SOFTWARE ARTICLE

CulebrONT: a streamlined long reads multi-assembler pipeline for prokaryotic and eukaryotic genomes

Julie Orjuela[1,2,3*†], Aurore Comte[2,3†], Sébastien Ravel[2,3], Florian Charriat[2,3], Tram Vi[2,4], François Sabot[1,3] and Sébastien Cunnac[2]

*Correspondence: julie.orjuela@ird.fr
[1]DIADE Unit, University of Montpellier, CIRAD, IRD, 911 Avenue Agropolis, F-34394 , Montpellier Cedex 5, France
[2]PHIM Plant Health Institute, Univ Montpellier, CIRAD, INRAE, Institut Agro, IRD, Montpellier, France
[3]IFB - South Green Bioinformatics Platform, Bioversity, CIRAD, INRAE, IRD, F-34398 Montpellier, France
Full list of author information is available at the end of the article
†Equal contributor

Abstract

Background: Using long reads provides higher contiguity and better genome assemblies. However, producing such high quality sequences from raw reads requires to chain a growing set of tools, and determining the best workflow is a complex task.

Results: To tackle this challenge, we developed CulebrONT, an open-source, scalable, modular and traceable Snakemake pipeline for assembling long reads data. CulebrONT enables to perform tests on multiple samples and multiple long reads assemblers in parallel, and can optionally perform, downstream circularization and polishing. It further provides a range of assembly quality metrics summarized in a final user-friendly report.

Conclusions: CulebrONT alleviates the difficulties of assembly pipelines development, and allow users to identify the best assembly options.

Keywords: long reads; assembly; QC; reproducibility; snakemake

Background

Third-generation sequencing technologies, namely Pacific Biosciences (PB) and Oxford Nanopore Technologies (ONT), provide reads up to 25kb in length, and even hundreds of thousands of bases for ONT. They can extend over repeats or structural variants, and thus result in higher contiguity and accuracy of genome assembly. Due to its low price, speed, portability and easy-access system, ONT is increasingly used worldwide, generally by laboratories sequencing their favorite organism, but with limited expertise in assembly methods. Nonetheless, assembly is not trivial: eukaryotic genomes assembly is a highly complex task (large genome sizes, high rates of repeated sequences, high heterozygosity levels and even polyploidy), and while prokaryotic genomes may appear less challenging, specific features such as circular DNA molecules, must be taken into consideration to achieve high quality assembly.

Numerous long-read assembly and post-assembly tools are available, relying on a large variety of approaches and algorithms, and many of them are frequently updated. However, even with this plethora of tools, there is no single silver bullet to genome assembly for all taxonomic groups. The systematic assessment of these assembly tools is needed to properly exploit data [1]. But performing benchmarks to find the best combination of tools for a given dataset and application is a highly complex task (see for example [2] and [3]), even if the computer code used for such

studies is increasingly being made available (e.g. [4] or [5]). This endeavor would presumably benefit from streamlined generalist data processing workflows that are accessible, scalable, traceable and reproducible.

A few solutions, such as Katuali [6] or CCBGpipe [7], were previously developed to tackle these issues, but they are dedicated to either eukaryotic or prokaryotic genomes only, or provide a restricted choice of assemblers. Some of these are also difficult to parallelize in a HPC environment, or to update with the latest versions of software components. In addition, for the subsequent evaluation of the assemblies, it is critical to organize and aggregate the numerous Quality Control (QC) metrics generated by various tools, in order to facilitate comparisons. As an example, microPIPE [8], a recently released Nextflow-based bacterial ONT reads assembly pipeline, fills some of these gaps but does not incorporate QC analysis apart from QUAST. To address these issues, we developed CulebrONT, a pipeline allowing users to easily assemble, circularize (if needed) and polish assemblies on multiple datasets with multiple alternative tools, while reporting various QC metrics for each assembly.

Implementation

CulebrONT assembles, circularises, polishes and corrects genome sequences from raw read sequences in fastq format, and provides QC metrics. To provide more flexibility to the user, individual tools are optional and include the most popular ones. Thus, CulebrONT can be used either to prototype and explore assemblies on new organisms or in assembly production. While originally developed primarily for working on ONT data, CulebrONT can also be used on PB data with generalist tools.

Six recent community-validated assemblers are currently included (Canu [9], Flye [10], Raven [11], MiniAsm [12] (coupled with minipolish [13] for an initial polishing), Shasta [14] and smartDeNovo [15]). Several tools for polishing (Racon [16], Pilon [17], Medaka [18] and Nanopolish [19]) were also included. If requested, Circlator [20] can automatically circularise the primary output of assemblers that cannot handle circular molecules (see Additional file 1 section 2.2 for more details). Each processing step in CulebrONT, from raw to final assembly, can be optionally assessed by various quality control tools (BUSCO [21], QUAST [22], BlobTools [23], KAT [24], Assemblytics [25], Samtools Flagstats [26], Mauve [27]) and Merqury [28].

CulebrONT uses Snakemake [29] principles and functionalities, enabling readability of the code, local and HPC scalability, reentry, reproducibility and modularity. The pipeline code is written following the Snakemake syntax, and include a dedicated Python module which should facilitate readability for developers. The Snakemake system implements scalability at the level of jobs that are scheduled independently to run, in parallel or sequentially depending on the available resources and the computing infrastructure. In addition, for rules that run programs with internal parallel processing capabilities, it is possible to configure the number of cores allocated to the corresponding individual jobs in CulebrONT. Snakemake

handles job failures and a restarted instance of the pipeline will compute only missing output. CulebrONT relies on conda [30] and singularity [31] that simplify installation of specific versions of the software, secure environments and greatly improve reproducibility. CulebrONT is available as a Python Package in PyPi to ease its installation https://pypi.org/project/culebrONT/. In addition, the CulebrONT API adapts the installation to local or distributed/HPC environment. In terms of modularity, relevant tools or processing steps can be selected or omitted in the user setup file *config.yaml*, and CulebrONT builds a dedicated instance of the workflow. Data can come from a single sample as well as from multiple ones (one fastq file per sample in the data folder). Tools paths are imported from *tools_path.yaml*, and cluster resources are managed using the *cluster_config.yaml* file (Figure 1.1). The dedicated CulebrONT class checks configuration files, controls if data and software environments exist and ensures the global coherence of the requested steps. This python class imports snakefiles to build a specific instance of the workflow (Figure 1.2). Upon execution (on HPC or a single machine, Figure 1.3), in addition to the expected final output, individual rules will generate log files (Figure 1.4).

Results

CulebrONT was developed in order to facilitate comparisons between primary assemblies, but also to check the effect of polishing on sequence accuracy. CulebrONT compiles information from these steps, but also essential quality information calculated at each activated step. An individual output directory with a specific subdirectories topology (see Additional file 1 to details) is generated for each samples. It notably contains a sub-directory corresponding to each activated step on the *config.yaml* file (assemblers, circularisation, polishing, correction, fixstart, and QC, for instance). In addition, a log directory includes execution information for all steps. CulebrONT generates an global HTML report, found in the *FINAL REPORT* folder. In this report, summarized statistics and relevant information can be found, such as the configuration parameters used or the tools versions and the computational time for each steps (Figure 2). These information can be useful for users that want to benchmark and identify suitable workflows for their data.

In order to provide an illustration of how CulebrONT can help benchmark assembly pipelines, ONT genome sequencing datasets for four highly contrasted species were tested : *i.* a strain of *Acinetobacter baumannii*[4] (ABJ9: reads N50 15,130 bp; median 7,621 bp; 741 Mb), *ii.* a strain of *Haemophilus haemolyticus*[4] (HM1C1321: reads N50 10,569 bp; median 7,027 bp; 1,6 Gb) *iii. Meloidogyne graminicola* nematoda[32] (VN18: reads N50 9,372 bp; median 2,853 bp; 3.2 Gb) and *iv.* a *Oryza sativa* dataset (unpublished data) (DJ123: N50 22,328 bp, median 13 544 bp, total 14,6 Gb).

For bacterial sequences, all available assemblers included in CulebrONT were tested using reads for samples ABJ9 and HM1C1321 from *A. baumannii* and *H. haemolyticus*. For ABJ9, good results were obtained by Canu and Miniasm, but the *best* (the longest N50 and lowest L50) were found using Miniasm + 2 minipolish

rounds + Medaka correction with only 2 circularised contig (N50 3,798,675 bp and L50 1) obtaining and a Busco score of 98.4%. Busco results obtained on this sample can be found on the figure 2. For HM1C1321, Flye gave the highest number of circularised contigs (N50 2,052,024 bp and L50 1) and the Busco score for conserved orthologous genes on the final assembly was 95.2%. More details can be found on the table 1).

Five assemblers were tested for *M. graminicola* (Flye, Shasta, Miniasm, Smartdenovo and Raven). With the VN18 sample, the lowest number of contigs (617) was obtained using a combination of Flye for assembly with a N50 of 402,454 bp and a L50 of 29 and followed by two rounds of Racon and ultimately Medaka polishing (see table 1).

On *Oryza sativa*, due to the larger genome size, only three assemblers were activated (Flye, Shasta and Raven) and polishing was performed with Racon only. For DJ123 rice sample, Flye + two Racon rounds seemed to be the optimal solution, with the best N50 (14,895,042 bp) and lower L50 (10 for 12 chromosomes). An overview of a CulebrONT report is shown on the figure 2.B using rice as an example. More details can be found on table 1 and more reports generated by CulebrONT are available at: https://itrop.ird.fr/culebront_utilities/BMC_RESULTS.

Applications

CulebrONT has been also successfully used on many various organisms, including more than 40 bacteria (*Xanthomonas sp*) (S. Cunnac, pers com), haploid fungi (*Pseudocercospora fijiensis*) (J. Carlier, pers com) and (*Fusarium oxysporum f.sp. cubense*) (E. Wicker, pers com), green algae (*Bathycoccus sp*), insects (*Drosophila sp* [33]), diploid grasses (*Paspalum sp* (O. Blanc, pers com) and other *Oryza sp*), as well as on allopolyploid plants (ABB triploid banana; M. Rouard, pers com).

Discussion

Obtaining a good assembly is a complex task, and the software ecosystem for this is evolving continually. While CulebrONT is currently proposing the state-of-the-art tools from today, we expect evolution in the next months or years for the next releases in different ways: On a short term, we will integrate the possibility of adding an external fasta sequence for using the QC part (for instance coming from an assembler not included in CulebrONT), re-entry at any steps, and also optimize the disk space usage.

On a medium term, we will re-evaluate the available tools (adding hifiAsm [34], removing tools not updated for years), restructure the final output structure for more readability, and better integrate the outputs from diverse QC tools (in particular BlobTools or RAM usage per tool).

Finally, in a longer term, *i.e.* in the next 2 years, we plan to integrate tools for contig integrity, possibilities for haplotyping/polyploids and so on.

Conclusion

In summary, CulebrONT simplifies the analysis of large-scale assemblies by allowing fast and reproducible processing of a single dataset or on a collection of samples

simultaneously. The output facilitates the comparative evaluation of the assembly workflows while keeping a traceable record of the analyses.

Availability and requirements

Project name: CulebrONT pipeline

Project home page: https://github.com/SouthGreenPlatform/CulebrONT_pipeline

Documentation: https://culebront-pipeline.readthedocs.io/en/latest/

Available Python Package: https://pypi.org/project/culebrONT

Source code: https://dataverse.ird.fr/dataset.xhtml?persistentId=doi:10.23708/TBPNWJ

Operating system: Platform independent

Programming language: Python >=3.6

Other requirements: Snakemake >=5.10

License: GPLv3

Any restrictions to use by non-academics: GPLv3

List of abbreviations

N50 : the smaller contig length required to cover 50 percent of the assembled genome sequence with the largest contigs.

L50 : The number of sequences required to obtain N50.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Availability of data and materials

Results generated by CulebrONT can be found in https://itrop.ird.fr/culebront_utilities/BMC_REPORTS/. Available data test can be found in https://itrop.ird.fr/culebront_utilities/Data-Xoo-sub/

Competing interests

The authors declare that they have no competing interests.

Funding

This work was supported by PHIM and DIADE research units. AC was recruited by a IRD engineer contract. TV was recruited by PHIM funding and the CRP-Rice program. FC contract was recruited by the MagMAX ANR-18-CE20-0016 grant awarded to Pierre Gladieux.

Author's contributions

JO, FS and SC conceived the original idea. JO, AC, SR, FC, TV and SC written code source. Documentation was performed by JO, SR, FS, SC and AC. JO and FS analysed data. JO, FS and SC drafted the manuscript. All authors read and approved the final manuscript.

Acknowledgements

Data processing was performed at the High-Performance Computing Cluster i-Trop of IRD. Authors thank N.Tando for administration support and the South Green Plateform. We also thanks the French Bioinformatics Infrastructure IFB for feedback on CulebrONT installation.

Author details

[1]DIADE Unit, University of Montpellier, CIRAD, IRD, 911 Avenue Agropolis, F-34394 , Montpellier Cedex 5, France. [2]PHIM Plant Health Institute, Univ Montpellier, CIRAD, INRAE, Institut Agro, IRD, Montpellier, France. [3]IFB - South Green Bioinformatics Platform, Bioversity, CIRAD, INRAE, IRD, F-34398 Montpellier, France. [4]Agricultural Genetics Institute, Vietnam Academy of Agricultural Sciences, Hanoi, Vietnam.

References

1. Murigneux, V., Rai, S.K., Furtado, A., Bruxner, T.J.C., Tian, W., Harliwong, I., Wei, H., Yang, B., Ye, Q., Anderson, E., Mao, Q., Drmanac, R., Wang, O., Peters, B.A., Xu, M., Wu, P., Topp, B., Coin, L.J.M., Henry, R.J.: Comparison of long-read methods for sequencing and assembly of a plant genome. GigaScience **9**(12) (2020). doi:10.1093/gigascience/giaa146. giaa146. https://academic.oup.com/gigascience/article-pdf/9/12/giaa146/35044370/giaa146.pdf
2. Wick, R., Holt, K.: Benchmarking of long-read assemblers for prokaryote whole genome sequencing [version 4; peer review: 4 approved]. F1000Research **8**(2138) (2021). doi:10.12688/f1000research.21782.4
3. Chen, Z., Erickson, D.L., Meng, J.: Benchmarking long-read assemblers for genomic analyses of bacterial pathogens using oxford nanopore sequencing. International Journal of Molecular Sciences **21**(23) (2020). doi:10.3390/ijms21239161
4. Wick, R.R., Judd, L.M., Cerdeira, L.T., Hawkey, J., Méric, G., Vezina, B., Wyres, K.L., Holt, K.E.: Trycycler: consensus long-read assemblies for bacterial genomes. Genome Biology **22**(1), 266 (2021). doi:10.1186/s13059-021-02483-z
5. Latorre-Pérez, A., Villalba-Bermell, P., Pascual, J., Vilanova, C.: Assembly methods for nanopore-based metagenomic sequencing: a comparative study **10**(1), 13588. doi:10.1038/s41598-020-70491-3
6. Katuali: https://nanoporetech.github.io/katuali/index.html (2020). https://nanoporetech.github.io/katuali/index.html
7. Liao, Y.-C., Cheng, H.-W., Wu, H.-C., Kuo, S.-C., Lauderdale, T.-L.Y., Chen, F.-J.: Completing circular bacterial genomes with assembly complexity by using a sampling strategy from a single minion run with barcoding. Frontiers in microbiology **10**, 2068–2068 (2019). doi:10.3389/fmicb.2019.02068. 31551994[pmid]
8. Murigneux, V., Roberts, L.W., Forde, B.M., Phan, M.-D., Nhu, N.T.K., Irwin, A.D., Harris, P.N.A., Paterson, D.L., Schembri, M.A., Whiley, D.M., Beatson, S.A.: MicroPIPE: validating an end-to-end workflow for high-quality complete bacterial genome construction **22**(1), 474. doi:10.1186/s12864-021-07767-z
9. Koren, S., Walenz, B.P., Berlin, K., Miller, J.R., Bergman, N.H., Phillippy, A.M.: Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. Genome Research **27**(5), 722–736 (2017). doi:10.1101/gr.215087.116. http://genome.cshlp.org/content/27/5/722.full.pdf+html
10. Kolmogorov, M., Yuan, J., Lin, Y., Pevzner, P.A.: Assembly of long, error-prone reads using repeat graphs. Nature Biotechnology **37**(5), 540–546 (2019). doi:10.1038/s41587-019-0072-8
11. Vaser, R., Šikić, M.: Raven: a de novo genome assembler for long reads. bioRxiv (2020). doi:10.1101/2020.08.07.242461. https://www.biorxiv.org/content/early/2020/08/10/2020.08.07.242461.full.pdf
12. Li, H.: Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences. Bioinformatics **32**(14), 2103–2110 (2016). doi:10.1093/bioinformatics/btw152. https://academic.oup.com/bioinformatics/article-pdf/32/14/2103/19567911/btw152.pdf
13. Wick, R., Holt, K.: Benchmarking of long-read assemblers for prokaryote whole genome sequencing [version 3; peer review: 4 approved]. F1000Research **8**(2138) (2020). doi:10.12688/f1000research.21782.3
14. Shafin, K., Pesout, T., Lorig-Roach, R., Haukness, M., Olsen, H.E., Bosworth, C., Armstrong, J., Tigyi, K., Maurer, N., Koren, S., Sedlazeck, F.J., Marschall, T., Mayes, S., Costa, V., Zook, J.M., Liu, K.J., Kilburn, D., Sorensen, M., Munson, K.M., Vollger, M.R., Monlong, J., Garrison, E., Eichler, E.E., Salama, S., Haussler, D., Green, R.E., Akeson, M., Phillippy, A., Miga, K.H., Carnevali, P., Jain, M., Paten, B.: Nanopore sequencing and the shasta toolkit enable efficient de novo assembly of eleven human genomes. Nature Biotechnology **38**(9), 1044–1053 (2020). doi:10.1038/s41587-020-0503-6
15. Liu, H., Wu, S., Li, A., Ruan, J.: Smartdenovo: A de novo assembler using long noisy reads. https://www.preprints.org/ (2020). doi:10.20944/preprints202009.0207.v1. https://www.preprints.org/manuscript/202009.0207/v1
16. Vaser, R., Sović, I., Nagarajan, N., Šikić, M.: Fast and accurate de novo genome assembly from long uncorrected reads. Genome research **27**(5), 737–746 (2017). doi:10.1101/gr.214270.116. 28100585[pmid]
17. Walker, B.J., Abeel, T., Shea, T., Priest, M., Abouelliel, A., Sakthikumar, S., Cuomo, C.A., Zeng, Q., Wortman, J., Young, S.K., Earl, A.M.: Pilon: An integrated tool for comprehensive microbial variant detection and genome assembly improvement. PLoS ONE **9**(11), 112963 (2014). doi:10.1371/journal.pone.0112963
18. Medaka: https://github.com/nanoporetech/medaka (2021). https://github.com/nanoporetech/medaka
19. Nanopolish: https://nanopolish.readthedocs.io/en/latest/index.html (2020). https://nanopolish.readthedocs.io/en/latest/index.html
20. Hunt, M., Silva, N.D., Otto, T.D., Parkhill, J., Keane, J.A., Harris, S.R.: Circlator: automated circularization of genome assemblies using long sequencing reads. Genome Biology **16**(1), 294 (2015). doi:10.1186/s13059-015-0849-0
21. Simão, F.A., Waterhouse, R.M., Ioannidis, P., Kriventseva, E.V., Zdobnov, E.M.: BUSCO: assessing genome assembly and annotation completeness with single-copy orthologs. Bioinformatics **31**(19), 3210–3212 (2015). doi:10.1093/bioinformatics/btv351. https://academic.oup.com/bioinformatics/article-pdf/31/19/3210/17086320/btv351.pdf
22. Gurevich, A., Saveliev, V., Vyahhi, N., Tesler, G.: QUAST: quality assessment tool for genome assemblies. Bioinformatics **29**(8), 1072–1075 (2013). doi:10.1093/bioinformatics/btt086
23. Laetsch, D., Blaxter, M.: Blobtools: Interrogation of genome assemblies [version 1; peer review: 2 approved with reservations]. F1000Research **6**(1287) (2017). doi:10.12688/f1000research.12232.1
24. KAT: The K-mer Analysis Toolkit, https://github.com/TGAC/KAT (2018). https://github.com/TGAC/KAT
25. Nattestad, M., Schatz, M.C.: Assemblytics: a web analytics tool for the detection of variants from an assembly. Bioinformatics **32**(19), 3021–3023 (2016). doi:10.1093/bioinformatics/btw369
26. Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., and, R.D.: The sequence alignment/map format and SAMtools. Bioinformatics **25**(16), 2078–2079 (2009). doi:10.1093/bioinformatics/btp352

27. Darling, A.C.E., Mau, B., Blattner, F.R., Perna, N.T.: Mauve: multiple alignment of conserved genomic sequence with rearrangements. Genome research **14**(7), 1394–1403 (2004). doi:10.1101/gr.2289704. 15231754[pmid]
28. Rhie, A., Walenz, B.P., Koren, S., Phillippy, A.M.: Merqury: reference-free quality, completeness, and phasing assessment for genome assemblies. Genome Biology **21**(1), 245 (2020). doi:10.1186/s13059-020-02134-9
29. Köster, J., Rahmann, S.: Snakemake—a scalable bioinformatics workflow engine. Bioinformatics **28**(19), 2520–2522 (2012). doi:10.1093/bioinformatics/bts480. https://academic.oup.com/bioinformatics/article-pdf/28/19/2520/819790/bts480.pdf
30. Anaconda: Anaconda Software Distribution. Anaconda Inc. (2021). https://docs.anaconda.com/
31. Kurtzer, G.M., Sochat, V., Bauer, M.W.: Singularity: Scientific containers for mobility of compute. PLOS ONE **12**(5), 1–20 (2017). doi:10.1371/journal.pone.0177459
32. Phan, N.T., Orjuela, J., Danchin, E.G.J., Klopp, C., Perfus-Barbeoch, L., Kozlowski, D.K., Koutsovoulos, G.D., Lopez-Roques, C., Bouchez, O., Zahm, M., Besnard, G., Bellafiore, S.: Genome structure and content of the rice root-knot nematode (meloidogyne graminicola). Ecology and Evolution **10**(20), 11006–11021 (2020). doi:10.1002/ece3.6680. https://onlinelibrary.wiley.com/doi/pdf/10.1002/ece3.6680
33. Mohamed, M., Dang, N.T.-M., Ogyama, Y., Burlet, N., Mugat, B., Boulesteix, M., Mérel, V., Veber, P., Salces-Ortiz, J., Severac, D., Pélisson, A., Vieira, C., Sabot, F., Fablet, M., Chambeyron, S.: A transposon story: From te content to te dynamic invasion of drosophila genomes using the single-molecule sequencing technology from oxford nanopore. Cells **9**(8), 1776 (2020). doi:10.3390/cells9081776. 32722451[pmid]
34. Cheng, H., Concepcion, G.T., Feng, X., Zhang, H., Li, H.: Haplotype-resolved de novo assembly using phased assembly graphs with hifiasm. Nature Methods **18**(2), 170–175 (2021). doi:10.1038/s41592-020-01056-5

Figures

[scale=0.25]image-diagrammeculebront.png

Figure 1 A graphical overview of the CulebrONT pipeline features. **A**. A conceptual diagram of the various tasks involved in setting up a custom CulebrONT workflow: setting up workflow steps and parameters, executing and viewing results of CulebrONT. (1) workflow steps and parameters are provided by the user in the *config.yaml*, *tools_path.yaml* and *cluster_config.yaml* files. (2) CulebrONT python class reads, transforms and chains rules to build the corresponding pipeline. (3) Rule execution can be performed sequentially or in parallel, on a single machine or a HPC infrastructure, with concurrent jobs and multi threaded tasks. (4) The results of individual steps are aggregated in a convenient HTML output report. **B**. Simplified diagrams showing the components of an example CulebrONT workflow for assembling a linear genome. In the upper part, the tools used for assembly (blue shape outline) and polishing/correction (yellow and red shape outline) are connected with arrows following the directional flow of the pipeline. In the central part, the various versions of the assembled genomes are gathered (*preparing_fasta_to_quality*) to serve as input for subsequent quality evaluation tools in the lower part. Ultimately, the output is aggregated in a final report.

[scale=0.25]image-report.png

Figure 2 Example of CulebrONT HTML reports. A. Busco statistics for *Acinetobacter sp.* dataset B. Quast for the rice dataset.

Tables

Table 1 Results obtained on ONT data using CulebrONT on various biological models and assemblers. The *"Organisms"* row provides information about the samples, the *"Reads"* subsection describes input reads features, the *"Activated tools"* section lists the tools activated ('yes') in the corresponding CulebrONT run, and a summary of the best results is presented in the *"Best workflow"* row (considering best by the longest N50 and lowest L50 in this study).

Specie			*Acinetobacter baumannii*	*Haemophilus haemolyticus*	*Meloidogyne graminicola*	*Oryza sativa*
Organism	Sample name		ABJ9	HM1C1321	VN18	DJ123
	Kingdom		Bacteria	Bacteria	Nematoda	Plantae
	Genome Size		3.9M	2.1M	35M	380M
	Ht %		-	-	2%	0.20%
	Ploidy		haploid	haploid	haploid	diploid
Reads						
	Number		69,135	185,001	671,385	930,909
	Read length N50 (bp)		15,130	10,569	9,372	22,328
	Total bases (bp)		741,274,205	1,619,105,604	3,283,238,647	14,599,973,374
	Depth		190X	771X	93X	38X
Activated tools						
	Assemblers					
		Flye	yes	yes	yes	yes
		Raven	yes	yes	yes	yes
		Miniasm	yes	yes	yes	no
		Smartdenovo	yes	yes	yes	no
		Shasta	yes	yes	yes	yes
		Canu	yes	yes	no	no
	Circulatisation		yes	yes	no	no
	Polishers					
		Racon	x2	x2	x2	x3
	Correcteurs					
		Medaka	yes	yes	yes	no
		Nanopolish	no	no	yes	no
		Pilon	no	no	yes	no
Best workflow			Miniasm+2xminipolish+Medaka	Flye+2xRacon+Medaka	Flye+2xRacon+Medaka	Flye+3xRacon
	Busco score		C:98.4%[S:98.4%,D:0%]	C:95.2%[S:95.2%,D:0%]	C:65.9%[S:64.7%,D:1.2%]	C:95.1%[S:93.7%,D:1.4%]
	Largest contig (bp)		3,798,675	2,175,110	1,984,010	26,745,371
	Contigs number		2	7	617	261
	N50 (bp)		3,798,675	2,052,024	402,454	14,895,042
	L50		1	1	29	10

Additional Files

Additional file 1 of CulebrONT pipeline

Supplemental file descriptions of the CulebrONT implementation.

A

CulebrONT 1.5.2 report

Rulegraph
Config file Parameters
Benchmark
BUSCO
QUAST
Versions

BUSCO

BUSCO sets are collections of orthologous groups with near-universally-distributed single-copy genes in each species

ABJ9

Assembler	Steps	Complete	Single	Duplicate	Fragmented	Missing	Total Genes
All	All	All	Al	All	All	Al	Al
CANU	ASSEMBLY	83.1%	82.3%	0.8%	13.7%	3.2%	124
CANU	CORRECTION_MEDAKA	97.6%	97.6%	0.0%	2.4%	0.0%	124
CANU	POLISHING_RACON	87.9%	87.9%	0.0%	11.3%	0.8%	124
FLYE	ASSEMBLY	93.5%	93.5%	0.0%	6.5%	0.0%	124
FLYE	CORRECTION_MEDAKA	97.6%	97.6%	0.0%	2.4%	0.0%	124
FLYE	POLISHING_RACON	91.1%	91.1%	0.0%	7.3%	1.6%	124
MINIASM	ASSEMBLY	93.5%	93.5%	0.0%	5.6%	0.9%	124
MINIASM	CORRECTION_MEDAKA	98.4%	98.4%	0.0%	1.6%	0.0%	124
MINIASM	POLISHING_RACON	93.5%	93.5%	0.0%	5.6%	0.9%	124
RAVEN	ASSEMBLY	5.6%	5.6%	0.0%	29.8%	64.6%	124
RAVEN	CORRECTION_MEDAKA	97.6%	97.6%	0.0%	2.4%	0.0%	124
RAVEN	POLISHING_RACON	89.5%	89.5%	0.0%	9.7%	0.8%	124
SHASTA	ASSEMBLY	49.2%	49.2%	0.0%	35.5%	15.3%	124
SHASTA	CORRECTION_MEDAKA	98.4%	98.4%	0.0%	1.6%	0.0%	124
SHASTA	POLISHING_RACON	89.5%	89.5%	0.0%	10.5%	0.0%	124
SMARTDENOVO	ASSEMBLY	92.7%	91.9%	0.8%	6.5%	0.8%	124
SMARTDENOVO	CORRECTION_MEDAKA	97.6%	96.8%	0.8%	2.4%	0.0%	124
SMARTDENOVO	POLISHING_RACON	88.7%	87.9%	0.8%	9.7%	1.6%	124

B

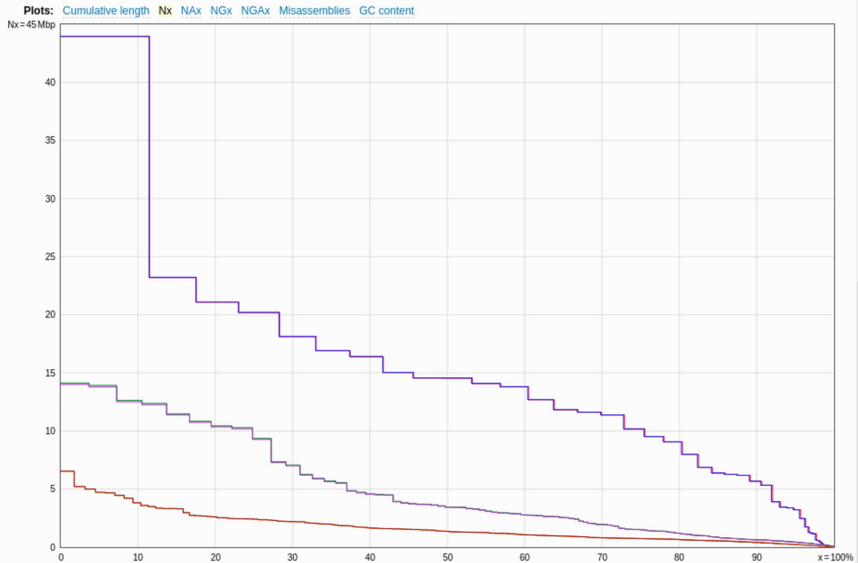
CulebrONT report

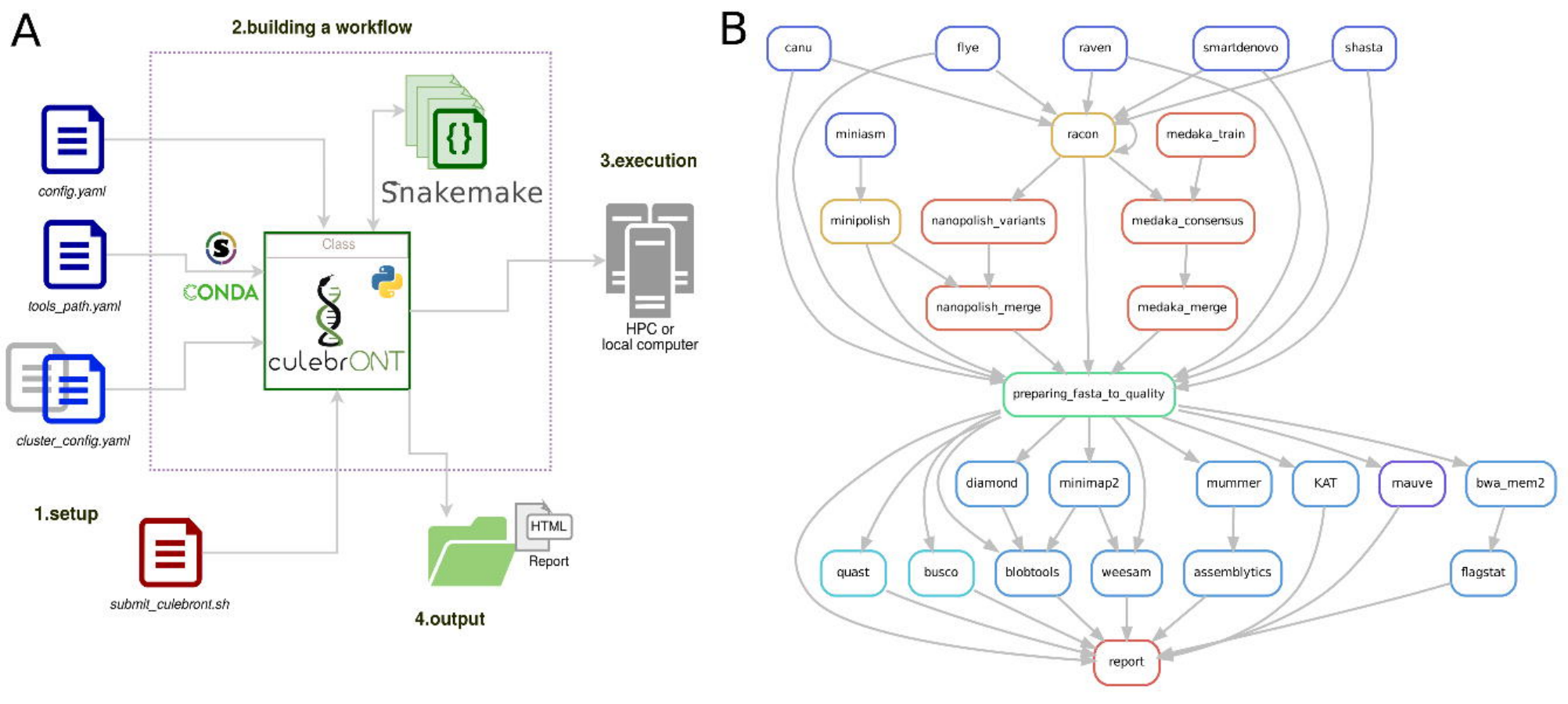
Rulegraph
Config file Parameters
Benchmark
QUAST
nerica4
chhomrongDhan
nericaL19
DJ123
Og132
sahel108
B88
IUPLR17
Og20
Ob589
tequing
Llanura11
Ob546
Ob569

QUAST is a good starting point to help evaluate the quality of assemblies. It provides many helpful contiguity statistics.

nerica4 chhomrongDhan nericaL19 DJ123 Og132 sahel108 B88 IUPLR17
Og20 Ob589 tequing Llanura11 Ob546 Ob569

Open Quast report on new window




A
2.building a workflow
config.yaml
tools_path.yaml
cluster_config.yaml
Snakemake
CONDA
Class
culebrONT
3.execution
HPC or
local computer
1.setup
submit_culebront.sh
HTML
Report
4.output
B
canu
flye
raven
smartdenovo
shasta
miniasm
racon
medaka_train
minipolish
nanopolish_variants
medaka_consensus
nanopolish_merge
medaka_merge
preparing_fasta_to_quality
diamond
minimap2
mummer
KAT
mauve
bwa_mem2
quast
busco
blobtools
weesam
assemblytics
flagstat
report