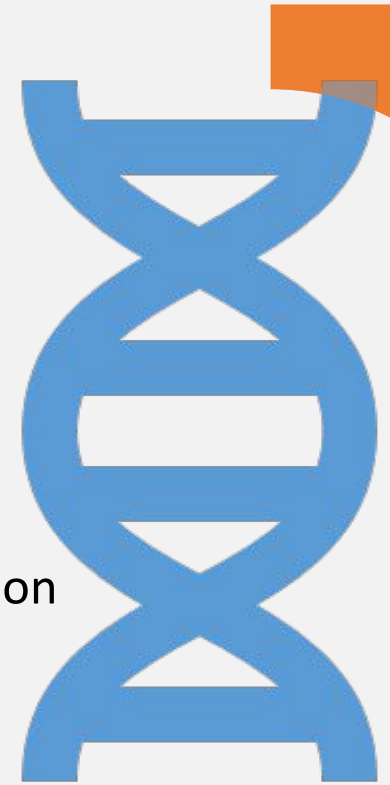
Genome Assembly Background & Strategy



Team 1 Genome Assembly

Lawrence McKinney, Laura Mora, Jessica Mulligan
Heather Patrick, Devishi Kesar, Cecilia (Hyeonjeong) Cheon

January 28, 2020

Outline

- Introduction
- Overview of Assembly Workflow
- Quality control and trimming
- De Novo Assembly and Evaluation Criteria
- Genome Assembly Tools
- Comparative Assessment
- Conclusion

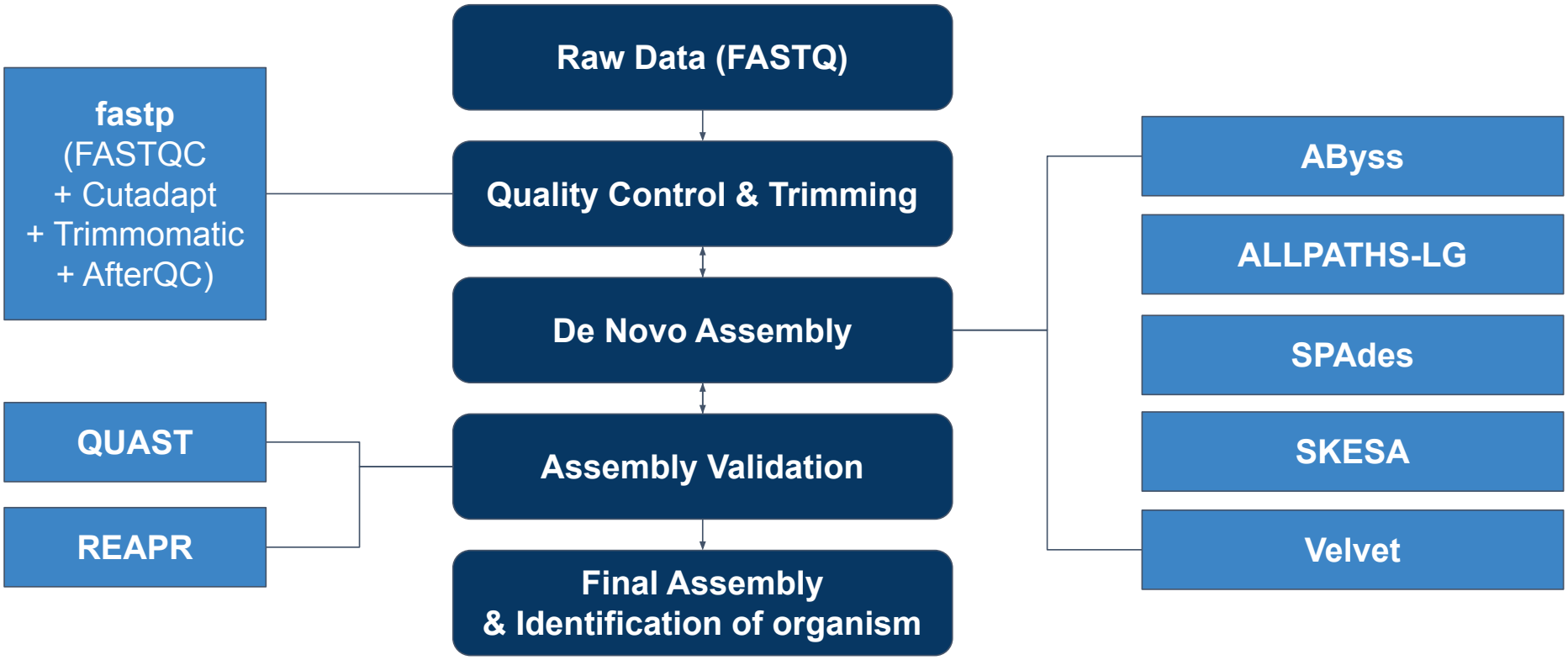
Introduction

Sequence assembly: The problem of merging and ordering shorter fragments, ‘‘reads,’’ sampled from a set of larger sequences.

Our Goals

- To perform quality control on reads before and after assembling the genome.
- To evaluate the performance of assembly tools.
  - Abyss
  - ALLPATHS-LG
  - SPADES
  - SKESA
  - Velvet
- To use the best 2 to perform de novo assembly based on the 50 isolates.
- To send off the highest quality result to gene prediction.

Genome Assembly Workflow

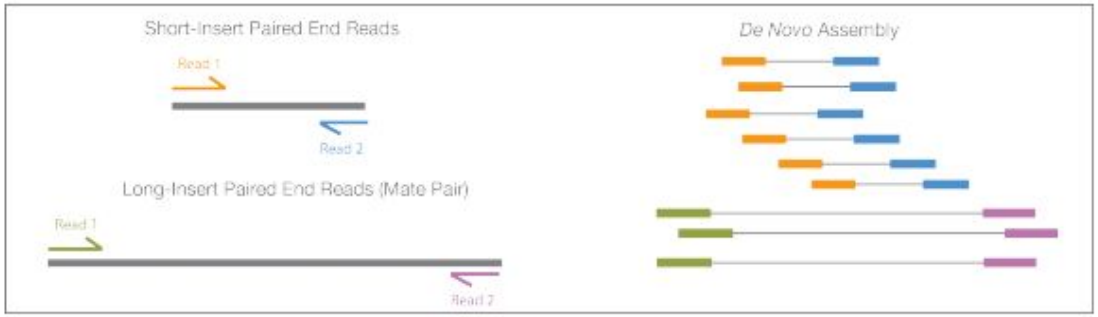


Step 1: Quality Control and Trimming

- Important to check quality of sequences.
- Using **fastp()** for quality control analysis as well as read trimming.
- **fastp:** includes most features of FASTQC + Cutadapt + Trimmomatic + AfterQC while running 2–5 times faster than any of them alone.
- Our threshold Minimum quality score: 20

Step 2: De Novo Assembly

- Most common type of genome assembly for short read sequences.
- Involves reconstructing entire genome from overlapping sequence reads.
- Quality depends on the size of the reads and number of gaps between them.
- Most tools use either of the two algorithms below:
  - de Bruijn graphs - Eulerian
  - Overlap graphs - Hamiltonian
- Can generate new, accurate reference sequences, even for complex genomes.



Rationale for Tool Selection

Based on literature that provide an unbiased assessment of pro/cons of various genome assembly tools.

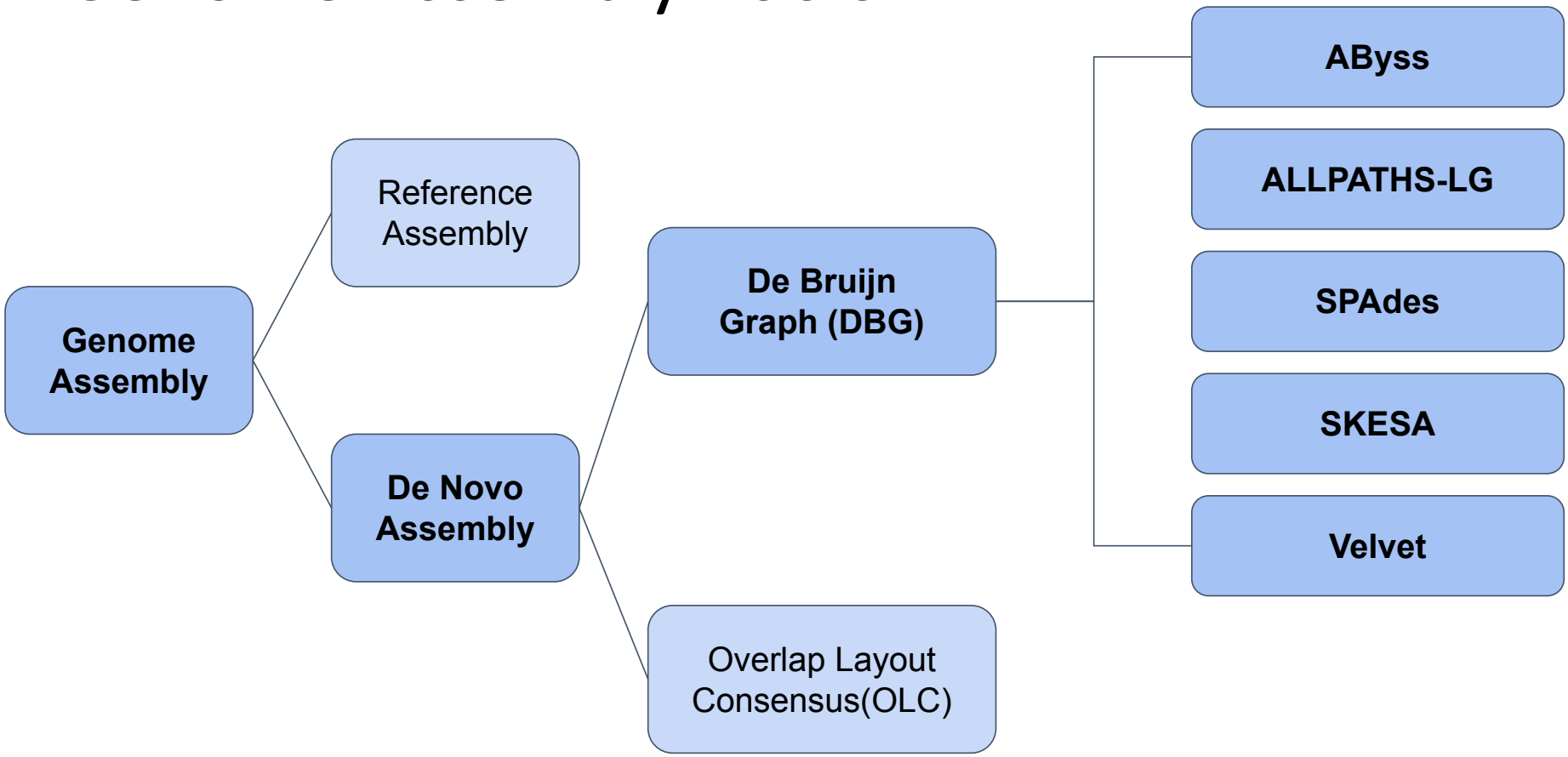
GAGE (Genome Assembly Gold-standard Evaluations)

- A study designed to provide a snapshot of how the latest genome assemblers compare on a sample of large-scale next-generation sequencing projects.
- Helps to answer questions like: *Which assembly software will produce the best results?*

Assemblathon I

- A (critical) competitive assessment of de novo short read assembly methods.
- Aims to comprehensively assess the state of the art in de novo assembly methods when applied to current sequencing technologies.

Genome Assembly Tools



Quality Metrics

Metric	Description
N50	The minimum contig length crossing the 50% threshold of the total assembled size of the genome.
NG50	The length of the scaffold at which 50% of the genome is covered.
Accuracy	The genome is considered accurate if 90% of the bases have at least 5x read coverage.
Continuity	An assembly is considered to have continuity if it's N90 > 5kb
Number of genes	The assembly which identifies most of the known genes in the organism is considered the better assembly
Validity	What fraction of the assembly (set of scaffold sequences) that can be validated by the reference sequence. Assembly cover > 90% of the actual genome is considered complete.
Scaffold statistics	Longest/shortest scaffold: typically the greater the length of largest contig, better assembly Number of scaffolds: typically an assembly with less number of scaffolds would be better
Contig statistics	Contigs may be joined into scaffolds or remain unscaffolded. This metric indicates how much of the assembly is represented by scaffolded contigs.

ABySS

- De novo assembly designed for short reads
- Parallel pair-end sequencing
- Assemble genomes up to 100 Mbases in size
- Uses a unique representation of a De Bruijn graph which distributes sequences over a cluster of computer nodes
- Performed in 2 steps:
  - Contigs are extended until they cannot be unambiguously extended further or come to a blunt end
  - Paired-end information is used to resolve ambiguities and merge contigs

Released: 2008 / Updated: 2018

Algorithm:De Bruijn graph (DBG)

Input Reads: Single-end/Paired-end

Image retrieved from: Meet the past SIB Awards Laureates – Inanc Birol

ABySS: Overall Performance

PROS	CONS
Highest performing assembler for *E.coli* based on predicted likelihood compared to Velvet and SOAP[2]	Not designed mainly for short reads
Uses a distributed k-mer hash table, making it more RAM-efficient	Produces low N50 contig and scaffold numbers
Relatively fast	

ALLPATHS-LG

- Trusts K-mers that occur at high frequency in reads
  - each base must be confirmed by a minimum number of base calls with quality value (QV)
  - Reads are restored if up to two substitutions to low-QV base calls make its K-mers trusted or if they are essential for building a path between paired-end reads.
- Resolves genomic repeats by assembling regions that are locally non-repetitive.
- Fills gaps between paired-end reads by searching the K-mer graph for instances where exactly one path satisfies the distance constraint.
- Removes erroneous errors and redundant paths as the last step.

Released: 2010 / Updated: 2014

Algorithm: De Bruijn graph (DBG)

Input Reads: Paired-end

ALLPATHS-LG

ALLPATHS-LG: Overall Performance

PROS	CONS
High contig N50 value	Memory efficiency: medium
Low incidence of chaff	Speed: low
Good error correction, with few assembly errors	
Produce good contig and scaffolds statistics	
Good trade-off between size and error rate	
Good performer in terms of correctness	

SPAdes (St. Petersburg genome assembler)

Released: 2012 / Updated: 2019
Programming language: Python
Algorithm: De Bruijn graph (DBG)
Input Reads:
Paired-end/Mate-paired/Single-end

- Was designed for small genomes
- Works with Illumina and IonTorrent reads
- Provides hybrid assemblies using PacBio, Oxford Nanopore and Sanger reads.
- Assembly is done in four stages to address sequencing errors, non-uniform coverage, insert-size variation, chimeric reads, and bi-reads:
    1. Assembly graph construction
    2. k-bimer adjustment
    3. Construct paired assembly graph
    4. Contig construction

SPAdes: Overall Performance

PROS	CONS
High contig N50 value	Generates small contigs if coverage is low
Large contigs	Mis-assemblies
Assembles high number of complete genes	
High NGA50 (QUAST value using reference genome)	

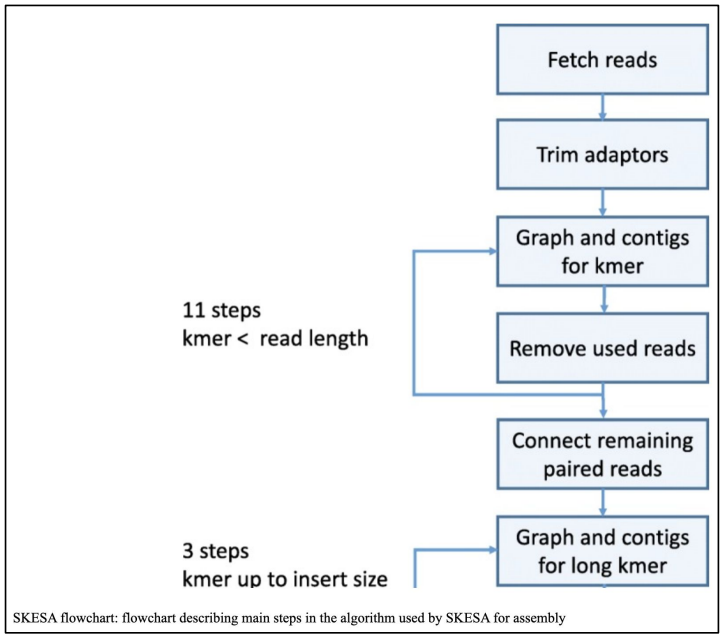
SKESA (Strategic Kmer Extension for Scrupulous Assemblies)

Released: 2018 / Updated: 2019

Algorithm: De Bruijn graph (DBG)

Input Reads: Paired-end/Single-end

- Was designed for assembling reads of microbial genomes sequenced using Illumina.
- Creates breaks at repeat regions in the genome using longer than mate-length k-mers and up to insert size.
- Quickly produces an identical assembly for the same input when assembled multiple times.
- Used for assembling over 272,000 read sets in the Sequence Read Archive at NCBI and for real-time pathogen detection.



SKESA flowchart: flowchart describing main steps in the algorithm used by SKESA for assembly

Souvorov A., (2018) SKESA: strategic k-mer extension for scrupulous assemblies. Genome Biology

SKESA: Overall Performance

PROS	CONS
High contig N50 value	Higher contiguity at longer read lengths
High sequence quality and contiguity	Does not have a built in scaffolding tool
Short assembling time (fast)	
Produces identical results regardless of the number of threads or memory	
Handles low-level contamination in reads	
Multithreaded	

Velvet

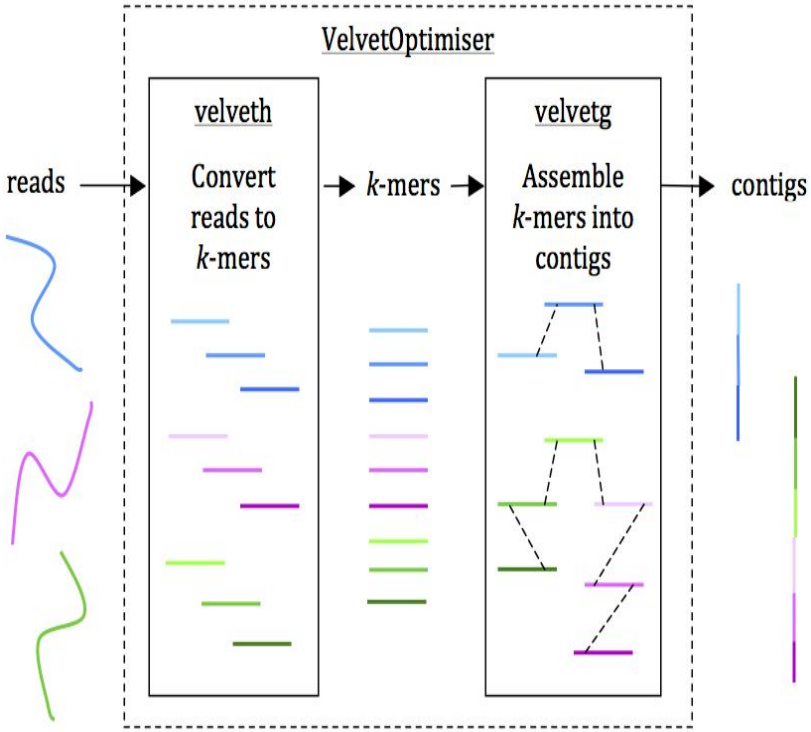
- Manipulates de Bruijn graphs for genomic sequence assembly by taking in short read sequences, remove errors, and produce high quality unique contigs.
- Uses an algorithm called *Tour Bus* for error corrections:
  - removes errors without disrupting connections within the graph
  - unique point of the genome with low coverage is not arbitrarily destroyed.

Released: 2008 / Updated: 2011

Programming language: C

Algorithm: De Bruijn graph (DBG)

Input Reads: Paired-end/Single-end



Holt, K. (2013) *Velvet*. Image retrieved from: https://holtlab.net/tag/velvet/

Velvet: Overall Performance

PROS	CONS
Great for sequences rich in repeat segments	Small N50 contig size
Automated parameter tuning for QC	Coverage cutoff excludes potentially correct low coverage vertices.
Web-based accessibility	Suitable for short reads ONLY

Image retrieved from: https://www.iconfinder.com/icons/3164312/computer_performance_seo_speed_web_web_page_web_performance_icon

Step 3: Assembly Validation

- Comparison of the tools used against some known quality metrics.
- Metrics calculated using tools like:
  - **QUAST** - evaluates genome assemblies and works both with and without a reference genome.
  - **REAPR** - a tool that evaluates the accuracy of a genome assembly using mapped paired-end reads, without the use of a reference genome for comparison.

Step 4 : Identification of organism

- Comparing the assembled contigs to available data on BLAST server to identify organism.
- The identification can help in decision making for the scientists sending in their data.
- Identify organism and strain (if relevant) of the 50 isolates.

Conclusion

- Our Objective for Genome Assembly
  - To perform quality control on reads before and after assembling the genome
  - To evaluate the performance of assembly tools.
    - Abyss
    - ALLPATHS-LG
    - SPADES
    - SKESA
    - Velvet
  - To use the best 2 to perform de novo assembly based on the 50 isolates.
  - To send off the highest quality result to gene prediction.

Thank you!

References

1. Alexey Gurevich, Vladislav Saveliev, Nikolay Vyahhi, Glenn Tesler, QUAST: quality assessment tool for genome assemblies, Bioinformatics, Volume 29, Issue 8, 15 April 2013, Pages 1072–1075, https://doi.org/10.1093/bioinformatics/btt086
2. Bankevich A, Nurk S, Antipov D, et al. SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. *J Comput Biol*. 2012;19(5):455–477. doi:10.1089/cmb.2012.0021
3. Butler, Jonathan et al. "ALLPATHS: de novo assembly of whole-genome shotgun microreads." *Genome research* vol. 18,5 (2008): 810-20. doi:10.1101/gr.7337908
4. Earl, Dent et al. "Assemblathon 1: a competitive assessment of de novo short read assembly methods." *Genome research* vol. 21,12 (2011): 2224-41. doi:10.1101/gr.126599.111
5. Maccallum, Iain et al. "ALLPATHS 2: small genomes assembled accurately and with high continuity from short paired reads." *Genome biology* vol. 10,10 (2009): R103. doi:10.1186/gb-2009-10-10-r103
6. Miller, Jason R et al. "Assembly algorithms for next-generation sequencing data." *Genomics* vol. 95,6 (2010): 315-27. doi:10.1016/j.ygeno.2010.03.001
7. Pritt, J., Chen, N. & Langmead, B. FORGe: prioritizing variants for graph genomes. Genome Biol 19, 220 (2018). https://doi.org/10.1186/s13059-018-1595-x
8. Quainoo, S., Coolen, J.P., Hijum, S.A., Huynen, M.A., Melchers, W.J., Schaik, W.V., & Wertheim, H.F. (2017). Whole-Genome Sequencing of Bacterial Pathogens: the Future of Nosocomial Outbreak Analysis. *Clinical microbiology reviews, 30 4*, 1015-1063 .
9. Rahman, A., Pachter, L. CGAL: computing genome assembly likelihoods. *Genome Biol* 14, R8 (2013). https://doi.org/10.1186/gb-2013-14-1-r8
10. Salzberg, Steven L et al. "GAGE: A critical evaluation of genome assemblies and assembly algorithms." *Genome research* vol. 22,3 (2012): 557-67. doi:10.1101/gr.131383.111
11. Shifu Chen, Yanqing Zhou, Yaru Chen, Jia Gu; fastp: an ultra-fast all-in-one FASTQ preprocessor, Bioinformatics, Volume 34, Issue 17, 1 September 2018, Pages i884–i890, https://doi.org/10.1093/bioinformatics/bty560
12. Sohn, Jang-il; Nam, Jin-Wu. "The present and future of de novo whole-genome assembly", Briefings in Bioinformatics, Vol 19.1 (2018). doi.org/10.1093/bib/bbw096
13. Souvorov A., Agarwala R., & Lipman D.J. SKESA: strategic k-mer extension for scrupulous assemblies. Genome Biology. 2018; 19(1). doi:10.1186/s13059-018-1540-z
14. Tanja Magoc, Stephan Pabinger, Stefan Canzar, Xinyue Liu, Qi Su, Daniela Puiu, Luke J. Tallon, Steven L. Salzberg, GAGE-B: an evaluation of genome assemblers for bacterial organisms, Bioinformatics, Volume 29, Issue 14, 15 July 2013, Pages 1718–1725, https://doi.org/10.1093/bioinformatics/btt273
15. Zerbino, D., & Birney, E. (n.d.). *Velvet: de novo assembly using very short reads*. Hinxton: European Bioinformatics Institute.