RESEARCH

A systematic comparison of chloroplast genome assembly tools

Jan A Freudenthal[1,2], Simon Pfaff[1,3], Niklas Terhoeven[1,2], Arthur Korte[1], Markus J Ankenbrand[1,2,4,5†] and Frank Förster[1,3,6,7,8*†]

[*]Correspondence: frank.foerster@computational.bio.uni-giessen.de
[6]Fraunhofer IME-BR, Ohlebergsweg 12, 35392 Gießen, Germany
Full list of author information is available at the end of the article
[†]Corresponding author

Abstract

Background: Chloroplasts are intracellular organelles that enable plants to conduct photosynthesis. They arose through the symbiotic integration of a prokaryotic cell into an eukaryotic host cell and still contain their own genomes with distinct genomic information. Plastid genomes accommodate essential genes and are regularly utilized in biotechnology or phylogenetics. Different assemblers that are able to assess the plastid genome have been developed. These assemblers often use data of whole genome sequencing experiments, which usually contain reads from the complete chloroplast genome.

Results: The performance of different assembly tools has never been systematically compared. Here we present a benchmark of seven chloroplast assembly tools, capable of succeeding in more than 60 % of known real data sets. Our results show significant differences between the tested assemblers in terms of generating whole chloroplast genome sequences and computational requirements. The examination of 105 data sets from species with unknown plastid genomes leads to the assembly of 20 novel chloroplast genomes.

Conclusions: We create docker images for each tested tool that are freely available for the scientific community and ensure reproducibility of the analyses. These containers allow the analysis and screening of data sets for chloroplast genomes using standard computational infrastructure. Thus, large scale screening for chloroplasts within genomic sequencing data is feasible.

Keywords: Chloroplast; Genome; Assembly; Software; Benchmark

Introduction

General introduction and motivation

Chloroplasts are essential organelles present in the cells of plants and autotrophic protists, which enable the conversion of light energy into chemical energy via photosynthesis. They harbor their own prokaryotic type of ribosomes and a circular DNA genome that varies in size between 120 kbp to 160 kbp [1]. Because of their small size, chloroplast genomes were one of the first targets for sequencing projects. The first chloroplast genome sequences were obtained in 1986 [2, 3]. These early efforts elucidated the general genome organization and structure of the chloroplast DNA and have been reviewed previously [4, 5]. Chloroplast genomes are widely used for evolutionary analyses [6, 7], barcoding [8, 9, 10], and meta-barcoding [11, 12]. Interesting features of chloroplast genomes include their small size (120 kbp to 160 kbp,[1]), due to endosymbiotic gene transfer [13, 14], and the low number of 100 to 120 genes that are encoded within the genome [4]. Despite the overall high sequence conservation of the chloroplast genome, there are striking differences in the gene content between different autotroph groups, exemplified by the loss of the whole *ndh* gene family in Droseraceae [15]). Even more extreme evolutionary cases, where chloroplasts show a very low GC content and a modified genetic code have been described [16].

Structurally, two inverted repeats (Inverted Repeats (IRs)) named $\mathrm{IR_A}$ and $\mathrm{IR_B}$ of 10 kbp to 76 kbp divide the chloroplast genome into a Large Single Copy (LSC) and a Small Single Copy (SSC) region [1], which complicates genome assembly with short read technologies[17]. Moreover, the existence of different chloroplasts within a single individual, and thus multiple different chloroplast genomes, have been described for various plants [18, 19, 20]. This phenomenon - called heteroplasmy - is only poorly understood in terms of its origin and evolutionary importance, but it impacts the assembly of whole chloroplast genomes.

Nonetheless, given its small size, it is still much easier to decipher a complete chloroplast genome than a complete core genome. Consequently, many comparative genomic approaches target the chloroplast genome. For example the *Arabidopsis thaliana* core genome is approximately 125 Mbp in length [21, 22] while the size of the *A. thaliana* chloroplast genome at 154 kbp is more than 800 times smaller [23].

Each single chloroplast contains several hundred copies of its genome [24, 25]. Therefore, many plant core genome sequencing projects contain reads that originate from chloroplasts as a by-product and permit the assembly of chloroplast genomes. Such sequences are available from databases such as the Sequence Read Archive at NCBI [26].

Complete chloroplast genomes can be used as super-barcodes [27], both for biotechnology applications and genetic engineering [28]. Furthermore, the availability of whole chloroplast genomes would enable large scale comparative studies [29].

Approaches to extracting chloroplasts sequences from whole genome data

Different strategies have been developed to assemble chloroplast genomes [30]. In general, obtaining a chloroplast genome from whole genome sequencing (WGS) data requires two steps: (1) extraction of chloroplast reads from the sequencing data; (2) assembly and resolution of the special circular structure including the IRs. The extraction of chloroplast reads can be achieved by mapping the reads to a reference chloroplast. [31]. A different approach that does not depend on the availability of a reference chloroplast, uses the higher coverage of reads originating from the chloroplast [32]. Here, a *k*-mer analysis can be used to extract the most frequent reads. An example for this is implemented in `chloroExtractor` [33]. A third method, which is for example used by `NOVOPlasty` [34], combines both approaches by using a reference chloroplast as seed and simultaneously assembling the reads based on *k*-mers.

Purpose and scope of this study

The goal of this study was to compare the effectiveness and efficiency of existing open source command-line tools to perform a de-novo assembly of whole chloroplast genomes from raw genomic data. We only compared tools that require minimal configuration, which includes no need for extensive data preparation, no need for a specific reference (apart from *A. thaliana*), no need to change default parameters, and no manual finishing. We further restricted our benchmark to paired end Illumina data as the sole input, as these are routinely generated by modern sequencing platforms [35].

Thus our analyses reflect the most common use cases: (1) trying a tool quickly without digging into options for fine tuning; (2) large scale automatic applications. We tested all tools on more than 100 real data sets for species without published chloroplast genomes. The performance of most tools might be significantly improved by optimizing parameters for each data set specifically, but this exhaustive comparison - including tuning of all different possible parameters for each tool- was out of the scope of this study.

To summarize, we provide new chloroplast genome sequences for many species and demonstrated the ability to discover and assemble novel chloroplast genomes as well as asses inter/intra-individual differences in the respective chloroplast genomes.

Results

Performance metrics

All described tools have been tested with regard to their assembly time, memory and CPU utilization.

Time requirements

Massive differences between the different tools were observed in terms of the run time for the assembly. Apart from tool-specific differences, input data and number of threads used had a huge impact on the time requirement. The observed run times varied from a few minutes to several hours (Figure 1).

Some assemblies failed to finish within our time limit of 48 h. On average, the longest time to generate an assembly was taken by `IOGA` and `Fast-Plast` followed by `ORG.Asm` and `GetOrganelle`. The most time efficient tool was `chloroExtractor`, which was a little faster than `NOVOPlasty` and `Chloroplast assembly protocol`.

Not all tested tools were able to benefit from having access to multiple threads. Both `NOVOPlasty` and `ORG.Asm` required almost the same time independent of being allowed to utilize 1, 2, 4, or 8 threads. In contrast, `Chloroplast assembly protocol`, `chloroExtractor`, `GetOrganelle`and `Fast-Plast` all profited from multi-threading settings (Figures 1 and 2 and Additional file 1: Tables S4 to S6).

Memory and CPU Usage

The peak and mean CPU usage, as well as peak memory and disk usage were recorded for all assemblers based on the same input data set and number of threads (Figure 2 and Additional file 1: Tables S4 to S6). In general, the size of the input data influenced the peak memory usage with the exception of `chloroExtractor` and `IOGA`. Those two assemblers showed a memory usage pattern, which was less influenced by the size of the data. The number of allowed threads had only a limited impact on the peak memory usage. All programs profited from a higher number of threads, if the size of the input data was increased concerning their memory and CPU usage footprint. In contrast, the disk usage was independent of the size of the input data and the number of threads for all assemblers.

Qualitative

On average, the user experience in terms of installation and running the analyses was evaluated as Good for all tools (Table 1).

However, we discovered the following slight problems:

Two minor dependencies were missing in the GetOrganelle installation instructions and there were no test data available [36]. Additionally, an issue occurred when running it on one particular *A. thaliana* data set. This was resolved after contact with the authors via GitHub [37].

The Fast-Plast installation instructions were missing some dependencies [38]. Like GetOrganelle, Fast-Plast does not offer a test data set or a tutorial, except for some example commands [36].

The ORG.Asm installation instructions did not work. We found some issues, which were probably related to the requirement of Python 3.7 [39]. A tutorial including sample data was available, but following the instructions resulted in a segmentation fault. We found a workaround for this bug and contacted the authors [40].

The main critique point of NOVOPlasty was the lack of test data and instructions. This was fixed by the authors after we contacted them [41]. Additionally, NOVOPlasty uses a custom license, where an OSI approved license would be preferable.

The chloroExtractor does come with test data and a short tutorial. However, it is currently not possible to evaluate the results of the test run as the expected results are not available [42].

The IOGA installation instructions were missing many dependencies [43]. There was also no test data or tutorial available and no license assigned to it [44]. After contacting the authors, the AGPL-3.0 license was added [45], as well as a note in the description explaining, that IOGA is no longer maintained.

Installation instructions for Chloroplast assembly protocol were also missing some dependencies. The list was updated after we contacted the authors [46]. This tool does come with an extensive tutorial and test data, but the expected outcome is not provided.

Quantitative

For a quantitative evaluation we tested the capacity of all programs to assemble chloroplasts based on different input data. Input data were either generated from existing chloroplast genomes or downloaded from sequencing repositories.

Simulated data

The different simulated data were all based on the *A. thaliana* chloroplast and core genome sequence. Some general trends could be observed: a ratio of 1:10 genome to chloroplast reads, contains too few chloroplastic reads for most tools (except Fast-Plast and GetOrganelle). A good performance for all tools was observed at a ratio of 1:100. Increasing the ratio further had no additional benefit, even if pure chloroplast reads were used (Figure 3). Using 250 bp paired read compared to 150 bp paired reads, did not produce improved results (Figure 3). In the case of Fast-Plast, the performance was even worse with the longer read length as more than a single copy of the chloroplast genome was returned.

Overall GetOrganelle and Fast-Plast were the most successful tools on the simulated data while Chloroplast assembly protocol and IOGA were unable to successfully assemble any chloroplasts out of the 16 different data sets.

Real data sets

To evaluate the performance on real data, we used publicly available short read data from NCBI's SRA with existing reference chloroplasts. We observed considerable differences for the tested assemblers, if we compared the generated alignments against the reference chloroplasts (Figure 4). The highest scores were achieved by GetOrganelle with a median of 99.8 and 210 circular assemblies out of a total of 360 assemblies that resulted in an output (Table 3). The performance of GetOrganelle was followed by Fast-Plast, NOVOPlasty, IOGA, and ORG.Asm. Fast-Plast outperformed NOVOPlasty and ORG.Asm in terms of score, producing twice as many 113 perfectly assembled chloroplast genomes (NOVOPlasty produced 58 and ORG.Asm 46 circular genomes). IOGA and Chloroplast assembly protocol were both unable to assemble a circular, single-contig genome (Table 3, Figure 5).

Consistency

Consistency was tested by re-running assemblies using the real data and comparison of the two assemblies (Figure 6). chloroExtractor was the only tool able to reproduce the same scores in all runs (Figure 6). GetOrganelle, ORG.Asm, Chloroplast assembly protocol, and IOGA generated some assemblies that were unsuccessful in one run, but produced an output in the other attempt. For these assemblers the scores were virtually identical if both runs were succesful. Both Fast-Plast and NOVOPlasty show only minor changes for the successful assemblies, leading to arrow-shaped scatter plots (Figure 6). chloroExtractor appears to be the most robust assembler, showing no deviations between the two runs.

Novel

Finally, the assembly of chloroplasts for species without a published chloroplast, was performed with the different tools. In total 49 out of 105 chloroplasts (46.7 %) with no reference sequence in CpBase were successfully assembled (Figure 8).

Almost half (44.9 %) of the successful assembled chloroplasts, were assembled by three or more different tools, while the remaining ones were only successfully generated by one or two different assemblers. Here, GetOrganelle showed the best performance and produced 15 distinct chloroplast genomes. For the assemblies obtained from multiple assemblers, we kept the GetOrganelle assemblies, after visually inspecting all assemblies using AliTV [47].

For three assemblies, that were obtained by different assemblers, but not by GetOrganelle, we kept one assembly obtained by NOVOPlasty and two from Fast-Plast. All resulting 49 sequences have been annotated with GeSeq [48]. The median number of distinct genes annotated were 80 for coding sequences, 4 for rRNA and 27 for tRNA (Table 5, Figure 10). All sequences were stored in our repository [49]. To avoid multi submissions of the same sequence to Genbank, all 49 sequences have been inspected against Genbank database via BLAST. Finally, 20 sequences were uploaded to NCBI TPA:inferential (Additional file 1: Table S1) as novel chloroplast genomes. Moreover, a search for the species name unveiled that 7 of the 20 sequences are used as ornamental plant, in folk medicine, or as crop plant.

Discussion

We compared the overall performance of the different chloroplast assemblers. Depending on the type of downstream applications, the various assessment criteria, need to be weighted differently. For example, ease of installation and use might not be a big concern if the tool is installed once and integrated in an automated pipeline. On the other hand this factor alone might prevent users from being able to use the respective tool in the first place. Similarly, computational requirements or run time might be less relevant, if the goal is to assemble a single chloroplast for further analysis, but are essential if hundreds or thousands of samples will be processed in parallel for a large scale study. Ultimately, both ease of use and computational requirements are irrelevant, if the tool is not able to successfully produce reliable assemblies.

All tools were evaluated under the assumption that they are used in their most basic form (e.g. using default parameters, no pre-processing of the data or post-processing of the result). It is important to note that any tool might perform significantly differently, if distinct parameters are specifically fine-tuned for each data set.

The best performance overall, both on simulated and real data, was achieved by `GetOrganelle`. `Fast-Plast` performed nearly as well on most data. Both tools complement each other, as one tool can achieve successful assemblies of full chloroplasts in cases where the other tool fails. This is highlighted by looking at the de-novo assemblies of chloroplasts, where `GetOrganelle` managed to generate assemblies for 15 different data sets, where no other tool succeeded and `Fast-Plast` was able to assemble 3 plastid genomes that defeated all other tools. `NOVOPlasty` was the only other tool, that could produce an assembly that was not generated with any other assembler. `Fast-Plast`, `NOVOPlasty`, and `ORG.Asm` produced the most variable results, and therefore re-running the tool after a failed attempt might be a valid strategy. `chloroExtractor` yielded only few complete chloroplast assemblies, but requires few resources and is easy to install and use. Thus `chloroExtractor` could be considered as a good option for a quick first try. Both `IOGA` and `Chloroplast assembly protocol` had unsatisfactory performances and failed to return reliable chloroplast assemblies. Nevertheless, multiple alignments of the assembled chloroplast genomes revealed some common challenges for the different tools. Those challenges include fragmented assemblies, invertions of the SSC, or a changed location of the IR (Figure 9).

We observed no phylogenetic pattern in the success rate of the assemblers (Figure 7). This indicates that the tools are generally able to reconstruct chloroplast genomes across the plant kingdom even without available reference genomes.

Guidelines for the end-user

Given these results, our recommendation is to use `GetOrganelle` as a default option for chloroplast assemblies. If `GetOrganelle` does not produce a use-able assembly, `Fast-Plast` is a valid back-up solution that might be successful. This procedure maximizes the chance of effectively and efficiently recovering the circular chloroplast genome. If both programs fail, it is recommended to try `NOVOPlasty` or manually fine-tune the parameters of the different tools. It is obviously not possible to provide general guidelines, as the exact procedure will differ for different data sets.

For an automated approach, running `GetOrganelle` and `Fast-Plast` in parallel appears to be a good trade-off between success rate and use of resources.

Ideas for future development

For further experiments, combining different components from different tools might be a promising approach. For example, read scaling from `chloroExtractor` followed by an assembly by `GetOrganelle` and finally structural resolution with `Fast-Plast` could be a promising approach, combing the respective strengths of the different tools.

Moreover, the installation issues need to be mitigated by modern software. Therefore, either containerization (docker, singularity, etc.) or install workflows (e.g. bioconda [50]) should be established by all software packages. Otherwise, the burden of the software installation might result in a low level of uptake by the research community.

A comprehensive documentation, which needs to be up-to-date and maintained, is another important feature of good tools.

All tools should improve their integrated guessing of default parameters, as these are seldom fine-tuned by users, and especially for larger screening approaches.

Finally, as sequencing technology is developing fast (e.g. PacBio or Nanopore), tools need to be updated to be able to handle this new generation of sequencing data and to not become obsolete. The hope would be that with ongoing software development and improved sequencing technologies, the generation of whole chloroplast assemblies from any species will become a routine technique.

Conclusion

WGS data are also a rich source for chlorplast assemblies. For nearly half of the analyzed data without available chloroplast genome, we could generate complete assemblies using at least one of the tools.

Still, even with simulated (i.e. "perfect") data, not all tools succeeded in generating complete chloroplast assemblies. Therefore, we determined the strengths and weaknesses of the specific tools and have provided guidelines for users. It might however be necessary to combine different methods or manually explore the parameter space. Ultimately, large scale studies reconstructing hundreds or thousands of chloroplast genomes are now feasible using the currently available tools.

Methods

Data availability

Source code for all methods used is available at [51] and archived in Zenodo under [49]. All used assembly tools are hosted on GitHub (Table 4) and are encapsulated in docker containers. That docker containers are published on dockerhub [52] and are named with a leading `benchmark_` (Additional file 1: Table S3).

To enable a fair comparison of all tools, we generated simulated sequencing data. Those simulated data sets are stored at Zenodo [53]. All resulting assemblies are available from Zenodo [54]. This study adheres to the guidelines for computational method benchmarking [55].

Tool Selection

We included tools designed for assembling chloroplasts from whole genome paired end Illumina sequencing data. As a requirement, all tools had to be available as open source software and allow execution via a command line interface. As a graphical user interface (GUI) is not suitable for automated comparisons, tools that only provided a graphical interface were also excluded. The following tools were determined to be within the scope of this study: `ORG.Asm` [27], `chloroExtractor` [33], `Fast-Plast` [56], `IOGA` [57], `NOVOPlasty` [34], `GetOrganelle` [58], and `Chloroplast assembly protocol` [59].

Some other related tools for assembling chloroplasts that did not meet our criteria and were therefore outside the scope of this study include: `Organelle PBA` [60]; `sestaton/Chloro` [61]; `Norgal` [62]; `MitoBim` [63].

`Organelle PBA` is designed for PacBio data and does not work with paired Illumina data alone. `sestaton/Chloro` fits our criteria, but is flagged as a work in progress and development and support seem to have ended two years ago. `Norgal` is a tool to extract organellar DNA from whole genome data based on a k-mer frequency approach. The final output is a set of contigs of mixed mitochondrial and plastid origin, however. The suggested approach to get a finished chloroplast genome is to run `NOVOPlasty` on the ten longest contigs. We therefore only included `NOVOPlasty` with the default settings and excluded `Norgal`. `MitoBim` is specifically designed for mitochondrial genomes. Even though there is a claim by the author that it can also be used for chloroplasts, there is no further description on how to do so [64].

Additionally, there is a protocol for the `Geneious` [65] software available [66]. However, `Geneious` is closed source and GUI based, which was not in the scope of this study. There is also another publication describing a method for assembling chloroplasts [67]. However, the link to the software is not active anymore.

Our Setup

We wanted to use a minimum of different parameter settings for all assembly programs to enable a fair comparison. Therefore, we decided to specify that all programs had to work based on two input files, representing the forward (`forward.fq`) and reverse (`reverse.fq`) sequence file of a data set in FASTQ format. Depending on the assembler, output files with different names and locations were generated. Those different files were copied and renamed to ensure that each assembly approach produced the same output file (`output.fa`). Additionally, we set an environment variable for all programs to control the number of allowed threads. All three requirements (defined input file names, defined output file name, thread number control via environment variable) were ensured by a simple wrapper script (`wrapper.sh`). Finally, for a maximum of reproducibility, all programs were bundled into individual docker images based on a central base image which provides all the required software. Those docker images were used for the recording of the consumption of computational resources on a four Intel CPU-E7 8867 v3 system offering 1 TB of RAM. Furthermore, all our docker images have been converted into singularity containers for quantitative measurement on simulated and real data sets. Singularity container were built from docker images for usage on an HPC-environment using

Singularity v.2.5.2 [68]. All singularity containers were run on Intel® Xeon® Gold 6140 Processors using a Slurm workload manager version 17.11.8 [69]. Assemblies were run on 4 threads using 10 GiB RAM with a time limit of 48 h.

Data

Simulated data

To avoid complications from sequencing errors and biological variation, we simulated perfect reads based on the *A. thaliana* (TAIR10) chloroplast and core genome assembly [70]. We used a sliding window approach with `seqkit` [71]. The exact commands are documented in `03_representative_datasets.md` in [53]. For the final simulated data sets reads are based on the TAIR10 reference genome. Different ratios between the *A. thaliana* core genome in combination with its mitochondrial sequence and the chloroplast sequence were generated (0:1, 1:10, 1:100, and 1:1000). The final data contained 30 × genome coverage and 300 × mitochondrial coverage, except the 0:1 ratio. Additionally, we generated data with different read lengths (150 bp and 250 bp). We further sampled each data set to create another version containing exactly 2 million read pairs.

Real data

We selected real data deposited at SRA [26]. We searched all data that matched `((((((("green plants"[orgn]) AND "wgs"[Strategy]) AND "illumina"[Platform]) AND "biomol dna"[Properties]) AND "paired"[Layout]) AND "random"[Selection])) AND "public"[Access]` [72]. For each species with a reference chloroplast in CpBase [73], we selected one data set. In total, this amounted to 369 data sets (Table S2) representing a broad spectrum of the green plants (Figure 7).

Novel data

To evaluate the performance for chloroplasts without a reference in CpBase [73], we sampled 105 data sets from the SRA [26] real data set described above (Additional file 1: Table S7). For each entry within that novel data set the number of lineage splits between the source taxon and the related references from CpBase was calculated according to NCBI Taxonomy [74]. The final successful assembly of 49 new chloroplasts was manually inspected and rotated to follow the expected orientation and order of chloroplast genome parts. Due to a lack of a clear definition, we followed the definition of `Fast-Plast` [75].

Evaluation Criteria

Computational Resources

We recorded the mean and the peak CPU usage, the peak memory consumption, and the size of the assembly folder for each program. As input data, we used different data sets comprising 25 000, 250 000 and 2 500 000 read pairs sampled from our simulated reads. We used our docker image setup (Additional file 1: Table S3) to run all assembly programs three times for each parameter setting. The different settings combined different input data and different number of threads to use (1, 2, 4 and 8).

Some programs want to use more CPU threads than specified, therefore, the number of CPUs available was limited using the `--cpu` option of the corresponding `docker run` command. For each assembly setting, we recorded the peak memory consumption, the CPU usage (mean and peak CPU usage) and the size of the folder where the assembly was calculated. The values of CPU and memory usage were obtained from docker. The disk usage was estimated using the GNU tool `du`. We used `GNU parallel` for queuing of the different settings [76].

Qualitative

The qualitative evaluation was mainly based on the reviewer guidelines for the Journal of Open Source Software (JOSS) [77]. To create a standard environment, all tools were tested in a fresh default installation of Ubuntu 18.04.2 running in a virtual machine (VirtualBox Version 5.2.18_Ubuntu r123745). We chose this setup instead of the docker container, because it resembles a typical user environment better than the minimal docker installation. The tools were installed according to their installation instructions and the provided tutorial or example usage was executed. During the evaluation, the following questions were asked: (1) Is the tool easy to install? (2) Is there a way to test the installation or a tutorial on how to use the tool? (3) Is there good documentation of the parameter settings? (4) Is the tool maintained (issues answered, implementation of new features)? (5) Is the tool Open Source?

These questions were subjectively answered with GOOD, OKAY or BAD, depending on the quality of the result. For example, a GOOD installation utilized an automated package or dependency management like `apt`, CRAN, docker, etc. An OKAY installation procedure provided a custom script to install everything or at least list all dependencies. A BAD installation procedure failed to list important dependencies or produced errors, that prevented a successful installation without extensive debugging.

After an initial evaluation, we contacted all authors via their GitHub or GitLab issue tracking to communicate potential flaws we found.

Quantitative

For each data set and assembler the generated chloroplast genome was compared to the respective reference genome using a pairwise alignment obtained with `minimap2` v2.16 [78]. Based on theses alignments a score was calculated (eq. (1)). The assemblies were scored on a scale from 0 to 100, with 100 being the best and 0 the worst possible score. Four different metrics were incorporated, each contributing a quarter to the total score: completeness; correctness; repeat resolution; continuity. These metrics are similar in concept by those used in the Assemblathon 2 project: coverage; validity; multiplicity; parsimony [79].

The completeness was estimated as the coverage of the assembled chloroplast genome versus the reference genome (cov_{ref}). It represents how many bases of the query genome can be mapped to its respective reference genome. Secondly, we mapped the reference genome against the query. The coverage of the reference genome (cov_{qry}) was used as a measurement of the correctness of the assembly. The repeat resolution was estimated from the size difference of the assembly and the reference genome ($min\left\{\frac{cov_{qry}}{cov_{ref}}, \frac{cov_{ref}}{cov_{qry}}\right\}$), leading to values between 0 and 1. The fourth metric used was the continuity, represented by the number of contigs.

A perfect score was achieved if one circular chromosome was assembled, while the score became worse as the number of contigs increased.

$$score = \frac{1}{4} \cdot \left(cov_{ref} + cov_{qry} + min \left\{ \frac{cov_{qry}}{cov_{ref}}, \frac{cov_{ref}}{cov_{qry}} \right\} + \frac{1}{n_{contigs}} \right) \cdot 100 \qquad (1)$$

Success

For assemblies with reference sequence we defined success as reaching a score of 99 or higher. For the novel chloroplasts our score could not be calculated. The following criteria were instead selected to classify a novel chloroplast as success: single contig of length at least 130 kbp and an IR region of at least 17 kbp. These cutoffs were selected as they produced the highest f-score on the real data set where true assignment (success/failure) was assumed based on the score (success if score 99 or higher).

Consistency

To ensure consistency of the obtained results, we rerun and re-evaluated all the assemblies. The resulting assemblies were scored again as described and the scores of the first and the second run were compared to each other This information was important to assess the robustness of the different programs.

Competing interests

Authors SP, NT, FF, and MJA are developers of `chloroExtractor`, one of the tools benchmarked in this article. JF, NT, and MJA are affiliated with the for-profit organization AnaLife Data Science.

Author's contributions

MJA and FF conceived the project and supervised the findings. SP and FF created the docker images. NT performed the qualitative analysis for all assemblers. MJA prepared the simulated and real data sets. JAF assembled the real data sets. FF ran the performance assemblies on the simulated data sets. All authors developed the score model. JAF and MJA implemented the score model and prepared the figures. All authors discussed the results and contributed to the final manuscript.

Acknowledgement

We thank Brooke Morriswood for proof-reading and English editing of the manuscript

Funding

Not applicable.

Ethics approval

Not applicable.

Availability of data and materials

The supplemental material is available from Zenodo [80]. The simulated data set is available from Zenodo [53]. All program code is available via Zenodo [49] or from Github [51]. The input data sets can be generated using the raw reads from NCBI SRA (links for each data set in Table S2). The resulting assemblies are available from Zenodo [54].

Author details

[1]Center for Computational and Theoretical Biology, University of Würzburg, Campus Hubland Nord, 97074 Würzburg, Germany. [2]AnaLife Data Science, Wiesengrund 16, 97295 Waldbrunn Würzburg, Germany. [3]Department of Bioinformatics, University of Würzburg, Biozentrum, Am Hubland, 97074, Würzburg, Germany. [4]Chair of Cellular and Molecular Imaging, Comprehensive Heart Failure Center, University Hospital Würzburg, Josef-Schneider-Str. 2, 97080 Würzburg, Germany. [5]@IIMOG. [6]Fraunhofer IME-BR, Ohlebergsweg 12, 35392 Gießen, Germany. [7]Bioinformatics Core Facility of the University of Gießen, Heinrich-Buff-Ring 58, 35392 Gießen, Germany. [8]@frank_foerster.

References

1. Palmer, J.D.: Comparative organization of chloroplast genomes. Annual Review of Genetics **19**(1), 325–354 (1985). doi:10.1146/annurev.ge.19.120185.001545. PMID: 3936406. https://doi.org/10.1146/annurev.ge.19.120185.001545
2. Ohyama, K., Fukuzawa, H., Kohchi, T., Shirai, H., Sano, T., Sano, S., Umesono, K., Shiki, Y., Takeuchi, M., Chang, Z., Aota, S.-i., Inokuchi, H., Ozeki, H.: Chloroplast gene organization deduced from complete sequence of liverwort marchantia polymorpha chloroplast DNA **322**(6079), 572. doi:10.1038/322572a0. Accessed 2019-05-20

3. Shinozaki, K., Ohme, M., Tanaka, M., Wakasugi, T., Hayashida, N., Matsubayashi, T., Zaita, N., Chunwongse, J., Obokata, J., Yamaguchi-Shinozaki, K., Ohto, C., Torazawa, K., Meng, B.Y., Sugita, M., Deno, H., Kamogashira, T., Yamada, K., Kusuda, J., Takaiwa, F., Kato, A., Tohdoh, N., Shimada, H., Sugiura, M.: The complete nucleotide sequence of the tobacco chloroplast genome: its gene organization and expression **5**(9), 2043–2049. doi:10.1002/j.1460-2075.1986.tb04464.x. Accessed 2019-05-20
4. Wicke, S., Schneeweiss, G.M., dePamphilis, C.W., Müller, K.F., Quandt, D.: The evolution of the plastid chromosome in land plants: gene content, gene order, gene function **76**(3), 273–297. doi:10.1007/s11103-011-9762-4. Accessed 2019-05-16
5. Green, B.R.: Chloroplast genomes of photosynthetic eukaryotes **66**(1), 34–44. doi:10.1111/j.1365-313X.2011.04541.x. Accessed 2019-05-16
6. Martín, M., Sabater, B.: Plastid ndh genes in plant evolution **48**(8), 636–645. doi:10.1016/j.plaphy.2010.04.009. Accessed 2016-06-12
7. Xiao-Ming, Z., Junrui, W., Li, F., Sha, L., Hongbo, P., Lan, Q., Jing, L., Yan, S., Weihua, Q., Lifang, Z., Yunlian, C., Qingwen, Y.: Inferring the evolutionary mechanism of the chloroplast genome size by comparing whole-chloroplast genome sequences in seed plants **7**(1), 1555. doi:10.1038/s41598-017-01518-5. Accessed 2019-05-16
8. Kress, W.J., Wurdack, K.J., Zimmer, E.A., Weigt, L.A., Janzen, D.H.: Use of DNA barcodes to identify flowering plants **102**(23), 8369–8374. doi:10.1073/pnas.0503123102. Accessed 2017-08-29
9. Hollingsworth, P.M., Forrest, L.L., Spouge, J.L., Hajibabaei, M., Ratnasingham, S., van der Bank, M., Chase, M.W., Cowan, R.S., Erickson, D.L., Fazekas, A.J., Graham, S.W., James, K.E., Kim, K.-J., Kress, W.J., Schneider, H., van AlphenStahl, J., Barrett, S.C.H., van den Berg, C., Bogarin, D., Burgess, K.S., Cameron, K.M., Carine, M., Chacón, J., Clark, A., Clarkson, J.J., Conrad, F., Devey, D.S., Ford, C.S., Hedderson, T.A.J., Hollingsworth, M.L., Husband, B.C., Kelly, L.J., Kesanakurti, P.R., Kim, J.S., Kim, Y.-D., Lahaye, R., Lee, H.-L., Long, D.G., Madriñán, S., Maurin, O., Meusnier, I., Newmaster, S.G., Park, C.-W., Percy, D.M., Petersen, G., Richardson, J.E., Salazar, G.A., Savolainen, V., Seberg, O., Wilkinson, M.J., Yi, D.-K., Little, D.P.: A DNA barcode for land plants **106**(31), 12794–12797. doi:10.1073/pnas.0905845106. Accessed 2019-05-21
10. de Vere, N., Rich, T.C.G., Trinder, S.A., Long, C.: DNA barcoding for plants **1245**, 101–118. doi:10.1007/978-1-4939-1966-6_8
11. Bell, K.L., Burgess, K.S., Okamoto, K.C., Aranda, R., Brosi, B.J.: Review and future prospects for DNA barcoding methods in forensic palynology **21**, 110–116. doi:10.1016/j.fsigen.2015.12.010. Accessed 2017-09-24
12. Deiner, K., Bik, H.M., Mächler, E., Seymour, M., Lacoursière-Roussel, A., Altermatt, F., Creer, S., Bista, I., Lodge, D.M., Vere, N.d., Pfrender, M.E., Bernatchez, L.: Environmental DNA metabarcoding: Transforming how we survey animal and plant communities **26**(21), 5872–5895. doi:10.1111/mec.14350. Accessed 2019-05-21
13. Martin, W., Rujan, T., Richly, E., Hansen, A., Cornelsen, S., Lins, T., Leister, D., Stoebe, B., Hasegawa, M., Penny, D.: Evolutionary analysis of arabidopsis, cyanobacterial, and chloroplast genomes reveals plastid phylogeny and thousands of cyanobacterial genes in the nucleus **99**(19), 12246–12251. doi:10.1073/pnas.182432999. Accessed 2019-05-20
14. Timmis, J.N., Ayliffe, M.A., Huang, C.Y., Martin, W.: Endosymbiotic gene transfer: organelle genomes forge eukaryotic chromosomes **5**(2), 123–135. doi:10.1038/nrg1271
15. Nevill, P.G., Howell, K.A., Cross, A.T., Williams, A.V., Zhong, X., Tonti-Filippini, J., Boykin, L.M., Dixon, K.W., Small, I.: Plastome-wide rearrangements and gene losses in carnivorous droseraceae **11**(2), 472–485. doi:10.1093/gbe/evz005. Accessed 2019-05-16
16. Su, H.-J., Barkman, T.J., Hao, W., Jones, S.S., Naumann, J., Skippington, E., Wafula, E.K., Hu, J.-M., Palmer, J.D., dePamphilis, C.W.: Novel genetic code and record-setting AT-richness in the highly reduced plastid genome of the holoparasitic plant balanophora **116**(3), 934–943. doi:10.1073/pnas.1816822116. Accessed 2019-05-20
17. Wang, W., Schalamun, M., Morales-Suarez, A., Kainer, D., Schwessinger, B., Lanfear, R.: Assembly of chloroplast genomes with long- and short-read data: a comparison of approaches using eucalyptus pauciflora as a test case. BMC genomics **19**(1), 977–977 (2018). doi:10.1186/s12864-018-5348-8. 30594129[pmid]
18. Corriveau, J.L., Coleman, A.W.: Rapid screening method to detect potential biparental inheritance of plastid dna and results for over 200 angiosperm species. American Journal of Botany **75**(10), 1443–1458 (1988)
19. Chat, J., Decroocq, S., Decroocq, V., Petit, R.J.: A Case of Chloroplast Heteroplasmy in Kiwifruit (Actinidia deliciosa) That Is Not Transmitted During Sexual Reproduction. Journal of Heredity **93**(4), 293–300 (2002). doi:10.1093/jhered/93.4.293. http://oup.prod.sis.lan/jhered/article-pdf/93/4/293/6454216/293.pdf
20. Scarcelli, N., Mariac, C., Couvreur, T.L.P., Faye, A., Richard, D., Sabot, F., Berthouly-Salazar, C., Vigouroux, Y.: Intra-individual polymorphism in chloroplasts from ngs data: where does it come from and how to handle it? Molecular Ecology Resources **16**(2), 434–445 (2016). doi:10.1111/1755-0998.12462. https://onlinelibrary.wiley.com/doi/pdf/10.1111/1755-0998.12462
21. Schmuths, H., Meister, A., Horres, R., Bachmann, K.: Genome size variation among accessions of arabidopsis thaliana. Annals of Botany **93**(3), 317–321 (2004)
22. Cao, J., Schneeberger, K., Ossowski, S., Günther, T., Bender, S., Fitz, J., Koenig, D., Lanz, C., Stegle, O., Lippert, C., *et al.*: Whole-genome sequencing of multiple arabidopsis thaliana populations. Nature genetics **43**(10), 956 (2011)
23. Sato, S., Nakamura, Y., Kaneko, T., Asamizu, E., Tabata, S.: Complete structure of the chloroplast genome of arabidopsis thaliana. DNA research **6**(5), 283–290 (1999)
24. Kumar, R.A., Oldenburg, D.J., Bendich, A.J.: Changes in DNA damage, molecular integrity, and copy number for plastid DNA and mitochondrial DNA during maize development. Journal of Experimental Botany **65**(22), 6425–6439 (2014). doi:10.1093/jxb/eru359. http://oup.prod.sis.lan/jxb/article-pdf/65/22/6425/16935653/eru359.pdf
25. Bendich, A.J.: Why do chloroplasts and mitochondria contain so many copies of their genome? BioEssays **6**(6),

279–282 (1987). doi:10.1002/bies.950060608. https://onlinelibrary.wiley.com/doi/pdf/10.1002/bies.950060608

26. Leinonen, R., Sugawara, H., Shumway, M., on behalf of the International Nucleotide Sequence Database Collaboration: The Sequence Read Archive. Nucleic Acids Research **39**(suppl_1), 19–21 (2010). doi:10.1093/nar/gkq1019. http://oup.prod.sis.lan/nar/article-pdf/39/suppl_1/D19/7624335/gkq1019.pdf
27. Coissac, E., Hollingsworth, P.M., Lavergne, S., Taberlet, P.: From barcodes to genomes: extending the concept of DNA barcoding **25**(7), 1423–1428. doi:10.1111/mec.13549. Accessed 2019-05-16
28. Daniell, H., Lin, C.-S., Yu, M., Chang, W.-J.: Chloroplast genomes: diversity, evolution, and applications in genetic engineering **17**. doi:10.1186/s13059-016-1004-2. Accessed 2019-05-20
29. Tonti-Filippini, J., Nevill, P.G., Dixon, K., Small, I.: What can we do with 1000 plastid genomes? **90**(4), 808–818. doi:10.1111/tpj.13491. Accessed 2018-01-26
30. Twyford, A.D., Ness, R.W.: Strategies for complete plastid genome sequencing **17**(5), 858–868. doi:10.1111/1755-0998.12626. Accessed 2018-01-26
31. Vinga, S., Carvalho, A.M., Francisco, A.P., Russo, L.M., Almeida, J.S.: Pattern matching through chaos game representation: bridging numerical and discrete data structures for biological sequence analysis. Algorithms for molecular biology : AMB **7**(1), 10–10 (2012). doi:10.1186/1748-7188-7-10. 22551152[pmid]
32. Chan, C.X., Ragan, M.A.: Next-generation phylogenomics. Biology direct **8**, 3–3 (2013). doi:10.1186/1745-6150-8-3. 23339707[pmid]
33. J Ankenbrand, M., Pfaff, S., Terhoeven, N., Qureischi, M., Gündel, M., L. Weiß, C., Hackl, T., Förster, F.: chloroExtractor: extraction and assembly of the chloroplast genome from whole genome shotgun data. The Journal of Open Source Software **3**(21), 464 (2018). doi:10.21105/joss.00464. Accessed 2019-05-22TZ
34. Dierckxsens, N., Mardulyn, P., Smits, G.: NOVOPlasty: de novo assembly of organelle genomes from whole genome data **45**(4), 18–18. doi:10.1093/nar/gkw955. Accessed 2018-01-26
35. Goodwin, S., McPherson, J.D., McCombie, W.R.: Coming of age: ten years of next-generation sequencing technologies. Nature Reviews Genetics **17**(6), 333–351 (2016). doi:10.1038/nrg.2016.49
36. `GetOrganelle` ISSUE 10. `https://github.com/Kinggerm/GetOrganelle/issues/10`
37. `GetOrganelle` ISSUE 11. `https://github.com/Kinggerm/GetOrganelle/issues/11`
38. `Fast-Plast` ISSUE 33. `https://github.com/mrmckain/Fast-Plast/issues/33`
39. `ORG.Asm` ISSUE 59. `https://git.metabarcoding.org/org-asm/org-asm/issues/59`
40. `ORG.Asm` ISSUE 57. `https://git.metabarcoding.org/org-asm/org-asm/issues/57`
41. `NOVOPlasty` ISSUE 82. `https://github.com/ndierckx/NOVOPlasty/issues/82`
42. `chloroExtractor` ISSUE 139. `https://github.com/chloroExtractorTeam/chloroExtractor/issues/139`
43. `IOGA` ISSUE 12. `https://github.com/holmrenser/IOGA/issues/12`
44. `IOGA` ISSUE 13. `https://github.com/holmrenser/IOGA/issues/13`
45. `IOGA` ISSUE 11. `https://github.com/holmrenser/IOGA/issues/11`
46. `Chloroplast assembly protocol` ISSUE 5. `https://github.com/eead-csic-compbio/chloroplast_assembly_protocol/issues/5`
47. Ankenbrand, M.J., Hohlfeld, S., Hackl, T., Förster, F.: Alitv—interactive visualization of whole genome comparisons. PeerJ Computer Science **3**, 116 (2017). doi:10.7717/peerj-cs.116
48. Tillich, M., Lehwark, P., Pellizzer, T., Ulbricht-Jones, E.S., Fischer, A., Bock, R., Greiner, S.: Geseq—versatile and accurate annotation of organelle genomes. Nucleic Acids Research **45**(W1), 6–11 (2017). doi:10.1093/nar/gkx391. http://oup.prod.sis.lan/nar/article-pdf/45/W1/W6/18137544/gkx391.pdf
49. Förster, F., Ankenbrand, M.J.: chloroExtractorTeam/benchmark: Benchmark container setup v2.0.1. Zenodo (2019). doi:10.5281/zenodo.2628061
50. Grüning, B., , Dale, R., Sjödin, A., Chapman, B.A., Rowe, J., Tomkins-Tinch, C.H., Valieris, R., Köster, J.: Bioconda: sustainable and comprehensive software distribution for the life sciences. Nature Methods **15**(7), 475–476 (2018). doi:10.1038/s41592-018-0046-7
51. GitHub Repository for Benchmark Project. `https://github.com/chloroExtractorTeam/benchmark`
52. Docker Hub Group for Benchmark Project. `https://cloud.docker.com/u/chloroextractorteam/`
53. Ankenbrand, M.J., Förster, F.: Simulated *Arabidopsis thaliana* sequencing datasets for chloroplast assembler benchmarking. Zenodo (2019). doi:10.5281/zenodo.2622875
54. Ankenbrand, M.J., Förster, F.: Assemblies based on real data sets for the manuscript "The landscape of chloroplast genome assembly tools". Zenodo (2019). doi:10.5281/zenodo.3240535
55. Weber, L.M., Saelens, W., Cannoodt, R., Soneson, C., Hapfelmeier, A., Gardner, P., Boulesteix, A.-L., Saeys, Y., Robinson, M.D.: Essential guidelines for computational method benchmarking. 1812.00661. Accessed 2019-03-14
56. McKain, M., Afinit: Mrmckain/Fast-Plast: Fast-Plast V.1.2.6. Zenodo (2017). doi:10.5281/zenodo.973887. `https://zenodo.org/record/973887` Accessed 2019-05-22TZ
57. Bakker, F.T., Lei, D., Yu, J., Mohammadin, S., Wei, Z., van de Kerke, S., Gravendeel, B., Nieuwenhuis, M., Staats, M., Alquezar-Planas, D.E., Holmer, R.: Herbarium genomics: plastome sequence assembly from a range of herbarium specimens using an Iterative Organelle Genome Assembly pipeline. Biological Journal of the Linnean Society **117**(1), 33–43 (2016). doi:10.1111/bij.12642. Accessed 2019-05-22TZ
58. Jin, J.-J., Yu, W.-B., Yang, J.-B., Song, Y., Yi, T.-S., Li, D.-Z.: GetOrganelle: a simple and fast pipeline for de novo assembly of a complete circular chloroplast genome using genome skimming data. bioRxiv (2018). doi:10.1101/256479. Accessed 2019-05-22TZ
59. Sancho, R., Cantalapiedra, C.P., López-Alvarez, D., Gordon, S.P., Vogel, J.P., Catalán, P., Contreras-Moreira, B.: Comparative plastome genomics and phylogenomics of *Brachypodium* : flowering time signatures, introgression and recombination in recently diverged ecotypes. New Phytologist **218**(4), 1631–1644 (2018). doi:10.1111/nph.14926. Accessed 2019-05-22TZ
60. Soorni, A., Haak, D., Zaitlin, D., Bombarely, A.: Organelle_pba, a pipeline for assembling chloroplast and mitochondrial genomes from pacbio dna sequencing data. BMC Genomics **18**(1), 49 (2017). doi:10.1186/s12864-016-3412-9
61. Staton, E.: Automated Chloroplast Genome Assembly. Accessed: 2019-05-28.

ttps://github.com/sestaton/Chloro

62. Al-Nakeeb, K., Petersen, T.N., Sicheritz-Pontén, T.: Norgal: extraction and de novo assembly of mitochondrial dna from whole-genome sequencing data. BMC Bioinformatics **18**(1), 510 (2017). doi:10.1186/s12859-017-1927-y
63. Hahn, C., Bachmann, L., Chevreux, B.: Reconstructing mitochondrial genomes directly from genomic next-generation sequencing reads—a baiting and iterative mapping approach. Nucleic Acids Research **41**(13), 129–129 (2013). doi:10.1093/nar/gkt371. http://oup.prod.sis.lan/nar/article-pdf/41/13/e129/25367803/gkt371.pdf
64. MITObim Issue 16. Accessed: 2019-05-27. https://github.com/chrishah/MITObim/issues/16
65. Geneious Prime. Accessed: 2019-05-28. https://www.geneious.com
66. Gibbs, M.D.: De Novo Assembly and Reconstruction of Complete Circular Chloroplast Genomes Using Geneious Prime. Accessed: 2019-05-28. https://assets.geneious.com/documentation/geneious/App+Note+-+De+Novo+Assembly+of+Chloroplasts.pdf
67. Izan, S., Esselink, D., Visser, R.G.F., Smulders, M.J.M., Borm, T.: De novo assembly of complete chloroplast genomes from non-model species based on a k-mer frequency-based selection of chloroplast reads from total dna sequences. Frontiers in Plant Science **8**, 1271 (2017). doi:10.3389/fpls.2017.01271
68. Kurtzer, G.M., Sochat, V., Bauer, M.W.: Singularity: Scientific containers for mobility of compute. PloS one **12**(5), 0177459 (2017)
69. Jette, M.A., Yoo, A.B., Grondona, M.: Slurm: Simple linux utility for resource management. In: In Lecture Notes in Computer Science: Proceedings of Job Scheduling Strategies for Parallel Processing (JSSPP) 2003, pp. 44–60. Springer, ??? (2002)
70. Berardini, T.Z., Reiser, L., Li, D., Mezheritsky, Y., Muller, R., Strait, E., Huala, E.: The *Arabidopsis* information resource: Making and mining the "gold standard"annotated reference plant genome. genesis **53**(8), 474–485 (2015). doi:10.1002/dvg.22877. https://onlinelibrary.wiley.com/doi/pdf/10.1002/dvg.22877
71. Shen, W., Le, S., Li, Y., Hu, F.: Seqkit: A cross-platform and ultrafast toolkit for fasta/q file manipulation. PLOS ONE **11**(10), 1–10 (2016). doi:10.1371/journal.pone.0163962
72. SRA Search Term. Accessed: 2019-03-28. https://www.ncbi.nlm.nih.gov/sra/?term=(((((((%22green+plants%22%5Borgn%5D)+AND+%22wgs%22%5BStrategy%5D)+AND+%22illumina%22%5BPlatform%5D)+AND+%22biomol+dna%22%5BProperties%5D)+AND+%22paired%22%5BLayout%5D)+AND+%22random%22%5BSelection%5D))+AND+%22public%22%5BAccess%5D
73. Rocaps Lab: CpBase. Accessed: 2019-04-01, Version: 8/20/2017. http://rocaplab.ocean.washington.edu/old_website/tools/cpbase
74. National Center for Biotechnology Information: NCBI Taxonomy. Accessed: 2019-10-01. https://www.ncbi.nlm.nih.gov/Taxonomy/taxonomyhome.html/
75. Fast-Plast ORIENTATION. https://github.com/mrmckain/Fast-Plast/issues/22
76. Tange, O.: Gnu parallel - the command-line power tool. ;login: The USENIX Magazine **36**(1), 42–47 (2011). doi:10.5281/zenodo.16303
77. JOSS Review Criteria. Accessed: 2019-05-15. https://joss.readthedocs.io/en/latest/review_criteria.html
78. Li, H.: Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics **34**(18), 3094–3100 (2018)
79. Bradnam, K.R., Fass, J.N., Alexandrov, A., Baranay, P., Bechner, M., Birol, I., Boisvert, S., Chapman, J.A., Chapuis, G., Chikhi, R., Chitsaz, H., Chou, W.-C., Corbeil, J., Del Fabbro, C., Docking, T.R., Durbin, R., Earl, D., Emrich, S., Fedotov, P., Fonseca, N.A., Ganapathy, G., Gibbs, R.A., Gnerre, S., Godzaridis, E., Goldstein, S., Haimel, M., Hall, G., Haussler, D., Hiatt, J.B., Ho, I.Y., Howard, J., Hunt, M., Jackman, S.D., Jaffe, D.B., Jarvis, E.D., Jiang, H., Kazakov, S., Kersey, P.J., Kitzman, J.O., Knight, J.R., Koren, S., Lam, T.-W., Lavenier, D., Laviolette, F., Li, Y., Li, Z., Liu, B., Liu, Y., Luo, R., MacCallum, I., MacManes, M.D., Maillet, N., Melnikov, S., Naquin, D., Ning, Z., Otto, T.D., Paten, B., Paulo, O.S., Phillippy, A.M., Pina-Martins, F., Place, M., Przybylski, D., Qin, X., Qu, C., Ribeiro, F.J., Richards, S., Rokhsar, D.S., Ruby, J.G., Scalabrin, S., Schatz, M.C., Schwartz, D.C., Sergushichev, A., Sharpe, T., Shaw, T.I., Shendure, J., Shi, Y., Simpson, J.T., Song, H., Tsarev, F., Vezzi, F., Vicedomini, R., Vieira, B.M., Wang, J., Worley, K.C., Yin, S., Yiu, S.-M., Yuan, J., Zhang, G., Zhang, H., Zhou, S., Korf, I.F.: Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species. GigaScience **2**(1) (2013). doi:10.1186/2047-217X-2-10. http://oup.prod.sis.lan/gigascience/article-pdf/2/1/2047-217X-2-10/25511257/13742_2013_article_29.pdf
80. Ankenbrand, M.J., Förster, F.: Supplemental data for the manuscript "The landscape of chloroplast genome assembly tools". Zenodo (2019). doi:10.5281/zenodo.3241963
81. Lex, A., Gehlenborg, N., Strobelt, H., Vuillemot, R., Pfister, H.: Upset: visualization of intersecting sets. IEEE transactions on visualization and computer graphics **20**(12), 1983–1992 (2014)
82. Federhen, S.: The ncbi taxonomy database. Nucleic acids research **40**(D1), 136–143 (2011)
83. Letunic, I., Bork, P.: Interactive tree of life (iTOL) v4: recent updates and new developments. Nucleic Acids Research (2019). doi:10.1093/nar/gkz239

Figures

Tables

Additional Files

Additional file 1 — supplemental data

Supplementary data contain a complete list of all real data sets used in this study. Additionally, a table with more details on the used docker images and the detailed results of the performance measurement are included. The file is available at [80].

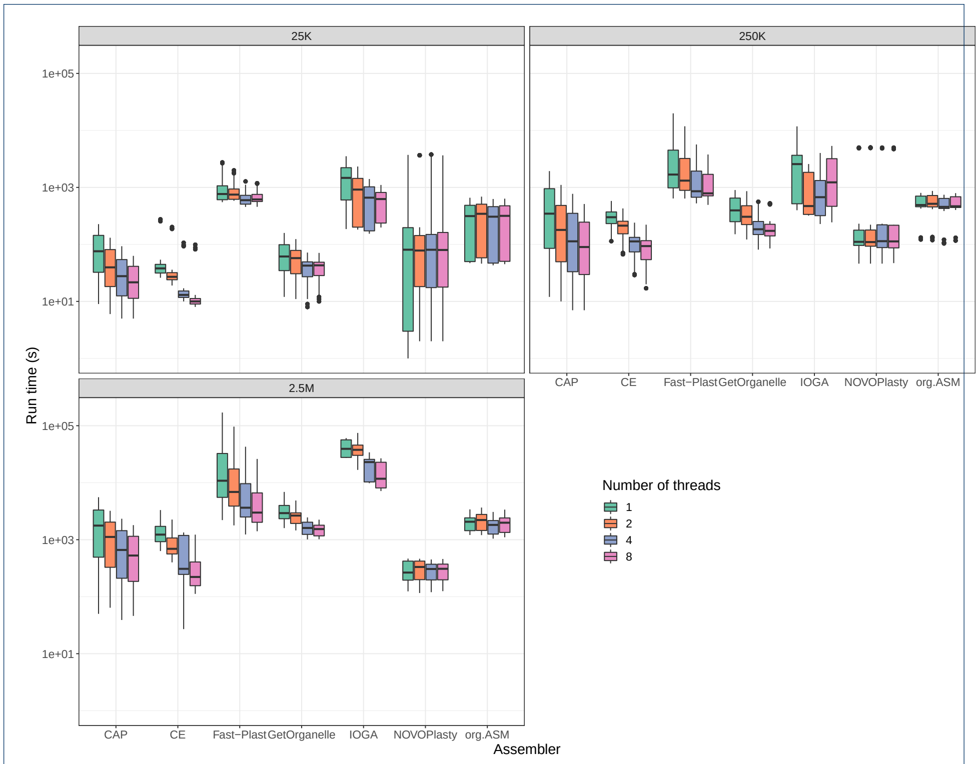

Figure 1 Computation time depending on number of threads and size of input data The boxplots show the differences in demand of CPU time for different number of threads and input data size for the seven different assemblers

Table 1 Overview of the results of the qualitative usability evaluation Each tool could score Good, Okay or Bad in each of the categories.

Tool	Installation	Test/Tutorial	Documentation	Maintenance	FLOSS
`chloroExtractor`	Good	Good	Good	Good	Good
`Chloroplast assembly protocol`	Okay	Good	Okay	Good	Good
`Fast-Plast`	Bad	Okay	Good	Good	Good
`GetOrganelle`	Okay	Okay	Good	Good	Good
`IOGA`	Bad	Bad	Okay	Bad	Good
`NOVOPlasty`	Good	Good	Good	Good	Okay
`ORG.Asm`	Bad	Bad	Okay	Good	Good

Table 2 Scores of assemblies of simulated data

	data set	CAP	CE	Fast-Plast	GetOrganelle	IOGA	NOVOPlasty	org.ASM
1	sim_150bp.0-1	79.10	100.00	99.48	100.00		91.52	100.00
2	sim_150bp.0-1.2M	79.10	100.00	99.72	100.00	79.10	91.52	91.50
3	sim_150bp.1-10		56.44	100.00	76.98		91.52	78.00
4	sim_150bp.1-10.2M			99.97	100.00		91.52	82.72
5	sim_150bp.1-100	75.72	100.00	99.48	100.00	66.09	91.52	91.50
6	sim_150bp.1-100.2M		100.00	99.47	100.00		100.00	100.00
7	sim_150bp.1-1000	79.10		99.72	100.00		91.52	100.00
8	sim_150bp.1-1000.2M	79.10	100.00	99.72	100.00		91.52	100.00
9	sim_250bp.0-1	79.10	100.00	93.82	100.00		91.52	91.50
10	sim_250bp.0-1.2M	79.10	100.00	93.83	100.00		91.52	91.50
11	sim_250bp.1-10		54.98	68.45	78.89	52.71	91.52	40.20
12	sim_250bp.1-10.2M			93.00	100.00	52.67	87.40	40.20
13	sim_250bp.1-100	72.81	100.00	93.82	100.00		87.40	100.00
14	sim_250bp.1-100.2M		100.00	93.83	100.00		87.40	100.00
15	sim_250bp.1-1000	79.10	21.30	93.83	100.00	76.96	91.52	91.50
16	sim_250bp.1-1000.2M	79.10	100.00	93.83	100.00	67.55	87.40	100.00

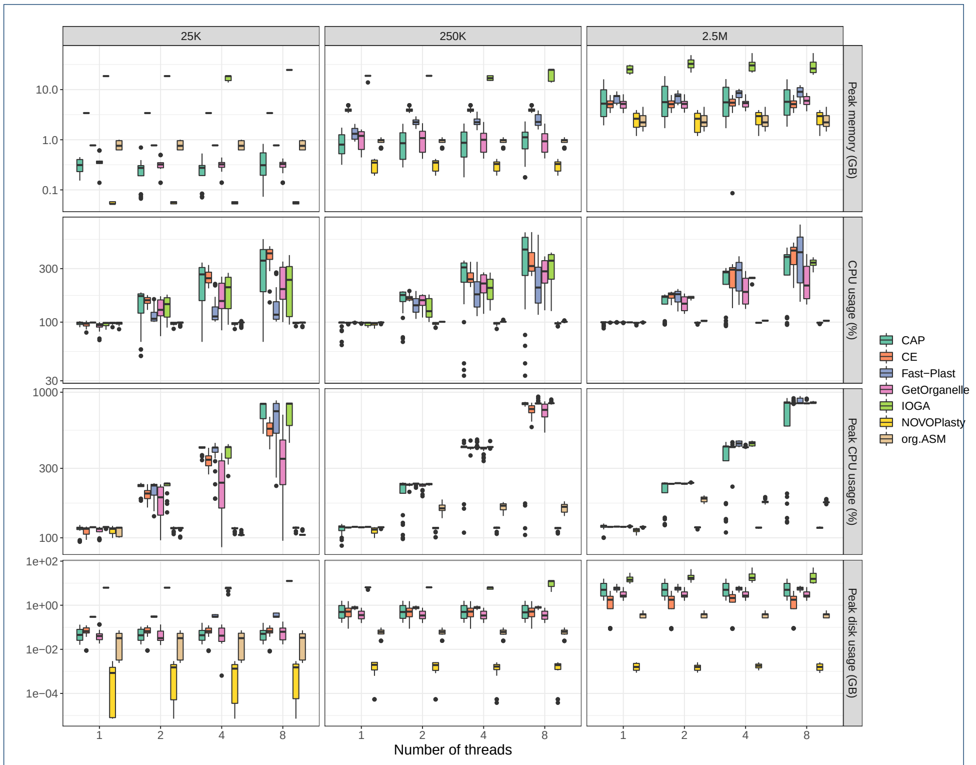

Figure 2 Performance metrics Boxplots depicting the demand of CPU and RAM and disk space needed depending on the assembler, input data size and number of threads

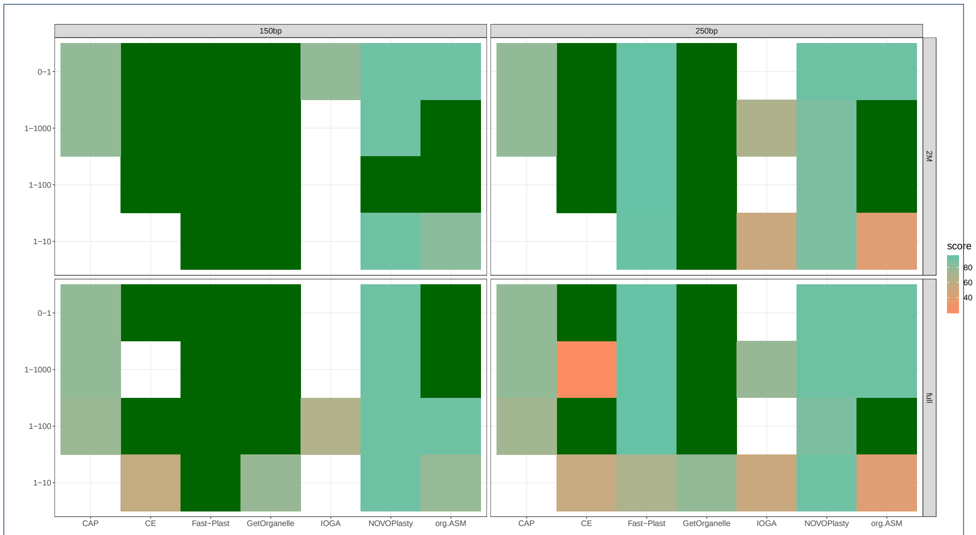

Figure 3 Score of assemblies on simulated data Results of assemblies from simulated data sets. Color scale of the tiles represents the score

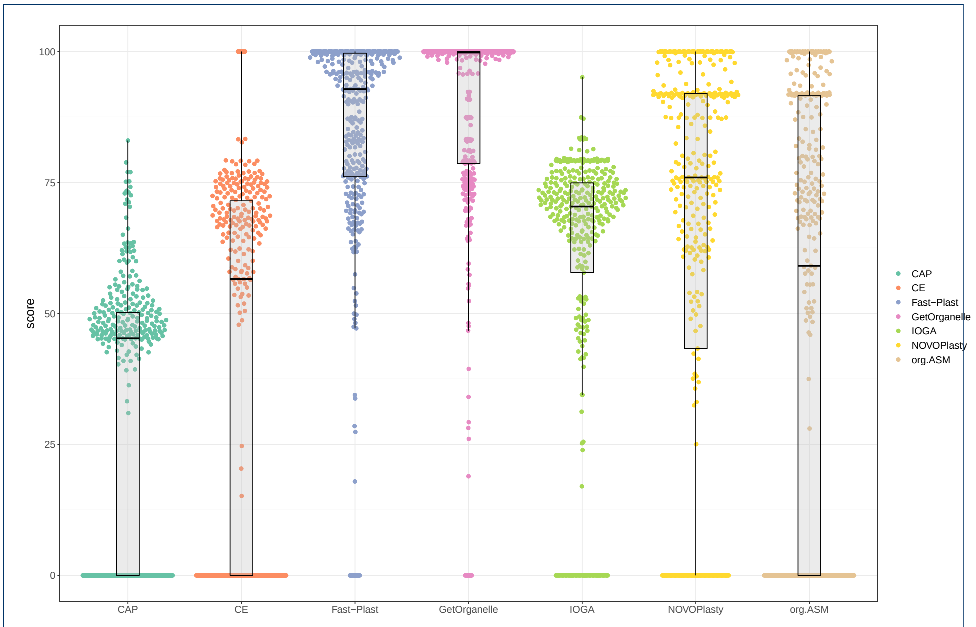

Figure 4 Results of scoring of the seven assemblers The box- and swarplots depict the results of the scoring algorithm we used. For the different assemblers. The whiskers of boxplots indicate the 1.5 x interquartile range.

Table 3 Mean scores of chloroplast genome assemblers

	assembler	Median	IQR	N_perfect	N_tot
1	CAP	45.25	50.19	0	369
2	CE	56.55	71.50	14	369
3	Fast-Plast	92.80	23.59	113	369
4	GetOrganelle	99.83	20.94	210	360
5	IOGA	71.10	11.21	0	338
6	NOVOPlasty	75.95	48.69	58	369
7	org.ASM	67.35	91.69	46	348

Table 4 Tools and version information used in our benchmark setup All tools are wrapped into docker containers and stored on dockerhub [52]. The corresponding tags and SHA256 checksums are reported in Additional file 1: Table S3

Tool	Source Repository	Commit used for benchmarking
GetOrganelle	https://github.com/Kinggerm/GetOrganelle.git	587c1c51c34e270eb9178a42a77a5150157e6925
IOGA	https://github.com/holmrenser/IOGA.git	c460ea9d9fe176fec2bd76d369b0cbb36793b2bf
NOVOPlasty	https://github.com/ndierckx/NOVOPlasty.git	6af0894f8ea1d76a1b71df9cb762cf6e48dceac1
chloroExtractor	https://github.com/chloroExtractorTeam/chloroExtractor.git	87364e48ec84a3f6ee91fc8d995b0bda5a0fa82d
Chloroplast assembly protocol	https://github.com/eead-csic-compbio/chloroplast_assembly_protocol.git	250d16ac02005d6a5939bf182b3d2995d0e88229
Fast-Plast	https://github.com/mrmckain/Fast-Plast.git	7e32b2e797fd1f49d32d6559e8345afefbaff803
ORG.Asm	https://git.metabarcoding.org/org-asm/org-asm.git	830313acae3ca773b63f6bea9fc6d017e021bde5

Table 5 Number of distinct features in novel chloroplast genomes The distribution (mean, standard deviation (SD), medain, interquartile range (IQR)) of feature types tRNA, rRNA, and coding sequence (CDS) are listed separately.

Feature	Mean	SD	Median	IQR
CDS	79.1	3.45	80	2
rRNA	4.2	0.37	4	0
tRNA	26.7	0.97	27	0

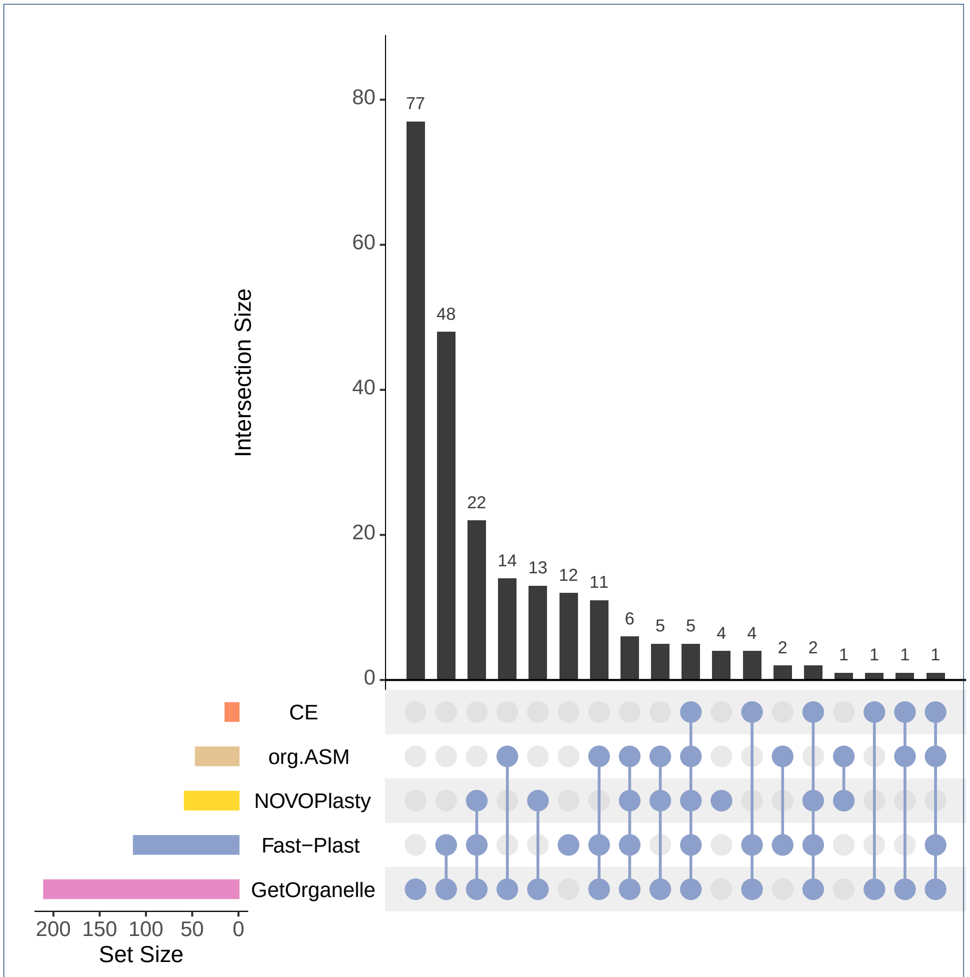

Figure 5 Upset plot [81] comparing success of assemblers on the real data sets The plot shows the intersection of success $(score > 99)$ between assemblers. For 69 data sets only `GetOrganelle` was able to obtain a complete chloroplast. 43 were successful with both `GetOrganelle` and `Fast-Plast` and so on

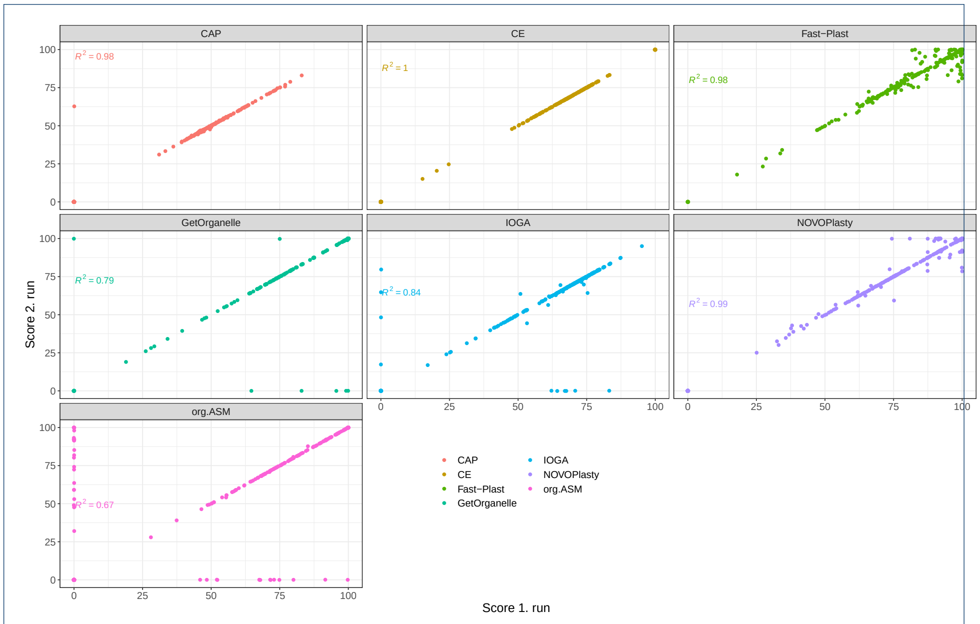

Figure 6 Scores between two repeated runs for consistency testing The scatter plots depicts the scores of the 1. runs x-axis versus the scores of the 2. run y-axis of the data sets that were selected for re-evaluation.

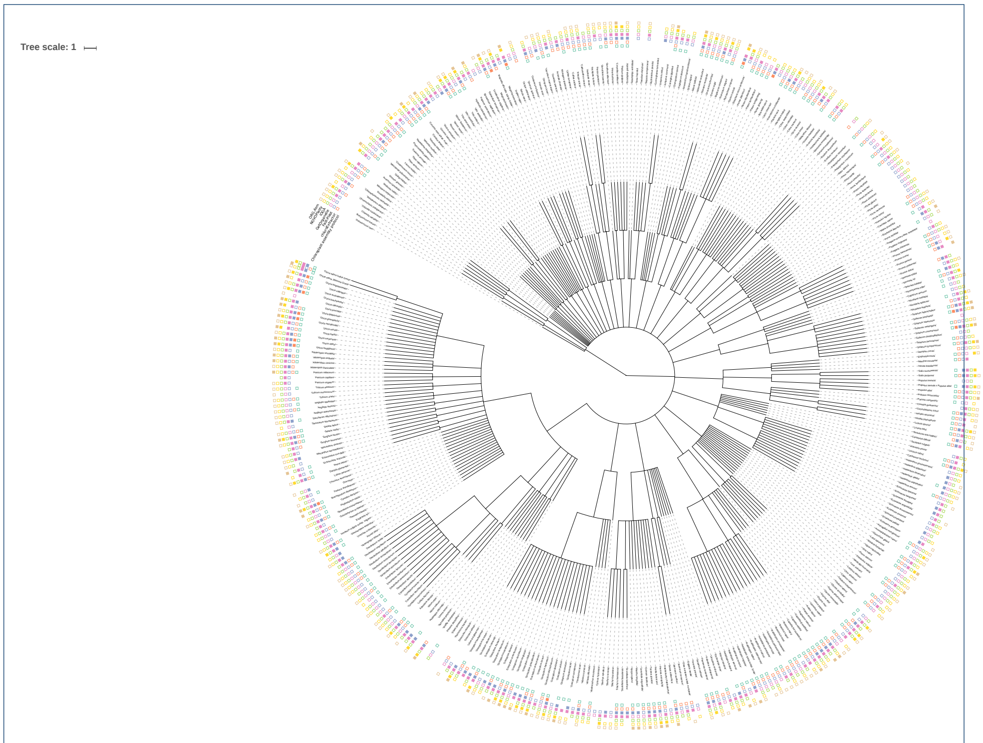

Figure 7 Success for chloroplast assembly shows no taxonomic bias Success of assemblers on real data sets on tree derived from NCBI taxonomy [82]. Plot was prepared using [83]

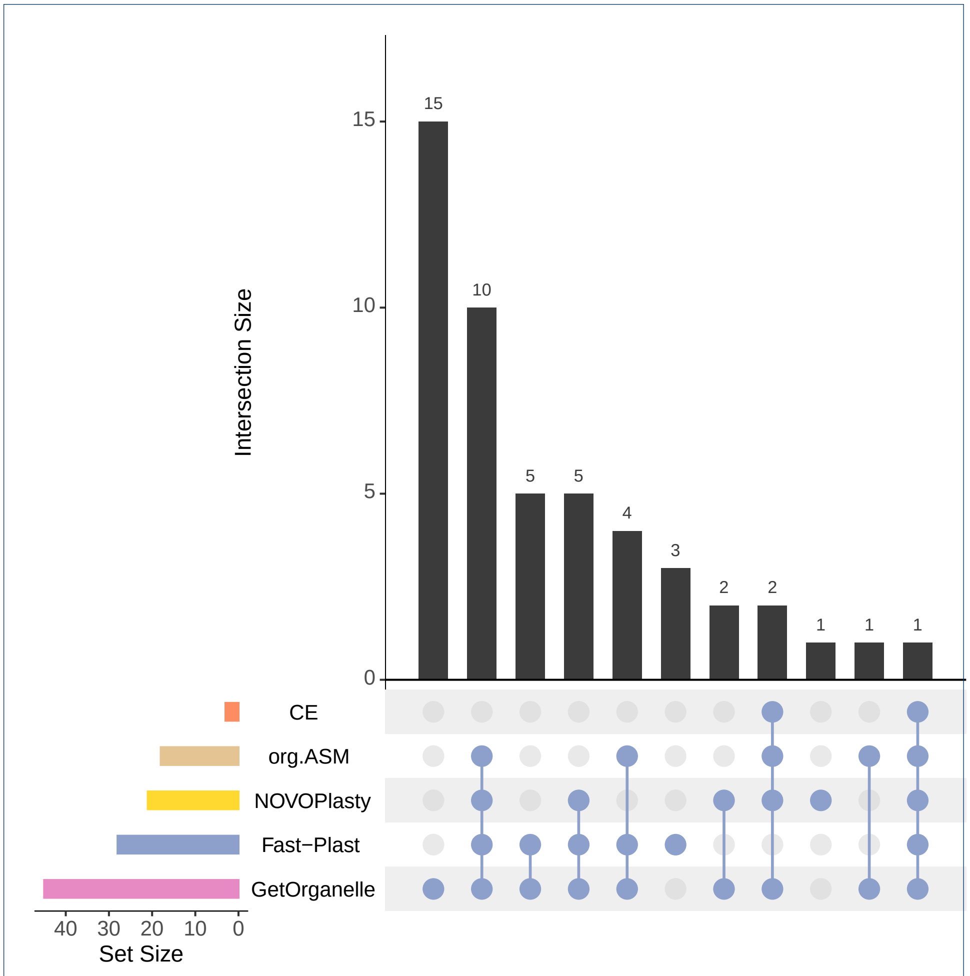

Figure 8 Upset plot [81] comparing success of assemblers on the novel data sets The plot shows the intersection of success (single contig, $length \geq 130kbp$, $ir \geq 17kbp$) between assemblers. For 15 data sets only `GetOrganelle` was able to obtain a complete chloroplast. 10 were successful with `GetOrganelle`, `Fast-Plast`, `NOVOPlasty`, and `ORG.Asm` and so on

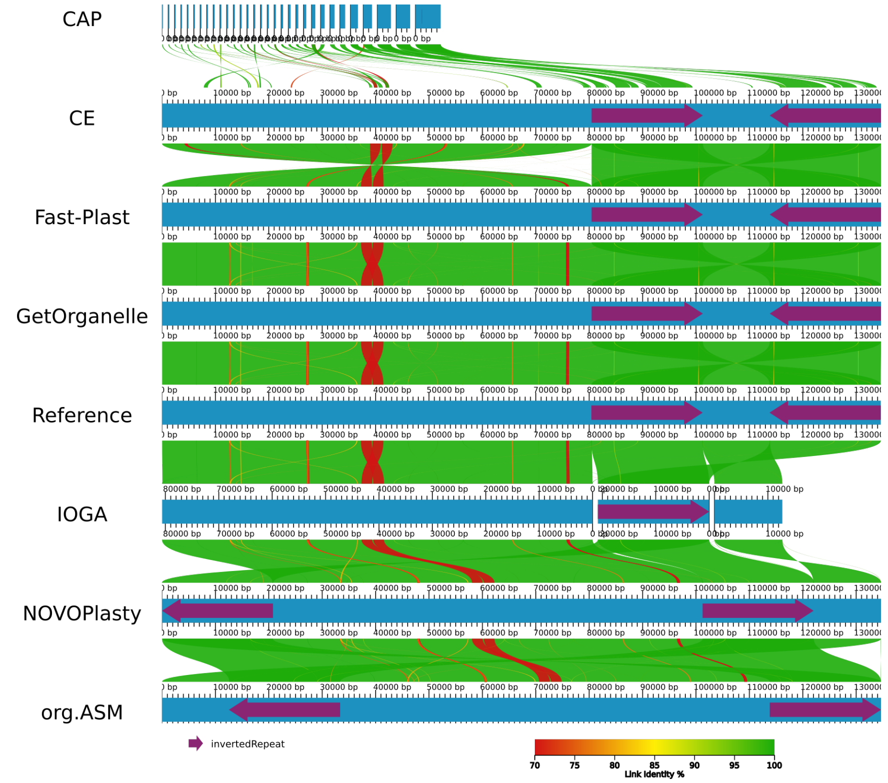

Figure 9 AliTV plot [47] showing alignments of the different assemblers compared to each other and the reference Each blue bar in the plot corresponds to an assembly of the *Oryza brachyantha* chloroplast (DRR053294). Regions in adjacent assemblies are connected with colored ribbons for similar regions in the alignment with identity coded as color from red to green. The purple arrows on the assemblies depict the IR regions. This plot highlights some common problems with chloroplast assembly. `Chloroplast assembly protocol` only assembled small fragments mostly from the IR region. `IOGA` returned three separate contigs corresponding to LSC, SSC, and IR. `Fast-Plast` and `GetOrganelle` produced assemblies that were identical to the reference. `chloroExtractor` has the same structure but the LSC is flipped compared to the reference (which is a biologically valid option). Both `NOVOPlasty` and `ORG.Asm` had the SSC flipped compared to the reference and did not start with the LSC but the IR and SSC, respectively (the start point is arbitrary in a circular sequence).

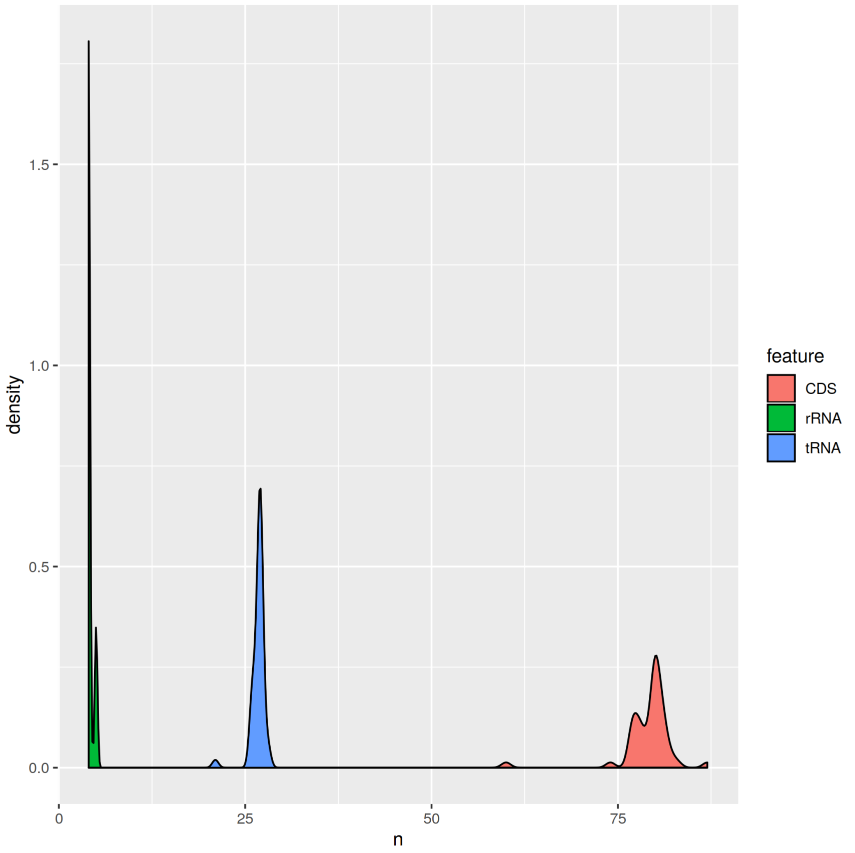

Figure 10 Number of distinct features in novel chloroplast genomes The distribution of feature types tRNA, rRNA, and coding sequence (CDS) are shown separately.