MUNI
FI

Datové struktury, algoritmy a nástroje pro zpracování genomických dat

IV110/IV114/E4014 Projekt z bioinformatiky (a systémové biologie)

MUNI
FI

NGS sequencing

Read length
Gbp per run

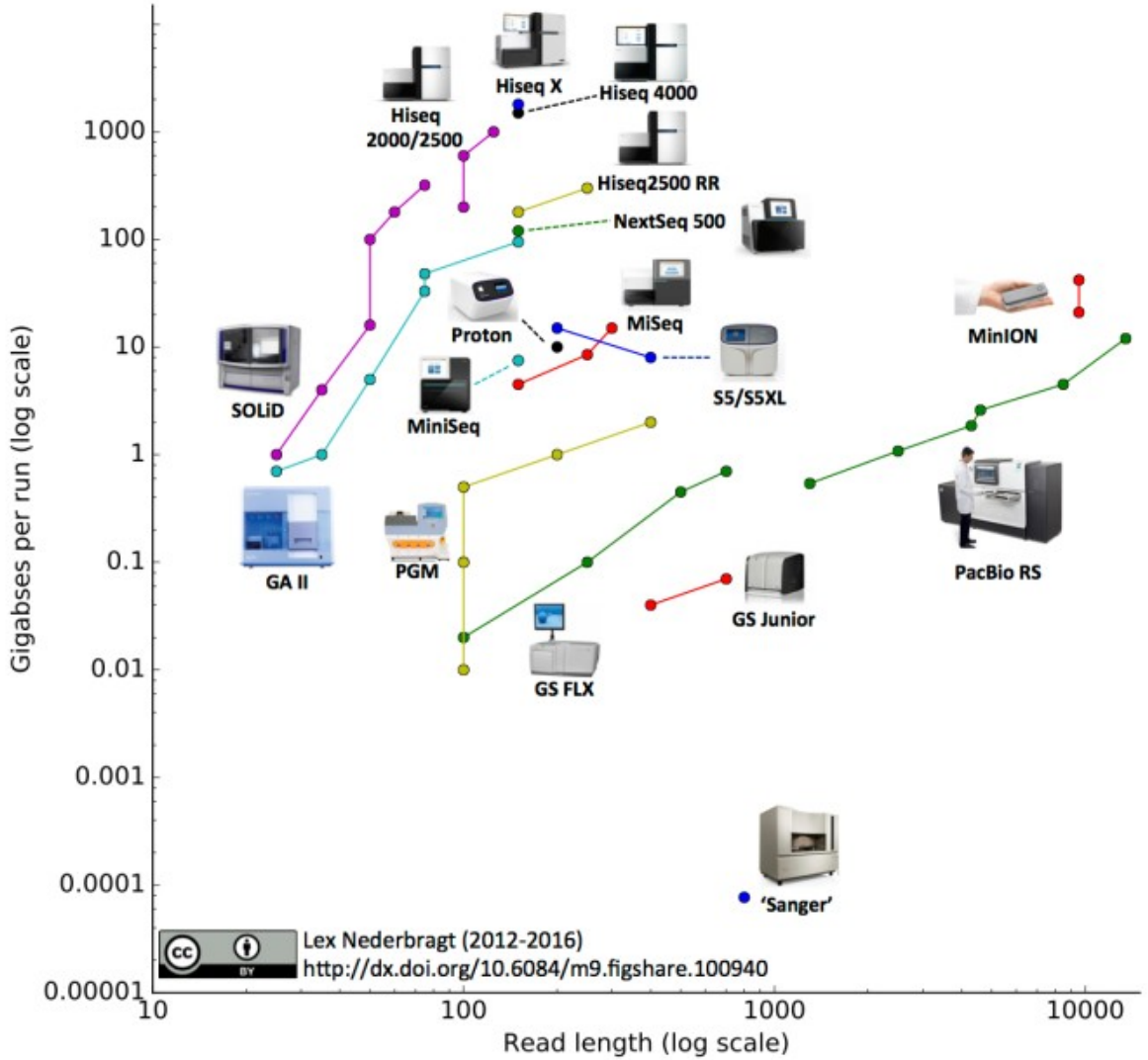


Figure 1.2: Comparison of sequencers based on their sequencing capacity and the length of the reads they produce (Nederbragt, 2016).

MUNI
FI

FASTQ data format

```
@ERR030887.1 HWI-BRUNOP16X_0001:8:1:7336:1073#0/1
TNTCGATTACATGTGGATCAGGTTGATTTAATAATGGCGATAGGGNNNCT
+
5#1455555555A;A84455555555>>>>.=@####################
@ERR030887.2 HWI-BRUNOP16X_0001:8:1:10288:1073#0/1
TNAGTCTTTCCCAGCCTAACAAAGAAAGCAAGAATAATTGGGCACNNNGA
+
5#156+43&4(0*55CFDAF##################################
@ERR030887.3 HWI-BRUNOP16X_0001:8:1:13787:1073#0/1
ANGTTGCTATTCCCGGCCGTCTAAACCAAACCACTTTCACCGCTANNNGA
+
5#55555554GGGG?FFFFFGGGGEGGGGGGGEGGCC>C############
@ERR030887.4 HWI-BRUNOP16X_0001:8:1:15389:1074#0/1
CNGTTCAAGCAGAAGACGTTCTGGGCGTCTGTATGGACACTGATCNNNAG
+
5#555525555544SEGGGGGGGGA@;>A>A<A>A################
@ERR030887.5 HWI-BRUNOP16X_0001:8:1:16693:1073#0/1
CNAGTCCGTCACTCCATCCTACCCTTATGGGCCAGGTAAGCCAACNNNCC
+
5#555))665=<H<F@1=E:88<(=55441A?AADCBFB############
```

Read ID

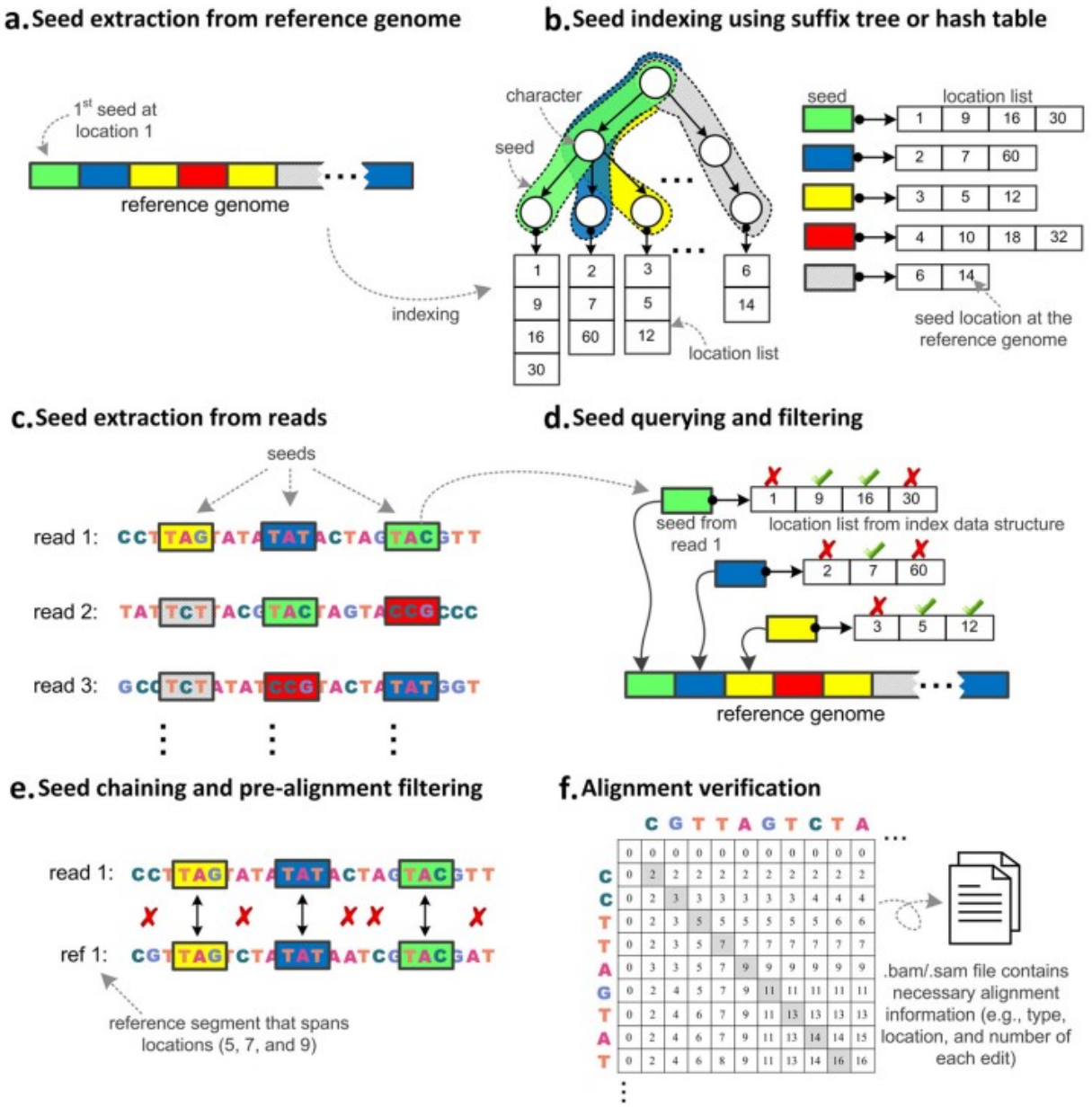
Sequenced Read

Ignore

Quality Info

MUNI FI

Efficient read mapping algorithms are based on k-mers



MUNI
FI

Suffix trees

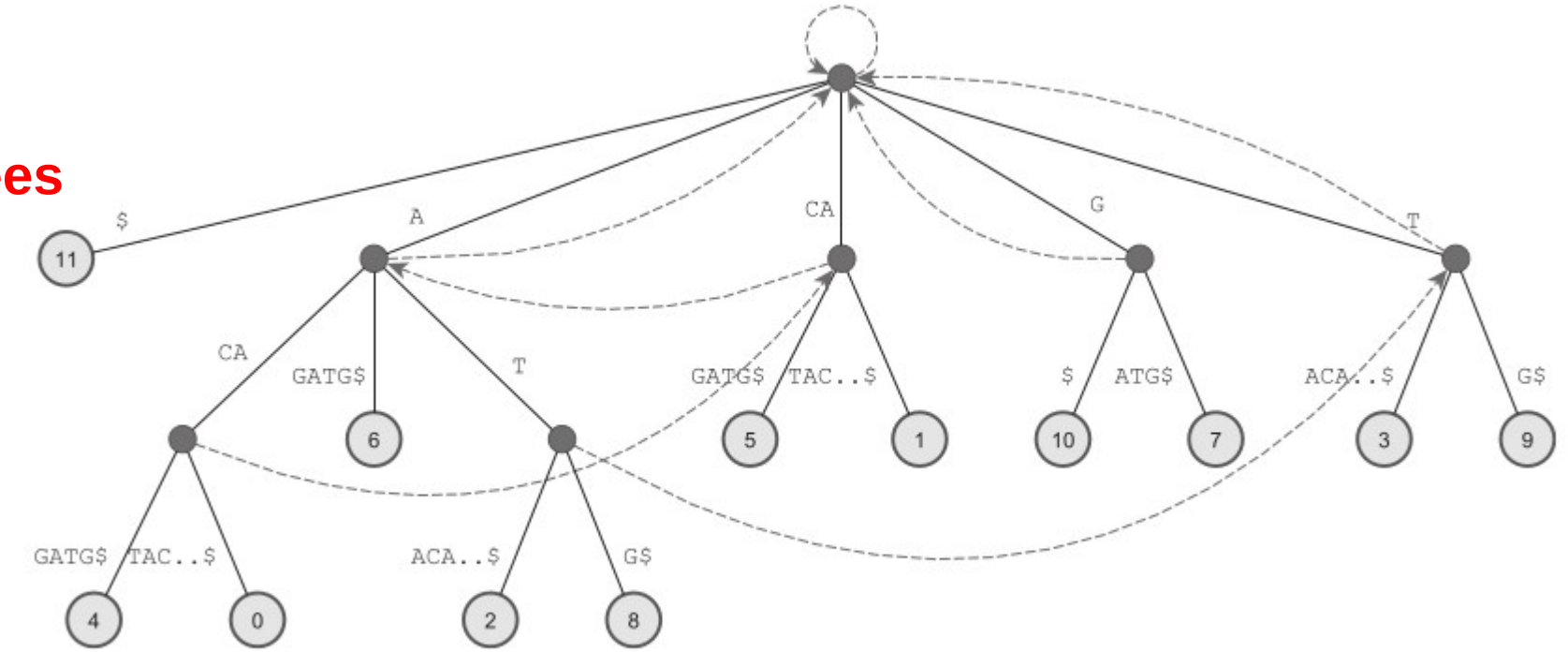


Figure 1. Suffix tree for string S = ACATACAGATG, where $ is the special end-character. Each number i inside a leaf represents suffix $S[i..]$ of the string S. Dashed arrows correspond to suffix links. Edges are arranged in lexicographical order. For the sake of brevity, only the first characters followed by two dots and the special end-character $ are shown for edge labels that spell out the rest of the suffix corresponding to the leaf the edge is connected with.

MUNI
FI

BWT-based index

Table 2. Conceptual matrix M containing the lexicographically ordered n cyclic shifts of S = ACATACAGATG$

i	S[SA[i]]		BWT[i]	offset[i]	LF[i]
0	$	ACATACAGAT	G	0	8
1	A	CAGATG$ACA	T	0	10
2	A	CATACAGATG	$	0	0
3	A	GATG$ACATA	C	0	6
4	A	TACAGATG$A	C	1	7
5	A	ATG$ACATAC	G	1	9
6	C	AGATG$ACAT	A	0	1
7	C	ATACAGATG$	A	1	2
8	G	$ACATACAGA	T	1	11
9	G	ATG$ACATAC	A	2	3
10	T	ACAGATG$AC	A	3	4
11	T	G$ACATACAG	A	4	5

M[0..11,0] contains the lexicographically ordered characters of S and M[0..11,11] equals BWT(S). The last two columns are required for the inverse transformation. offset[i] stores the number of times BWT[i] has appeared earlier in BWT(S). The last column LF[i] contains pointers used during the inverse transformation algorithm: if $S[i]$ = BWT[j], then BWT[LF[j]] = $S[i-1]$.

MUNI FI

Enhanced suffix array and BWT-based indexes

Table 1. Arrays used by enhanced suffix arrays (columns 2–5), compressed suffix arrays (columns 2, 6 and 7) and FM-indexes (columns 8 – 14) for string S = ACATACAGATG$

		ESA			CSA		FM-index 'rank'							
i	SA	LCP	*child*	*sl*	SA^{-1}	Ψ	BWT	$	A	C	G	T	LF	S[SA[i]..]
0	11	−1			2	2	G	0	0	0	1	0	8	$
1	4	0	6	[0..11]	7	6	T	0	0	0	1	1	10	ACAGATG$
2	0	3	2	[6..7]	4	7	$	1	0	0	1	1	0	ACATACAGATG$
3	6	1	4	[0..11]	10	9	C	1	0	1	1	1	6	AGATG$
4	2	1	5		1	10	C	1	0	2	1	1	7	ATACAGATG$
5	8	2	3	[10..11]	6	11	G	1	0	2	2	1	9	ATG$
6	5	0	8		3	3	A	1	1	2	2	1	1	CAGATG$
7	1	2	7	[1..5]	9	4	A	1	2	2	2	1	2	CATACAGATG$
8	10	0	10		5	0	T	1	2	2	2	2	11	G$
9	7	1	9	[0..11]	11	5	A	1	3	2	2	2	3	GATG$
10	3	0			8	1	A	1	4	2	2	2	4	TACAGATG$
11	9	1	11	[0..11]	0	8	A	1	5	2	2	2	5	TG$

From left to right: index position, suffix array, LCP array, child array, suffix link array, inverse suffix array, Ψ-array, BWT text, 'rank' array, LF-mapping array and suffixes of string S. FM-indexes also require an array $C(S)$.

MUNI
FI

Read mapping tool
Performance
- **CPU**
- **RAM**

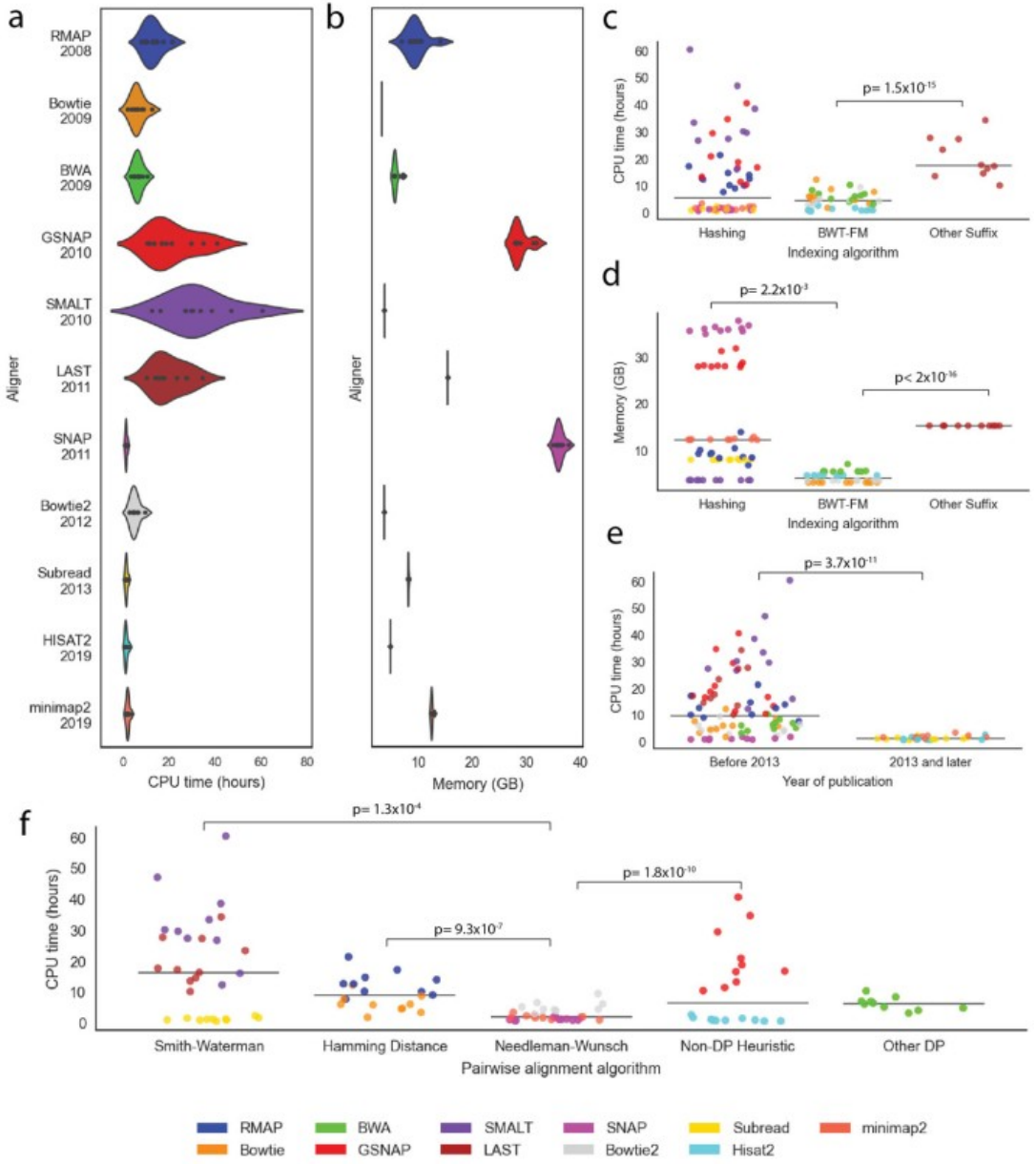
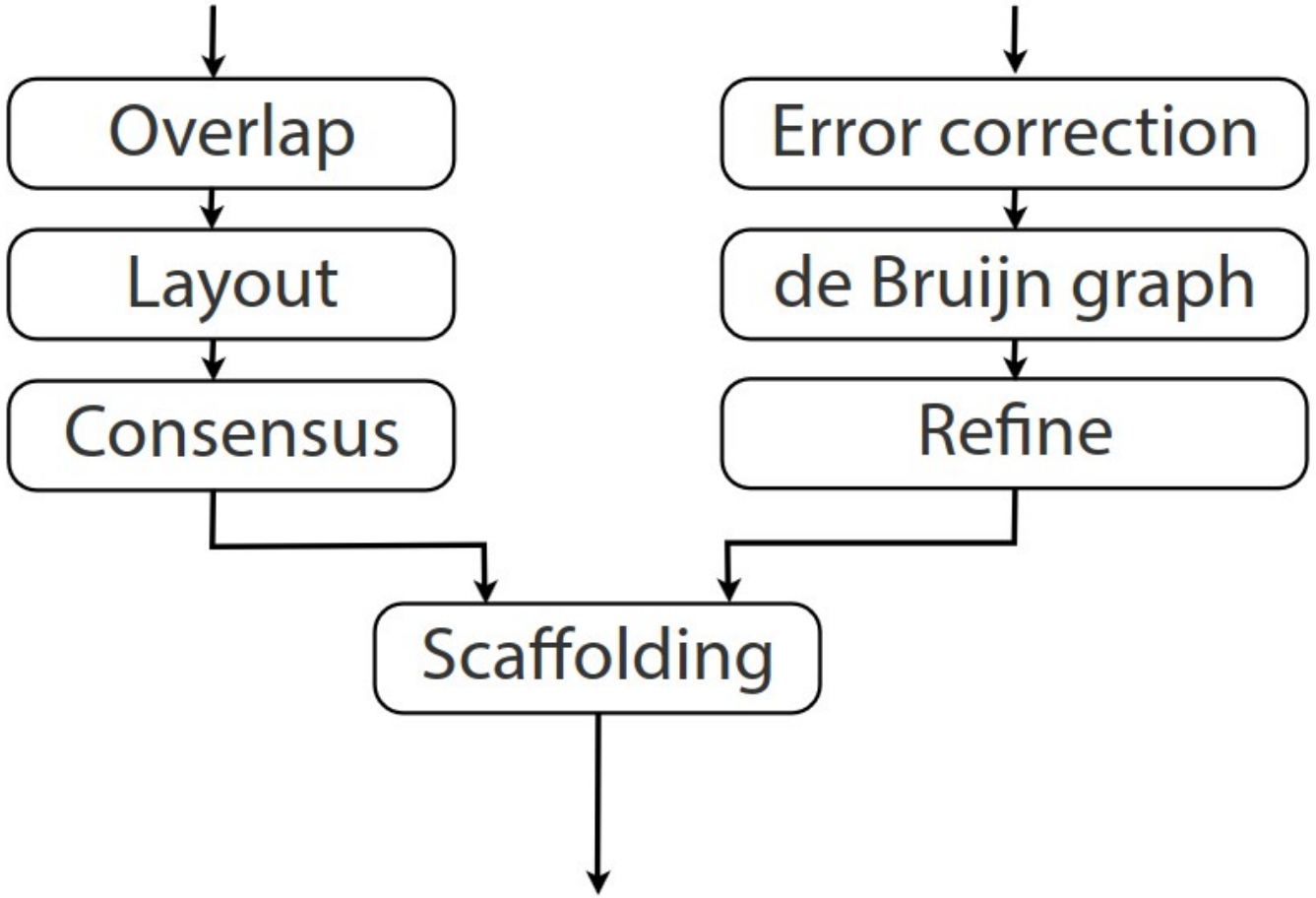


Fig. 4 The effect of read alignment algorithms on the speed of alignment and computational resources.

MUNI
FI

Assembly alternatives



Overlap graph for overlap-layout-consensus assembly

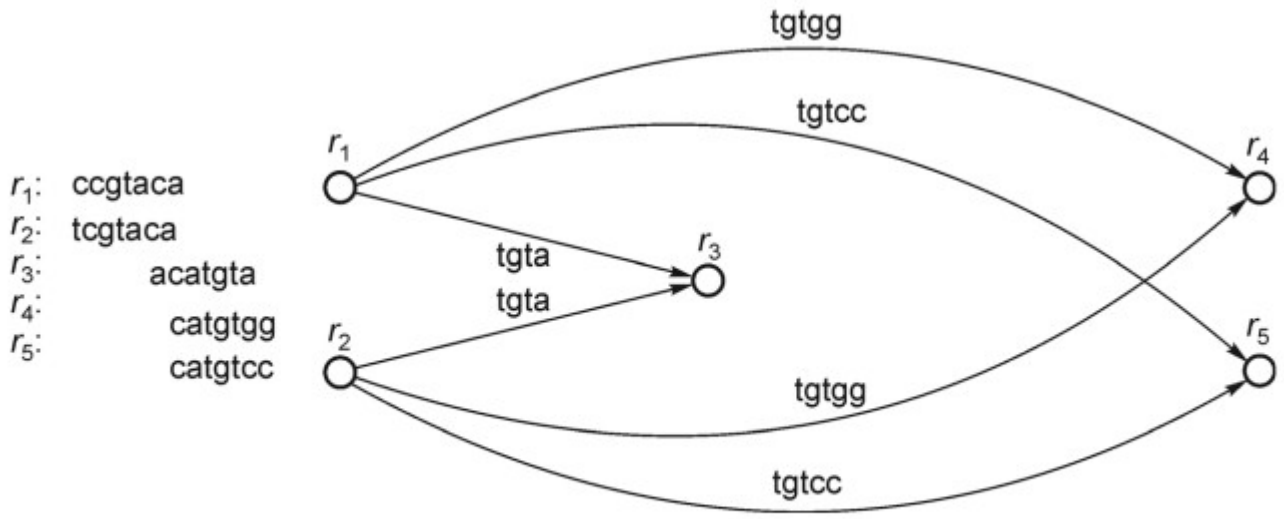


Figure 2. Example of the overlap graph for five reads r_1, r_2, r_3, r_4, r_5. Each edge (r_i, r_j) is labelled by the *extension* $e_{i,j}$.

MUNI FI

DeBruijn graph for assembly

1: ccgt
2: tcgt
3: acat
4: catg
5: cgta
6: gtac
7: taca
8: atgt
9: tgtg
10: gtgg
11: tgta
12: tgtc
13: gtcc

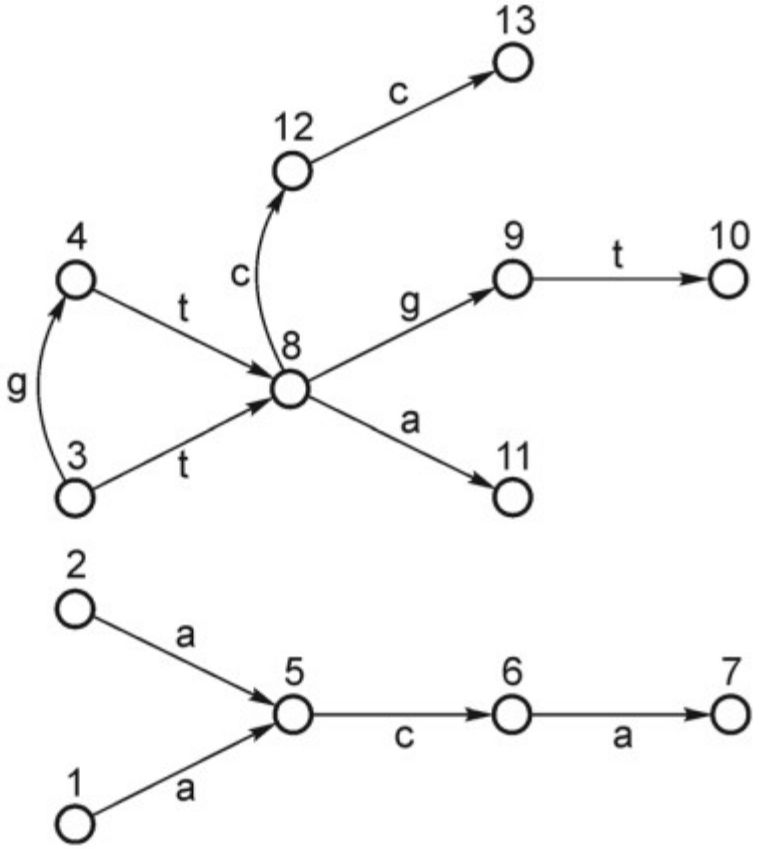


Figure 3. **Example of the *de Bruijn* graph for $k = 3$ of the two reads *ccgtac* and *catgtg*.** The nodes are the sixteen *k*-mers reported on the left. Each arc is labelled by the last character of its second node.

MUNI
FI

Tools for NGS
Read processing

MAPPING

Bowtie2, STAR, BWA-MEM

ASSEMBLY

SHORT READ

Velvet, AbySS, SOAPdenovo

LONG READ

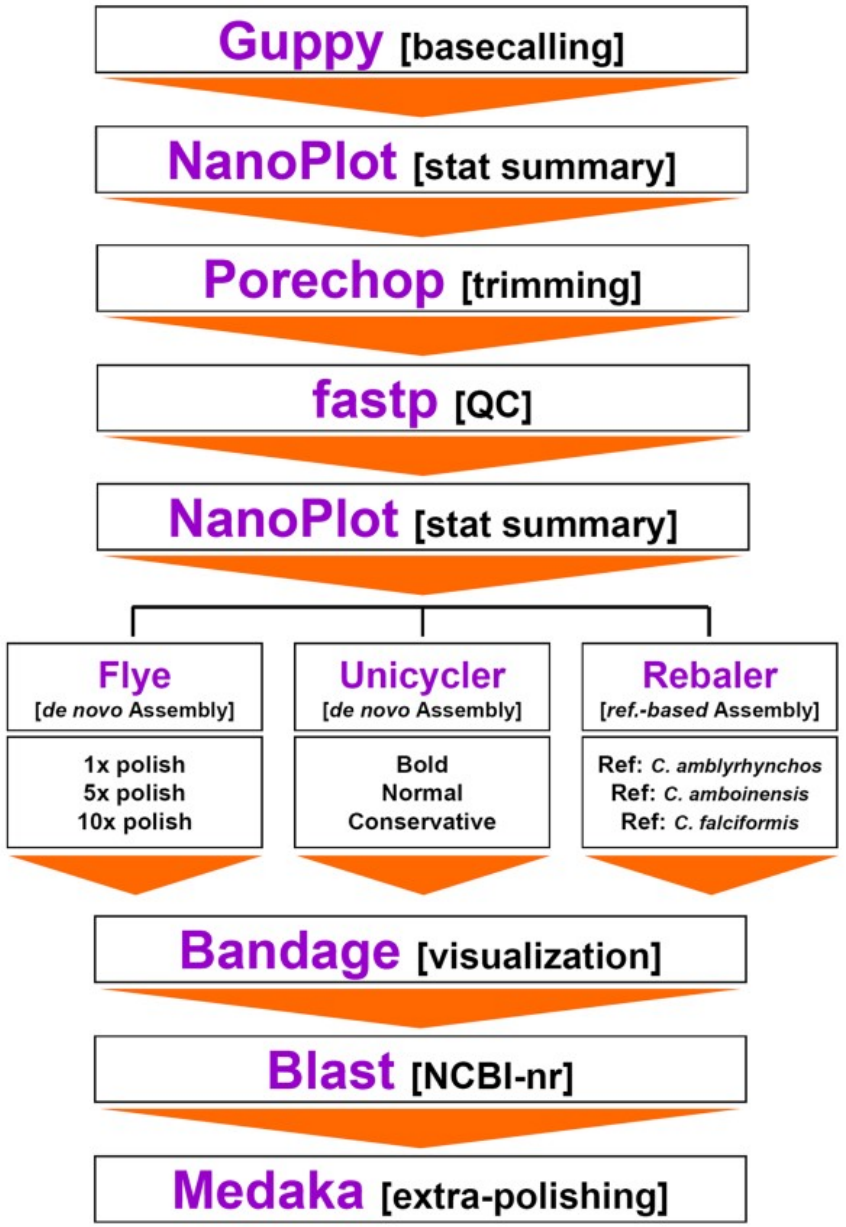
Flye, Canu, miniasm, Minipolish, NECAT,
NextDeNovo, Nextpolish, Raven, Redbean, Shasta

HYBRID

SPADes, MaSuRCA, Unicycler

MUNI
FI

Nanopore read Processing pipeline

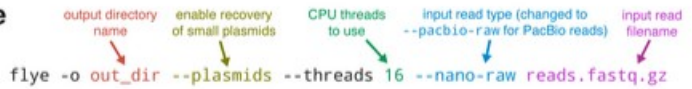


MUNI FI

Command-line examples for various nanopore assembly tools

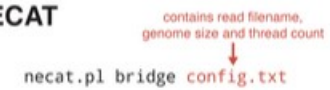
Canu

Flye

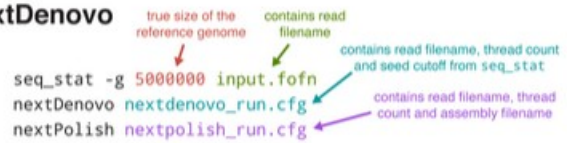


Miniasm

NECAT



NextDenovo



Raven

Redbean

Shasta

MUNI
FI

Genome assembly quality assessment

Contig N50

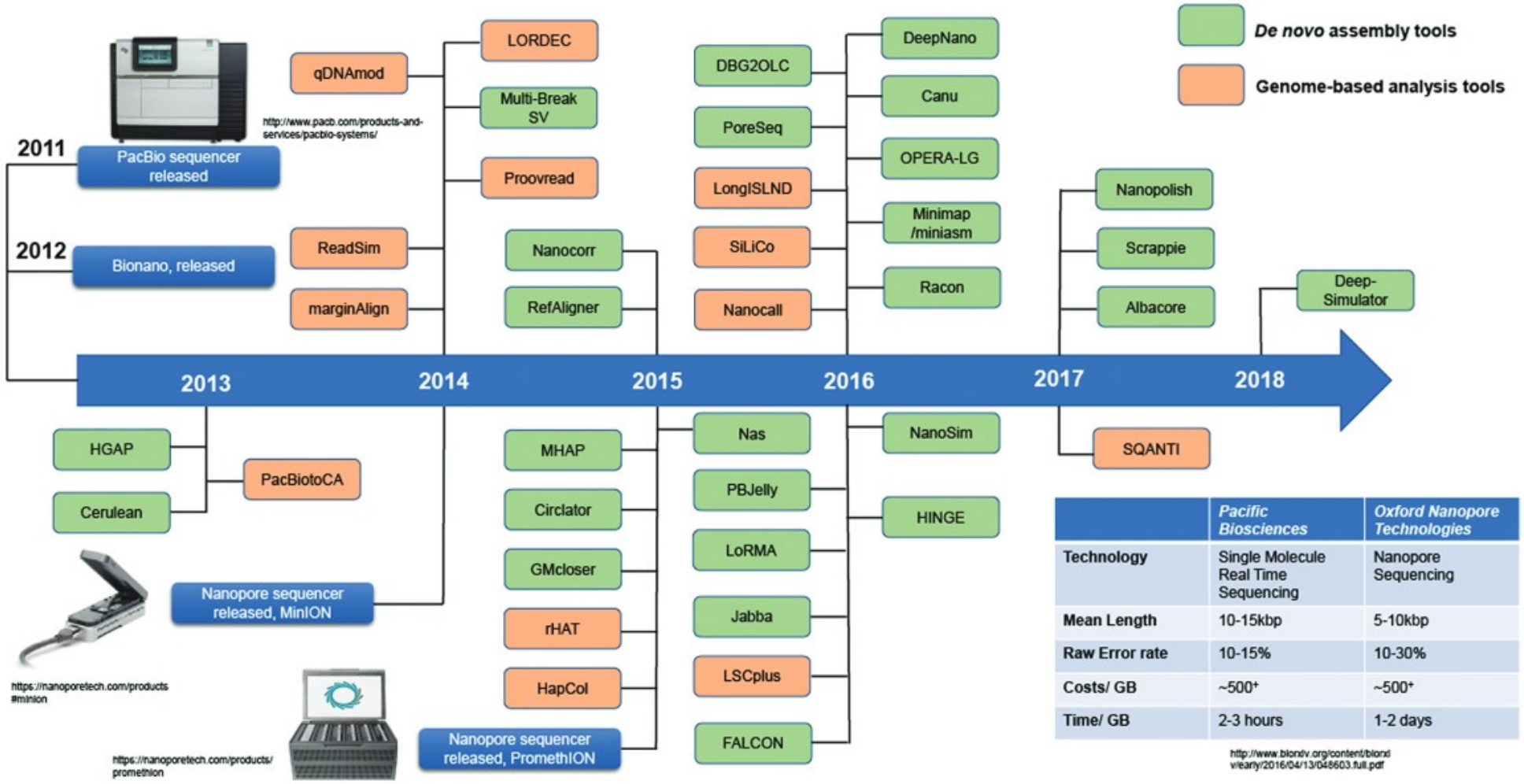
COMPASS

BUSCO score

LAI score

MUNI FI

3rd gen-seq tools



	Pacific Biosciences	Oxford Nanopore Technologies
Technology	Single Molecule Real Time Sequencing	Nanopore Sequencing
Mean Length	10-15kbp	5-10kbp
Raw Error rate	10-15%	10-30%
Costs/ GB	~500$	~500$
Time/ GB	2-3 hours	1-2 days

http://www.biomedcentral.com/content/biorxiv/early/2016/04/13/048603.full.pdf

Figure 1. Milestones in TGS analysis software development. The green box refers to the *de novo* assembly tool while the orange box refers to the genome-based analysis tool.

MUNI
FI

JBrowse

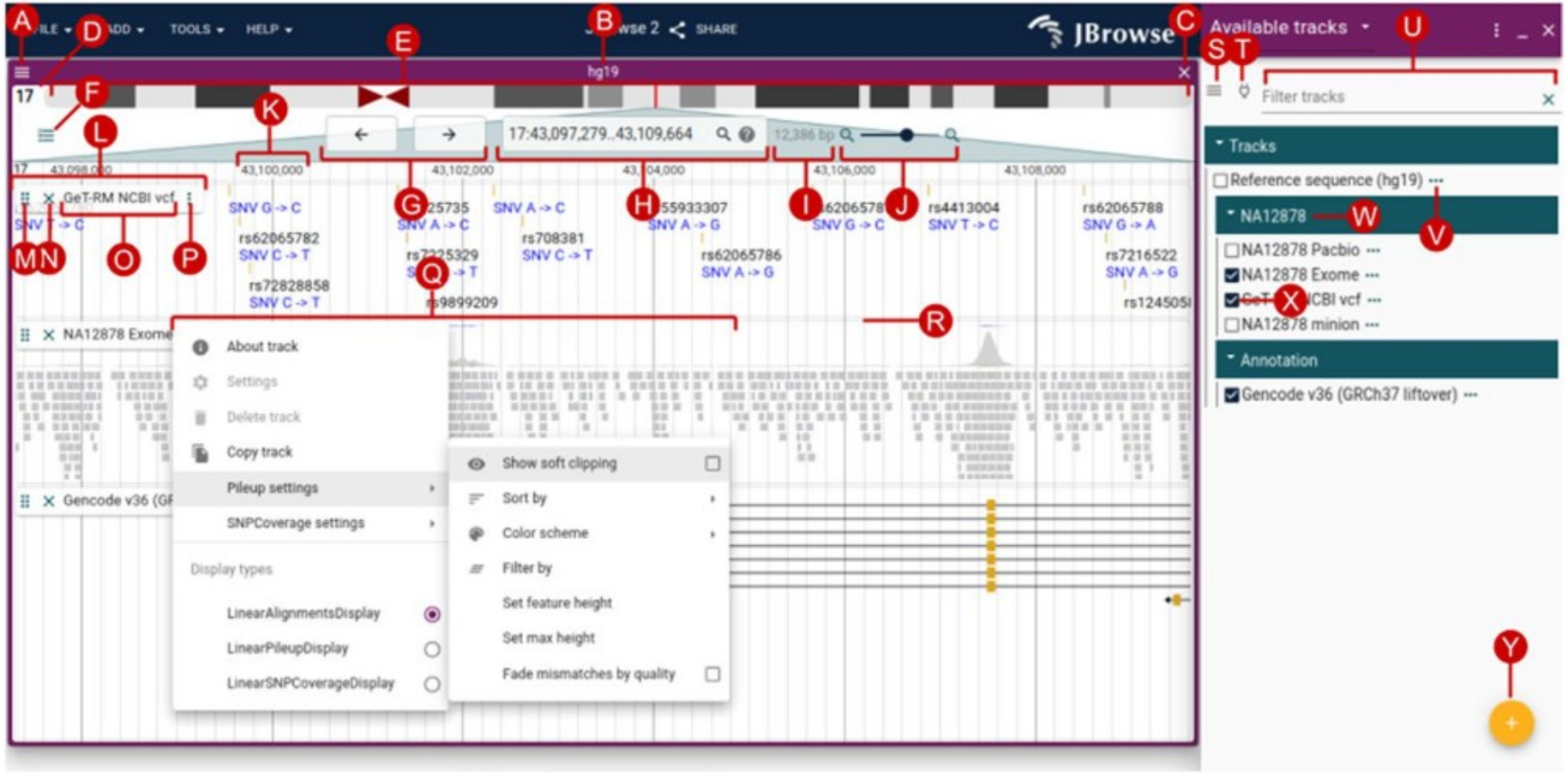


Fig. 2 The Linear Genome View is the core view of JBrowse, allowing flexible and interactive examination

MUNI
FI

JBrowse

Table 2 The list of available track types in JBrowse 2, which are specialized to render different kinds of data from various sources or file formats. Some of the tracks can be used in multiple view types as well

Track type	Appears in	Function	Supported file types
Quantitative Track	Linear Genome View	Displays dense, continuous, quantitative data	BigWig, GC content (from sequence files), GWAS scores (from BED files)
Synteny Track	Dotplot View, Linear Synteny View	Displays alignments between different genome assemblies	PAF [21],.delta from MUMmer [22], mashmap.out files [23],.chain (UCSC), MCScan.anchors files [24]
Alignments Track	Linear Genome View	Displays a combination of a pileup and a coverage visualization of alignments	BAM, CRAM
Hi-C Track	Linear Genome View	Displays Hi-C contact matrix	.hic files, generated by Juicebox [25]
Variant Track	Linear Genome View, Circular View	Displays feature glyphs corresponding to variants; specialized feature details panel show all genotypes in multi-sample VCF	VCF (plaintext or tabix)
Feature Track	Linear Genome View	Displays feature glyphs corresponding to genome annotations, e.g. genes	GTF (plaintext), GFF3 (tabix or plaintext), BigBed, BED (tabix or plaintext), features from REST APIs, etc
Reference Sequence Track	Linear Genome View	Displays a reference/assembly sequence and a three-frame translation	FASTA (indexed FASTA or bgzipped indexed FASTA), TwoBit (.2bit)

MUNI FI

JBrowse