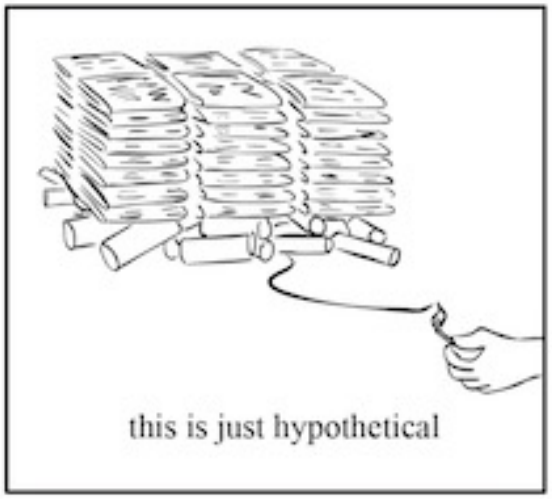
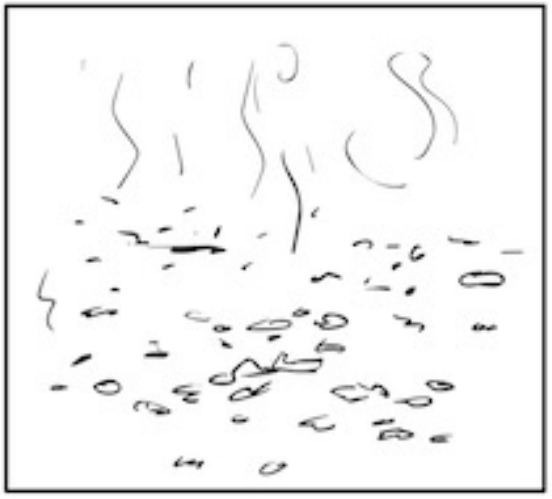
Topic 5: de novo assembly

Outcomes

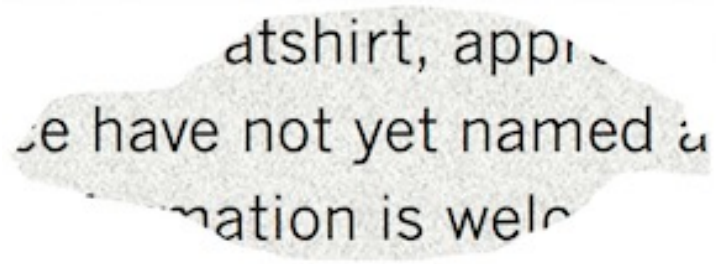
- Evaluate two different approaches to de novo genome assembly
- Understand factors complicating genome assembly
- Identify approaches for overcoming these complications
- Learn how to evaluate assembly quality

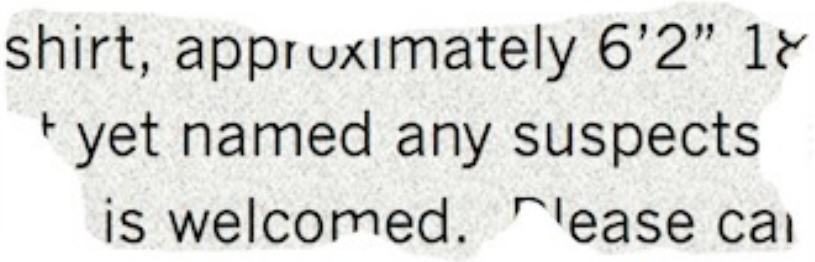
Introduction





Introduction



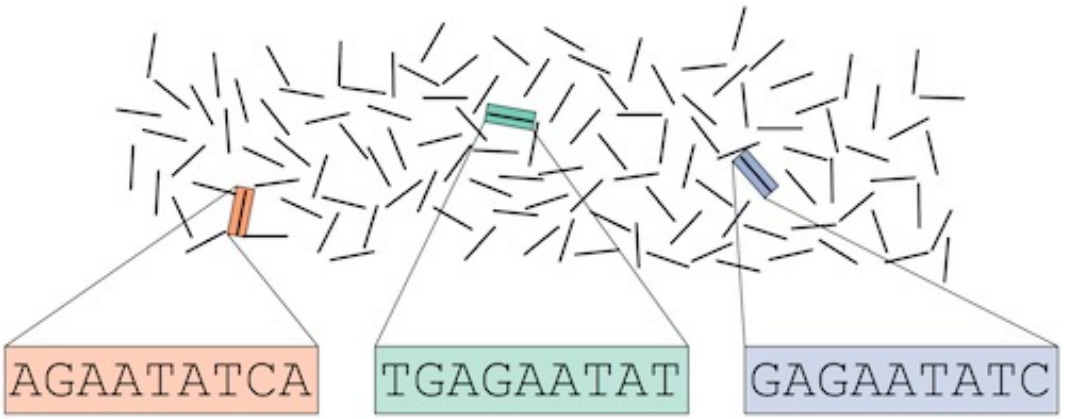


Introduction

Multiple identical copies of a genome

Shatter the genome into reads

Sequence the reads



Assemble the genome using overlapping reads

AGAATATCA
GAGAATATC
TGAGAATAT
...TGAGAATATCA...

Alignment vs assembly

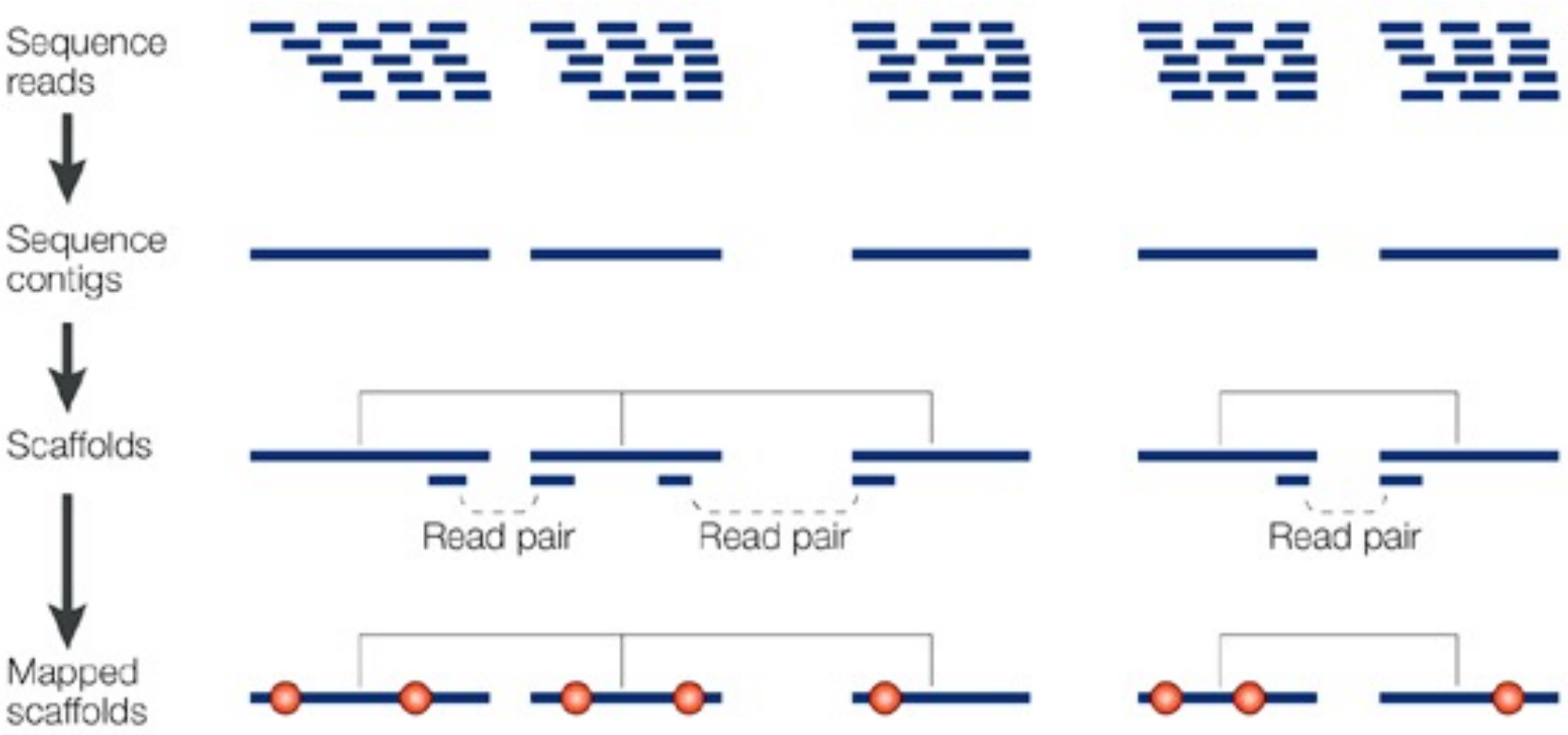
Aligning to a reference:

- 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

GATAGAAGGGTCCGCTCGCTCAGCTACCGGTTTTTTATAGATCTA

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

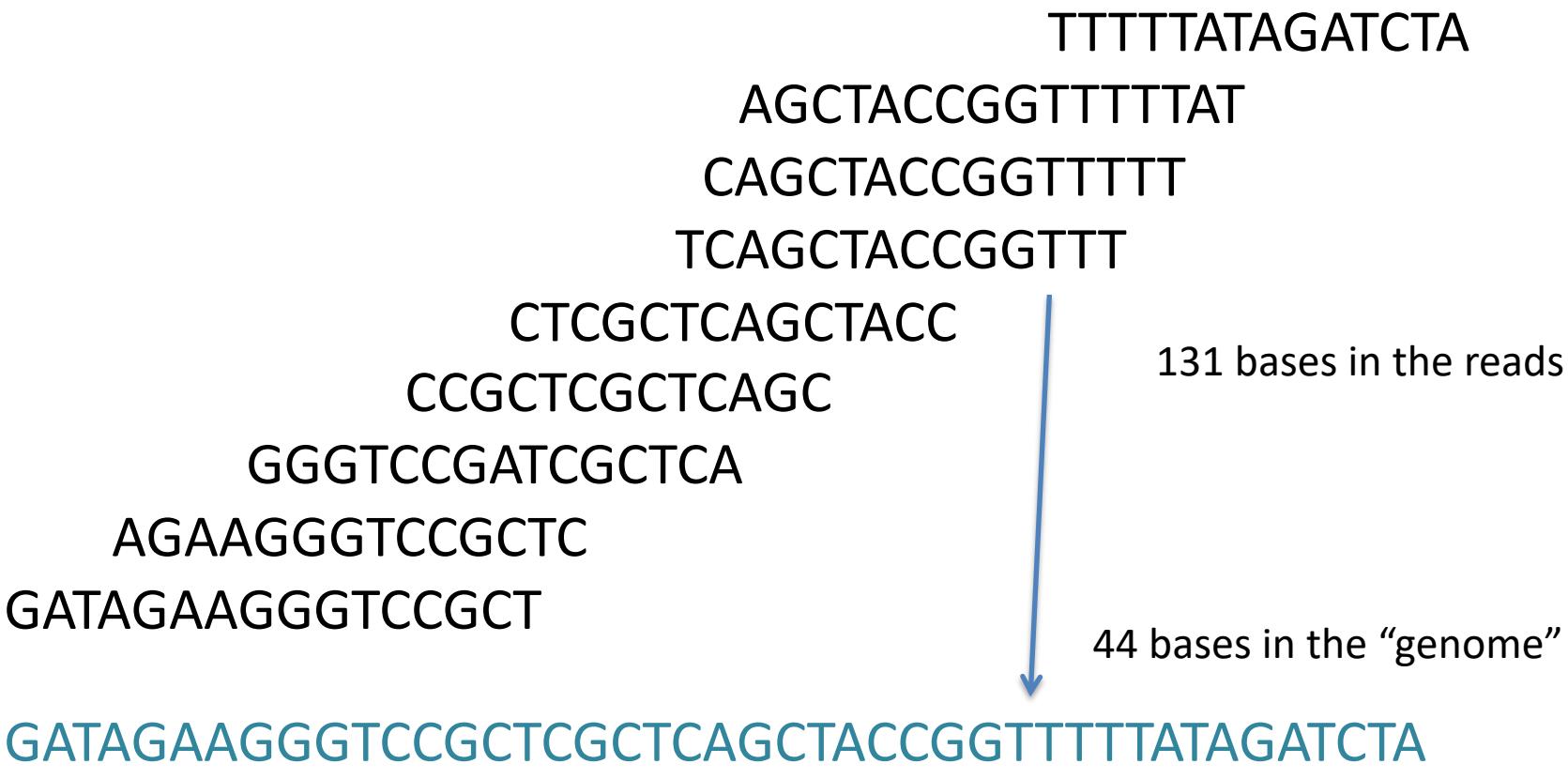
TTTTTATAGATCTA
AGCTACCGGTTTTTAT
CAGCTACCGGTTTTT
TCAGGTACCGGTTT
CTCGCTCAGCTACC
CCGCTCGCTCAGC
GGGTCCGATTGCTCA
AGAAGGGTCCGCTC
GATAGAAGGGTCCGCT

GATAGAAGGGTCCGCTCGCTCAGCTACCGGTTTTTATAGATCTA

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

Introduction

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



What is the coverage at the arrow?

What is our average coverage?

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 (i.e. the beginning or the end) exactly 4 bp in length between CTCTAGGCC and a list of other sequences in the file /mnt/data/codebreaks/overlaps.fa

What complicates assembly?

- Repetitive sequence
  - causes nodes to be shared, locality confusion
- Variation and Heterozygosity
  - Causes lots of branch points in a graph
- Sequencing read issues (technical, but very important)
  - Coverage
  - Sequencing artifacts (adapters, vector sequence, etc)
  - Sequence errors
    - Generally random, but some technologies have systematic errors

Assembly Algorithms

Overlap-Layout-Consensus (OLC)

- **Very** computationally intensive
- Makes use of read-level information

De Bruijn graphs

- More efficient algorithm
- Some loss of information
- Developed to deal with high coverage, short read data

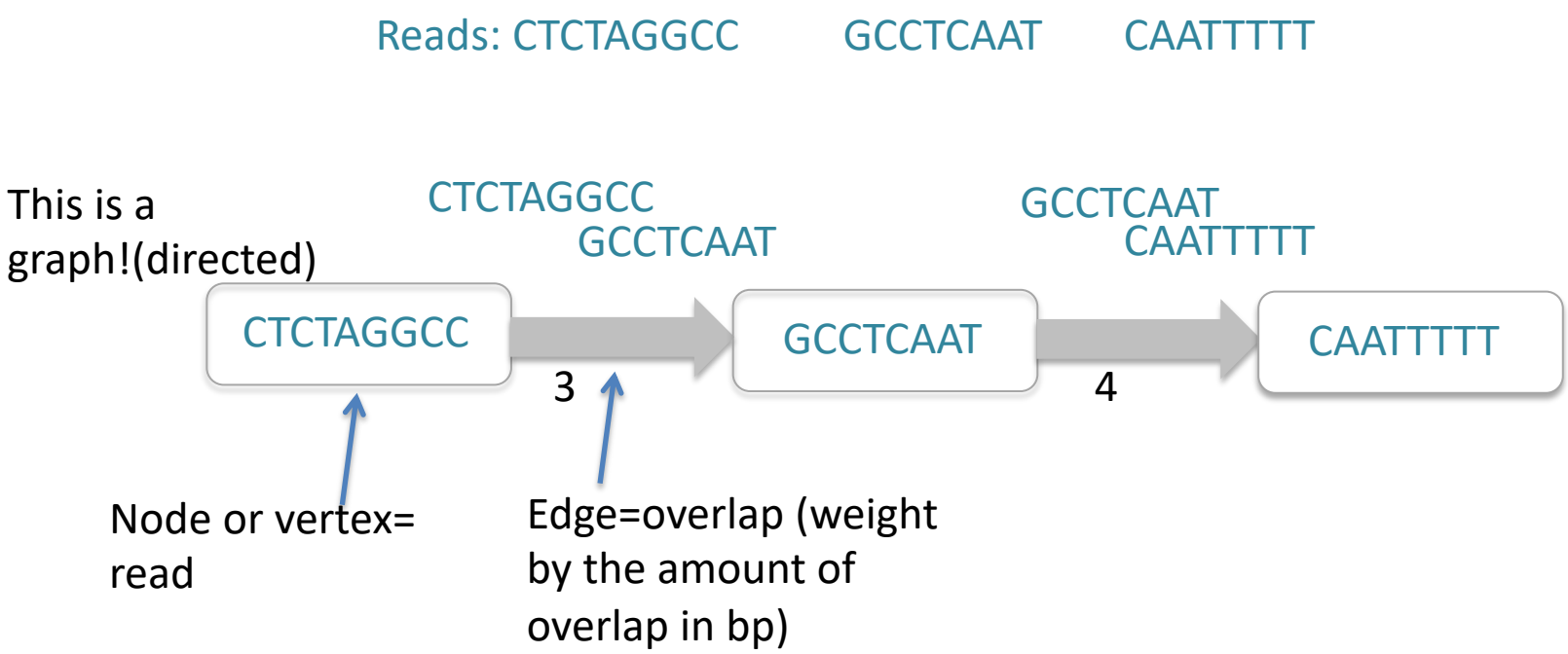
Overlap-layout-consensus

Overlap: make an overlap graph

Layout: find the path through the graph

Consensus: find the most likely contig sequence

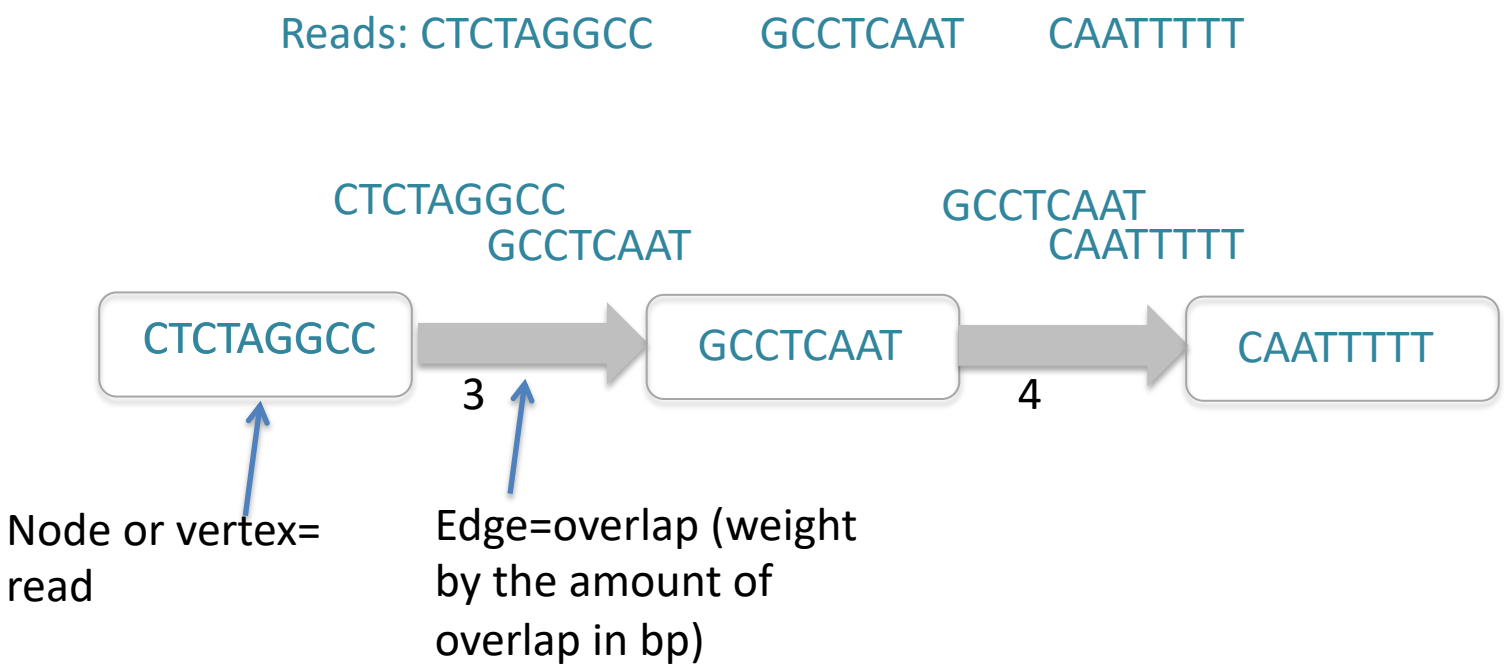
Overlap-layout-consensus



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

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

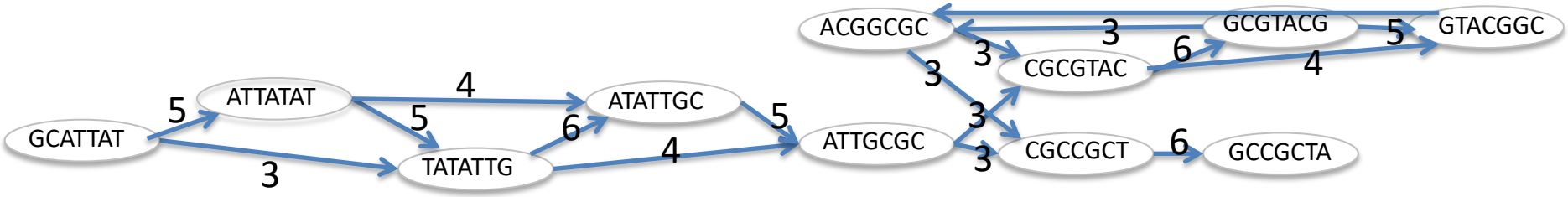
Overlap-**layout**-consensus



Here we have only one path through the graph

Overlap-**layout**-consensus

These graphs get complicated!



Minimum
overlap = 3
Read length = 7

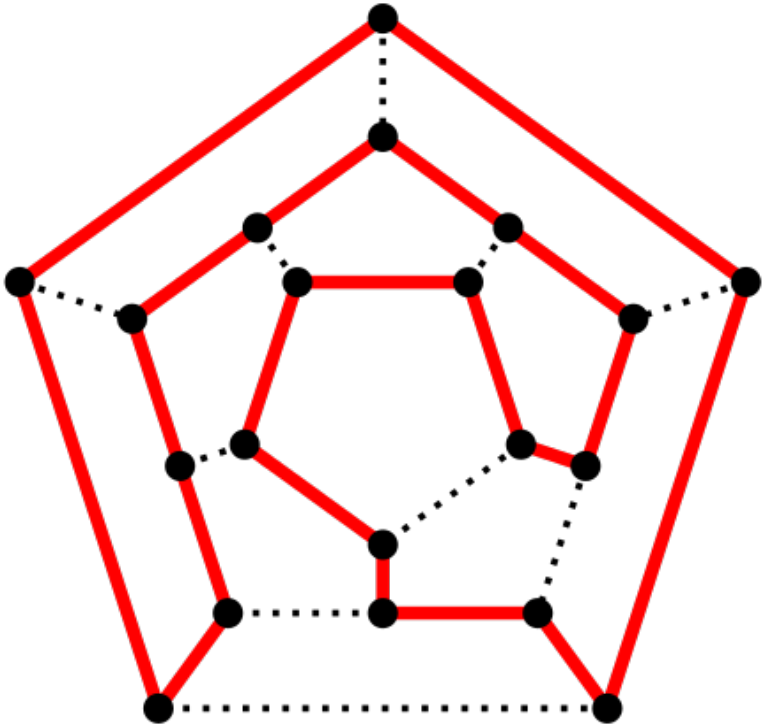
GCATTATATATTGCGCGTACGGCGCCGCTACA

Original sequence

How can we find the best path?

Overlap-**layout**-consensus

Hamiltonian path: hit each node (read) once
–no quick or strategic way to figure it out
–not practical and not implemented

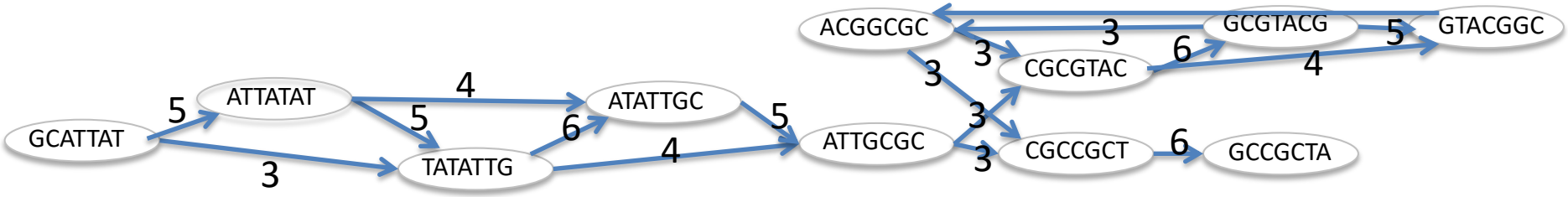


Overlap-**layout**-consensus

Shortest superstring: find the shortest final sequence (greatest overlap between reads)
- Even harder than finding the Hamiltonian path

Overlap-**layout**-consensus

These graphs get complicated!



Minimum
overlap = 3
Read length = 7

GCATTATATATTGCGCGTACGGCGCCGCTACA

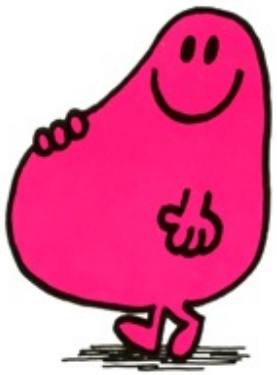
Original sequence

How can we find the best path?

Overlap-**layout**-consensus

Greedy algorithm (example)

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



the greedy one

Overlap-layout-**consensus**

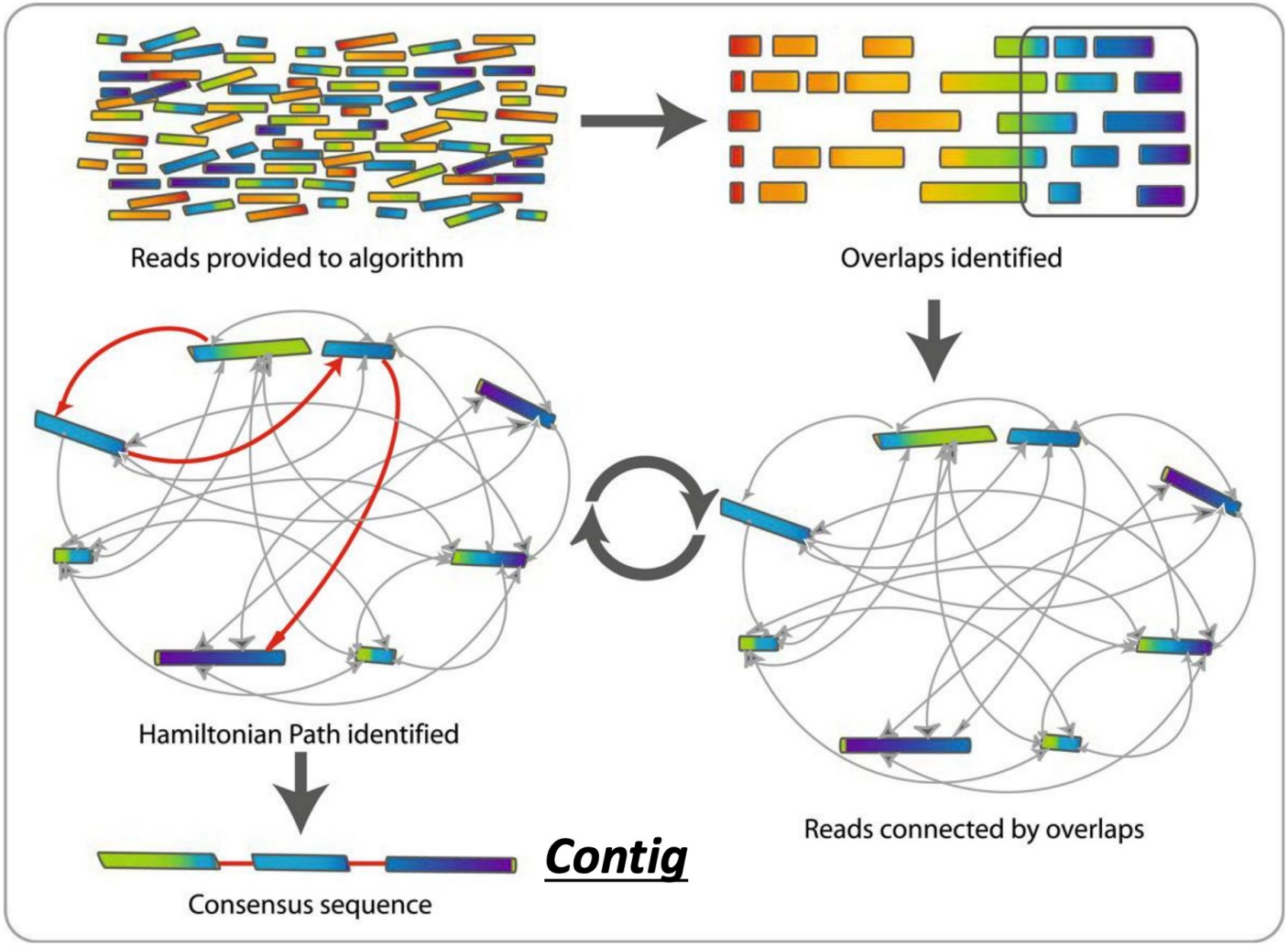
Join sequences together into one sequence

CTCTAGGCC
GCCTCAAT
CAATTTTT

CTCTAGGCCTCAATTTTT

Find a path which visits each node once

Form consensus sequences from paths



Limitations of OLC

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

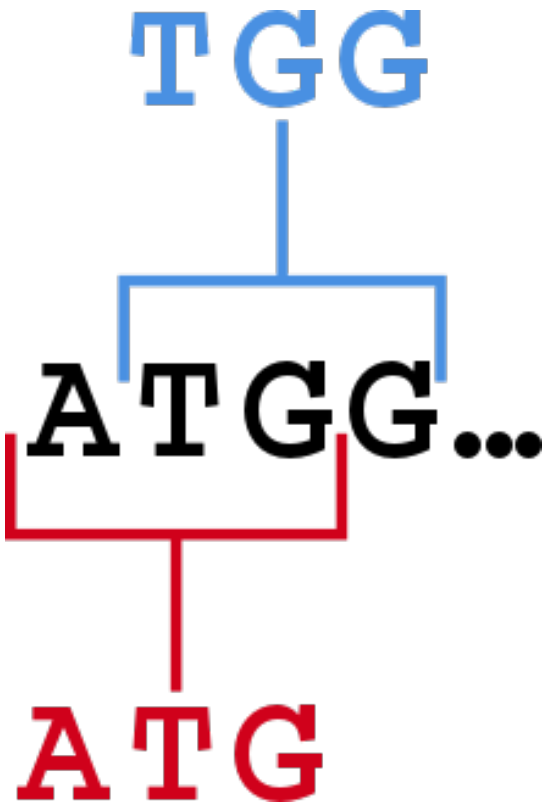
- finding the best path through the graph with a huge number of nodes (reads) is computationally challenging

De Bruijn graphs

Simplify the problem by simplifying the information!

k-mers

A substring of length k



This 4bp sequence has two 3mers

k-mers

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

2 reads of 9 bp

read 1
ATGGGGAAC

ATGGG
TGGGG
GGGGA
GGGAA
GGAAC

read 2
GGGAACCCC

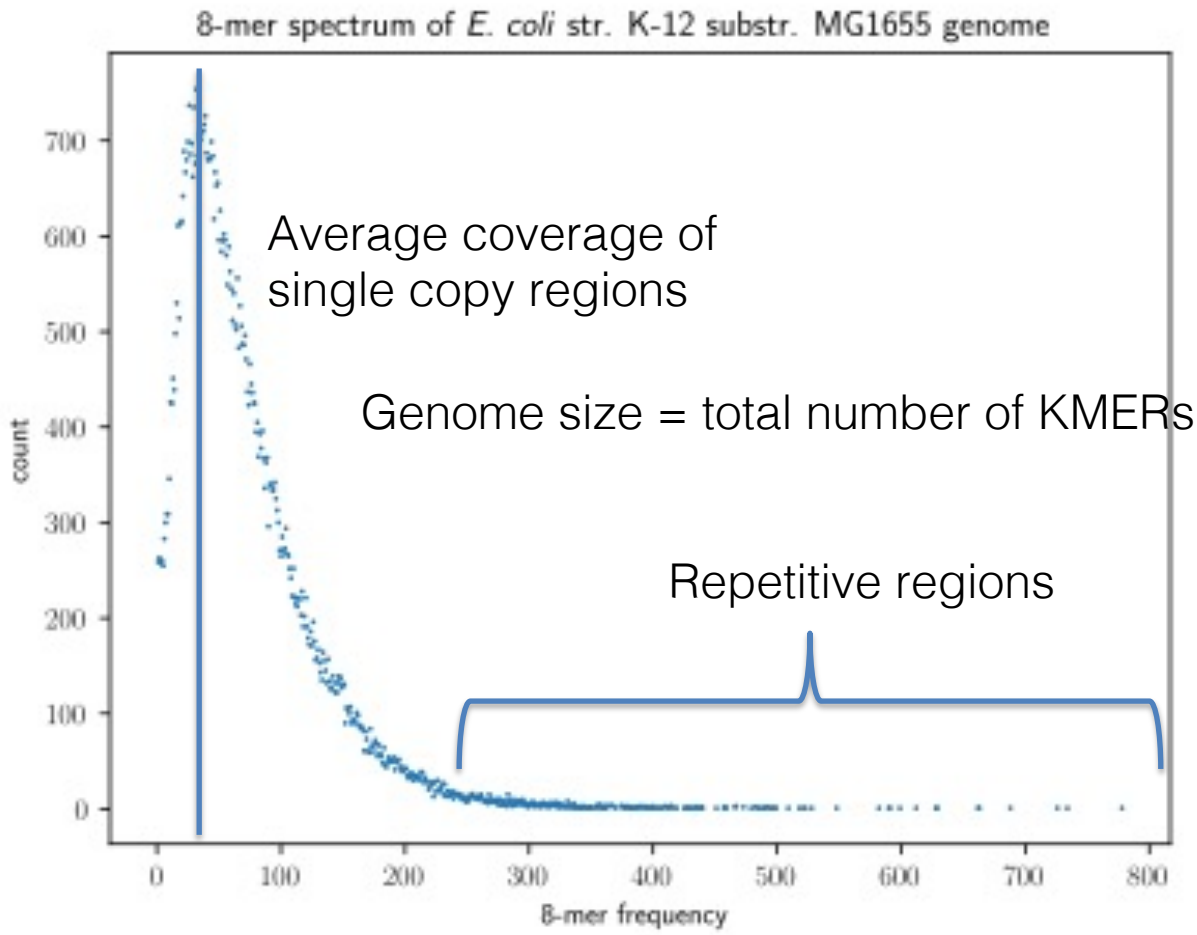
GGGAA
GGAAC
GAACC
AACCC
ACCCC

Short problem:

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

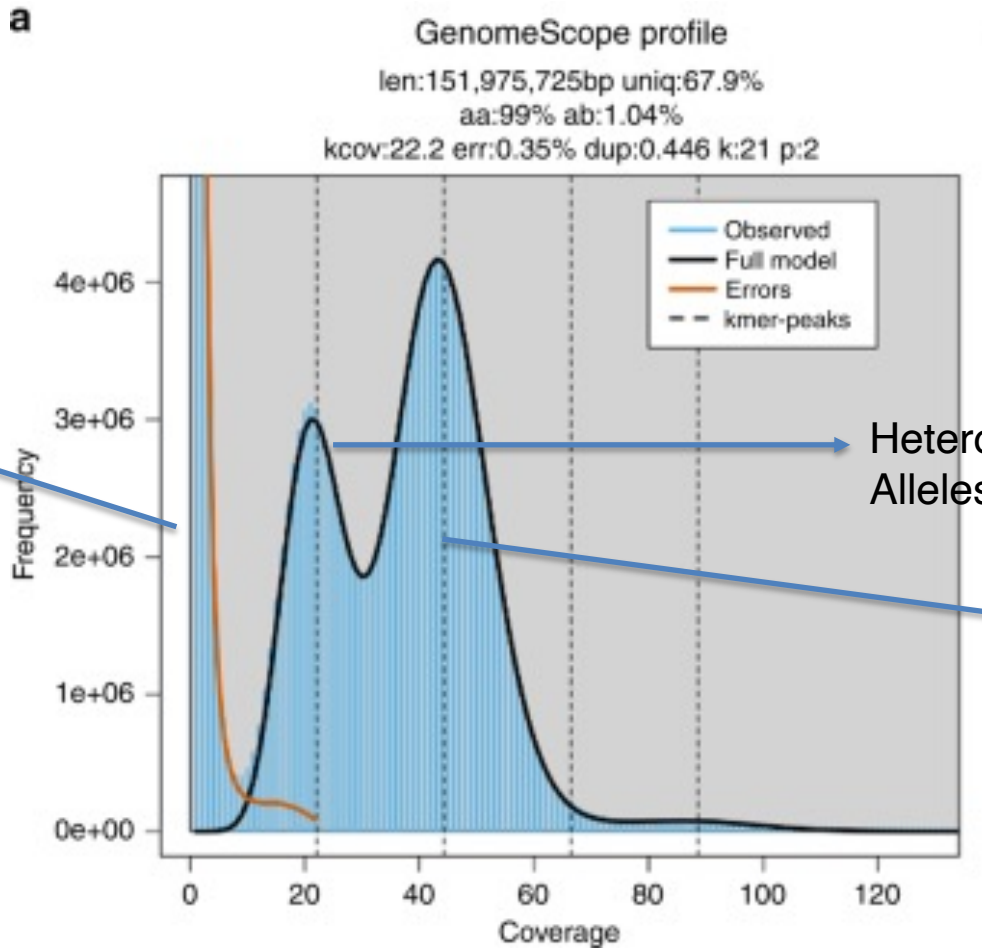
k-mers

The frequency of KMERs tells us a lot about the genome & our sequencing



k-mers

The frequency of KMERs tells us a lot about the genome & our sequencing



Unique KMERs with very low coverage

Heterozygous Alleles

Homozygous alleles

Code break

Find all the 9mers in a fasta sequence
/mnt/data/codebreaks/kmer.fa

1. Find all the kmers in this fasta sequence.
   Hints: test out the following commands

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

var=`echo $((1+1))`

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

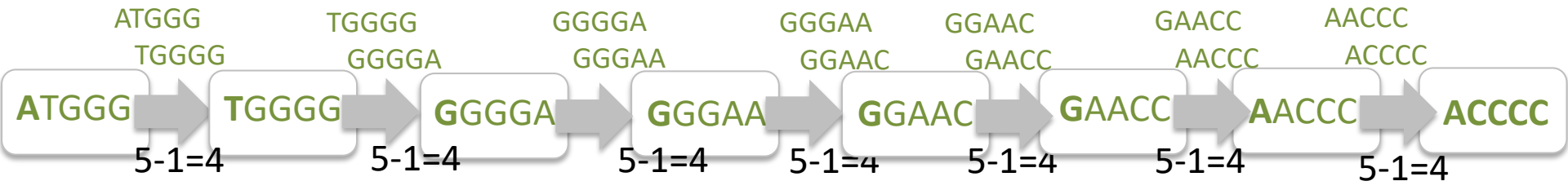
De Bruijn graphs

- 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

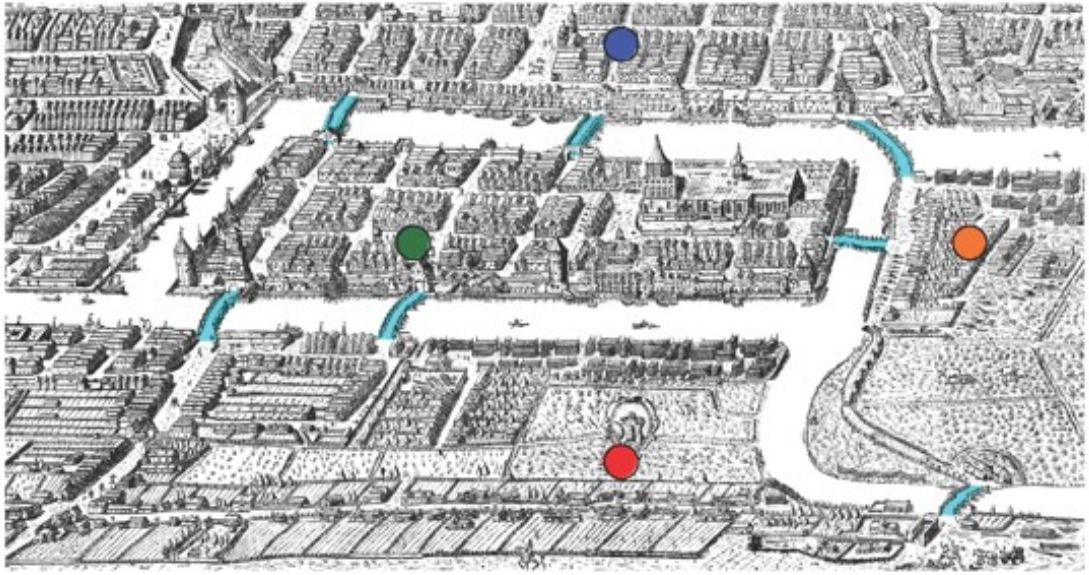
ATGGG
TGGGG
GGGGA
GGGAA
GGAAC
GAACC
AACCC
ACCCC



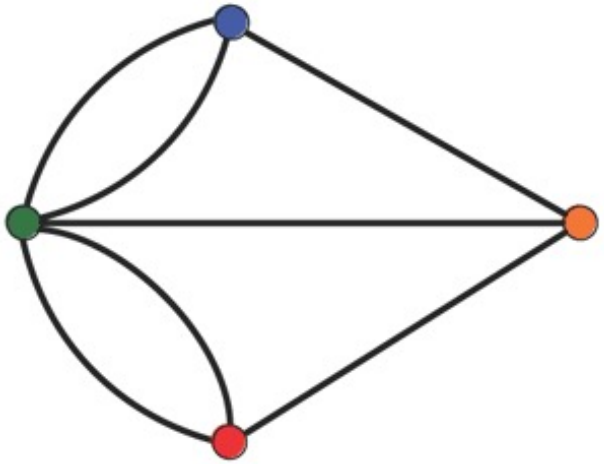
- Traverse through the graph
- The first base of each node spells out the sequence

De Bruijn graphs

a

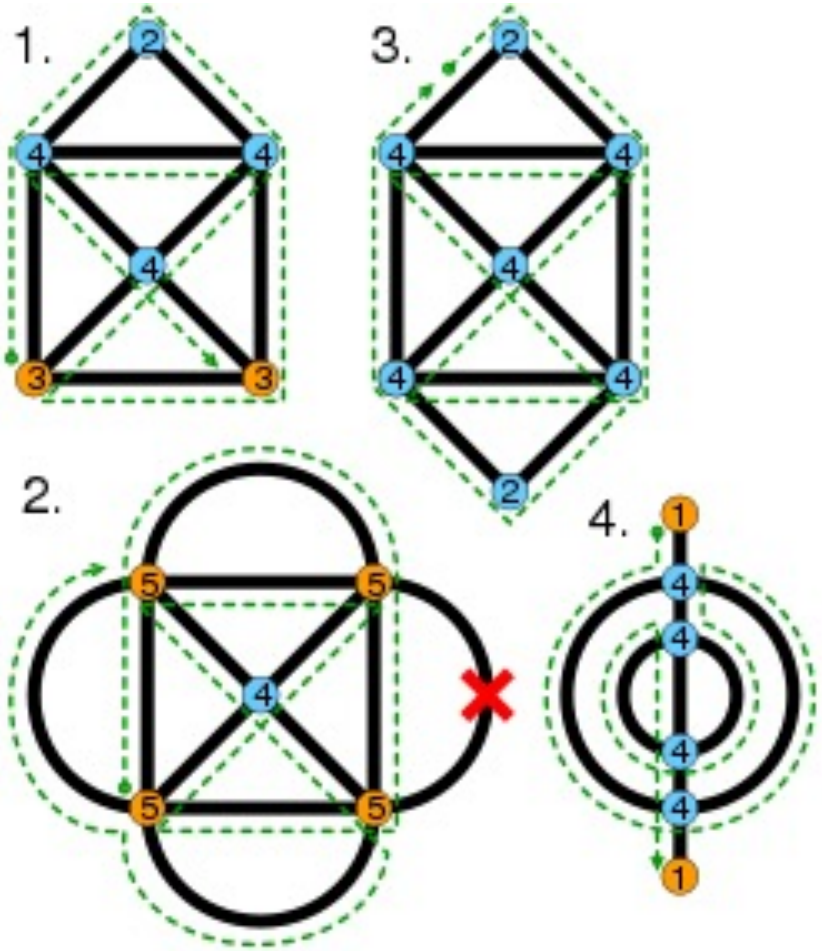


b

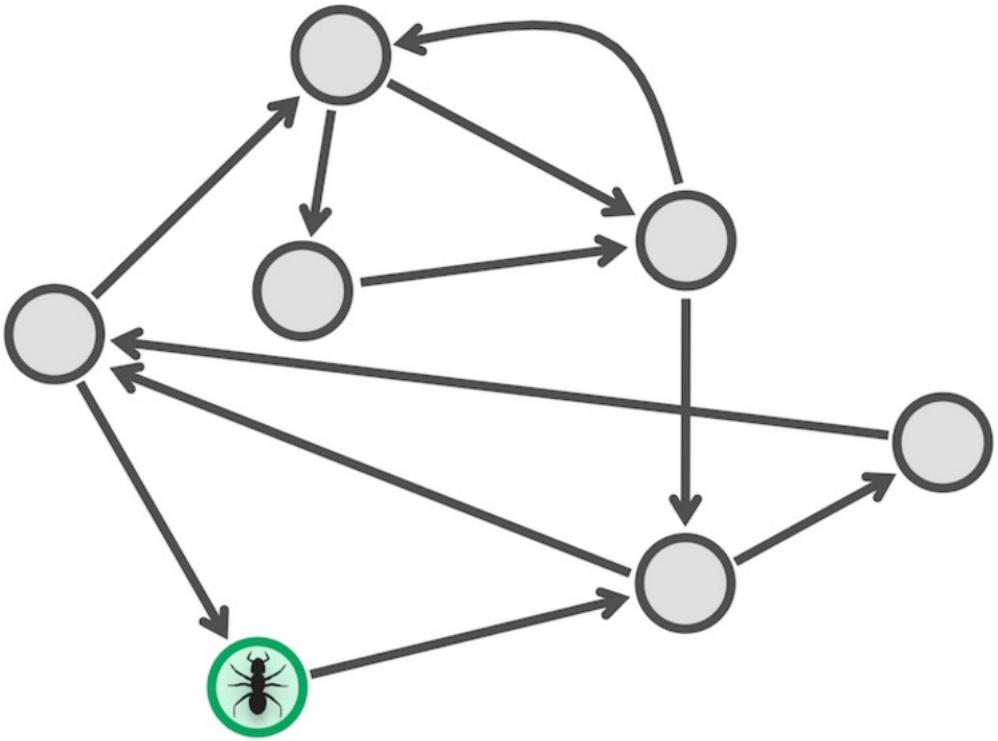


De Bruijn graphs

Eulerian graph must be both balanced and strongly connected

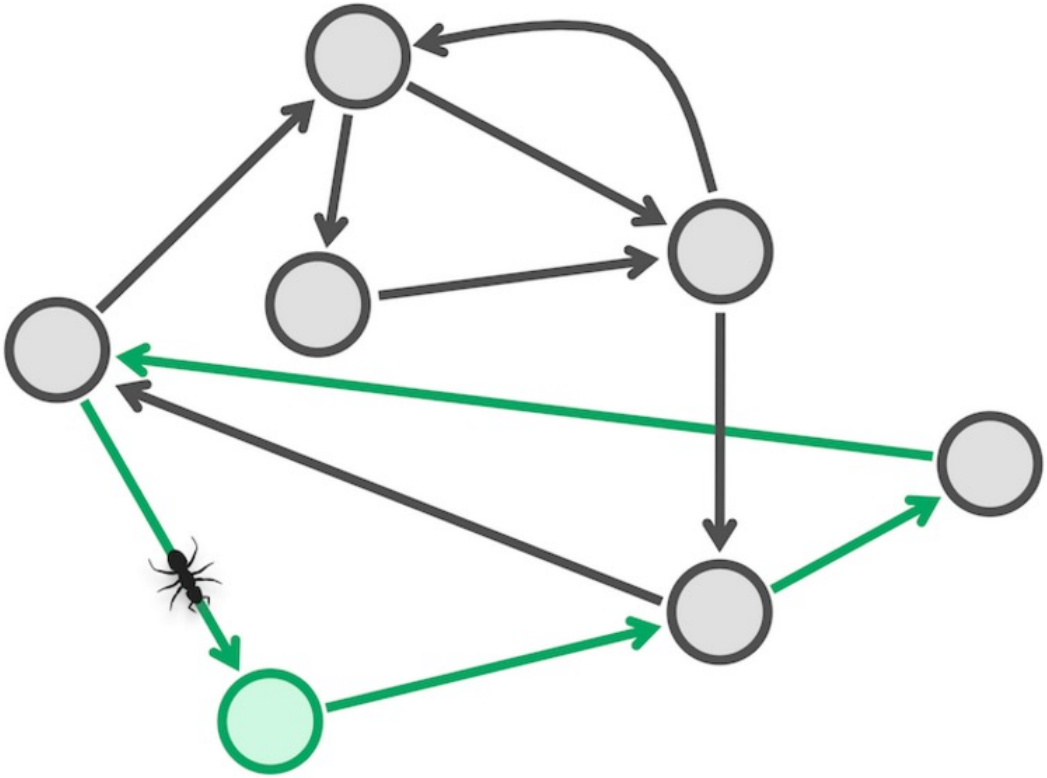


De Bruijn graphs



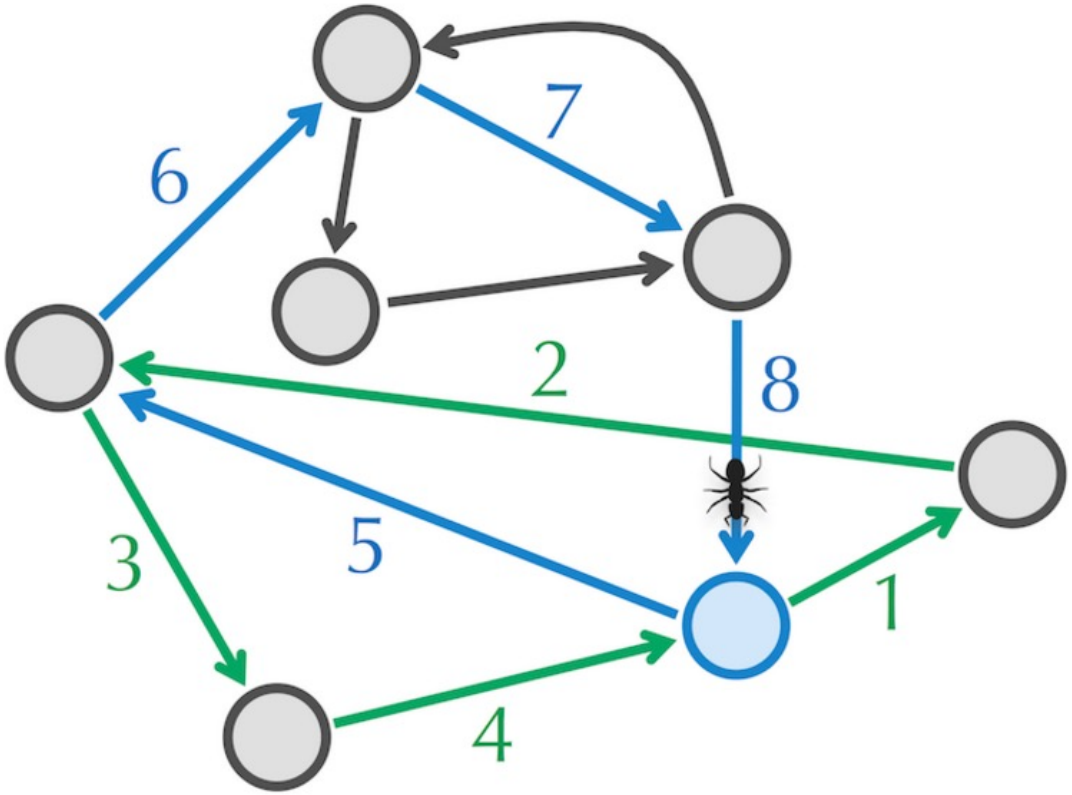
Algorithm to find a path through an Eulerian graph

De Bruijn graphs



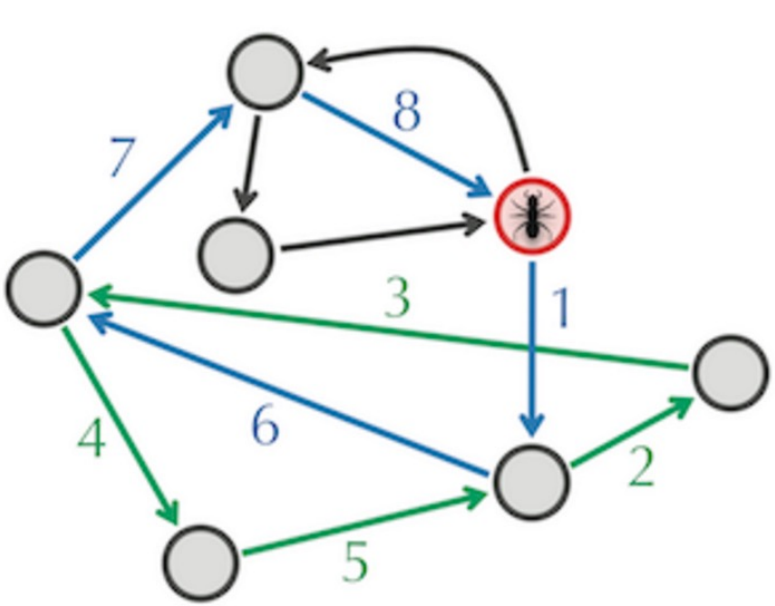
Algorithm to find a path through an Eulerian graph

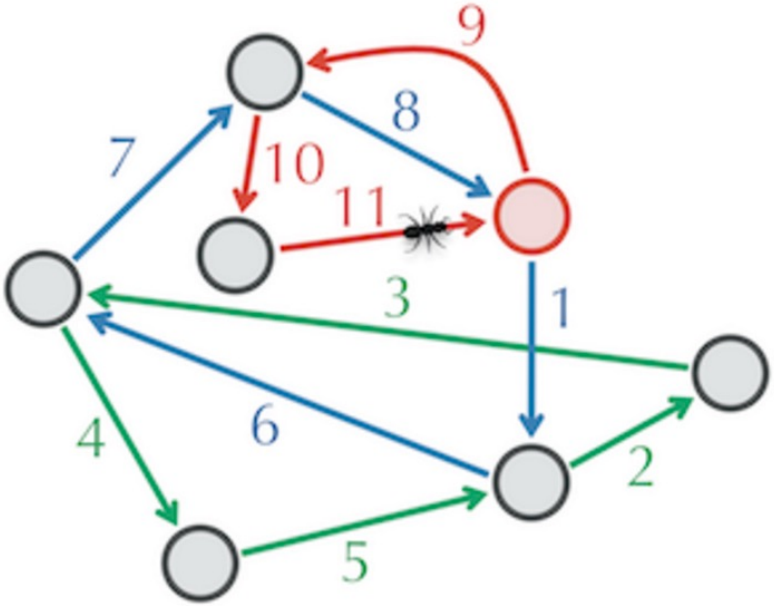
De Bruijn graphs



Algorithm to find a path through an Eulerian graph

De Bruijn graphs





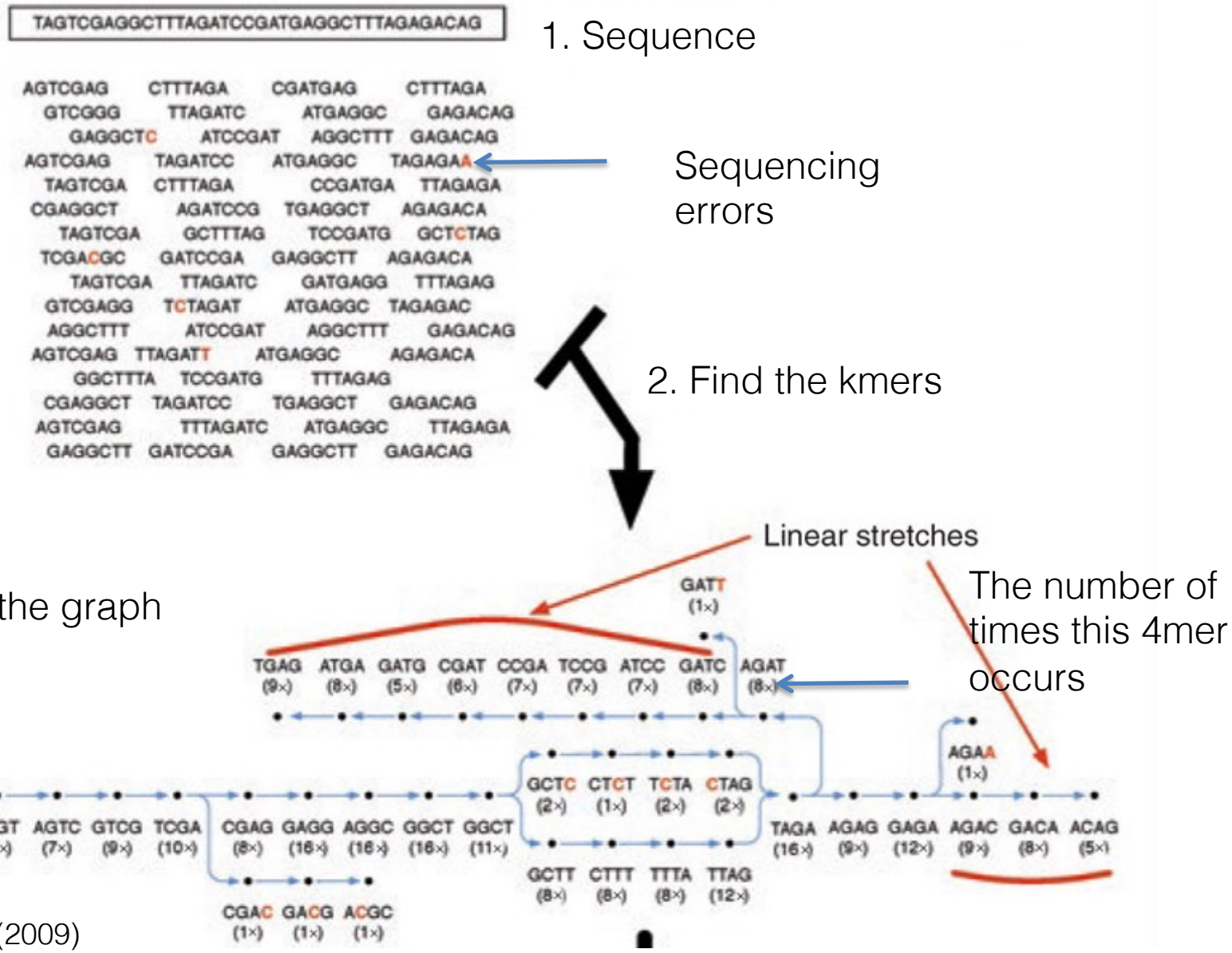
Algorithm to find a path through an Eulerian graph

De Bruijn graphs

Limitations of the Eulerian path:

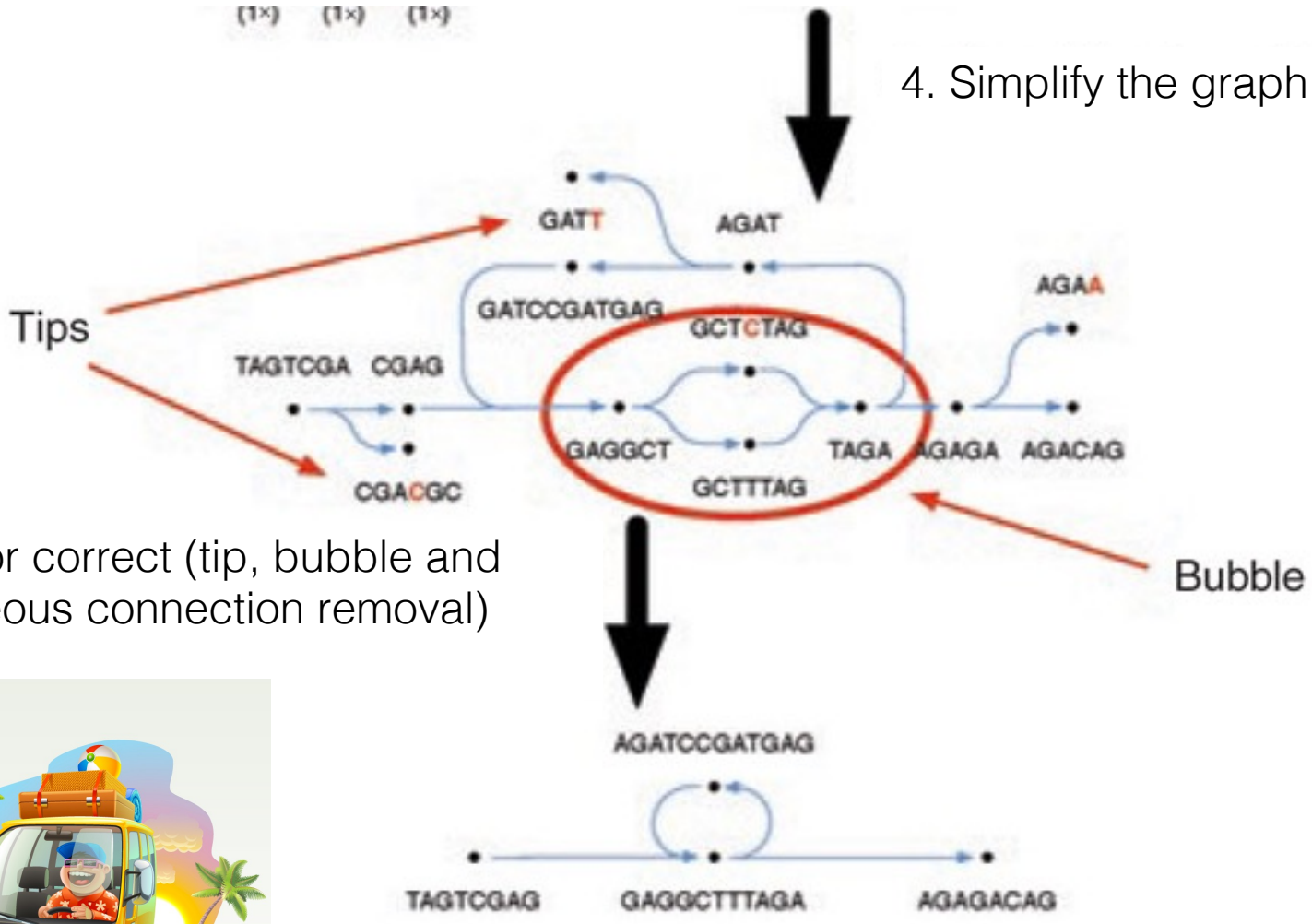
- With “perfect” genomic data there are usually many Eulerian tours
- Data is not perfect (areas of low coverage, errors, repeats, etc.)

De Bruijn graphs



Flicek & Birney
Nature Methods (2009)

De Bruijn graphs



Flicek & Birney Nature Methods (2009)

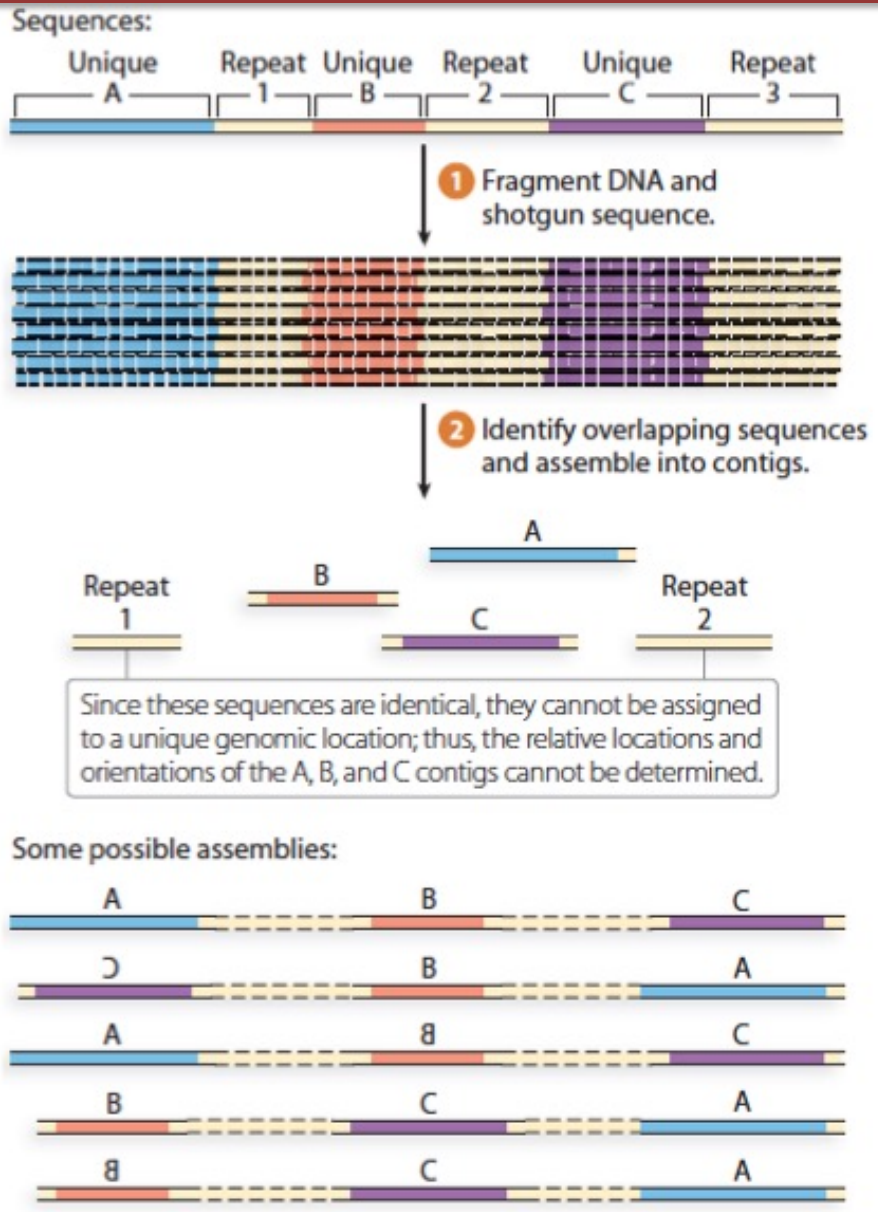
De Bruijn graphs

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



Sanders and Bowman

Figure 18.2 The problem of repetitive DNA.

Repetitive regions

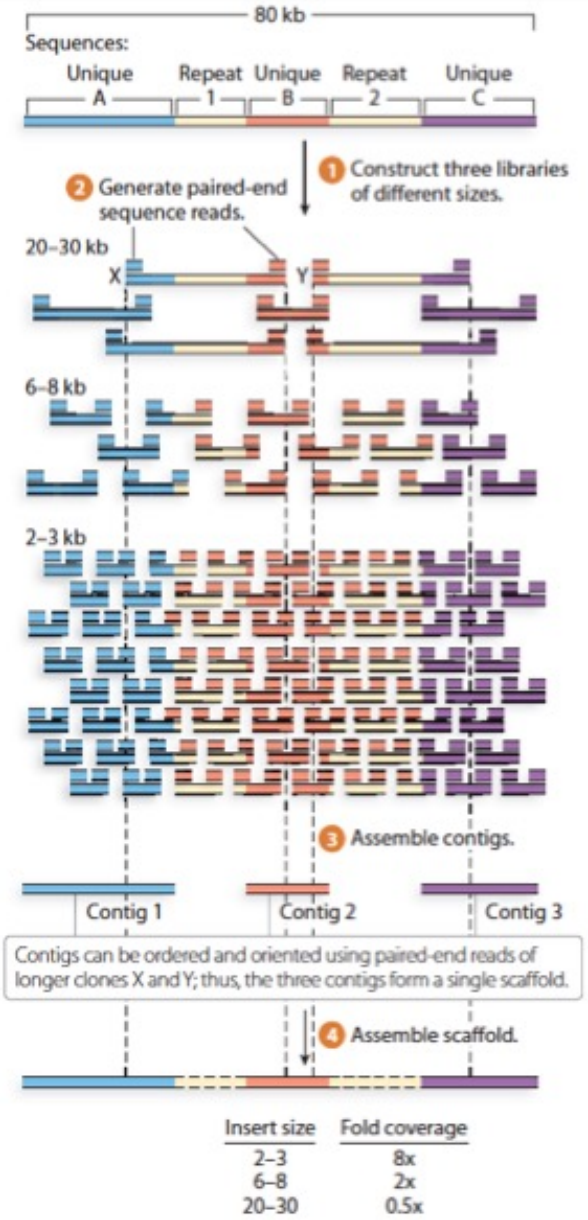


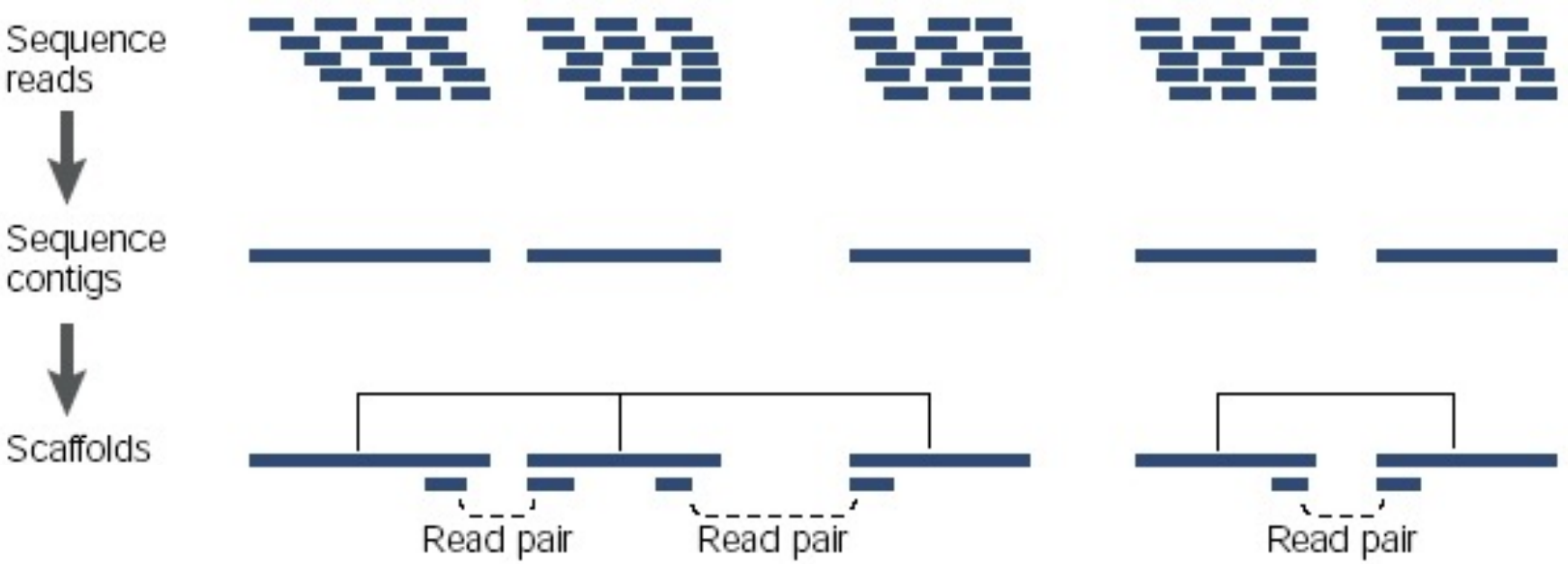
Figure 18.3 Paired-end shotgun sequencing strategy.

Sanders and Bowman

Repetitive regions

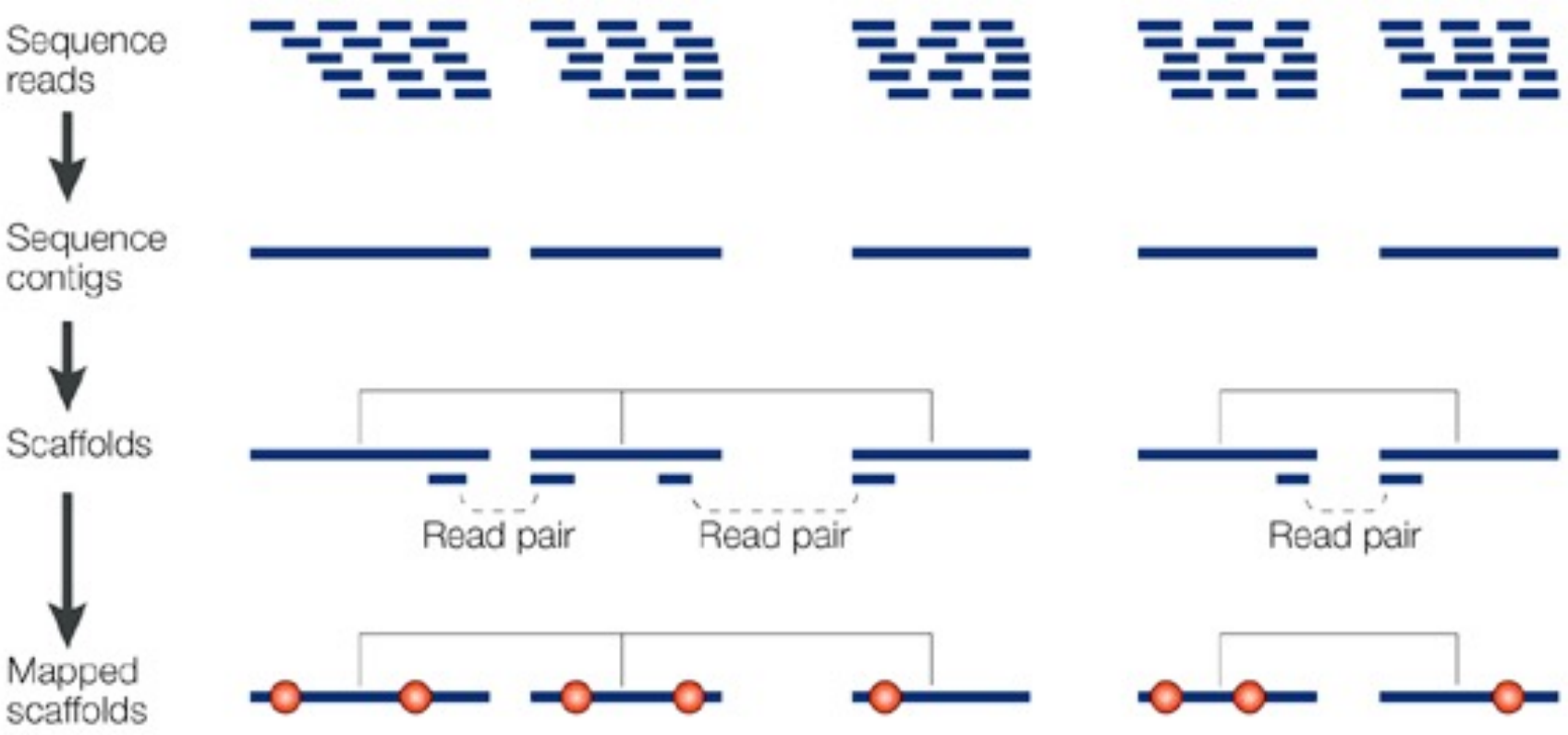
- Finishing eukaryotic genome assemblies can be challenging because much of the genome is repetitive
- This repetitive DNA breaks up the assembly and obscures the order and orientation of the assembled contigs
- Even well studied model organisms can have poorly assembled regions of their genome

Repetitive regions



Adams 2008

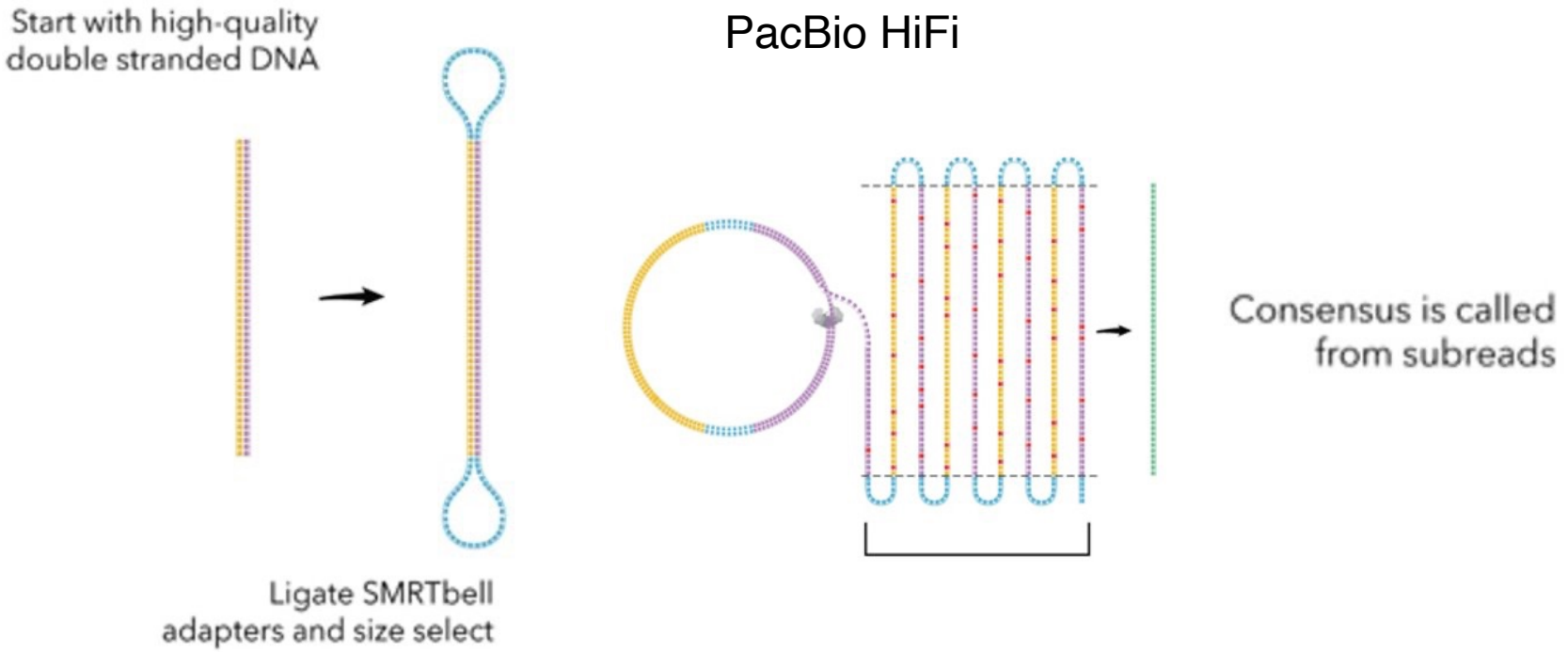
Repetitive regions



Adams 2008

Current assembly approaches

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



Long read assemblies

Long read only *de novo* assembly. PacBio/Nanopore reads are assembled using an OLC algorithm (>50x PacBio)

Hybrid *de novo* assembly. Error correct long reads with more accurate short reads before performing long read assembly. (~20x PacBio)

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

Scaffolding. Using an *existing* (short read) assembly, 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)]

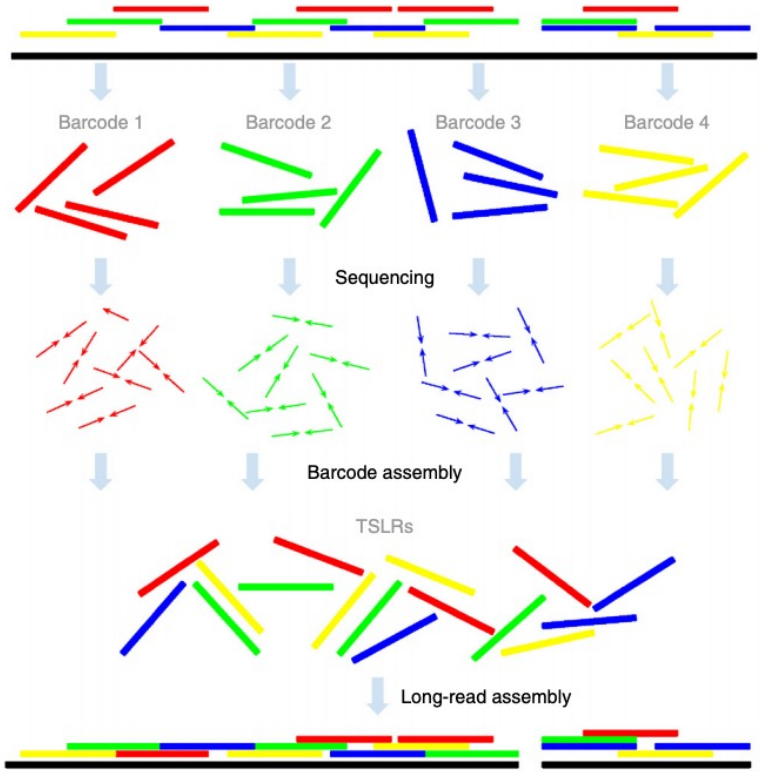


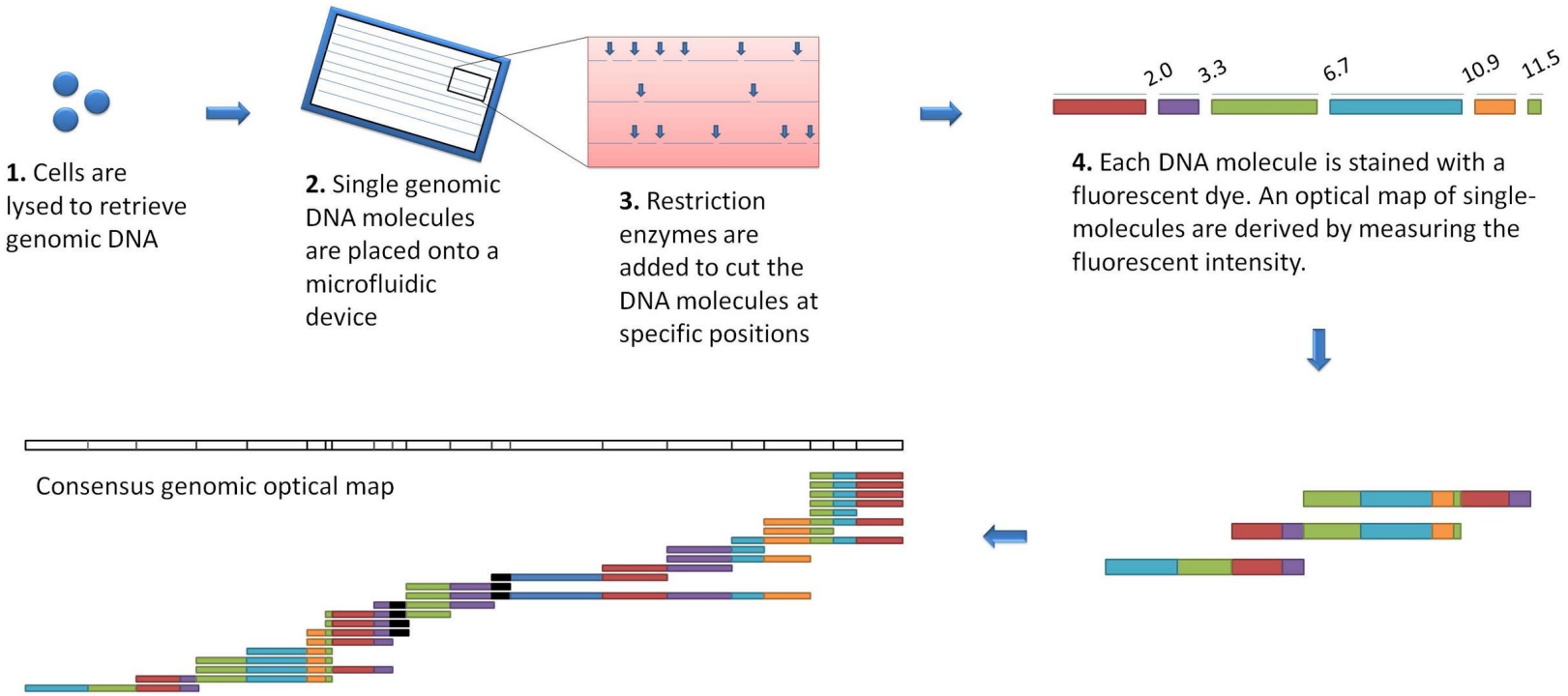
Figure 1 | The TSLR technology. The barcode assembly step generates virtual long reads. In an idealized scenario, the barcode assembly would result in ~300 TSLRs with lengths of ~10 kb. In reality, it results in 350–450 TSLRs varying in length from 1 to 10 kb.

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

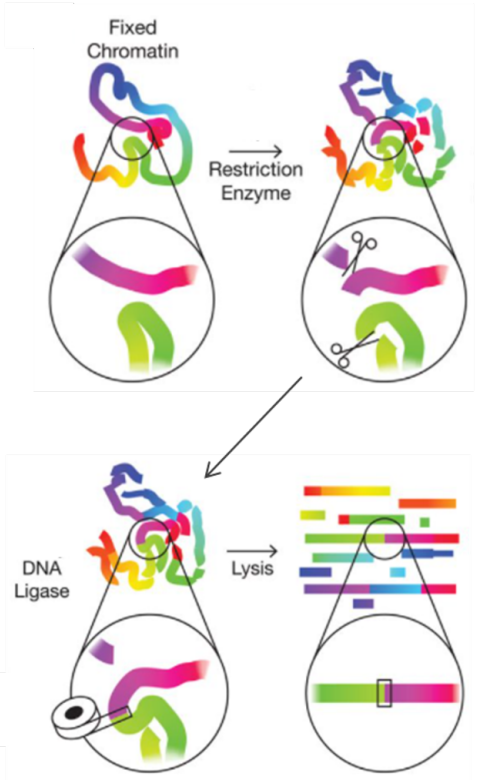
- Bionano Genomics https://vimeo.com/116090215

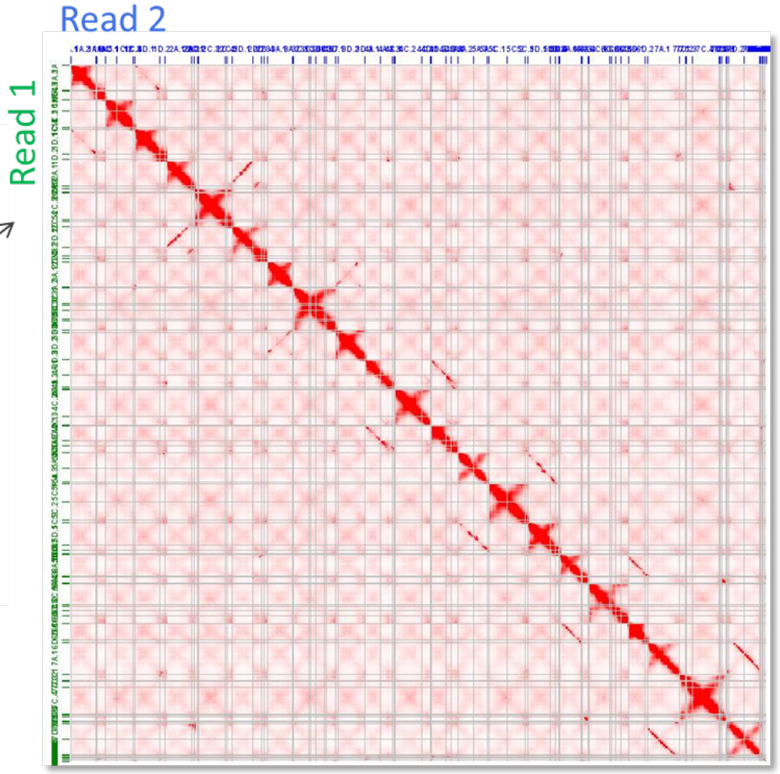


Long-range scaffolding technologies

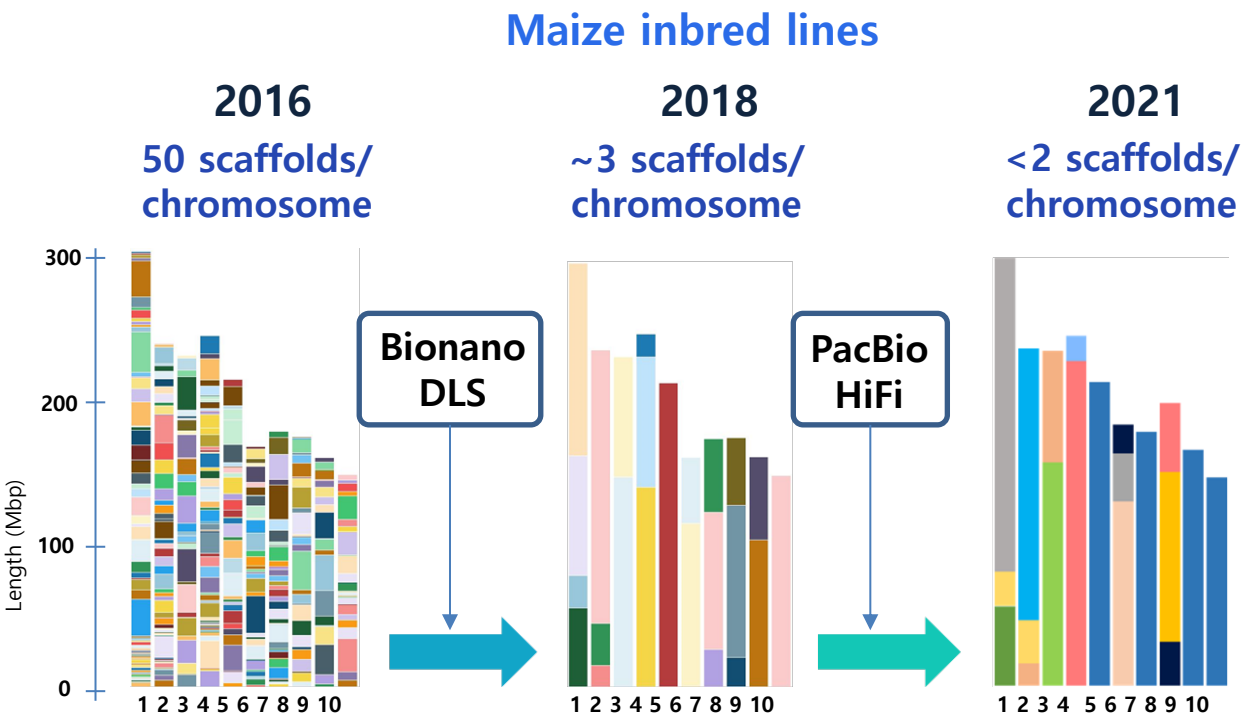
Hi-C

- Short read data and specialized library prep
- Frequency of 3D interactions over long ranges





Advances in chromosome reconstruction



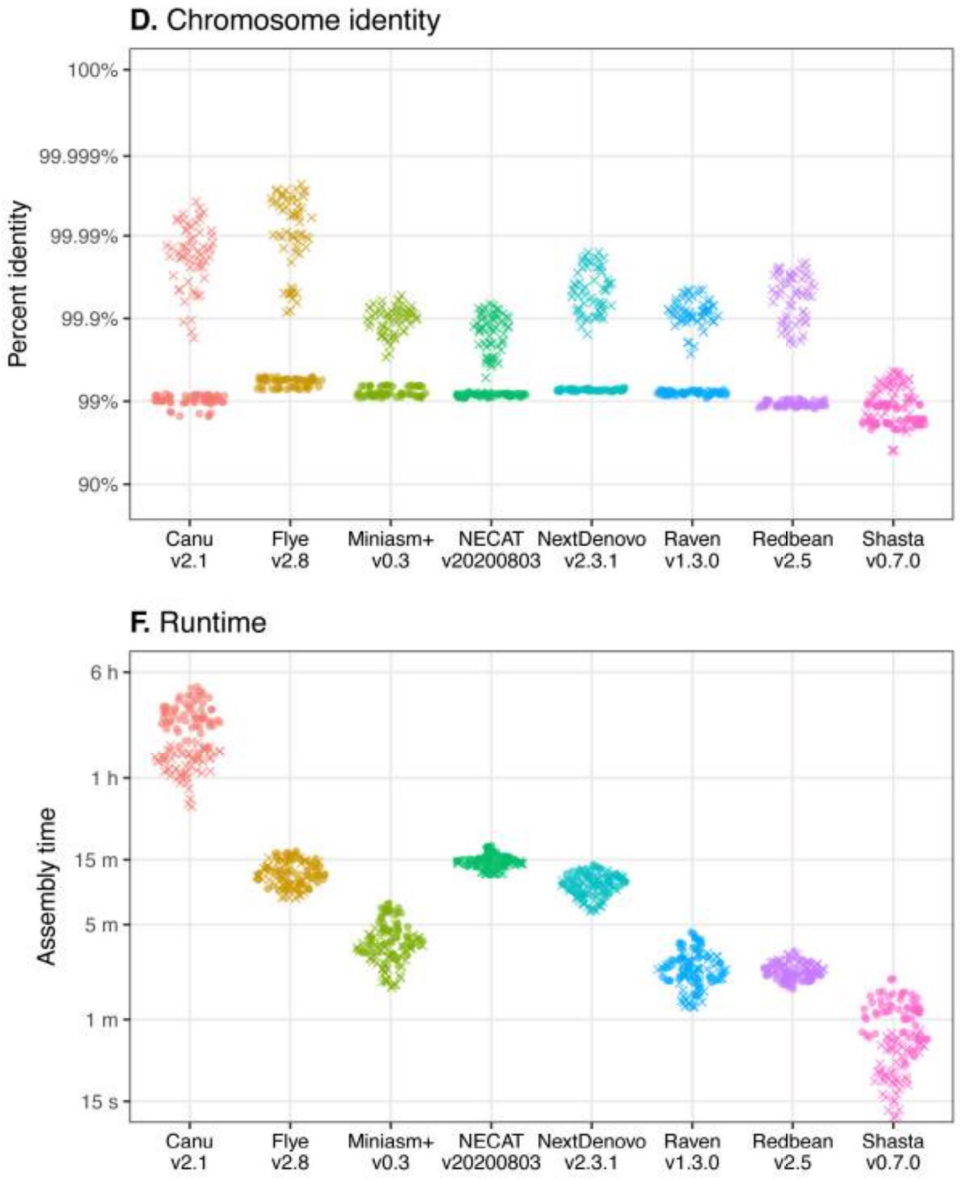
Innovation
Early adoption

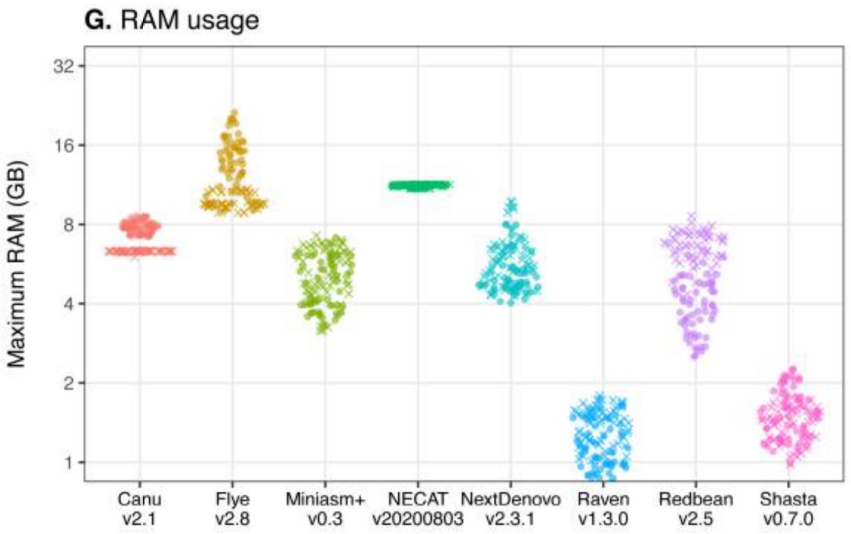
5-year gains:

25X contiguity
40X cost
15X time

Pacbio HiFi (low error rate long reads) + Scaffolding (Hi-C/Bionano)

But, not one perfect assembler!





Wick et al., 2019

Assemblers we will be using

- Spades (2012) - de Bruijn graph assembler
  - Hybrid version: gap closure and repeat resolution
- HASLR (2020) – computational “synthetic read” approach: hybrid assembly
  - Align short reads to long reads
- Flye (2019)
  - Long read assembly
  - Approach to deal with highly fragmented graphs (i.e. repetitive genomes)
    - Local-self alignment to get “repeat graphs”

How good is my assembly?

- How much total sequence is in the assembly relative to estimated genome size?
- How many pieces, and what is their size distribution?
- Are the contigs assembled correctly?
- Are the scaffolds connected in the right order / orientation?
- How were the repeats handled?
- Are all the genes I expected in the assembly?

The three c's: *contiguity, completeness, correctness*

Tutorial: assembly metrics

There are a number of metrics to asses genome quality. 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 to investigate these stats.

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.

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/