exercise 5.8: Assembly structural annotation.

N. benthamiana chloroplast



Input data for the exercise

/data/99_genomicclass/00_shared_data/Genomics/

- 00_Reference
- 01_Illumina
- 02_PacBio
- 03_HiC

Library type Pair

```
Niben_MP02Kb_CHL_R1.fastq
Niben_MP02Kb_CHL_R2.fastq
Niben_MP05Kb_CHL_R1.fastq
Niben_MP05Kb_CHL_R2.fastq
Niben_MP15Kb_CHL_R1.fastq
Niben_MP15Kb_CHL_R2.fastq
Niben_MP20Kb_CHL_R1.fastq
Niben_MP20Kb_CHL_R2.fastq
Niben_PE250bp_CHL_R1.fastq
Niben_PE250bp_CHL_R2.fastq
Niben_PE500bp_CHL_R1.fastq
Niben_PE500bp_CHL_R2.fastq
```

- Exercise 5.1: Get read dataset stats.
- Exercise 5.2: Short single read assembly - Coverage.
- Exercise 5.3: Short single read assembly - Kmer.
- Exercise 5.4: Short pair end read assembly - Scaffolding.
- Exercise 5.5: Long read assembly.
- Exercise 5.6: Assembly evaluation - Coverage.
- Exercise 5.7: Assembly evaluation - Variants.
- Exercise 5.8: Assembly structural annotation.

- Exercise 5.1: Get read dataset stats.

<u>OBJECTIVE:</u> To obtain the stats for all the read files from the 01_Illumina and 02_PacBio directories

<u>TOOLS:</u> Fastq-stats

<u>COMMAND:</u>

```
fastq-stats input_file.fq > input_file.stats.txt
## To get all the read data
grep read *stats.txt
```

<u>INPUT FILES LOCATION:</u>

```
/data/99_genomicclass/00_shared_data/Genomics/
01_Illumina/
/data/99_genomicclass/00_shared_data/Genomics/
02_PacBio/
```

- Exercise 5.1: Get read dataset stats.
- Exercise 5.2: Short single read assembly - Coverage.
- Exercise 5.3: Short single read assembly - Kmer.
- Exercise 5.4: Short pair end read assembly - Scaffolding.
- Exercise 5.5: Long read assembly.
- Exercise 5.6: Assembly evaluation - Coverage.
- Exercise 5.7: Assembly evaluation - Variants.
- Exercise 5.8: Assembly structural annotation.

- Exercise 5.2: Short single read assembly - Coverage.

<u>OBJECTIVE:</u> To run different assemblies with different number of single reads and then compare the size of the contains. We will run **minia** over the file Niben_PE250bp_CHL_R1.fastq. Then we will select the 10% (0.1), 1% (0.01) and 0.1% (0.001) of the reads using the tool **seqtk sample** and we run mini again**.**

Name the inputs as Nb_IN100.0, Nb_IN010.0, Nb_IN001.0 and Nb_IN000.1 and the outputs as Minia_CHL100.0, Minia_CHL010.0, Minia_CHL001.0 and Minia_CHL000.1

<u>TOOLS:</u> Minia, Seqtk

- Exercise 5.2: Short single read assembly - Coverage.

COMMANDS:

```
## Assembly
minia -in Nb_INXXX.fastq -kmer-size 63 -abundance-min 5 -out Minia_CHLXXX -nb-cores 10

## Get a subset
seqtk sample 01_Illumina/Niben_PE250bp_CHL_R1.fastq XXX > Nb_INXXX.fastq
fastq-stats Nb_INXXX.fastq > Nb_INXXX.stats.txt

## Get the stats
FastaSeqStats -i Minia_CHLXXX.contigs.fasta > Minia_CHLXXX.contigs.stats.txt
```

- Exercise 5.2: Short single read assembly - Coverage.

INPUT FILES LOCATION:

`/data/99_genomicclass/00_shared_data/Genomics/Niben_PE250bp_CHL_R1.fastq`

SUGGESTIONS FOR DATA VISUALIZATION

1. Once you get the assembly stats with FastaSeqStats retrieve all the assemblies sizes running: `grep Total NibenSR_CHL*.contigs.stats.txt | sed -r 's/.+:\s+//' | sed -r 's/\s+bp//' > NibenSR_CHL_ALL_TotalSize.txt`

2. Copy the file into your computer using FileZilla.

3. Load the file in R Studio

4. Use barplot to represent the different sizes.

- Exercise 5.2: Short single read assembly - Coverage.

GROUP DISCUSSION

Estimate the coverage used for each of the assemblies based in the input datasets and discuss the which dataset delivered better results.

- Exercise 5.1: Get read dataset stats.
- Exercise 5.2: Short single read assembly - Coverage.
- Exercise 5.3: Short single read assembly - Kmer.
- Exercise 5.4: Short pair end read assembly - Scaffolding.
- Exercise 5.5: Long read assembly.
- Exercise 5.6: Assembly evaluation - Coverage.
- Exercise 5.7: Assembly evaluation - Variants.
- Exercise 5.8: Assembly structural annotation.

- Exercise 5.3: Short single read assembly - Kmer.

<u>OBJECTIVE:</u> To run different assemblies with different kmers. You will run **minia** over the file Nb_IN010.0.fastq from the previous exercise. Run four different assemblies using the Kmers of 17, 31, 63 and 81.

Name the outputs as Minia_CHL_K17, Minia_CHL_K31, Minia_CHL_K63 and Minia_CHL_K81

<u>TOOLS:</u> Minia, Seqtk

- Exercise 5.3: Short single read assembly - Kmer.

COMMANDS:

```
## Assembly
minia -in Nb_IN010.0.fastq -kmer-size 63 -abundance-
min 5 -out Minia_CHL_KXX -nb-cores 10

## Get the stats
FastaSeqStats -i Minia_CHLXXX.contigs.fasta >
Minia_CHLXXX.contigs.stats.txt
```

- Exercise 5.3: Short single read assembly - Kmer.

GROUP DISCUSSION

Discuss the use of different Kmer sizes for the assembly. Is longer always better?

- Exercise 1: Get read dataset stats.
- Exercise 2: Short single read assembly - Coverage.
- Exercise 3: Short single read assembly - Kmer.
- Exercise 4: Short pair end read assembly - Scaffolding.
- Exercise 5: Long read assembly.
- Exercise 6: Assembly evaluation - Coverage.
- Exercise 7: Assembly evaluation - Variants.
- Exercise 8: Assembly structural annotation.

- Exercise 5.4: Short pair end read assembly - Scaffolding.

<u>OBJECTIVE:</u> To run different assemblies with different combinations of pair end and mate pair datasets. For this assembly you will be using ABYSS instead Minia because it has a scaffolding pipeline. We want to evaluate the effect of scaffolding rather than the assembly so we will need to prepare different datasets containing the same number of total reads.

The different assemblies to run are:

- ABYSS_01SR
- ABYSS_02PE
- ABYSS_03PE
- ABYSS_04PEMP
- ABYSS_05PEMP

- Exercise 5.4: Short pair end read assembly - Scaffolding.

COMMANDS

- ABYSS_01SR

```
## Select 50,000 reads
seqtk sample 01_Illumina/Niben_PE250bp_CHL_R1.fastq 50000
> Niben_CHL50K.fq

## Run ABYSS
abyss-pe name="NibenCHL_Ab01SR" k=63 in="Niben_CHL50K.fq"

## Get the stats
FastaSeqStats -i NibenCHL_Ab01SR-3.fa
>NibenCHL_Ab01SR-3.stats.txt
```

- Exercise 5.4: Short pair end read assembly - Scaffolding.

COMMANDS

- ABYSS_02PE

```
## Select 50,000 reads (25,000 per pair)
seqtk sample -s1000 01_Illumina/Niben_PE250bp_CHL_R1.fastq
25000 > Niben_PE250bp_25K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_PE250bp_CHL_R2.fastq
25000 > Niben_PE250bp_25K_R2.fq

## Run ABYSS
abyss-pe name="NibenCHL_Ab02PE" k=63
in=“Niben_PE250bp_25K_R1.fq Niben_PE250bp_25K_R2.fq"

## Get the stats
FastaSeqStats -i NibenCHL_Ab02PE1-scaffolds.fa >
NibenCHL_Ab02PE1-scaffolds.stats.txt
```

- Exercise 5.4: Short pair end read assembly - Scaffolding.

COMMANDS

- ABYSS_03PE

```
## Select 50,000 reads (12,500 per pair)
seqtk sample -s1000 01_Illumina/Niben_PE250bp_CHL_R1.fastq
12500 > Niben_PE250bp_12K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_PE250bp_CHL_R2.fastq
12500 > Niben_PE250bp_12K_R2.fq
seqtk sample -s1000 01_Illumina/Niben_PE500bp_CHL_R1.fastq
12500 > Niben_PE500bp_12K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_PE500bp_CHL_R2.fastq
12500 > Niben_PE500bp_12K_R2.fq

## Run ABYSS
abyss-pe name="NibenCHL_Ab03PE2" k=63 lib="pe250 pe500"
pe250="Niben_PE250bp_12K_R1.fq Niben_PE250bp_12K_R2.fq"
pe500="Niben_PE500bp_12K_R1.fq Niben_PE500bp_12K_R2.fq"
```

- Exercise 5.4: Short pair end read assembly - Scaffolding.

COMMANDS

- ABYSS_03PE

```
## Get the stats
FastaSeqStats -i NibenCHL_Ab03PE3-scaffolds.fa >
NibenCHL_Ab03PE2-scaffolds.stats.txt
```

- Exercise 4: Short pair end read assembly - Scaffolding.

COMMANDS

- ABYSS_04PEMP
  Note: The MP reads will not be used to generate the contigs

```
## Select 50,000 reads (12,500 per pair) and 50,000 for
the mate pairs
seqtk sample -s1000 01_Illumina/Niben_PE250bp_CHL_R1.fastq
12500 > Niben_PE250bp_12K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_PE250bp_CHL_R2.fastq
12500 > Niben_PE250bp_12K_R2.fq
seqtk sample -s1000 01_Illumina/Niben_PE500bp_CHL_R1.fastq
12500 > Niben_PE500bp_12K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_PE500bp_CHL_R2.fastq
12500 > Niben_PE500bp_12K_R2.fq
seqtk sample -s1000 01_Illumina/Niben_MP02Kb_CHL_R1.fastq
25000 > Niben_MP02Kb_25K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_MP02Kb_CHL_R2.fastq
25000 > Niben_MP02Kb_25K_R1.fq
```

- Exercise 5.4: Short pair end read assembly - Scaffolding.

COMMANDS

- ABYSS_04PEMP

```
## Run ABYSS
abyss-pe name="NibenCHL_Ab04MP1" k=63 lib="pe250 pe500"
mp="mp02" pe250="Niben_PE250bp_CHL00025_R1.fastq
pe250="Niben_PE250bp_12K_R1.fq Niben_PE250bp_12K_R2.fq"
pe500="Niben_PE500bp_12K_R1.fq Niben_PE500bp_12K_R2.fq"
mp02=“Niben_MP02Kb_25K_R1.fq Niben_MP02Kb_25K_R2.fq"

## Get the stats
FastaSeqStats -i NibenCHL_Ab04MP1-scaffolds.fa >
NibenCHL_Ab04MP1-scaffolds.stats.txt
```

- Exercise 5.4: Short pair end read assembly - Scaffolding.

COMMANDS

- ABYSS_05PEMP
  Note: The MP reads will not be used to generate the contigs

```
## Use the same dataset than in the previous exercise plus
the following ones.
seqtk sample -s1000 01_Illumina/Niben_MP05Kb_CHL_R1.fastq
25000 > Niben_MP05Kb_25K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_MP05Kb_CHL_R2.fastq
25000 > Niben_MP05Kb_25K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_MP15Kb_CHL_R1.fastq
25000 > Niben_MP15Kb_25K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_MP15Kb_CHL_R2.fastq
25000 > Niben_MP15Kb_25K_R1.fq
```

- Exercise 5.4: Short pair end read assembly - Scaffolding.

COMMANDS

- ABYSS_05PEMP

```
## Run ABYSS
abyss-pe name="NibenCHL_Ab05MP2" k=63 lib="pe250 pe500"
mp="mp02 mp05 mp15" pe250="Niben_PE250bp_CHL00025_R1.fastq
pe250="Niben_PE250bp_12K_R1.fq Niben_PE250bp_12K_R2.fq"
pe500="Niben_PE500bp_12K_R1.fq Niben_PE500bp_12K_R2.fq"
mp02=“Niben_MP02Kb_25K_R1.fq Niben_MP02Kb_25K_R2.fq”
mp05=“Niben_MP05Kb_25K_R1.fq Niben_MP05Kb_25K_R2.fq”
mp15=“Niben_MP15Kb_25K_R1.fq Niben_MP15Kb_25K_R2.fq"

## Get the stats
FastaSeqStats -i NibenCHL_Ab05MP2-scaffolds.fa >
NibenCHL_Ab05MP2-scaffolds.stats.txt
```

- Exercise 5.4: Short pair end read assembly - Scaffolding.

GROUP DISCUSSION

Discuss the use of different combination of pair ends and mate pair reads to generate different assemblies.

- Exercise 5.1: Get read dataset stats.
- Exercise 5.2: Short single read assembly - Coverage.
- Exercise 5.3: Short single read assembly - Kmer.
- Exercise 5.4: Short pair end read assembly - Scaffolding.
- Exercise 5.5: Long read assembly.
- Exercise 5.6: Assembly evaluation - Coverage.
- Exercise 5.7: Assembly evaluation - Variants.
- Exercise 5.8: Assembly structural annotation.

- Exercise 5.5: Long read assembly.

OBJECTIVE: To run different a assembly using PacBio reads and compare with the previous assembly made with Illumina reads (NibenCHL_Ab05MP2-scaffolds.fa). You will select the longest sequence of both assemblies, copy in your computer with FileZilla and Blast them.

TOOLS: Canu, FastaExtract

COMMANDS:

```
## Run a canu assembly
canu -d NibenCHL_canu01 -p NibenCHL genomeSize=153k -
pacbio-raw 02_PacBio/Niben_PB_rawreads.fastq
```

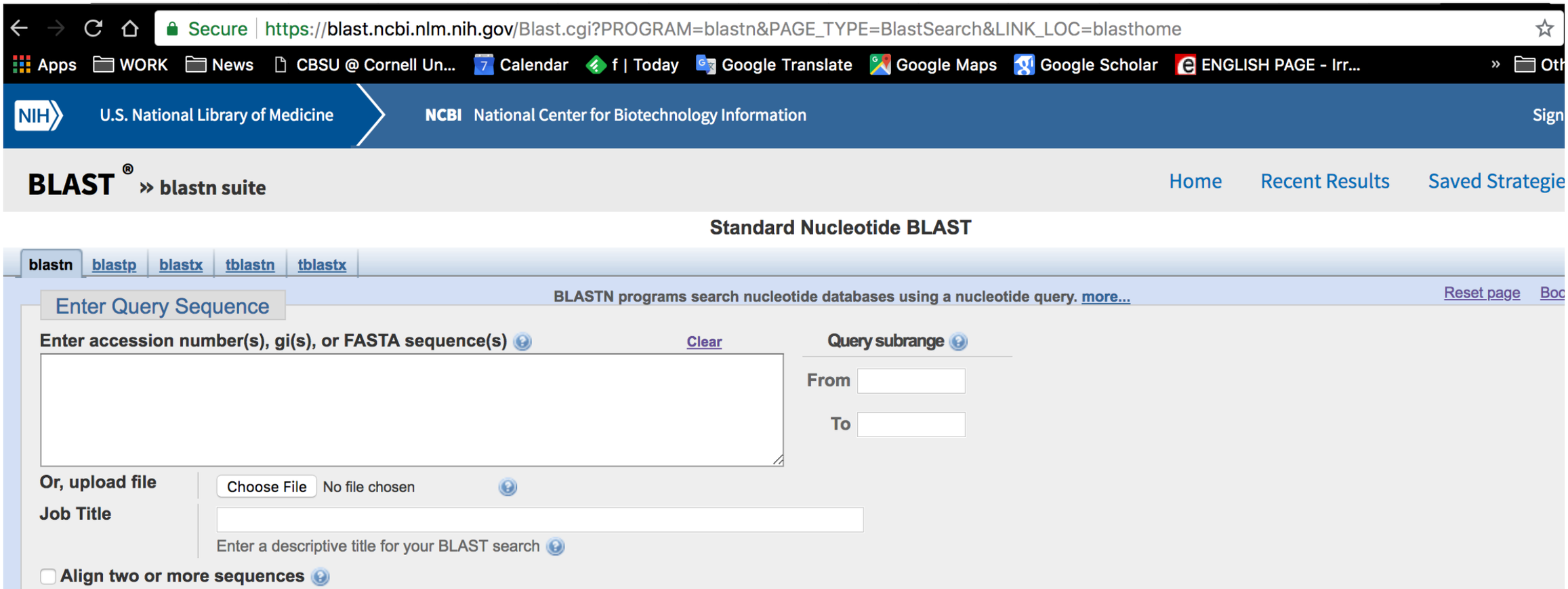
- Exercise 5.5: Long read assembly.

COMMANDS:

```
## Get the stats
FastaSeqStats -i NibenCHL_canu01/NibenCHL.contigs.fasta >
NibenCHL.contigs.stats.txt

## Select the longest canu assembled contig
FastaExtract -f NibenCHL_canu01/NibenCHL.contigs.fasta -l
100000 -o NibenCHL_canu01.longest.fa

## Select the longest ABYSS assembled contig
FastaExtract -f ../Exercise04/NibenCHL_Ab05MP2-scaffolds.fa
-l 100000 -o NibenCHL_abyss05.longest.fa
```

- Exercise 5.5: Long read assembly.

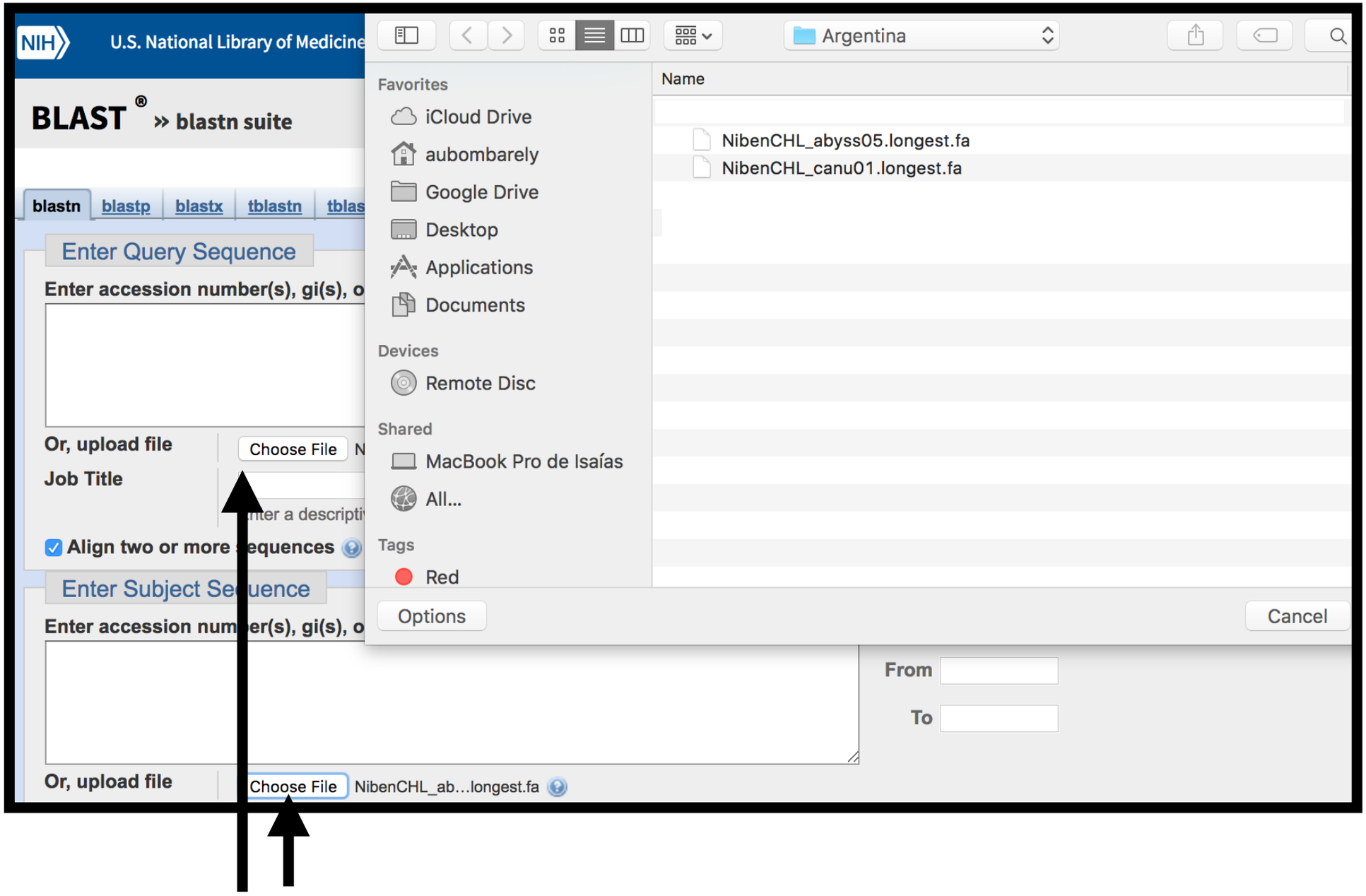
https://blast.ncbi.nlm.nih.gov/Blast.cgi



Click “Align two or more sequences”

- Exercise 5.5: Long read assembly.

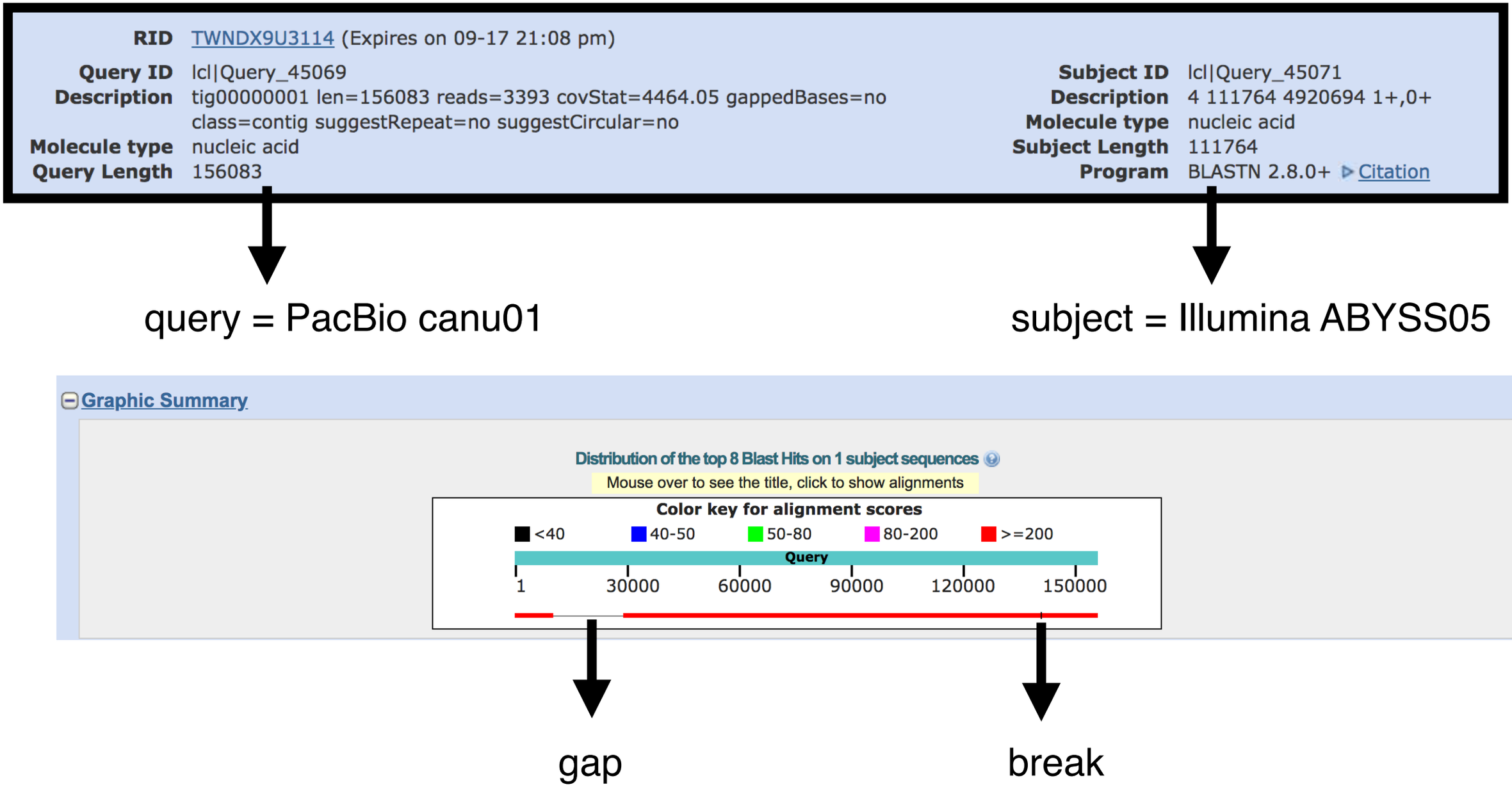
https://blast.ncbi.nlm.nih.gov/Blast.cgi



Upload both sequences and run the blast

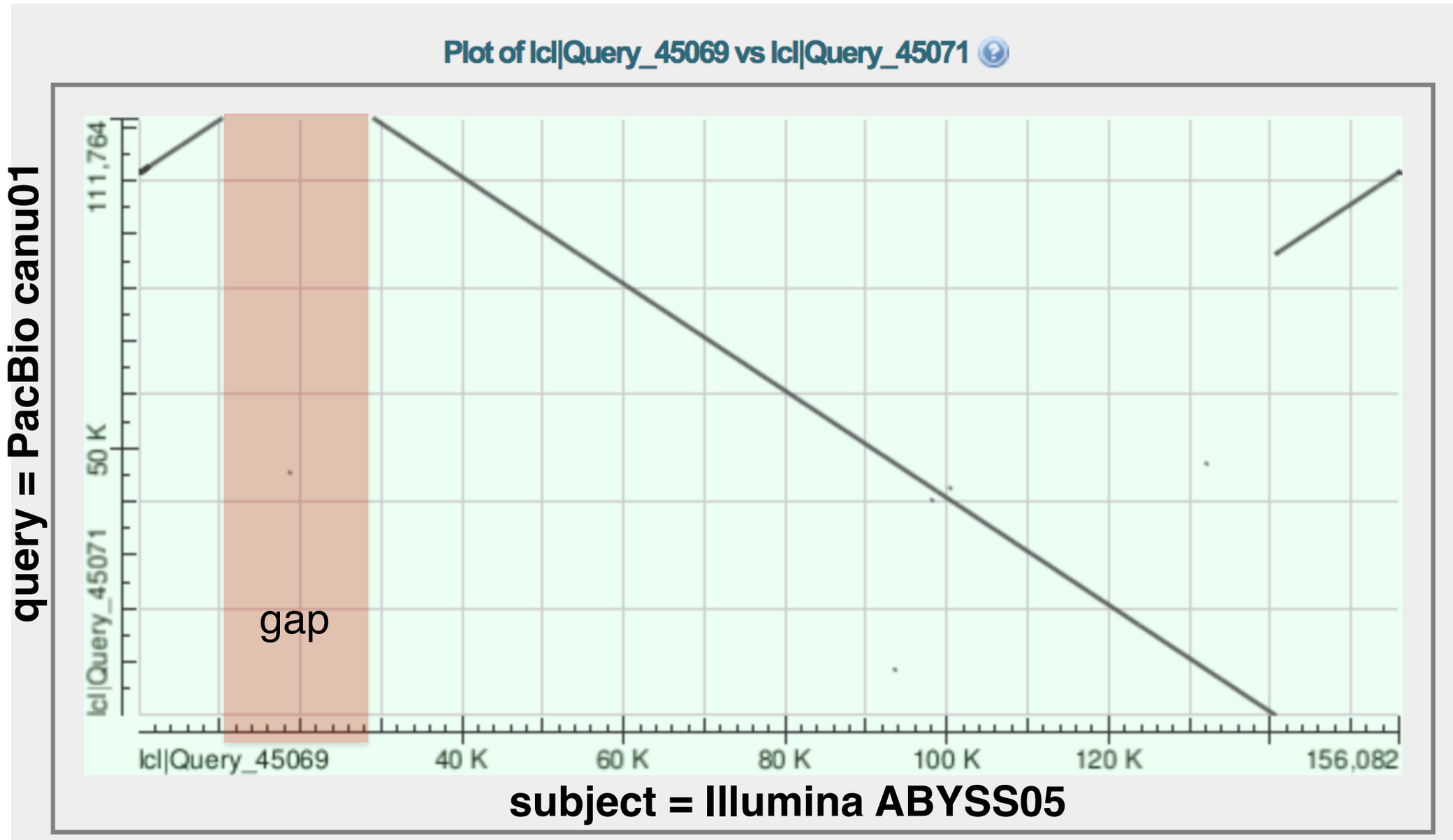
- Exercise 5.5: Long read assembly.

https://blast.ncbi.nlm.nih.gov/Blast.cgi



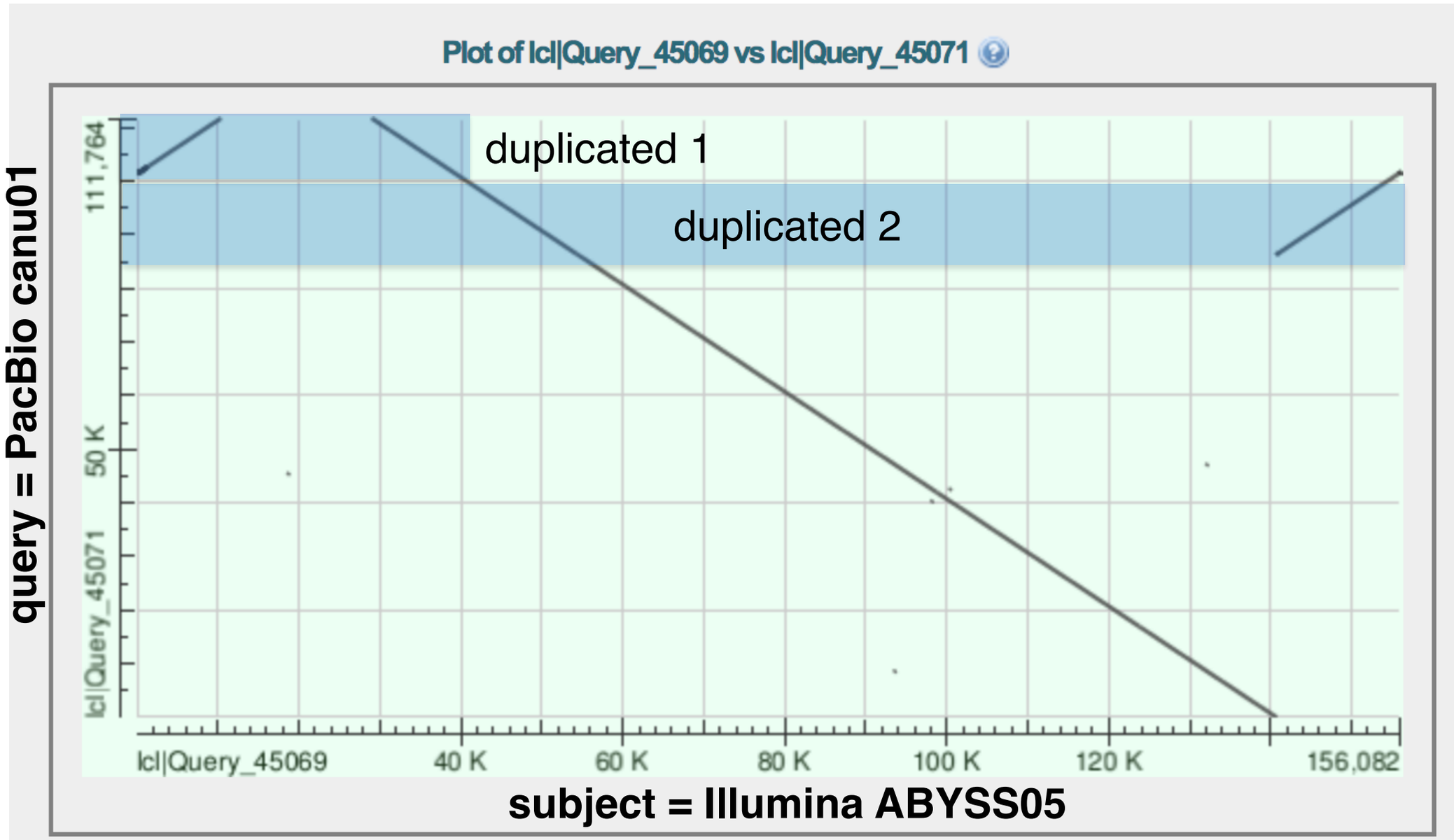
- Exercise 5.5: Long read assembly.

https://blast.ncbi.nlm.nih.gov/Blast.cgi



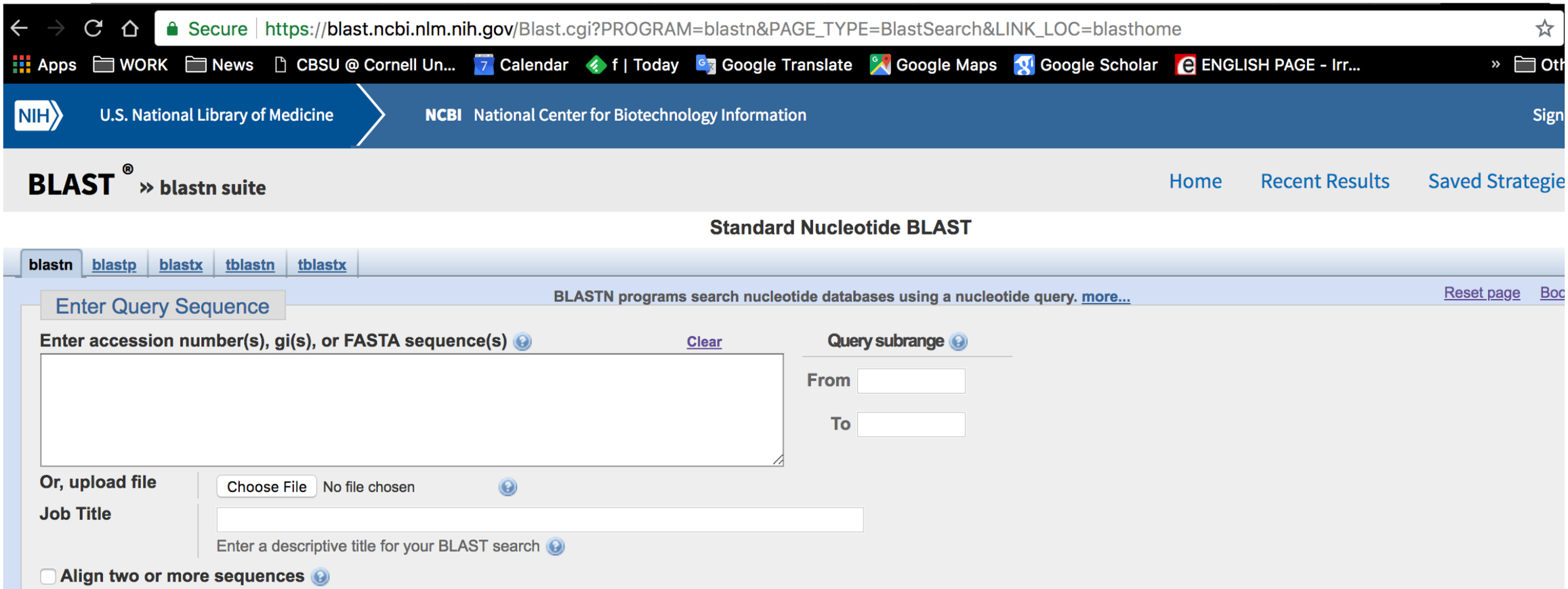
- Exercise 5.5: Long read assembly.

https://blast.ncbi.nlm.nih.gov/Blast.cgi



- Exercise 5.5: Long read assembly.

https://blast.ncbi.nlm.nih.gov/Blast.cgi



Run the canu assembly against the whole Nucleotide Collection NR Genbank database

- Exercise 5.5: Long read assembly.

RID TWP12V7V015 (Expires on 09-17 21:18 pm)

Query ID lcl|Query_126885
Description tig00000001 len=156083 reads=3393 covStat=4464.05 gappedBases=no class=contig suggestRepeat=no suggestCircular=no
Molecule type nucleic acid
Query Length 156083

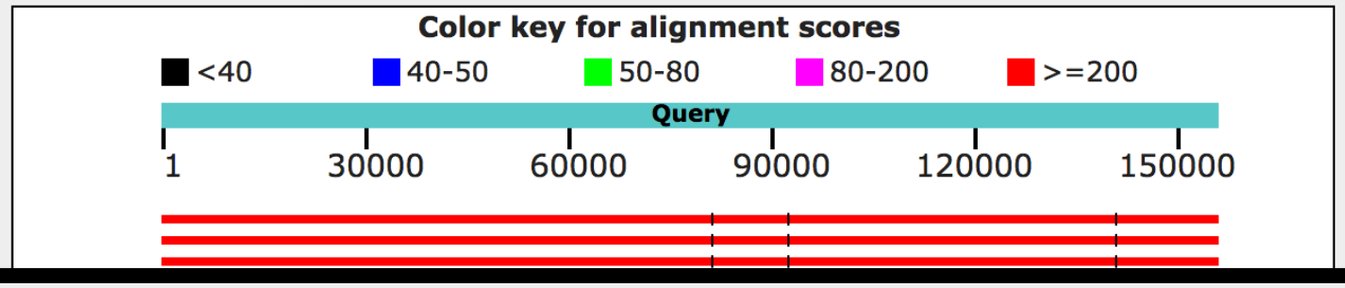
Database Name nr
Description Nucleotide collection (nt)
Program BLASTN 2.8.0+ Citation

Other reports: Search Summary [Taxonomy reports] [Distance tree of results] [MSA viewer]

Graphic Summary

Distribution of the top 200 Blast Hits on 100 subject sequences

Mouse over to see the title, click to show alignments



Sequences producing significant alignments:

Select: All None Selected:0

Alignments Download GenBank Graphics Distance tree of results

Description	Max score	Total score	Query cover	E value	Ident	Accession
Nicotiana tabacum cultivar TN90 plastid, complete genome	1.465e+05	3.749e+05	99%	0.0	99%	KU199713.1
Nicotiana tabacum chloroplast genome DNA	1.465e+05	3.747e+05	99%	0.0	99%	Z00044.2
Nicotiana sylvestris chloroplast DNA, complete sequence						

Nicotiana tabacum cultivar TN90 plastid, complete genome

Sequence ID: KU199713.1 **Length:** 155992 **Number of Matches:** 21

- Exercise 5.1: Get read dataset stats.
- Exercise 5.2: Short single read assembly - Coverage.
- Exercise 5.3: Short single read assembly - Kmer.
- Exercise 5.4: Short pair end read assembly - Scaffolding.
- Exercise 5.5: Long read assembly.
- Exercise 5.6: Assembly evaluation - Coverage.
- Exercise 5.7: Assembly evaluation - Variants.
- Exercise 5.8: Assembly structural annotation.

- Exercise 5.6: Assembly evaluation - Coverage.

OBJECTIVE: To map Illumina and PacBio reads back to the PacBio assembly and evaluate the coverage.

TOOLS: Bowtie2, ngmlr, Samtools, Bedtools, R

- Exercise 5.6: Assembly evaluation - Coverage.

COMMANDS:

```
##Copy the reference file
cp NibenCHL_canu01.longest.fa NibenCHL_canu01.longest_ref.fa

## Create the index using Bowtie2-Build
bowtie2-build NibenCHL_canu01.longest_ref.fa
NibenCHL_canu01.longest_ref

## Create a 200X dataset: Genome size ~ 150Kb x 200 =
30,000Kb = 30 Mb / 150 = 200,000 reads (100,000 pairs).
seqtk sample -s1000 01_Illumina/Niben_PE250bp_CHL_R1.fastq
100000 > Niben_PE250bp_100K_R1.fq
seqtk sample -s1000 01_Illumina/Niben_PE250bp_CHL_R2.fastq
100000 > Niben_PE250bp_100K_R2.fq

## Map the reads and filter the output with samtools
bowtie2 -p 8 -x NibenCHL_canu01.longest_ref -1
Niben_PE250bp_100K_R1.fq -2 Niben_PE250bp_100K_R2.fq |
samtools view -Sb -F 4 -o Niben_PE250bp.map.canu01.bam -
```

- Exercise 5.6: Assembly evaluation - Coverage.

COMMANDS:

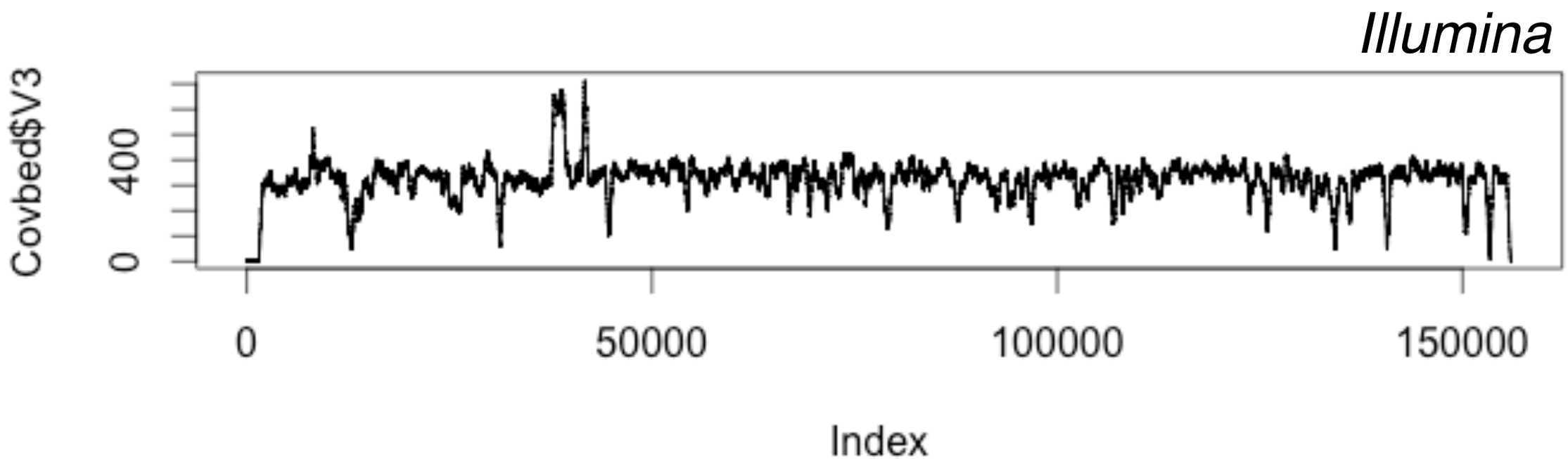
```
##Sort the mapped reads by position
samtools sort -o Niben_PE250bp.map.canu01.bam
Niben_PE250bp.map.canu01.bam

##Create a fasta index file for bedtools
samtools faidx NibenCHL_canu01.longest_ref.fa

##Run bedtools and get the coverage per position
bedtools genomecov -d -ibam Niben_PE250bp.map.canu01.bam -g
NibenCHL_canu01.longest_ref.fa.fai >
Niben_PE250bp.map.canu01.covbed.txt
```

- Exercise 5.6: Assembly evaluation - Coverage.

<u>To visualize the coverage</u>:
1- Copy reference fasta and the bam into your computer and use IGV or Tablet.
2- Copy the bedtools coverage file in your computer and use R plot to visualize it.

```
## R commands
Covbed <-
read.delim("Niben_PE250bp.map.canu01.covbed.txt",
header=FALSE)
plot(Covbed$V3, type="l")
```

- Exercise 5.6: Assembly evaluation - Coverage.



- Exercise 5.6: Assembly evaluation - Coverage.

COMMANDS:

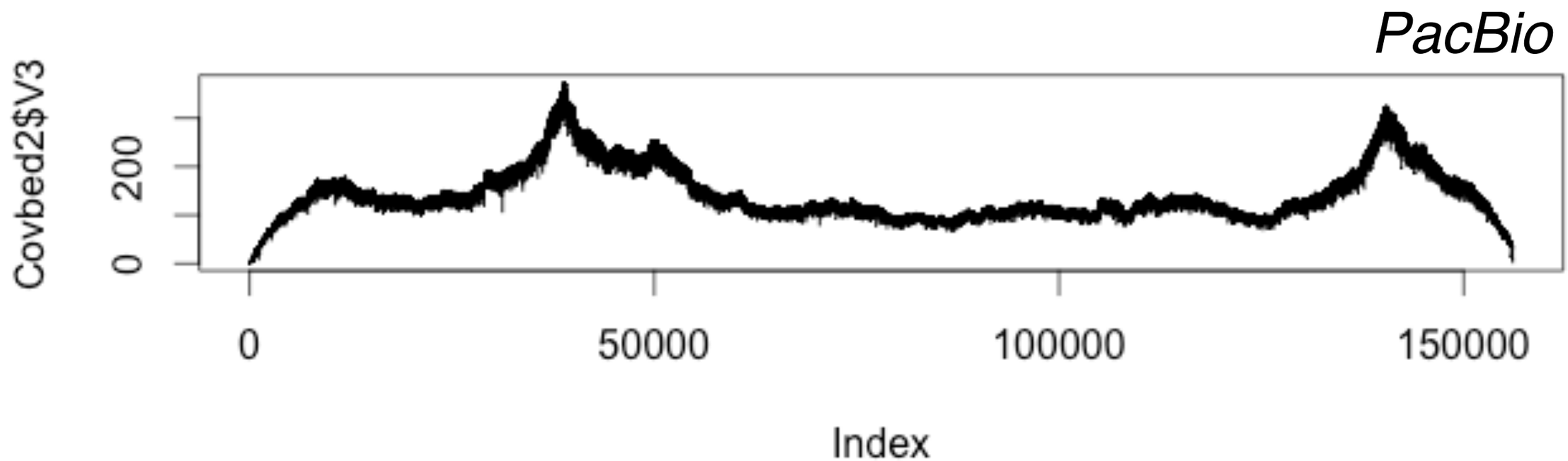
```
## Select 200X of PacBio reads. The average read size is
6.6Kb. Genome size ~ 150Kb x 200 = 30,000Kb / 6.6Kb = 4,545
seqtk sample -s1000 ../../00_data/02_PacBio/
Niben_PB_rawreads.fastq 4545 > Niben_PB_5K_R1.fq

##Map the PacBio reads with ngmlr
ngmlr -t 4 -r NibenCHL_canu01.longest_ref.fa -q
Niben_PB_5K_R1.fq | samtools view -Sb -F 4 -o
Niben_PB.map.canu01.bam -

##Sort the reads
samtools sort -o Niben_PB.map.canu01.bam
Niben_PB.map.canu01.bam

##Run bedtools and get the coverage per position
bedtools genomecov -d -ibam Niben_PB.map.canu01.bam -g
NibenCHL_canu01.longest_ref.fa.fai >
Niben_PB.map.canu01.covbed.txt
```

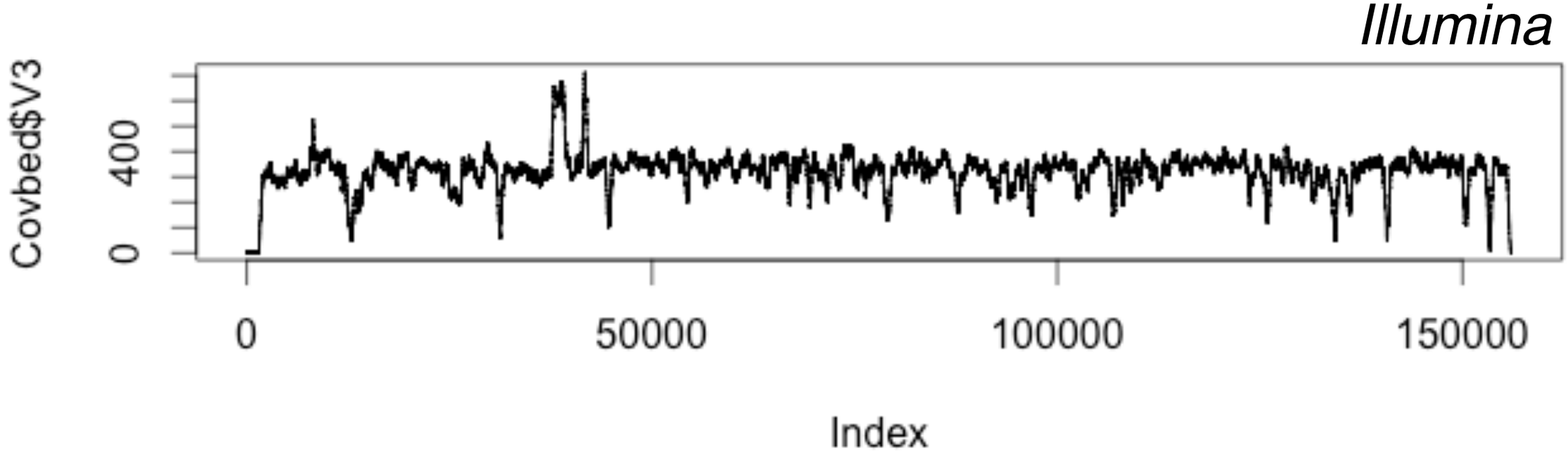
- Exercise 5.6: Assembly evaluation - Coverage.

<u>To visualize the coverage</u>:
1- Copy reference fasta and the bam into your computer and use IGV or Tablet.
2- Copy the bedtools coverage file in your computer and use R plot to visualize it.

```
## R commands
Covbed2 <-
read.delim("Niben_PB.map.canu01.covbed.txt",
header=FALSE)
plot(Covbed2$V3, type="l")
```

- Exercise 5.6: Assembly evaluation - Coverage.





- Exercise 5.1: Get read dataset stats.
- Exercise 5.2: Short single read assembly - Coverage.
- Exercise 5.3: Short single read assembly - Kmer.
- Exercise 5.4: Short pair end read assembly - Scaffolding.
- Exercise 5.5: Long read assembly.
- Exercise 5.6: Assembly evaluation - Coverage.
- Exercise 5.7: Assembly evaluation - Variants.
- Exercise 5.8: Assembly structural annotation.

- Exercise 5.7: Assembly evaluation - Variants

<u>OBJECTIVE:</u> Call the variants over the previously mapped Illumina reads.

<u>TOOLS:</u> Freebayes, R

- Exercise 5.7: Assembly evaluation - Variants

COMMANDS:

```
##Index the Bam file
samtools index Niben_PE250bp.map.canu01.bam

## Call the variants over the Illumina BAM file
freebayes -v Niben_PE250bp.map.canu01.vcf -b
Niben_PE250bp.map.canu01.bam -f
NibenCHL_canu01.longest_ref.fa

## Get some stats from the VCF
## 1- Count total variants
grep -v "#" Niben_PE250bp.map.canu01.vcf | wc -l

## 2- Count variants by type
grep -v "#" Niben_PE250bp.map.canu01.vcf | cut -f8 | sed -r
's/.+;TYPE=//' | sort | uniq -c

## 3- Count non homozygous variants
grep -v "#" Niben_PE250bp.map.canu01.vcf | cut -f8,10 | sed -
r 's/.+;TYPE=//' | sed -r 's/:.+//'
```

- Exercise 5.1: Get read dataset stats.
- Exercise 5.2: Short single read assembly - Coverage.
- Exercise 5.3: Short single read assembly - Kmer.
- Exercise 5.4: Short pair end read assembly - Scaffolding.
- Exercise 5.5: Long read assembly.
- Exercise 5.6: Assembly evaluation - Coverage.
- Exercise 5.7: Assembly evaluation - Variants.
- Exercise 5.8: Assembly structural annotation.

- Exercise 5.8: Assembly structural annotation.

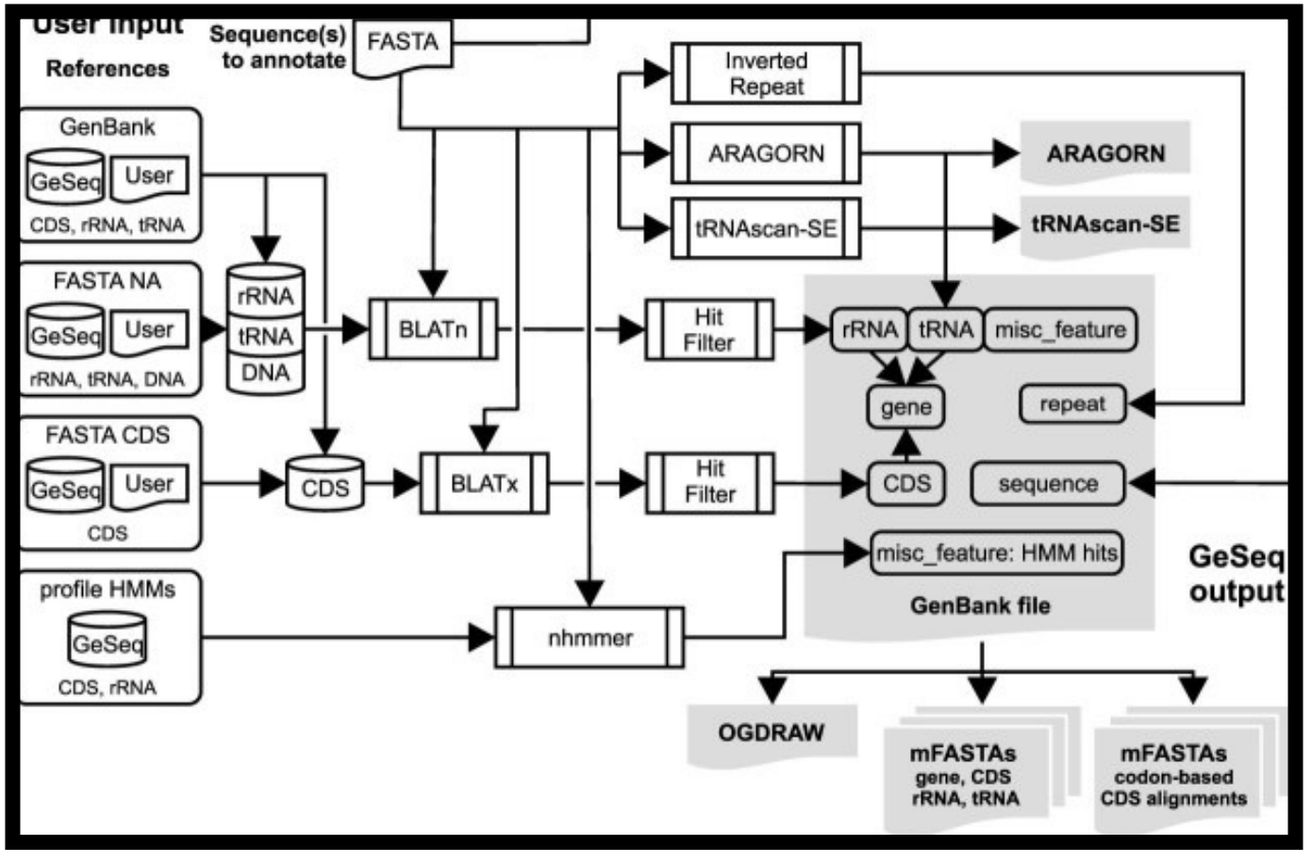
https://chlorobox.mpimp-golm.mpg.de/geseq.html

Nucleic Acids Res. 2017 Jul 3; 45(Web Server issue): W6–W11.
Published online 2017 May 9. doi: 10.1093/nar/gkx391

PMCID: PMC5570176
PMID: 28486635

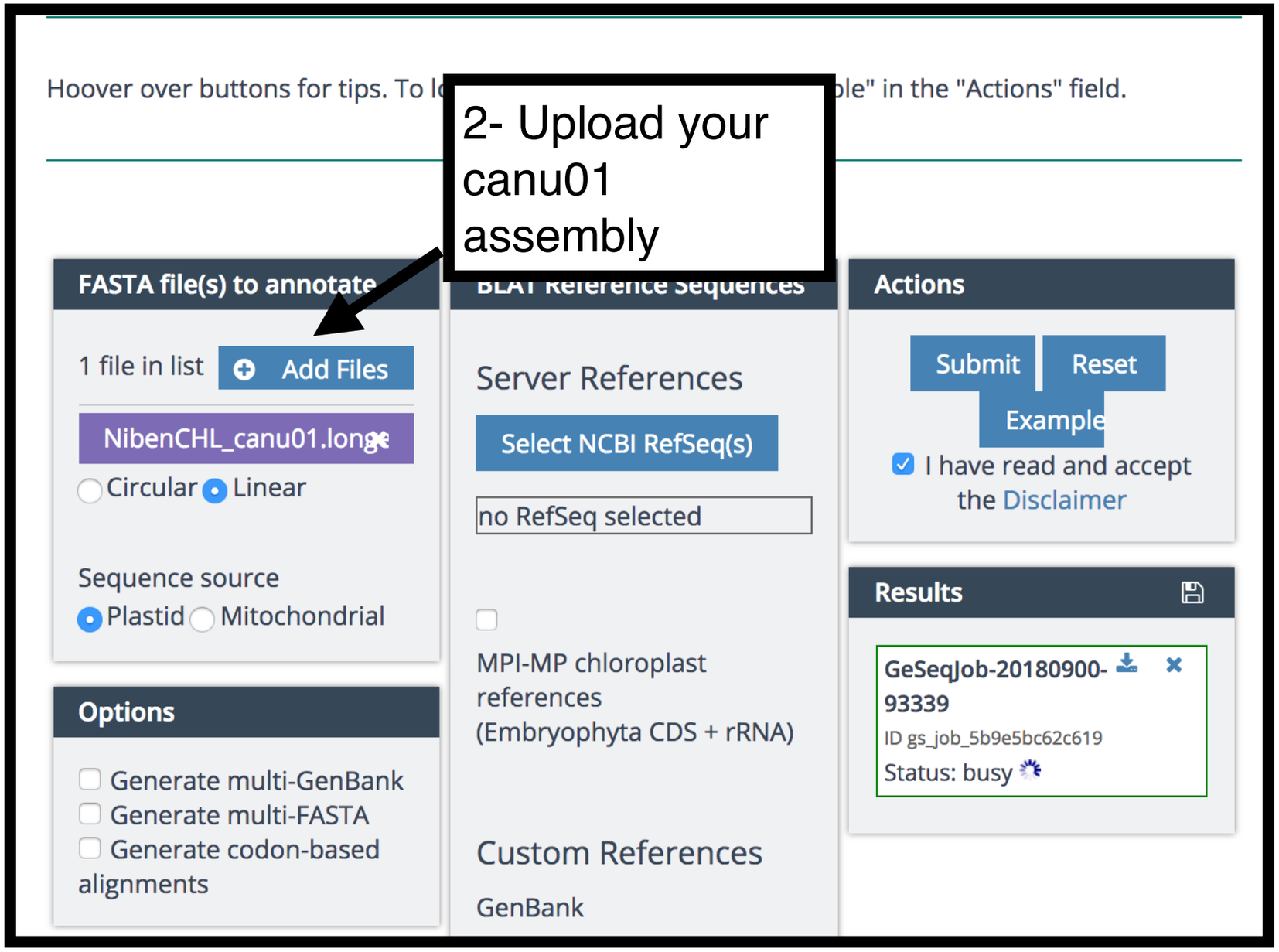
GeSeq – versatile and accurate annotation of organelle genomes

Michael Tillich,[1] Pascal Lehwark,[2] Tommaso Pellizzer,[1] Elena S. Ulbricht-Jones,[1] Axel Fischer,[1] Ralph Bock,[1] and Stephan Greiner[1]

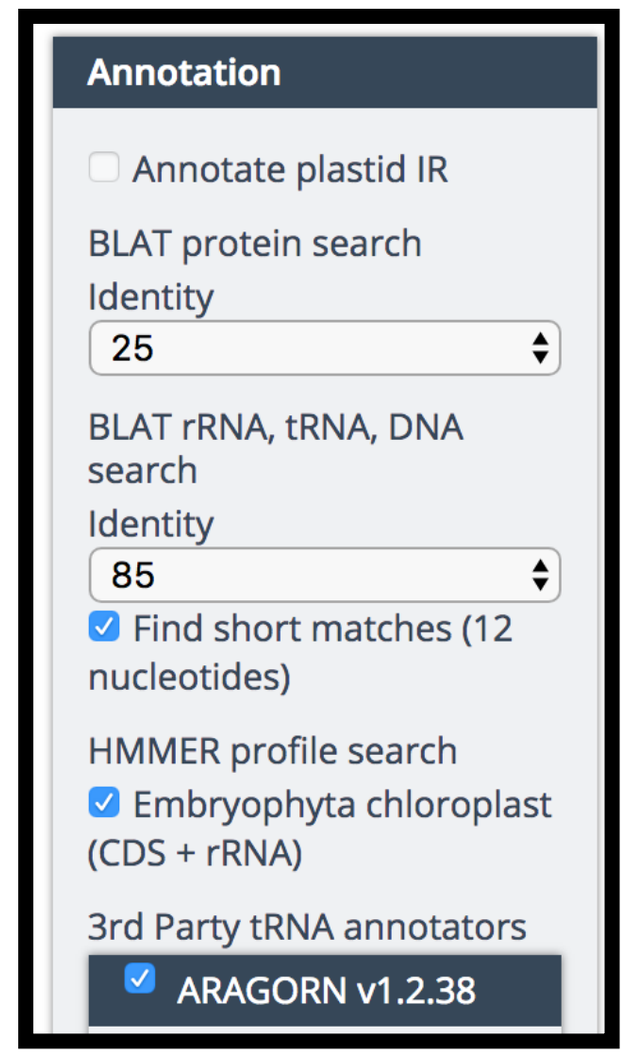


- Exercise 5.8: Assembly structural annotation.

https://chlorobox.mpimp-golm.mpg.de/geseq.html



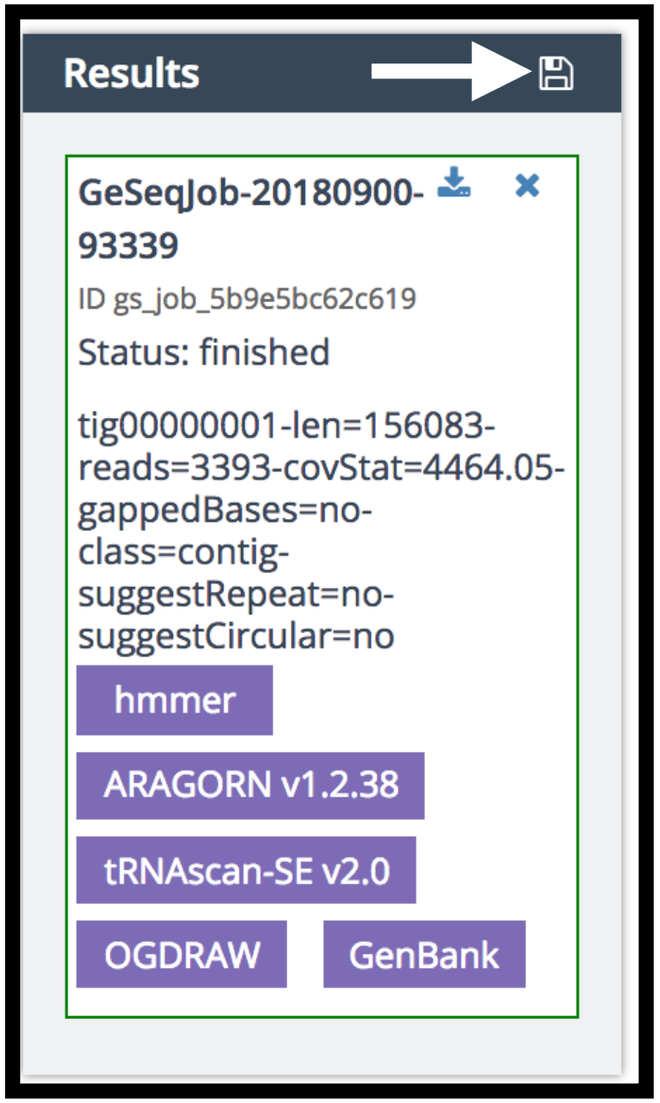
1- Set up the annotation boxes

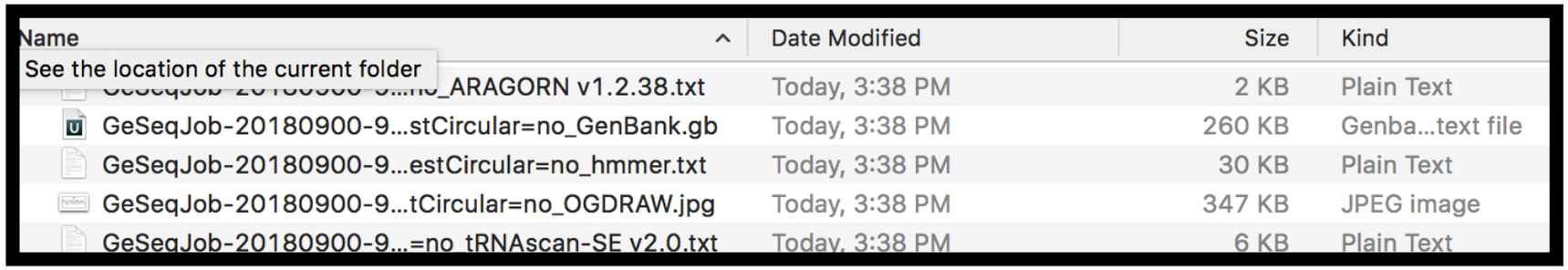


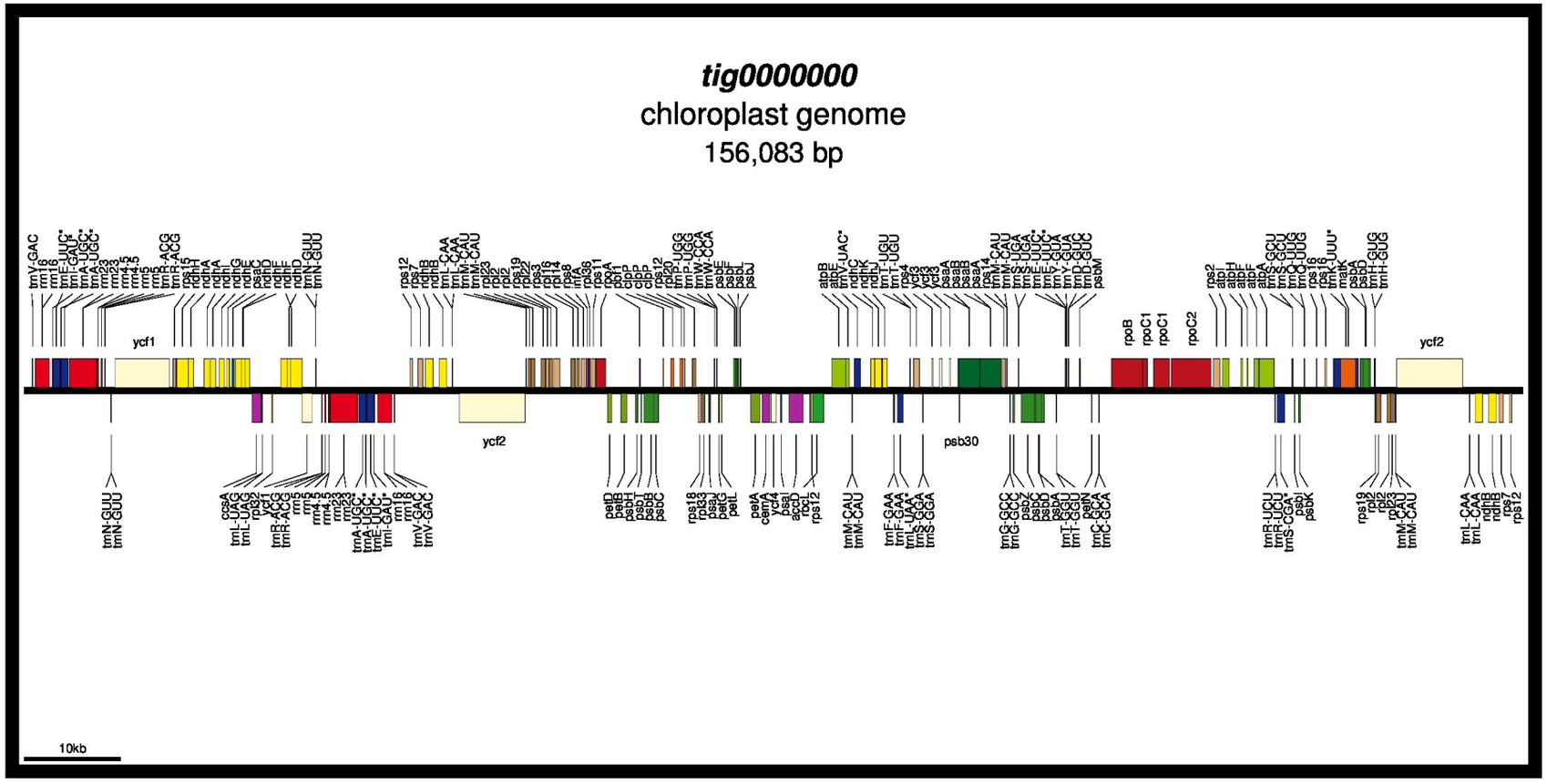
- Exercise 5.8: Assembly structural annotation.

https://chlorobox.mpimp-golm.mpg.de/geseq.html

3- Download the results



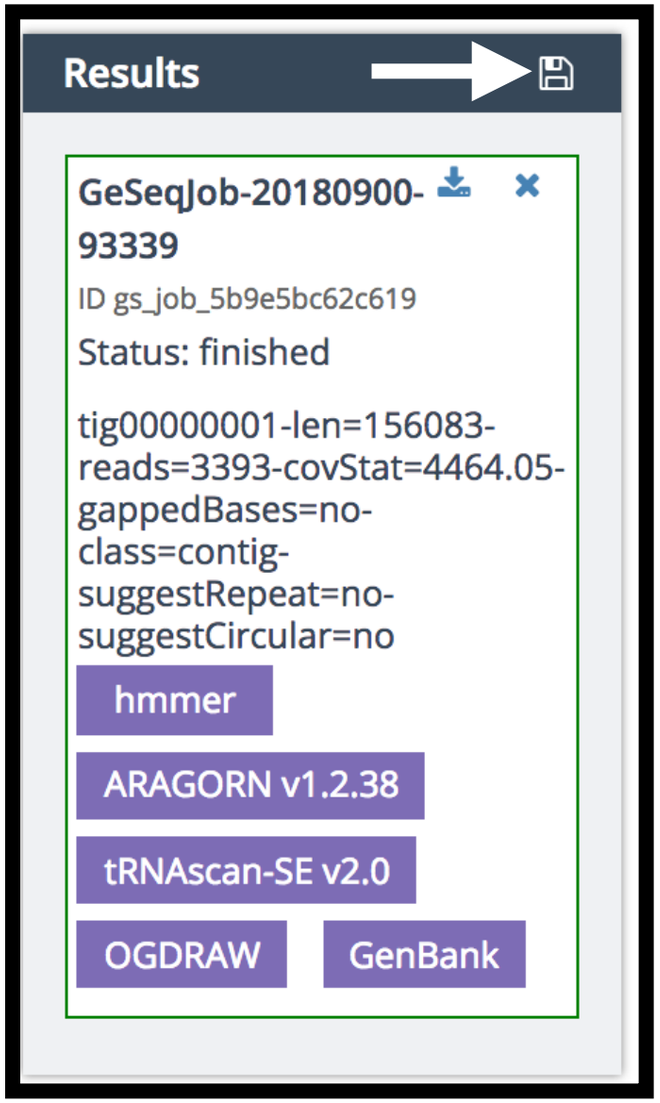


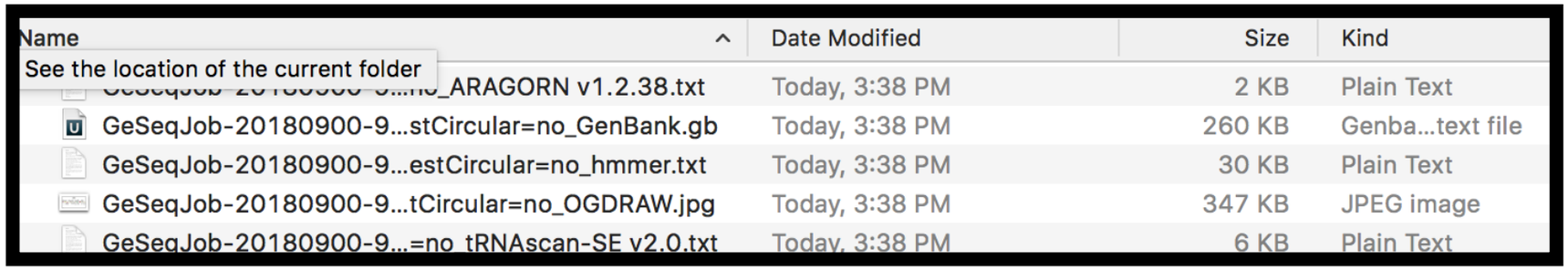


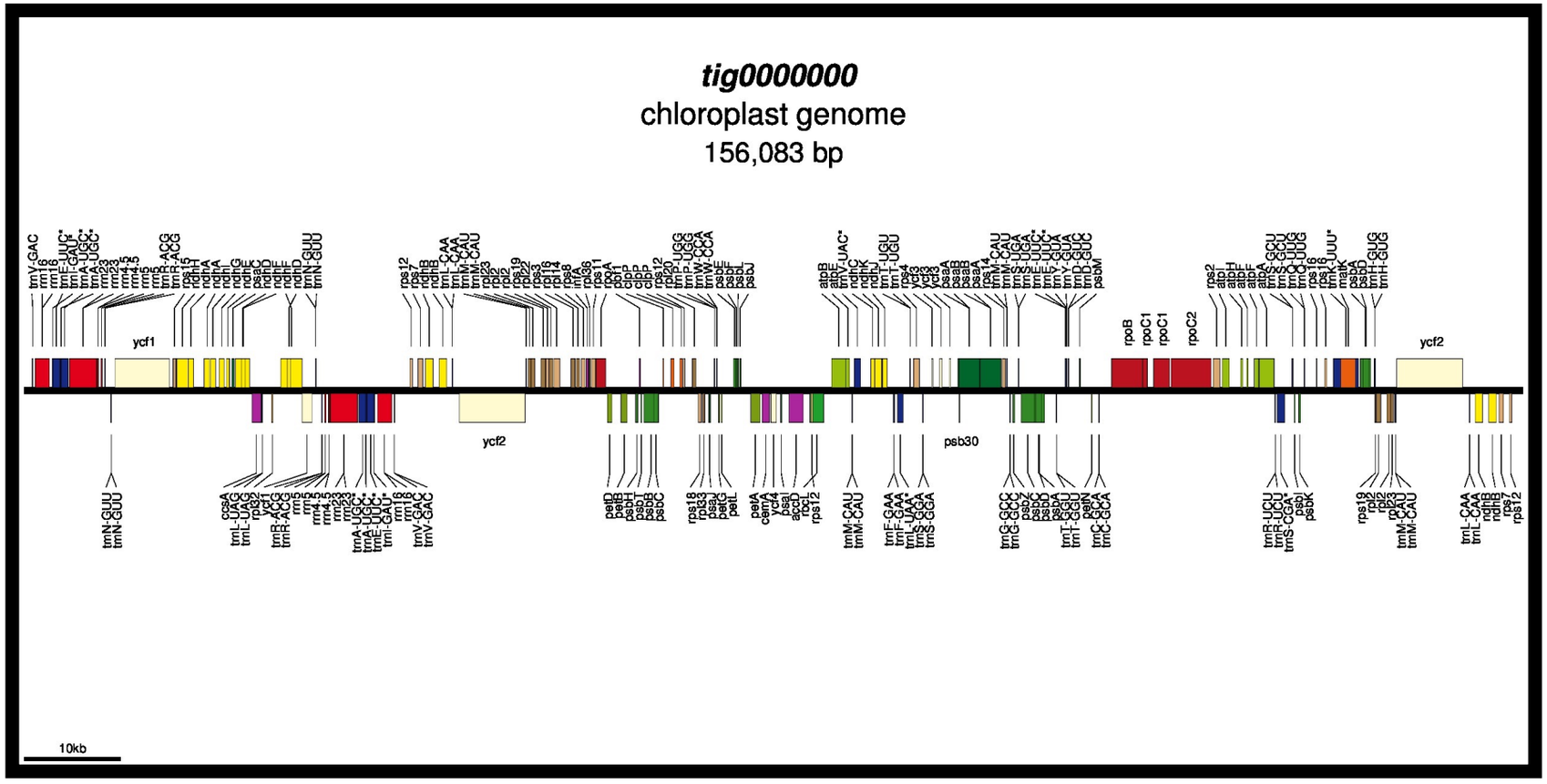
- Exercise 5.8: Assembly structural annotation.

https://chlorobox.mpimp-golm.mpg.de/geseq.html

3- Download the results







Notes about sending the results

Please, send the results as two files:

- Discussion_ExerciseX.X_MyLastName.txt
- Script_ExerciseX.X_MyLastName.sh

In the subject of the email write “GT2020 - Solution Exercise X.X” to facilitate the use of search tools in the email inbox.

Deadlines:

- Exercises 5.1 to 5.3 - March 31, 2020
- Exercises 5.4 to 5.6 - April 3, 2020
- Exercises 5.7 and 5.8 - April 6, 2020