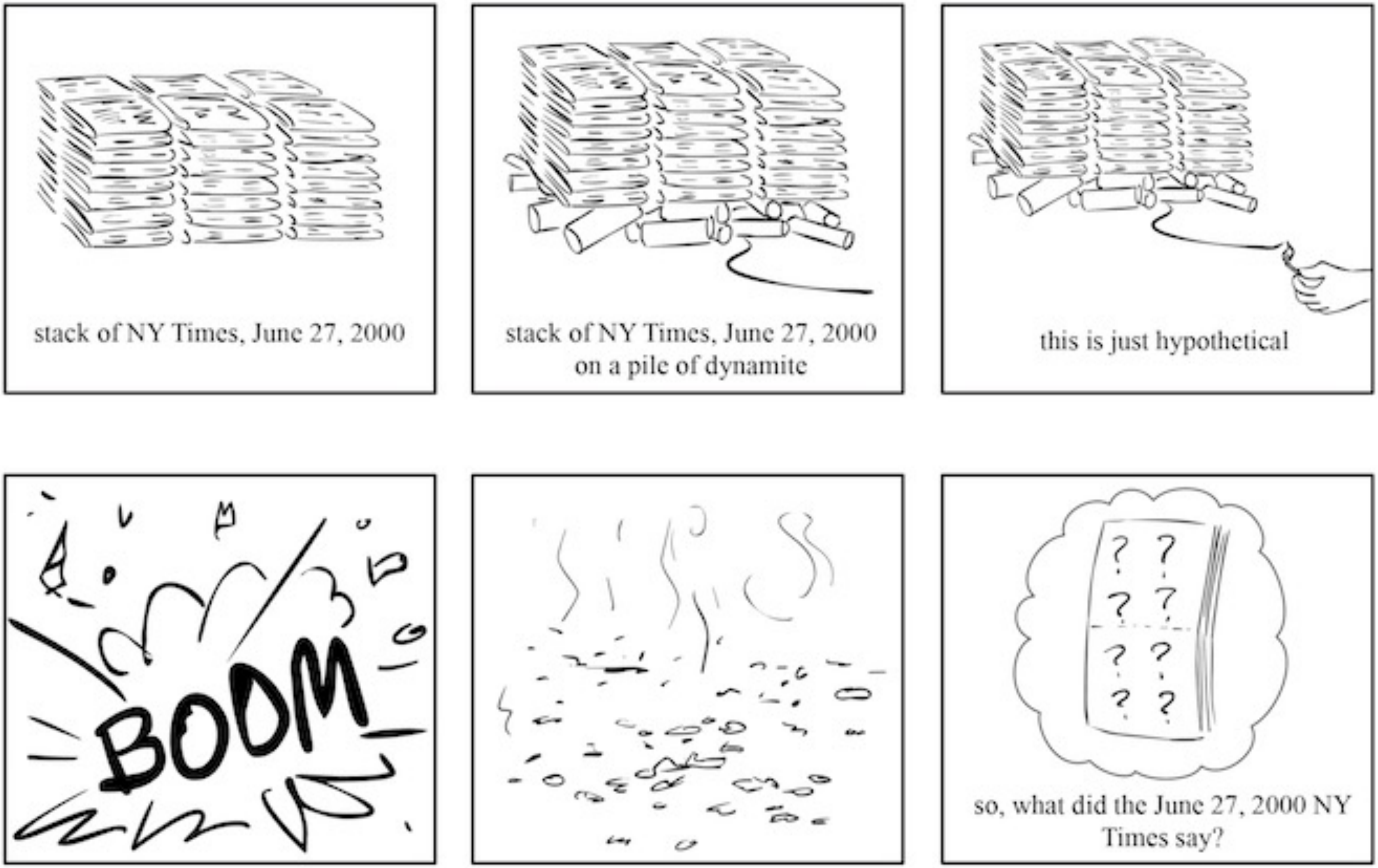
Topic 5: de novo assembly

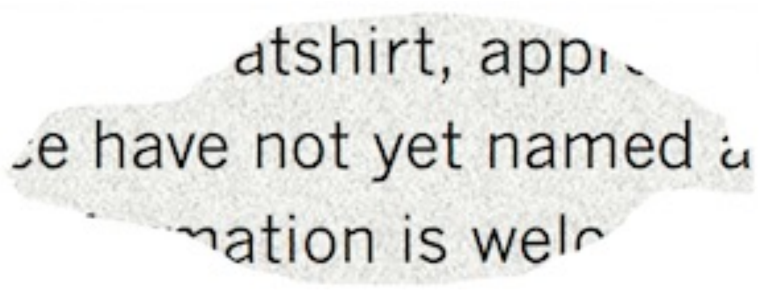
Outcomes

- Identify the difference between de novo assembly and reference guided alignment
- Evaluate two different approaches to de novo genome assembly
- Describe how repetitive elements can hamper proper assembly and compare approaches that can overcome this problem
- Describe approaches for transcriptome/GBS de novo assembly

Introduction


stack of NY Times, June 27, 2000
stack of NY Times, June 27, 2000 on a pile of dynamite
this is just hypothetical
BOOM
? ? ? ? ? ? ? ?
so, what did the June 27, 2000 NY Times say?

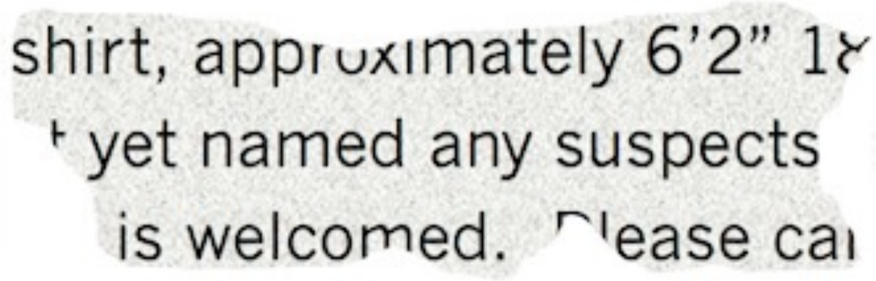
Introduction



atshirt, appr

e have not yet named a

mation is welc

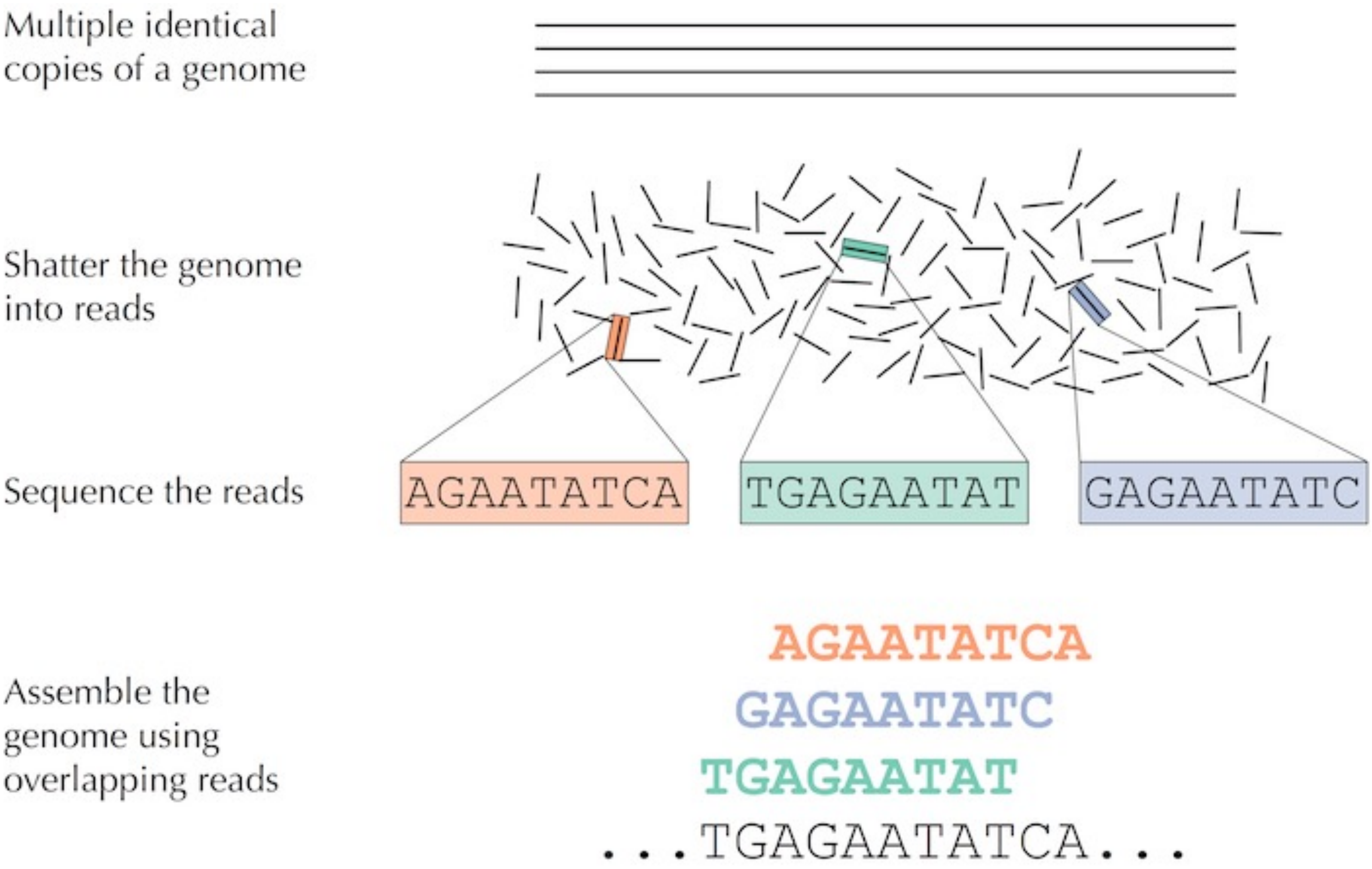


shirt, approximately 6’2” 18

t yet named any suspects

is welcomed. Please cal

Introduction



Multiple identical copies of a genome

Shatter the genome into reads

Sequence the reads

Assemble the genome using overlapping reads

Alignment vs assembly

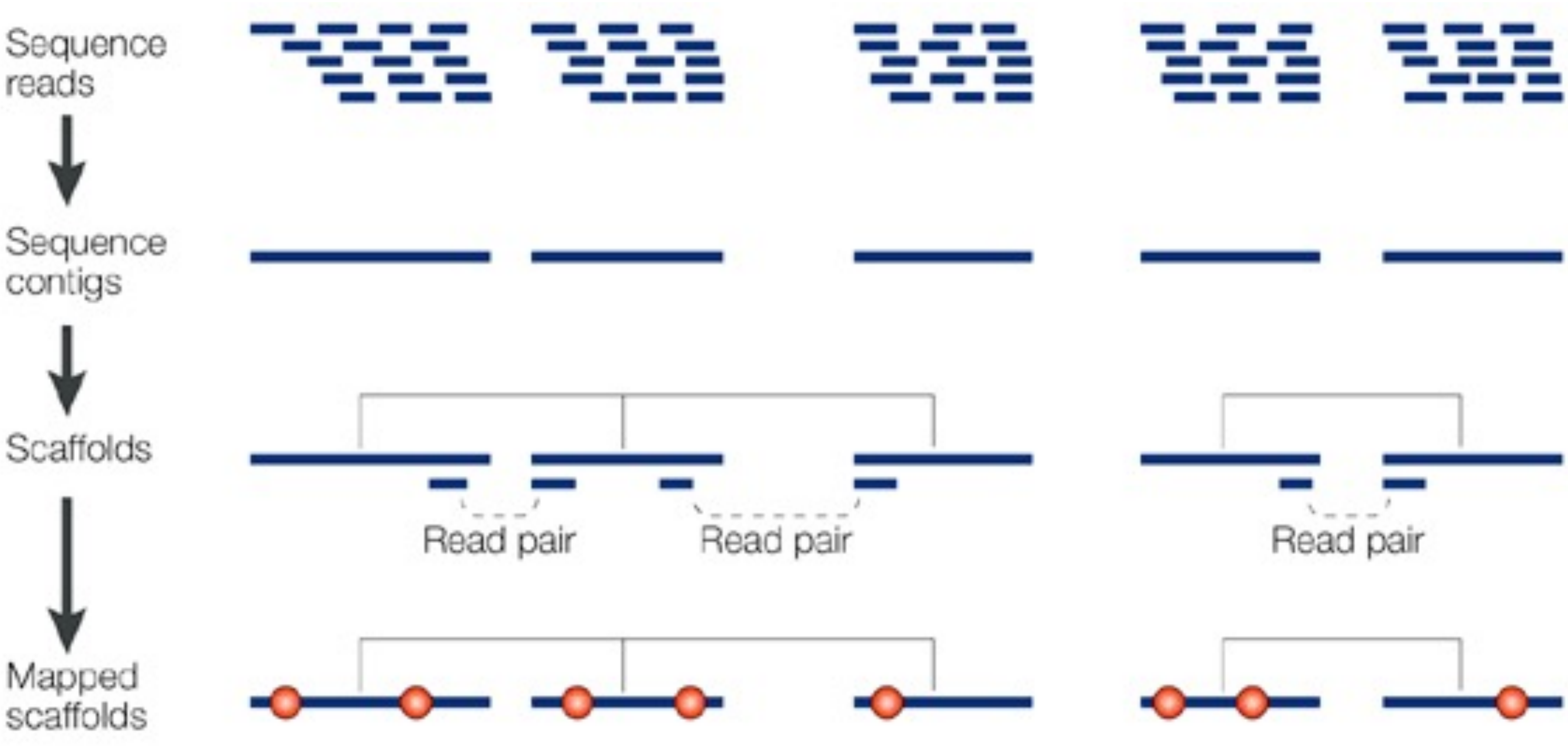
Aligning to a reference:

- Reference guided alignments: align the reads to a reference genome and looks for differences

Building a reference:

- *De novo* assembly: no previous genome assembly is used
- Comparative genome assembly: assemble a newly sequenced genome by mapping it on to a reference
- Hybrid approach: reference-guided and *de novo* for unused reads or *de novo* and then reference guided alignments

Introduction



Adams 2008

Introduction

Original sequence

GATAGAAGGGTCCGCTCGCTCAGCTACCGGTTTTTATAGATCTA

GATAGAAGGGTCCGCT
AGAAGGGTCCGCTC
GGGTCCGCTCGCTCA
CCGCTCGCTCAGC
CTCGCTCAGCTACC
TCAGCTACCGGTTT
CTACCGGTTTTT
AGCTACCGGTTTTTAT
TTTTTATAGATCTA

fragmented sequences
from sequencer
(reads)

Introduction

assembled

fragmented sequences
from sequencer
(reads)

```
                              TTTTTATAGATCTA
                     AGCTACCGGTTTTTAT
                    CAGCTACCGGTTTTT
                   TCAGCTACCGGTTT
               CTCGCTCAGCTACC
            CCGCTCGCTCAGC
       GGGTCCGCTCGCTCA
   AGAAGGGTCCGCTC
GATAGAAGGGTCCGCT
```

GATAGAAGGGTCCGCTCGCTCAGCTACCGGTTTTTATAGATCTA

We want to reconstruct this from the reads

Introduction

Simplified scenario

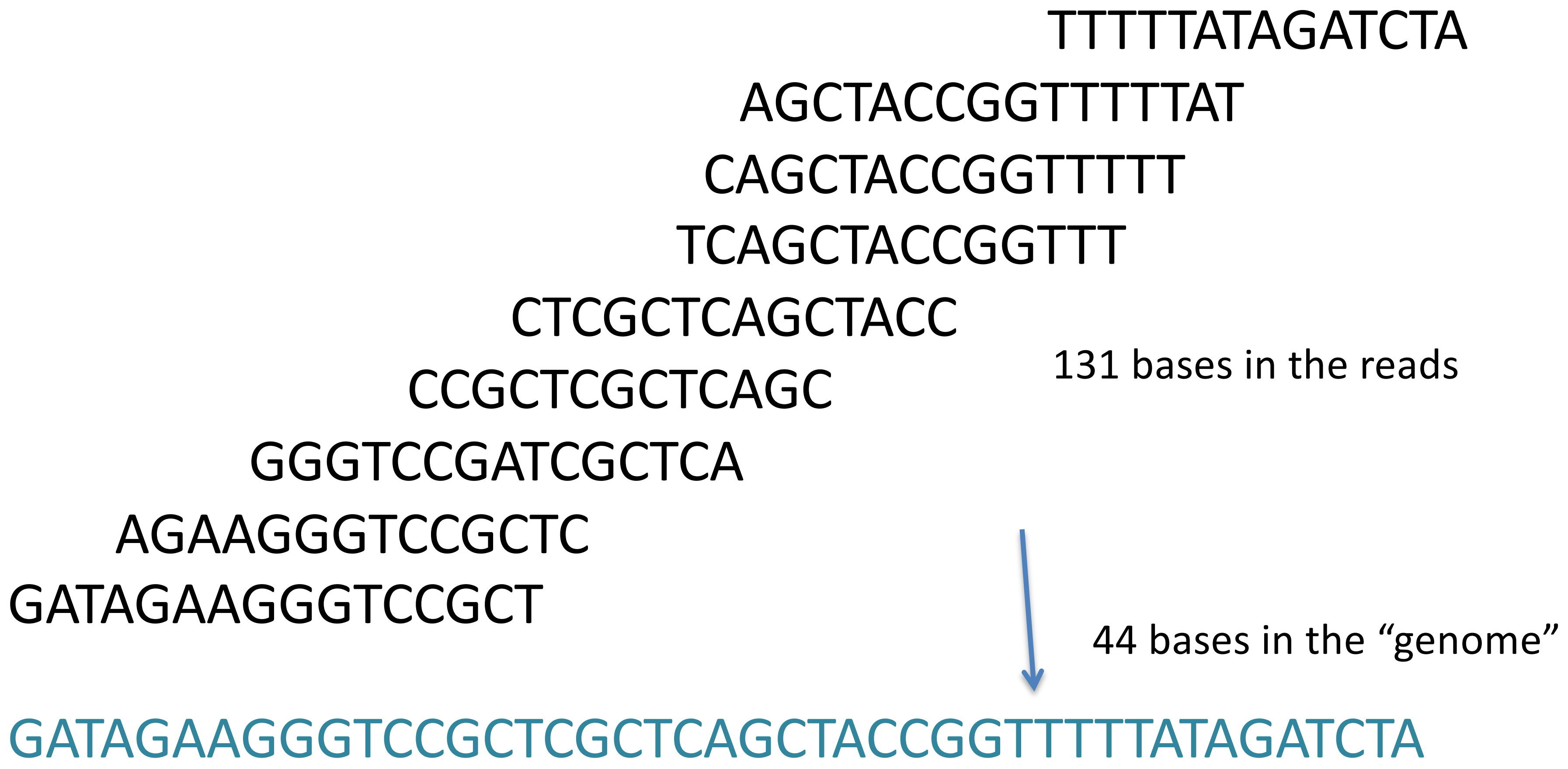
- Single strand
- Error free
- Complete coverage

```
                              TTTTTATAGATCTA
                     AGCTACCGGTTTTTAT
                    CAGCTACCGGTTTTT
                   TCAGCTACCGGTTT
              CTCGCTCAGCTACC
           CCGCTCGCTCAGC
       GGGTCCGCTCGCTCA
   AGAAGGGTCCGCTC
GATAGAAGGGTCCGCT

GATAGAAGGGTCCGCTCGCTCAGCTACCGGTTTTTATAGATCTA
```

Introduction

Coverage: reads "covering" a position in the genome (average or at a single base or region)



TTTTTATAGATCTA

AGCTACCGGTTTTTAT

CAGCTACCGGTTTTT

TCAGCTACCGGTTT

CTCGCTCAGCTACC

CCGCTCGCTCAGC

GGGTCCGATCGCTCA

AGAAGGGTCCGCTC

GATAGAAGGGTCCGCT

131 bases in the reads

44 bases in the "genome"

GATAGAAGGGTCCGCTCGCTCAGCTACCGGTTTTTATAGATCTA

What is our average coverage?

What is the coverage at the arrow?

Introduction

```
                              TTTTTATAGATCTA
                     AGCTACCGGTTTTTAT
                    CAGCTACCGGTTTTT
                   TCAGGTACCGGTTT
              CTCGCTCAGCTACC
           CCGCTCGCTCAGC
       GGGTCCGATTGCTCA
   AGAAGGGTCCGCTC
GATAGAAGGGTCCGCT

GATAGAAGGGTCCGCTCGCTCAGCTACCGGTTTTTATAGATCTA
```

Why might there be differences among reads covering the same position?

Introduction

CCGCTCGCTCAGC

TCAGCTACCGGTTT

CTCGCTCAGCTACC

CAGCTACCGGTTTTT

AGAAGGGTCCGCTC

GATAGAAGGGTCCGCT

AGCTACCGGTTTTTAT

TTTTTATAGATCTA

GGGTCCGCTCGCTCA

How would you go about "assembling" these reads when you have no reference?

Code break

Write some code to find all the overlaps exactly 4 bp in length between CTCTAGGCC and a list of other sequences in the file ~/Topic_5/data/overlaps.fa

Overlap-layout-consensus

Overlap: make an overlap graph

Layout: find the path through the graph

Consensus: find the most likely contig sequence

OLC programs:
ARACHNE, PHRAP, CAP, TIGR, CELERA

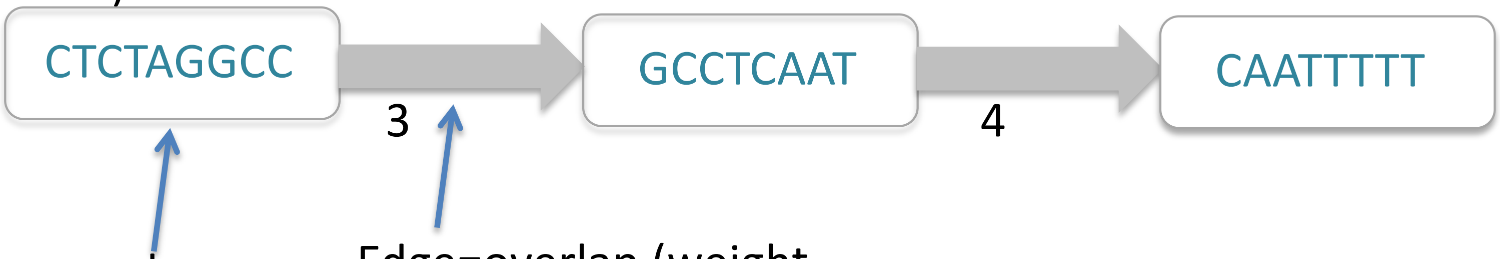
Overlap-layout-consensus

Reads: CTCTAGGCC GCCTCAAT CAATTTTT

This is a
graph!(directed)

```
CTCTAGGCC
      GCCTCAAT
```

```
GCCTCAAT
    CAATTTTT
```



CTCTAGGCC

3

GCCTCAAT

4

CAATTTTT

Node or vertex=
read

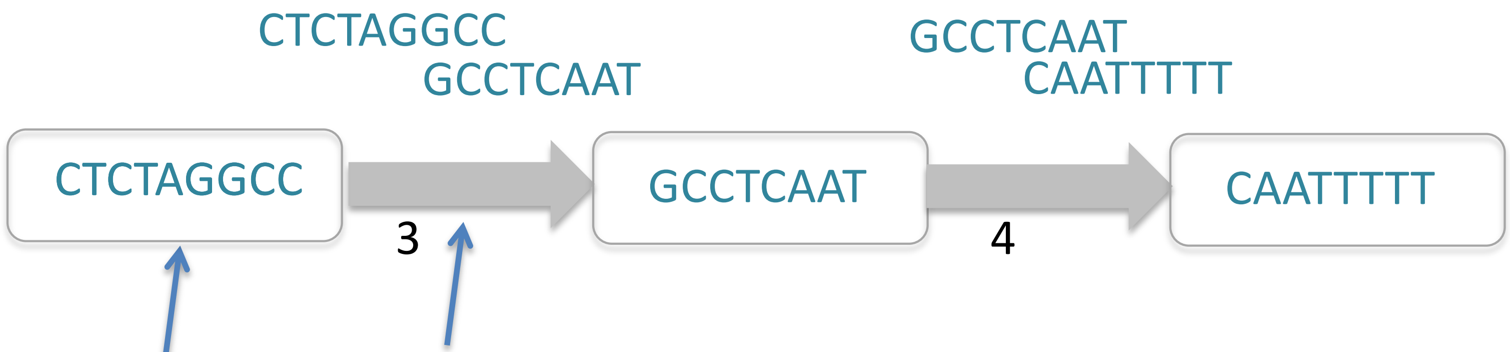
Edge=overlap (weight
by the amount of
overlap in bp)

Can pick a minimum overlap length (e.g. 3 bp)

Finding overlaps can be computationally challenging
when you have millions of reads!

Overlap-**layout**-consensus

Reads: CTCTAGGCC GCCTCAAT CAATTTTT



CTCTAGGCC
GCCTCAAT

GCCTCAAT
CAATTTTT

CTCTAGGCC → GCCTCAAT → CAATTTTT

3

4

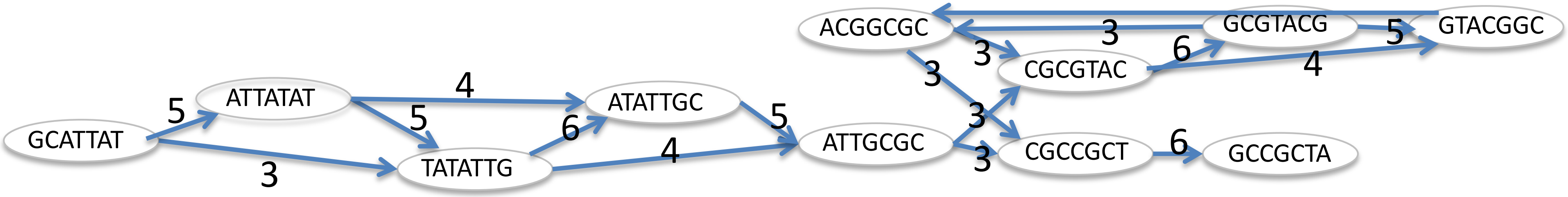
Node or vertex= read

Edge=overlap (weight by the amount of overlap in bp)

Here we have only one path through the graph

Overlap-**layout**-consensus

These graphs get complicated!



Minimum
overlap = 3
Read length = 7

GCATTATATATTGCGCGTACGGCGCCGCTACA

Original sequence

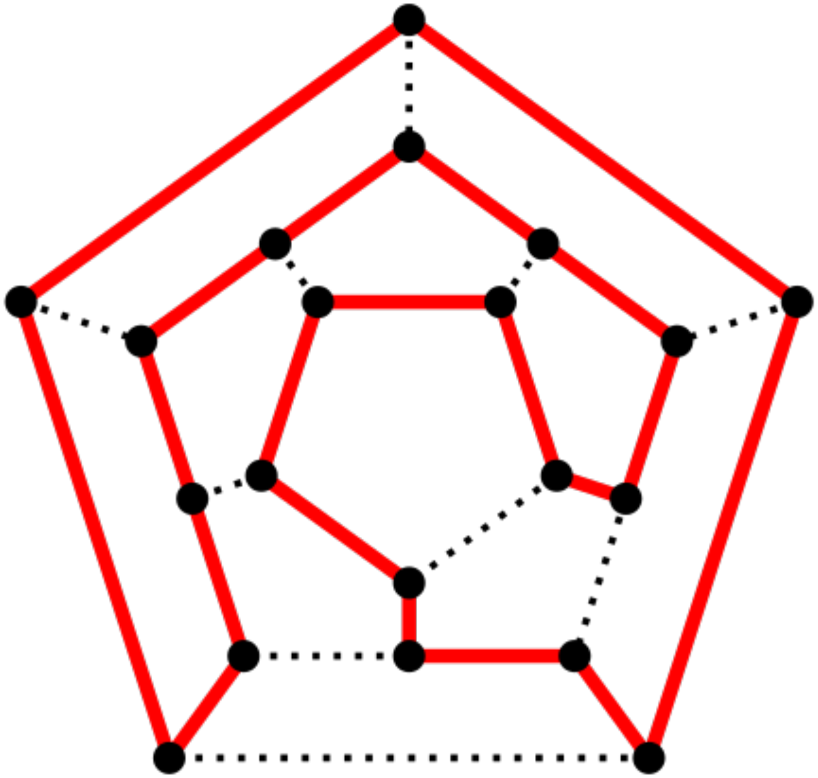
How can we find the best path?

Overlap-**layout**-consensus

Hamiltonian path: hit each node (read) once

–no quick way to figure it out (NP-complete)

–not practical and not implemented



Overlap-**layout**-consensus

Shortest superstring: find the shortest final sequence (greatest overlap between reads)

-hit each node (read) once

-NP-hard

Overlap-**layout**-consensus

Greedy algorithm (example)

1) Pairwise alignments between all fragments
2) Pick the two with the largest overlap
3) Merge chosen fragments
4) Repeat

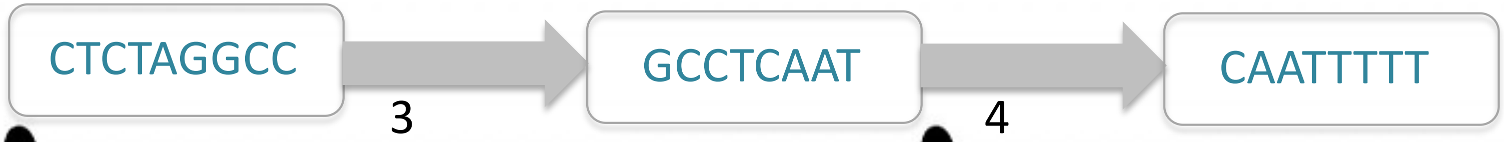
Overlap-layout-**consensus**

Join sequences together into one sequence

Reads: CTCTAGGCC GCCTCAAT CAATTTTT

```
CTCTAGGCC
      GCCTCAAT
```

```
GCCTCAAT
    CAATTTTT
```



CTCTAGGCC

3

GCCTCAAT

4

CAATTTTT

```
CTCTAGGCC
      GCCTCAAT
          CAATTTTT
```

CTCTAGGCCTCAATTTTT

Limitations of OLC

- require overlaps to be scored between all possible pairs of reads. This is a problem when you have millions of reads

- finding the best path through the graph with a huge number of nodes (reads) is computationally challenging

Is there a faster way to assemble many short reads?

De Bruijn graphs

What are all the 5-mers (5 bp fragments) in these reads?

	read 1	read 2
2 reads of 9 bp	ATGGGGAAC	GGGAACCCC
	ATGGG	GGGAA
	TGGGG	GGAAC
	GGGGA	GAACC
	GGGAA	AACCC
	GGAAC	ACCCC

If a read is L bp long, how many kmers of size k can you make?

Code break

Find all the unique 9mers in a fasta sequence and sort them alphabetically ~/Topic_5/data/kmer.fa

1. Find all the kmers in this fasta sequence.

    Hints: test out the following commands.

    The cut program lets you pull out columns of text

    ```
    cut -c2- kmer.fa
    cut -c1-4 kmer.fa

    for num in {1..10}
    do
    echo $num >> file.txt
    done
    ```

2. Sort them and keep the unique ones

    Hint: try sort

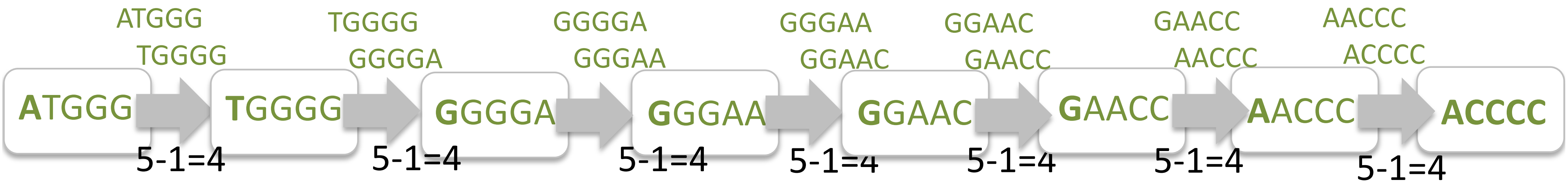
De Bruijn graphs

- Join up all the k-mers (length = k bp) into a graph with an overlap of k-1 (here k=5)

read 1
ATGGGGAAC

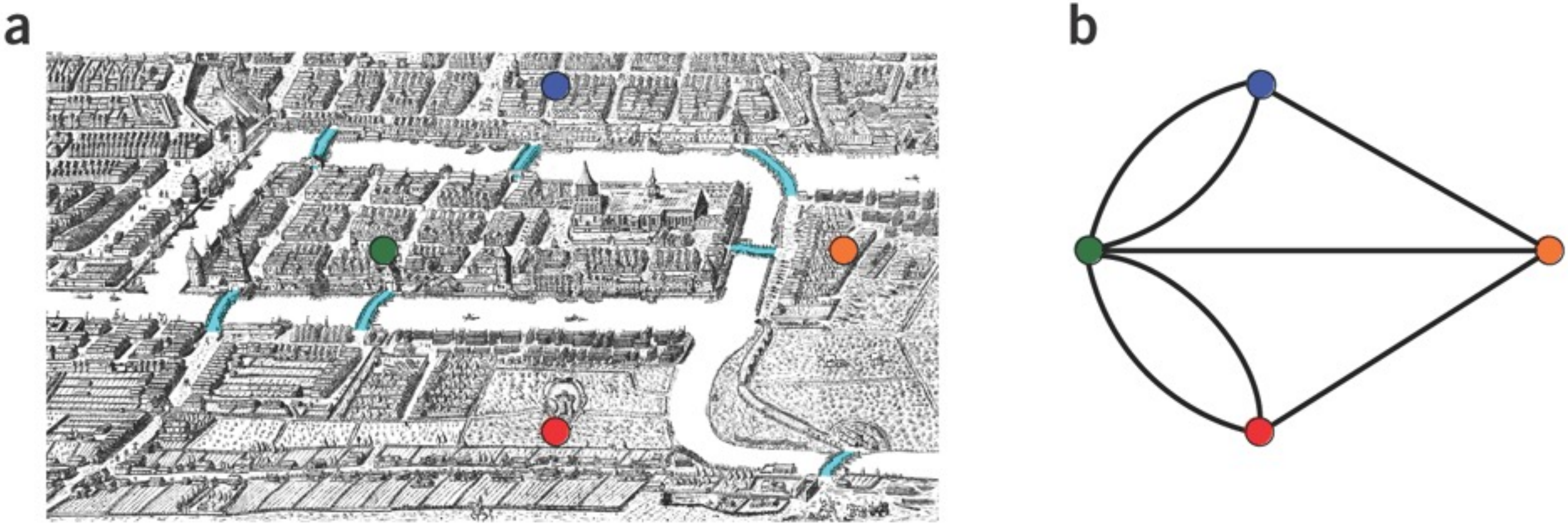
read 2
GGGAACCCC

ATGGG
TGGGG
GGGGA
GGGAA
GGAAC
GAACC
AACCC
ACCCC



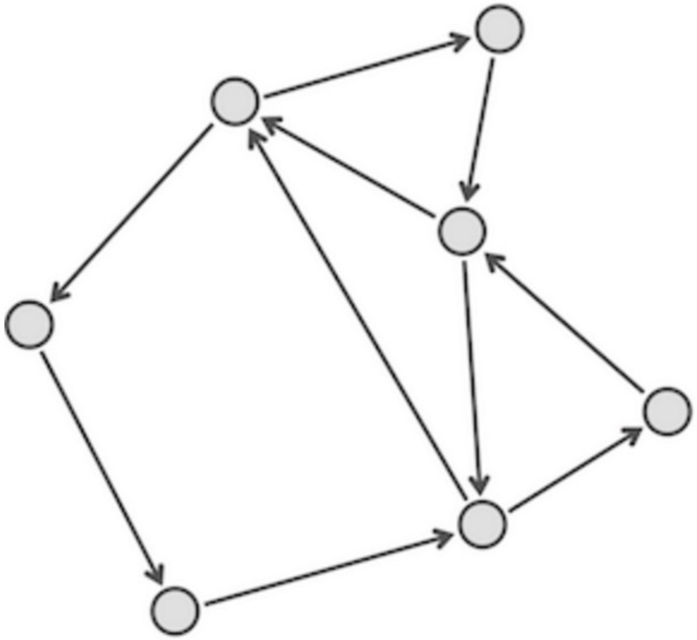
- Traverse through the graph
- The first base of each node spells out the sequence

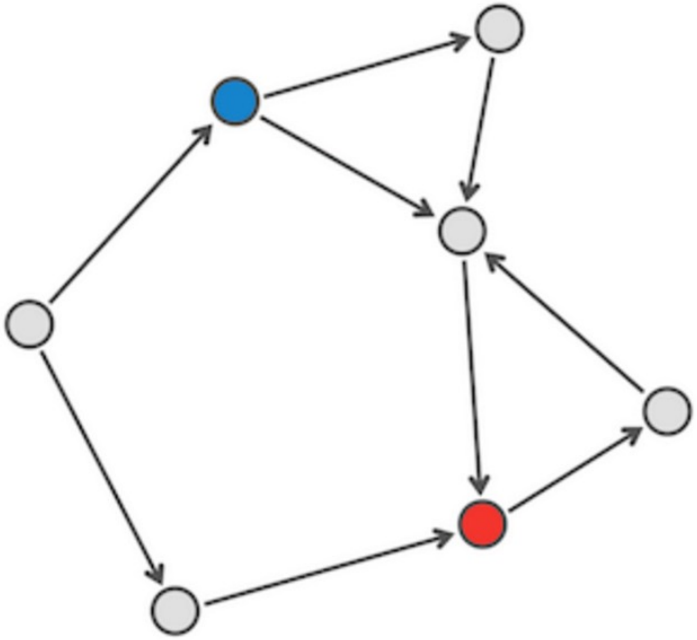
De Bruijn graphs



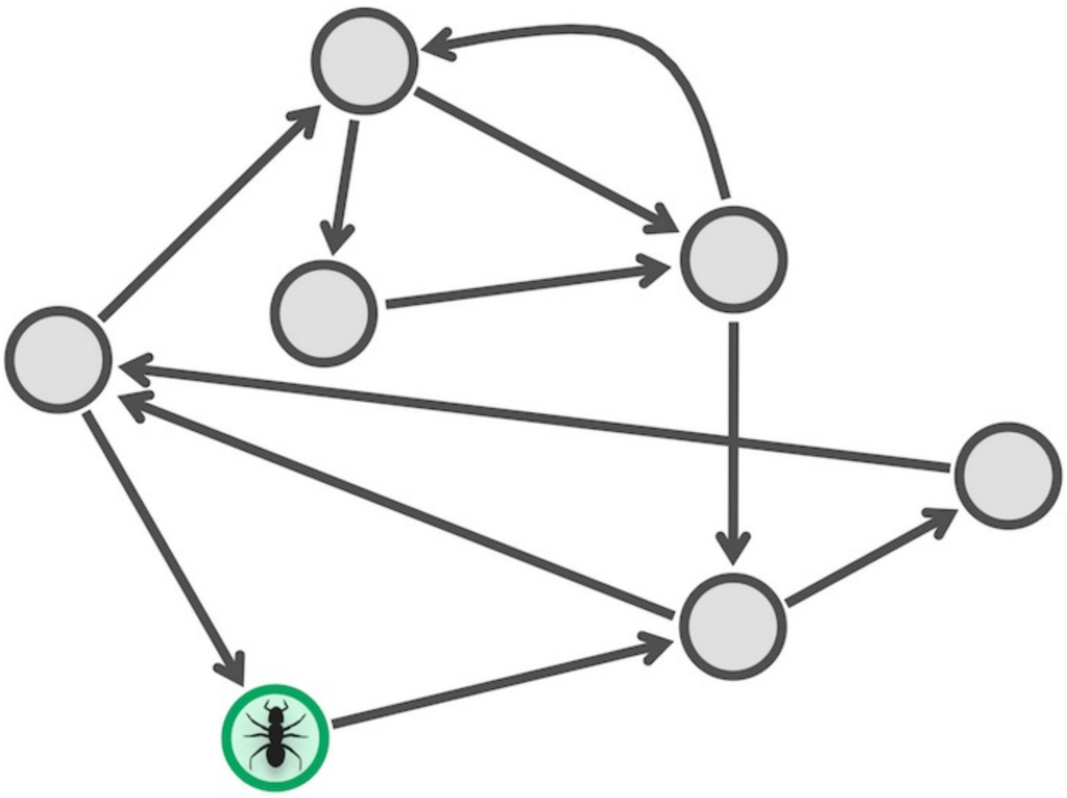
De Bruijn graphs

Eulerian graph must be both balanced and strongly connected



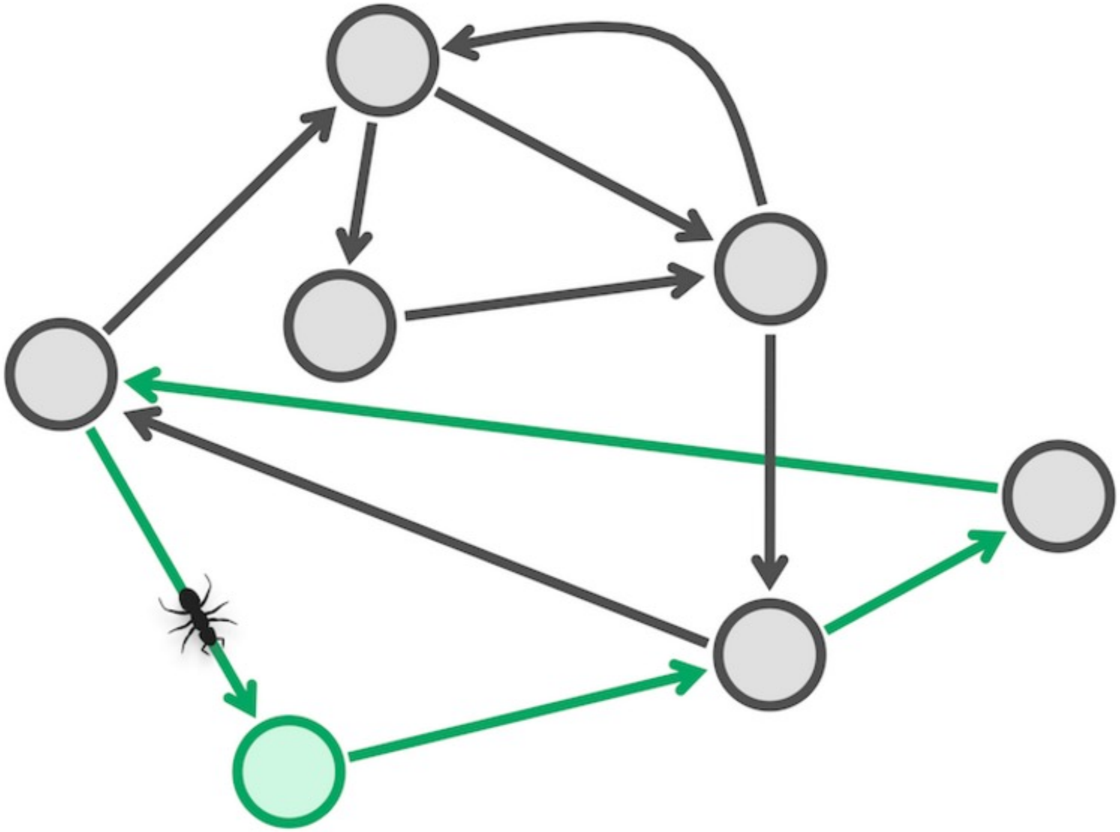


De Bruijn graphs



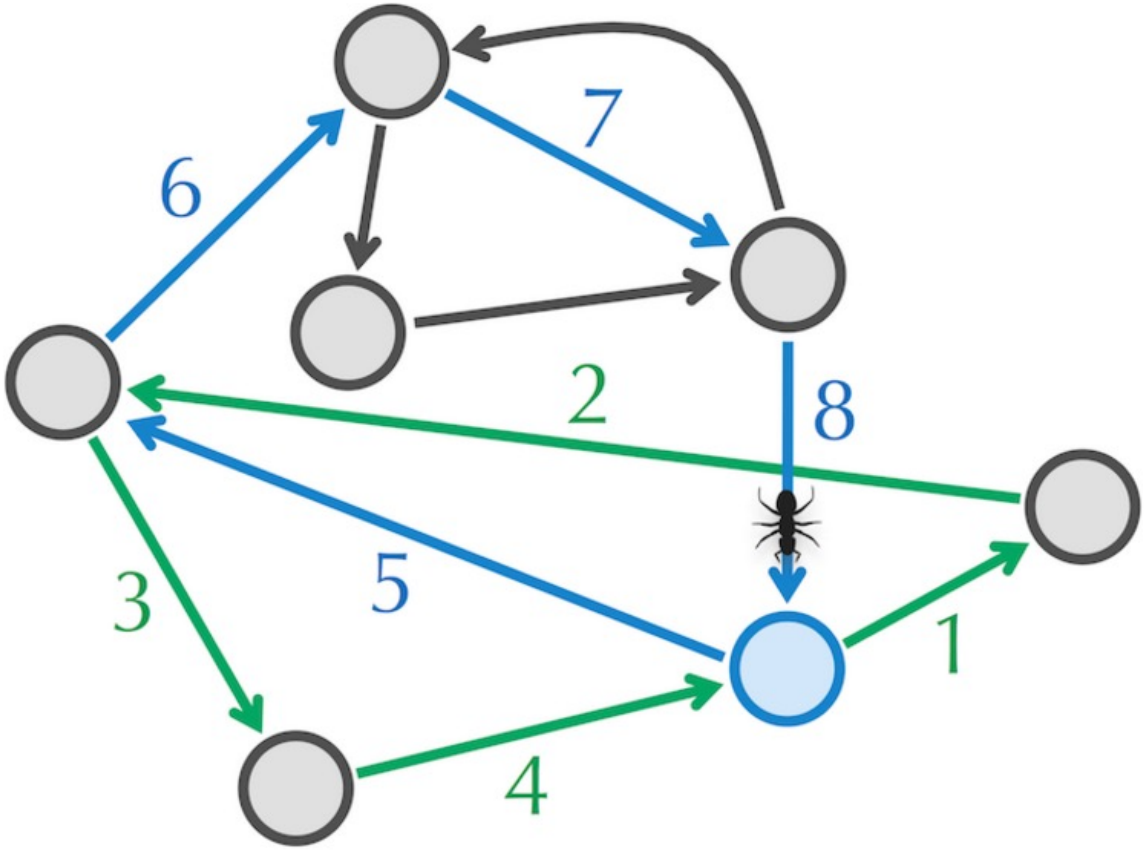
Algorithm to find a path through an Eulerian graph

De Bruijn graphs



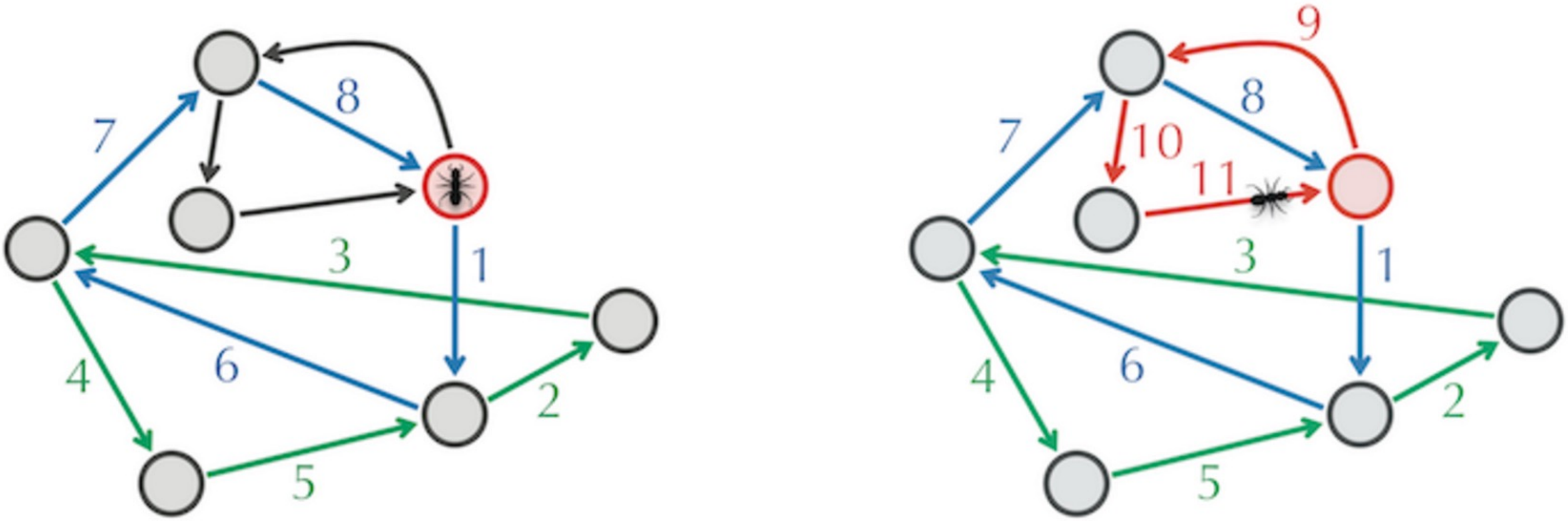
Algorithm to find a path through an Eulerian graph

De Bruijn graphs



Algorithm to find a path through an Eulerian graph

De Bruijn graphs



Algorithm to find a path through an Eulerian graph

De Bruijn graphs

Limitations of the Eulerian path:

- With “perfect” genomic data there are usually many Eulerian tours

- Data is not perfect (areas of low coverage, errors, repeats, etc.)

De Bruijn graphs

TAGTCGAGGCTTTAGATCCGATGAGGCTTTAGAGACAG

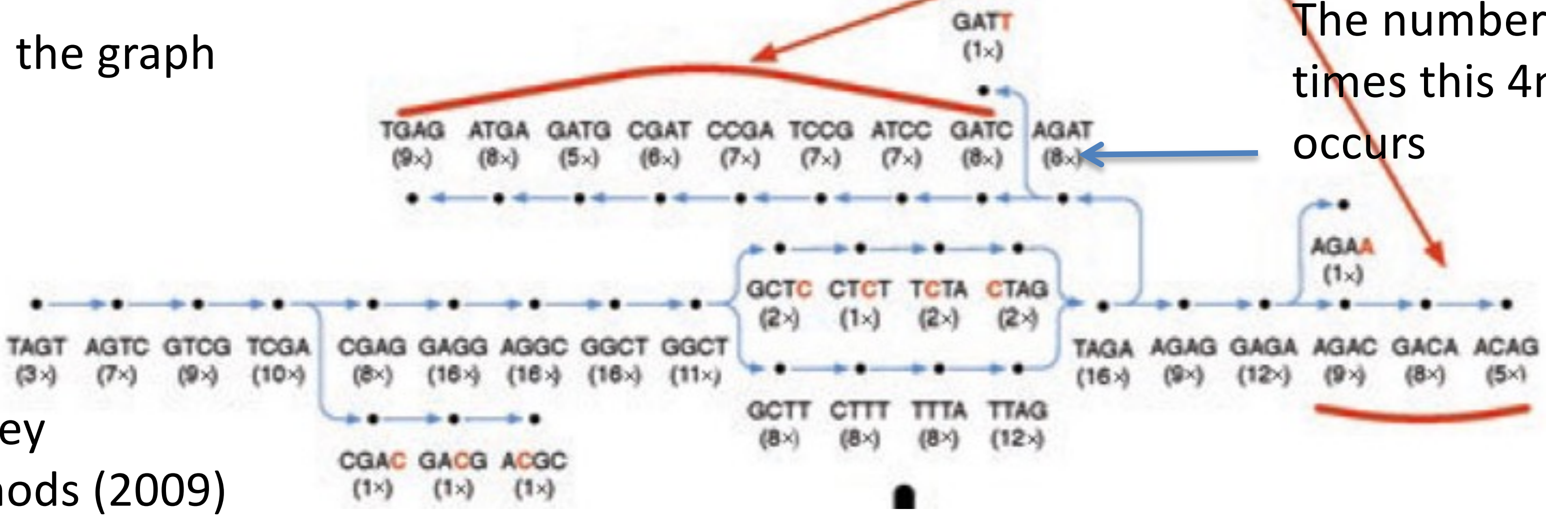
1. Sequence

```
AGTCGAG   CTTTAGA   CGATGAG   CTTTAGA
GTCGGG    TTAGATC   ATGAGGC   GAGACAG
GAGGCTC   ATCCGAT   AGGCTTT   GAGACAG
AGTCGAG   TAGATCC   ATGAGGC   TAGAGAA
TAGTCGA   CTTTAGA   CCGATGA   TTAGAGA
CGAGGCT   AGATCCG   TGAGGCT   AGAGACA
TAGTCGA   GCTTTAG   TCCGATG   GCTCTAG
TCGACGC   GATCCGA   GAGGCTT   AGAGACA
TAGTCGA   TTAGATC   GATGAGG   TTTAGAG
GTCGAGG   TCTAGAT   ATGAGGC   TAGAGAC
AGGCTTT   ATCCGAT   AGGCTTT   GAGACAG
AGTCGAG   TTAGATT   ATGAGGC   AGAGACA
GGCTTTA   TCCGATG   TTTAGAG
CGAGGCT   TAGATCC   TGAGGCT   GAGACAG
AGTCGAG   TTTAGATC  ATGAGGC   TTAGAGA
GAGGCTT   GATCCGA   GAGGCTT   GAGACAG
```

Sequencing errors

2. Find the kmers

3. Build the graph

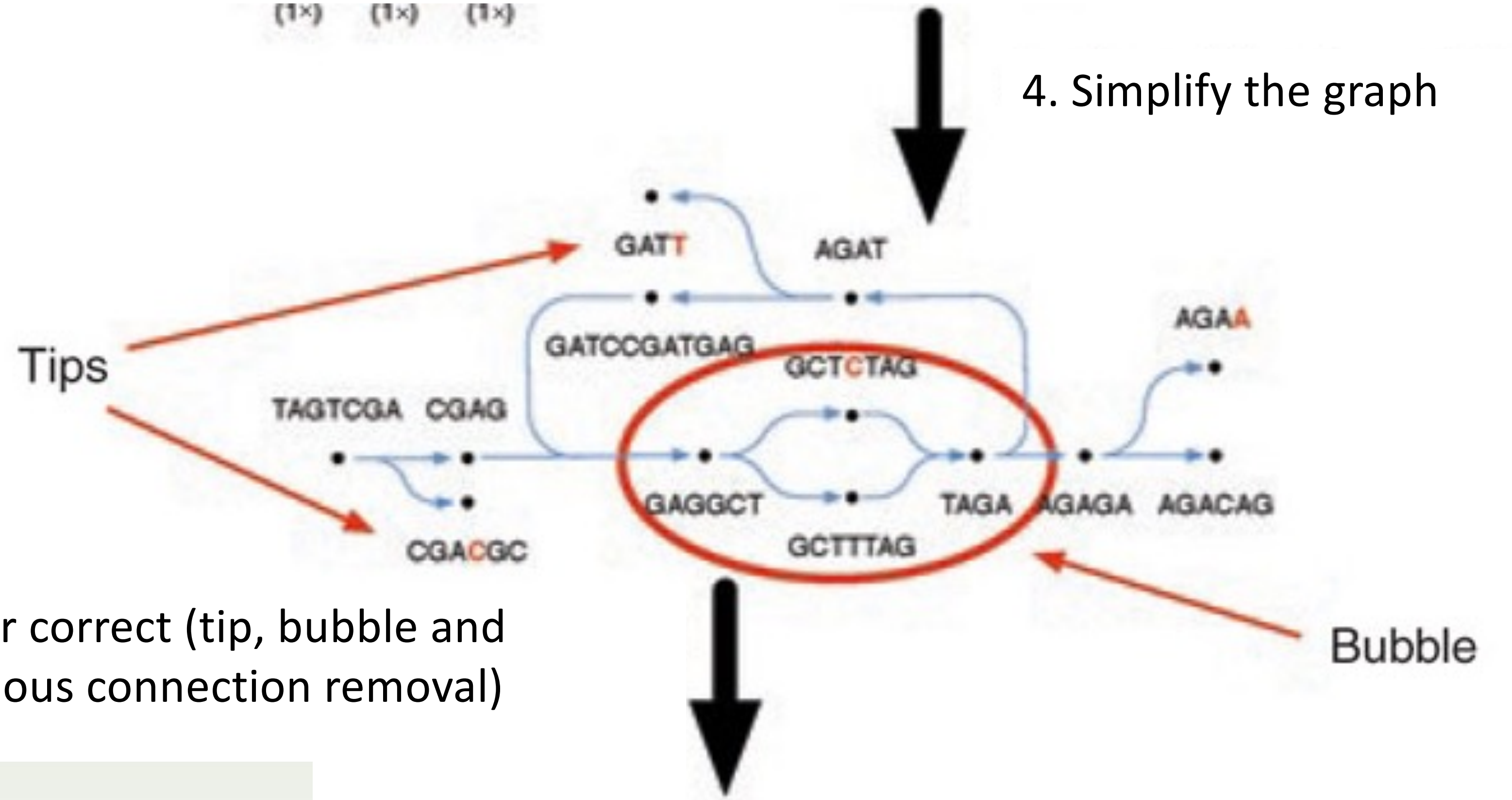


Linear stretches

The number of times this 4mer occurs

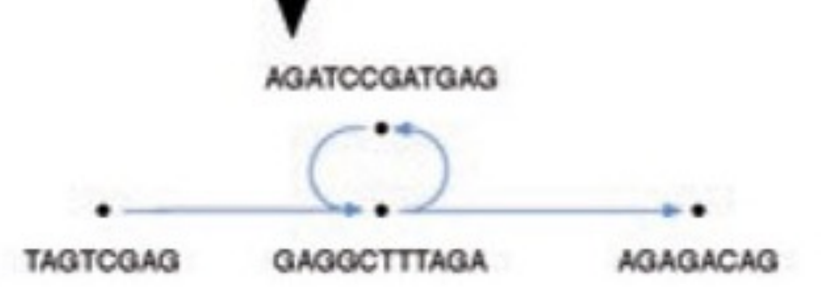
Flicek & Birney
Nature Methods (2009)

De Bruijn graphs



4. Simplify the graph

5. Error correct (tip, bubble and erroneous connection removal)



Flicek & Birney Nature Methods (2009)

De Bruijn graphs

Advantages:

1) Set node length (no overlap algorithm)
2) Easy approaches for traversing through the graph
3) Simpler representation of repeats in the graph

Disadvantages:

1) Lose information
2) Shorter contigs

For PacBio and other long read sequences, what type of assembly strategy would you use?

Repetitive regions

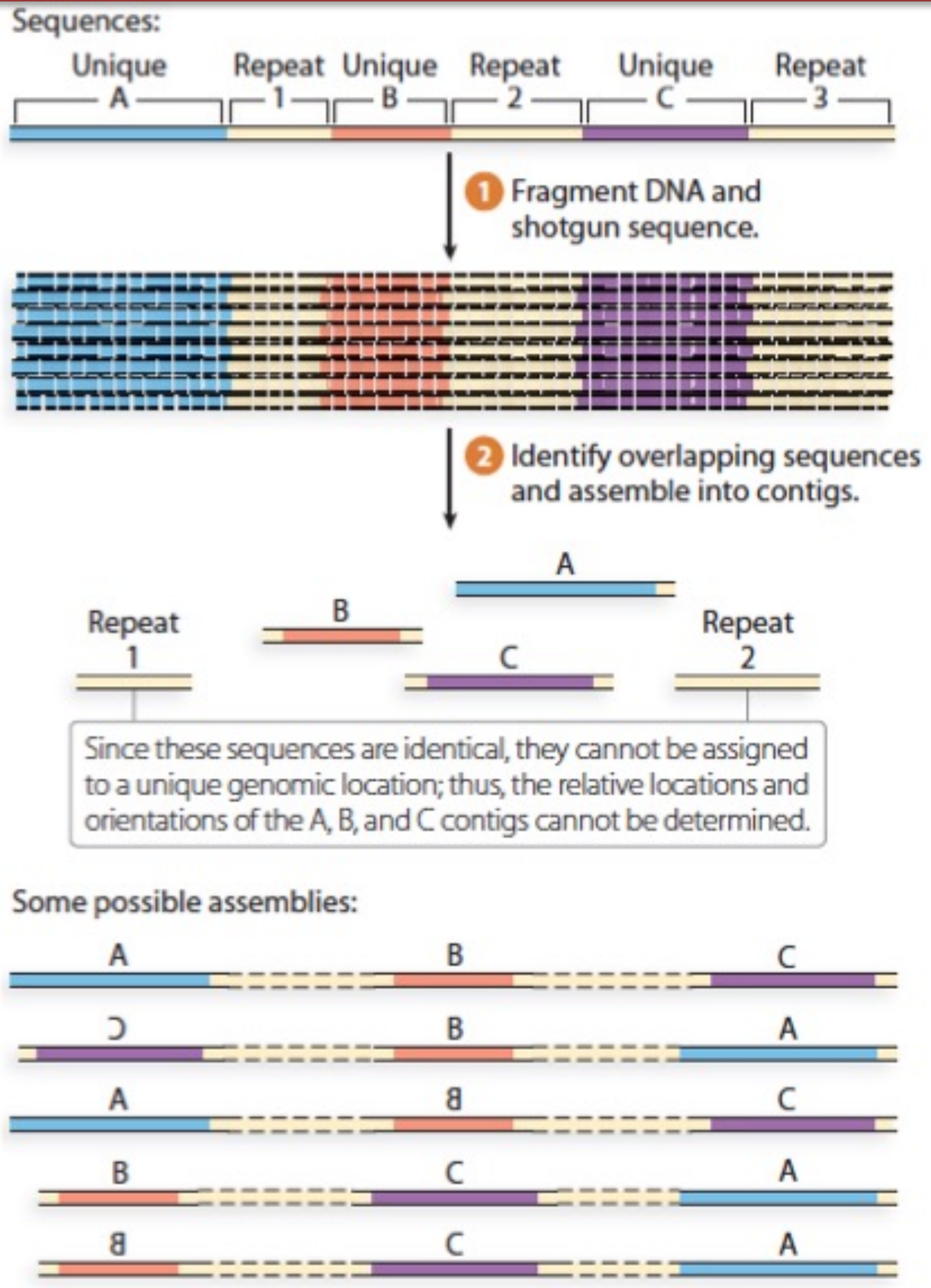
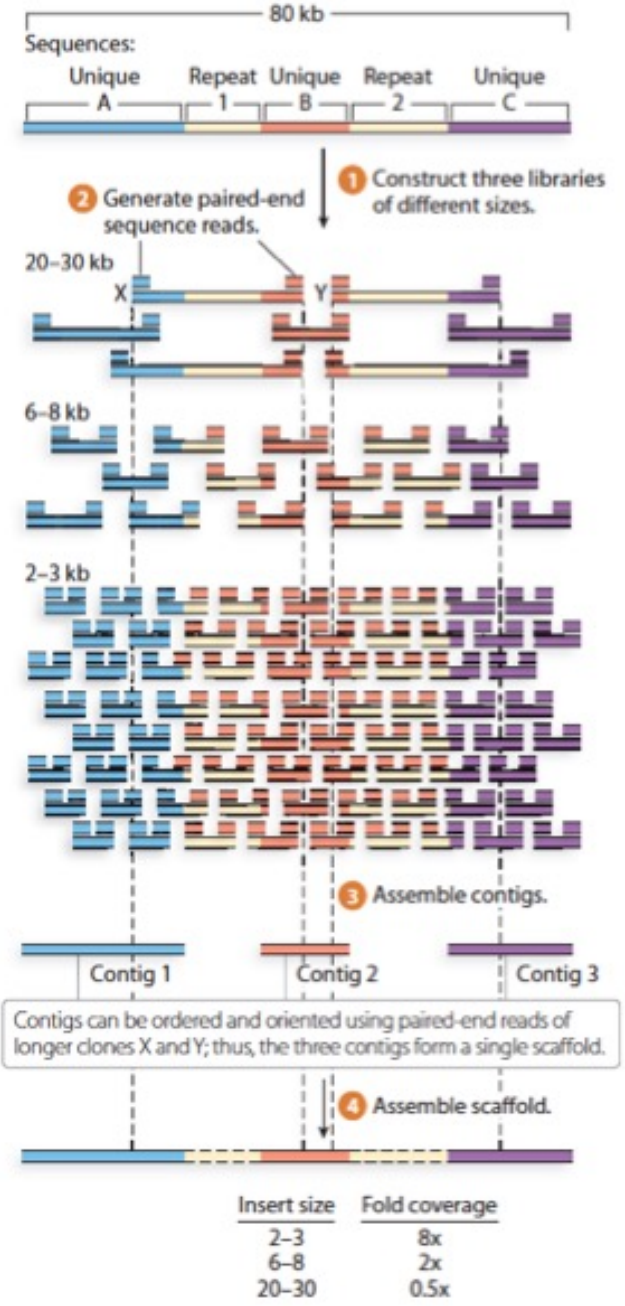

Sequences:
Unique A
Repeat 1
Unique B
Repeat 2
Unique C
Repeat 3
1 Fragment DNA and shotgun sequence.
2 Identify overlapping sequences and assemble into contigs.
A
B
C
Repeat 1
Repeat 2
Since these sequences are identical, they cannot be assigned to a unique genomic location; thus, the relative locations and orientations of the A, B, and C contigs cannot be determined.
Some possible assemblies:
A
B
C
Ɔ
B
A
A
ꓭ
C
B
C
A
ꓭ
C
A

Figure 18.2 The problem of repetitive DNA.

Sanders and Bowman

Repetitive regions



Contigs can be ordered and oriented using paired-end reads of longer clones X and Y; thus, the three contigs form a single scaffold.

Insert size	Fold coverage
2–3	8x
6–8	2x
20–30	0.5x

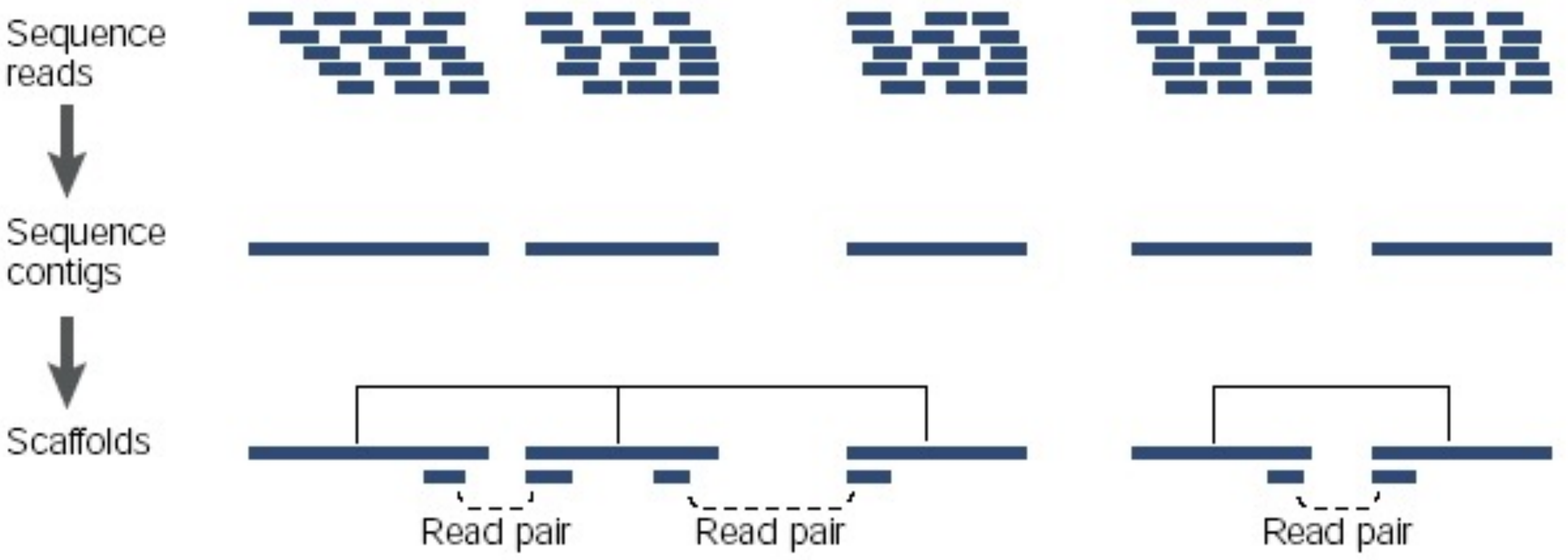
Figure 18.3 Paired-end shotgun sequencing strategy.

Sanders and Bowman

Repetitive regions

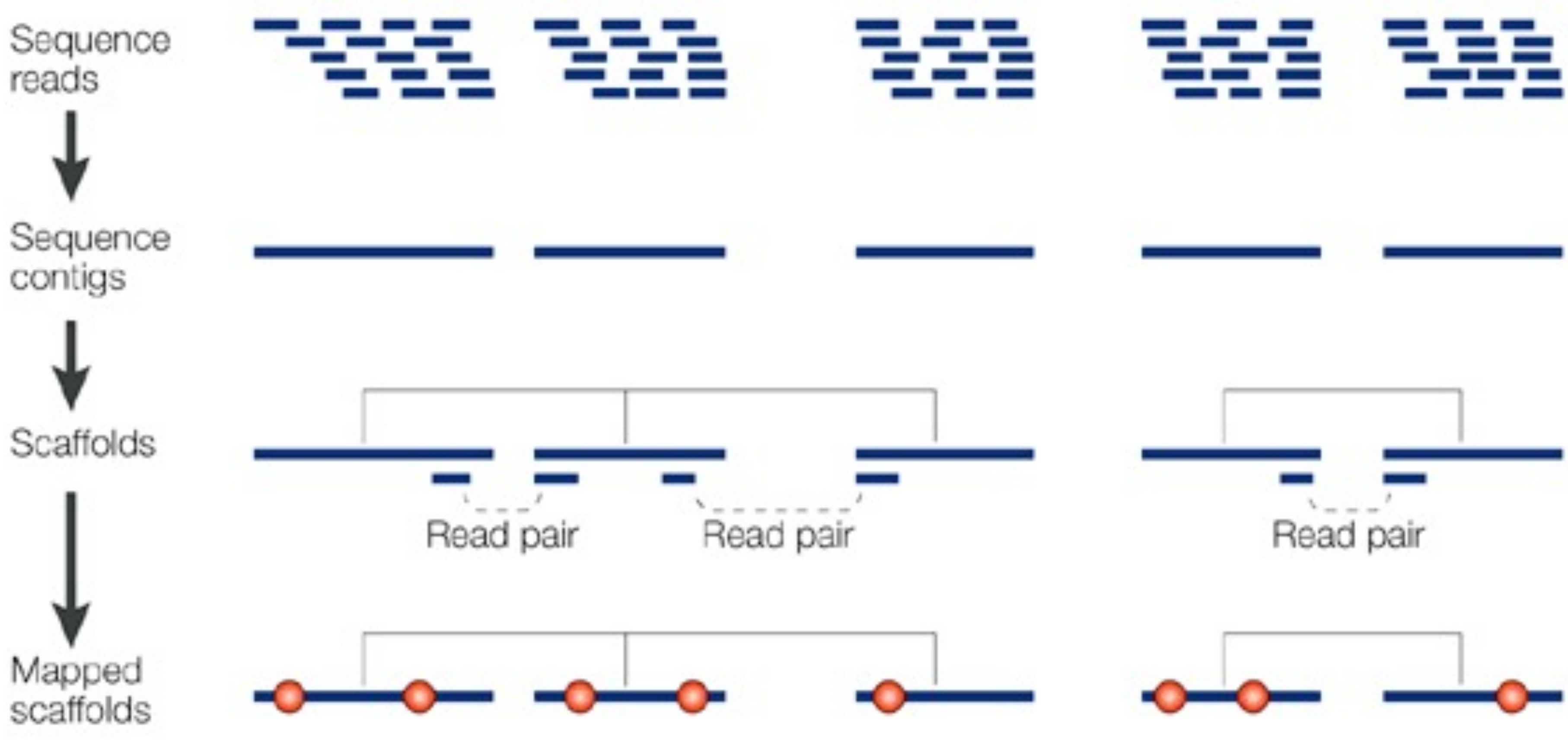
- Finishing eukaryotic genome assemblies can be challenging because much of the genome is repetitive
- This repetitive DNA breaks up the assembly and obscures the order and orientation of the assembled contigs
- Even well studied model organisms can have poorly assembled regions of their genome

Repetitive regions



Adams 2008

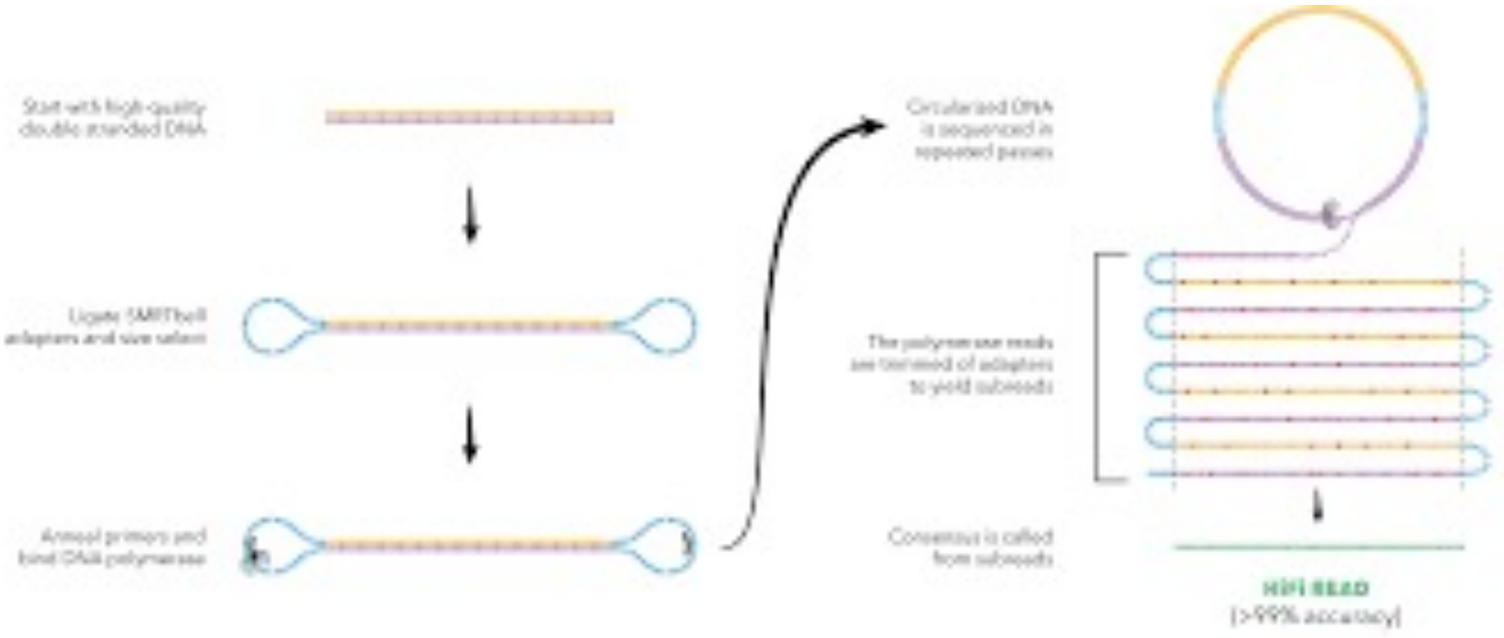
Repetitive regions



Adams 2008

Current assembly approaches

- Long read sequencing
- Synthetic long reads
- Long-range scaffolding technologies


Start with high-quality double stranded DNA
Ligate SMRTbell adapters and size select
Anneal primers and bind DNA polymerase
Circularized DNA is sequenced in repeated passes
The polymerase reads are trimmed of adapters to yield subreads
Consensus is called from subreads
HiFi READ
(>99% accuracy)

Long read assemblies

Long read only *de novo* assembly. PacBio/Nanopore reads are assembled using an OLC algorithm (e.g., HGAP).(>50x PacBio, or 15X PacBio HiFi)

Hybrid *de novo* assembly. Error correct long reads with more accurate short reads (e.g., PacBioToCA module of Celera) before performing long read assembly. (~20x PacBio)

Gap filling. Starting with an *existing* mate-pair based assembly, the internal gaps (consisting of Ns) inside the scaffolds are filled using PacBio sequences. (~5x PacBio)

Scaffolding. Using an *existing* assembly (such as an assembly based on short read data), PacBio reads are used to join contigs. (~5x PacBio)

Synthetic long read assemblies

Synthetic long reads (SLRs) technologies [Illumina, 10X Genomics, Loop Genomics, and Universal Sequencing Technology (UST)]

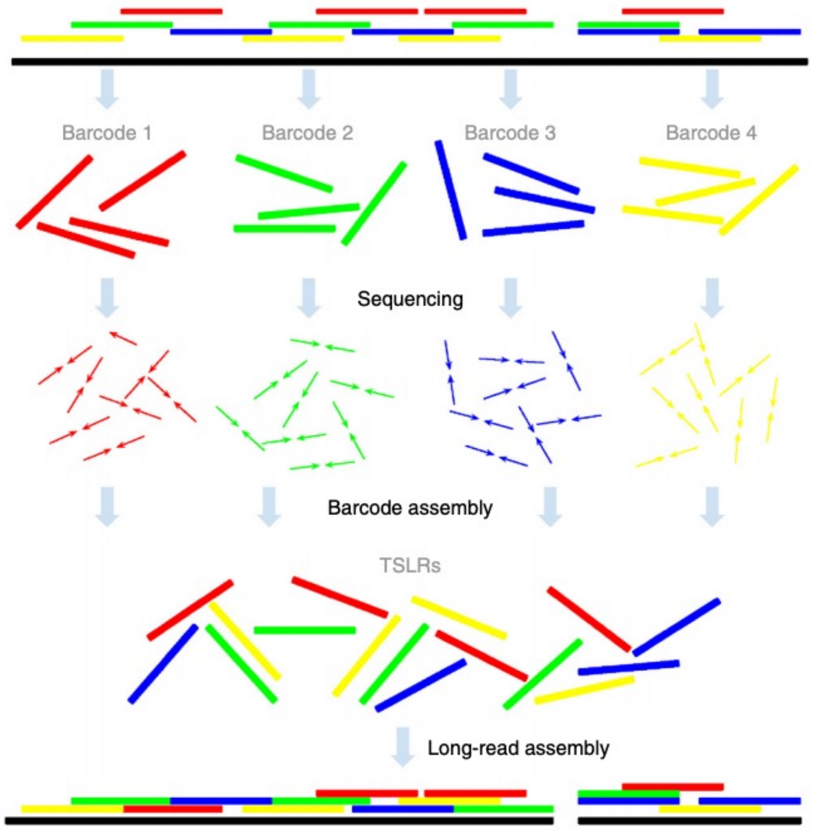


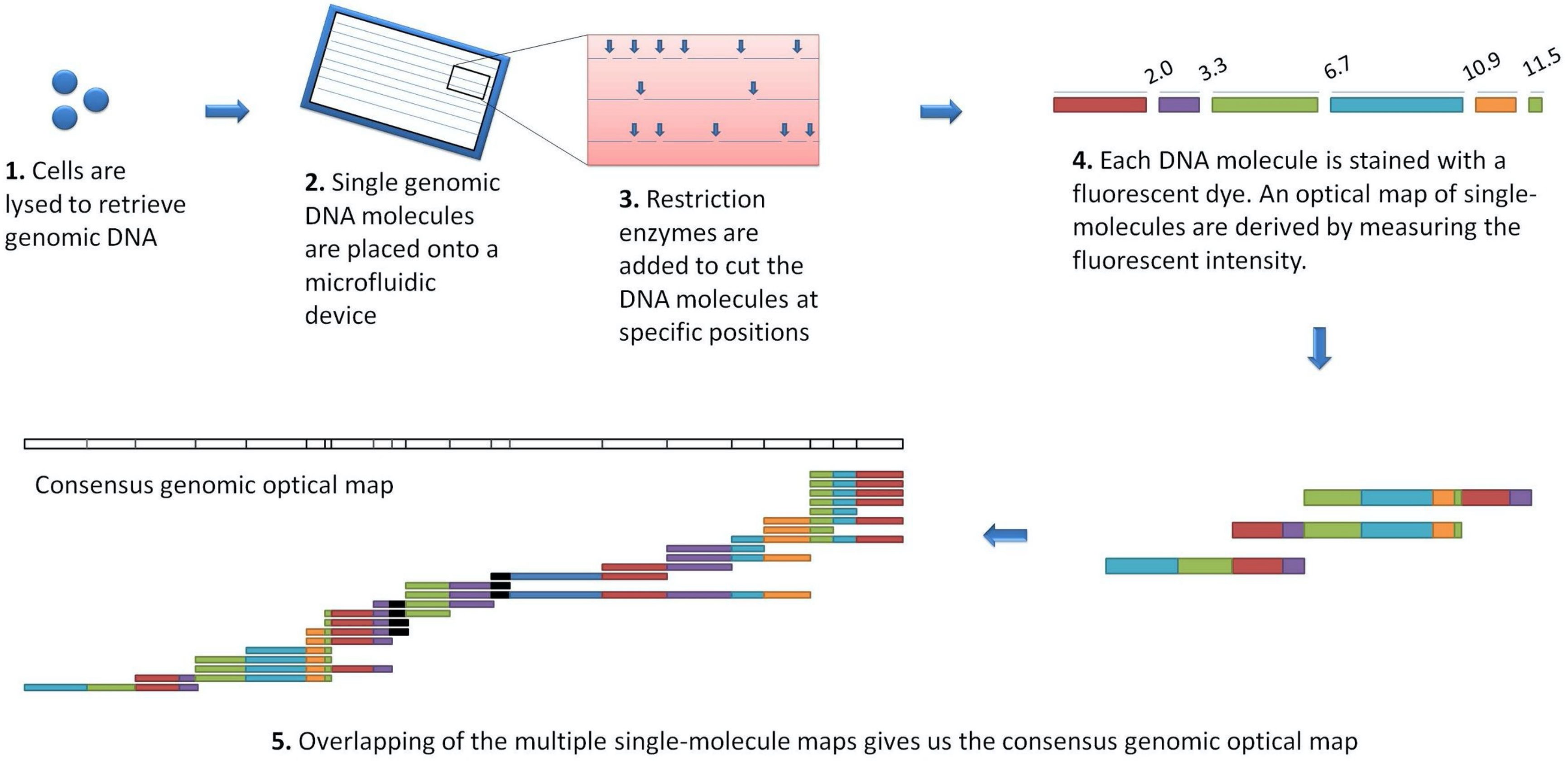
Figure 1 | The TSLR technology. The barcode assembly step generates virtual long reads. In an idealized scenario, the barcode assembly would result in ~300 TSLRs with lengths of ~10 kb. In reality, it results in 350–450 TSLRs varying in length from 1 to 10 kb.

Anton Bankevich, & Pavel A Pevzner. (2016). TruSPAdes: Barcode assembly of TruSeq synthetic long reads. Nature Methods, 13(3), 248-250.

Long-range scaffolding technologies

Optical mapping

- Bionano Genomics https://vimeo.com/116090215



1. Cells are lysed to retrieve genomic DNA

2. Single genomic DNA molecules are placed onto a microfluidic device

3. Restriction enzymes are added to cut the DNA molecules at specific positions

4. Each DNA molecule is stained with a fluorescent dye. An optical map of single-molecules are derived by measuring the fluorescent intensity.

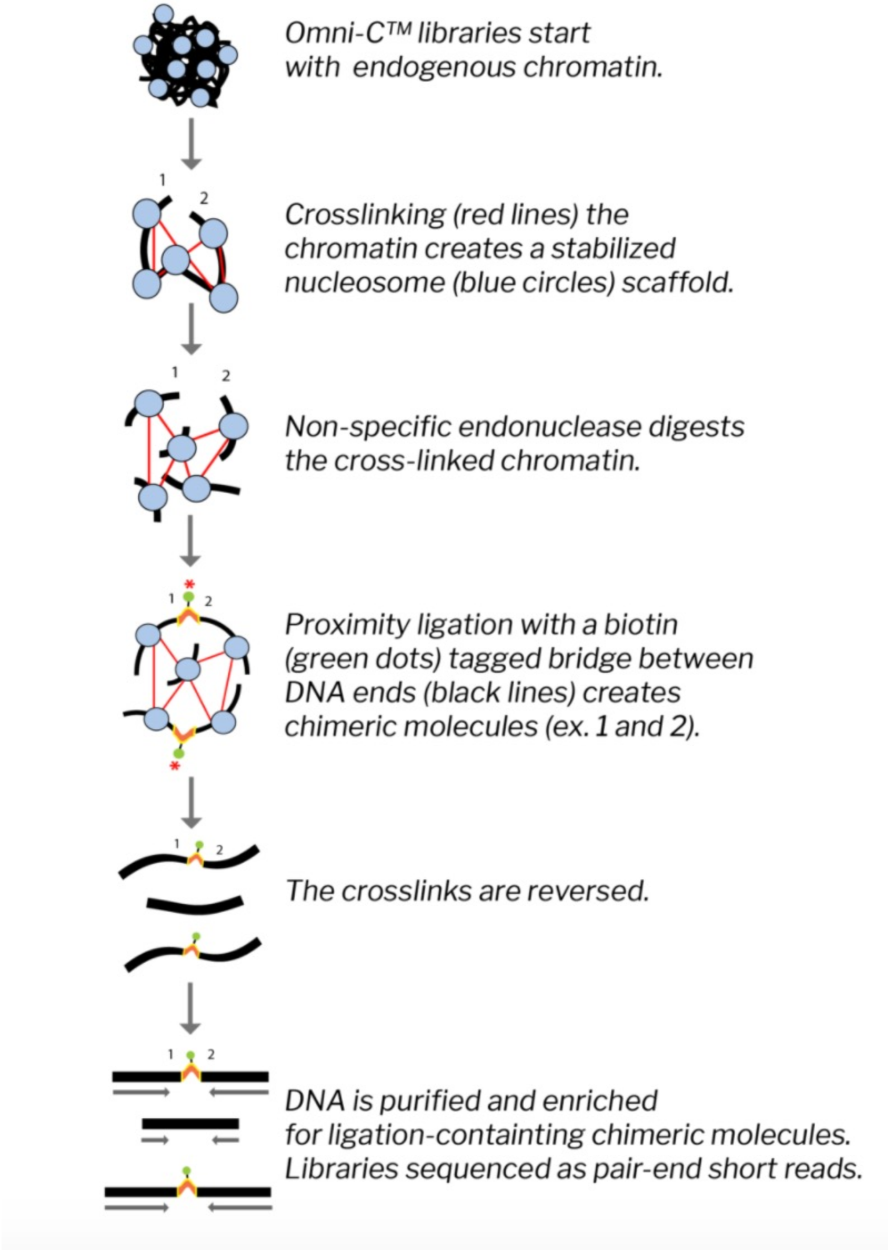
5. Overlapping of the multiple single-molecule maps gives us the consensus genomic optical map

Long-range scaffolding technologies

Omni-C (similar to Hi-C) is one approach to get chromosome level assemblies

Paired reads with small and big distances between them are created. There are more paired reads close together than far apart.

These reads are aligned to the contigs
Contigs with more joins (paired reads aligned across two contigs and therefore "joined") are closer together in the genome than those with few joins



Omni-C™ libraries start with endogenous chromatin.

Crosslinking (red lines) the chromatin creates a stabilized nucleosome (blue circles) scaffold.

Non-specific endonuclease digests the cross-linked chromatin.

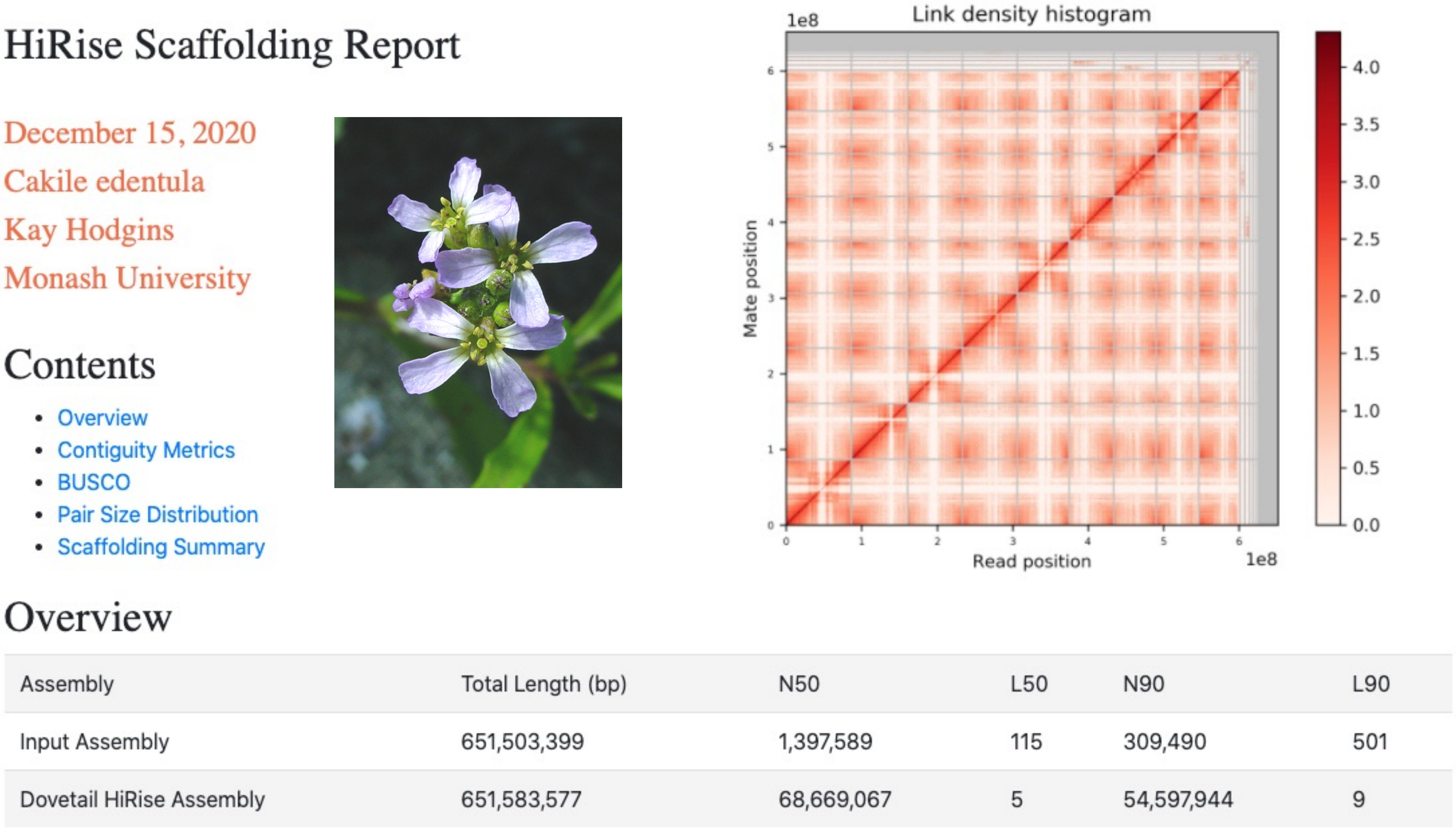
Proximity ligation with a biotin (green dots) tagged bridge between DNA ends (black lines) creates chimeric molecules (ex. 1 and 2).

The crosslinks are reversed.

DNA is purified and enriched for ligation-containting chimeric molecules. Libraries sequenced as pair-end short reads.

https://www.youtube.com/watch?v=-MxEw3IXUWU

Long-range scaffolding technologies



HiRise Scaffolding Report

December 15, 2020

Cakile edentula

Kay Hodgins

Monash University

Contents

- Overview
- Contiguity Metrics
- BUSCO
- Pair Size Distribution
- Scaffolding Summary

Overview

Assembly	Total Length (bp)	N50	L50	N90	L90
Input Assembly	651,503,399	1,397,589	115	309,490	501
Dovetail HiRise Assembly	651,583,577	68,669,067	5	54,597,944	9

Other types of de novo assembly

Transcriptome

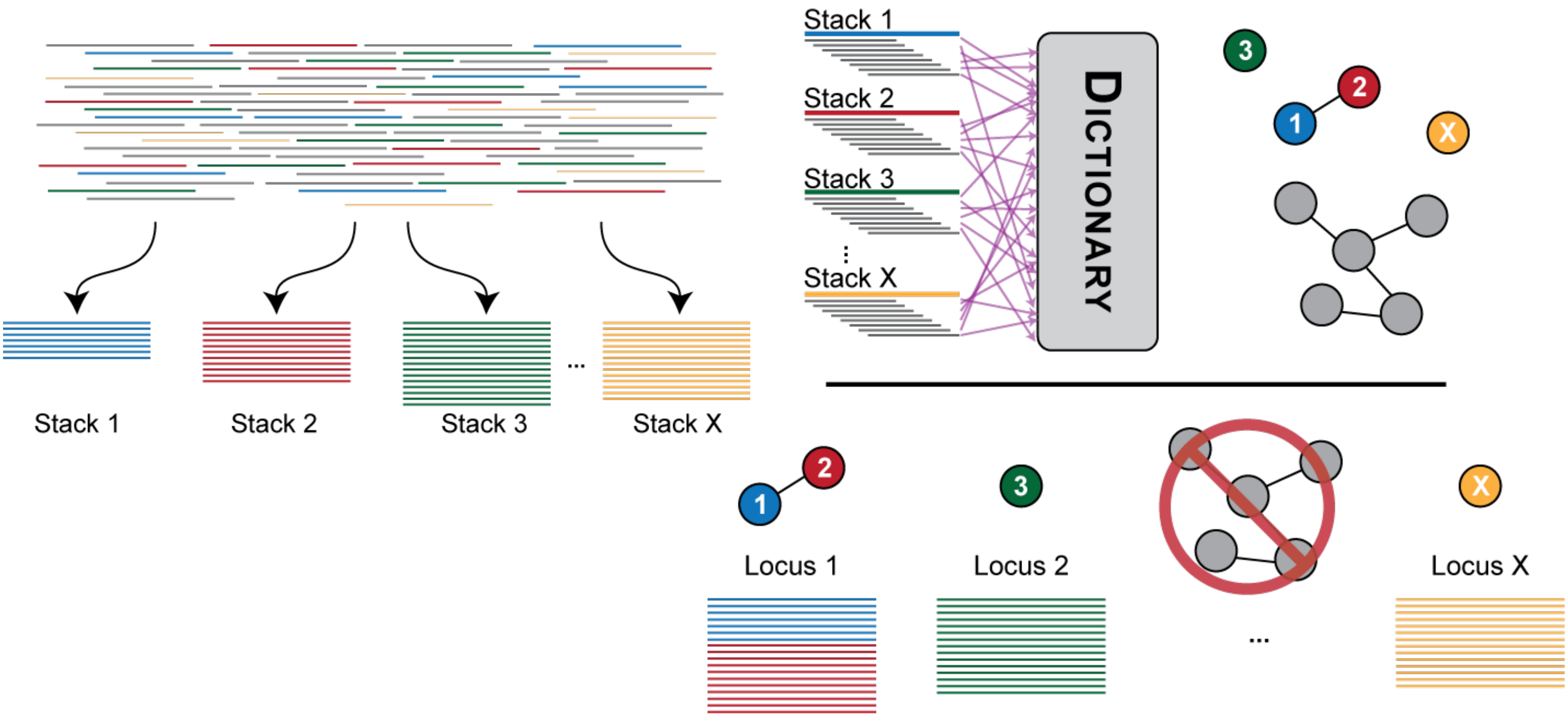
- Variable coverage among genes/isoforms
- Alternative splicing
  promotors, exons, and poly(A)

PacBio/Nanopore long reads – full length isoforms

Illumina short reads – Trinity (recommended)

Other types of de novo assembly

De novo assembly of GBS reads: Stacks



http://catchenlab.life.illinois.edu/stacks/param_tut.php

Further Reading

Jiao WB, Schneeberger K. 2017. The impact of third generation genomic technologies on plant genome assembly. Current Opinion in Plant Biology 36: 64–70.

Flicek, P., & Birney, E. (2009). Sense from sequence reads: methods for alignment and assembly. Nature methods, 6, S6-S12.

Zerbino DR, Birney E. Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. Genome Research. 2008;18(5):821-829. doi:10.1101/gr.074492.107.

http://computing.bio.cam.ac.uk/local/doc/velvet.pdf

Li, Z., Chen, Y., Mu, D., Yuan, J., Shi, Y., Zhang, H., ... & Yang, B. (2012). Comparison of the two major classes of assembly algorithms: overlap–layout–consensus and de-bruijn-graph. Briefings in functional genomics, 11(1), 25-37.

Grabherr MG, Haas BJ, Yassour M, et al. Trinity: reconstructing a full-length transcriptome without a genome from RNA-Seq data. Nature biotechnology. 2011;29(7):644-652. doi:10.1038/nbt.1883.

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

J. Catchen, A. Amores, P. Hohenlohe, W. Cresko, and J. Postlethwait. Stacks: building and genotyping loci de novo from short-read sequences. G3: Genes, Genomes, Genetics, 1:171-182, 2011.

http://catchenlab.life.illinois.edu/stacks/

Tutorial: short read assembly

Today you will assemble a bacterial genome using short reads and long reads and then compare the assemblies

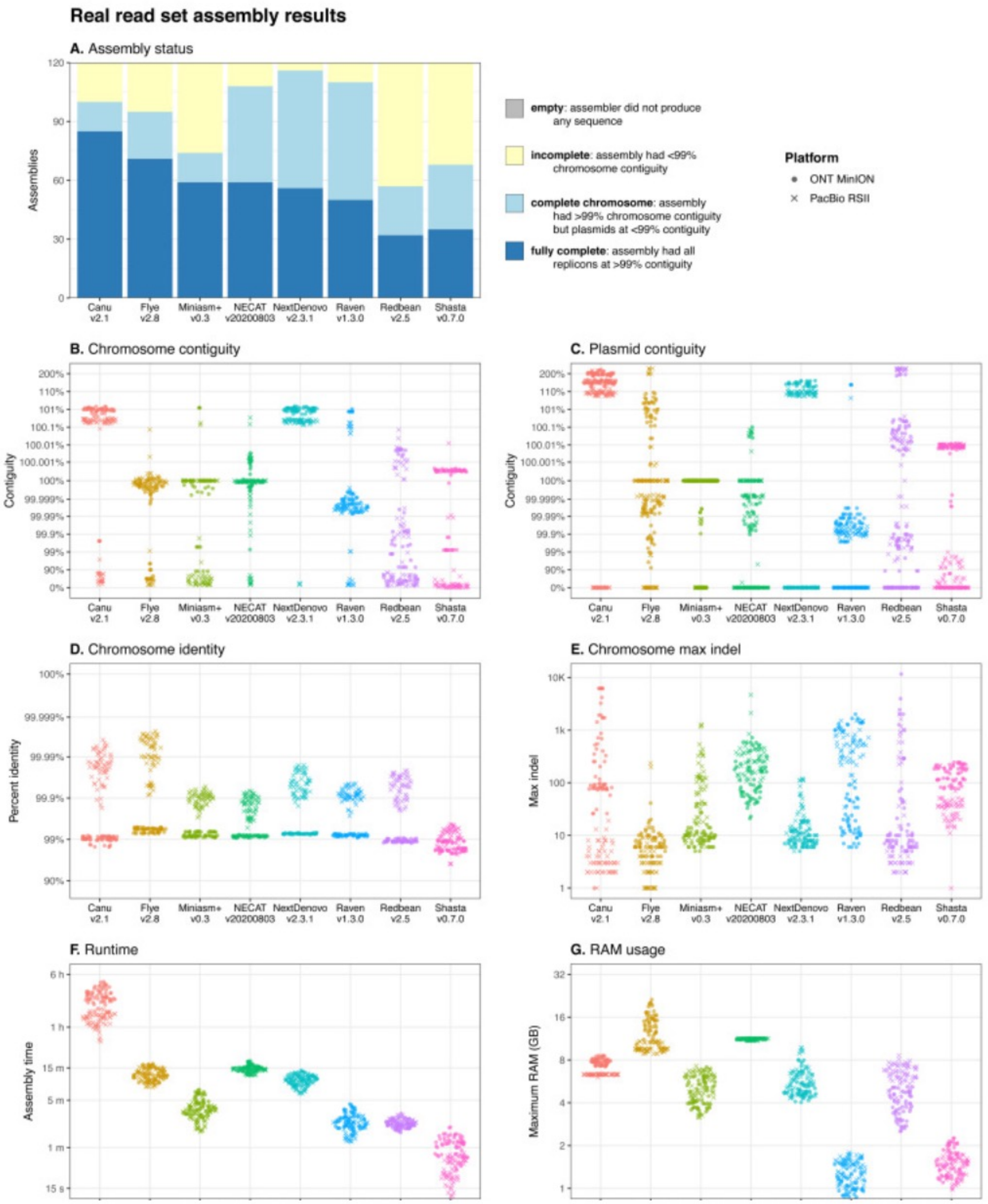
Velvet overview (short reads):

1. Hash k-mers
2. Construct the graph
3. Correct for errors
4. Resolve the repeats

Minimap2/Miniasm/Racon (long reads):

1. Minimap2 to map the reads to each other (overlap)
2. Miniasm to assemble (layout)
3. Minimap2/Racon (consensus)

Long read assemblers benchmarking



Tutorial: assembly metrics

Finally, we need to assess the quality of the assemblies

There are a number of metrics. Below are a few examples:

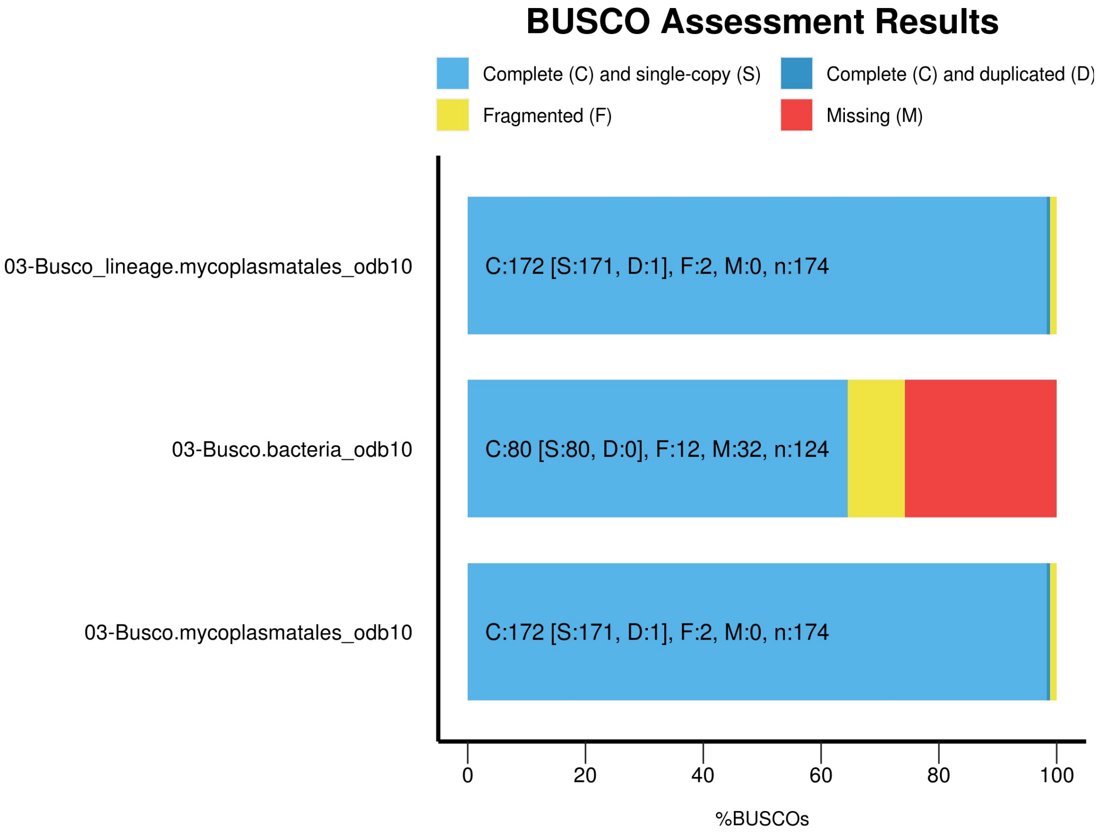
Final assembly length (is it close to the expected size?)
N50 (50% of the genome is in a contig of that size or larger)
L50 (as the smallest number of contigs whose length sum makes up half of genome size)
The percentage of Ns

We are going to use the program Quast

We are also going to view the assembly graphs using Bandage

Tutorial: assembly metrics

The number of conserved single-copy orthologs (e.g., BUSCO) that are complete, fragmented or duplicated


BUSCO Assessment Results
Complete (C) and single-copy (S)
Complete (C) and duplicated (D)
Fragmented (F)
Missing (M)
03-Busco_lineage.mycoplasmatales_odb10
C:172 [S:171, D:1], F:2, M:0, n:174
03-Busco.bacteria_odb10
C:80 [S:80, D:0], F:12, M:32, n:124
03-Busco.mycoplasmatales_odb10
C:172 [S:171, D:1], F:2, M:0, n:174
%BUSCOs