



HAL
open science

GraphUnzip: unzipping assembly graphs with long reads and Hi-C

Roland Faure, Nadège Guiglielmoni, Jean-François Flot

► To cite this version:

Roland Faure, Nadège Guiglielmoni, Jean-François Flot. GraphUnzip: unzipping assembly graphs with long reads and Hi-C. JOBIM 2021 - Journées Ouvertes en Biologie, Informatique et Mathématiques, Jul 2021, Paris, France. pp.1-7. hal-03441016

HAL Id: hal-03441016

<https://hal.science/hal-03441016>

Submitted on 22 Nov 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

GraphUnzip: unzipping assembly graphs with long reads and Hi-C

Roland FAURE¹, Nadège GUIGLIELMONI¹ and Jean-François FLOT^{1,2}

¹ Service Evolution Biologique et Ecologie, Université libre de Bruxelles, 1050 Brussels, Belgium

² Interuniversity Institute of Bioinformatics in Brussels - (IB)², 1050 Brussels, Belgium

Corresponding author: nadege.guiglielmoni@ulb.be

Abstract *Long reads and Hi-C have revolutionized the field of genome assembly as they have made highly contiguous assemblies accessible even for challenging genomes. As haploid chromosome-level assemblies are now commonly achieved for all types of organisms, phasing assemblies has become the new frontier for genome reconstruction. Several tools have already been released using long reads and/or Hi-C to phase assemblies, but they all start from a set of linear sequences and are ill-suited for non-model organisms with high levels of heterozygosity. We present GraphUnzip, a fast, memory-efficient and flexible tool to unzip assembly graphs into their constituent haplotypes using long reads and/or Hi-C data. As GraphUnzip only connects sequences that already had a potential link in the assembly graph, it yields high-quality gap-less supercontigs. To demonstrate the efficiency of GraphUnzip, we tested it on the human HG00733 and the potato *Solanum tuberosum*. In both cases, GraphUnzip yielded phased assemblies with improved contiguity.*

Keywords genome assembly, phasing, long reads, Hi-C

Introduction

The field of genomics is thriving and chromosome-level assemblies are now commonly achieved for all types of organisms, thanks to the combined improvements of sequencing and assembly methods. Chromosome-level assemblies are generally haploid, regardless of the ploidy of the genome. To obtain a haploid assembly of a multiploid (i.e. diploid or polyploid) genome, homologous chromosomes are collapsed into one sequence. However, assemblers often struggle to collapse highly heterozygous regions, which leads to breaks in the assembly and duplicated regions [1]. Furthermore, haploid assemblies provide a partial representation of multiploid genomes: ideally, multiploid genomes should be phased rather than collapsed if the aim is to grasp their whole complexity [2].

The combination of low-accuracy long reads, such as Oxford Nanopore Technologies (ONT) reads and Pacific Biosciences (PacBio) Continuous Long Reads (CLRs), with proximity ligation (Hi-C) reads has made chromosome-level assemblies accessible for all types of organisms. The latest development of PacBio, high-accuracy long circular consensus sequencing (CCS) reads (a.k.a. HiFi), is now starting to deliver highly contiguous phased assemblies [3,4,5]. Hi-C scaffolding is commonly used in genome assembly projects to obtain chromosome-level scaffolds. This approach relies on the interaction frequency in the genome and these interactions are heightened between loci belonging to the same chromosome [6]. Based on this principle, alleles can be associated using their interaction frequencies.

A first approach to phase assemblies is called trio-binning and uses sequencing data from the individual and its parents to retrieve haplotypes [7]; yet this method is unavailable when the parents cannot be identified, or for asexual species. Existing tools are able to use either long reads (Falcon-Unzip [8], WhatsHap [9]) or Hi-C reads (Falcon-Phase [10], ALLHiC [11]) for phasing assemblies, but they are limited to phasing local variants or well-identified haplotypes and are not suited for complex, highly heterozygous genomes. WhatsHap takes as input a collapsed assembly and searches for alternative haplotypes. As collapsing haplotypes can be too difficult for highly heterozygous regions, it seems more intuitive to phase these assemblies *de novo*. FALCON-Unzip and FALCON-Phase offer this alternative, yet they are dependant on the output of the FALCON assembler and cannot be combined

with other assemblers.

We present GraphUnzip, a new tool to phase assemblies using long reads and/or Hi-C. GraphUnzip implements a radically new approach to phasing that starts from an assembly graph instead of a set of linear sequences. In an assembly graph, heterozygous regions result in bubbles every time the assembler is unable to collapse the haplotypes or to choose one of them. GraphUnzip "unzips" the graph, meaning that it separates the haplotypes by duplicating homozygous regions that have been collapsed and partitioning heterozygous regions into haplotypes. This tool is based on a simple principle that was implemented in many scaffolders since SSPACE [12]: long-range data (mate-pair reads, long reads, linked reads, proximity ligation...) provide information on the linkage between contigs that can be used to group and orient them into scaffolds. As GraphUnzip takes as input and produces as output assembly graphs, it only connects contigs that are actually adjacent in the genome and yields gap-less scaffolds, i.e. supercontigs. GraphUnzip is compatible with any assembler that produces an assembly graph. We tested GraphUnzip on the genomes of the human HG00733 and the potato *Solanum tuberosum*. GraphUnzip is available at github.com/nadegeguiglielmoni/GraphUnzip.

Methods

Inputs

GraphUnzip requires an assembly graph in GFA (Graphical Fragment Assembly) format. The Hi-C input is a sparse matrix, such as the one obtained when processing the reads with hicstuff [13]. hicstuff also provides a module to convert other file formats (e.g. cool, a common Hi-C format) to a sparse matrix. The long reads are mapped to the assembly graph using GraphAligner [14].

Overview of GraphUnzip

In an assembly graph, contigs that are inferred to be adjacent or to overlap in the assembly are connected with edges. However, some of these connections between contigs may be artefacts. To discriminate correct edges from erroneous ones, GraphUnzip relies on long reads and/or Hi-C data. These data are translated into interactions between contigs: the strength of interaction between two contigs is defined as the number of long reads bridging both contigs when using long reads as input; and as the number of Hi-C contacts between the two contigs when using Hi-C as input. In both cases, a strong interaction is a sign of proximity on the genome.

GraphUnzip first builds one or two interaction matrices containing all pairwise interactions between contigs, depending on whether long-read data, Hi-C data or both are provided (Figure 1). In the next step, GraphUnzip iteratively reviews all contigs and their edges. The strength of an edge i is computed based on the strength of interaction between the contigs it connects. A high strength supports the reality of the link, while a low strength may signal an artefactual edge. When a contig has several edges at one of its extremities, these edges are compared in a pairwise fashion. This comparison uses two user-provided thresholds: the rejection threshold T_R and the acceptance threshold T_A , where $T_R < T_A$. Considering two edges X and Y and their respective strengths $i(X)$ and $i(Y)$, if $i(X) < i(Y)$, Y is considered strong; if $i(X)/i(Y) < T_R$, then X is considered weak, else, if $T_R \leq i(X)/i(Y) < T_A$, X is flagged as dubious. X is labelled as strong when $i(X)/i(Y) \geq T_A$. The algorithm thereafter considers weak edges as artefacts that do not actually exist in the genome, whereas strong edges represent true connections. If both long reads and Hi-C input data are provided, strengths based on long reads are used first because they are more reliable locally, and strengths based on Hi-C are only used if some edges are flagged as dubious.

Edges identified as weak in the previous calculation are removed. Then, every contig that has more than one strong edge and no dubious edge at one end is duplicated as many times as the number of these strong edges. Such contigs are typically collapsed homozygous regions that need to be present in several copies to be included in every haplotypes. All the copies retain the edges of the original

contig at its other end. This entails that the duplication of contigs creates many new (and potentially artefactual) edges. Contigs that are unambiguously linked are merged in supercontigs that will be handled as regular contigs thereafter.

When assessing the strength of two putative edges (S_1, S_2) and (S_1, S_3) connecting the supercontigs S_1 , S_2 , and S_3 , the strength of these edges are calculated as the strength of interaction between contigs in S_1 and contigs present in S_2 but not in S_3 (and vice versa). For example, in the third step of Figure 1, when trying to associate supercontig a-b to either d-e or d'-f, only the interactions between the supercontig a-b and the contigs e and f are considered. Interactions between the supercontig a-b and the contigs d and d' are not considered in the calculation because d and d' actually originate from the duplication of a collapsed region.

All contigs and edges are iteratively processed s times to phase the assembly, where s is a user-provided parameter. Because extremely long contigs tend to share a significant number of Hi-C contacts even if they are not adjacent, we observed that in extreme cases the algorithm could join two chromosomes by their telomeric ends. The Hi-C matrix is used at the end of the process to detect such chimeric connections in the assembly graph, based on low Hi-C interactions, and break them.

***Homo sapiens* HG00733 assemblies**

We used HiFi, ONT and Hi-C reads from [15]. HiFi reads were assembled using hifiasm with the parameter `-l 0`, and the resulting `p_utg` assembly graph was used for downstream analyses. All HiFi reads and the ONT reads longer than 30 kb were mapped to the assembly using GraphAligner with the parameter `-x vg`. Hi-C reads were processed with hicstuff using the parameters `--aligner bowtie2 --enzyme 200 --iterative`. GraphUnzip was run with parameters `--accept 0.10 --reject 0.05 --exhaustive --whole_match --minimum_match 0.8`. All non-ambiguous paths in the GFA were merged using Bandage. The assemblies were compared to the DipAsm reference [16] using QUAST v5.0.2 [17] with the parameters `-m 0 --eukaryote --large --min-identity 99.9`.

***Solanum tuberosum* assemblies**

HiFi, ONT and Hi-C reads published in [18] were retrieved from the NCBI Sequence Read Archive with the Bioproject accession number PRJNA573826. The HiFi reads were assembled using hifiasm with the parameter `-l 0`, and the `p_utg` assembly graph was used for downstream analyses. All HiFi reads and the ONT reads longer than 25 kb were mapped to the assembly using GraphAligner with the parameter `-x vg`. Hi-C reads were processed with hicstuff using the parameters `--aligner bowtie2 --enzyme MboI --iterative`. GraphUnzip was run with parameters `--accept 0.40 --reject 0.10 --exhaustive --whole_match --minimum_match 0.8`. All non-ambiguous paths in the GFA were merged using Bandage. To check the output of GraphUnzip, we mapped the published assembly to the assembly graph using GraphAligner. We used calN50 (available at github.com/lh3/calN50) to compute the NG50 against the published assembly size of 1.67 Gb [18]. BUSCO v4 [19] was run with parameters `-m genome --long` against the dataset `viridiplantae odb10`.

Computational performance

RAM usage and CPU time were measured with the command `/usr/bin/time -v` on a desktop computer with 128 GB of RAM and a i9-9900X 3.5 GHz processor.

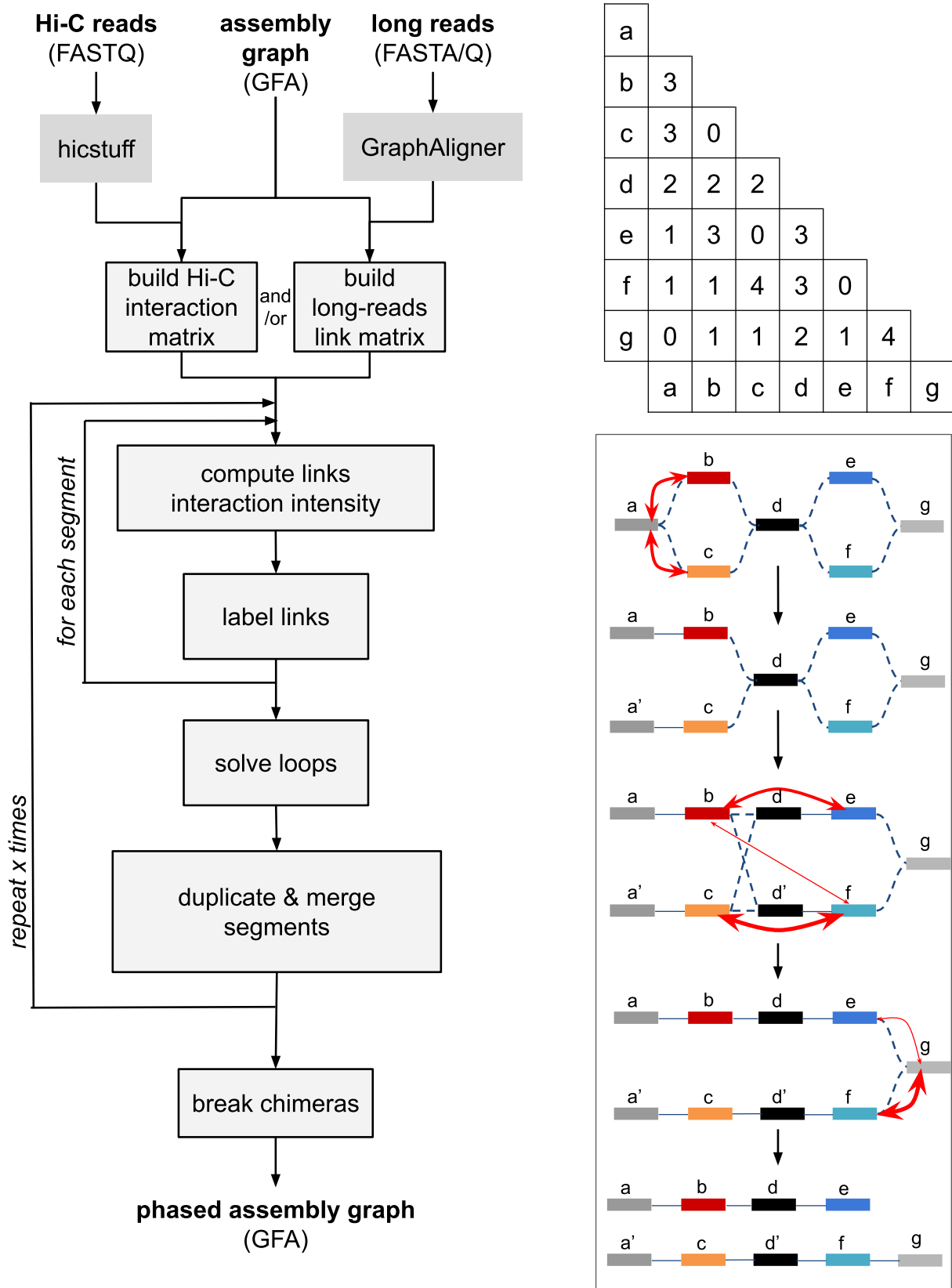


Fig. 1. Description of GraphUnzip: workflow of the program (left), interaction matrix (top right), and overview of the algorithm to discriminate links (bottom right). This example algorithm analyzes the potential links between the segments a, b, c, d, e, f, g. The red arrows represent the intensity of interactions between the segments, computed based on the values in the matrix.

Tab. 1. Assembly metrics of *Homo sapiens* HG00733 compared with the DipAsm reference.

Assembly	GraphUnzip	Size	N50	NA50	Misassemblies	CPU	RAM
Reference	-	5.9 Gb	27.8 Mb	27.8 Mb	84	-	-
hifiasm	-	5.5 Gb	397 kb	343 kb	9146	-	-
	ONT + Hi-C	6.2 Gb	1.5 Mb	1.2 Mb	8091	33min 46s	23.5 GB

Results

Homo sapiens HG00733

We compared the hifiasm + GraphUnzip assembly of the human HG00733 genome with a published reference obtained using DipAsm, based on the N50, the NA50 and the number of misassemblies. The N50 represents the contiguity of the assembly: it is defined as the length of the largest contig for which 50% of the assembly size is contained in contigs of equal or greater length. The NA50 is the N50 of the assembly broken at every misassembly (compared to a reference). GraphUnzip increased the size of the hifiasm assembly (from 5.5 Gb to 6.2 Gb), and the N50 rose as well (from 397 kb to 1.2 Mb) (Table 1). The NA50 was improved while the number of misassemblies decreased in the GraphUnzip supercontigs. Notably, the reference assembly size is only 5.9 Gb, while the GraphUnzip assembly reaches 6.2 Gb, which is the expected size for a phased human genome.

We also tried an assembly of the HiFi reads with Flye, but the draft assembly was only 2.9 Gb, little below half the expected size, which indicates that the haplotypes were nearly completely collapsed. A good candidate assembly for GraphUnzip should have uncollapsed heterozygous regions, as GraphUnzip is not able to retrieve a missing haplotype in collapsed heterozygous regions and can only duplicate the collapsed region, leading in that case to a suboptimal result.

Solanum tuberosum

Tab. 2. Assembly metrics of *Solanum tuberosum*. The NG50 values were computed based on an estimated genome size of 1.67 Gb.

Assembly	GraphUnzip	Size	NG50	BUSCO		CPU	RAM
				Single	Dup.		
Reference	-	1.67 Gb	66.1 Mb	21.6%	76.9%	-	-
hifiasm	-	1.51 Gb	2.2 Mb	21.2%	77.9%	-	-
	HiFi	1.69 Gb	3.7 Mb	7.1%	91.5%	16s	0.2 GB
	ONT	1.67 Gb	3.4 Mb	6.8%	92.2%	52s	0.2 GB
	Hi-C	1.69 Gb	5.6 Mb	7.8%	91.5%	38min 27s	11.5 GB
	HiFi + Hi-C	1.69 Gb	4.9 Mb	9.4%	89.4%	39min 59s	11.5 GB
	ONT + Hi-C	1.73 Gb	5.9 Mb	7.3%	91.8%	39min 10s	11.5 GB

We tested GraphUnzip on the diploid genome of the potato *Solanum tuberosum* RH89-039-16, for which a phased assembly of 1.67 Gb [18] was recently published. We assembled the HiFi reads with hifiasm and then ran GraphUnzip using the HiFi, ONT and/or Hi-C reads. The draft assembly was 1.51 Gb, and after phasing with GraphUnzip, the assembly size rose to 1.67-1.73 Gb (Table 2). In this case, we compared the NG50s, a value similar to N50 but based on a reference genome size rather than the assembly size. GraphUnzip increased the contiguity: from 2.2 Mb, the NG50 reached 3.4 to 5.9 Mb. The combination of both ONT and Hi-C reads yielded the highest NG50. Hi-C reads improved the contiguity better than long reads. The overall BUSCO completeness of the GraphUnzip supercontigs was slightly improved compared to the reference: 98.6-99.3% against 98.5% for the reference, and the number of duplicated BUSCO features was higher as well (89.4-92.2% against 76.9%). We mapped the published assembly to the GraphUnzip assembly graph obtained when using Hi-C and ONT reads. We found that there were no differences in phasing between the two assemblies. However, some regions that were phased by hifiasm and GraphUnzip were collapsed in the published assembly. This result, in conjunction with the higher number of duplicated features, indicates that GraphUnzip

led to an improved phased assembly.

Computational performance

For both the human and *Solanum tuberosum* genomes, GraphUnzip required limited computational resources as it ran in less than 1 hour on a single thread and used up to 23.5 GB of memory. For *Solanum tuberosum*, the run time was also shorter when using only long reads (less than a minute). The longer run time when using Hi-C reads was due to the building of the interaction matrix. As this interaction matrix is outputted by the program, this file can be reused for other runs, which will consequently finish faster. Therefore, users can try several sets of parameters to optimize the result, with short runtimes.

Conclusion

GraphUnzip is a flexible tool that can phase assemblies of high-accuracy long reads with long reads and/or Hi-C. A limitation of GraphUnzip is that it does not necessarily reach chromosome-level assemblies like most Hi-C scaffolders do, but it aims instead to produce more contiguous gap-less supercontigs by fully exploiting assembly graphs. As genome projects now usually include long reads and Hi-C to obtain chromosome-level assemblies, GraphUnzip can easily be integrated in assembly projects to obtain *de novo* phased assemblies for non-model organisms.

Acknowledgments

This project was funded by the Horizon 2020 research and innovation program of the European Union under the Marie Skłodowska-Curie grant agreement No 764840 (ITN IGNITE, www.itn-ignite.eu). Part of this analysis was performed on computing clusters of the Leibniz-Rechenzentrum (LRZ) and the Consortium des Équipements de Calcul Intensif (CÉCI) funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under Grant No. 2.5020.11.

References

- [1] Nadège Guiguelmoni, Antoine Houtain, Alessandro Derzelle, Karine Van Doninck, and Jean-François Flot. Overcoming uncollapsed haplotypes in long-read assemblies of non-model organisms. *BMC Bioinformatics*, 22(1):1–23, 2021.
- [2] Xingtang Zhang, Ruoxi Wu, Yibin Wang, Jiabin Yu, and Haibao Tang. Unzipping haplotypes in diploid and polyploid genomes. *Computational and Structural Biotechnology Journal*, 18:66–72, 2020.
- [3] 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.
- [4] Haoyu Cheng, Gregory T Concepcion, Xiaowen Feng, Haowen Zhang, and Heng Li. Haplotype-resolved *de novo* assembly using phased assembly graphs with hifiasm. *Nature Methods*, pages 1–6, 2021.
- [5] Sergey Nurk, Brian P Walenz, Arang Rhie, Mitchell R Vollger, Glennis A Logsdon, Robert Grothe, Karen H Miga, Evan E Eichler, Adam M Phillippy, and Sergey Koren. HiCanu: accurate assembly of segmental duplications, satellites, and allelic variants from high-fidelity long reads. *Genome Research*, 30(9):1291–1305, 2020.
- [6] Jean-François Flot, Hervé Marie-Nelly, and Romain Koszul. Contact genomics: scaffolding and phasing (meta) genomes using chromosome 3d physical signatures. *FEBS Letters*, 589(20):2966–2974, 2015.
- [7] Sergey Koren, Arang Rhie, Brian P Walenz, Alexander T Dilthey, Derek M Bickhart, Sarah B Kingan, Stefan Hiendleder, John L Williams, Timothy PL Smith, and Adam M Phillippy. *De novo* assembly of haplotype-resolved genomes with trio binning. *Nature Biotechnology*, 36(12):1174–1182, 2018.
- [8] Chen-Shan Chin, Paul Peluso, Fritz J Sedlazeck, Maria Nattestad, Gregory T Concepcion, Alicia Clum, Christopher Dunn, Ronan O’Malley, Rosa Figueroa-Balderas, Abraham Morales-Cruz, et al. Phased diploid genome assembly with single-molecule real-time sequencing. *Nature Methods*, 13(12):1050–1054, 2016.
- [9] Murray Patterson, Tobias Marschall, Nadia Pisanti, Leo Van Iersel, Leen Stougie, Gunnar W Klau, and Alexander Schönhuth. WhatsHap: weighted haplotype assembly for future-generation sequencing reads. *Journal of Computational Biology*, 22(6):498–509, 2015.

- [10] Zev N Kronenberg, Arang Rhie, Sergey Koren, Gregory T Concepcion, Paul Peluso, Katherine M Munson, Stefan Hiendleder, Olivier Fedrigo, Erich D Jarvis, Adam M Phillippy, et al. Extended haplotype phasing of de novo genome assemblies with FALCON-Phase. *bioRxiv*, 2019.
- [11] Xingtang Zhang, Shengcheng Zhang, Qian Zhao, Ray Ming, and Haibao Tang. Assembly of allele-aware, chromosomal-scale autopolyploid genomes based on Hi-C data. *Nature Plants*, 5(8):833–845, 2019.
- [12] Marten Boetzer, Christiaan V Henkel, Hans J Jansen, Derek Butler, and Walter Pirovano. Scaffolding pre-assembled contigs using sspace. *Bioinformatics*, 27(4):578–579, 2011.
- [13] Cyril Matthey-Doret, Lyam Baudry, Amaury Bignaud, Axel Cournac, Rémi Montagne, Nadège Guiglielmoni, Théo Foutel-Rodier, and Vittore F. Scolari. *koszullab/hicstuff*, October 2020.
- [14] Mikko Rautiainen and Tobias Marschall. GraphAligner: rapid and versatile sequence-to-graph alignment. *Genome Biology*, 21(1):1–28, 2020.
- [15] David Porubsky, Peter Ebert, Peter A Audano, Mitchell R Vollger, William T Harvey, Pierre Marijon, Jana Ebler, Katherine M Munson, Melanie Sorensen, Arvis Sulovari, et al. Fully phased human genome assembly without parental data using single-cell strand sequencing and long reads. *Nature Biotechnology*, 39(3):302–308, 2021.
- [16] Shilpa Garg, Arkarachai Fungtammasan, Andrew Carroll, Mike Chou, Anthony Schmitt, Xiang Zhou, Stephen Mac, Paul Peluso, Emily Hatas, Jay Ghurye, et al. Chromosome-scale, haplotype-resolved assembly of human genomes. *Nature Biotechnology*, 39(3):309–312, 2021.
- [17] Alexey Gurevich, Vladislav Saveliev, Nikolay Vyahhi, and Glenn Tesler. QUASt: quality assessment tool for genome assemblies. *Bioinformatics*, 29(8):1072–1075, 2013.
- [18] Qian Zhou, Dié Tang, Wu Huang, Zhongmin Yang, Yu Zhang, John P Hamilton, Richard GF Visser, Christian WB Bachem, C Robin Buell, Zhonghua Zhang, et al. Haplotype-resolved genome analyses of a heterozygous diploid potato. *Nature Genetics*, pages 1–6, 2020.
- [19] 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.