
DTU

DTU Health Technology
Bioinformatics

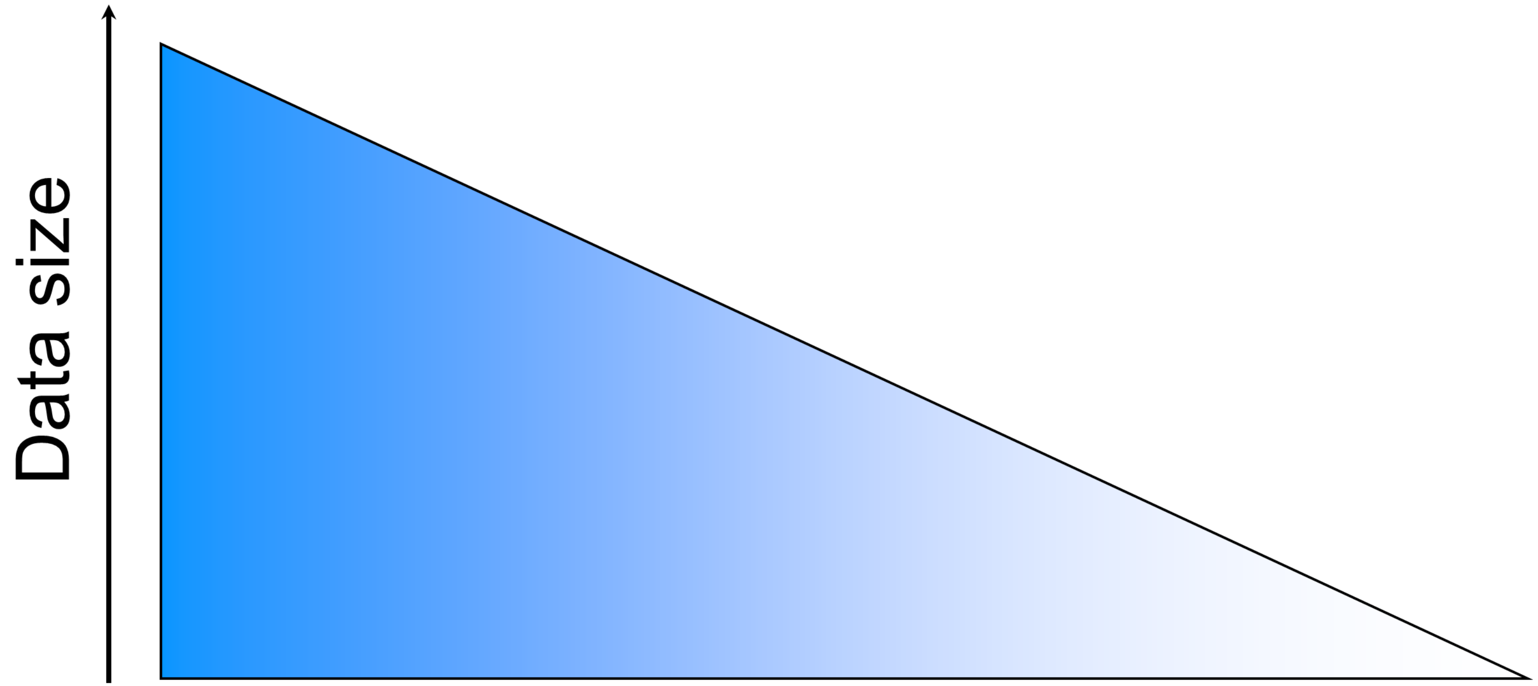
Metagenomic *de novo* assembly

Gisle Vestergaard
Associate Professor
Section of Bioinformatics
Technical University of Denmark
gisves@dtu.dk

Menu

- What do I mean by assembly?
- Assembly approaches
- Assembly graphs
- Graph postprocessing filtering
- The woes of repetition
- Benchmarking your assembly

Generalized NGS analysis



Question

Raw reads

Pre-processing

Assembly: Alignment / *de novo*

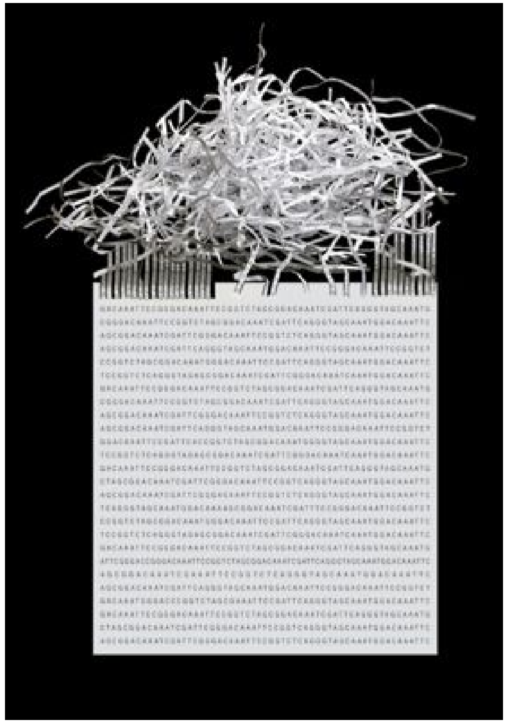
Application specific: Variant calling, count matrix, ...

Compare samples / methods

Answer?

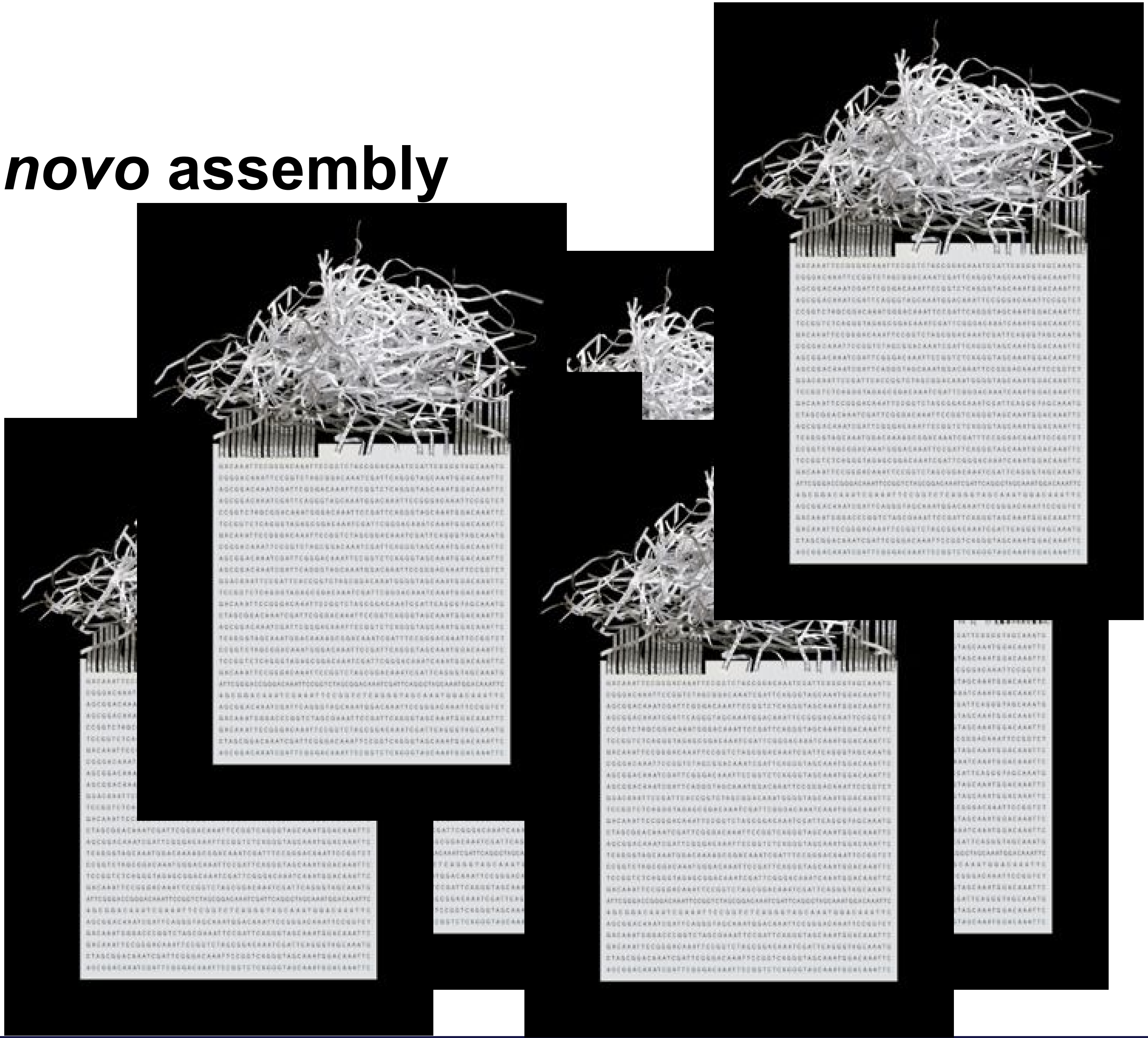
What is *de novo* assembly?

- Technological limitions means we only get fragments of DNA molecules
- *de novo* assembly means merging these DNA fragments together so they form a previously unknown sequence
- Imagine a book written in a language with 4 letters shredded into a million pieces



Metagenomic de novo assembly

- Thousands of books
- Some in many copies others not
- Written with four letters
- All shredded to small pieces



Which approaches?

- Greedy (“Simple” approach)
- Overlap Layout Consensus
- de Bruijn graphs
- Repeat graphs

Simple approach - Greedy

- Principle:
    1. Pairwise alignment of all reads
    2. Identify fragments that have largest overlap
    3. Merge these
    4. Repeat until all overlaps are used
- Can only resolve repeats smaller than read length
- High computational cost with increasing no. reads

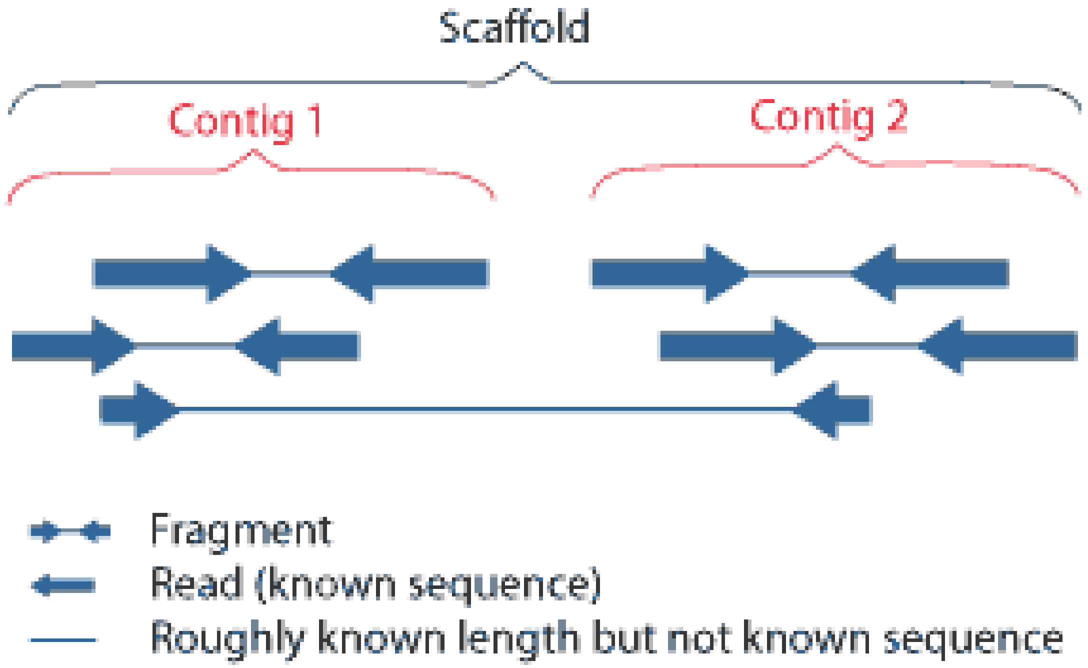
Reads > Contigs > Scaffolds

- **Overlap Layout Consensus** and **de Bruijn** use a similar general approach.

1. Try to correct sequence errors in reads with high coverage
2. Assemble reads to contiguous sequence fragments "contigs"
3. Identify repeat contigs
4. Combine and order contigs to "scaffolds", with gaps representing regions of uncertainty

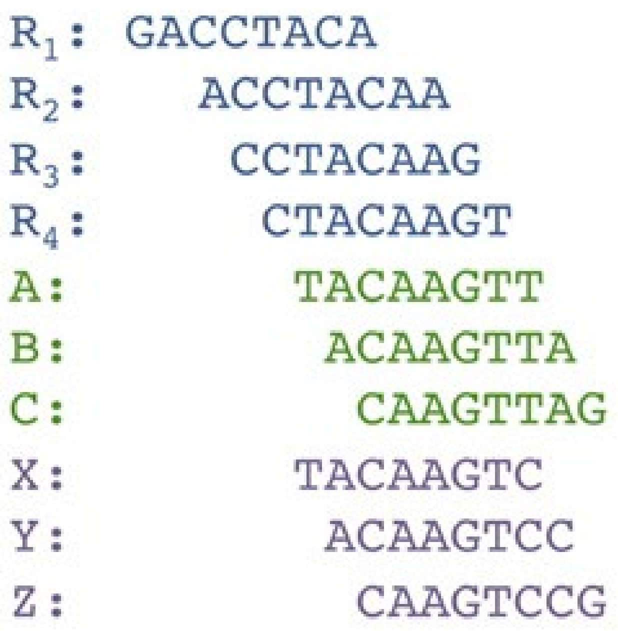
Glueing the reads together

- Reads are assembled into contigs
- Contigs can be bridged into scaffolds by
  - – Mapping against reference genome
  - – Low quality sequence
  - – Paired-end read information
  - – Other methods such as Hi-C

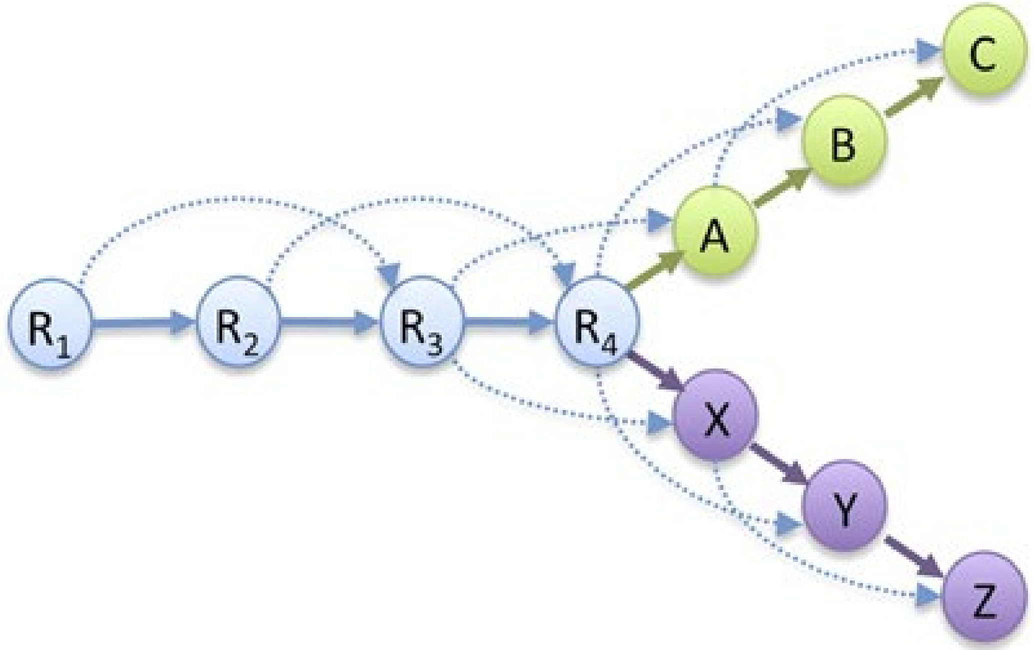


Overlap-Layout-Consensus

- Create overlap graph by all-vs-all alignment (Overlap)
- Build graph where each node is a read, edges are overlaps between reads (Layout)
- Create consensus sequence



```
R1: GACCTACA
R2:  ACCTACAA
R3:   CCTACAAG
R4:    CTACAAGT
A:      TACAAGTT
B:       ACAAGTTA
C:        CAAGTTAG
X:      TACAAGTC
Y:       ACAAGTCC
Z:        CAAGTCCG
```



Schatz et al., Genome Res, 2010

Overlap-Layout-Consensus

- Not good with many short reads -> lots of alignment!

- Good for large read lengths:
  - PacBio & Oxford Nanopore
  - Example assemblers: Canu, Celera, Newbler

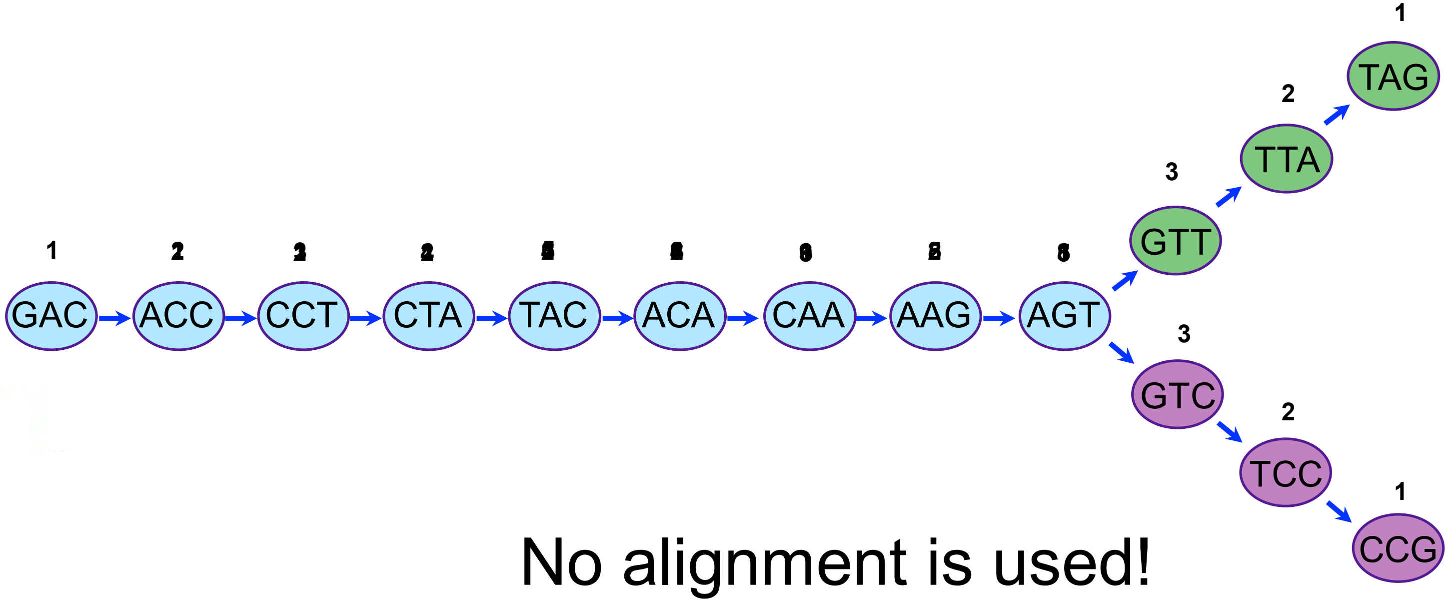
de Bruijn graph

- Directed graph of overlapping items (here DNA sequences)
- Instead of comparing reads, decompose reads into *k*-mers
  - Graph is created by mapping the *k*-mers to the graph
  - Each *k*-mer only exists once in the graph
  - No all-vs-all alignment

How is the graph constructed?

- Same 10 reads, extract *k*-mers from reads and map onto graph, *k* = 3:

R$_1$: GACCTACA



No alignment is used!

Different assemblers uses different modifications of the de Bruijn graphs

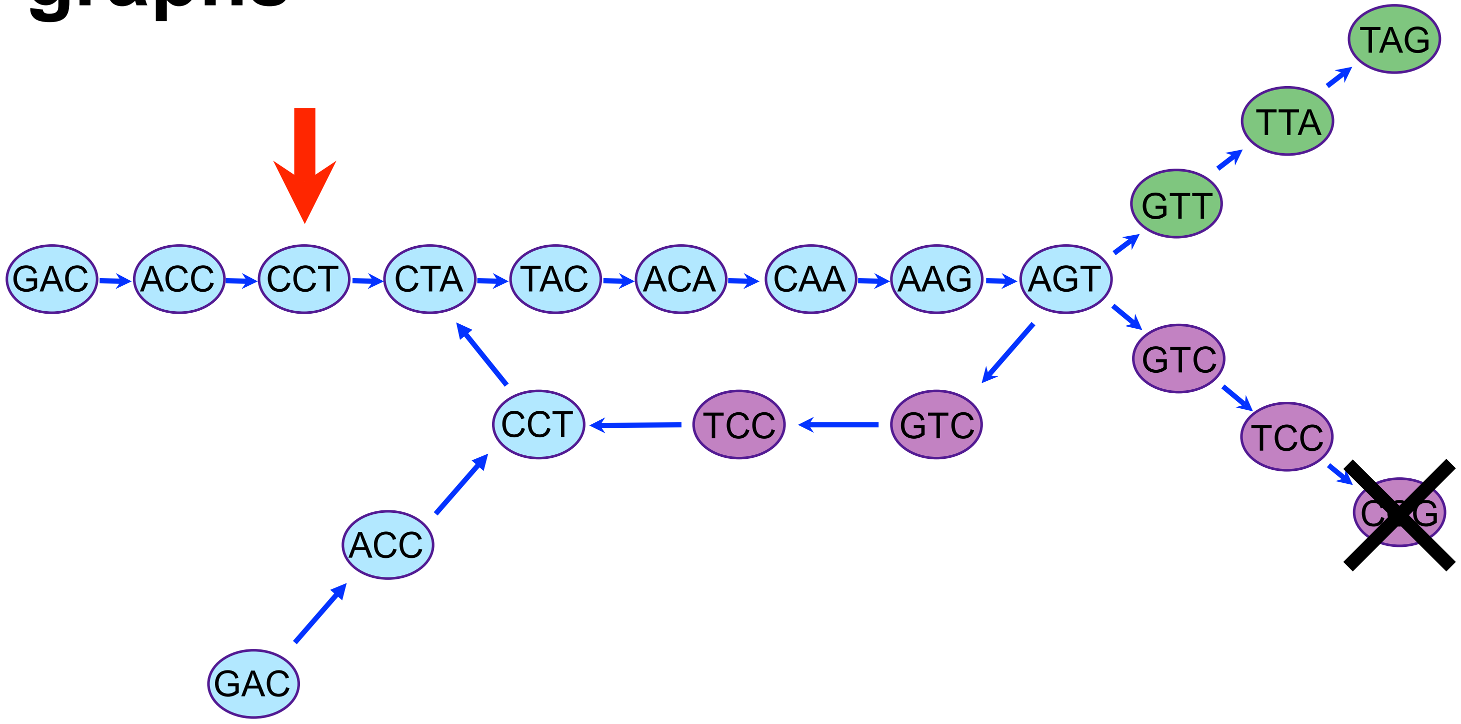
Drawbacks ...

- Lots of RAM required (**1-1000 GB !**)
- Optimal *k* can not be identified *a priori*, must be experimentally tested for each dataset
- small *k*: very complex graph, large *k*: limited overlap in low coverage areas
- Iterative approach to find best assembly

Complicated graphs

R_1: GACCTACA
R_2: ACCTACAA
R_3: CCTACAAG
R_4: CTACAAGT
A: TACAAGTT
B: ACAAGTTA
C: CAAGTTAG
X: TACAAGTC
Y: ACAAGTCC
Z: CAAGTCCT

G to T

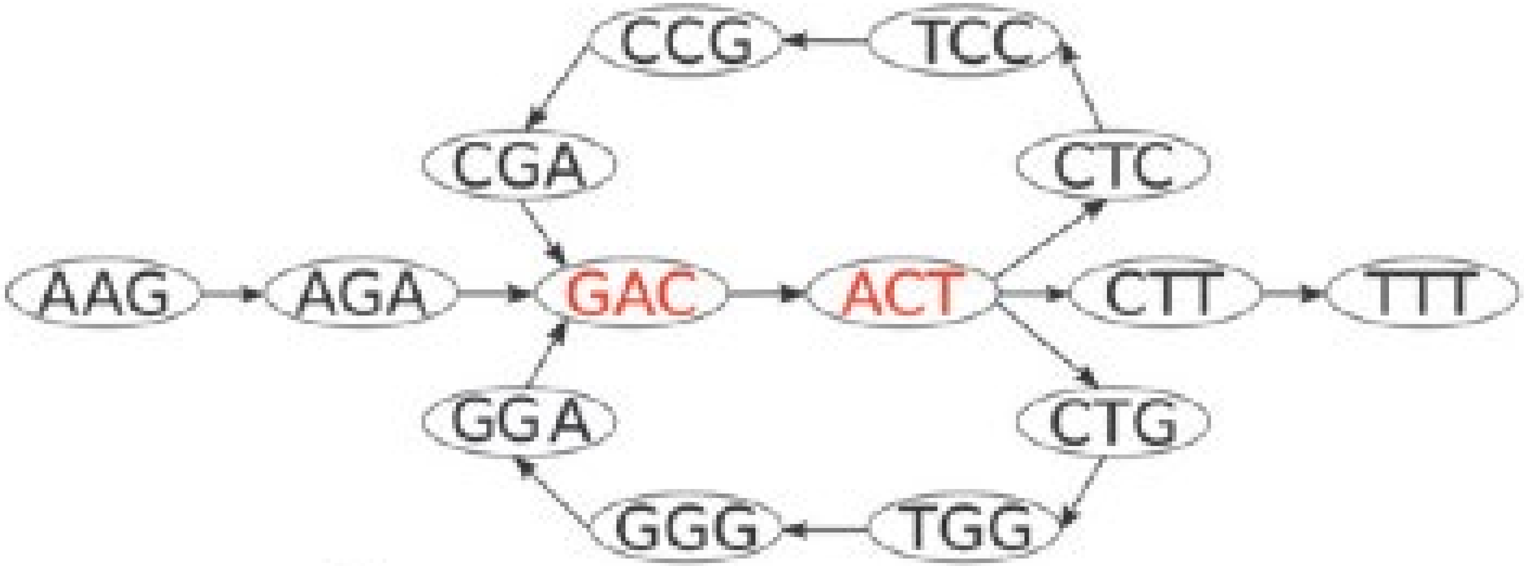


Large genomes with many repeats/errors creates very large graphs

Create the *de* Bruijn graph of this genome using *k*=3

AAGACTCCGACTGGGACTTT

AAGACTCCGACTGGGACTTT

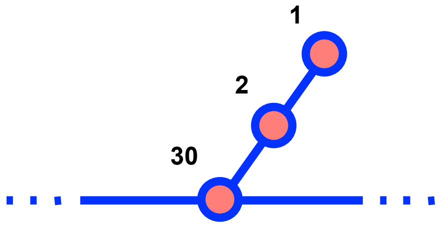


A de Bruijn graph of a sequence

After building: Simplify

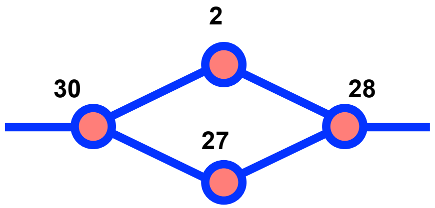
Clip tips

(seq err, end)

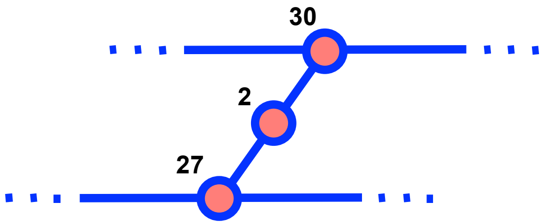


Pinch bubbles

(seq err, middle, SNP)

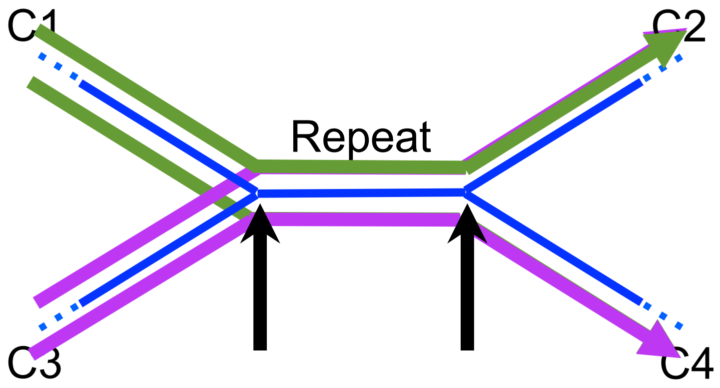


Remove low cov. links

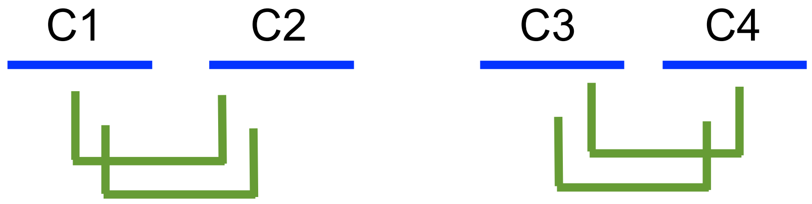


Create contigs and scaffolds

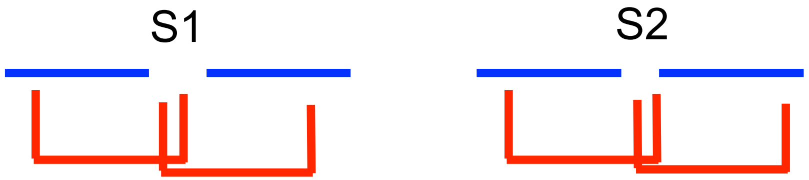
Cut graph at repeat boundaries to create contigs



Use paired end information to resolve repeats and combine to scaffolds



Fill potential gaps using PE reads



The assembly is done

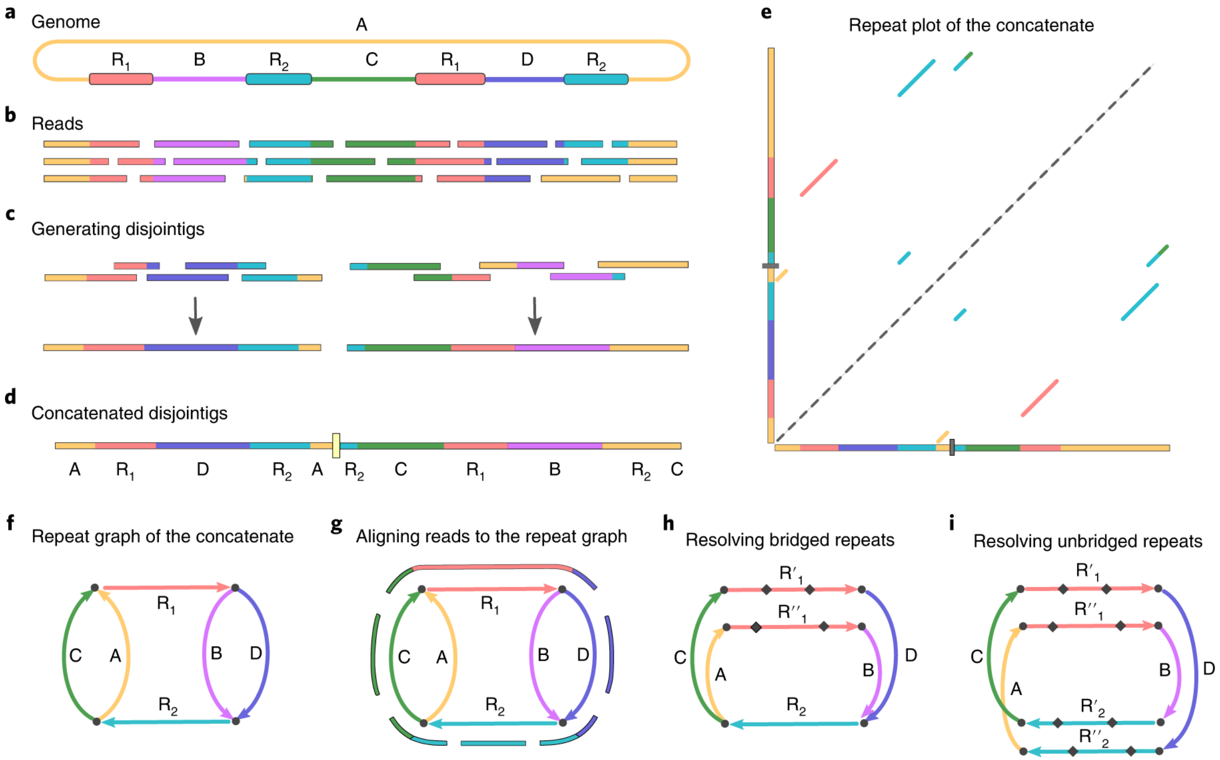
Iterate parameters

- Re-run with different *k*-sizes, find optimum
- Run with multiple k-mers at the same time! (eg. SPAdes)
- Compare assembly statistics such as, assembly length, N50, no. contigs

- Assembly refinement
  - Break contigs not supported by PE/MP reads
  - Analyze assembly using REAPR or QUAST

Repeats graphs and long reads

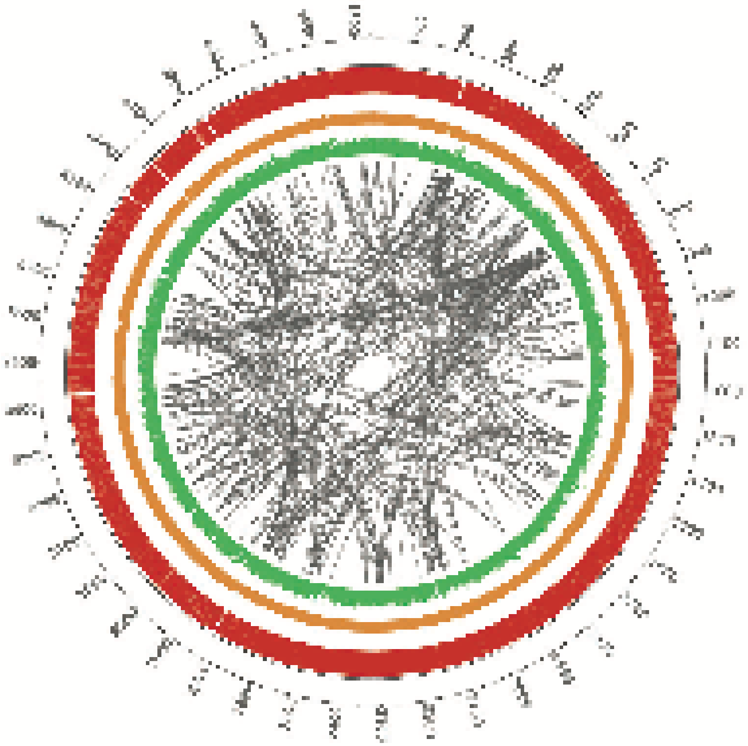
- Long-read assembler such as Flye
- Constructs an accurate repeat graph from these error-riddled disjointigs
- Extend resolved disjointigs using overlap strategy



Successful *de novo* assembly

- Success is a factor of:
  - Genome size, **genomic repeats(!)**, ploidy

  - High coverage, long read lengths, PE/MP libraries



Repeats in *E. coli*

Improving *de novo* assemblies

- Paired end & Mate pair for long range continuity
- Hybrid approaches (combine Illumina with PacBio/Oxford Nanopore)

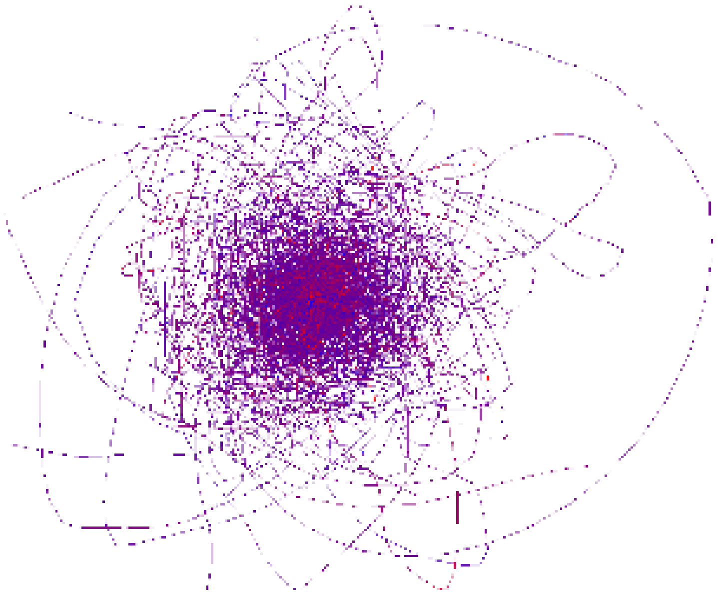
- Synthetic long reads: Illumina Synthetic Reads (Moleculo) or 10X Genomics
- Hi-C contact maps

Two bacterial genomes *de Bruijn* graphs

Few repeats

“more” repeats

Flicek & Birney, Nat.Methods 2009

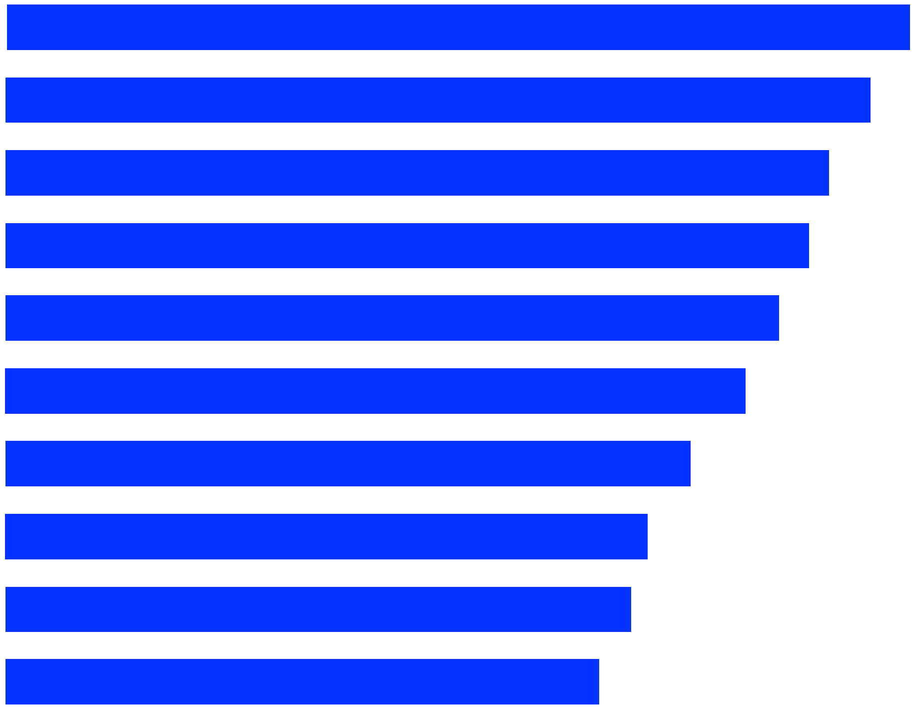


Zerbino, 2009

N50: Assembly quality

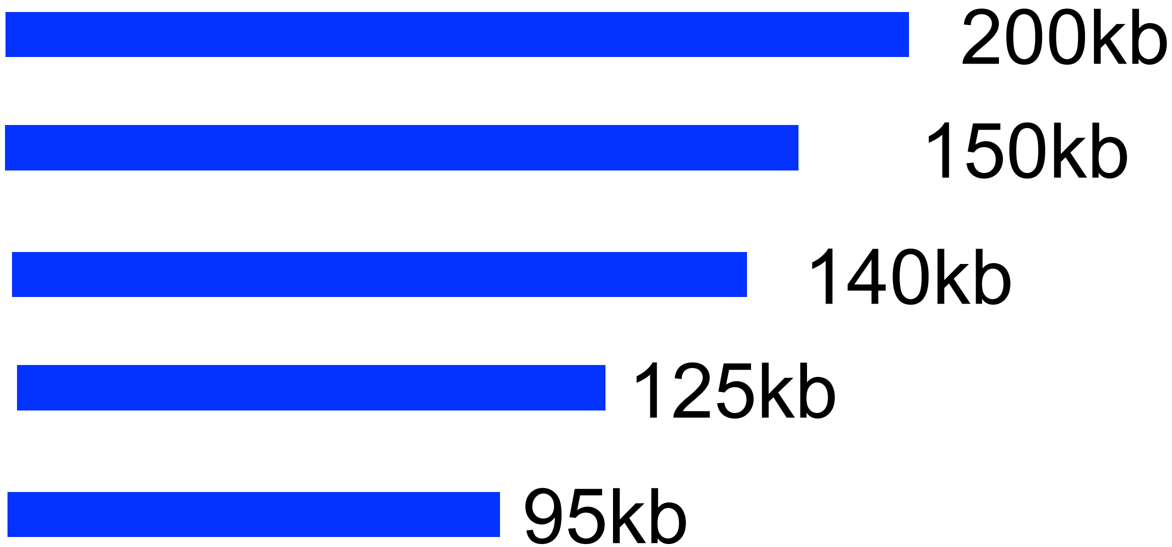
N50: What is the smallest piece in the largest half of the assembly?

- Calculate sum of assembly
- Order contigs by size
- Sum contigs starting by largest
- When half the sum is reached, N50 is the length of the contig



N50 example

5 scaffolds, calculate N50:



Sum: 200+150+140+125+95=710kb

Half: 710 / 2 = 355kb

200kb + 150kb = 350kb

350kb + 140kb = 490kb

490kb > 355kb => **N50: 140kb**

Some assemblers

- OLC: Canu, Newbler
- de Bruijn: Allpaths-LG, SPAdes (best), Velvet, SOAPdenovo, Megahit (very lean), …
- other: MIRA, SGA, Flye (very good for 3g NGS)

Used in exercises today

Exercise time!

https://teaching.healthtech.dtu.dk/22136/index.php/Metagenomic_assembly_exercise