Day 4: Genome Assembly

MMB-114

Schedule

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

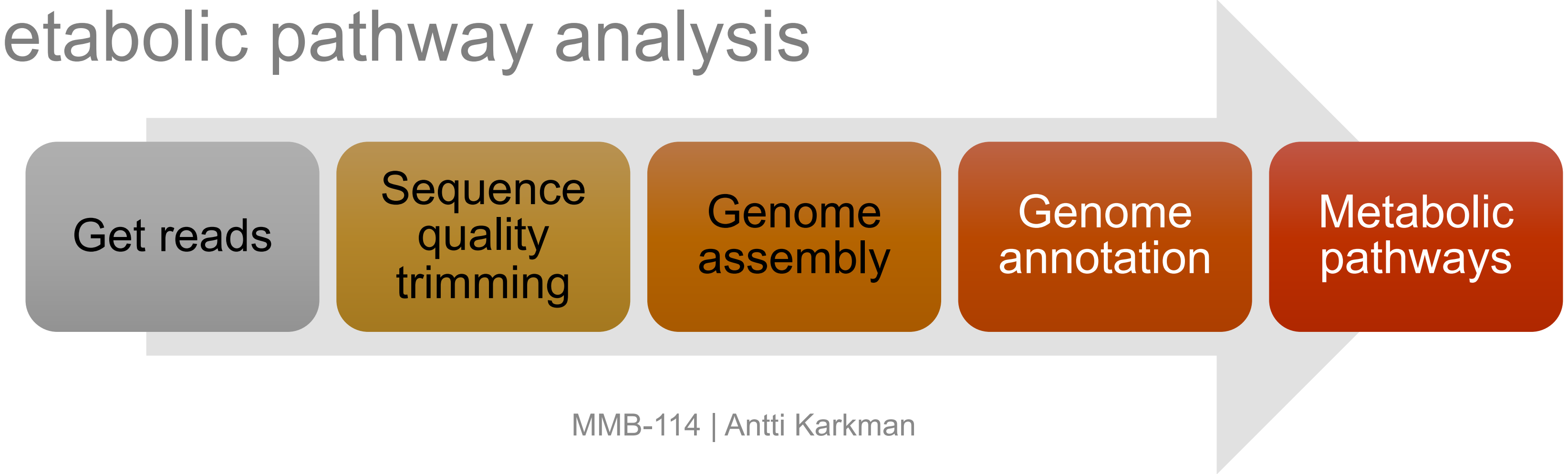
Day 2: Handling of Nanopore/Illumina data

Day 3: Check-up

Day 4: Genome assembly

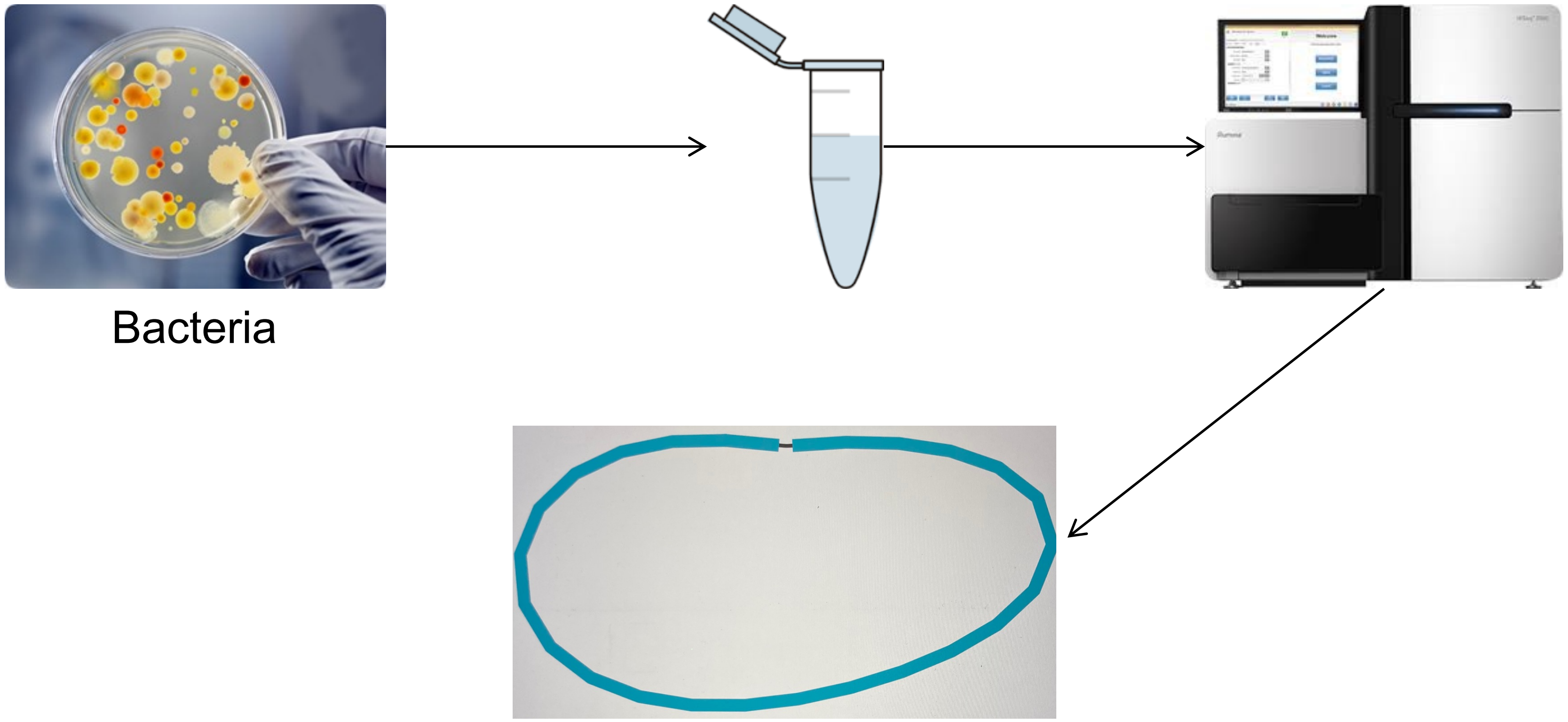
Day 5: Genome annotation

Day 6: Metabolic pathway analysis



De novo genome assembly

In an ideal world:



DNA

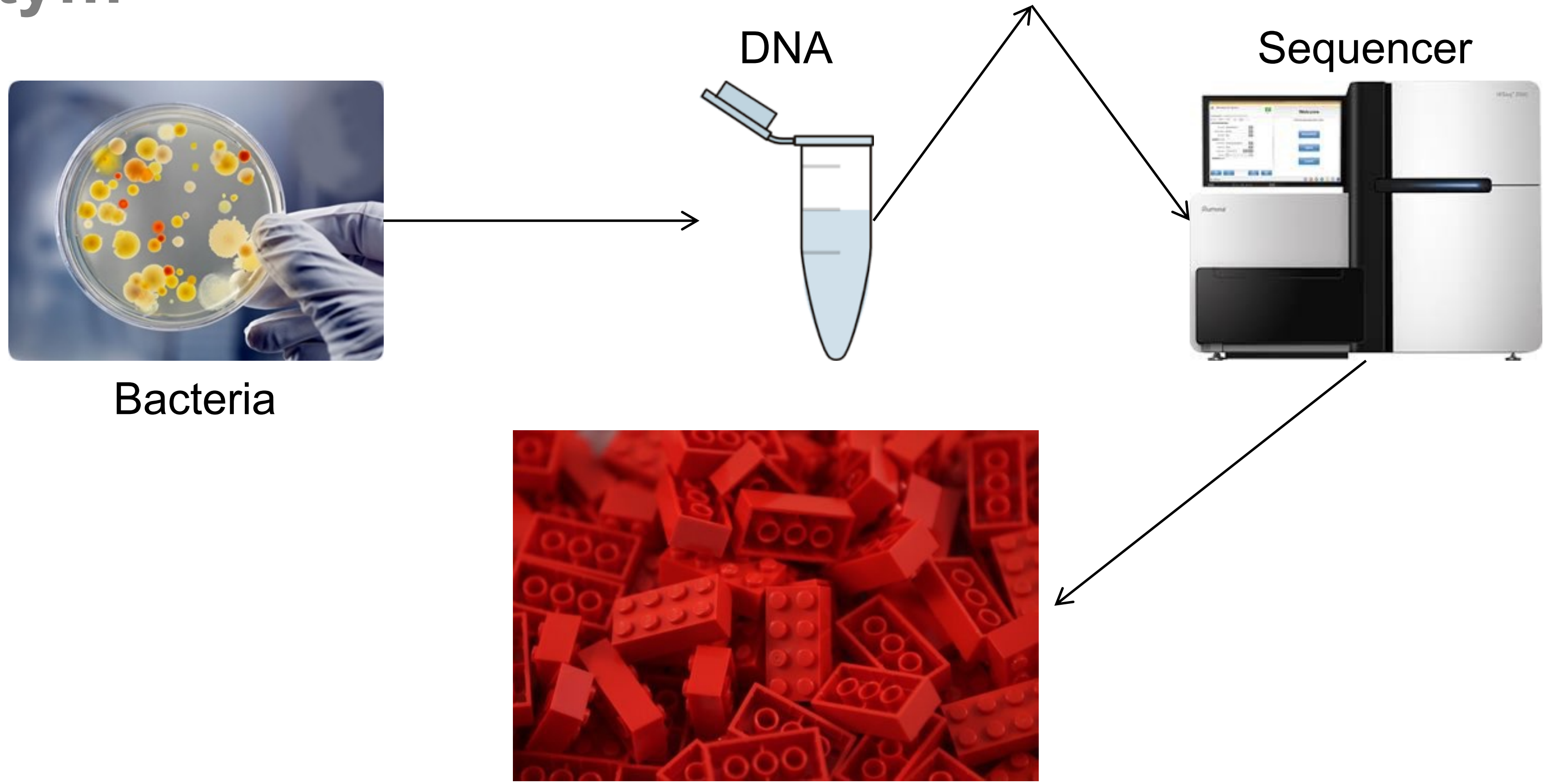
Sequencer

Bacteria

Genome sequence

De novo genome assembly

In reality…



DNA

Fragmentation

Sequencer

Bacteria

Fragments of a genome

Genome assembly

- Reconstruct the original genome from long/short sequence reads

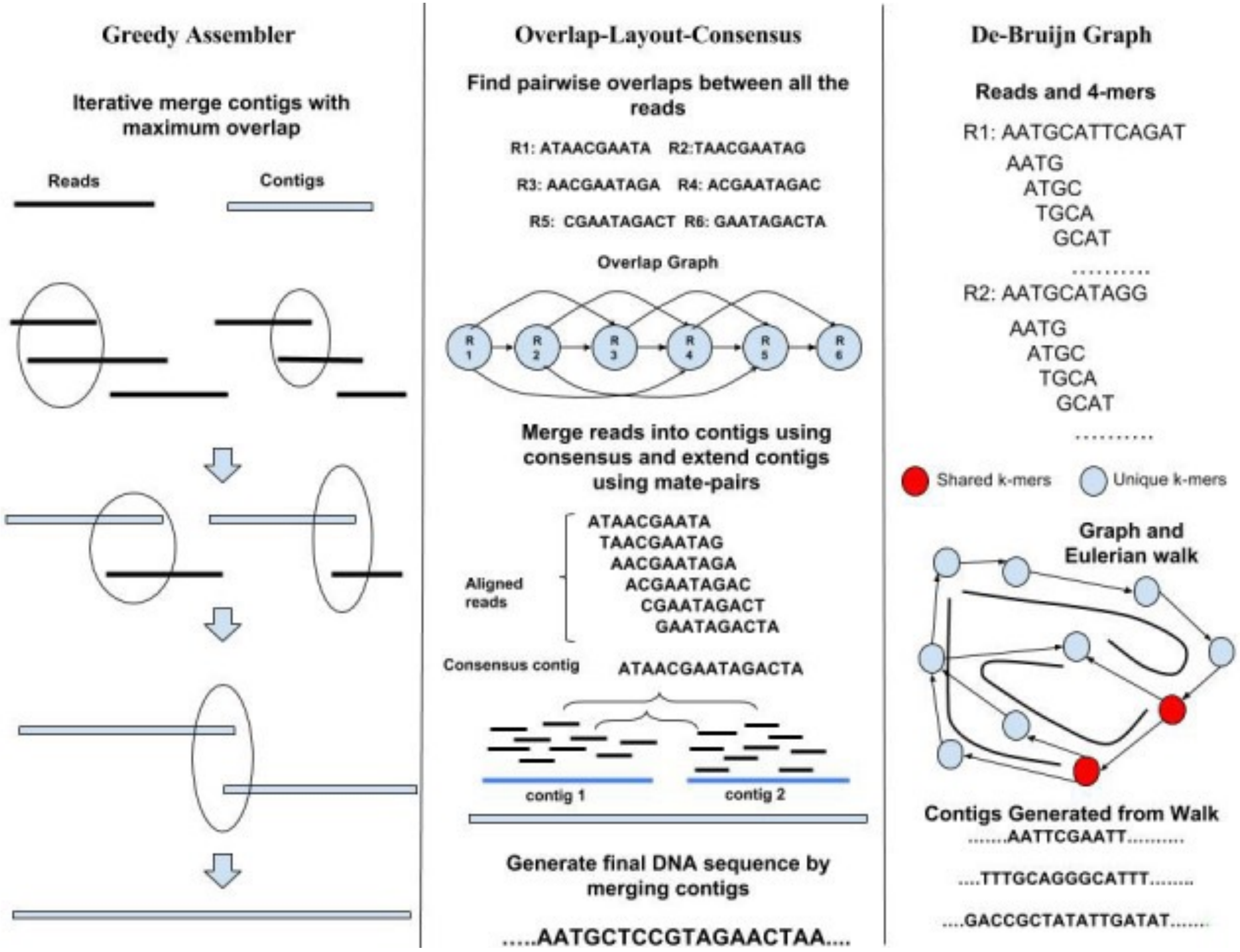
Short reads:

- **In an ideal world:** sequences = one complete genome
- **In reality:** sequences = multiple contigs
  - Contiguous, unambiguous stretches of sequences

Long reads:

- Sometimes we reach the ideal world

Assembly strategies



Greedy Assembler
Iterative merge contigs with maximum overlap
Reads
Contigs
Overlap-Layout-Consensus
Find pairwise overlaps between all the reads
R1: ATAACGAATA R2:TAACGAATAG
R3: AACGAATAGA R4: ACGAATAGAC
R5: CGAATAGACT R6: GAATAGACTA
Overlap Graph
R1
R2
R3
R4
R5
R6
Merge reads into contigs using consensus and extend contigs using mate-pairs
Aligned reads
ATAACGAATA
TAACGAATAG
AACGAATAGA
ACGAATAGAC
CGAATAGACT
GAATAGACTA
Consensus contig
ATAACGAATAGACTA
contig 1
contig 2
Generate final DNA sequence by merging contigs
.....AATGCTCCGTAGAACTAA....
De-Bruijn Graph
Reads and 4-mers
R1: AATGCATTCAGAT
AATG
ATGC
TGCA
GCAT
....
R2: AATGCATAGG
AATG
ATGC
TGCA
GCAT
....
Shared k-mers
Unique k-mers
Graph and Eulerian walk
Contigs Generated from Walk
.......AATTCGAATT..........
....TTTGCAGGGCATTT........
....GACCGCTATATTGATAT......

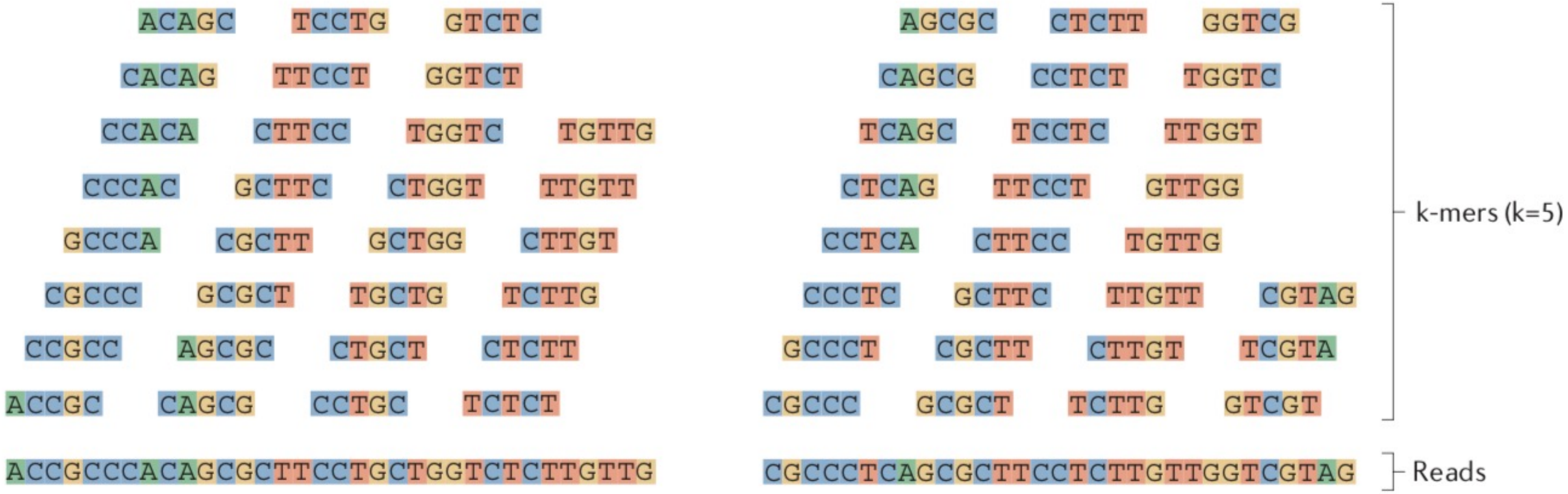
https://doi.org/10.1093/bib/bbz020

Assemblers

- Bioinformatic tools that combine short sequencing reads into longer contigs
- **Spades:** kmer-based assembler for short reads (accepts also long reads). De Bruijn graphs.
- **Flye:** uses repeat graphs. Can tolerate the higher noise of single-molecule sequencing reads.

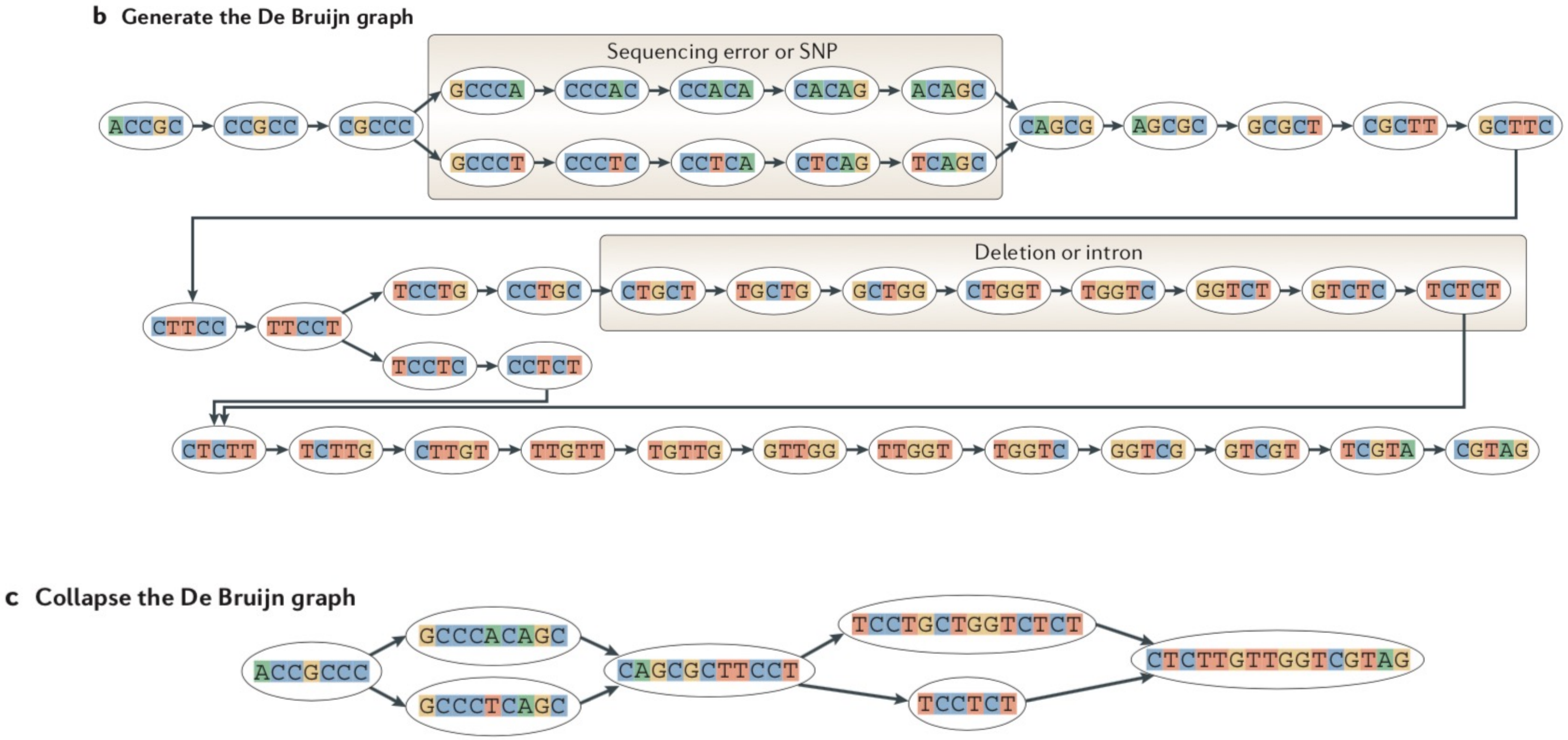
De Bruijn graph

- 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



https://doi.org/10.1038/nrg3068

De Bruijn graph

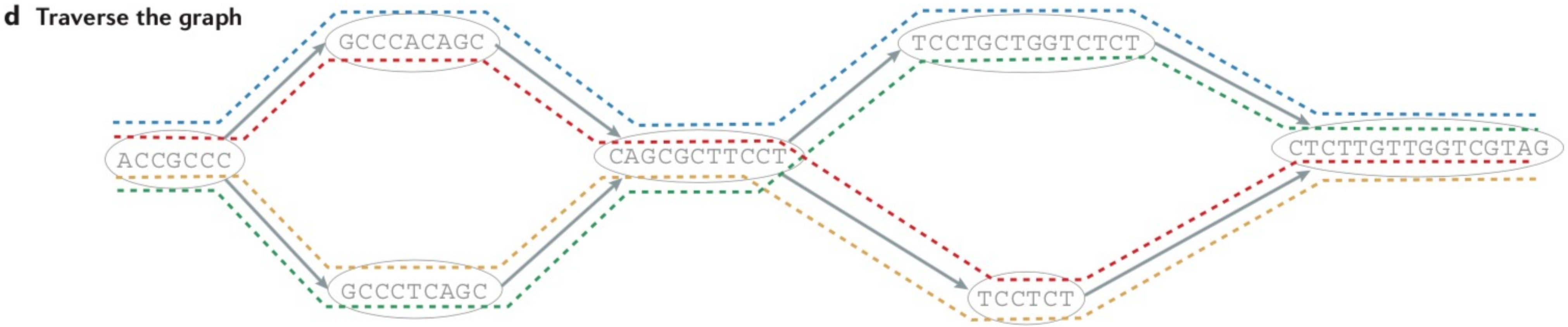


b Generate the De Bruijn graph

c Collapse the De Bruijn graph

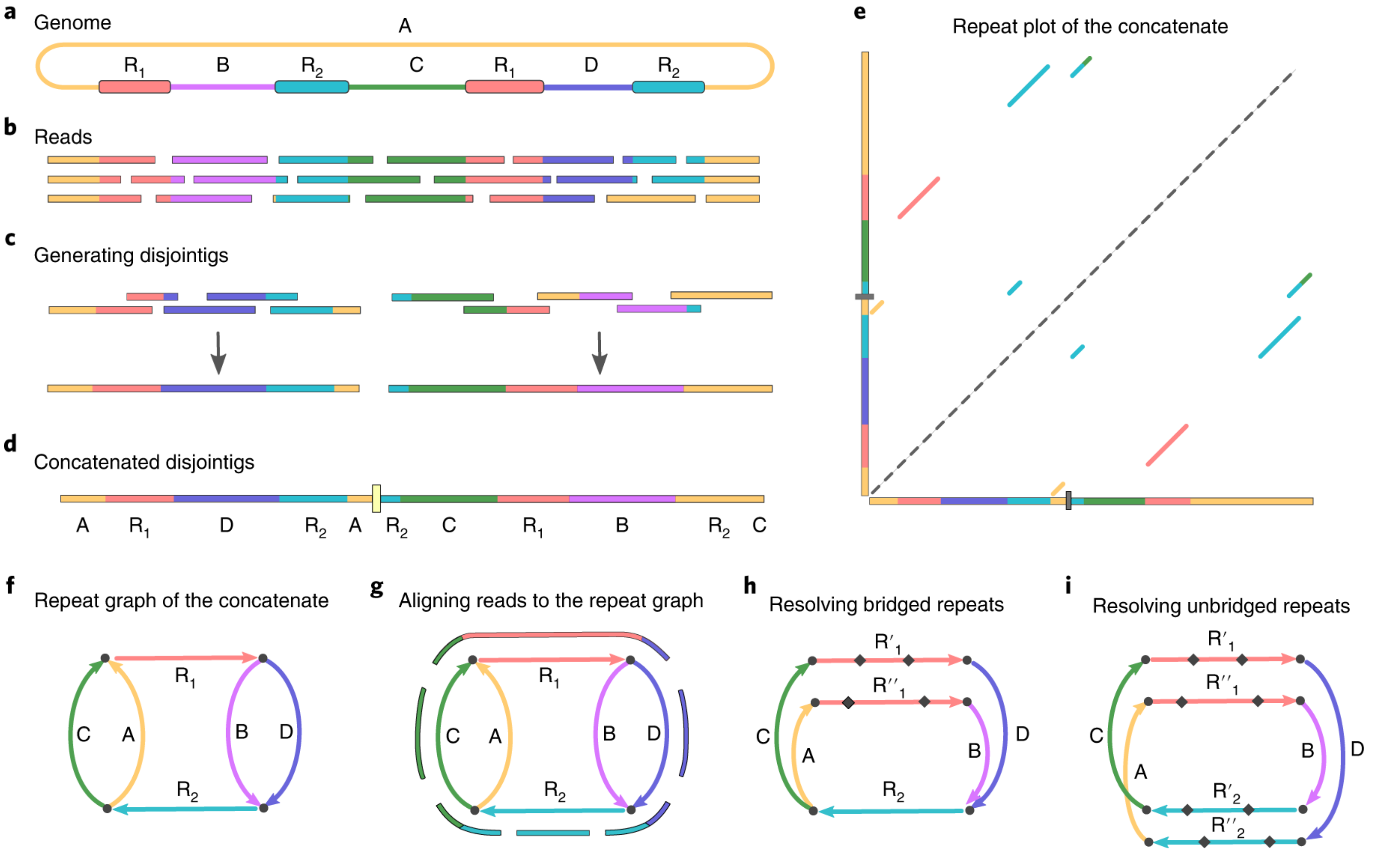
https://doi.org/10.1038/nrg3068

De Bruijn graph



https://doi.org/10.1038/nrg3068

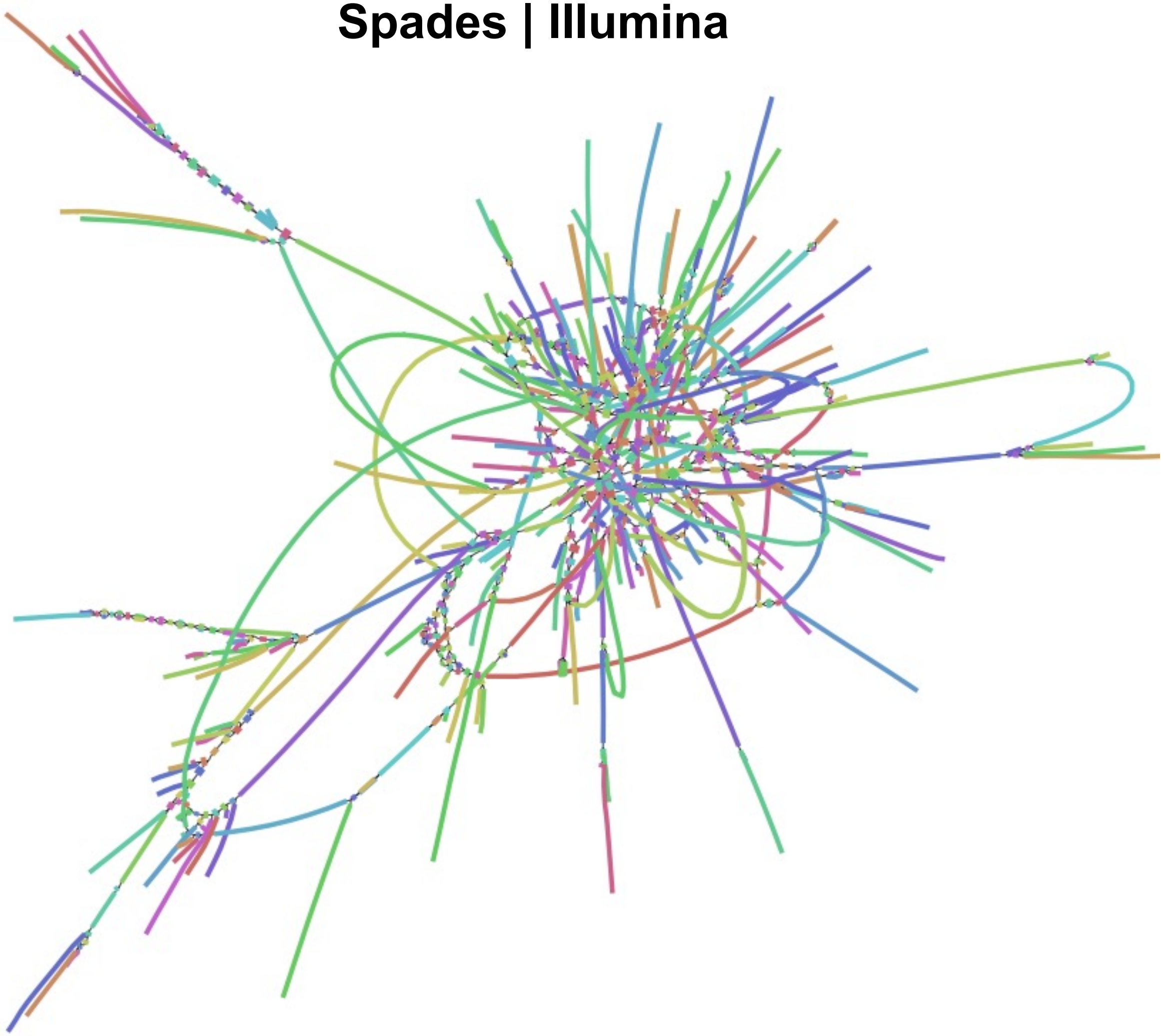
Repeat graphs


a
Genome
A
R1
B
R2
C
R1
D
R2
b
Reads
c
Generating disjointigs
d
Concatenated disjointigs
A
R1
D
R2
A
R2
C
R1
B
R2
C
e
Repeat plot of the concatenate
f
Repeat graph of the concatenate
R1
C
A
B
D
R2
g
Aligning reads to the repeat graph
R1
C
A
B
D
R2
h
Resolving bridged repeats
R′1
R′′1
C
A
B
D
R2
i
Resolving unbridged repeats
R′1
R′′1
C
A
B
D
R′2
R′′2

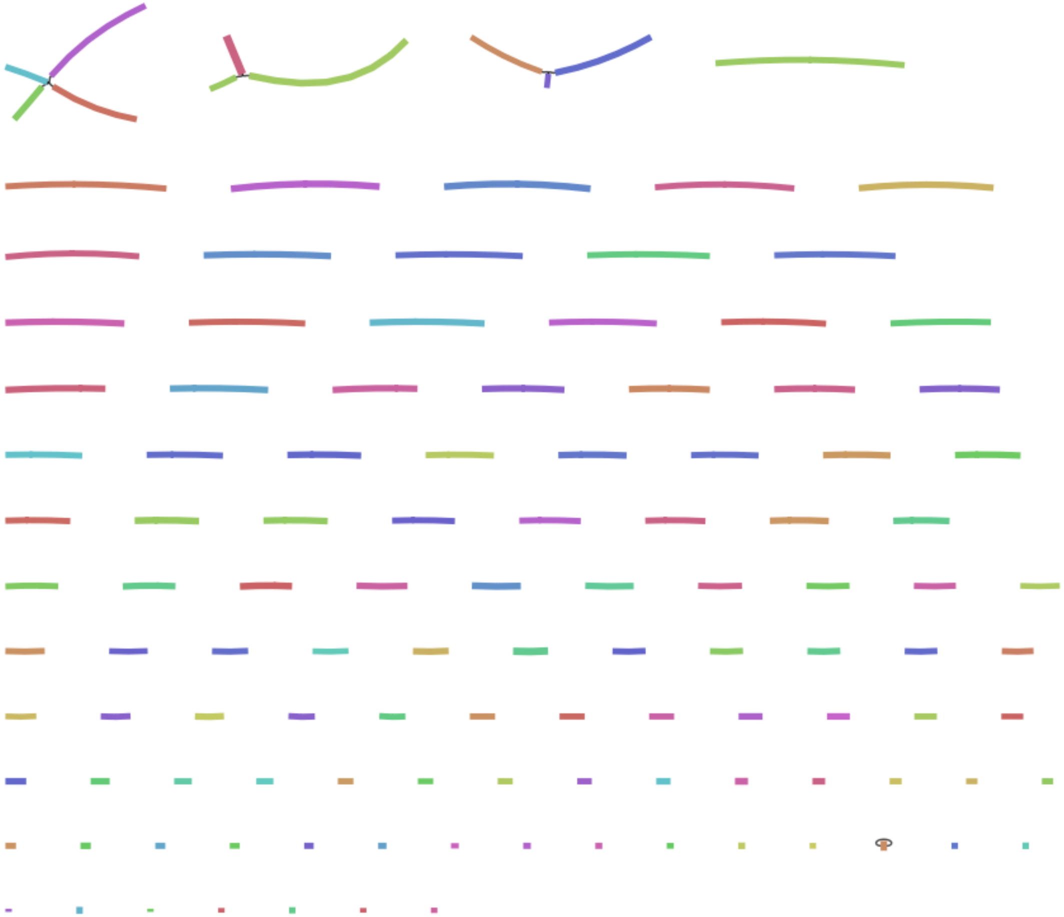
https://doi.org/10.1038/s41587-019-0072-8

Real world assemblies

Spades | Illumina



Flye | Nanopore

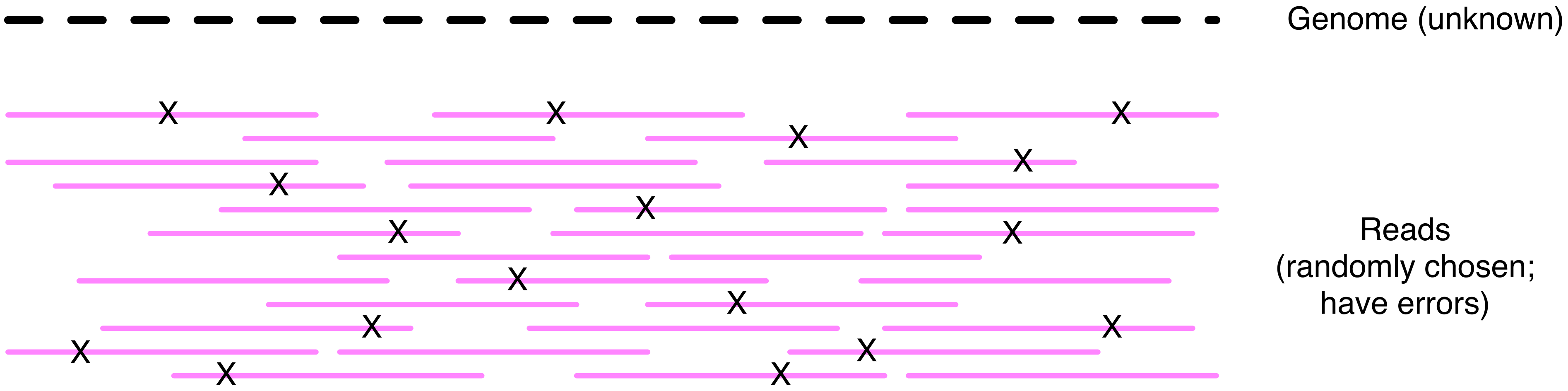


What makes assembly tricky?

- 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 average coverage of your genome

- Your genome size: **X bp**
- Received sequence data: **Y bp**

- Coverage = Y / X

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

Additional notes about Puhti

Batch jobs:

```
sbatch
scancel
squeue
seff
```

batch_job.sh

```
#!/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 spades

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

```
sbatch batch_job.sh
```

Assembly outputs

Flye

- `assembly.fasta` – assembled contigs
- `assembly_graph.gfa` – assembly graph
- `assembly_info.txt` – information about each contig

Spades

- `contigs.fasta` – assembled contigs
- `assembly_graph.fastg –` assembly graph

Let’s assemble your genomes

Go to Github and follow the instructions:

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

(**Day 3:** Genome assembly)