Session 4.2 – Ensamblado

<u>Isabel Cuesta</u>

<u>BU-ISCIII</u>

<u>Unidades Comunes Científico Técnicas – SGSAFI-ISCIII</u>

3 al 10 noviembre 2022

CURSO FORMACIÓN AESAN-CNA

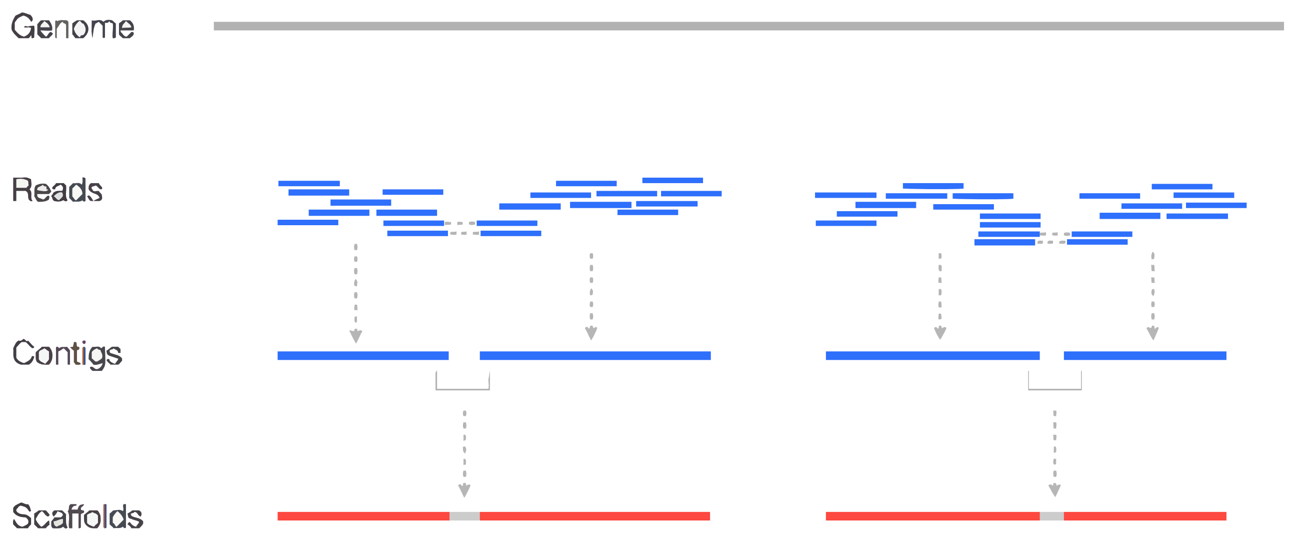
Assembly

Reconstruct the sequence of the original DNA from shorter DNA sequences or small fragments known as reads

- ***De novo:*** with no previous knowledge of the genome to be assembled. It overlap the end of the end of each read in order to create a longer sequence.
- ***Assembly with reference:*** A similar but not identical genome guides the assembly process. Map reads over supplied genome.

Assembly: contig y scaffold

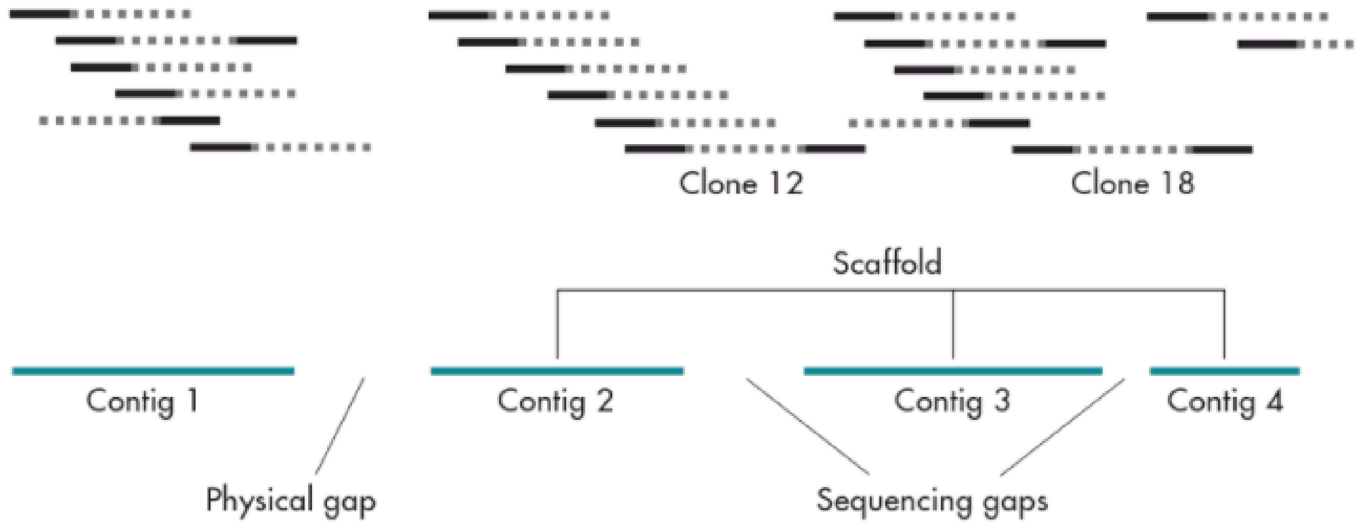
- **Contig:** continuous sequence made up of overlaping shorter sequences
- **Scaffold:** two or more contigs located and rearranged according to spatial information (pair-end, mate pair, reference)



https://www.biostars.org/p/253222/

Assembly: gaps

- **Sequencing gaps:** Position and orientation known by spatial information
- **Physical gaps:** No information about adjacent contigs

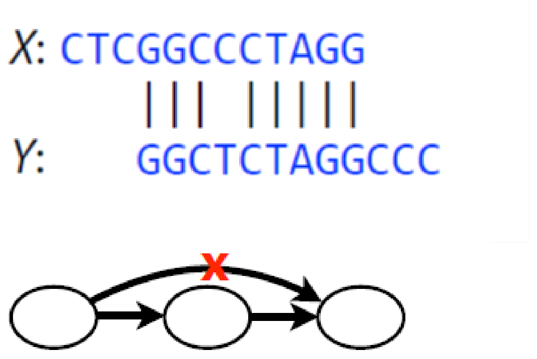


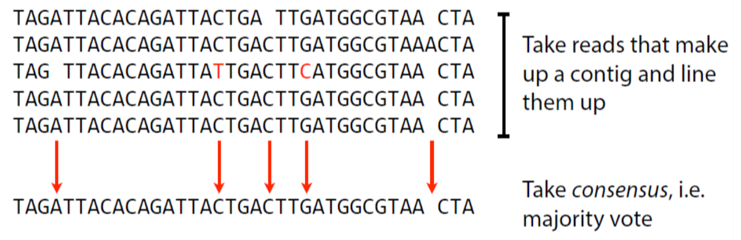
Gene Cloning, Lodge *et al*.

Assembly: Algorithms

- **Overlap, Layout, Consensus (OLC - overlap graph):**
  - O - first overlaps among all the reads are found
  - L - then it carries out a layout of all the reads and overlaps information on a graph
    - Removes redundant and low quality overlaps
  - C - and finally the consensus sequence is inferred

Ex. Newbler, Mira, Celera Assembler, CAP3, PCAP, Phrap, Phusion.





```
TAGATTACACAGATTACTGA TTGATGGCGTAA CTA
TAGATTACACAGATTACTGACTTGATGGCGTAAACTA
TAG TTACACAGATTATTGACTTCATGGCGTAA CTA
TAGATTACACAGATTACTGACTTGATGGCGTAA CTA
TAGATTACACAGATTACTGACTTGATGGCGTAA CTA

TAGATTACACAGATTACTGACTTGATGGCGTAA CTA
```

Take reads that make up a contig and line them up

Take *consensus*, i.e. majority vote

https://pt.slideshare.net/anton_alexandrov/combining-de-bruijn-graph-overlap-graph-and-microassembly/12?smtNoRedir=1

Assembly: Algorithms

- **De Brujin Graph (DBG: k-mer graph)**

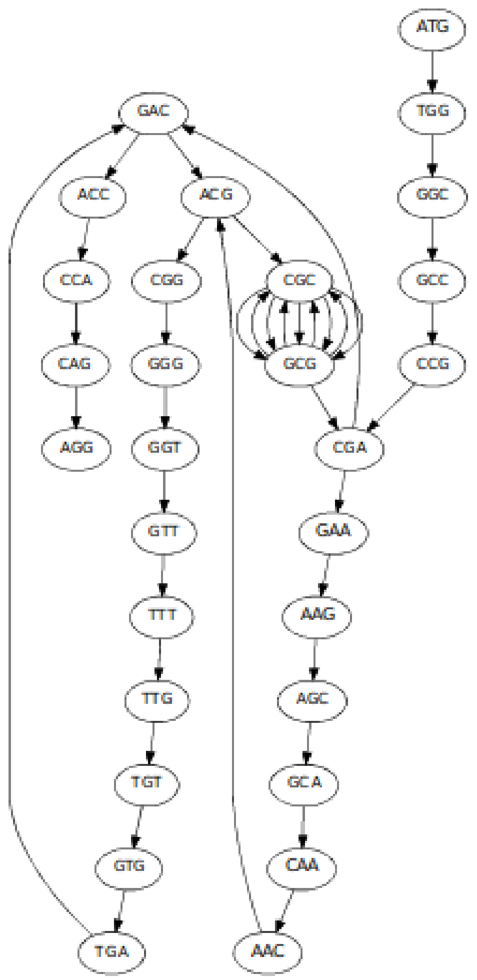
Chopping reads into much shorter k-mers (fixed length fragments) and then using all the k-mers to form a DBG and infer the contigs.

- Nodes in the graph are k-mers

- Edges represent consecutive k-mers (which overlap by k-n symbols)

Ex. SPAdes, ABySS, Velvet, AllPaths, Soap….

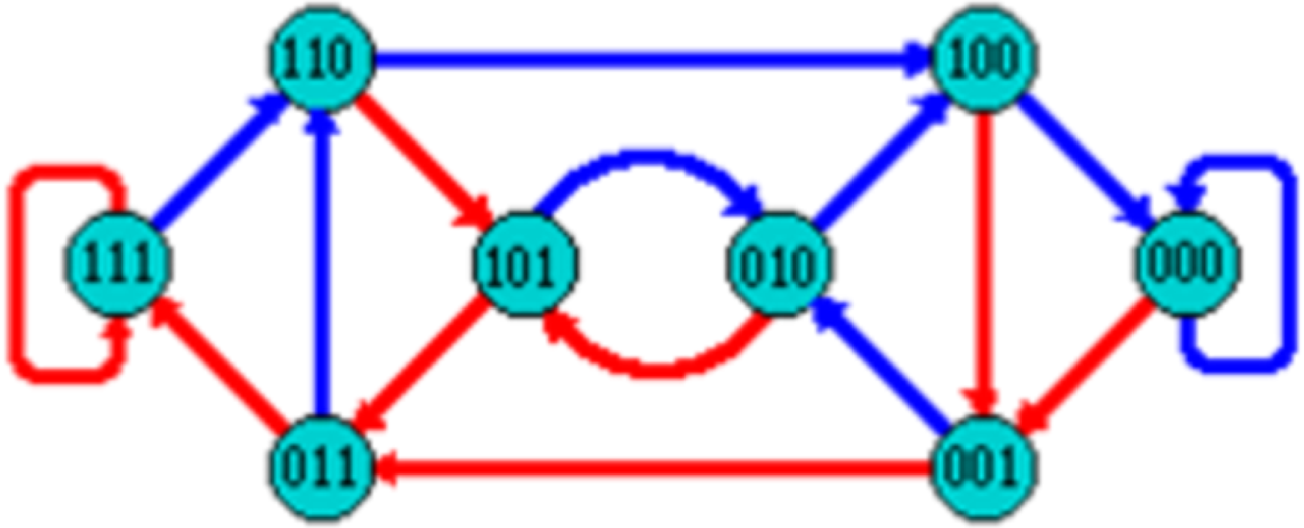
https://medium.com/@han_chen



de Bruijn Graphs

- A directed graph of sequences of symbols
- Nodes in the graph are k-mers
- Edges represent consecutive k-mers (which overlap by k-1 symbols)

Consider the 2 symbol alphabet (0 & 1) de Bruijn Graph for k =3



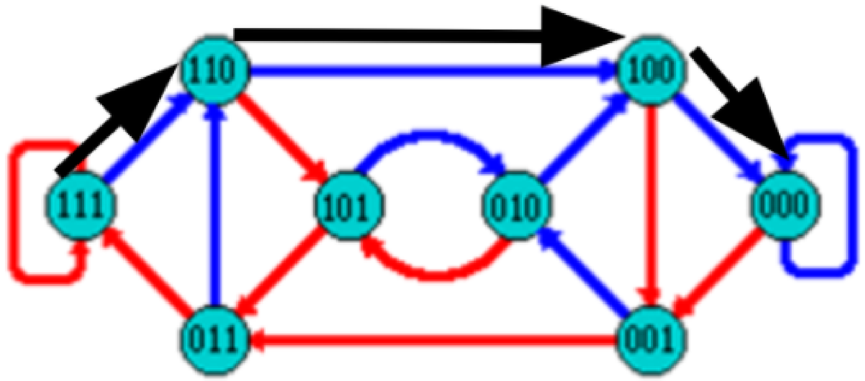
https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

Producing sequences

- Sequences of symbols are produced by moving through the graph

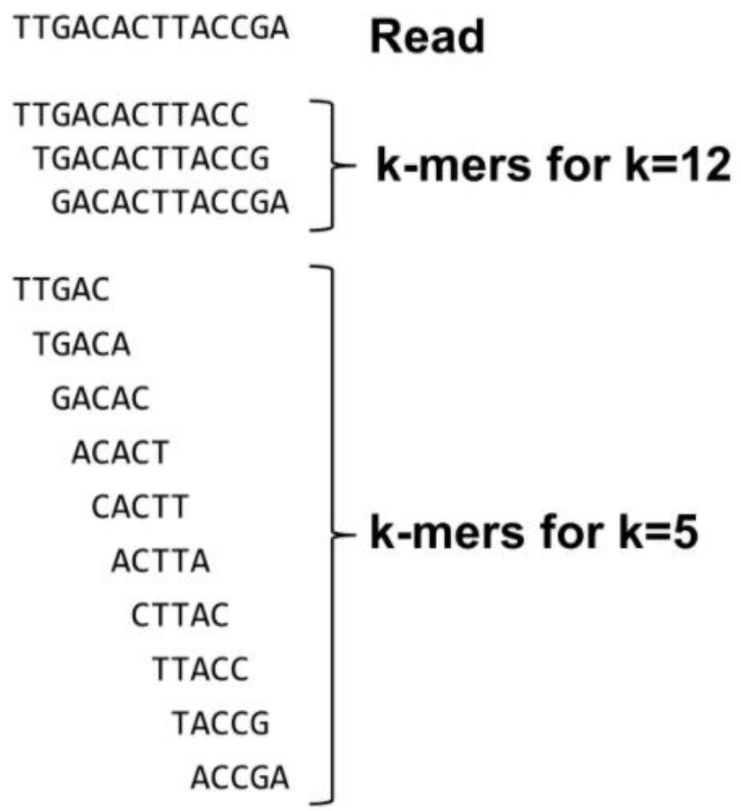
e.g. 111000 = 111 -> 110 -> 100 -> 000

```
1 1 1
  1 1 0
    1 0 0
      0 0 0
-----------
1 1 1 0 0 0
```



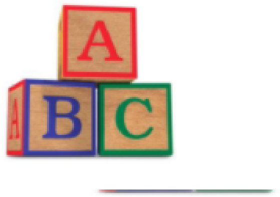
https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

What are K-mers?



https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

K-mers de Bruijn graph



Example #1:

```
HAPPI   PINE   INESS   APPIN
```

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

K-mers de Bruijn graph

Example #1:

```
HAPPI   PINE   INESS   APPIN
```

All 4-mers:

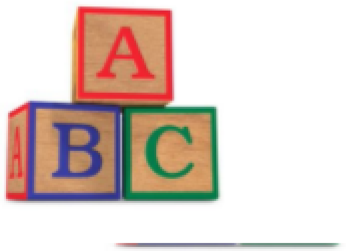
```
HAPP    PINE   INES    APPI
 APPI           NESS    PPIN
```

Unique 4-mers:

```
HAPP APPI PINE PPIN INES NESS
```

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

K-mers de Bruijn graph



Example #1:

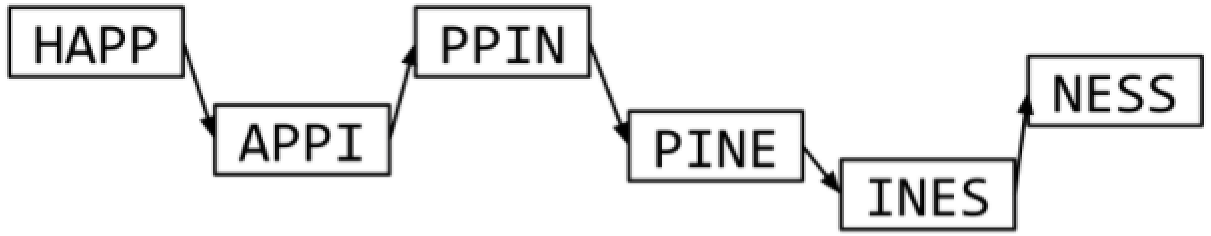
HAPPI PINE INESS APPIN

k = 4 k-mers:

HAPP APPI

PINE PPIN

INES NESS



K-mers de Bruijn graph

Example #1:

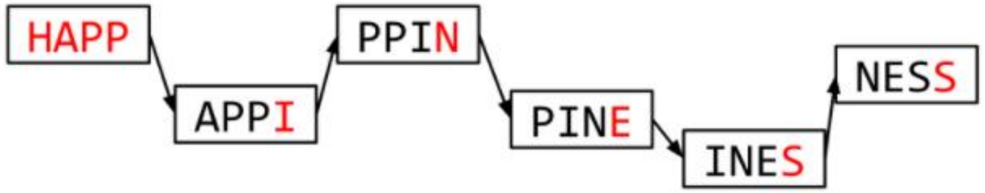
HAPPI PINE INESS APPIN

k = 4 k-mers:

HAPP APPI

PINE PPIN

INES NESS



HAPPINESS

Easy!

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

The problem of repeats

Example #2:

```
MISSIS SSISSI SSIPPI
```

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

The problem of repeats

Example #2:

```
MISSIS SSISSI SSIPPI
```

All 4-mers (9):

```
MISS   SSIS   SSIP
 ISSI   SISS   SIPP
  SSIS   ISSI   IPPI
```

Unique 4-mers (7):

```
MISS SSIS SSIP ISSI SISS SIPP IPPI
```

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

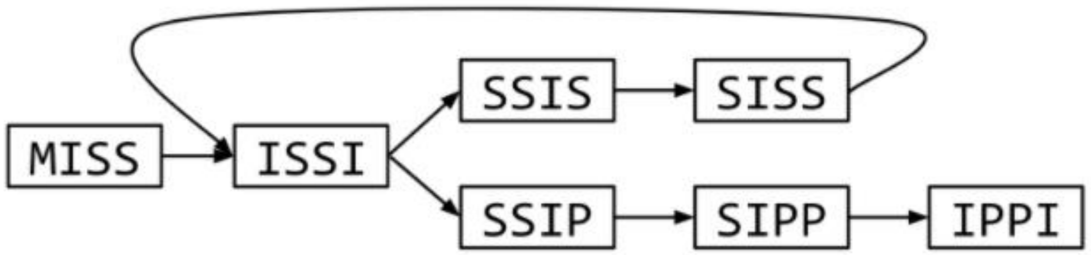
The problem of repeats

Example #2:

`MISSIS SSISSI SSIPPI`

All 4-mers:

`MISS ISSI SSIS SISS SSIP SIPP IPPI`



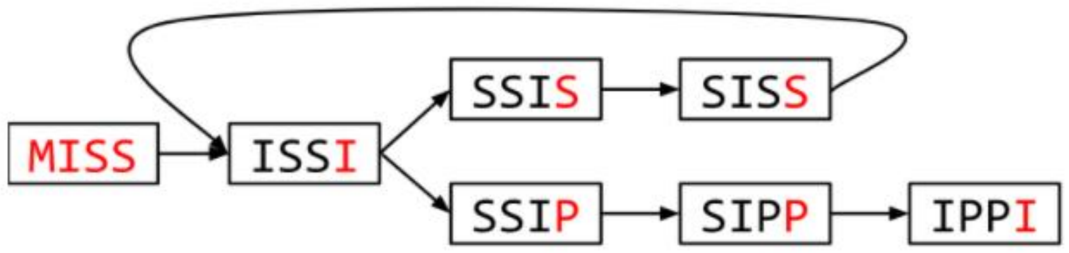
The problem of repeats

Example #2:

```
MISSIS SSISSI SSIPPI
```

All 4-mers:

```
MISS ISSI SSIS SISS SSIP SIPP IPPI
```



MISSISSIPPI or MISSISSISSIPPI or ...

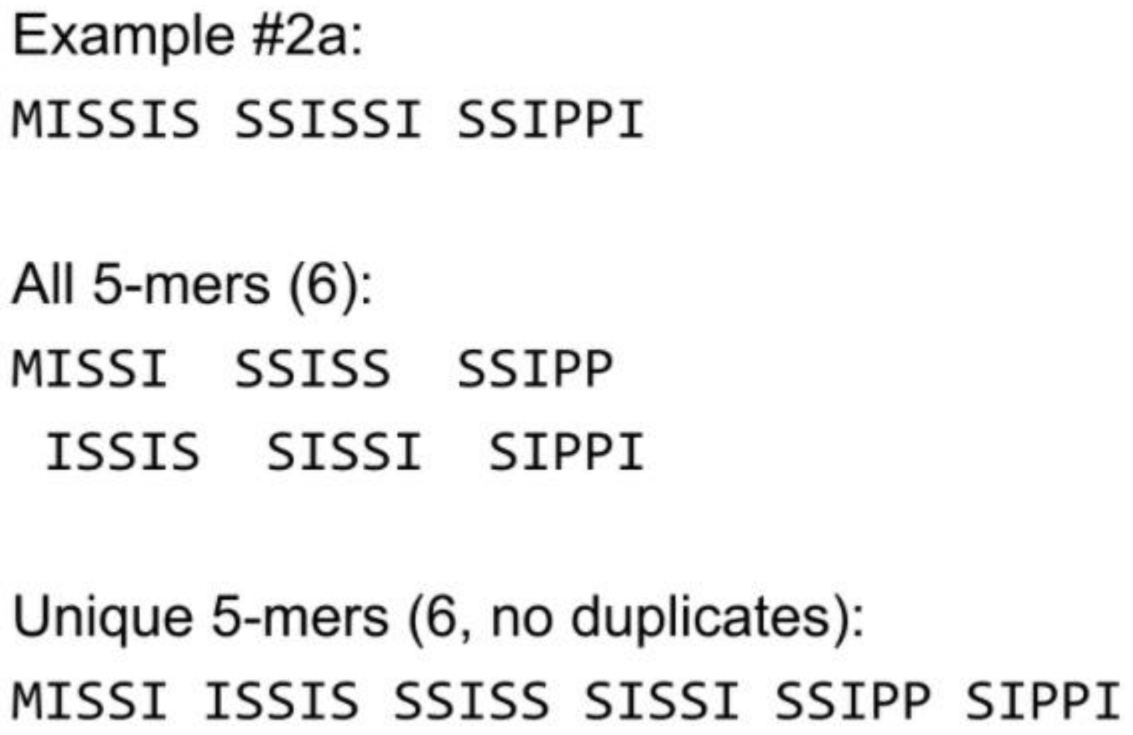
https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

Different k

Example #2a:

```
MISSIS SSISSI SSIPPI
```

Different k



Example #2a:

```
MISSIS SSISSI SSIPPI
```

All 5-mers (6):

```
MISSI   SSISS   SSIPP
 ISSIS   SISSI   SIPPI
```

Unique 5-mers (6, no duplicates):

```
MISSI ISSIS SSISS SISSI SSIPP SIPPI
```

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

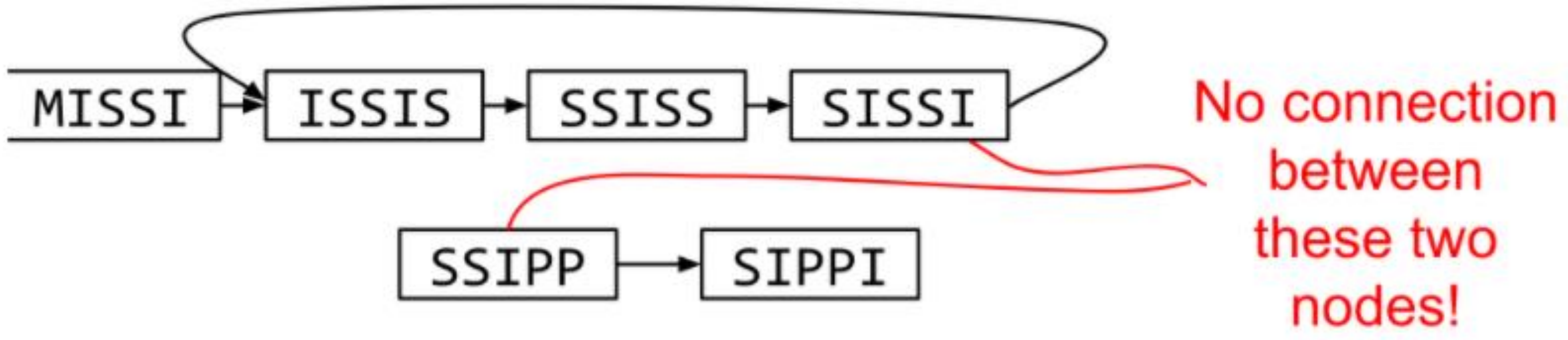
Different k

Example #2a:

MISSIS SSISSI SSIPPI

This time k = 5 k-mers:

MISSI ISSIS SSISS SISSI SSIPP SIPPI



https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

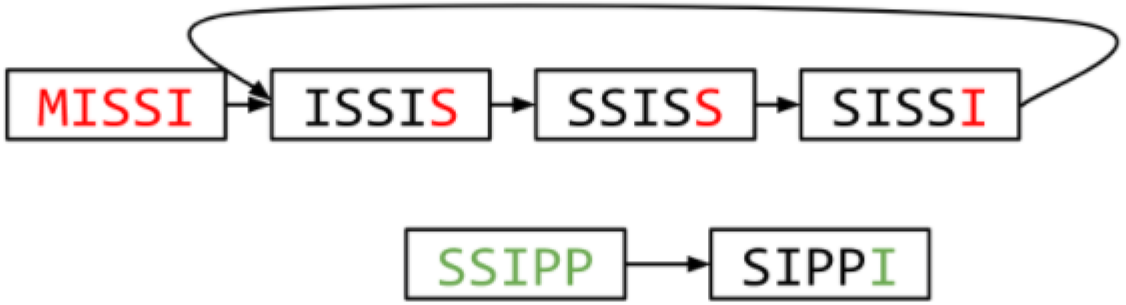
Different k

Example #2a:

MISSIS SSISSI SSIPPI

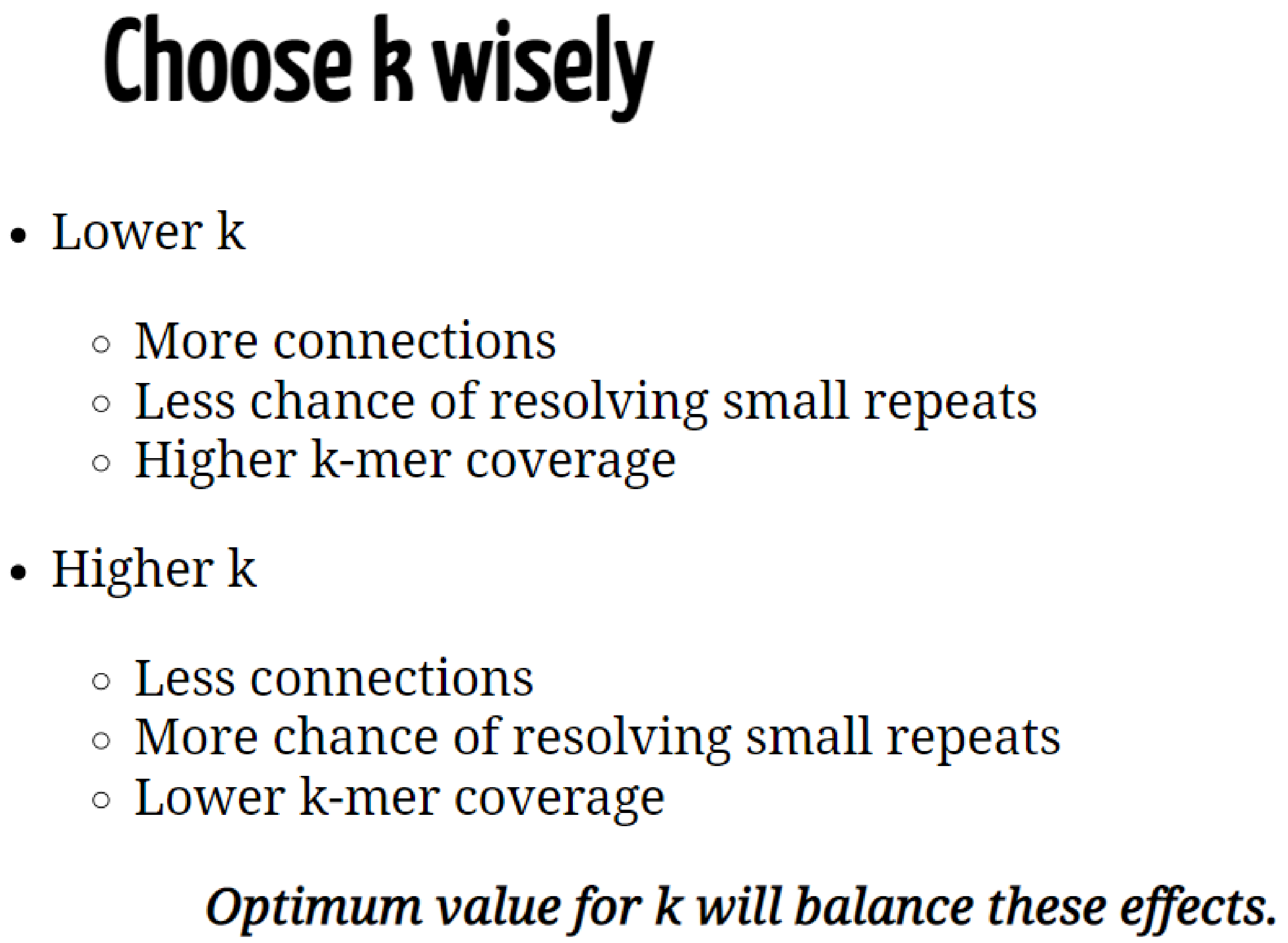
This time k = 5 k-mers:

MISSI ISSIS SSISS SISSI SSIPP SIPPI



MISSISSIS SSIPPI

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23



Choose k wisely

- Lower k
  - More connections
  - Less chance of resolving small repeats
  - Higher k-mer coverage
- Higher k
  - Less connections
  - More chance of resolving small repeats
  - Lower k-mer coverage

Optimum value for k will balance these effects.

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

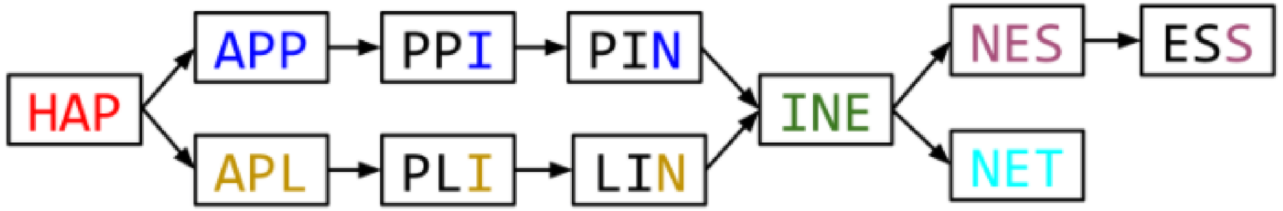
Read errors

Example #3:

HAPPI INESS APLIN PINET

k = 3 k-mers:

HAP APP PPI INE NES ESS APL PLI LIN PIN NET



6 contigs: HAP APPIN APLIN INE NESS NET

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

More coverage

- Errors won't be duplicated in every read
- Most reads will be error free
- We can count the frequency of each k-mer
- Annotate the graph with the frequencies
- Use the frequency data to clean the de Bruijn graph

More coverage depth will help overcome errors!

https://galaxyproject.github.io/training-material/topics/assembly/tutorials/debruijn-graph-assembly/slides.html#23

SPAdes

- de Bruijn graph assembler by Pavel Pevzner's group out of St. Petersburg
- Uses multiple k-mers to build the graph
  - Graph has connectivity **and** specificity
  - Usually use a low, medium and high k-mer size together.
- Performs error correction on the reads first
- Maps reads back to the contigs and scaffolds as a check
- Under active development
- Much slower than Velvet
- Should be used in preference to Velvet now.

Assembly: Scaffolding

- **From draft:**

  Order contigs (Nucmer, if there is reference it can be used to align and guide)

  Fill the GAPs (GapFiller, fill sequencing gap (not physical gap)

  Solve repeated sequence ambiguities (Expander)

  Resequence with different library:

  - Longer fragments and/or distance

- **Tools for assembly improvement**

  SSPACE (Scaffolding) REAPR (evaluate scaffolding, breaking incorrect scaffolds)

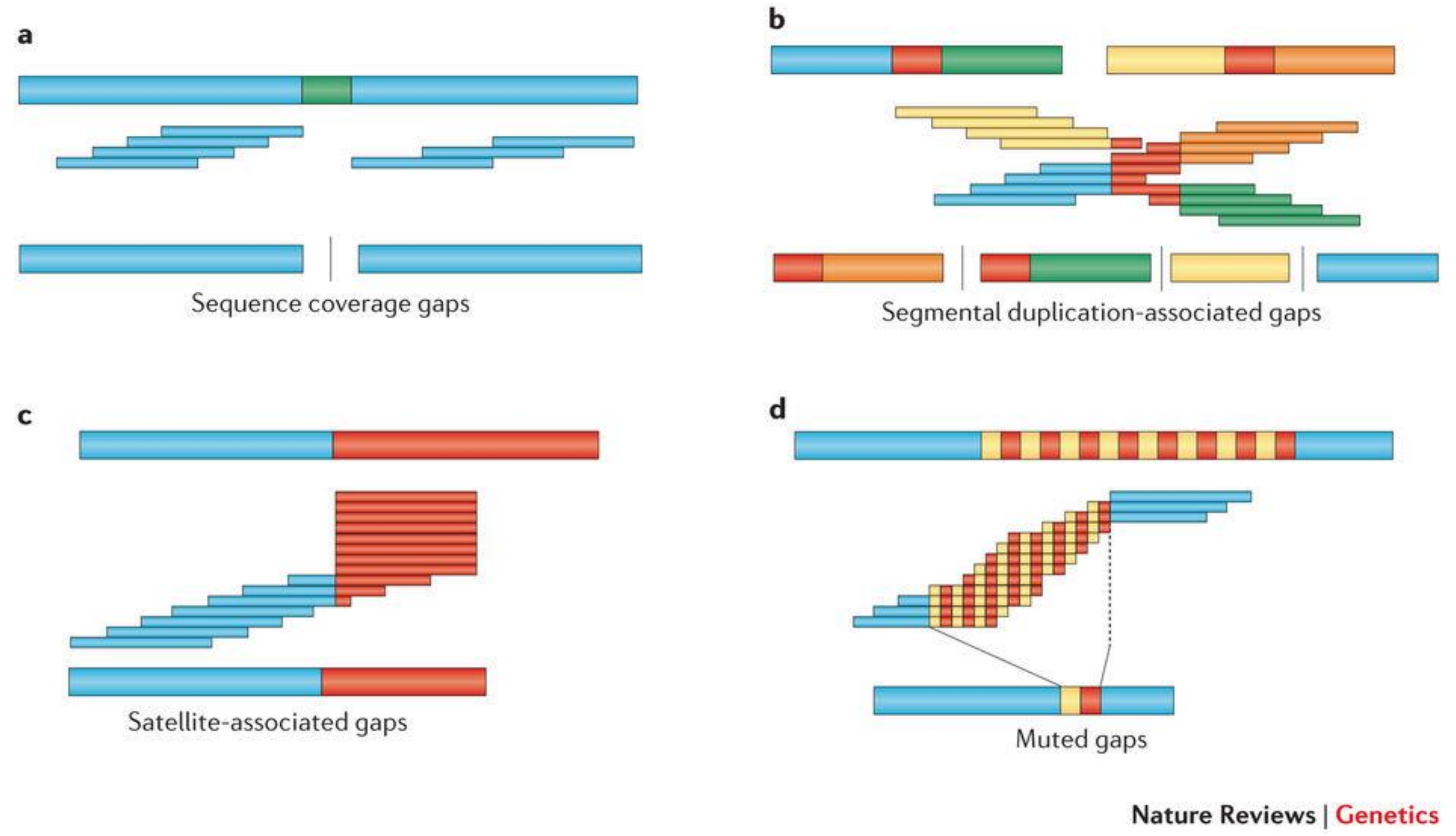
- **Assembly visualyzing**

  Artemis, ACT (compare two or more sequences), Icarus (Quast)

A move back to OLC

- New long read technologies
  - PacBio and MinIon
- Assemblers: HGap, CANU
  - Use overlap, layout consensus approach
- CANU can perform hybrid assemblies with long and short reads

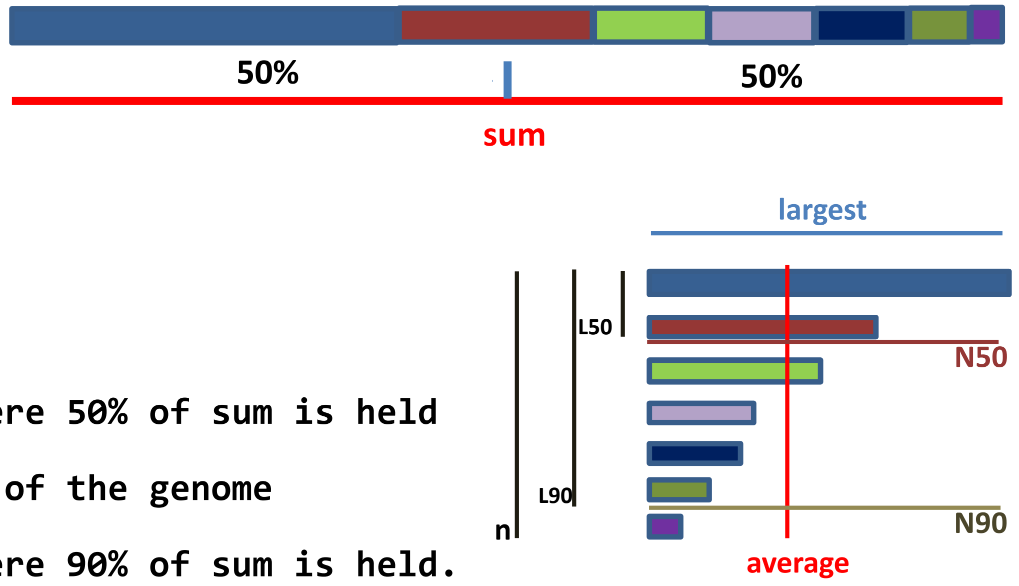
Ensamblado: Errores


a
Sequence coverage gaps
b
Segmental duplication-associated gaps
c
Satellite-associated gaps
d
Muted gaps
Nature Reviews | Genetics

- **A. Gaps – región del genoma sin secuenciar**
- **B. Duplicaciones de gran tamaño**
  - Quimeras
- **Regiones repetidas colapsadas**
  - **C. Terminales**
  - **D. Intersticiales**

Genetic variation and the de novo assembly of human genomes
Chaisson *et al*. 2015

Assembly: Metrics



- sum = total bases number
- n = contigs number
- average = average contig length
- largest = largest contig
- N50 = length of the shortest contig where 50% of sum is held
- L50 = number of contigs which have 50% of the genome
- N90 = length of the shortest contig where 90% of sum is held.
- L90 = number of contigs which have 90% of the genome

Assembly: Evaluation

- Software that evaluate differets algorithms & parameters
  - iMetAMOS, *Koren et al., BMCBioinformatics 2014, 15:126*
  - GAGE-B, *Magoc et al.,* Bioinformatics 2013,29(14):1718-25

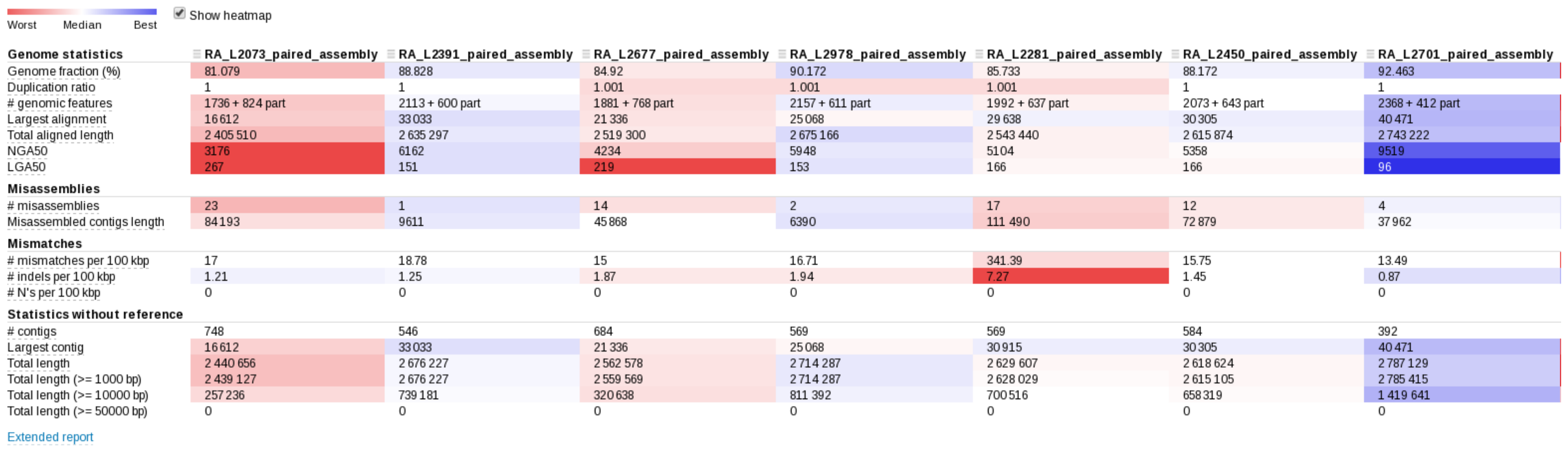
- **Graph evaluation**: Bandage, Wick R.R., Schultz M.B., Zobel J. & Holt K.E. (2015)

- **Assembly evaluation**: Quast, *Gurevich et al., Bioinformatics 2013, 29:8*

- **Metrics for a good assembly:**
  Large N50
  Sum closest to expected
  Low n
  Low L50

Assembly: Evaluation - Quast

- Assembly evaluation: Quast, *Gurevich et al., Bioinformatics 2013, 29:8*



Worst Median Best

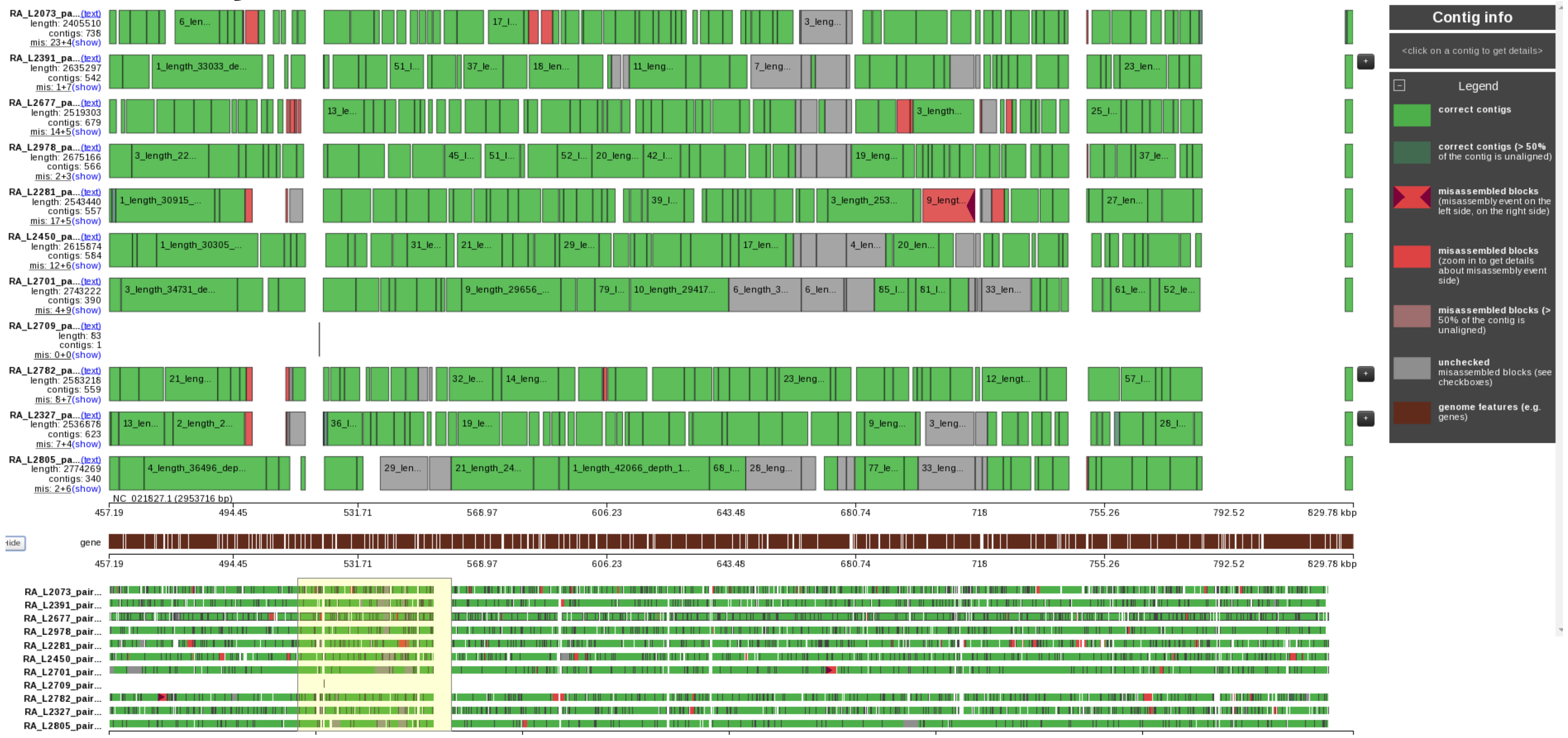
✓ Show heatmap

Genome statistics	RA_L2073_paired_assembly	RA_L2391_paired_assembly	RA_L2677_paired_assembly	RA_L2978_paired_assembly	RA_L2281_paired_assembly	RA_L2450_paired_assembly	RA_L2701_paired_assembly
Genome fraction (%)	81.079	88.828	84.92	90.172	85.733	88.172	92.463
Duplication ratio	1	1	1.001	1.001	1.001	1	1
# genomic features	1736 + 824 part	2113 + 600 part	1881 + 768 part	2157 + 611 part	1992 + 637 part	2073 + 643 part	2368 + 412 part
Largest alignment	16 612	33 033	21 336	25 068	29 638	30 305	40 471
Total aligned length	2 405 510	2 635 297	2 519 300	2 675 166	2 543 440	2 615 874	2 743 222
NGA50	3176	6162	4234	5948	5104	5358	9519
LGA50	267	151	219	153	166	166	96
Misassemblies							
# misassemblies	23	1	14	2	17	12	4
Misassembled contigs length	84 193	9611	45 868	6390	111 490	72 879	37 962
Mismatches							
# mismatches per 100 kbp	17	18.78	15	16.71	341.39	15.75	13.49
# indels per 100 kbp	1.21	1.25	1.87	1.94	7.27	1.45	0.87
# N's per 100 kbp	0	0	0	0	0	0	0
Statistics without reference							
# contigs	748	546	684	569	569	584	392
Largest contig	16 612	33 033	21 336	25 068	30 915	30 305	40 471
Total length	2 440 656	2 676 227	2 562 578	2 714 287	2 629 607	2 618 624	2 787 129
Total length (>= 1000 bp)	2 439 127	2 676 227	2 559 569	2 714 287	2 628 029	2 615 105	2 785 415
Total length (>= 10000 bp)	257 236	739 181	320 638	811 392	700 516	658 319	1 419 641
Total length (>= 50000 bp)	0	0	0	0	0	0	0

Extended report

Assembly: Evaluation - Quast

- Assembly evaluation: Quast, *Gurevich et al., Bioinformatics 2013, 29:8*


RA_L2073_pa...(text) length: 2405510 contigs: 738 mis: 23+4(show)
RA_L2391_pa...(text) length: 2635297 contigs: 542 mis: 1+7(show)
RA_L2677_pa...(text) length: 2519303 contigs: 679 mis: 14+5(show)
RA_L2978_pa...(text) length: 2675166 contigs: 566 mis: 2+3(show)
RA_L2281_pa...(text) length: 2543440 contigs: 557 mis: 17+5(show)
RA_L2450_pa...(text) length: 2615874 contigs: 584 mis: 12+6(show)
RA_L2701_pa...(text) length: 2743222 contigs: 390 mis: 4+9(show)
RA_L2709_pa...(text) length: 83 contigs: 1 mis: 0+0(show)
RA_L2782_pa...(text) length: 2583218 contigs: 559 mis: 8+7(show)
RA_L2327_pa...(text) length: 2536878 contigs: 623 mis: 7+4(show)
RA_L2805_pa...(text) length: 2774269 contigs: 340 mis: 2+6(show)
NC_021827.1 (2953716 bp)
457.19
494.45
531.71
568.97
606.23
643.48
680.74
718
755.26
792.52
829.78 kbp
gene
Contig info
<click on a contig to get details>
Legend
correct contigs
correct contigs (> 50% of the contig is unaligned)
misassembled blocks (misassembly event on the left side, on the right side)
misassembled blocks (zoom in to get details about misassembly event side)
misassembled blocks (> 50% of the contig is unaligned)
unchecked misassembled blocks (see checkboxes)
genome features (e.g. genes)

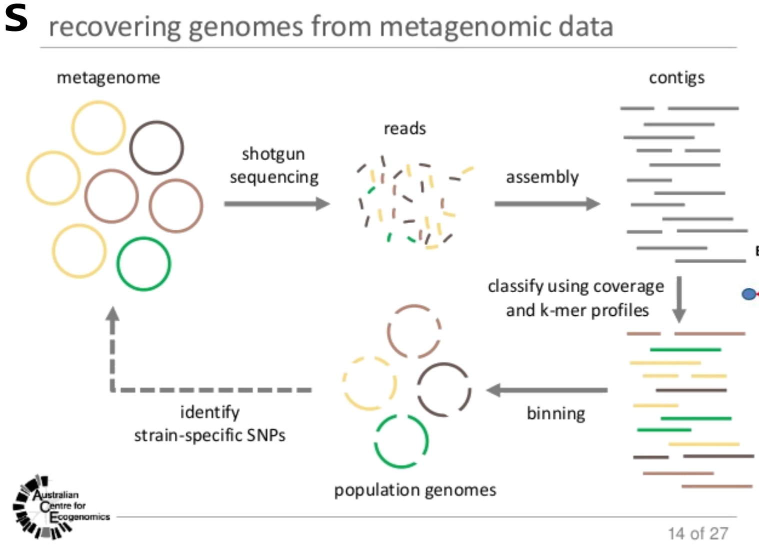
Assembly: Assemblers

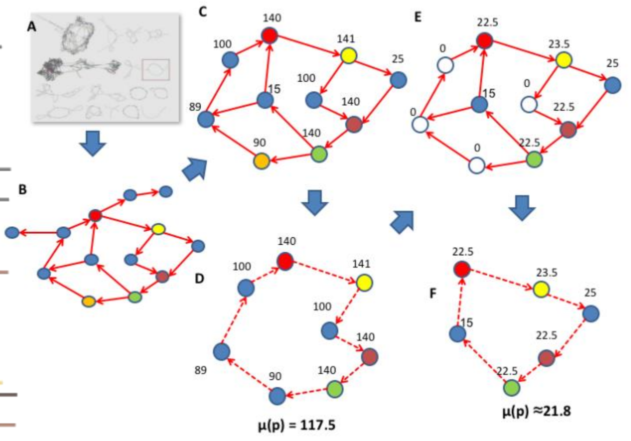
Name	Type	Technologies	Author	Presented /Last updated	Licence*	Homepage
DNASTAR Lasergene Genomics Suite	(large) genomes, exomes, transcriptomes, metagenomes, ESTs	Illumina, ABI SOLiD, Roche 454, Ion Torrent, Solexa, Sanger	DNASTAR	2007 / 2016	C	link
Newbler	genomes, ESTs	454, Sanger	454/Roche	2004/2012	C	link
Canu	Small and large, haploid/diploid genomes	PacBio/Oxford Nanopore reads	Koren et al.[8]	2001 / 2018	OS	link
SPAdes	(small) genomes, single-cell	Illumina, Solexa, Sanger, 454, Ion Torrent, PacBio, Oxford Nanopore	Bankevich, A et al.	2012 / 2017	OS	link
Velvet	(small) genomes	Sanger, 454, Solexa, SOLiD	Zerbino, D. et al.	2007 / 2011	OS	link

***Licences:** OS = Open Source; C = Commercial; C / NC-A = Commercial but free for non-commercial and academics

Assembly: Specials assemblers

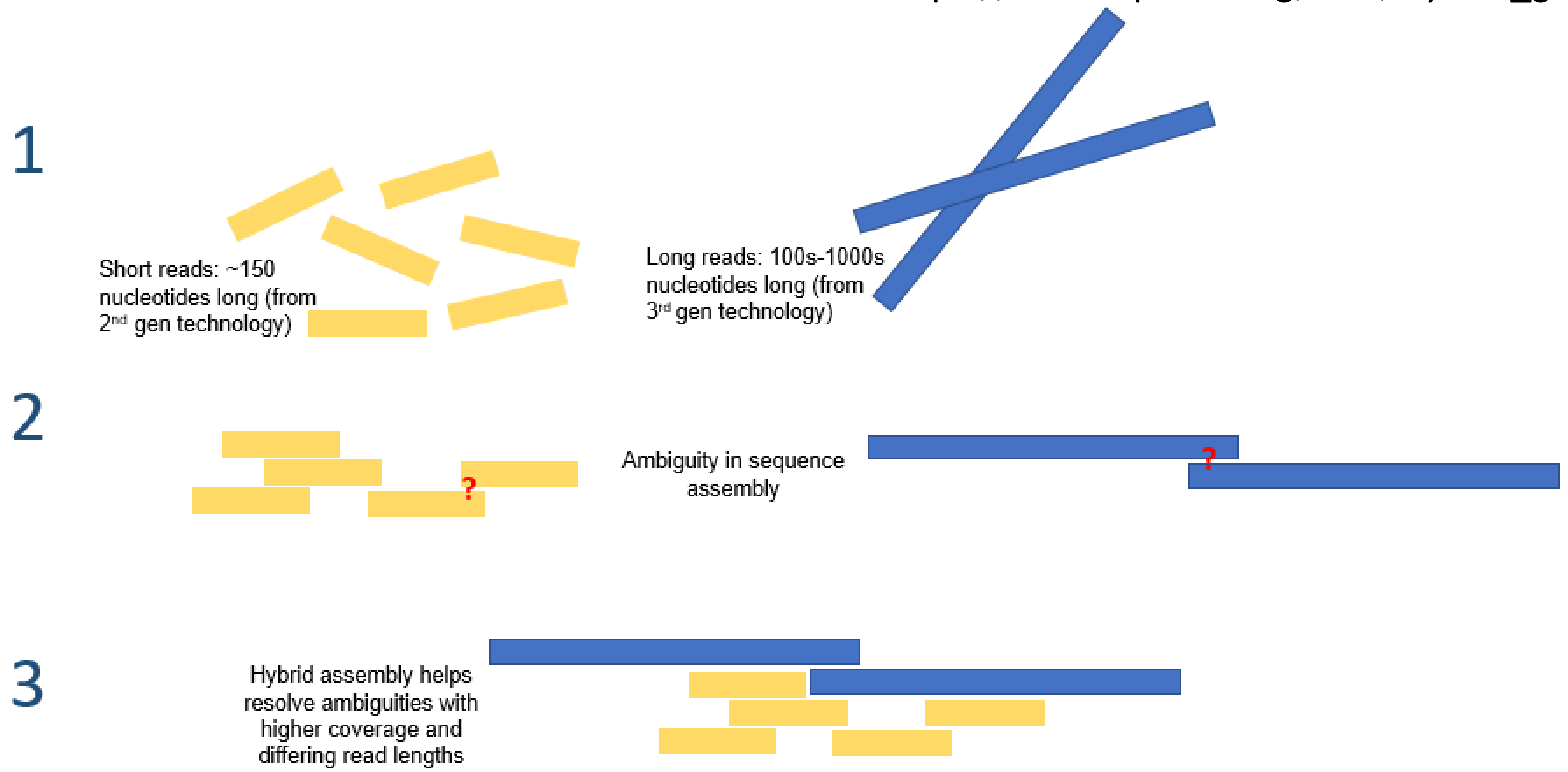
- Diploid genomes
- Metagenomics
- Plasmids
- Transcriptome


recovering genomes from metagenomic data
metagenome
shotgun sequencing
reads
assembly
contigs
classify using coverage and k-mer profiles
binning
population genomes
identify strain-specific SNPs
Australian Centre for Ecogenomics
14 of 27



Hybrid genome assembly – short and long reads

https://en.wikipedia.org/wiki/Hybrid_genome_assembly

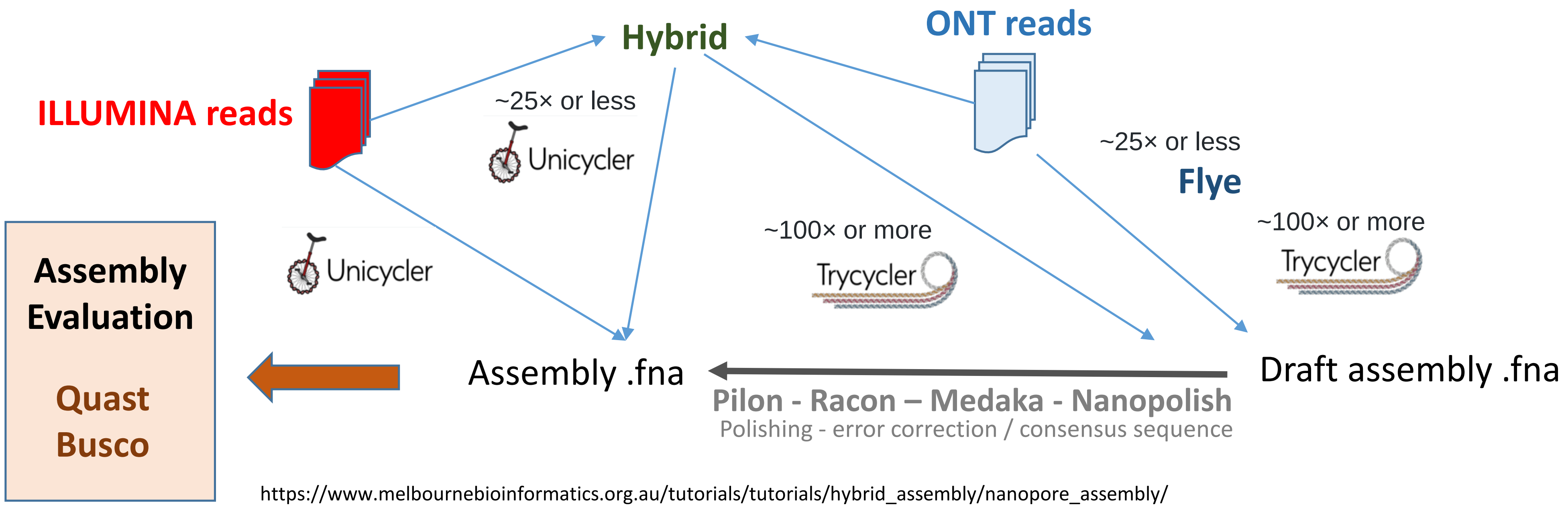

1
Short reads: ~150 nucleotides long (from 2nd gen technology)
Long reads: 100s-1000s nucleotides long (from 3rd gen technology)
2
?
Ambiguity in sequence assembly
?
3
Hybrid assembly helps resolve ambiguities with higher coverage and differing read lengths

Hybrid genome assembly - nanopore and illumina

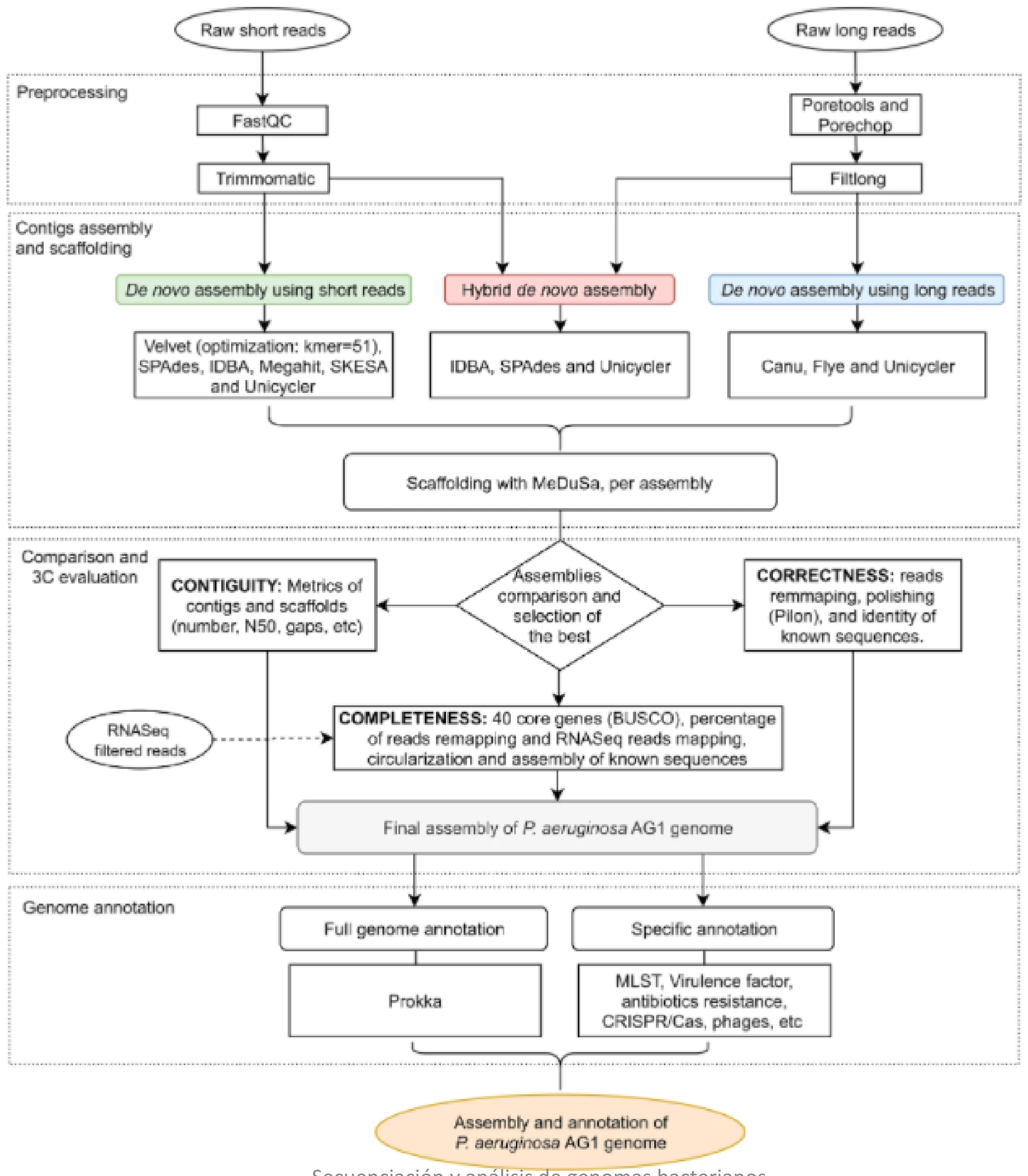
Short reads (ILLUMINA) + Long reads (ONT) → deNovo assembly (De novo assembly is the process of assembling a genome from scratch using only the sequenced reads as input - no reference genome is used.) → **high-quality assembly**

ONT: >40.000b, higher error rate – **genome structure**
ILLUMINA: 300b, lower error rate – **high base-level accuracy**

Higher COST



https://www.melbournebioinformatics.org.au/tutorials/tutorials/hybrid_assembly/nanopore_assembly/
https://github.com/rrwick/Unicycler
https://denbi-nanopore-training-course.readthedocs.io/en/latest/index.html

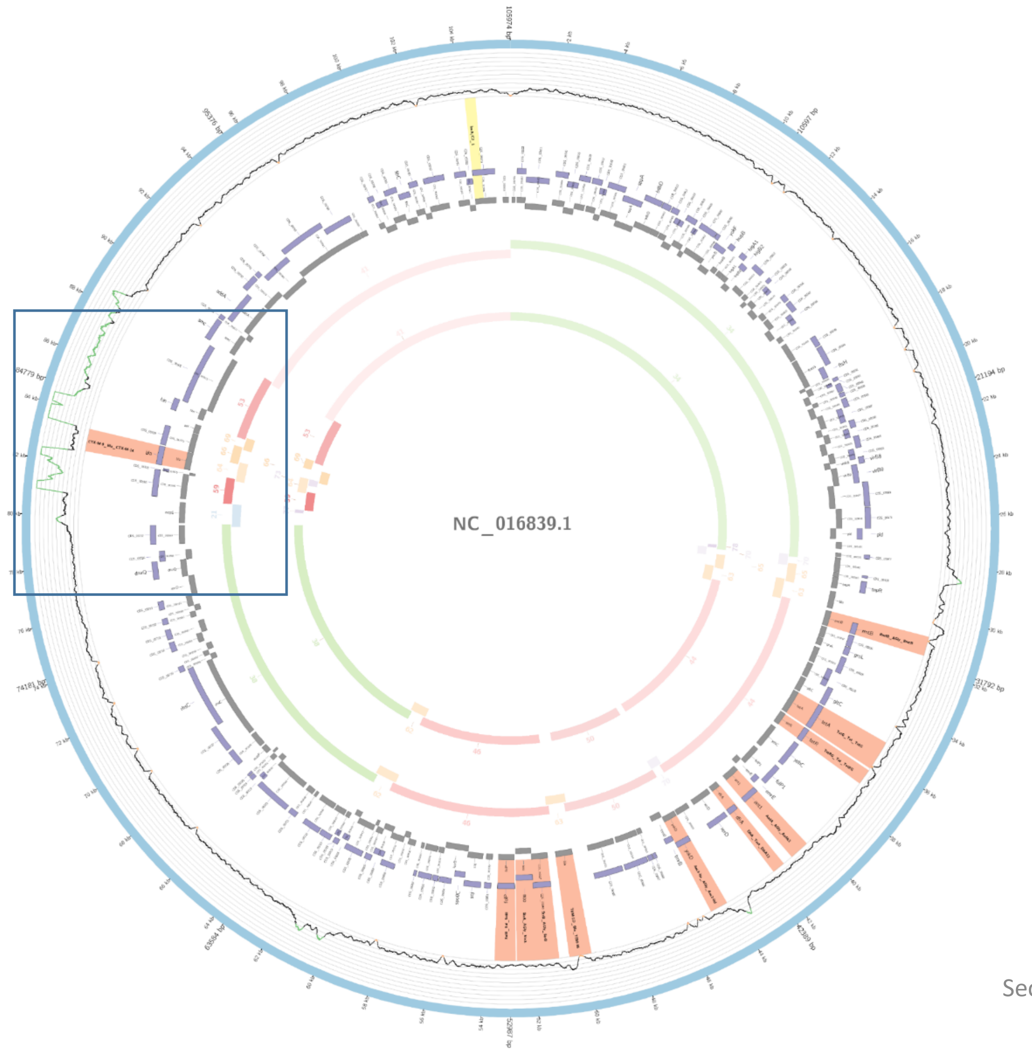


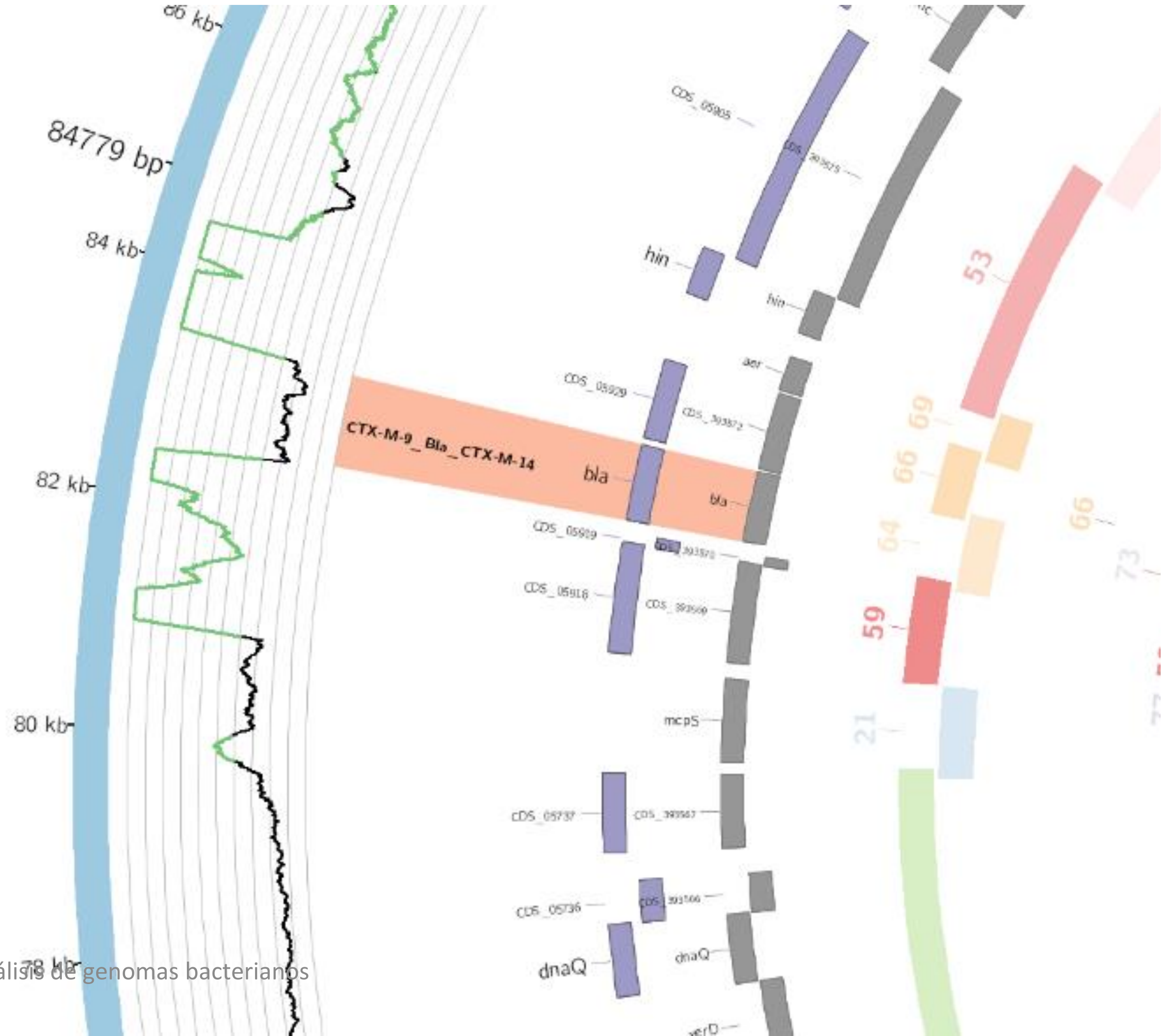
Raw short reads
Raw long reads
Preprocessing
FastQC
Poretools and Porechop
Trimmomatic
Filtlong
Contigs assembly and scaffolding
De novo assembly using short reads
Hybrid *de novo* assembly
De novo assembly using long reads
Velvet (optimization: kmer=51), SPAdes, IDBA, Megahit, SKESA and Unicycler
IDBA, SPAdes and Unicycler
Canu, Flye and Unicycler
Scaffolding with MeDuSa, per assembly
Comparison and 3C evaluation
CONTIGUITY: Metrics of contigs and scaffolds (number, N50, gaps, etc)
Assemblies comparison and selection of the best
CORRECTNESS: reads remmaping, polishing (Pilon), and identity of known sequences.
RNASeq filtered reads
COMPLETENESS: 40 core genes (BUSCO), percentage of reads remapping and RNASeq reads mapping, circularization and assembly of known sequences
Final assembly of *P. aeruginosa* AG1 genome
Genome annotation
Full genome annotation
Specific annotation
Prokka
MLST, Virulence factor, antibiotics resistance, CRISPR/Cas, phages, etc
Assembly and annotation of *P. aeruginosa* AG1 genome

General bioinformatic pipeline to assemble, compare and annotate the *Pseudomonas aeruginosa* AG1 genome using short and long reads as well as hybrid approaches.

Molina-Mora et al., Scientific Reports 2020

PlasmidID


NC_016839.1
84779 bp


86 kb
84779 bp
84 kb
82 kb
80 kb
78 kb
CTX-M-9_Bla_CTX-M-14
bla
bla
hin
hin
aer
mcpS
dnaQ
dnaQ
CDS_05929
CDS_05919
CDS_05916
CDS_05737
CDS_05736
53
69
66
66
64
59
21
73

Thanks for your attention!

Questions ?