Overcoming uncollapsed haplotypes in long-read assemblies of non-model organisms

Nadège Guiglielmoni[1], Alessandro Derzelle[2], Karine van Doninck[2] and Jean-François Flot[1,3]
[1] Service Evolution Biologique et Ecologie, Université libre de Bruxelles, 1050 Brussels, Belgium
[2] Laboratoire d'Ecologie et Génétique Evolutive, Université de Namur, 5000 Namur, Belgium
[3] Interuniversity Institute of Bioinformatics in Brussels - (IB)[2], 1050 Brussels, Belgium

Corresponding author: nadege.guiglielmoni@ulb.be

Abstract *Third-generation sequencing, also called long-read sequencing, has revolutionized genome assembly: as PacBio and Nanopore technologies have become more accessible in technicity and in cost (with decreasing error rates and increasing read lengths), long-read assemblers have flourished and are starting to deliver chromosome-level assemblies. However, an independent, comparative assessment of the performance of these programs on a common, real-life dataset is still lacking.*
To fill this gap, we tested the efficiency of long-read assemblers on the genome of the rotifer Adineta vaga, a non-model organism for which both PacBio and Nanopore reads were available. Although all the assemblers included in our benchmark aimed to produce a haploid genome assembly with collapsed haplotypes, we observed strikingly different behaviors of these assemblers on highly heterozygous regions: allelic regions that were most divergent were sometimes not merged, resulting in variable amounts of duplicated regions. We identified three strategies to alleviate this problem: setting a read-length threshold to filter out shorter reads; choosing an assembler less prone to retaining uncollapsed haplotypes; and post-processing the assembled set of contigs using a downstream tool to remove uncollapsed haplotypes. These three strategies are not mutually exclusive and, when combined, generate haploid assemblies with genome sizes, coverage distributions, and k-mer completeness matching expectations.

Keywords genome assembly, long reads, benchmark, heterozygosity

Introduction

With the advent of long-read sequencing, high-quality assemblies are now commonly achieved on all types of organisms. The competition between the two main long-read sequencing companies, Pacific Biosciences (PacBio) and Oxford Nanopore Technologies (Nanopore), has prompted an increase in output as well as a decrease in cost, making their technologies more accessible to research teams and more applicable to challenging genomes. The main advantage of long reads over short reads (such as those generated by Illumina sequencing platforms) is their typical length that averages around 10 kilobases (kb) [1]. Their length facilitates genome assembly into contigs and scaffolds as they can span repetitive regions [2]; they can also be used to resolve haplotypes [3].

However, long leads have a much higher error rate than Illumina data, and these errors are mainly insertions and deletions for long reads vs. substitutions for Illumina reads. PacBio data have a random error pattern that can be compensated with high coverage: as a result, reading the same DNA regions over and over several times can be used to generate a consensus with an accuracy close to 99%, in a process dubbed Circular Consensus Sequencing (CSS) and marketed as PacBio HiFi (standing for "high-fidelity")[4]. Nanopore reads, on the other hand, have systematic errors in homopolymeric regions and are thus often combined with Illumina sequencing to correct the errors still present in contigs, in a process called "polishing" [5][6]. Nanopore reads keep getting longer, with runs attaining N50s over 100 kilobases (kb) and longest reads spanning over 1 Megabase (Mb) [7][8].

This progress has prompted the development of many programs to produce *de novo* assemblies from long reads, all of which follow the Overlap Layout Consensus (OLC) paradigm [9]. Briefly, OLC methods start by building an overlap graph (the "O" step), then simplify it and clean it by applying various heuristics (the "L" step), and finally compute the consensus sequence of each contig (the "C"

step). Some long-read assemblers follow strictly this paradigm, such as Flye [10], Ra [11], Raven [11] (a further development of Ra by the same author), Shasta [12] and wtdbg2 [13]; whereas others such as Canu [14] add a preliminary correction step based on a all-versus-all alignment of the reads (Table 1).

Tab. 1. Assemblers included in this study.

Assembler	Nanopore	PacBio	Version used	Access
Canu	✓	✓	1.9	github.com/marbl/canu
Flye	✓	✓	2.5-g315122d	github.com/fenderglass/Flye
Ra	✓	✓	0.2.1	github.com/lbcb-sci/ra
Raven	✓	✓	0.0.7	github.com/lbcb-sci/raven
Shasta	✓	×	0.3.0	github.com/chanzuckerberg/shasta
wtdbg2	✓	✓	2.5	github.com/ruanjue/wtdbg2

Long-read assemblers were recently benchmarked on real and simulated PacBio and Nanopore bacterial datasets [15], and all assemblers tested proved their efficiency at reconstructing full microbial genomes within one hour and with a low RAM usage. The Flye publication [10] provides an evaluation of Canu, Flye, Ra and wtdbg2 on several eukaryotic genomes. However, more complete evaluations of these tools on their ability to provide structurally correct (i.e., without artefactually duplicated genome regions) haploid assemblies from non-model diploid organisms are still lacking. To fill this gap, we present here a quantitative and qualitative assessment of six long-read assemblers on a relatively small eukaryotic genome, *Adineta vaga*, for which a short-read, fragmented assembly was published some years ago [16]. As with most non-model organisms, *Adineta vaga*'s genome presents a mid-range heterozygosity of ca. 2% with a mix of highly heterozygous and low-heterozygosity regions, making such genome more challenging to assemble than those of model organisms that have often a very low level of polymorphism [17].

Results

Assemblies of PacBio reads

The PacBio assemblies had variable lengths, ranging from 108 Mb (Ra for reads > 15 kb) to 165 Mb (Canu with all reads). Larger assemblies also presented higher numbers of duplicated BUSCOs as well as higher k-mer completeness and bimodal coverage distributions (Figure 1). N50s ranged from 395 kb (Canu, reads > 15 kb) to 2.7 Mb (Flye, all reads).

Canu produced the largest assemblies, between 147 Mb and 165 Mb, as well as the highest number of duplicated BUSCOs. Besides, its k-mer completeness was systematically higher than the expected value of 50% for a haploid assembly of a diploid genome. Canu's assembly coverage distributions exhibited two peaks around 50X and 100X. The half-coverage peak represents allelic regions that have not been collapsed, resulting in artefactually duplicated regions in the assembly.

Flye and Raven assemblies also had two peaks in their coverage distribution, although the 50X peak was smaller than with Canu. Flye assemblies all had similar length, BUSCO score, k-mer completeness and coverage distribution regardless of the read-length threshold used; however, their N50 decreased with increasing read-length threshold. Raven, on the other hand, produced shorter assemblies when smaller reads were removed but the contig N50 remained stable.

Ra and wtdbg2 both produced assemblies with very small 50X peaks. The sizes of the Ra assemblies decreased when using higher read-length threshold, with low-coverage contigs disappearing from the assembly in a fashion similar to Raven; their contig N50, however, remained fairly constant. By contrast, assembly size did not vary for wtdbg2 when using higher read-length threshold but contig N50 increased.

Assemblies of Nanopore reads

Similarly to the PacBio assemblies, Nanopore assembly sizes ranged from 93 Mb (Ra for reads > 40 kb) to 152 Mb (Canu with all reads) (Figure 2). Except when a stringent read-length threshold was applied, all assemblies exceeded the expected size of 102.3 Mb. Nanopore assemblies achieved a much

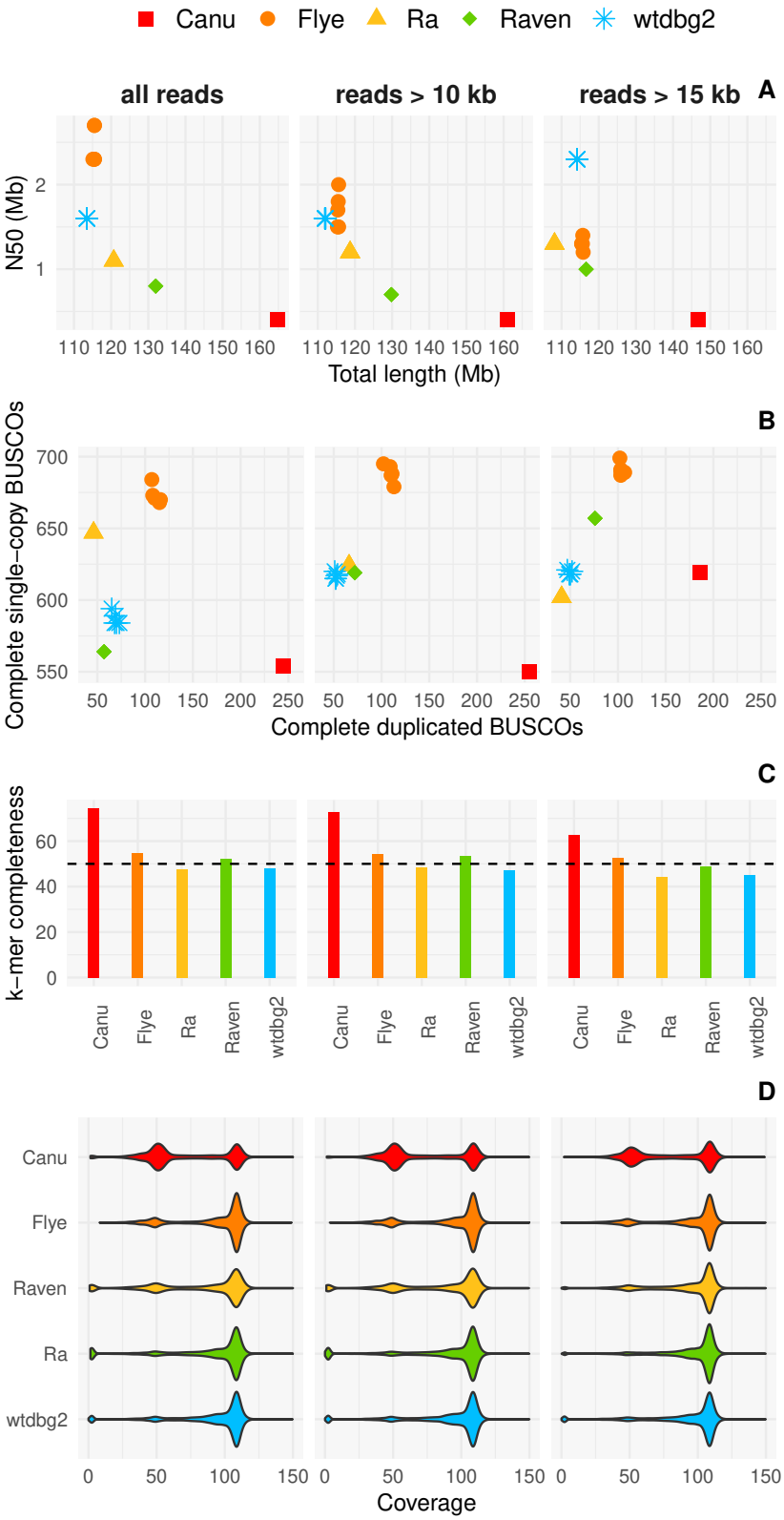

Fig. 1. Statistics of PacBio assemblies, with A) N50 plotted against total assembly length, B) number of complete single-copy BUSCOs plotted against number of complete duplicated BUSCOs, C) mean *k*-mer completeness and D) short-reads coverage distribution of the contigs of one replicate assembly for each program. In A and B, each program is represented by five points that correspond to five replicate runs, except for Canu that was run only once.

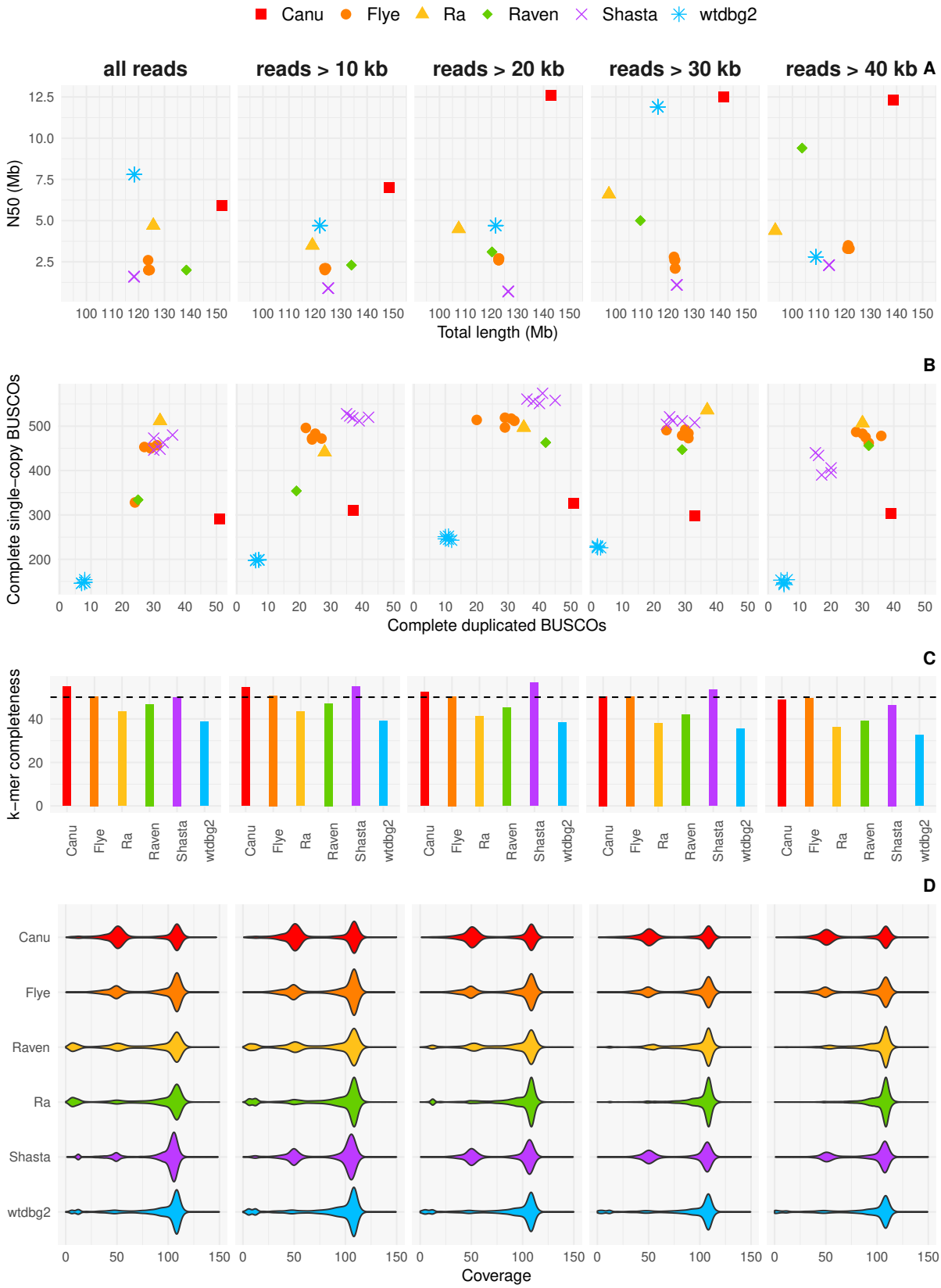

Fig. 2. Statistics of Nanopore assemblies, with with A) N50 plotted against total assembly length, B) number of complete single-copy BUSCOs plotted against number of complete duplicated BUSCOs, C) mean k-mer completeness and D) short-reads coverage distribution of the contigs of one replicate assembly for each program. In A and B, each program is represented by five points that correspond to five replicate runs, except for Canu that was run only once.

higher contiguity than PacBio assemblies: while PacBio assemblies yielded a highest N50 of about 2.5 Mb, Nanopore assemblies reached 12.5 Mb. The number of complete single-copy BUSCOs was lower in Nanopore assemblies (up to 600) than in PacBio assemblies (up to 700), and the number of duplicate complete BUSCOs was also much smaller (up to 50 with Nanopore vs. up to 250 with PacBio). These lower BUSCO scores, together with the lower k-mer completeness of Nanopore assemblies, likely result from the non-random error pattern of Nanopore reads that produces errors (mostly indels) in the consensus sequences produced by the assemblers.

As for PacBio reads, Canu assemblies of Nanopore reads were oversized and the coverage distribution showed two distinct peaks around 50X and 100X, indicating that many haplotypes had not been collapsed and remained present in two copies in the assemblies. In contrast to PacBio assemblies, however, the N50 of Nanopore assemblies produced by Canu increased markedly when using a higher read-length threshold, reaching the highest value among all assemblers tested for reads > 20 kb. Flye, Raven and Shasta also presented two peaks at 50X and 100X, and Raven has a third peak corresponding to low-coverage contigs. Although Raven's N50 increased, like Canu's, when shorter reads were filtered out, this was not the case for Flye and Shasta. Raven assemblies improved also with increasing read-length threshold: low-coverage contigs disappeared, assembly length became closer to expectations, and BUSCO score increased. Likewise, Ra assemblies improved with read selection: when using only reads longer than 30 kb, the low-coverage and half-coverage regions disappeared completely. However, when keeping only reads over 40 kb, the assembly N50 decreased and genome size dropped under the expected value.

Purging duplicated regions

As our long-read assemblies contained various amount of uncollapsed haplotypes, resulting in assemblies larger than expected, we further tested the possibility of improving these assemblies by collapsing haplotypes *a posteriori* with purge_haplotigs [18]. We tested this program on assemblies produced with all reads, as they were systematically oversized, and on only one replicate run for each assembler tested (as the replicates showed minimal differences).

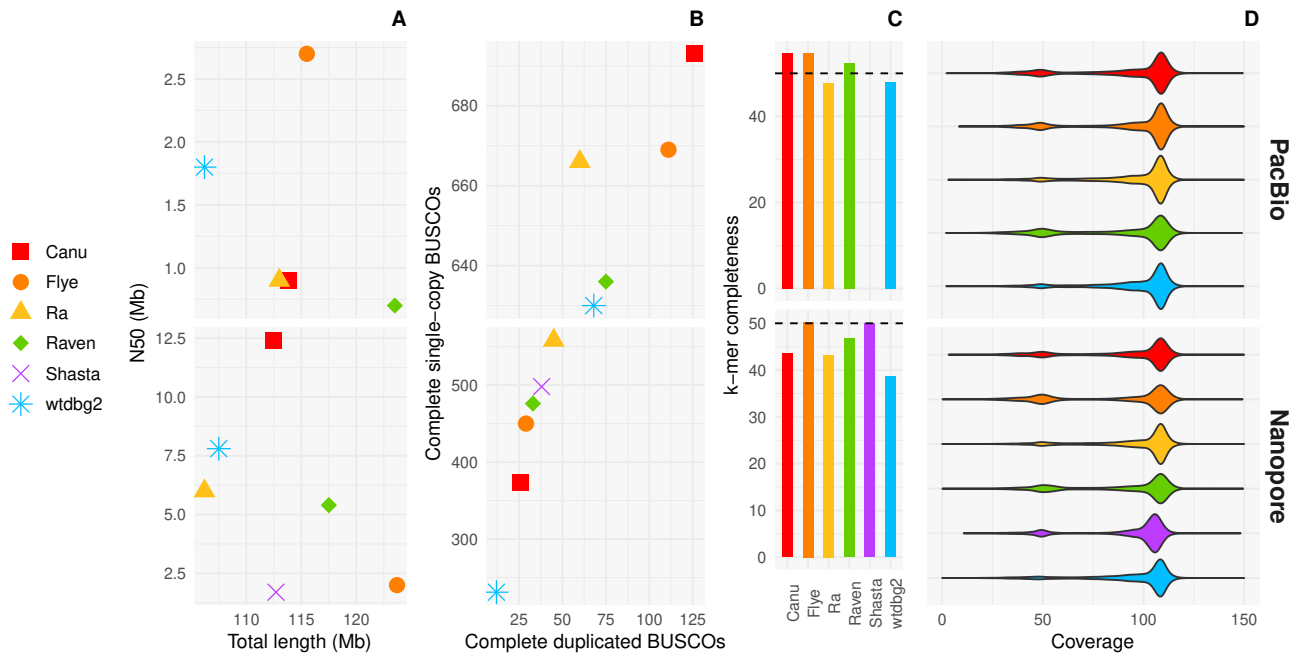

Fig. 3. Statistics of PacBio and Nanopore assemblies after purge_haplotigs, with A) N50 plotted against total assembly length, B) number of complete single-copy BUSCOs plotted against number of complete duplicated BUSCOs, C) mean k-mer completeness and D) short-reads coverage distribution of the contigs.

After purging duplicated regions, all assemblies improved except for Flye that did not change much (Figure 3): N50s remained stable, but genome sizes became closer to the expected values. k-mer completeness also decreased and became closer to the expected 50%. None of the purged assemblies

exhibited any low-coverage contigs, and 50X peaks were reduced. purge_haplotigs drastically changed the output of Canu, both for PacBio and Nanopore assemblies. Assembly decreased markedly, together with a large reduction of the 50X peak. Canu followed by purge_haplotigs achieved the highest N50 of all Nanopore assemblies and the best BUSCO score of all PacBio assemblies. Ra, Raven and wtdbg2 assemblies after haplotype purging were similar to those obtained with a high read-length threshold, although low-coverage contigs were better removed from the wtdbg2 assembly by purge_haplotigs.

Computational performance

As computational resources can be a limiting factor in genome assembly, we provide CPU time and RAM measurements for all the assemblers, except Canu that required significantly higher resources and was therefore run on a computer cluster (Figure 4). The assembler with the smallest resource consumption was wtdbg2: it required the lowest amount of RAM, ran the fastest on PacBio reads and was also time-efficient on Nanopore reads. As would be expected, using a read-length threshold improved RAM usage and CPU time, with the exception of Flye that still required a high amount of RAM for Nanopore reads. Shasta ran fast but required the largest amount of RAM for the full Nanopore dataset. Ra and Raven used only a limited amount of RAM, with a peak usage at 50 GB.

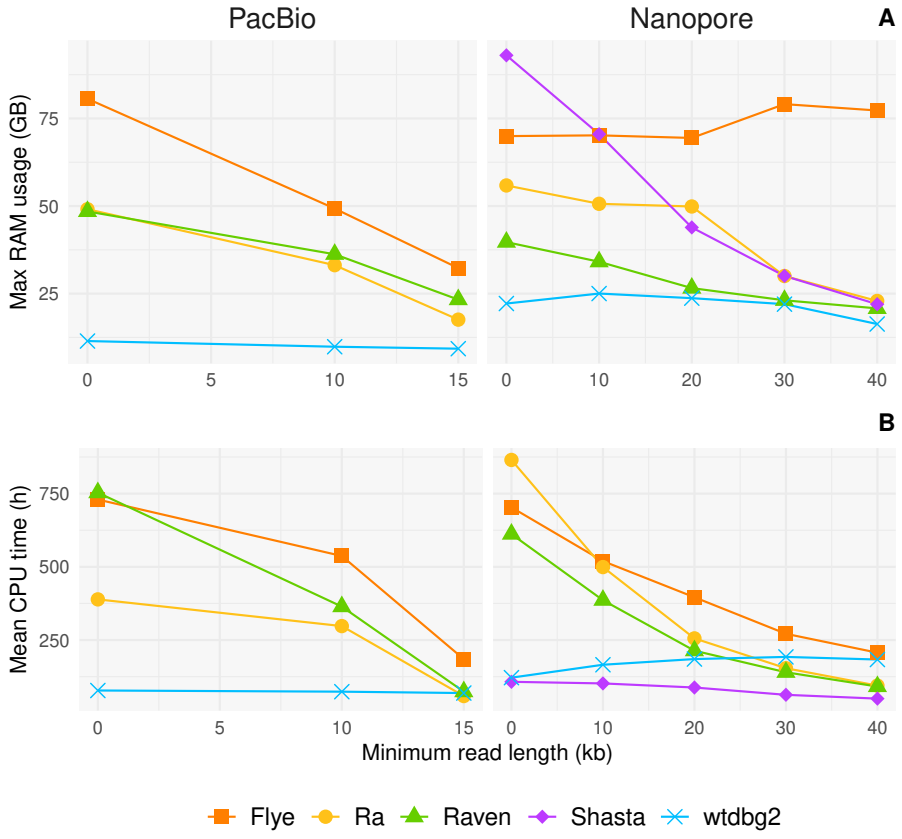

Fig. 4. Computational resources (RAM and CPU time) used by the assemblers with the full Nanopore and PacBio datasets and the subsets. Canu is not included as it was run on a cluster.

Discussion

While PacBio assemblies were superior in terms of BUSCO scores and k-mer completeness, the contiguity of Nanopore assemblies was far greater. These results are coherent with the characteristics of these reads. An important finding is that keeping only reads longer than a given threshold improves in many cases the quality of haploid assemblies.

We found that Canu poorly collapses allelic regions and yields oversized assemblies. The program does not seem adequate to solve highly divergent regions on its own, but it can produce high quality

haploid assemblies when combined with a downstream tool such as purge_haplotigs. Flye assemblies also exhibited uncollapsed haplotypes; selecting the longest reads did not help and neither did purge_haplotigs. Still, Flye exhibited both good contiguity and good quality.

wtdbg2 performed well on PacBio data, but less on Nanopore reads. This program did not seem to have difficulty with heterozygous regions but was rather affected by low-coverage contigs. Read selection on size did not significantly improve the assemblies, but purge_haplotigs removed low-coverage contigs, therefore improving the output. Although Shasta was less good in collapsing divergent haplotypes (and neither read selection nor purge_haplotype helped with that) and did not achieve particularly high contiguity on our PacBio datasets, its k-mer completeness and BUSCO scores were very good.

Ra and Raven performed better on size-selected reads, which led to smaller genome sizes closer to expectation. With these assemblers, purge_haplotigs was also efficient at purging uncollapsed haplotypes. While Ra and Raven both achieved convincing contiguity and quality, Ra stood out among all the assemblers as the one less prone to retain uncollapsed haplotypes.

We believe that this benchmark will help researchers working on non-model organisms select a long-read sequencing technology and an assembly method suitable for their project, and will also help them better understand the resulting assemblies.

Material & Methods

All command lines are provided in Supplementary Table S1.

Genome size estimation

The genome size of *Adineta vaga* was estimated using KAT [19] on an Illumina dataset of 25 millions paired-end 250 basepairs (bp) reads (see Table S2). The diploid size was estimated to 204.6 Mb, therefore a haploid assembly should have a length around 102.3 Mb.

Long-read assemblies

Canu, Flye, Ra, Raven, Shasta and wtdbg2 were tested on two *Adineta vaga* long-read datasets: PacBio reads totalling 23.5 Gb with a N50 of 11.6 kb; and Nanopore read totalling 17.5 Gb with a N50 of 18.8 kb (after trimming using Porechop, `github.com/rrwick/Porechop`). All assemblers were used with default parameters, except for Shasta for which the minimum read length was set to zero (instead of the default 10 kb setting). Shasta was not used on PacBio datasets, as running it with default parameters gave results much below par compared to the other assemblers tested (results not shown). When assemblers required an estimated size, the value 100 Mb was provided. PacBio assemblies were run on all reads, on reads > 10 kb (14.4 Gb) and on reads > 15 kb (4.7 Gb). Nanopore assemblies were run on all reads, on reads > 10 kb (13,3 Gb), on reads > 20 kb (8.3 Gb), on reads > 30 kb (5.7 Gb) and on reads > 40 kb (4.10 Gb). For more details on long-reads datasets, see Table S2. To test for reproducibility, all assemblers were run five times with the exception of Canu that was run only once (because it required much more computational resources to run).

Purging duplicated regions

Reads were mapped on assemblies using minimap2 [20]. For each assembly, we ran `purge_haplotigs hist` to compute coverage histograms that we used to set low, mid and high cutoffs; these values were then used by `purge_haplotigs cov` to detect suspect contigs. Finally, we ran `purge_haplotigs purge` to eliminate duplicated regions.

Assembly evaluation

To evaluate the assemblies, we ran BUSCO 4 [21] against metazoa odb10 (954 features) without the parameter –long. We ran `KAT comp` [19] to calculate k-mer completeness by reference to the same Illumina 2*250 bp dataset used to estimate the genome size. To compute coverage, thes Illumina reads were mapped on one replicate assembly per assembler using bowtie2 [22] and the coverage was computed with tinycov, available at `github.com/cmdoret/tinycov`, with a window size of 20 kb.

Performance evaluation

For Flye, Ra, Raven, Shasta and wtdbg2, maximal RAM usage and mean CPU time were measured using the command `time` with 14 threads on a computer with an i9-9900X 3.5 Ghz processor and 128 GB RAM.

Author contributions

NG and AD ran the assemblies. NG evaluated the assemblies. NG and JFF wrote the manuscript. KVD and JFF provided the sequencing data.

Acknowledgements

We thank Antoine Limasset for his useful advice. Nanopore reads were generated at Genoscope as part of the France Génomique project 'ALPAGA' coordinated by Etienne Danchin (www.france-genomique.org/projet/alpaga/). Part of this analysis was performed on computing clusters of the Leibniz-Rechenzentrum (LRZ) and the Consortium des Équipements de Calcul Intensif (CÉCI). This project is funded by the Horizon 2020 research and innovation program of the European Union under the Marie Skłodowska-Curie grant agreement 764840 (ITN IGNITE, www.itn-ignite.eu).

Competing financial interests

The authors declare no competing financial interests.

References

[1] Fritz J. Sedlazeck, Hayan Lee, Charlotte A. Darby, and Michael C. Schatz. Piercing the dark matter: bioinformatics of long-range sequencing and mapping. *Nature Reviews Genetics*, 19(6):329–346, 2018.

[2] Martin O. Pollard, Deepti Gurdasani, Alexander J. Mentzer, Tarryn Porter, and Manjinder S. Sandhu. Long reads: their purpose and place. *Human Molecular Genetics*, 27(R2):R234–R241, 2018.

[3] Murray Patterson et al. WhatsHap: weighted haplotype assembly for future-generation sequencing reads. *Journal of Computational Biology*, 22(6):498–509, 2015.

[4] Aaron M. Wenger et al. Accurate circular consensus long-read sequencing improves variant detection and assembly of a human genome. *Nature Biotechnology*, 37(10):1155–1162.

[5] Ritu Kundu, Joshua Casey, and Wing-kin Sung. HYPO: super fast & accurate polisher for long read assemblies. *bioRxiv*, 2019.

[6] Aleksey V. Zimin and Steven L. Salzberg. The genome polishing tool POLCA makes fast and accurate corrections in genome assemblies. *bioRxiv*, 2019.

[7] Miten Jain et al. Nanopore sequencing and assembly of a human genome with ultra-long reads. *Nature Biotechnology*, 36(4):338–345, 2018.

[8] Karen H. Miga et al. Telomere-to-telomere assembly of a complete human X chromosome. *bioRxiv*, 2019.

[9] Jason R. Miller, Sergey Koren, and Granger Sutton. Assembly algorithms for next-generation sequencing data. *Genomics*, 95(6):315–327, 2010.

[10] Mikhail Kolmogorov, Jeffrey Yuan, Yu Lin, and Pavel A. Pevzner. Assembly of long, error-prone reads using repeat graphs. *Nature Biotechnology*, 37(5):540–546, 2019.

[11] Robert Vaser and Mile Šikić. Yet another de novo genome assembler. *International Symposium on Image and Signal Processing and Analysis, ISPA*, pages 147–151, 2019.

[12] Kishwar Shafin et al. Efficient de novo assembly of eleven human genomes using PromethION sequencing and a novel nanopore toolkit. *bioRxiv*, 2019.

[13] Jue Ruan and Heng Li. Fast and accurate long-read assembly with wtdbg2. *Nature Methods*, 17(2):155–158, 2020.

[14] Sergey Koren et al. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *Genome Research*, 25(2):1–11, 2017.

[15] Ryan R. Wick and Kathryn E. Holt. Benchmarking of long-read assemblers for prokaryote whole genome sequencing. *F1000Research*, 8:2138, 2019.

[16] Jean-François Flot et al. Genomic evidence for ameiotic evolution in the bdelloid rotifer *Adineta vaga*. *Nature*, 500(7463):453–457, 2013.

[17] Ellen M. Leffler et al. Revisiting an old riddle: what determines genetic diversity levels within species? *PLoS Biology*, 10(9):e1001388, 2012.

[18] Michael J Roach, Simon A Schmidt, and Anthony R Borneman. Purge Haplotigs: allelic contig reassignment for third-gen diploid genome assemblies. *BMC Bioinformatics*, 19(1):1–10, 2018.

[19] Daniel Mapleson, Gonzalo Garcia Accinelli, George Kettleborough, Jonathan Wright, and Bernardo J. Clavijo. KAT: a K-mer analysis toolkit to quality control NGS datasets and genome assemblies. *Bioinformatics*, 33(4), 2016.

[20] Heng Li. Sequence analysis Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, 2018.

[21] Felipe A Simão, Robert M Waterhouse, Panagiotis Ioannidis, Evgenia V Kriventseva, and Evgeny M Zdobnov. BUSCO: assessing genome assembly and annotation completeness with single-copy orthologs. *Bioinformatics*, 31(19):3210–3212, 2015.

[22] Ben Langmead and Steven L. Salzberg. Fast gapped-read alignment with Bowtie 2. *Nature Methods*, 9(4):357–359, 2012.