Genome Assembly Results, Protocol & Demo

Monday, March 7, 2016

BIOL 7210: Genome Assembly Group

Aroon Chande, Cheng Chen, Alicia Francis, Alli Gombolay, Namrata Kalsi, Ellie Kim, Tyrone Lee, Wilson Martin, Tannishtha Som, Peijue Zhang

Outline

1. Introduction
2. Methods
3. Results
4. Discussion
5. Conclusion

Objectives

1. Research classic and new assembly tools
2. Evaluate and compare these available tools
3. Assemble reads and combine results into super-assembly
4. Compare results from assembly methods and choose best
5. Create efficient wrapper for pipeline of assembly methods

Introduction

Dataset

Haemophilus influenzae:

- 1.86 Mb long[4]
- 1 chromosome[4]
- 38% GC content[4]

Sequencing Data:

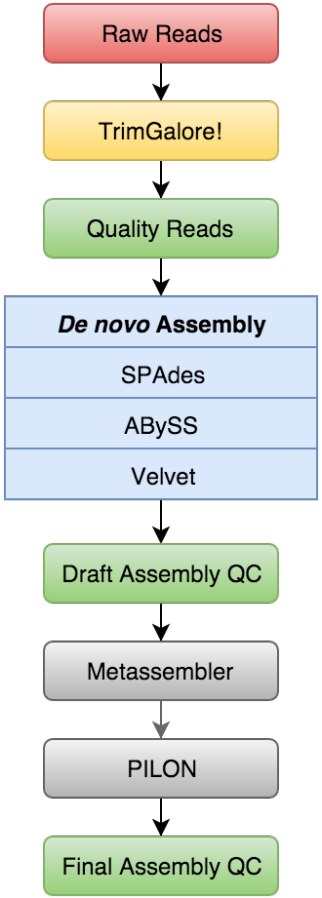
- 36 genomes from CDC
- Paired-end, 250bp reads
- From Illumina HiSeq2500
- Already demultiplexed

http://www.denniskunkel.com/DK/Bacteria/97500E.html

Strain	Year of Collection	Culture Source / Serotype	Number of Reads	Read Length (bp)	Depth of Coverage	GC %
M05964	1998	Pleural fluid / NT	1,688,806	250	207x	37.9
M07572	2000	Sinus drainage / NT	1,526,484	250	175x	38.3
M10540	2003	Ankle fluid / NT	1,530,670	250	187x	37.9
M16180	-	- / aegyptius	1,869,641	250	230x	38.1
M26026	2013	- / NT	1,282,043	250	163x	37.9
M26032	2013	Blood / NT	1,675,312	250	208x	38.0
M27986	2014	Blood / NT	1,577,274	250	197x	37.9
M27987	2014	Wound / NT	1,753,556	250	218x	37.9
M28356	2014	CSF / NT	1,351,570	250	160x	38.0
M28405	2014	Blood / NT	1,541,507	250	187x	38.0
M28687	2014	Blood / NT	1,530,756	250	191x	37.9
M28702	2014	Blood / NT	1,753,861	250	212x	38.0
M28745	2014	Blood / NT	1,549,230	250	197x	37.9
M28770	2014	Blood / NT	1,791,907	250	224x	38.0
M28801	2014	Blood / NT	1,394,019	250	164x	37.9
M28853	2014	Blood / NT	1,755,484	250	207x	38.1
M28888	2014	Brain tissue / NT	1,618,228	250	188x	38.2
M29179	2014	Blood / NT	1,569,746	250	200x	38.0
M29197	2014	Lymph node / NT	1,569,746	250	212x	38.0
M29202	2014	Blood / NT	1,468,078	250	169x	38.0
M29227	2014	Blood / NT	1,417,918	250	169x	37.9
M29307	-	- / NT	1,264,736	250	155x	37.9
M29323	2014	Blood / NT	1,350,437	250	159x	38.1
M29331	2014	Blood / NT	1,774,968	250	221x	38.0
M29400	2015	Blood / NT	1,461,989	250	173x	38.0
M29658	2015	Blood / NT	1,749,558	250	213x	38.0
M29684	2015	Sputum / NT	1,250,286	250	155x	37.9
M29695	2015	Blood / NT	1,644,928	250	203x	37.9
M29697	2015	Blood / NT	1,668,380	250	204x	38.1
M36557	2015	Sputum / NT	1,476,824	250	189x	37.9
M36564	2015	Blood / NT	1,478,552	250	188x	37.9
M36580	2015	Blood / NT	1,564,475	250	190x	37.9
M36582	2015	Blood / NT	1,699,167	250	215x	37.9
M36605	2015	Blood / NT	1,457,217	250	170x	38.0
M36606	2015	Blood / NT	1,709,123	250	215x	37.9
M37982	-	-	846,931	250	101x	37.9

Methods

Updated Assembly Pipeline



Assembly scoring

- Scores are needed to select best assemblies

- Some considerations:
  - Total Bases assembled: More is better, to a point
  - N50/L50: Fewer, larger sized contigs are preferable
  - Number of contigs: Ideally 1 contig per chromosome

Contig Weighted Score[7]:

$$\frac{\log_{10}(N50 \cdot Length)}{\#contigs}$$

L50 Weighted Score[7]:

$$\frac{\log_{10}\left(\frac{N50}{\#contigs}\right)}{\left(\frac{AssemblyLength}{ExpectedLength}\right)^2}$$

Trim Galore!

Babraham Bioinformatics

- Pre-processing: Read cleaning and quality assessment
- Cutadapt removes adapters introduced during sequencing
- Summary statistics provided by FastQC
- Usage:

```
trim_galore --illumina --clip_R1 10 --clip_R2 10 --three_prime_clip_R1 5 --three_prime_clip_R2 5

--no_report_file  --length 100 --paired <reads1 file> <reads2 file> -o <output directory>
```

KmerGenie

- Estimates best k-mer length for *de novo* assembly
- Method:
  - Computes k-mer abundance histogram for many k values
  - Then, predicts number of different genomic k-mers in dataset
  - Finally, returns the k-mer length which maximizes that number
- Single-k genome assemblers (Velvet, ABySS):
  - KmerGenie predictions can be applied to these assemblers
- Multi-k genome assemblers (i.e. SPAdes):
  - Perform better with default parameters (using multiple k values)
- Usage: ./kmergenie -k <upper bound> -l <lower bound> <reads file>

Sambamba

- High performance, fast implementation of sam/bamtools
- Supports multi-threaded and cluster-based parallelization
- Extensively uses caching to speed up IO-bound operations
    - view, sort, merge
- Not a direct drop-in replacement for sam/bamtools
    - Uses different syntax and flags
    - Does not implement all functions
- Compared to samtools, Sambamba gave a net speedup of ~6-8x

Velvet

EMBL-EBI 

- Manipulates de Bruijn graphs for *de novo* genome assembly
- Four stages of the algorithm:
    - 1. Hashing reads into kmers, 2. Graph construction, 3. Error removal (tips, bulges, erroneous connections), and 4. Resolving repeats
- VelvetOptimiser:
    - Multi-threaded Perl script for automatically optimising the three primary parameter options (K, expected coverage and coverage cutoff) for Velvet
- Usage:

```
VelvetOptimiser.pl -d <output dir> -s <start kmer> -e <end kmer> -x <step size> -f '-fastq.gz -shortPaired -separate $r1 $r2' -t <number of threads> --optFuncKmer 'n50'
```

ABySS (Assembly By Short Sequences)

- Distributed representation of de Bruijn graph
- Allows parallel computation of algorithm across a network:
    1. Generation of kmers to build distributed de Bruijn graph.

    Then, the initial contigs built after removal of read errors.

    2. Mate pair information used to extend the contigs
- Usage:

Paired end: abyss-pe name=<name> k=<kmer> in='reads1.fa reads2.fa'

Paired de Bruijn graph:

abyss-pe name=<name> K=<kmer size> k=<kmer pair span> in='reads1.fa read2.fa'

SPAdes

- Short read de Bruijn graph assembler, takes single and paired ends
- High level view of SPAdes assembly:
  - Assembly graph construction with multi-sized de Bruijn graphs and bulge resolution
  - Integration of paired-end data to determine genomic distance
  - Paired assembly graph construction
  - Contig reconstruction
- Error correction by BayesHammer
- Usage: spades.py -1 $r1 -2 $r2 -o <output dir> -k <kmer list> --careful

DISCOVAR

- Small genome *de novo* assembler and variant caller
- Input consists of Illumina reads of 250 bp or longer
- Two phases:
  1. Error correction in initial graph with graph constructed similar to ALLPATHS
  2. Optimization of graph
- Usage: DiscovarDeNovo READS=bam-file OUT-HEAD=output-file

Metassembler

- Merges and optimizes multiple *de novo* assemblies
- Combines locally best sequence from all input assemblies at each region of the genome and merges them into a final sequence
- Compression-expansion(CE) statistic used to select locally best assembly
- Ranking assemblies by N50 size from largest to smallest usually gives the best superassembly
- Usage: metassemble --conf <conf file> --out-d <output dir>

CISA (Contig Integrator for Sequence Assembly)

- Four phases:
  1. Identification of representative contigs and possible extensions
  2. Removal and splitting of contigs that may be misassembled
  3. Iterative merging of contigs with a minimum of 30% overlap
  4. Merging of contigs based on size of repetitive regions
- Usage: python CISA.py <config file>

Pilon

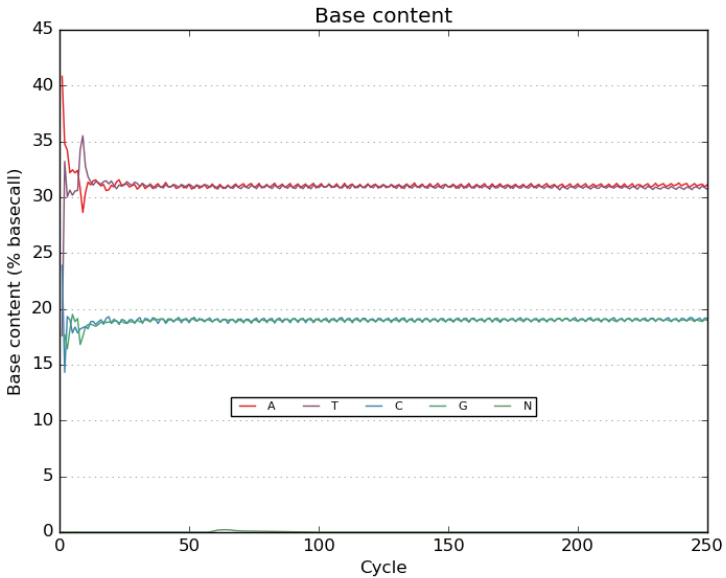
- Polishes and fixes assemblies using short read mapping
- Tries to fix individual bases and small in-deletions, ambiguous bases in fasta output, fill gaps, and try to detect and fix local misassemblies
- Required inputs:
    - .fasta (genome assembly)
    - .bam files (read files aligned to genome; map with bwa or bowtie2 and then sort and index with samtools)
- Usage:

java –Xmx15G –jar path/to/pilon-1.16.jar --genome <path/to/genome.fasta> --frags <path/to/mapping.bam> --output <sample_name> --changes --variant --tracks

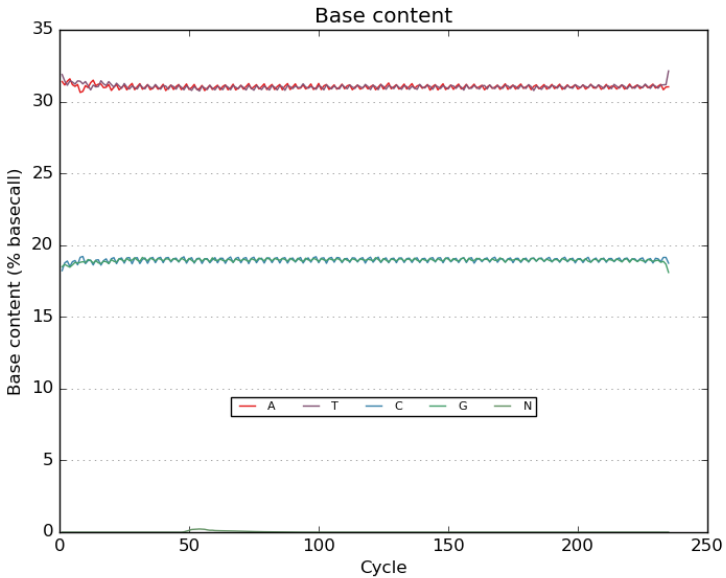
Results

Pre-processing

Before

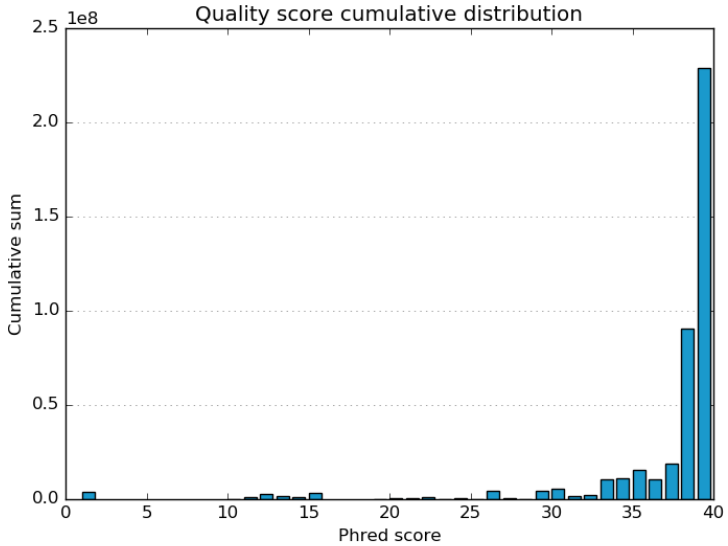


After

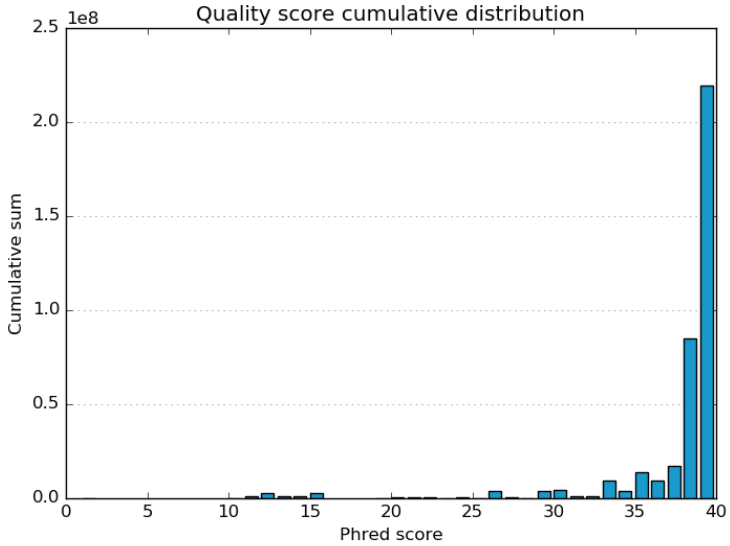


Pre-processing

Before

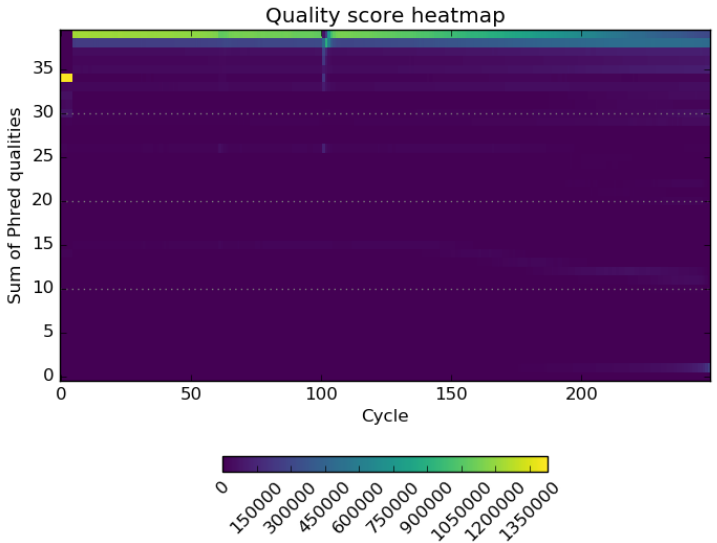


After

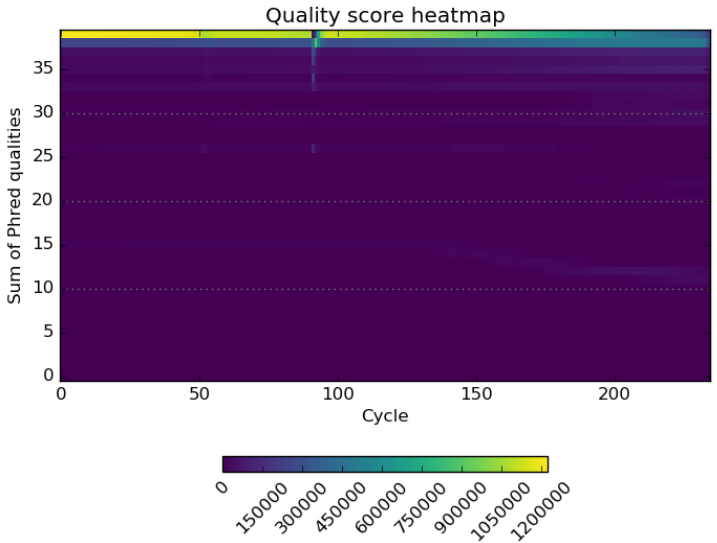


Pre-processing

Before

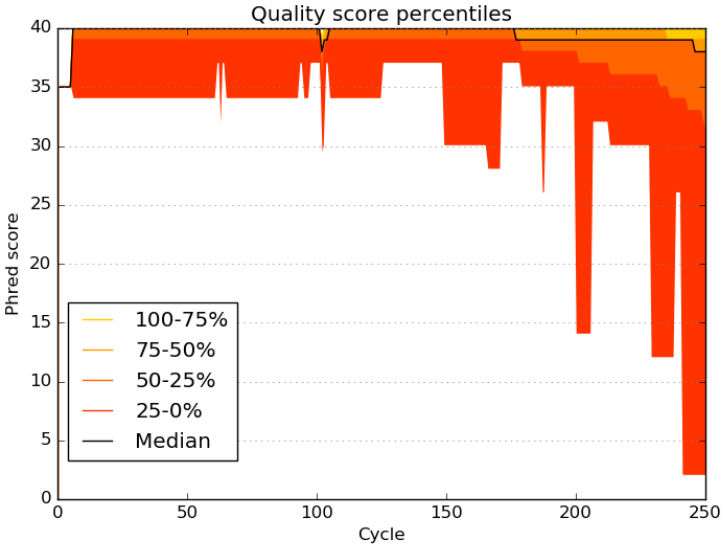


After

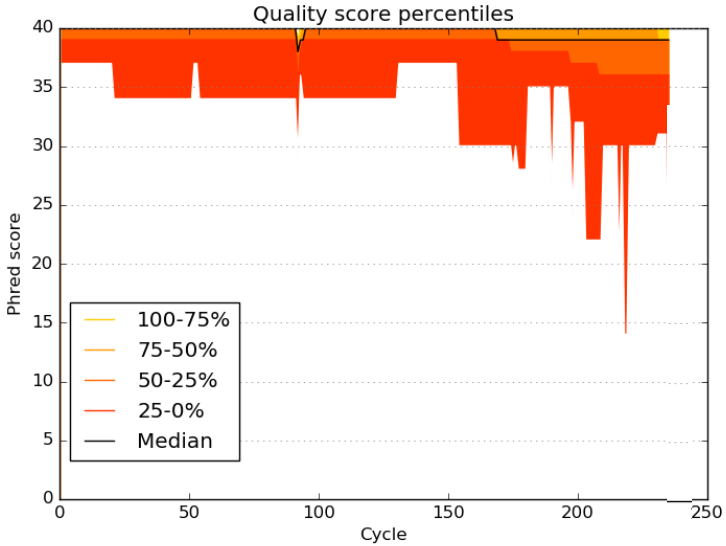


Pre-processing

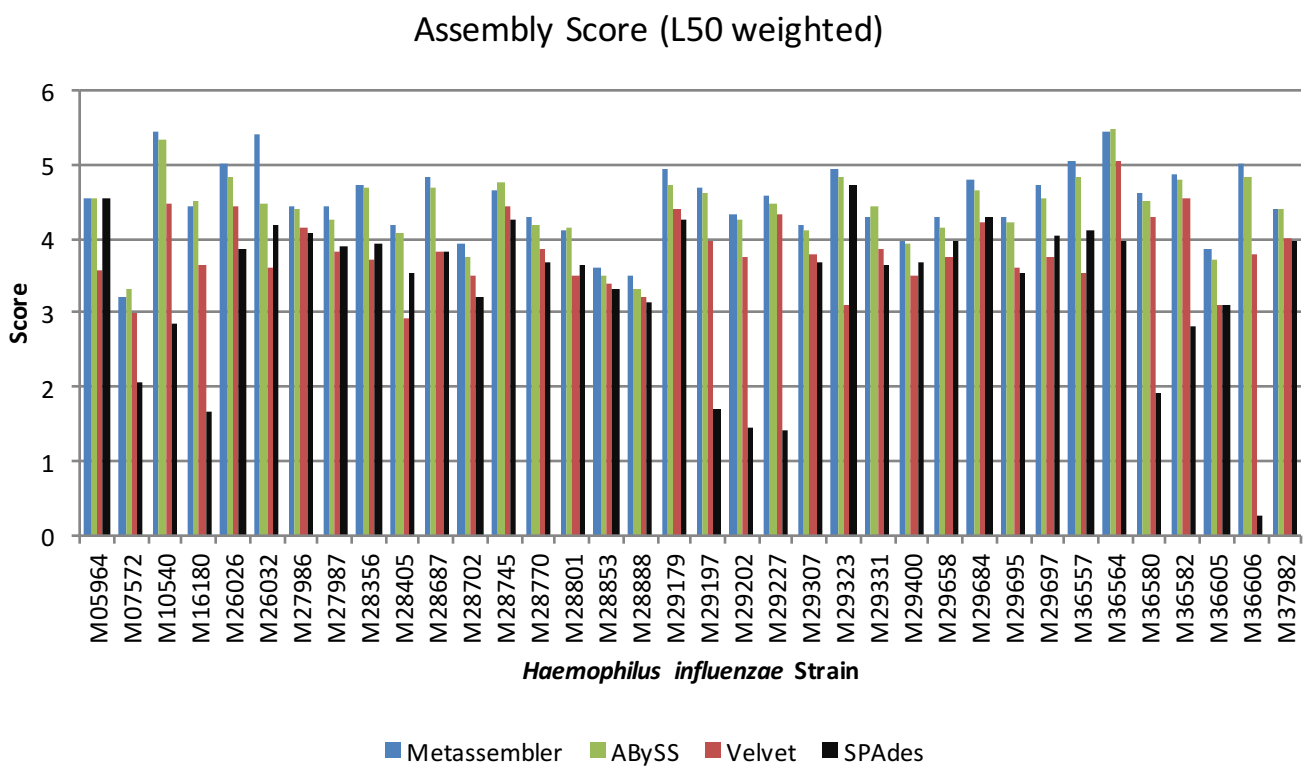
Before



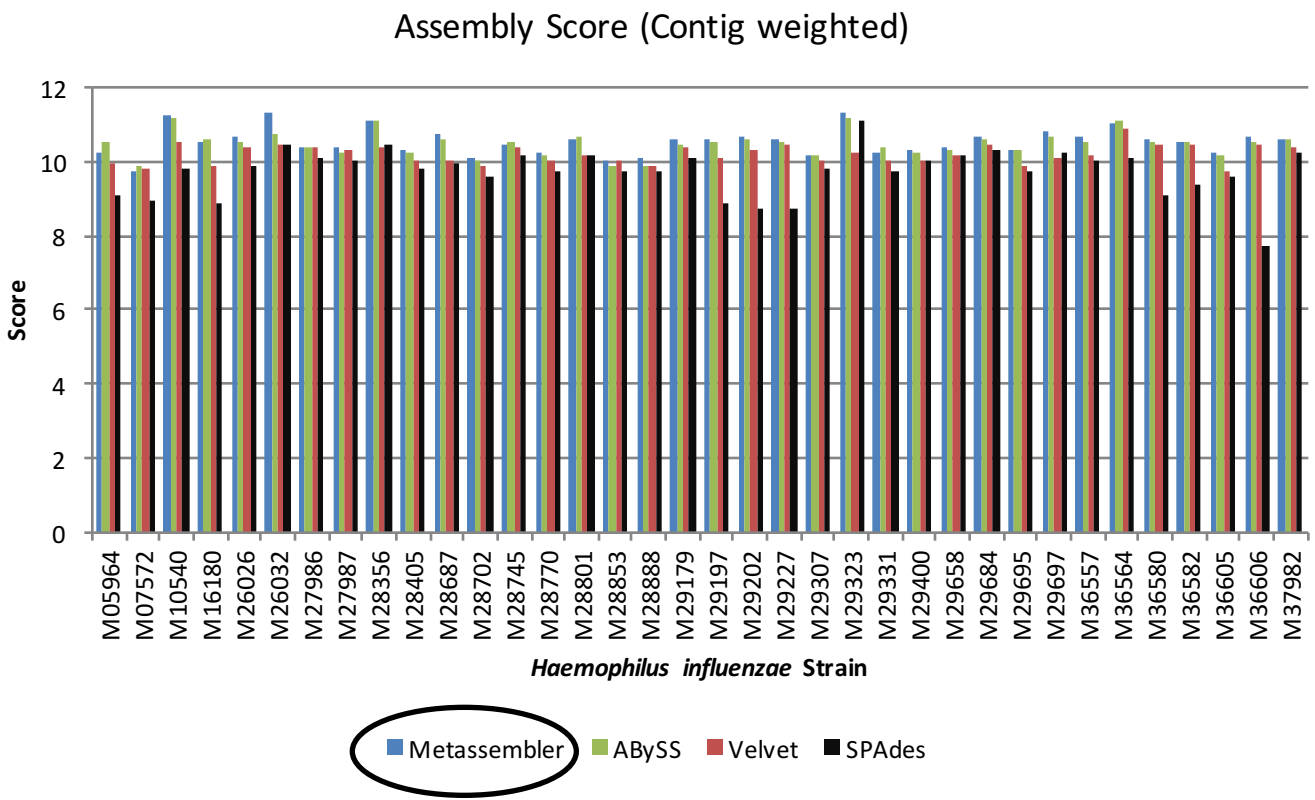
After



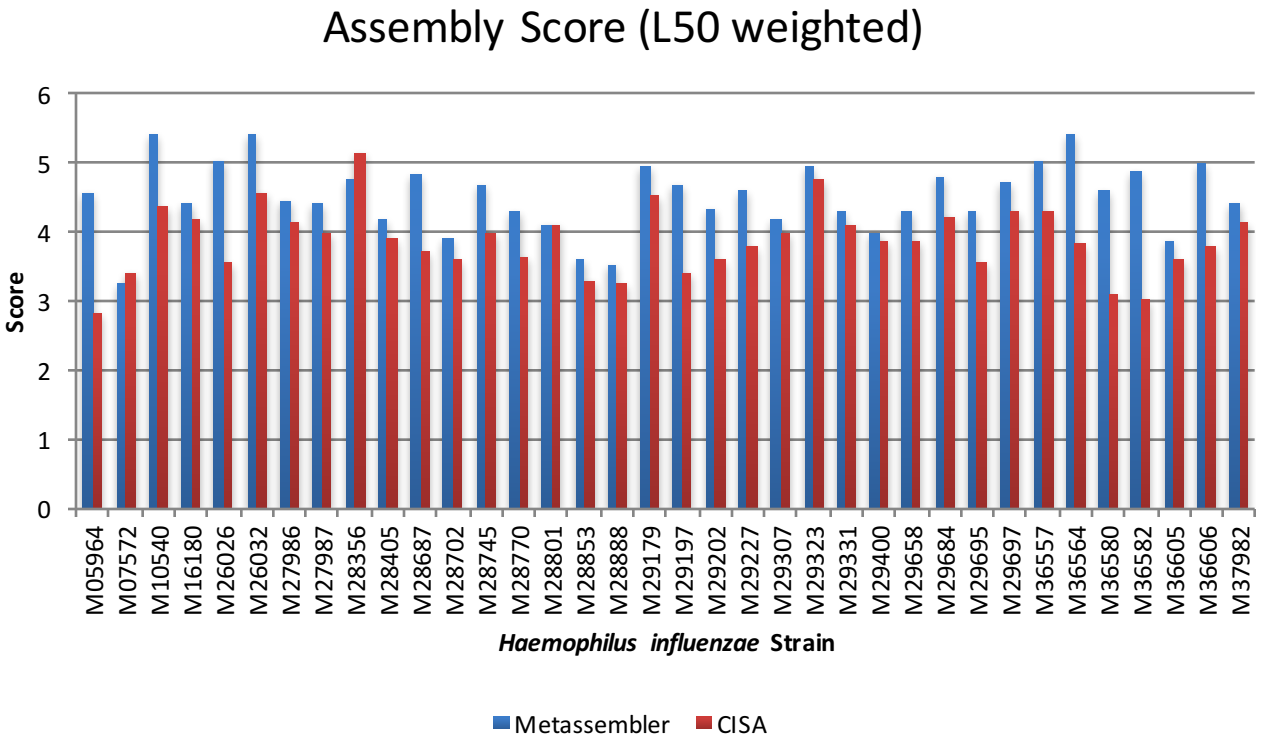
de novo Assembly Scores



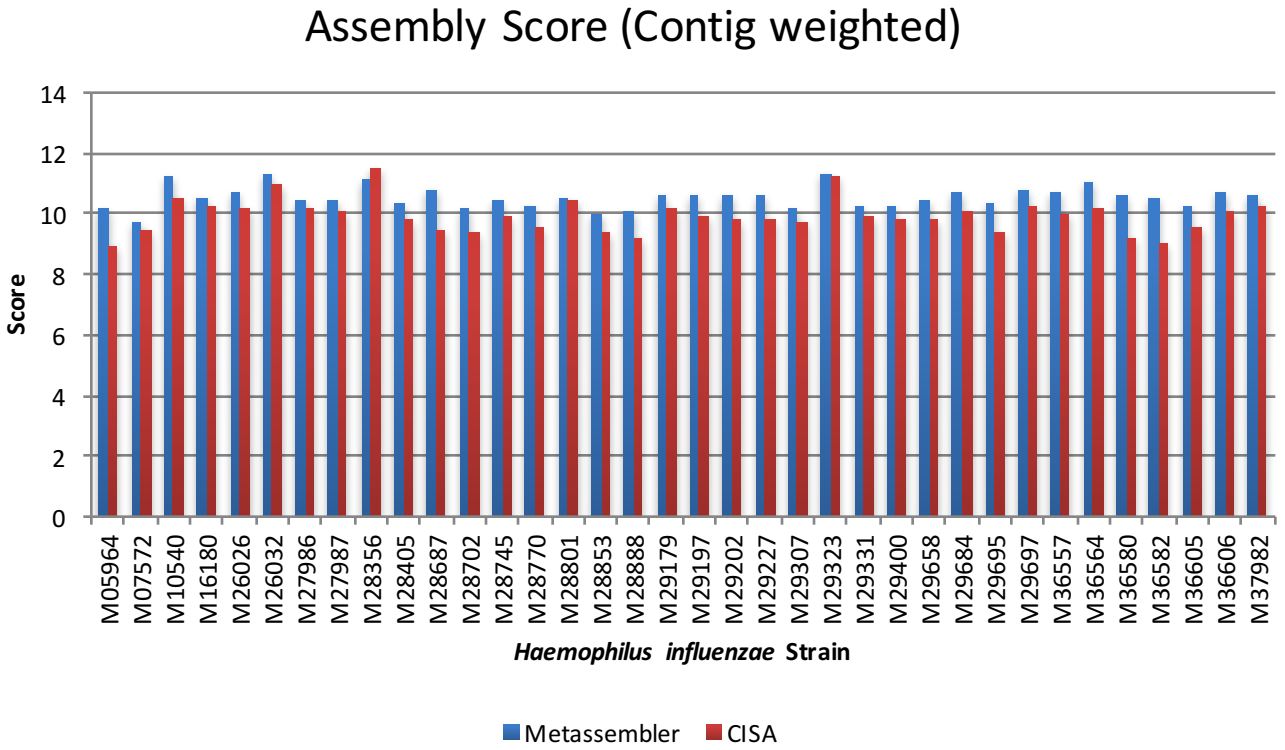
de novo Assembly Scores



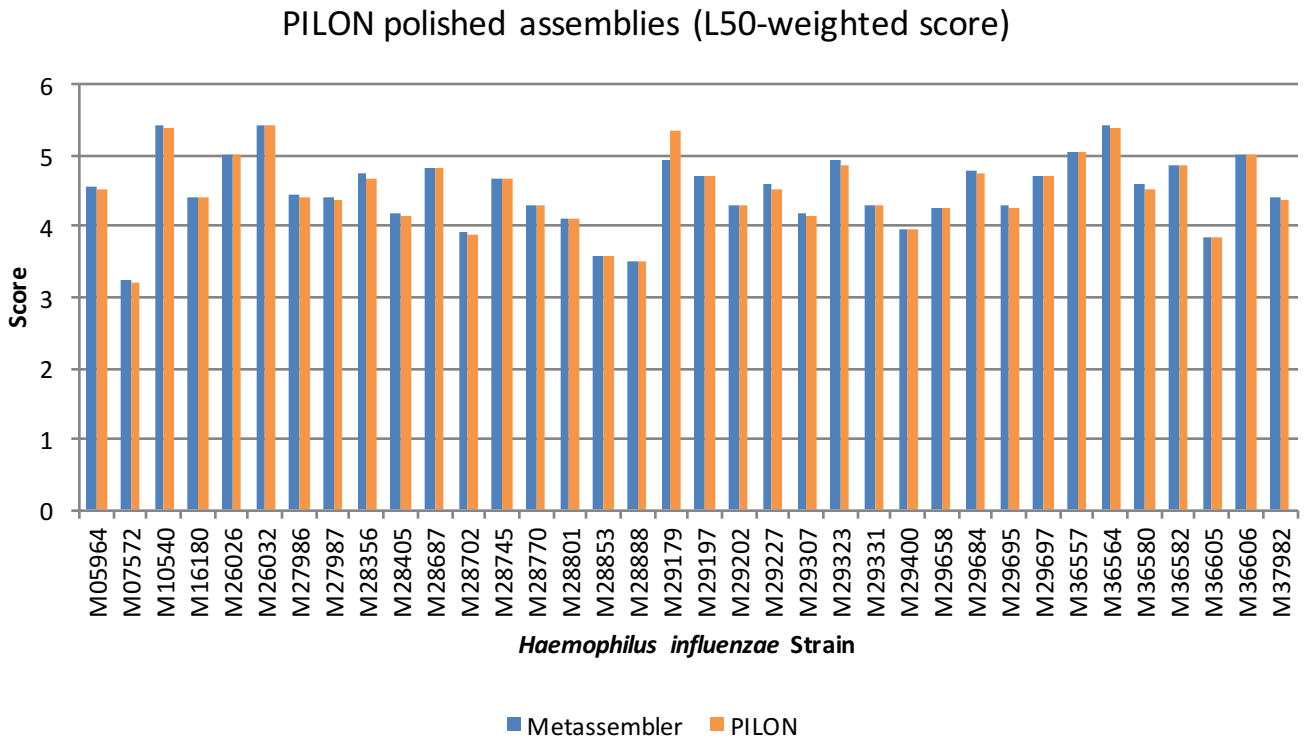
Integrated Assembly Scores



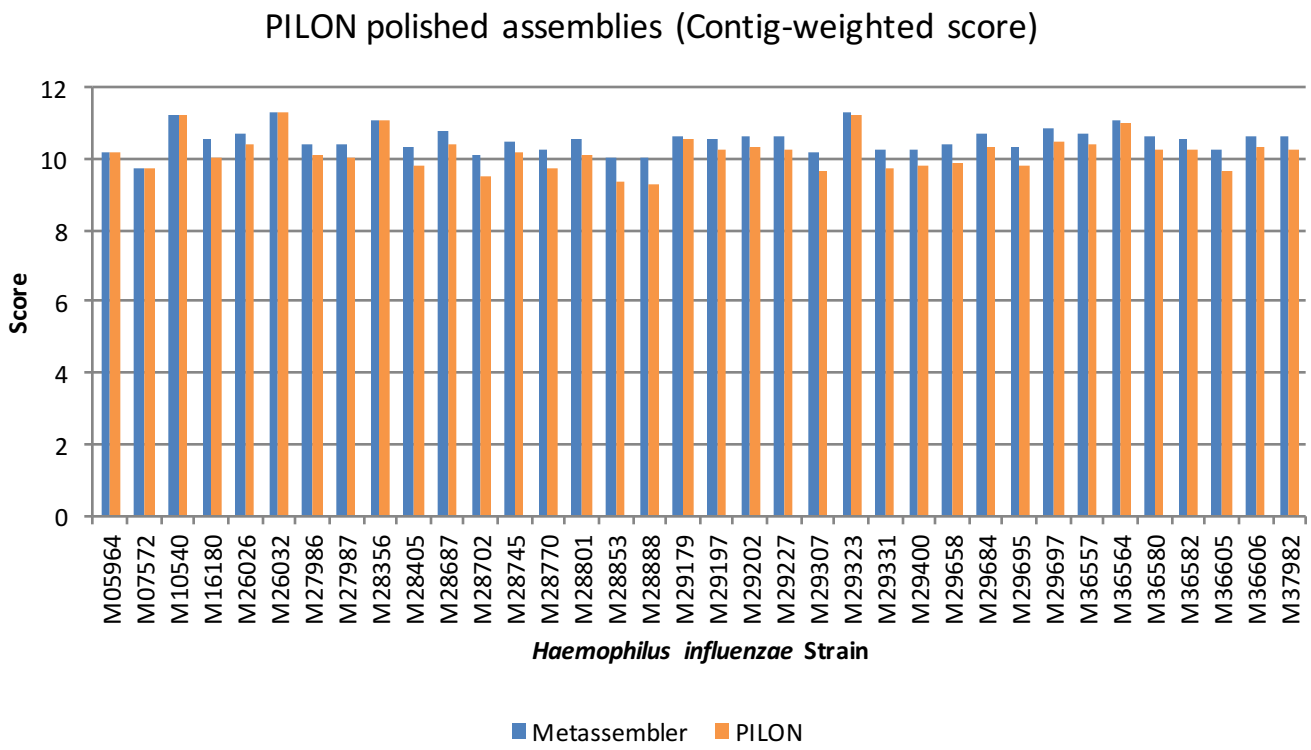
Integrated Assembly Scores



Polished Assembly Scores



Polished Assembly Scores



Final assemblies

- Finalized assemblies can be found in:
  /data/projects2/assembly/_final-assemblies
- Metassembler generated assemblies are used for all strains except **M29179** for which PILON fixed a large assembly artifact

Discussion

Range of assembly quality

	Assembly	# contigs	Total length	GC %	N50	L50	#N's/100kbp	Assembly score
Top 3	**velvet.M36564**	16	1778145	37.85	1099989	1	131.6	5.46515
	meta.M36564	17	1776313	37.85	1062004	1	24.43	5.42934
	pilon_meta.M36564	17	1776316	37.85	1062007	1	8.05	5.42932
Median	**velvet.M29307**	34	1840667	37.94	266131	3	45.69	4.10524
	abyss.97.M36582	25	1815648	38.1	151469	4	4.35	4.09863
	pilon_meta.M28801	15	1930915	37.91	283359	3	0	4.09706
Bottom 3	**discovar.M29227**	314	2203184	37.74	25786	23	9.08	1.4089
	discovar.M36606	626	2962079	46.04	7166	33	10.13	0.431039
	spades.M36606	779	3887713	45.64	10967	21	0	0.271455

Assembler Choice

1. Reference guided methods were too slow

- SMALT mapping is time and resource intensive, lose unique regions
- AlignGraph (hybrid assembly method) is bound by single thread speed

2. *De novo* assemblers (ABySS, Velvet and SPAdes) produced sane, mostly high quality assemblies

3. SPAdes sometimes fails spectacularly, rerunning improves quality ~Could it be bad random seed for dBG construction?

4. DISCOVAR sometimes gives assembly length of 1Mb to 3Mb

5. Metassembler generated better super assemblies than CISA did

- Integrates mate-pair information, trades run time for higher accuracy

Assembly Time Considerations

AlignGraph: > 5 hours

DISCOVAR: time DiscovarDeNovo READS=M05964.bam OUT_DIR=./M05964

1216.590s 4531.588s 75:6.296s total

SMALT: time ./smalt.sh 2&>/dev/null

8235.84s user 71.57s system 276% cpu 49:44.43 total

ABySS: time abyss-pe -C abyss k=137 name=M05964 in="$PWD/$r1 $PWD/$r2" j=6 &>/dev/null

1899.76s user 26.68s system 205% cpu 15:39.06 total

SPAdes: time spades.py -1 $r1 -$r2 -o ./spades -k 93,115,127 --careful &>/dev/null

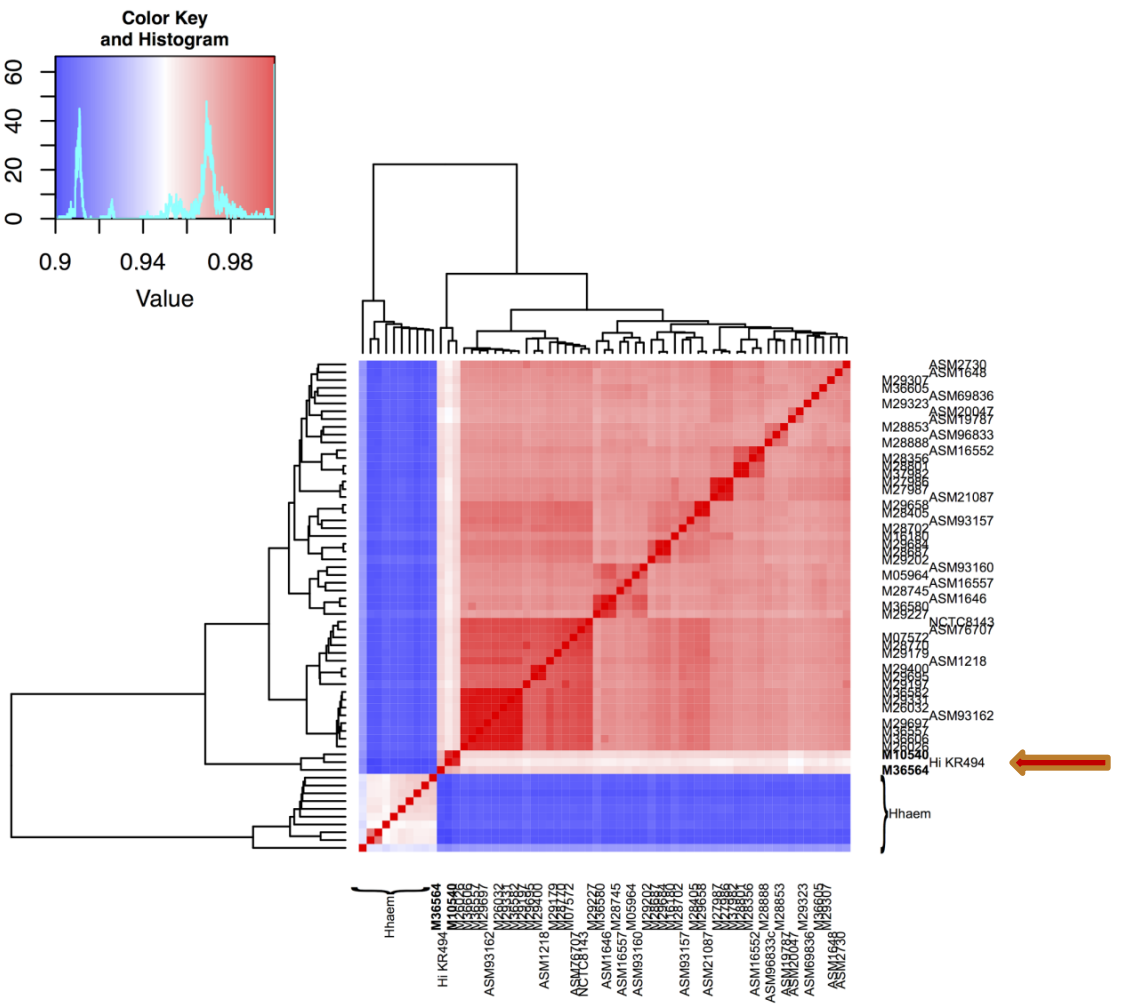
7130.26s user 142.26s system 461% cpu 26:14.43 total

Velvet: time VelvetOptimiser.pl -d ./velvet -s 97 -e 127 -x 10 -f '-fastq.gz –shortPaired -separate $r1 $r2' -t 6 --optFuncKmer 'n50' &>/dev/null

1453.15s user 22.38s system 248% cpu 9:54.59 total

ANI

- 34 samples clustered with *H.influenzae* NT references.
- 2 samples (M10540 and M36564) clustered closely with ref KR494 (H.influenzae serotype f)



Conclusion

Final pipeline

- Pipeline script found on Github:
  - https://github.com/biol7210-genomes/pipeline_scripts/

- Takes as options:
  - Assembly program(s) to run, PE reads location and output directory
  - `assembly_pipeline.pl -in $reads -o $data/assemblies --steps meta,velvet,abyss,spades`

- Gives visual progress display and writes logs to single location
  - Checkpointed for assemblers with checkpoint support

- Optional Quast quality reporting after run

Results Table

Pilon Meta 

Strain	Year of Collection	Culture Source / Serotype	Depth of Coverage	Total Bases Assembled	Number of Scaffolds	N50	L50	Largest contig	GC %	Ns/100Kbp
M05964	1998	Pleural fluid / NT	207x	1,824,203	21	371,808	2	544,736	37.87	5.48
M07572	2000	Sinus drainage / NT	175x	1,955,171	49	140,620	5	305,777	38.25	N/A
M10540	2003	Ankle fluid / NT	187x	1,810,497	12	1,150,585	1	1,150,585	37.91	N/A
M16180	-	- / aegyptius	230x	1,854,419	16	291,100	3	402,078	38.06	N/A
M26026	2013	- / NT	163x	1,776,526	14	373,925	2	526,627	37.92	2.81
M26032	2013	Blood / NT	208x	1,827,955	9	1,058,158	1	1,058,158	38.01	3.28
M27986	2014	Blood / NT	197x	1,825,083	26	361,753	2	605,950	37.92	N/A
M27987	2014	Wound / NT	218x	1,826,333	27	361,741	2	605,950	37.92	18.67
M28356	2014	CSF / NT	160x	1,906,262	15	991,456	1	991,456	37.95	1.57
M28405	2014	Blood / NT	187x	1,863,688	20	233,244	3	543,219	38.00	N/A
M28687	2014	Blood / NT	191x	1,819,709	14	431,153	2	544,261	37.94	18.9
M28702	2014	Blood / NT	212x	1,872,971	19	135,038	4	544,650	38.00	12.81
M28745	2014	Blood / NT	197x	1,790,810	23	355,451	2	558,278	37.89	N/A
M28770	2014	Blood / NT	224x	1,820,403	22	209,505	3	526,215	38.02	N/A
M28801	2014	Blood / NT	164x	1,931,069	15	283,359	3	597,434	37.91	1.55
M28853	2014	Blood / NT	207x	1,921,072	29	151,450	4	437,683	38.10	N/A
M28888	2014	Brain tissue / NT	188x	1,958,450	26	150,882	6	264,883	38.24	3.06
M29179	2014	Blood / NT	200x	1,755,418	10	403,891	2	526,684	39.50	N/A
M29197	2014	Lymph node / NT	212x	1,814,493	19	403,060	2	527,691	37.99	44.64
M29202	2014	Blood / NT	169x	1,900,446	19	433,561	2	544,673	38.03	11.05
M29227	2014	Blood / NT	169x	1,835,719	18	383,249	2	549,253	37.89	4.9
M29307	-	- / NT	155x	1,831,870	32	266,488	3	392,818	37.94	N/A
M29323	2014	Blood / NT	159x	1,908,860	10	1,066,307	1	1,066,307	38.06	N/A
M29331	2014	Blood / NT	221x	1,819,118	17	157,413	3	526,431	38.00	21.99
M29400	2015	Blood / NT	173x	1,896,011	21	211,015	3	529,588	37.98	N/A
M29658	2015	Blood / NT	213x	1,859,623	17	233,236	3	543,253	38.01	N/A
M29684	2015	Sputum / NT	155x	1,817,595	16	427,645	2	544,210	37.94	N/A
M29695	2015	Blood / NT	203x	1,837,865	18	210,681	3	538,090	37.94	N/A
M29697	2015	Blood / NT	204x	1,857,529	13	456,784	2	526,399	38.09	N/A
M36557	2015	Sputum / NT	189x	1,773,948	15	409,658	2	526,407	37.91	N/A
M36564	2015	Blood / NT	188x	1,776,313	17	1,062,004	1	1,062,004	37.85	24.43
M36580	2015	Blood / NT	190x	1,834,591	17	369,054	2	549,016	37.9	N/A
M36582	2015	Blood / NT	215x	1,774,123	19	362,272	2	526,459	37.92	N/A
M36605	2015	Blood / NT	170x	1,919,813	22	207,443	4	292,424	38.00	N/A
M36606	2015	Blood / NT	215x	1,773,452	16	404,670	2	526,368	37.92	N/A
M37982	-	-	101x	1,873,588	16	346,095	2	595,448	37.87	N/A

References

1. Yeh S. Microbiology, epidemiology and treatment of Haemophilus influenzae. <http://www.uptodate.com/contents/microbiology-epidemiology-and-treatment-of-haemophilus-influenzae>.

2. Mayer L. Overview of Haemophilus influenzae biology and genetics. <http://compgenomics2016.biology.gatech.edu/images/3/36/Lecture3-Mayer-2016.pdf>.

3. Fleishmann RD, et al. Whole-genome random sequencing and assembly of Haemophilus influenzae Rd. Science. 1995;269(5223):496-512.

4. Haemophilus influenzae. NCBI. <http://www.ncbi.nlm.nih.gov/genome/165>.

5. Krueger, F. TrimGalore! Babraham Bioinformatics. <http://www.bioinformatics.babraham.ac.uk/projects/trim_galore/>.

6. Chikhi R, Medvedev P. Informed and Automated k-Mer Size Selection for Genome Assembly. Bioinformatics. 2013;30(1):31-37. doi:10.1093/bioinformatics/btt310.

References (Continued)

7. Genome Assembly Group 2015. Evaluation of Assemblies. <http://compgenomics2015.biology.gatech.edu/index.php/Genome_Assembly_Group>.

8. Tarasov, et al. Sambamba: fast processing of NGS alignment formats. Bioinformatics. 2015;31(12):2032–2034. doi: 10.1093/bioinformatics/btv098.

9. Zerbino DR, Birney E. Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. Genome Research. 2008;18(5):821-829. doi:10.1101/gr.074492.107.

10. Simpson JT, et al. ABySS: A parallel assembler for short read sequence data. Genome Research. 2009;19(6):1117-1123. doi:10.1101/gr.089532.108.

11. Bankevich A, et al. SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing. Journal of Computational Biology. 2012;19(5):455-477. doi:10.1089/cmb.2012.0021.

12. Weisenfeld NI, et al. Comprehensive variation discovery in single human genomes. Nature genetics. 2014;46(12):1350-1355. doi:10.1038/ng.3121.

References (Continued)

13. Gurevich A, et al. QUAST: quality assessment tool for genome assemblies. Bioinformatics. 2013;29(8):1072-1075. doi:10.1093/bioinformatics/btt086.

14. Lin S-H, Liao Y-C. CISA: Contig Integrator for Sequence Assembly of Bacterial Genomes. Watson M, ed. PLoS ONE. 2013;8(3):e60843. doi:10.1371/journal.pone.0060843.

15. Wences AH, Schatz MC. Metassembler: merging and optimizing de novo genome assemblies. Genome Biology. 2015;16:207. doi:10.1186/s13059-015-0764-4.

16. Walker BJ, et al. Pilon: An Integrated Tool for Comprehensive Microbial Variant Detection and Genome Assembly Improvement. Wang J, ed. PLoS ONE. 2014;9(11):e112963. doi:10.1371/journal.pone.0112963.

17. Ponstingl H. SMALT. Sanger Institute. <http://www.sanger.ac.uk/science/tools/smalt-0>.

18. Bao E, et al. AlignGraph: algorithm for secondary de novo genome assembly guided by closely related references. Bioinformatics. 2014;30(12):i319-i328. doi:10.1093/bioinformatics/btu291.