Genome assembly and annotation

Day 3: Genome assembly

Antti Karkman

Department of Microbiology – UH

antti.karkman@helsinki.fi

Slides modified from Igor Pessi, with permission

Aims for this part of MMB-114

Day 1: Basics of UNIX and working with the command line

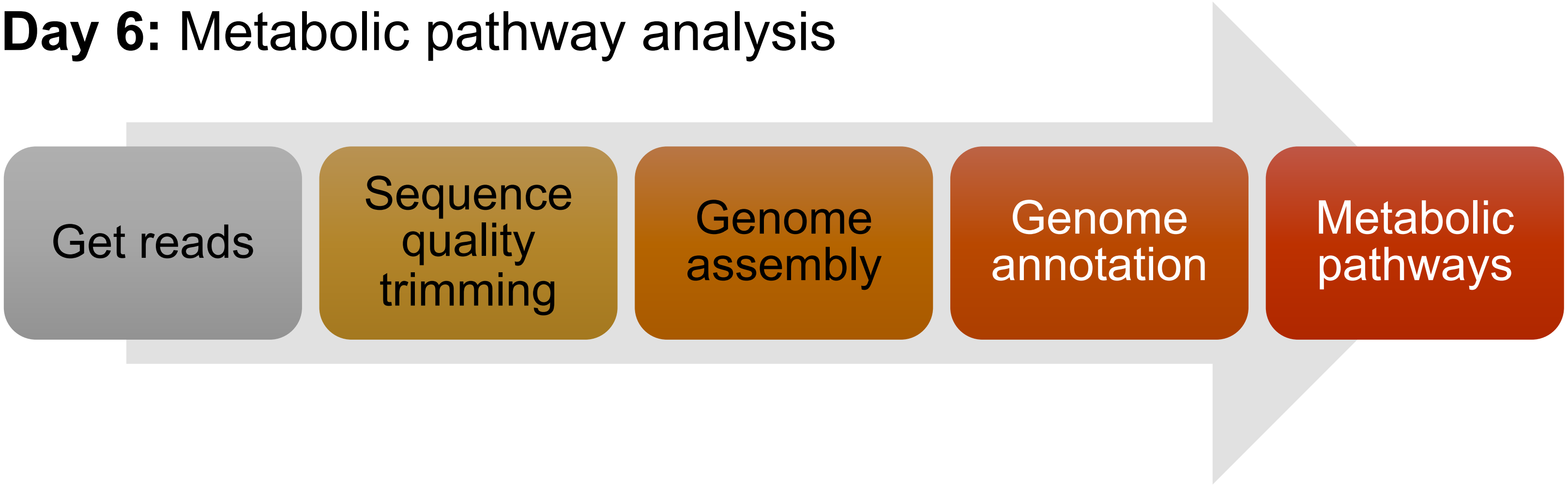
Day 2: Handling of Illumina data

Day 3: Genome assembly

Day 4: Check-up and report

Day 5: Genome annotation

Day 6: Metabolic pathway analysis



Get reads

Sequence quality trimming

Genome assembly

Genome annotation

Metabolic pathways

Before we start...

How does the raw data look like?

How many sequences we have for each genome? Are these numbers the same for the R1 and R2 files?

Is the length of the reads the same in the R1 and R2 files?

The "Per base sequence quality" module shows a problem. Why?

- Which part of the reads have the best quality? Beginning, middle, end?

What is the mean sequence quality for the majority of the reads?

The "Per base sequence content" module shows a problem. Why?

The "Adapter content" module also shows a problem. Why?

- Are there adapters in our reads?
- In which part of the reads?
- What are adapters and why should we remove them?

Which of the three genomes has the best sequence quality?

How does the data look like now?

CUTADAPT

How many times adapters were trimmed in R1 and R2?

How many reads were removed because they were too short?

How many low-quality bases were trimmed?

FASTQC

Do the "Per base sequence quality" plots look different now? Why?

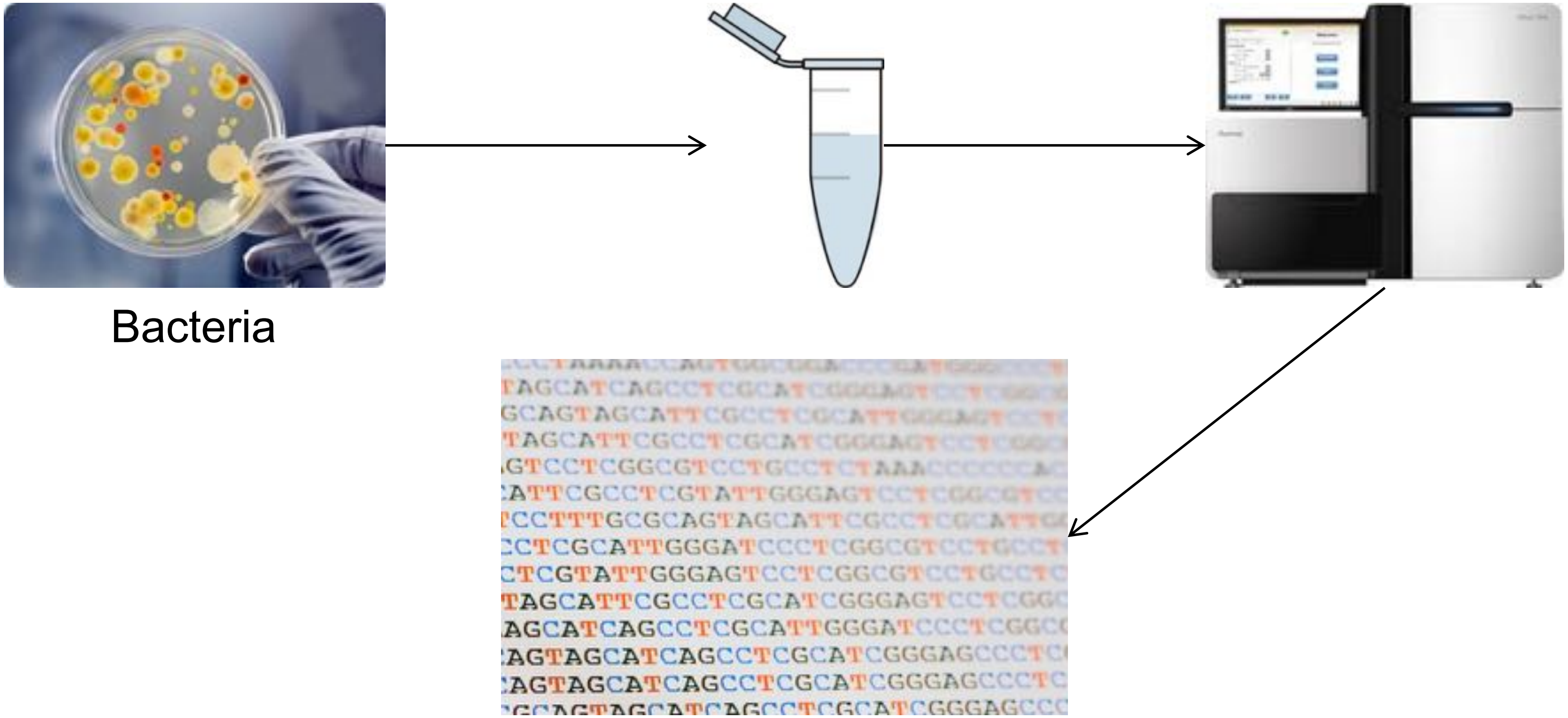
What is the mean sequence quality for the majority of the reads now?

The "Sequence length distribution" module shows a warning now. Have the read lengths changed? Why?

Are there still adapter sequences in our reads?

De novo genome assembly

In an ideal world:



DNA

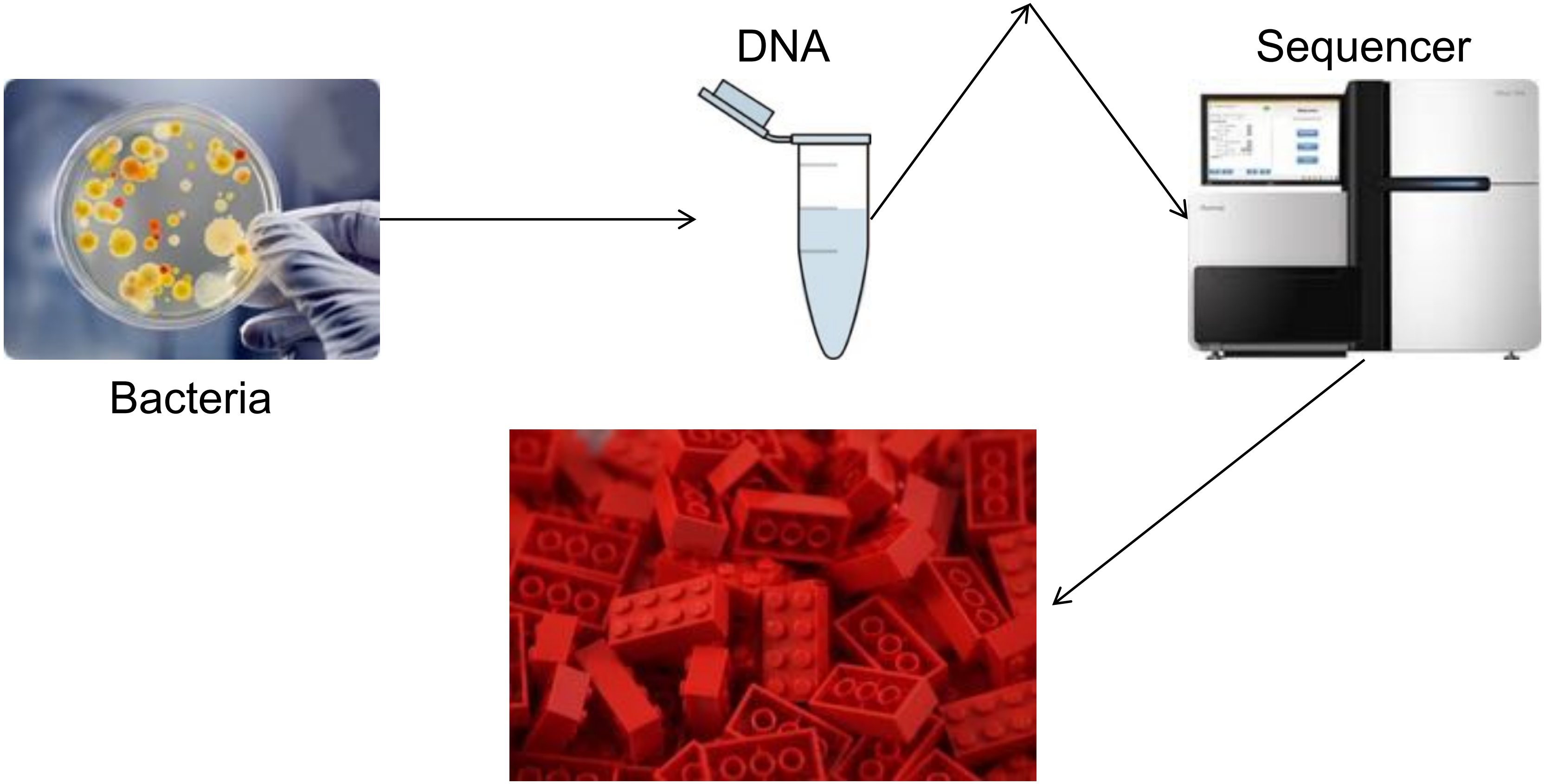
Sequencer

Bacteria

Genome sequence

De novo genome assembly

In reality…



Fragmentation

DNA

Sequencer

Bacteria

Fragments of a genome

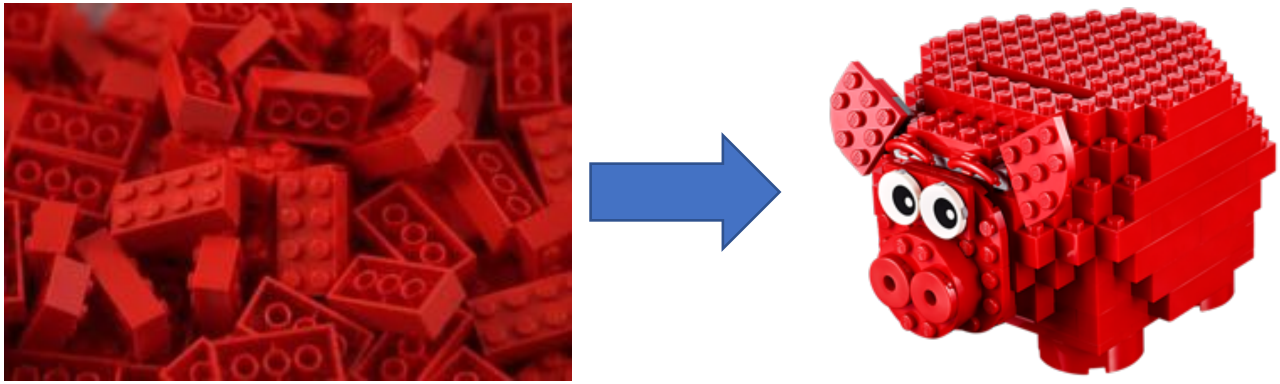
Genome assembly

Reconstruct the original genome from short sequence reads

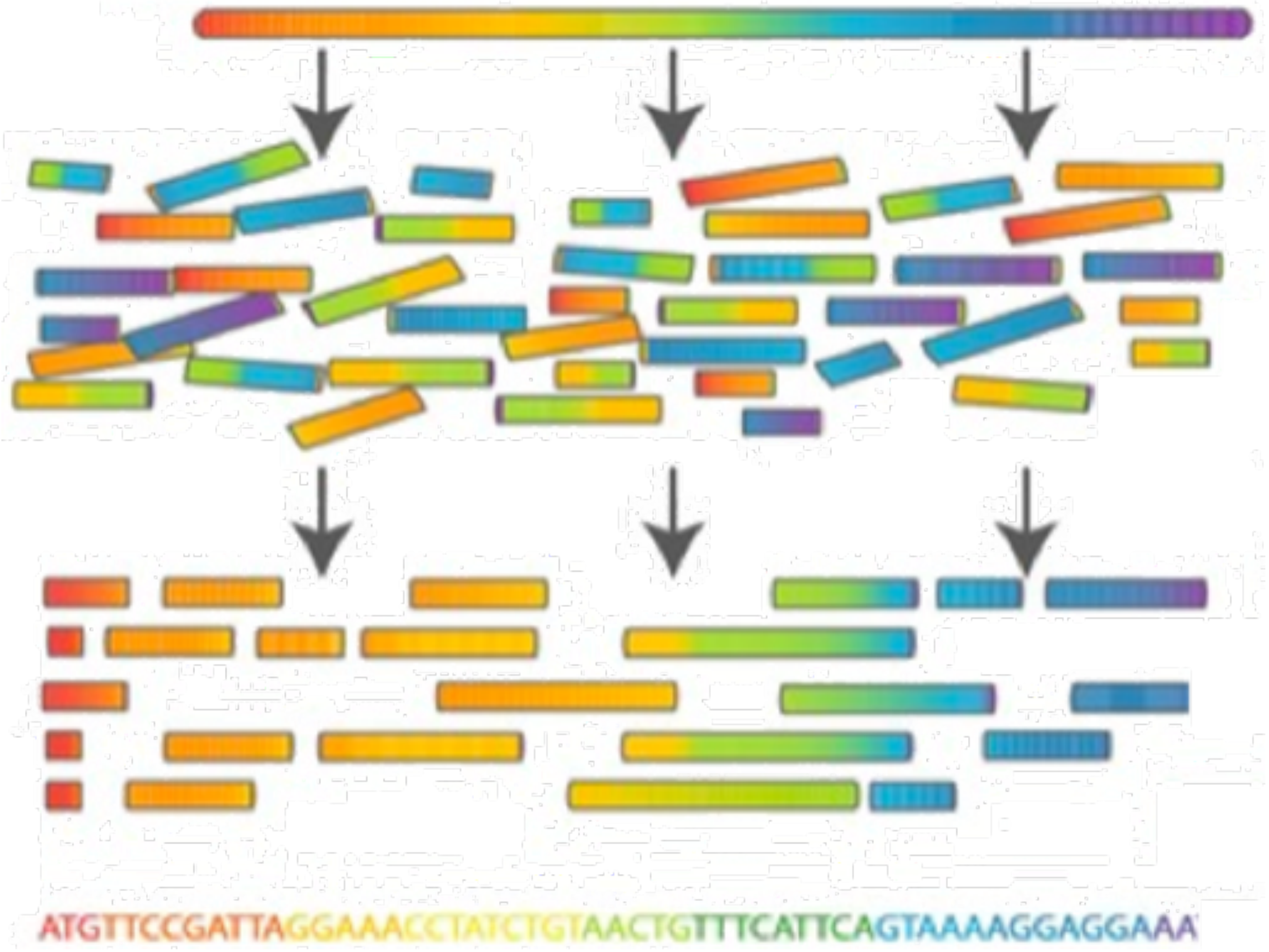
In an ideal world: sequences = one complete genome

In reality: sequences = multiple contigs

- Contiguous, unambiguous stretches of sequences



Genome assembly



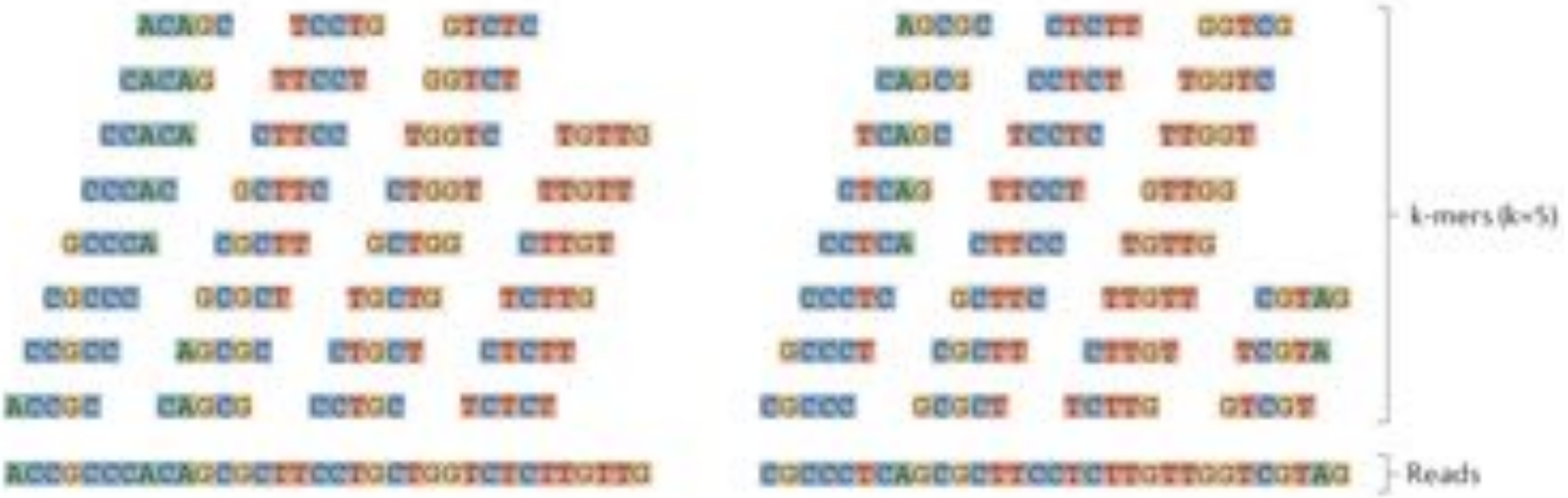
De Bruijn graph

In real life 10^6 sequences cannot be compared with each other

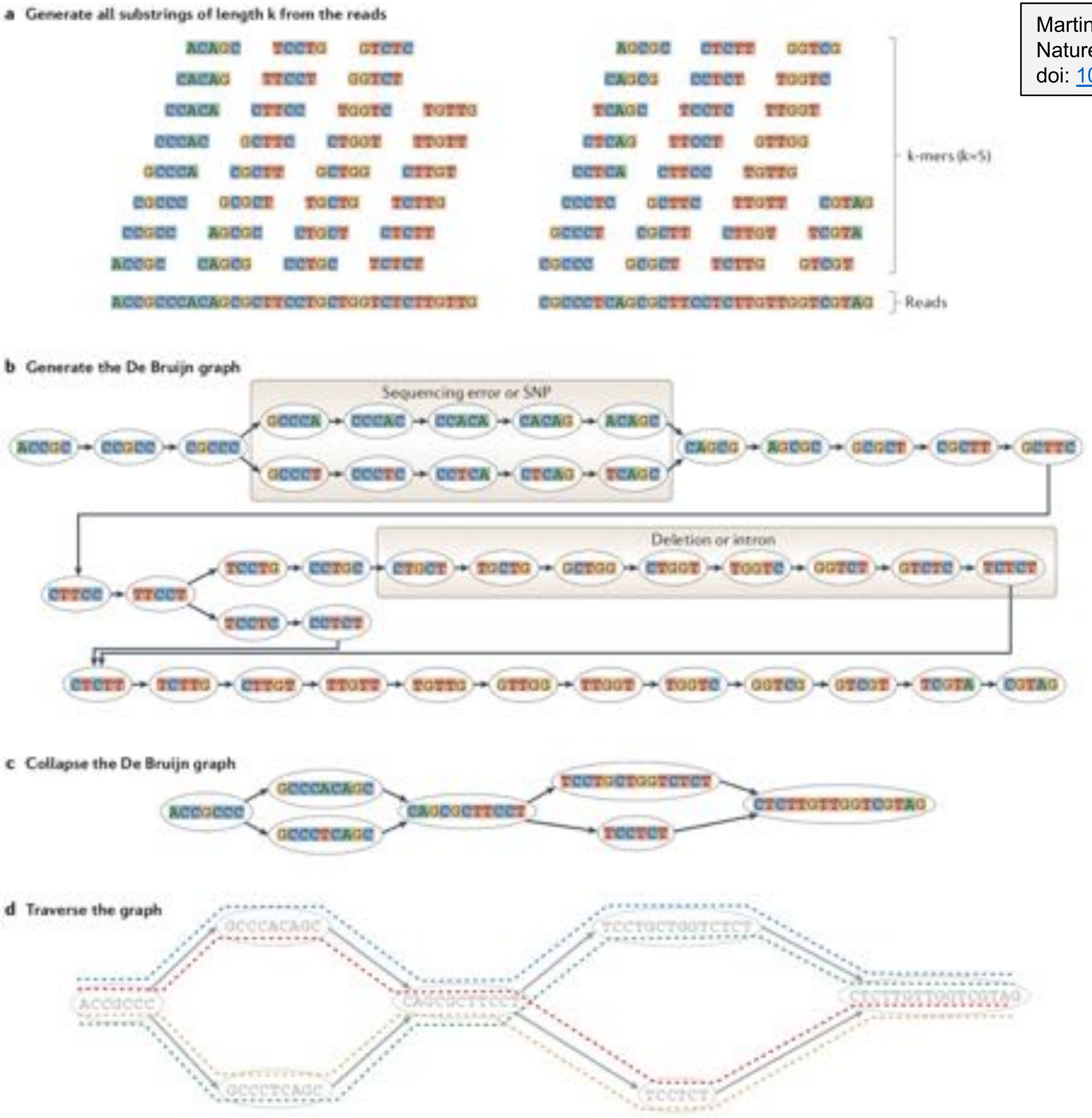
- 10^6 x 10^6 = 10^{12} comparisons

Sequences are reduced to k-mers

- Smaller subsets of length k



Martin & Wang 2011
Nature Reviews Genetics
doi: 10.1038/nrg3068



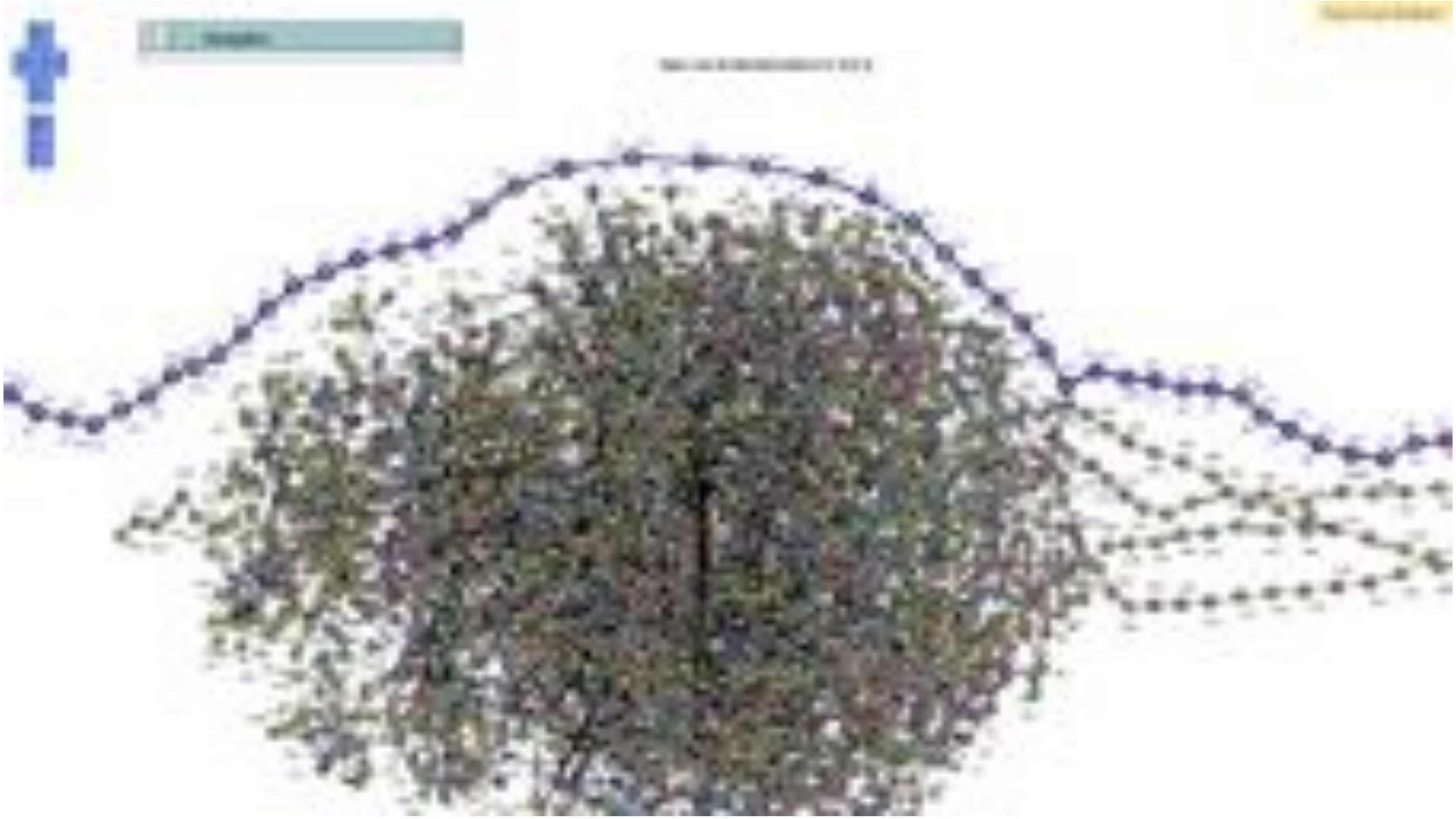
a Generate all substrings of length k from the reads

b Generate the De Bruijn graph

c Collapse the De Bruijn graph

d Traverse the graph

In real life with 10^6 sequences



What makes assembly tricky?

Many pieces (computationally-intensive)

Errors in sequence (which one is correct?)

Missing fragments

Repetitive fragments

Multiple copies (rRNA gene as an example)

Circular genome (no starting point)

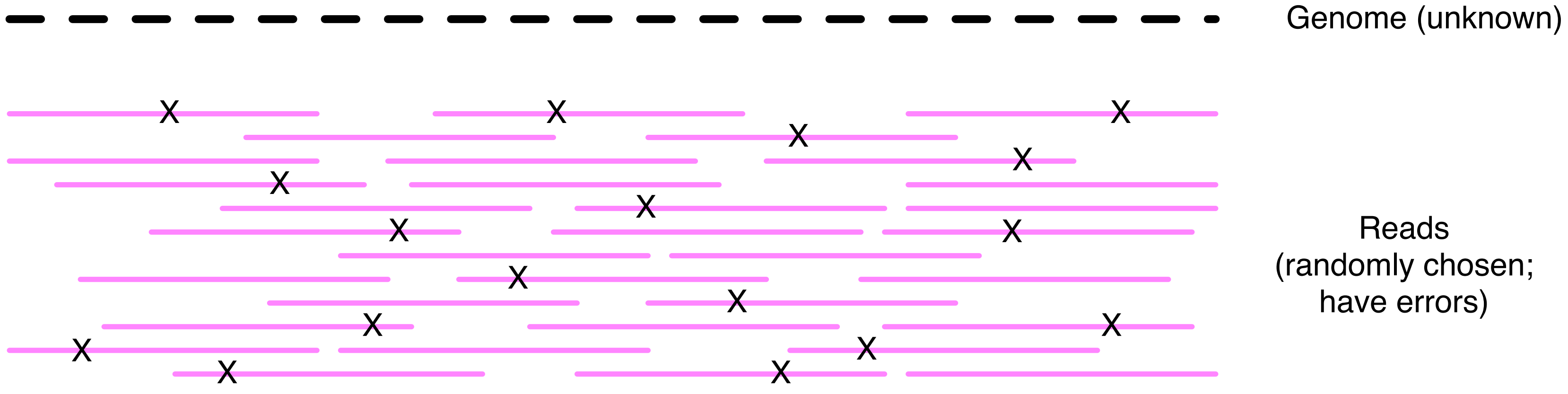
Choice of k-mer

- Too small = misassembly (anything can assemble)
- Too long = no assembly

Sequencing coverage

Coverage describes the number of time that each base of the genome is present in the reads

Assemblers expect equal and high enough coverage (>50x) to work optimally in genome assembly.



Estimated coverage for our *Vogesella* (?) strain

Vogesella genomes: ~3 – 4 Mb

Received data

- 112 530 read pairs
- 326 + 278 bp

Coverage

Number of bases sequenced/genome size

Bases = (112 530 x 326 bp) + (112 530 x 278 bp)

19 089 256 bp + 16 278 568 bp = 67 968 120 bp

Coverage = 35 367 824 bp / 3 000 000 bp = **22.7**

Assemblers

Bioinformatic tools that combine short sequencing reads into longer contigs

For our genome, we will use one assembler called SPAdes

- http://bioinf.spbau.ru/spades
- Uses De Bruijn graphs

We will use the batch job system in Puhti

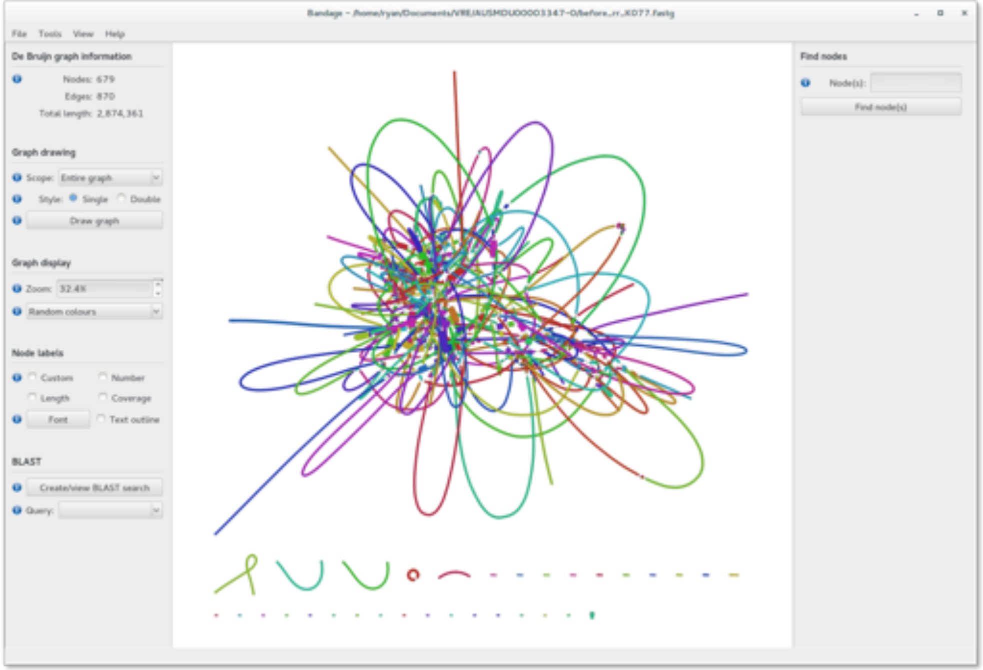
```
#!/bin/bash -l
#SBATCH --job-name spades
#SBATCH --output spades_out_%j.txt
#SBATCH --error spades_err_%j.txt
#SBATCH --time 1:00:00
#SBATCH --nodes 1
#SBATCH --ntasks-per-node 1
#SBATCH --cpus-per-task 4
#SBATCH --mem 5000
#SBATCH --account project_XXX

module load biokit

spades.py -1 Data/MMB-114_trimmed_1.fastq.gz \
          -2 Data/MMB-114_trimmed_2.fastq.gz \
          -o SPADES \
          -t 4 \
          --careful
```

SPAdes output

contigs.fasta:this contains the contigs generated by SPAdes

scaffolds.fasta: contigs generated by SPAdes after repeat resolution and scaffolding using paired-end information

assembly_graph.fastg: the assembly graph generated by SPAdes (can be viewed in Bandage)



Bandage - /home/ryan/Documents/VRE/AUSMDU00003347-0/before_rr_K077.fastg

De Bruijn graph information

Nodes: 679
Edges: 870
Total length: 2,874,361

Graph drawing

Scope: Entire graph
Style: ● Single ○ Double

Graph display

Zoom: 32.4X
Random colours

Node labels

○ Custom ○ Number
○ Length ○ Coverage
○ Text outline

BLAST

Query:

Find nodes

Node(s):

Additional notes about Puhti

Batch jobs

`sbatch`

`scancel`

`squeue`

`seff`

```
Job ID: 3750729

Cluster: puhti

User/Group: stelmach/stelmach

State: COMPLETED (exit code 0)

Nodes: 1

Cores per node: 4

CPU Utilized: 00:15:57

CPU Efficiency: 74.53% of 00:21:24 core-walltime

Job Wall-clock time: 00:05:21

Memory Utilized: 2.49 GB

Memory Efficiency: 51.06% of 4.88 GB

Job consumed 0.40 CSC billing units based on following used resources

CPU BU: 0.36

Mem BU: 0.04
```

Assembly quality

What is a good assembly?

Good coverage throughout the contigs

Correct size (for bacteria, not 400 kb or 14 Mb)

As few contigs as possible

Similar GC across contigs

QUAST

http://bioinf.spbau.ru/quast

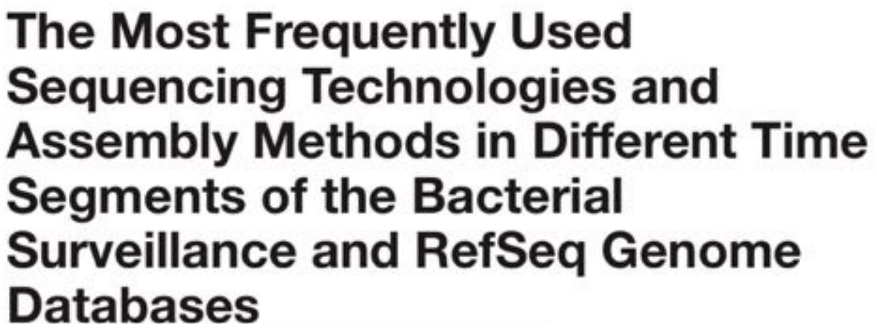
Tool to evaluate the quality of assemblies

After running QUAST, move to your computer with FileZilla:

- report.html
- report.pdf
- icarus.html
- icarus_viewers (folder)

Let’s assemble our genome

But first read an article:



The Most Frequently Used Sequencing Technologies and Assembly Methods in Different Time Segments of the Bacterial Surveillance and RefSeq Genome Databases

And then we go on to the assembly

https://github.com/karkman/MMB-114_Genomics

(**Day 3:** Genome assembly)