A step-by-step guide to sequencing and assembly of complete bacterial genomes using the Oxford Nanopore MinION

Anil Kumar [1], Max M. Häggblom [1] and Lee J. Kerkhof [2]

[1] Department of Biochemistry and Microbiology

[2] Department of Marine and Coastal Sciences

Rutgers University

New Brunswick, NJ 08901

Running title: Guide to long-read assembly of bacterial genomes

Abstract

The Oxford Nanopore (ONT) MinION enables sequencing of longer DNA/RNA fragments compared to other sequencers, such as Illumina, etc. This nanopore method provides distinct advantages for generating complete genome assemblies from microorganisms. Specifically, the R9.4 flow cells used for MinION sequencing have much lower error rates compared with earlier versions of the ONT platform. Coupled with base calling using Dorado software, higher quality long reads can now be generated for complete bacterial genome assembly. In this chapter, we describe a detailed MinION method to assemble a complete genome from a microorganism, polish the final assembly, and evaluate the genome quality using various software tools. Because of the low cost for MinION sequencing, this platform could be an asset for virtually any laboratory interested in generating complete genomes from microorganisms.

Keywords: Oxford Nanopore, bacterial genome, long read assemblers

1. Introduction

The genome represents the complete genetic material of an organism that stores information needed for its functioning, including growth and metabolism. Determining the base-pair order of the genome is vital in many research fields including microbiology, biotechnology, forensic sciences, and genetics [1]. Furthermore, DNA sequencing of short and long read amplicons can also provide information regarding the composition of the microbial community in a particular environment [2, 3]. This evaluation of microbial community composition using amplicons requires databases of curated and reference sequences/genomes [2, 4]. Having high quality genomes of representative strains also provides foundational information for elucidating the functioning of the microbes in their habitats via subsequent metagenome and metatranscriptome sequencing.

DNA sequencing has improved significantly with time since Sanger methods were introduced [5]. Currently, most nucleic acid sequencing is done by different next-generation sequencing (NGS) platforms [6]. The Illumina sequencer is probably the most widely used second-generation NGS platform because of its high throughput and low error rates. These second-generation sequencers have been followed by third-generation sequencers, such as PacBio and Oxford Nanopore, which can provide longer sequence length, albeit with higher error rates. However, sequencing longer segments of the genomes enables reconstruction of a complete gapless genome assembly [7]. In this chapter, we provide a detailed methodology for generating a complete, closed genome sequence from a microorganism using the Oxford Nanopore sequencer. Additionally, software has been developed to polish the long-read assembly by

combining Nanopore and Illumina raw reads [8, 9]. We have found that hybrid assembly methods using both long and short reads is often not necessary with the increased sequence accuracy of the R9.4 chemistry on the MinION platform.

2. Materials

2.1 DNA extraction, assembly preparation, and sequencing

1. High molecular weight DNA extraction kit
2. Rapid sequencing or ligation sequencing kit (Oxford Nanopore Technologies, UK)
3. Thermocycler
4. R-9.4 flow-cells
5. ONT MK1C or MK1B nanopore sequencer

2.2 Software Tools

1. Dorado basecaller (https://github.com/nanoporetech/dorado)
2. Flye [10]
3. Canu [11]
4. Miniasm [12]
5. Trycycler [8]
6. Raven assembler [13]
7. Racon assemblers [14]
8. Polypolish [15]
9. Medaka (https://github.com/nanoporetech/medaka)
10. Minimap2 [16]
11. Filtlong (https://github.com/ rrwick/Filtlong)
12. bwa [17]
13. POLCA [18]

14. CheckM [19]
15. BUSCO [20]
16. QUAST [21]

3. Methods

3.1 DNA extraction, assembly preparation, and sequencing

The first crucial step for generating genomes using the Nanopore sequencer is obtaining high molecular weight (HMW) genomic DNA. Certain commercially available DNA extraction kits, such as Monarch HMW DNA Extraction Kit (New England Biolabs, Ipswich, USA) and Wizard HMW DNA Extraction Kit (Promega Corporation, Madison, USA) can provide high molecular weight nucleic acids for genome generation. Additional, manual methods such as hexadecyl-trimethylammonium bromide (CTAB)-phenol-chloroform extraction [22] or phenol/chloroform extraction [23] can also generate HMW DNA. These manual methods generally require additional clean-up steps, such as cesium chloride gradient ultracentrifugation or Ampure bead (Beckmann Coulter, Brea, USA) purification prior to sequencing.

Nanopore sequence library construction is most easily accomplished using the R9.4 rapid sequencing kit (RAD-004), following the manufacturer's instructions. This is a transposase-based system which randomly inserts ONT sequencing adaptors into HMW DNA at 4-8 kb intervals. The procedure takes approximately 5 minutes and is best accomplished in a PCR thermocycler. During the library preparation, it is also possible to prime the nanopore flow cell using the flush tether and flush buffer for sequencing. Detailed methods for preparing the sequence library and running on the sequencer are given on the Oxford Nanopore website: https://nanoporetech.com/.

3.2 Basecalling

The nanopore sequencer delivers the raw reads in a fast5 or pod5 format. These raw reads must be converted into standard fastq formats using an ONT basecaller. Dorado is our current choice for basecalling and this program also performs other functions, such as removing barcode sequences and demultiplexing. The basecaller can be downloaded from (https://github.com/nanoporetech/dorado). Before running the software, the user needs to download the appropriate model based on the chemistry and the type of flow cell. For example, the recommended model for DNA sequencing using an R9.4 flow cell is dna_r9.4.1_e8_sup@v3.6. The standard code for running Dorado is given as:

- dorado basecaller dna_r9.4.1_e8_sup@v3.6 [pod5/fast5-file-path] > -r --emit-fastq >[output-file.fastq]

3.3 Assembly generation and polishing

Three long-read assemblers for whole genome assembly generation are used to compare the final assemblies generated from the assemblers.

3.3.1 Canu assembler

Canu is a long-read assembler designed for polishing and assembling PacBio and Oxford Nanopore reads [11]. The assembler can be downloaded from (https://github.com/marbl/canu), and a detailed tutorial can be accessed at (https://canu.readthedocs.io/en/latest/tutorial.html). The Linux command to run the assembler is given as:

- canu -p [assembly-prefix] -d [assembly-directory] genomeSize=[number[g|m|k] -Nanopore [*fastq]

The command will correct, trim, and assemble the whole genome from the fastq reads. The polishing of the Canu-generated assembly is done in several steps to improve the final assembly.

1. First, map the Canu-generated assembly to the raw Nanopore reads using minimap2. The command used for mapping of the reads is as follows:
    - minimap2 -ax map-ont [canu-assembly.fasta] [nanopore-reads.fastq] > [canu.sam]
2. Then, polish the Canu assembly using two rounds of Racon assembler (round 1)
    - racon [nanopore-reads.fastq] [canu.sam] [canu-assembly.fasta] -t [number] > [r1-racon-polished-canu.fasta]
3. After that, map the racon polished assembly to the Nanopore raw reads using the command:
    - minimap2 -ax map-ont [r1-racon-polished-canu.fasta] [nanopore-reads.fastq] > [r2-canu.sam]
4. Then, perform the second round of Racon polishing using the command:
    - racon [canu-assembly.fasta] [r2-canu.sam] [r1-racon-polished-canu.fasta] -t [number] > [r2_racon-polished-canu.fasta]
5. Lastly, polish the final assembly using the Medaka tool (https://github.com/nanoporetech/medaka) as follows:
    - medaka_consensus -d [r2-racon-polished-canu.fasta] -i [nanopore-reads.fastq] -t [number] -o [dir-path]

3.3.2 Flye assembler

Flye is a genome assembler for PacBio and Nanopore long reads [10]. The assembler can be downloaded from (https://github.com/fenderglass/Flye) and the user manual can be assessed from (https://github.com/fenderglass/Flye/blob/flye/docs/USAGE.md). The Linux command to run the assembler is:

- flye --nano-raw [nanopore-reads.fastq] -g [number[g|m|k] -o [dir-path] -t [number]

The polishing of the Flye generated assembly is done just like the Canu assembly polishing as follows.

1. First, map the Flye-generated assembly to the raw Nanopore reads using minimap2. The command used for mapping of the reads is as follows:
    - minimap2 -ax map-ont [flye-assembly.fasta] [nanopore-reads.fastq] > [Flye.sam]
2. Secondly, polish the Flye assembly using two rounds of Racon assembler (round 1)
    - racon [nanopore-reads.fastq] [Flye.sam] [flye-assembly.fasta] -t [number] > [r1-racon-polished-flye.fasta]
3. After that, map the Racon polished assembly to the Nanopore raw reads using the command:
    - minimap2 -ax map-ont [r1-racon-polished-flye.fasta] [nanopore-reads.fastq] > [r2-flye.sam]
4. Then, perform a second round of Racon polishing using the command:
    - racon [flye-assembly.fasta] [r2-flye.sam] [r1-racon-polished-flye.fasta] -t [number] > [r2-racon-polished-flye.fasta]
5. Lastly, polish the final assembly using the medaka tool as follows:
    - medaka_consensus -d [r2-racon-polished-flye.fasta] -i [nanopore-reads.fastq] -t [number] -o [dir-path]

3.3.3 Trycycler assembler

Trycycler is a long-read assembler effective in generating complete genome assemblies [8]. The assembler can be downloaded from (https://github.com/rrwick/Trycycler/wiki). To run

Trycycler, the user should have at least 60x read coverage. Trycycler uses different assemblers such as Flye [10], Raven [13] and Miniasm [12] for genome assembly. All the commands are provided on the Trycycler homepage. We have also indicated the commands used to run the assembler below.

1. Reads QC:
    - Filtlong --min_length 1000 --keep_percent 95 [input_reads.fastq.gz] > [reads.fastq]
2. Reads subsampling:
    - trycycler subsample --reads [reads.fastq] --out_dir [read_subsets]

    The command will subsample the reads into 12 default reads sets.
3. Assemblies: the assembly will be performed using three different assemblers on twelve reads sets as follows:
    - Flye --nano-hq [read_subsets/sample_01.fastq] --threads [number] --out-dir [assembly_01] && cp [assembly_01/assembly.fasta] [assemblies/assembly_01.fasta] && cp [assembly_01/assembly_graph.gfa] [assemblies/assembly_01.gfa] && rm -r [assembly_01]
    - miniasm_and_minipolish.sh [read_subsets/sample_02.fastq] [number] > [assemblies/assembly_02.gfa] && any2fasta [assemblies/assembly_02.gfa] > [assemblies/assembly_02.fasta]
    - raven --threads [number] --disable-checkpoints --graphical-fragment-assembly [assemblies/assembly_03.gfa] [read_subsets/sample_03.fastq] > [assemblies/assembly_03.fasta]

    Rerun all the three assemblers on the other reads set, i.e., Flye for the 4^{th}, 7^{th} and 10^{th} set, Miniasm for the 5^{th}, 8^{th} and 11^{th} set, and Raven for the 6^{th}, 9^{th} and 12^{th} set.

4. Clustering contigs: the command will cluster the assemblies into pre-replicon groups.
    - trycycler cluster --assemblies [assemblies/*.fasta] --reads [reads.fastq] --out_dir [Trycycler]
5. Reconciling contigs: The command should be run separately for all clusters that are sufficiently similar to each other as follows:
    - trycycler reconcile --reads [reads.fastq] --cluster_dir [trycycler/cluster_number]
6. Multiple sequence alignment: this command will take the reconciled contigs and run the multiple sequence alignment. The command should be run on each of the good clusters separately.
    - trycycler msa --cluster_dir [trycycler/cluster_number]
7. Partitioning reads: the command is used to partition the reads between clusters, i.e., each read is assigned to the clusters to which it aligned best.
    - trycycler partition --reads [reads.fastq] --cluster_dirs [trycycler/cluster_*]
8. Generating a consensus: this command generates the consensus contig for every cluster. The command should be run for each of the good clusters separately.
    - trycycler consensus --cluster_dir [trycycler/cluster_number]

    if the user has multiple clusters, combine the consensus sequences into one fasta file using the command:

    - cat [trycycler/cluster_*/7_final_consensus.fasta] > [trycycler/consensus.fasta]
9. Polishing using long reads: if the user only has the long reads, then polishing of the final assembly can be done using Medaka polisher using the command:
    - for c in trycycler/cluster_*; do

        medaka_consensus -i "$c"/4_reads.fastq -d "$c"/7_final_consensus.fasta -o "$c"/medaka -m r941_min_sup_g507 -t 12

```
    mv "$c"/medaka/consensus.fasta "$c"/8_medaka.fasta
    rm -r "$c"/medaka "$c"/*.fai "$c"/*.mmi  # clean up
done
```

10. Polishing using short reads: if the user has short reads of the sequenced genome, then the short reads can also be used for polishing using Polypolish and POLCA as:

    I. Using Polypolish [15]:

    - bwa index [consensus.fasta]
    - bwa mem -t [number] -a [consensus.fasta 1.fastq.gz] > [alignments_1.sam]
    - bwa mem -t [number] -a [consensus.fasta 2.fastq.gz] > [alignments_2.sam]
    - polypolish [consensus.fasta] [alignments_1.sam] [alignments_2.sam] > [polypolish.fasta]

    II. Using POLCA [18]:

    - polca.sh -a [polypolish.fasta] -r [1.fastq.gz 2.fastq.gz] -t [number] -m 1G
    - mv [*.PolcaCorrected.fa] [polypolish_polca.fasta]

The final assembly will be stored as polypolish_polca.fasta.

3.4 Checking the quality of the assembly

Different genome assemblers generate distinct assemblies of the same genome, so an assessment of the genome assembly's quality is necessary to determine the best assembly. Three different tools are used routinely to assess the quality of the genome assembly.

3.4.1 CheckM

CheckM is a standard tool for assessing the quality of a prokaryotic genome assembly constructed from isolates, single cells, or metagenomes [19]. The tool uses universal, single-

copy genes of the phylogenetic lineage and provides the completeness and contamination levels of the genome assembly. The tool can be run using the command:

- checkm lineage_wf -t [number] -x [assembly-extension] [bin-folder] [output-folder]

3.4.2 BUSCO

BUSCO is a tool to assess the quality of a prokaryotic and eukaryotic genome and metagenome assemblies [20]. The tool uses universal single-copy orthologs to assess the assembly quality and provides results in the form of completeness and contamination percentages. To run the tool, the following command is used:

- busco -i [sequence-file] -l [lineage] -o [output-name] -m [genome/transcriptome/proteins]

3.4.3 QUAST

The Quast tool is used to evaluate the genome assembly by computing various matrices such as the number of contigs, N50-value, and total genome size [21]. The tool can be used with or without a reference genome of the closest strain or species sequenced previously. The tool can be run using the following command:

- quast.py [contigs-1.fasta] [contigs-2.fasta] -r [reference.fasta.gz] -o [quast-test-output]

3.5 Comparison of the assembly programs

To test the quality of various assemblers, the bacterial genome of *Mucilaginibacter* sp. strain E4BP6 was sequenced and assembled using the methods given above. A DNA library was prepared using the rapid sequencing kit, sequenced using a R9.4 flow cell, and basecalled using Dorado. Genome assembly utilized either the Canu, Flye, or Trycycler assemblers. The assemblies generated from the Canu and Flye assembler were polished using the Nanopore-only fastq reads, while the assembly generated using the Trycycler assembler was polished

using both Nanopore and Illumina fastq reads. The unpolished and polished assemblies were screened and compared for completion and quality using QUAST (Table 1), CheckM (Table 2), and BUSCO (Table 3) tools. Comparison of the assembly quality indicated that the Trycycler generated assembly was the best, indicating hybrid approaches using short and long reads outperform other methods. However, it is worth noting that the Flye assembler was also capable of generating a complete genome sequence (one contig; 97% completeness; ~1.2% contamination) with CheckM. These results are comparable to the Trycycler generated assembly. The BUSCO results suggested the Flye generated assembly was 94.4% complete while Trycycler assembly was 98.4% complete. CheckM and BUSCO results indicate that the Trycycler generated assembly was better compared to other assemblers, but the Flye assembler also produced a good quality assembly. The Canu assembler was unable to generate the complete closed genome sequence using the tested bacterial strain.

4. Notes

1. The rapid sequencing kit works well for library preparation and takes only 5-10 mins compared to the ligation sequencing kit.
2. Use the R9.4.1, and R10.4.1 flow cells as they have low error rates.
3. The final assembly generated from the assemblers should be polished using Nanopore raw fastq reads.
4. If the user has Illumina reads, those reads can also be used for assembly polishing using the Trycycler assembler, which generally gives a better assembly.
5. Generally, Flye and Trycycler assemblers give better assembly quality with a complete, closed genome. Both methods should both be tried, and the user needs to evaluate the final assembly quality to determine the better genome assembly.

Acknowledgements

This work was funded in part by the U.S. National Science Foundation (Award Number 2129351) and the USDA National Institute of Food and Agriculture Hatch project accession number 1012785 through the New Jersey Agricultural Experiment Station (Hatch Project NJ01160).

References

1. Giani AM, Gallo GR, Gianfranceschi L, Formenti G (2020) Long walk to genomics: History and current approaches to genome sequencing and assembly. Comput. Struct. Biotechnol. J. 18:9–19

2. Kerkhof LJ, Roth PA, Deshpande S V., Bernhards RC, Liem AT, Hill JM, Häggblom MM, Webster NS, Ibironke O, Mirzoyan S, Polashock JJ, Sullivan RF (2022) A ribosomal operon database and MegaBLAST settings for strain-level resolution of microbiomes. FEMS Microbes 3:. https://doi.org/10.1093/femsmc/xtac002

3. Bolyen E, Rideout JR, Dillon MR, Bokulich NA, Abnet CC, Al-Ghalith GA, Alexander H, Alm EJ, Arumugam M, Asnicar F, Bai Y, Bisanz JE, Bittinger K, Brejnrod A, Brislawn CJ, Brown CT, Callahan BJ, Caraballo-Rodríguez AM, Chase J, Cope EK, Da Silva R, Diener C, Dorrestein PC, Douglas GM, Durall DM, Duvallet C, Edwardson CF, Ernst M, Estaki M, Fouquier J, Gauglitz JM, Gibbons SM, Gibson DL, Gonzalez A, Gorlick K, Guo J, Hillmann B, Holmes S, Holste H, Huttenhower C, Huttley GA, Janssen S, Jarmusch AK, Jiang L, Kaehler BD, Kang K Bin, Keefe CR, Keim P, Kelley ST, Knights D, Koester I, Kosciolek T, Kreps J, Langille MGI, Lee J, Ley R, Liu YX, Loftfield E, Lozupone C, Maher M, Marotz C, Martin BD, McDonald D, McIver LJ, Melnik A V., Metcalf JL, Morgan SC, Morton JT, Naimey AT, Navas-

Molina JA, Nothias LF, Orchanian SB, Pearson T, Peoples SL, Petras D, Preuss ML, Pruesse E, Rasmussen LB, Rivers A, Robeson MS, Rosenthal P, Segata N, Shaffer M, Shiffer A, Sinha R, Song SJ, Spear JR, Swafford AD, Thompson LR, Torres PJ, Trinh P, Tripathi A, Turnbaugh PJ, Ul-Hasan S, van der Hooft JJJ, Vargas F, Vázquez-Baeza Y, Vogtmann E, von Hippel M, Walters W, Wan Y, Wang M, Warren J, Weber KC, Williamson CHD, Willis AD, Xu ZZ, Zaneveld JR, Zhang Y, Zhu Q, Knight R, Caporaso JG (2019) Reproducible, interactive, scalable and extensible microbiome data science using QIIME 2. Nat. Biotechnol. 37:852–857

4. Quast C, Pruesse E, Gerken J, Schweer T, Yilmaz P, Peplies J, Glöckner FO (2015) SILVA Databases. In: Encyclopedia of Metagenomics. Springer, Boston, MA, pp 626–635

5. Sanger F, Nicklen S, Coulson AR (1977) DNA sequencing with chain-terminating inhibitors. Proc Natl Acad Sci U S A 74:5463–5467. https://doi.org/10.1073/pnas.74.12.5463

6. Heather JM, Chain B (2016) The sequence of sequencers: The history of sequencing DNA. Genomics 107:1–8

7. Mukhia S, Kumar A, Kumar R (2023) Antioxidant prodigiosin-producing cold-adapted Janthinobacterium sp. ERMR3:09 from a glacier moraine: Genomic elucidation of cold adaptation and pigment biosynthesis. Gene 857:147178. https://doi.org/10.1016/j.gene.2023.147178

8. Wick RR, Judd LM, Cerdeira LT, Hawkey J, Méric G, Vezina B, Wyres KL, Holt KE (2021) Trycycler: consensus long-read assemblies for bacterial genomes. Genome Biol 22:. https://doi.org/10.1186/s13059-021-02483-z

9. Wick RR, Judd LM, Gorrie CL, Holt KE (2017) Unicycler: Resolving bacterial

genome assemblies from short and long sequencing reads. PLoS Comput Biol 13:e1005595. https://doi.org/10.1371/journal.pcbi.1005595

10. Kolmogorov M, Yuan J, Lin Y, Pevzner PA (2019) Assembly of long, error-prone reads using repeat graphs. Nat Biotechnol 37:540–546. https://doi.org/10.1038/s41587-019-0072-8

11. Koren S, Walenz BP, Berlin K, Miller JR, Bergman NH, Phillippy AM (2017) Canu: Scalable and accurate long-read assembly via adaptive κ-mer weighting and repeat separation. Genome Res 27:722–736. https://doi.org/10.1101/gr.215087.116

12. Li H (2016) Minimap and miniasm: Fast mapping and de novo assembly for noisy long sequences. Bioinformatics 32:2103–2110. https://doi.org/10.1093/bioinformatics/btw152

13. Vaser R, Šikić M (2021) Time- and memory-efficient genome assembly with Raven. Nat Comput Sci 1:332–336. https://doi.org/10.1038/s43588-021-00073-4

14. Vaser R, Sović I, Nagarajan N, Šikić M (2017) Fast and accurate de novo genome assembly from long uncorrected reads. Genome Res 27:737–746. https://doi.org/10.1101/gr.214270.116

15. Wick RR, Holt KE (2022) Polypolish: Short-read polishing of long-read bacterial genome assemblies. PLoS Comput Biol 18:. https://doi.org/10.1371/journal.pcbi.1009802

16. Li H (2018) Minimap2: Pairwise alignment for nucleotide sequences. Bioinformatics 34:3094–3100. https://doi.org/10.1093/bioinformatics/bty191

17. Li H, Durbin R (2010) Fast and accurate long-read alignment with Burrows-Wheeler transform. Bioinformatics 26:589–595. https://doi.org/10.1093/bioinformatics/btp698

18. Zimin A V., Salzberg SL (2020) The genome polishing tool POLCA makes fast and accurate corrections in genome assemblies. PLoS Comput Biol 16:. https://doi.org/10.1371/journal.pcbi.1007981

19. Parks DH, Imelfort M, Skennerton CT, Hugenholtz P, Tyson GW (2015) CheckM: Assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes. Genome Res 25:1043–1055. https://doi.org/10.1101/gr.186072.114

20. Simão FA, Waterhouse RM, Ioannidis P, Kriventseva E V., Zdobnov EM (2015) BUSCO: Assessing genome assembly and annotation completeness with single-copy orthologs. Bioinformatics 31:3210–3212. https://doi.org/10.1093/bioinformatics/btv351

21. Gurevich A, Saveliev V, Vyahhi N, Tesler G (2013) QUAST: quality assessment tool for genome assemblies. Bioinformatics 29:1072–1075. https://doi.org/10.1093/BIOINFORMATICS/BTT086

22. Gadkari PS, Mcguinness LR, Männistö MK, Kerkhof LJ, Häggblom MM (2020) Arctic tundra soil bacterial communities active at subzero temperatures detected by stable isotope probing. FEMS Microbiol Ecol 96:. https://doi.org/10.1093/femsec/fiz192

23. Trigodet F, Lolans K, Fogarty E, Shaiber A, Morrison HG, Barreiro L, Jabri B, Eren AM (2022) High molecular weight DNA extraction strategies for long-read sequencing of complex metagenomes. Mol Ecol Resour 22:1786–1802. https://doi.org/10.1111/1755-0998.13588

Table 1: QUAST report on the unpolished and polished assemblies of *Mucilaginibacter* sp. strain E4BP6 generated from Canu, Flye, and Trycycler assemblers

Statistics without reference	**Canu-unpolished**	**Canu-polished**	**Fly-unpolished**	**Flye-polished**	**Trycycler-unpolished**	**Trycycler-polished**
# contigs	4	4	1	1	1	1
# contigs (>= 0 bp)	4	4	1	1	1	1
# contigs (>= 1000 bp)	4	4	1	1	1	1
# contigs (>= 5000 bp)	4	4	1	1	1	1
# contigs (>= 10000 bp)	4	4	1	1	1	1
# contigs (>= 25000 bp)	4	4	1	1	1	1
# contigs (>= 50000 bp)	2	2	1	1	1	1
Largest contig	4890185	4890116	4864724	4864673	4864759	4864841
Total length	5016475	5016275	4864724	4864673	4864759	4864841
Total length (>= 0 bp)	5016475	5016275	4864724	4864673	4864759	4864841
Total length (>= 1000 bp)	5016475	5016275	4864724	4864673	4864759	4864841

Total length (>= 5000 bp)	5016475	5016275	4864724	4864673	4864759	4864841
Total length (>= 10000 bp)	5016475	5016275	4864724	4864673	4864759	4864841
Total length (>= 25000 bp)	5016475	5016275	4864724	4864673	4864759	4864841
Total length (>= 50000 bp)	4951474	4951315	4864724	4864673	4864759	4864841
N50	4890185	4890116	4864724	4864673	4864759	4864841
N90	4890185	4890116	4864724	4864673	4864759	4864841
auN	4768245	4768298	4864724	4864673	4864759	4864841
L50	1	1	1	1	1	1
L90	1	1	1	1	1	1
GC (%)	40.02	40.02	39.99	39.99	39.99	39.99
Mismatches						
# N's per 100 kbp	0	0	0	0	0	0
# N's	0	0	0	0	0	0

Worst Median Best

Table 2: CheckM report on the unpolished and polished assemblies of *Mucilaginibacter* sp. strain E4BP6 generated from Canu, Flye and Trycycler assemblers. The table shows the completeness and contamination percentage of each assembly

Bin Id	**Marker lineage**	**genomes**	**markers**	**Marker sets**	**0**	**1**	**2**	**3**	**4**	**5+**	**Completeness**	**Contamination**	**Strain heterogeneity**
Trycycler unpolished	p__Bacteroidetes (UID2605)	350	316	210	5	308	3	0	0	0	97.62	1.19	0
Trycycler polished	p__Bacteroidetes (UID2605)	350	316	210	5	308	3	0	0	0	97.62	1.19	0
Flye unpolished	p__Bacteroidetes (UID2605)	350	316	210	7	306	3	0	0	0	96.9	1.19	0
Flye polished	p__Bacteroidetes (UID2605)	350	316	210	5	308	3	0	0	0	97.62	1.19	0
Canu unpolished	p__Bacteroidetes (UID2605)	350	316	210	6	304	5	1	0	0	97.38	3.1	62.5
Canu polished	p__Bacteroidetes (UID2605)	350	316	210	5	305	5	1	0	0	97.62	3.1	62.5

Table 3: BUSCO report on the unpolished and polished assemblies of *Mucilaginibacter* sp. strain E4BP6 generated from Canu, Flye and Trycycler assemblers. The table shows the completeness and contamination percentage of each assembly

Input file	Dataset	Complete	Single	Duplicated	Fragmented	Missing	n_markers	Scaffold N50	Contigs N50	Percent gaps	Number of scaffolds
Trycycler unpolished	bacteria_odb10	96.8	96.8	0	3.2	0	124	4864759	4864759	0.00%	1
Trycycler polished	bacteria_odb10	98.4	98.4	0	1.6	0	124	4864841	4864841	0.00%	1
Fly unpolished	bacteria_odb10	94.4	94.4	0	4.8	0.8	124	4864724	4864724	0.00%	1
Flye polished	bacteria_odb10	94.4	94.4	0	4.8	0.8	124	4864673	4864673	0.00%	1
Canu unpolished	bacteria_odb10	94.4	94.4	0	5.6	0	124	4890185	4890185	0.00%	4

Canu polished	bacteria_odb10	94.4	94.4	0	4.8	0.8	124	4890116	4890116	0.00%	4