

**A draft genome for the Venus flytrap,
*Dionaea muscipula***

Evaluation of assembly strategies for a complex genome
– Development of novel approaches and bioinformatics solutions

Dissertation zur Erlangung des
naturwissenschaftlichen Doktorgrades
der Julius-Maximilians-Universität Würzburg

vorgelegt von

Thomas Hackl

geboren in Augsburg

Würzburg 2016

Eingereicht am:

Mitglieder der Promotionskommission:

Vorsitzender:

Gutachter: Prof. Dr. Jörg Schultz

Gutachter: Prof. Dr. Rainer Hedrich

Tag des Promotionskolloquiums:

Doktorurkunde ausgehändigt am:

Hackl, Thomas
Lechallee 43
86399 Bobingen
Tel.: +49 170 6895449
E-Mail: thomas.hackl@uni-wuerzburg.de

Erklärungen nach §4 Abs. 3 Satz 3, 5, 8 der Promotionsordnung der Fakultät für Biologie

Affidavit

I hereby declare that my thesis entitled: "A draft genome for the Venus flytrap, *Dionaea muscipula*" is the result of my own work. I did not receive any help or support from commercial consultants. All sources and / or materials applied are listed and specified in the thesis.

Furthermore I verify that the thesis has not been submitted as part of another examination process neither in identical nor in similar form.

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, die Dissertation: "A draft genome for the Venus flytrap, *Dionaea muscipula*", eigenständig, d. h. insbesondere selbständig und ohne Hilfe eines kommerziellen Promotionsberaters, angefertigt und keine anderen, als die von mir angegebenen Quellen und Hilfsmittel verwendet zu haben.

Ich erkläre außerdem, dass die Dissertation weder in gleicher noch in ähnlicher Form bereits in einem anderen Prüfungsverfahren vorgelegen hat.

Würzburg, den 11. Mai 2016

Thomas Hackl

Contents

Affidavit	iii
Summary	ix
Zusammenfassung	xi
Publications and Contributions	xiii
Acknowledgements	xv
I. A draft genome for the Venus flytrap, <i>Dionaea muscipula</i>	1
1. Introduction	5
1.1. Carnivory in plants	5
1.2. A <i>most wonderful</i> plant	6
1.3. Whole genome sequencing and assembly	9
2. Sequencing data inventory	11
2.1. BGI Illumina data	11
2.1.1. Raw BGI paired-end libraries	11
2.1.2. Raw BGI mate-pair libraries	12
2.1.3. Trimmed BGI paired-end libraries	13
2.1.4. Normalized BGI paired-end libraries	14
2.2. LGC Illumina data	16
2.2.1. Clipped LGC paired-end libraries	16
2.2.2. Trimmed LGC paired-end libraries	16
2.2.3. Normalized LGC paired-end libraries	17
2.3. PacBio data	18
2.3.1. Raw PacBio libraries	18
2.3.2. <i>In silico</i> mate-pair libraries from corrected PacBio reads	21
3. Software and parameters	23
3.1. Computing resources	23
3.2. Sequencing data quality control	23
3.3. Illumina data processing	24
3.3.1. Illumina adapter and quality trimming	24

Contents

3.3.2.	Insert size estimation	24
3.3.3.	Overlap merging	25
3.3.4.	Read Error correction	25
3.3.5.	Digital normalization	26
3.4.	PacBio data processing	27
3.4.1.	Preparation	27
3.4.2.	Hybrid correction	27
3.5.	k -mer analysis, genome size estimation and k -mer-derived plots	28
3.6.	Assembly and scaffolding software	29
3.6.1.	SOAP-denovo 2	29
3.6.2.	ALLPATHS-LG	30
3.6.3.	ABYSS	31
3.6.4.	MaSuRCA	31
3.6.5.	Platanus	32
3.6.6.	Meraculous 2	32
3.6.7.	Minia	32
3.6.8.	Discover de novo	33
3.6.9.	MIRA	33
3.6.10.	Celera	33
3.6.11.	DBG2OLC	34
3.6.12.	PBJelly	34
3.7.	Assembly analysis	34
3.7.1.	Assembly metrics with SeqFilter and QUAST	34
3.7.2.	Completeness of gene content with CEGMA and BUSCO	35
3.7.3.	Transcriptome coverage	35
4.	Assembling the genome of <i>D. muscipula</i>	37
4.1.	Evaluation of the BGI draft assembly reveals substantial deficiencies	37
4.1.1.	Basic metrics	37
4.1.2.	Conserved gene content	38
4.1.3.	Transcript coverage	39
4.1.4.	Contig coverage and GC landscape	40
4.2.	Assessment of the BGI read data shows technical bias and high level of heterogeneity	47
4.2.1.	Quality control of BGI libraries	47
4.2.2.	Insert-size distribution of BGI libraries	48
4.2.3.	k -mer analysis, genome size estimation and repeat content	53
4.2.4.	GC bias in BGI sequencing data	55
4.3.	Various SOAP and ALLPATHS assemblies shed light on specific challenges	58
4.3.1.	SOAP assemblies	58
4.3.2.	ALLPATHS assemblies from low coverage data sets	59
4.3.3.	Transcriptome coverage and comparison of SOAP and ALLPATHS assemblies	60

4.3.4.	ALLPATHS assembly with a high coverage data set	61
4.4.	Redundancy reduction through digital normalization boosts assembly computation and quality	63
4.5.	Resequencing solves heterogeneity issue and mitigates technical bias . . .	67
4.5.1.	Quality control of LGC libraries	68
4.5.2.	Insert-size distribution of LGC Illumina libraries	69
4.5.3.	k -mer-analysis, genome size estimation and repeat content	70
4.5.4.	Digital normalization of LGC sequencing data	74
4.5.5.	Merging of overlapping libraries	74
4.6.	LGC reads further improve assembly contiguity and completeness	75
4.6.1.	SOAP assemblies corroborate higher quality of LGC reads	75
4.6.2.	Alternative assemblers fail on the <i>D. muscipula</i> data	76
4.6.3.	Combination of ALLPATHS and digital normalization is the superior strategy	77
4.7.	PacBio sequencing, correction and assembly	80
4.7.1.	Low coverage PacBio sequencing	80
4.7.2.	PacBio hybrid correction with proovread	81
4.7.3.	Transcriptome coverage of the PacBio data	83
4.7.4.	PacBio-only and PacBio-Illumina hybrid assemblies	84
4.7.5.	PacBio hybrid assemblies with DBG2OLC and corrected PacBio reads	86
4.7.6.	Custom approaches towards a PacBio assembly	89
4.8.	The <i>D. muscipula</i> draft genome assembly	91
5.	Conclusion	95
II.	Development of bioinformatics tools and libraries	99
6.	proovread – large-scale hybrid PacBio correction through iterative consensus	101
6.1.	Introduction	101
6.2.	proovread 1.0 – legacy implementation and evaluation	103
6.2.1.	Hybrid scoring model and suitable mappers	103
6.2.2.	Localized score comparison and trusted alignments	104
6.2.3.	Optimized speed and sensitivity through an iterative mapping procedure	105
6.2.4.	High flexibility through scalability and parallelization	107
6.2.5.	Benchmark of proovread and comparison to other tools	107
6.3.	Upgrades and extensions to the proovread pipeline	110
6.3.1.	Storage-efficient best alignment filter through progressive minimum score adjustment	110
6.3.2.	bwa-proovread – a modified BWA-MEM implementation	112
6.3.3.	Redundancy reduction through circular consensus computation	112
6.3.4.	Advanced trimming of undetected SMRT-bell adapters	113

Contents

6.3.5.	Usability of pre-computed unitigs for correccion	115
6.3.6.	Optimized correction for transcriptomic and metagenomic data . .	116
6.3.7.	Haplotype awareness through variant calling and read-backed phasing	118
6.4.	perl5lib-Sam – a perl API to read alignment data	123
6.4.1.	Sam::Alignment	124
6.4.2.	Sam::Parser	125
6.4.3.	Sam::Seq	125
6.4.4.	Sam::Phase	133
6.5.	Conclusion	135
7.	Efficient processing of sequence data	137
7.1.	perl5lib-Fasta/-Fastq – a Perl API to sequence data	137
7.2.	SeqFilter – versatile manipulation of sequence files	138
7.3.	SeqChunker – efficient subset generation and reproducible sampling for sequence data	138
8.	Analysis and visualization of k-mer distributions and derived data	141
8.1.	Introduction	141
8.1.1.	The concept of k -mers in sequence analysis	141
8.1.2.	Interpretation of k -mer distributions	143
8.1.3.	Estimation of genome size, repeat content and heterozygosity . . .	145
8.2.	Anscombe transformation facilitates computational analysis and visualiza- tion of k -mer distributions	146
8.3.	kmer-plot kcov – visualization and automated annotation of k -mer distri- butions	149
8.4.	kmer-coverage – representative, frequency adjusted k -mer coverage	151
8.5.	kmer-plot gccov – automated visualization of GC-coverage plots	153
8.6.	Conclusion	155
Back matter		159
	Glossary	159
	List of Software Packages	160
	List of Tables	162
	List of Figures	165
	References	167
APPENDIX		181

Summary

The Venus flytrap, *Dionaea muscipula*, with its carnivorous life-style and its highly specialized snap-traps has fascinated biologist since the days of Charles Darwin. The goal of the *D. muscipula* genome project is to gain comprehensive insights into the genomic landscape of this remarkable plant.

The genome of the diploid Venus flytrap with an estimated size between 2.6 Gbp to 3.0 Gbp is comparatively large and comprises more than 70 % of repetitive regions. Sequencing and assembly of genomes of this scale are even with state-of-the-art technology and software challenging. Initial sequencing and assembly of the genome was performed by the BGI (Beijing Genomics Institute) in 2011 resulting in a 3.7 Gbp draft assembly. I started my work with thorough assessment of the delivered assembly and data. My analysis showed that the BGI assembly is highly fragmented and at the same time artificially inflated due to overassembly of repetitive sequences. Furthermore, it only comprises about on third of the expected genes in full-length, rendering it inadequate for downstream analysis.

In the following I sought to optimize the sequencing and assembly strategy to obtain an assembly of higher completeness and contiguity by improving data quality and assembly procedure and by developing tailored bioinformatics tools. Issues with technical biases and high levels of heterogeneity in the original data set were solved by sequencing additional short read libraries from high quality non-polymorphic DNA samples. To address contiguity and heterozygosity I examined numerous alternative assembly software packages and strategies and eventually identified ALLPATHS-LG as the most suited program for assembling the data at hand. Moreover, by utilizing digital normalization to reduce repetitive reads, I was able to substantially reduce computational demands while at the same time significantly increasing contiguity of the assembly.

To improve repeat resolution and scaffolding, I started to explore the novel PacBio long read sequencing technology. Raw PacBio reads exhibit high error rates of 15 % impeding their use for assembly. To overcome this issue, I developed the PacBio hybrid correction pipeline proovread (Hackl et al., 2014). proovread uses high coverage Illumina read data in an iterative mapping-based consensus procedure to identify and remove errors present in raw PacBio reads. In terms of sensitivity and accuracy, proovread outperforms existing software. In contrast to other correction programs, which are incapable of handling data sets of the size of *D. muscipula* project, proovread's flexible design allows for the efficient distribution of work load on high-performance computing clusters, thus enabling the correction of the Venus flytrap PacBio data set.

Summary

Next to the assembly process itself, also the assessment of the large de novo draft assemblies, particularly with respect to coverage by available sequencing data, is difficult. While typical evaluation procedures rely on computationally extensive mapping approaches, I developed and implemented a set of tools that utilize k -mer coverage and derived values to efficiently compute coverage landscapes of large-scale assemblies and in addition allow for automated visualization of the of the obtained information in comprehensive plots.

Using the developed tools to analyze preliminary assemblies and by combining my findings regarding optimizations of the assembly process, I was ultimately able to generate a high quality draft assembly for *D. muscipula*. I further refined the assembly by removal of redundant contigs resulting from separate assembly of heterozygous regions and additional scaffolding and gapclosing using corrected PacBio data. The final draft assembly comprises 86×10^3 scaffolds and has a total size of 1.45 Gbp. The difference to the estimated genomes size is well explained by collapsed repeats. At the same time, the assembly exhibits high fractions full-length gene models, corroborating the interpretation that the obtained draft assembly provides a complete and comprehensive reference for further exploration of the fascinating biology of the Venus flytrap.

Zusammenfassung

Die Venus Fliegenfalle, *D. muscipula* fasziniert aufgrund ihres karnivoren Lebensstil und ihrer hochspezialisierten Fallen Biologen schon seit der Zeit von Charles Darwins. Das Ziel des *D. muscipula* Genomprojekts ist es, neue Einblicke in den genomischen Grundlagen dieser besonderen Pflanze zu gewinnen.

Die diploide Venus Fliegenfalle verfügt mit eine geschätzten Größe von 2.6 Gbp bis 3 Gbp über ein vergleichsweise großes Genom, das zudem zu über 70 % aus repetitiven Regionen besteht. Sequenzierung und Assembly von Genomen dieser Größenordnung stellen selbst mit neusten technischen und informatischen Methoden eine große Herausforderung dar. Zum ersten mal sequenziert und assembliert wurde das Genom 2011 durch das BGI (Beijing Genomics Institute). Meine Arbeit am Genom der Fliegenfalle begann mit der Analyse des 3.7 Gbp großen Assemblies, welches wir vom BGI erhalten haben. Mit meinen Untersuchungen könnte ich zeigen, dass das Assembly stark fragmentiert und gleichzeitig durch überrepräsentierte repetitive Sequenzen stark aufgebläht ist. Darüberhinaus beinhaltet es gerade ein mal eine drittel der erwarteten Gene in Volllänge, wodurch es für die weiter Analyse ungeeignet ist.

In meiner weiteren Arbeit habe ich mich daher darauf konzentriert, unsere Sequenzierungs- und Assemblierungsstrategie zu verfeinern um ein stärker zusammenhängendes und vollständigeres Assembly zu erhalten. Dafür war es notwendig die Qualität der Sequenzierdaten so wie den Assemblierungsprozess selbst zu optimieren, und Programme zu entwickeln, die eine Verbesserung der Daten und eine Analyse der Zwischenergebnisse ermöglichen. So wurden etwa zur neue Bibliotheken von nicht-polymorphen DNA-Proben sequenziert um die Heterogenität im Datensatz zu verringern. Um die Kontinuität der Assemblies zu verbessern und Probleme mit der Heterozygotität der Daten zu lösen habe ich eine Reihe verschiedener Assemblierungsprogramme getestet. Dabei zeigte sich, dass das Programm ALLPATHS-LG am besten geeignet ist für die Assemblierung von *D. muscipula* Daten. Durch den Einsatz von digitaler Normalisierung konnte ich den Bedarf an Computerressourcen für einzelne Assemblierungen deutlich reduzieren und gleichzeitig die Kontinuität der Assemblies deutlich erhöhen.

Zur besseren Auflösung repetitiver Strukturen im Genom, habe ich auf eine neu entwickelte Sequenzierertechnologie von PacBio zurückgegriffen, die deutlich länger Sequenzen erzeugt. Um die neuen Daten trotz ihrer hohen Fehlerrate von 15 % für Assemblierungen nutzen zu können, entwickelte ich das Korrekturprogramm `proofread` (Hackl et al., 2014). `proofread` nutzt kurze Illumina Sequenzen mit hoher Sequenzierertiefe um innerhalb eines iterativen Prozess Fehler in PacBio Daten ausfindig zu machen und zu korrigieren.

Zusammenfassung

Das Programm erreicht dabei eine bessere Genauigkeit und eine höhere Sensitivität als vergleichbare Software. Darüber hinaus erlaubt sein flexibles Design auch Datensätze in der Größenordnung des Fliegenfallengenoms effizient auf großen Rechenclustern zu bearbeiten.

Neben dem Assemblierungsprozess an sich, stellt auch die Analyse von Assemblies großer Genome eine Herausforderung dar. Klassische Methoden basieren oft auf der rechenintensiven Berechnung von Alignments zwischen Sequenzierdaten und Assembly. Um vergleichbare Analysen deutlich schneller generieren zu können, habe ich Programme entwickelt die auf der Auswertung von k -mer Häufigkeiten beruhen, und die gewonnenen Ergebnisse in übersichtlichen Graphiken darstellen.

Durch Kombination der so gewonnenen Einblicke und der verschiedenen Erkenntnisse bezüglich der Optimierung des Assemblierungsprozesses, war es mir am Ende möglich, ein Assembly von hoher Qualität für das Genom der Venus Fliegenfalle zu rekonstruieren. Dieses habe ich weiter verfeinert, unter anderem durch das Entfernen heterozygoter Sequenzen und durch das Flickern von Lücken mit Hilfe von PacBio Daten. Das so erstellte Assembly besteht aus 86×10^3 Sequenzen und hat eine Gesamtgröße von 1.45 Gbp. Der Unterschied zur erwarteten Genomgröße lässt sich dabei gut durch kollabierte repetitive Regionen erklären. Gleichzeitig untermauert ein hoher Anteil an Vollhängengen im Assembly die Interpretation, dass das vorliegende Assembly eine vollständiges und umfassendes Abbild der *D. muscipula* Genom zeigt, und dass es sich damit als gute Grundlage für weitere Untersuchungen zur Biologie dieser faszinierenden Pflanze eignet.

Publications and Contributions

Publications

T. Hackl et al. (2014). “proofread: large-scale high-accuracy PacBio correction through iterative short read consensus”. *Bioinformatics* 30.21, pp. 3004–3011

B. Slaby, T. Hackl et al. “A PacBio-Illumina hybrid assembly strategy for metagenomes”. *in preparation*

Invited talks

“proofread: high accuracy PacBio hybrid correction for large genomes and transcriptomes”, *PacBio User seminar Wageningen UR*, 2014, Wageningen, Netherlands

Poster presentations

“proofread: 3rd generation sequencing length with 2nd generation accuracy”, *ISM-B/ECCB*, 2013, Berlin, Germany

“Large-scale PacBio hybrid correction through iterative short read consensus”, *NGS Data Congress*, 2013, London, United Kingdom

Acknowledgements

While this dissertation carries my name, I could not have done it without the support of many great people around me. To these people, I want to express my deep gratitude.

First, I would like to thank Prof. Dr. Rainer Hedrich for giving me the opportunity to work on this fascinating project.

Further, I want to thank Prof. Dr. Jörg Schultz and Dr. Frank Förster for their guidance and advice on the project, and their inspirational enthusiasm that led me to become a computational biologist in the first place.

I want to thank all my peers, in particular, Felix, Niklas, Markus, Simon and Beate for a great working atmosphere, great collaborations and fruitful discussion.

I would like to thank all the people of the Department of Botany and everybody else that in one way or another was involved in the Venus flytrap project.

My special thanks are extended to Dr. Matthias Fischer and his family, as well as the entire department of Biomolecular Mechanisms for a great last year in Heidelberg.

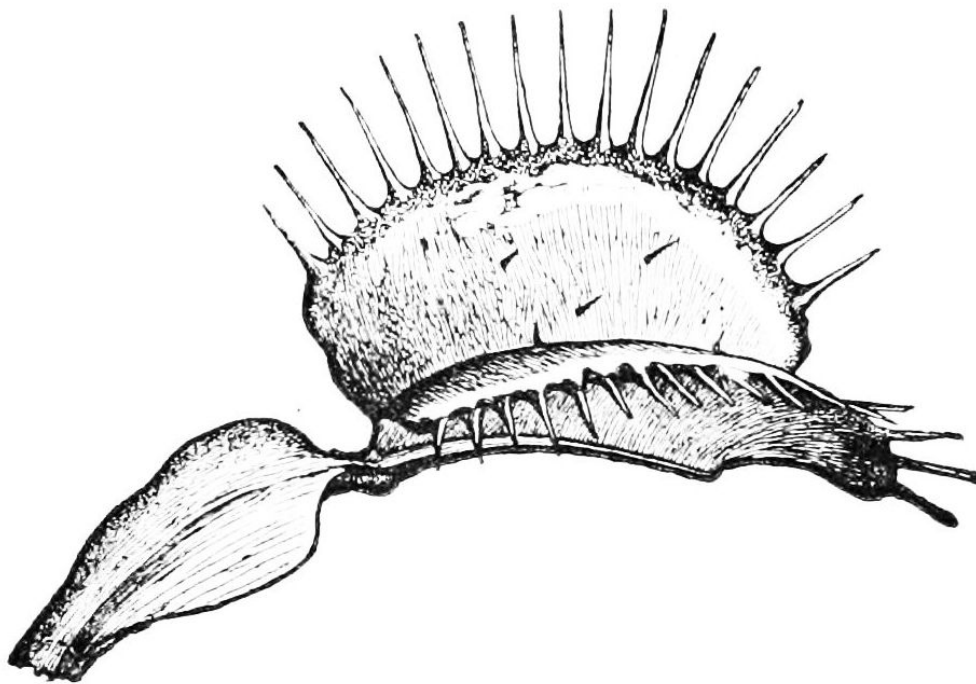
I am grateful to all my family and friends, in particular, Christiane, Markus, Thomas, Stephan, Sebb, Johannes, Kristin, Almuth, Patrick, and many, more, for all their support and at times necessary distractions.

I want to thank my parents for always supporting and believing in me, and in particular, for always encouraging my undoubtedly often annoying curiosity.

Finally, and most importantly I want to thank my wife Kristina. She constantly inspires me to become a better person as well as a better scientist and her unconditional love and support are most essential to my work and my life.

Part I.

**A draft genome for the Venus flytrap,
*Dionaea muscipula***



»THIS plant, commonly called Venus' fly-trap, from the rapidity and force of its movements, is one of the most wonderful in the world.«

— Charles Darwin

Illustration and quote: C. Darwin (1875). "Insectivorous Plants Table of Contents", pp. 286-287

1. Introduction

1.1. Carnivory in plants

Carnivorous plants with their seemingly paradox life-style – plants feeding on animals – have fascinated *naturalists* since the days of Darwin and Hooker. For his work “Insectivorous Plants Table of Contents” (1875), Darwin conducted detailed studies and experiments on morphology, physiology and ecology of a series of carnivorous plants to shed light onto their biology. He and later his successors showed that carnivorous plants can feed on insects and exhibit enhanced growth and increased rates of reproduction, although, feeding was found not to be essential for normal growth (Goebel, 1893; Oosterhuis, 1927; Lloyd, 1942). Cost-benefit analysis suggest that the investment in the formation of traps is paid off by the uptake of nutrients rare to the respective habitats, such as nitrate, potassium and phosphate (T. Givnish et al., 1984; A. Ellison, 2006). Carnivory in plants is thus considered an adaptation strategy to complement the unfavorable conditions of nutrient-poor environments by extending the range of resources that the plants are able to tap into (Adamec, 1997).

In general, carnivorous plants share three common characteristics: the capability to dissolve animal prey, usually by the secretion of digestive fluids, the ability to absorb the metabolites from the digested matter for their own benefit, and morphological adaptation to passively or actively attract and trap their prey (Darwin, 1875; Lloyd, 1942; T. Givnish et al., 1984). As with most definitions, exceptions exist, such as the genera *Pinguicula* and *Philcoxia*, which lack obvious attractants, or *Brocchinia*, *Darlingtonia* and some species of *Sarracenia* that depend on commensal microbes or insect larvae to break down their prey (T. J. Givnish, 2015). Today, based on the aforementioned criteria, almost 600 plant species with a carnivorous life-style have been identified.

Despite the seeming uniqueness of the trait, already Darwin deduced from the existence of carnivorous characteristics in distantly related phyla without a shared common carnivorous ancestor that carnivory has evolved multiple times independently within the plant kingdom. His findings have since been confirmed by molecular phylogenetic studies suggesting at least nine independent origins of carnivory in five different orders of flowering plants (Albert et al., 1992; A. Ellison and N. Gotelli, 2001; A. M. Ellison and N. J. Gotelli, 2009; T. J. Givnish, 2015; Pavlovi and Saganová, 2015).

The diversity of carnivorous plants is also reflected in the broad variety of trapping mechanisms that have evolved in the different lineages. There are five main types of traps:


1. Introduction

(1) pitcher plants possess pitfall traps with chambers formed by leaves and filled with digestive fluid; (2) flypaper traps secrete a sticky mucilage from specialized glands; (3) snap traps actively catch prey through rapid movement of modified leaves; (4) bladder traps suck in prey by negative pressure, and (5) in lobster pot traps, the prey is steered into the digestive organ by modified inward-facing hairs. These main types, however, are not confined to specific lineages, but have evolved independently several times across different taxa, too. Pitfall traps, for example, can be found in five different families (Sarraceniaceae, Nepenthaceae, Bromeliaceae, Cephalotaceae and Eriocaulaceae) with six proposed independent origins, and flypaper traps have evolved at least five times in Droseraceae, Lentibulariaceae (*Pinguicula*), Drosophyllaceae (*Drosophyllum*) and Dioncophyllaceae (*Triphyophyllum*) (T. J. Givnish, 2015).

In the cases of snap, bladder, and many flypaper traps, active recognition of prey and the ability of the plant to respond to excitation by active movement are of high importance to the trapping process. These specialized adaptations further underpin the unique evolutionary position of carnivorous plants amongst angiosperms. The polyphyletic origin of carnivory and trapping mechanisms, with a high degree of novelty and convergent evolution render carnivorous plants valuable model systems for addressing a large variety of physiological, ecological and evolutionary questions (A. Ellison and N. Gotelli, 2001).

1.2. A most wonderful plant



Figure 1.1.: Open *D. muscipula* snap trap with trigger hairs and glands on the inner surface of the cilia framed lobes. Photo by Noah Elhardt 

Even among the carnivorous plants, the Venus flytrap *D. muscipula* with its highly specialized and beautiful snap trap, stands out (Darwin, 1875). The plant is native to the nutrient-poor swamps found on the U.S. East Coast in North and South Carolina (Govaerts, 2016). Phylogenetically, the flytrap is grouped into the family of sundews (Droseraceae) within the order of Caryophyllales, in the monotonous genera of *Dionaea* (Rivadavia et al., 2003). *D. muscipula* and its aquatic sister species, the waterwheel plant *Aldrovanda vesiculosa*, are the only two members among the carnivorous plants that possess an active snap trap. Both species share a common ancestor and their traps are thought to have evolved from flypaper traps found in many other members of the Droseraceae family (Cameron et al., 2002; Gibson and Waller, 2009).

The trap of *D. muscipula* is formed by a modified leaf comprising a pair of terminal lobes hinged at the midrib (fig. 1.1) (Darwin, 1875). The edges of the lobes are decorated with stiff, slightly inward-bent cilia, that upon closing aid in preventing the prey from escaping. Below the cilia extrafloral nectaries are located, producing nectar and UV-reflective

1.2. A most wonderful plant

substances (Gibson and Waller, 2009). In addition, the flytrap is secreting more than 60 volatile organic compounds mimicking the scent of fruits and flowers (Kreuzwieser et al., 2014). In open state, the lobes of the trap are bent konvexly, exposing the enticing red inner surface colored by anthocyanin pigments. The combination of chemical and visual stimuli allows the plant to efficiently lure insects into its deadly trap.

The inner surface of the trap is covered with minute glands and typically possesses six mechanosensitive trigger hairs, three on each lobe, triangularly placed (Darwin, 1875; Lloyd, 1942; Scala et al., 1968; Robins and Juniper, 1980; Bailey and McPherson, 2012). If two of the hairs or one hair within 20 s to 25 s are touched, the trap snaps shut (Hodick and Sievers, 1988; Escalante-Pérez et al., 2011; Volkov et al., 2011). The transduction of the mechanical stimuli is converted into receptor potentials by mechanosensitive ion channels. The signal is propagated throughout the trap by action potentials caused by fluxes of chloride and calcium ions (Burdon-Sanderson, 1873; Hodick and Sievers, 1988; Krol et al., 2006; Escalante-Pérez et al., 2011). The subsequent motion is driven by the rapid release of elastic energy stored in the tension of the konvex bent lobes of the meta-stable open trap.

During closing, the lobes switch into a konkav shape enclosing the prey within. With a duration of 100 ms the closing of the trap is one of the fastest movements in the plant kingdom (Division et al., 2005). Including transduction of stimuli and propagation of the signal, the entire trapping process takes between 350 ms and 500 ms (Escalante-Pérez and Scherzer, 2014). The underlying mechanisms initiating the release of the elastic energy is still subject to debate: An early model proposed by Williams and Bennett (1982) indicates a fast, pH-induced, irreversible growth of the outer part of the lobes, while more recent studies favor rapid, reversible changes in turgor in particular cell layers within the lobes (Volkov et al., 2008; Escalante-Pérez and Scherzer, 2014).

After the rapid closing phase the trap is shut almost completely with large prey prevented from escaping by the cilia while very small prey is still allowed to leave (fig. 1.2). This approach ensures that only prey of sufficient size from a cost-benefit point of view is actually digested by the plant. The movement of the restricted prey further stimulates the trigger hairs leading to an increase in cytosolic calcium (Escalante-Pérez et al., 2011), which together with the detection of nitrogenous compounds derived from the prey, results in a complete closing of the trap by growth and the formation of the digestive *stomach* (Darwin, 1875). Dissolving of the prey is achieved by the secretion of a digestive fluid – an acidic cocktail comprising a variety of hydrolytic enzymes (Scala et al., 1968; Robins and Juniper, 1980; Rea, 1982; Takahashi et al., 2009; Escalante-Pérez et al., 2011; Schulze et al., 2012) – from the gland complexes located on the trap’s inner surface (Lloyd, 1942; Lüttge, 1963; Lüttge, 1964). The uptake of the solubilized nutrients such



Figure 1.2.: *D. muscipula* snap trap after rapid closing phase with prey captured and cilia preventing escape through the remaining gap. Photo by [xenocerebral](#)

1. *Introduction*

as NH_4^+ , K^+ and PO_4^{3-} occurs through specific channels expressed in the gland cells (Scherzer et al., 2013; Gao et al., 2015; Böhm et al., 2015; Böhm et al., 2016) and is assisted by the absorption of small organic compounds through endocytosis (Adlassnig and KollerPeroutka, 2012).

1.3. Whole genome sequencing and assembly

DNA (Deoxyribonucleic acid) molecules, first discovered by Miescher F (1871), today are known to be the primary carriers of heredity information in living organisms and many viruses (Avery, 1946; Chase, 1952). The discovery of the three-dimensional structure of the double-stranded DNA helix (Watson and Crick, 1953) and the associated concept of semiconservative replication (Meselson and Stahl, 1958), followed by the decryption of the genetic code (Crick et al., 1961; M. W. Nirenberg and Matthaei, 1961; Wahba et al., 1963; M. Nirenberg et al., 1965), set the foundation for the understanding of the information encoded in biological DNA sequences.

In the 1970's, Sanger and Coulson (1975, 1977) and independently Maxam and Gilbert (1977) were the first to establish universally applicable experimental techniques for the determination of the exact order of nucleotides – the four bases (bp) adenine (A), guanine (G), cytosine (C), and thymine (T) – present in biological DNA molecules. Soon after, with the sequencing of the complete 5 kbp genome of the ϕ X174 phage, Sanger, Air, et al. (1977) heralded the era of genome sequencing and analysis, a new scientific field today referred to as *genomics*. In the following, technical advances, particularly with respect to automation of the procedure, lead to the sequencing and analyses of larger genomes. In 1995, Fleischmann and Adams published the first fully sequenced bacterial genome, the genome of *Haemophilus influenzae* with a size of 1.8 Mbp. At the beginning of the century, a huge collaborative effort from researchers around the globe over a period of more than a decade culminated in the release of "[t]he sequence of the human genome" (Lander et al., 2001; Venter et al., 2001) with a total size of 3 Gbp.

Already in 1979, Staden noted that "[w]ith modern fast sequencing techniques^{1,2} and suitable *computer programs* it is now possible to sequence whole genomes". This statement shows that right from the beginning, the experimental procedure of sequencing was tightly linked to computational processing of the obtained sequencing data. The primary reason for this relation lies with the fact that while genomic DNA sequences can be up to several hundred megabases in length, sequencing technologies are only able to read much shorter fragments, often of less than a couple of hundred bases and even with recently emerging long read technologies rarely more than some 10 kbp (Pettersson et al., 2009; Quail et al., 2012). In order to obtain a complete genome sequence, the sequencing-derived *reads* need to be tied together in a coherent manner, a process later labeled *assembly*. While this is manually achievable for short genomes comprised of a couple of dozen reads, it is infeasible for data sets comprising more than a few hundred reads.

The basic steps toward a systematic and thus computationally automatable reconstruction of a continuous sequence from reads are the (1) identification of overlaps among different reads and (2) the layout of the fragments in a way that they can be concatenated into a consistent, longer sequence. The main challenges of the approach are (1) sequencing

¹Sanger and Coulson, 1975.

²Maxam and Gilbert, 1977.

1. Introduction

errors in read data, which obscure overlaps and (2) the existence of repeated subsequences within longer sequences which complicate the layout (E. W. Myers, 1995).

Initially, most assembly algorithms were based on the OLC (Overlap-Layout-Consensus) paradigm. Here, read overlaps are determined by exhaustive all-vs-all alignments. The computation and assessment of pair-wise sequence alignments is a common problem in bioinformatics and a series of algorithms and programs were developed (Needleman and Wunsch, 1970; Smith and Waterman, 1981; Altschul et al., 1990; Chaisson and Tesler, 2012; G. Myers, 2014), and subsequently adapted for the use with assembly software. The overlap information obtained from the alignments is usually represented as a *string graph*, from which continuous, unambiguous paths, or *contigs* are reconstructed (Nagarajan and Pop, 2013). Typical OLC assemblers are Phrap (Green, 2016), MIRA (Chevreux et al., 1999), ARACHNE (Batzoglou et al., 2002) and Celera (E. Myers et al., 2000; Berlin et al., 2015).

With the rise of *next generation* sequencing technologies (Pettersson et al., 2009), and in particular, Illumina’s massively parallel *sequencing by synthesis* systems, which generate highly accurate but very short reads, a novel assembly approach based on De Bruijn graphs (Bruijn, 1946s) was introduced by Pevzner et al. (2001). In a De Bruijn graph, a reads set is represented as a network of k -mers (all possible substrings of a read with the length k), with connections made between k -mers occurring directly adjacent to each other within a read. The resulting paths implicitly model read overlaps, yet without the need of actually having to compute read-vs-read alignments (Nagarajan and Pop, 2013). While k -mer graph approaches are more susceptible to sequencing errors, they offer a computationally much more efficient way to represent and reconstruct sequencing data than OLC approaches, in particular if the number of overlaps is high (Pevzner et al., 2001; Zerbino and Birney, 2008; Pettersson et al., 2009). Commonly used De Bruijn graph assemblers include Velvet (Zerbino and Birney, 2008), ABySS (Simpson et al., 2009), SOAP (SOAP-denovo2, R. Li et al., 2009), ALLPATHS (ALLPATHS-LG, Gnerre et al., 2011) and SPAdes (Bankevich et al., 2012).

More recently, the emergence of third generation long read sequencing technologies (PacBio, Oxford Nanopore Technologies), producing longer reads at the expense of higher error rates, led to a renaissance of OLC-based assemblers, with novel implementations such as FALCON (J. Chin, 2016), DAZZLER (G. Myers, 2014), DBG2OLC (Ye et al., 2014) and miniasm (H. Li, 2015). While these new technologies and assembly programs are considered *the next big step* toward perfect assemblies, current instruments, algorithms and implementations are still under development and their actual impact has yet to be substantiated.

Nevertheless, due to the developments in the field of sequencing and assembly as well as downstream analysis and annotation over the last decades, genomic sequences today constitute comprehensive blueprints of life, providing fundamental insights in the genetic mark-up, the metabolic and functional capabilities as well as the evolutionary origins of organisms.

2. Sequencing data inventory

2.1. BGI Illumina data

DNA extraction, library design and construction as well as sequencing of the initial *D. muscipula* short read libraries were carried out by the BGI. A total of 35 libraries were sequenced on Illumina HiSeq (HiSeq instrument) instruments. 17 libraries were designed in paired-end configuration, comprising three overlapping libraries with reads of 100 bp in length and a mean targeted insert size of 170 bp, eleven libraries of 100 bp reads with insert sizes of 200 bp, 350 bp, 500 bp and 800 bp, three libraries of 150 bp reads, with inserts of 200 bp, 250 bp and 500 bp. 18 libraries were designed and processed in mate-pair configuration, with all libraries having 49 bp reads and 4 libraries each of insert size 2 kbp and 5 kbp and 5 libraries each of 10 kbp and 20 kbp insert size. In total 383 Gbp, (128 *x* assuming 3 Gbp genome) of paired-end and 176 Gbp (59 *x* assuming 3 Gbp genome) of mate-pair data were sequenced.

2.1.1. Raw BGI paired-end libraries

Table 2.1.: Inventory of raw BGI paired-end sequencing libraries showing number of reads, total size of the library, read length (rl), insert size (is) and relative GC and N content. Libraries are grouped into three subsets according to read length and insert size, with sums of reads, library size and estimated coverage given in highlighted rows.

IID	reads	size [bp]	rl [bp]	is [bp]	GC [%]	N [%]
Dm_GenIl_019	89811803	8981180300	100	170	42.44	0.01
	89811803	8981180300	100	170	42.32	0.46
Dm_GenIl_020	85363731	8536373100	100	170	42.19	0.02
	85363731	8536373100	100	170	42.00	0.64
Dm_GenIl_021	116389434	11638943400	100	170	42.09	0.01
	116389434	11638943400	100	170	42.13	0.04
total (is=170)	583129936	58312993600	19.4	X		
Dm_GenIl_022	84767343	8476734300	100	200	41.44	0.08
	84767343	8476734300	100	200	41.32	0.30
Dm_GenIl_023	60975254	6097525400	100	350	42.55	0.17
	60975254	6097525400	100	350	42.50	0.50
Dm_GenIl_024	97487970	9748797000	100	350	41.00	0.07
	97487970	9748797000	100	350	40.89	0.32
Dm_GenIl_025	106036232	10603623200	100	350	42.40	0.01
	106036232	10603623200	100	350	42.48	0.03

Continued on next page

2. Sequencing data inventory

Table 2.1.: Inventory of raw BGI paired-end sequencing libraries showing number of reads, total size of the library, read length (rl), insert size (is) and relative GC and N content. Libraries are grouped into three subsets according to read length and insert size, with sums of reads, library size and estimated coverage given in highlighted rows.

IID	reads	size [bp]	rl [bp]	is [bp]	GC [%]	N [%]
Dm_GenII_026	81957925	8195792500	100	350	42.48	0.13
	81957925	8195792500	100	350	42.67	0.01
Dm_GenII_027	57973259	5797325900	100	500	42.17	0.13
	57973259	5797325900	100	500	42.23	0.40
Dm_GenII_028	93532466	9353246600	100	500	41.02	0.10
	93532466	9353246600	100	500	40.98	0.24
Dm_GenII_029	60677178	6067717800	100	500	41.43	0.11
	60677178	6067717800	100	500	41.49	0.23
Dm_GenII_030	84125305	8412530500	100	800	41.44	0.05
	84125305	8412530500	100	800	41.44	0.32
Dm_GenII_031	82914350	8291435000	100	800	41.69	0.26
	82914350	8291435000	100	800	41.88	0.16
Dm_GenII_032	90320320	9032032000	100	800	41.22	0.05
	90320320	9032032000	100	800	41.22	0.28
total (is>170)	1801535204	180153520400	60.1 X			
Dm_GenII_033	136116175	20417426250	150	200	41.70	0.07
	136116175	20417426250	150	200	41.58	0.25
Dm_GenII_034	185983869	27897580350	150	250	41.44	0.00
	185983869	27897580350	150	250	41.46	0.08
Dm_GenII_035	161057909	24158686350	150	500	41.36	0.00
	161057909	24158686350	150	500	41.42	0.08
total (rl=150)	966315906	144947385900	48.3 X			
total	3350981046	383413899900	127.8 X			

2.1.2. Raw BGI mate-pair libraries

Table 2.2.: Inventory of raw BGI mate-pair sequencing libraries showing number of reads, total size of the library, read length (rl), insert size (is) and relative GC and N content. Libraries are grouped into three subsets according to read length and insert size, with sums of reads, library size and estimated coverage given in highlighted rows.

IID	reads	size [bp]	rl [bp]	is [bp]	N50 [bp]	GC [%]
Dm_GenII_001	97111962	4758486138	49	2000	43.37	0.01
	97111962	4758486138	49	2000	43.43	0.44
Dm_GenII_002	83011755	4067575995	49	2000	41.89	0.01
	83011755	4067575995	49	2000	42.15	0.16
Dm_GenII_003	94143871	4613049679	49	2000	43.45	0.01
	94143871	4613049679	49	2000	43.49	0.43
Dm_GenII_004	106125453	5200147197	49	2000	41.61	0.01
	106125453	5200147197	49	2000	41.91	0.07
total (is=2000)	760786082	37278518018	12.4 X			
Dm_GenII_005	109995613	5389785037	49	5000	41.25	0.01
	109995613	5389785037	49	5000	41.46	0.26

Continued on next page

2.1. BGI Illumina data

Table 2.2.: Inventory of raw BGI mate-pair sequencing libraries showing number of reads, total size of the library, read length (rl), insert size (is) and relative GC and N content. Libraries are grouped into three subsets according to read length and insert size, with sums of reads, library size and estimated coverage given in highlighted rows.

IID	reads	size [bp]	rl [bp]	is [bp]	N50 [bp]	GC [%]
Dm_GenII_006	89316107	4376489243	49	5000	42.70	0.01
	89316107	4376489243	49	5000	42.88	0.16
Dm_GenII_007	90045200	4412214800	49	5000	41.47	0.01
	90045200	4412214800	49	5000	41.71	0.16
Dm_GenII_008	102128251	5004284299	49	5000	42.55	0.01
	102128251	5004284299	49	5000	42.63	0.47
total (is=5000)	782970342	38365546758	12.8 X			
Dm_GenII_009	82318433	4033603217	49	10000	42.20	0.03
	82318433	4033603217	49	10000	42.39	0.04
Dm_GenII_010	105420967	5165627383	49	10000	42.60	0.01
	105420967	5165627383	49	10000	42.82	0.13
Dm_GenII_011	104402212	5115708388	49	10000	42.61	0.01
	104402212	5115708388	49	10000	42.83	0.14
Dm_GenII_012	88151111	4319404439	49	10000	42.77	0.00
	88151111	4319404439	49	10000	43.03	0.09
Dm_GenII_013	87288979	4277159971	49	10000	42.19	0.01
	87288979	4277159971	49	10000	42.44	0.02
total (is=10000)	935163404	45823006796	15.3 X			
Dm_GenII_014	88805534	4351471166	49	20000	41.60	0.00
	88805534	4351471166	49	20000	41.79	0.04
Dm_GenII_015	104439599	5117540351	49	20000	42.17	0.01
	104439599	5117540351	49	20000	42.79	0.02
Dm_GenII_016	169157218	8288703682	49	20000	41.83	0.01
	169157218	8288703682	49	20000	42.40	0.16
Dm_GenII_017	105071761	5148516289	49	20000	42.16	0.01
	105071761	5148516289	49	20000	42.75	0.10
Dm_GenII_018	88296535	4326530215	49	20000	41.48	0.03
	88296535	4326530215	49	20000	41.68	0.03
total (is=20000)	1111541294	54465523406	18.2 X			
total	3590461122	175932594978	58.6 X			

2.1.3. Trimmed BGI paired-end libraries

Table 2.3.: Inventory of trimmed BGI paired-end sequencing libraries showing number of reads, total size of the library, read length (rl), insert size (is) and relative GC and N content.

IID	reads	size [bp]	rl [bp]	is [bp]	GC [%]	N [%]
Dm_GenII_019	76537195	7584573949	75-100	170	41.45	0.00
	76537195	7560928597	75-100	170	41.45	0.00
Dm_GenII_020	73141684	7250065851	75-100	170	41.20	0.00
	73141684	7226430829	75-100	170	41.20	0.00
Dm_GenII_021	93450264	9146614827	75-100	170	40.62	0.00
	93450264	9184917381	75-100	170	40.72	0.00

Continued on next page

2. Sequencing data inventory

Table 2.3.: Inventory of trimmed BGI paired-end sequencing libraries showing number of reads, total size of the library, read length (rl), insert size (is) and relative GC and N content.

IID	reads	size [bp]	rl [bp]	is [bp]	GC [%]	N [%]
Dm_GenIl_022	71075309	7002660562	75-100	200	40.30	0.00
	71075309	6999519580	75-100	200	40.27	0.00
Dm_GenIl_023	50693908	4982498772	75-100	350	41.27	0.00
	50693908	4966029357	75-100	350	41.33	0.00
Dm_GenIl_024	80078902	7909148528	75-100	350	39.88	0.00
	80078902	7877032104	75-100	350	39.76	0.00
Dm_GenIl_025	83068511	8128602927	75-100	350	40.88	0.00
	83068511	8134553475	75-100	350	41.02	0.00
Dm_GenIl_026	67178569	6617736594	75-100	350	41.23	0.00
	67178569	6583167243	75-100	350	41.26	0.00
Dm_GenIl_027	47139008	4639144370	75-100	500	40.87	0.00
	47139008	4613081880	75-100	500	40.90	0.05
Dm_GenIl_028	75995500	7509976442	75-100	500	39.89	0.00
	75995500	7465373009	75-100	500	39.78	0.00
Dm_GenIl_029	50131206	4932765695	75-100	500	40.13	0.00
	50131206	4897978659	75-100	500	40.10	0.00
Dm_GenIl_030	65931985	6522998784	75-100	800	40.07	0.00
	65931985	6411327612	75-100	800	39.95	0.00
Dm_GenIl_031	62259973	6095929703	75-100	800	39.97	0.00
	62259973	6039389384	75-100	800	39.89	0.00
Dm_GenIl_032	69625681	6885349502	75-100	800	39.76	0.00
	69625681	6758247330	75-100	800	39.62	0.00
Dm_GenIl_033	109638366	15746813428	100-150	200	40.77	0.00
	109638366	15891839422	100-150	200	40.75	0.00
Dm_GenIl_034	163273896	24243191798	100-150	250	41.11	0.00
	163273896	23982381383	100-150	250	41.03	0.00
Dm_GenIl_035	129136047	19138795517	100-150	500	40.79	0.00
	129136047	18791643727	100-150	500	40.64	0.00
total	2736712008	307720708221				

2.1.4. Normalized BGI paired-end libraries

2.1. BGI Illumina data

Table 2.4.: Inventory of digitally normalized BGI paired-end sequencing libraries showing number of reads, total size of the library, read length (rl), insert size (is), relative amount of retained reads and relative GC and N content.

IID	reads	size [bp]	kept [%]	rl [bp]	is [bp]
Dm_GenIl_019	88441386	8844188883	49.2	78-100	170
Dm_GenIl_020	84765504	8476597655	49.6	78-100	170
Dm_GenIl_021	110193906	11019449432	47.3	78-100	170
Dm_GenIl_027	62416166	6241646924	53.8	78-100	500
Dm_GenIl_033	102572142	15385976632	37.7	117-150	200
Dm_GenIl_034	137122500	20568588443	36.9	118-150	250
total	585511604	70536447969	43.6	-	

2. Sequencing data inventory

2.2. LGC Illumina data

Four additional overlapping paired-end libraries were constructed and sequenced by the LGC (LGC Genomics) with a read length of 100 bp and a targeted insert size of 180 bp.

2.2.1. Clipped LGC paired-end libraries

Table 2.5.: Inventory of clipped LGC paired-end sequencing libraries showing number of reads, total size of the library, read length (rl), insert size (is) and relative GC and N content.

IID	reads	size [bp]	rl [bp]	is [bp]	GC [%]	N [%]
dm-il-01	342983444	34210566518	20-100	180	41.28	0.00
	342983444	34208198231	20-100	180	41.50	0.04
dm-il-02	346461493	34564374885	20-100	180	41.27	0.00
	346461493	34562088172	20-100	180	41.48	0.04
dm-il-03	344489284	34374751189	20-100	180	41.32	0.00
	344489284	34372624536	20-100	180	41.52	0.04
dm-il-04	441747363	44052248516	20-100	180	41.37	0.00
	441747363	44048861064	20-100	180	41.55	0.04
total	2951363168	294393713111				

2.2.2. Trimmed LGC paired-end libraries

Table 2.6.: Inventory of trimmed BGI paired-end sequencing libraries showing number of reads, total size of the library, read length (rl), insert size (is) and relative GC and N content.

IID	reads	size [bp]	rl [bp]	is [bp]	GC [%]	N [%]
dm-il-01_1	313,793,768	31,210,462,856	75-100	180	40.97	0.00
	313,793,768	31,120,247,523	75-100	180	41.02	0.00
dm-il-02_1	316,052,118	31,437,234,885	75-100	180	40.95	0.00
	316,052,118	31,336,387,709	75-100	180	40.99	0.00
dm-il-03_1	315,219,698	31,350,938,056	75-100	180	40.99	0.00
	315,219,698	31,264,539,637	75-100	180	41.04	0.00
dm-il-04_1	400,282,020	39,799,873,463	75-100	180	41.03	0.00
	400,282,020	39,687,222,247	75-100	180	41.03	0.00
total	2,690,695,208	267,206,906,376				

2.2.3. Normalized LGC paired-end libraries

Table 2.7.: Inventory of normalized LGC paired-end sequencing libraries showing number of reads, total size of the library and relative amount of retained reads after normalization.

IID	reads	size [bp]	kept [%]
dm-il-nkh	518238768	51702495021	35.1
	518238768	51701548333	
dm-il-nbb	575343861	57404351676	39.0
	575343861	57401075176	

2. Sequencing data inventory

2.3. PacBio data

2.3.1. Raw PacBio libraries

Table 2.8.: Inventory of raw PacBio sequencing libraries showing number of reads, total size of the library, read length range (rl), N50 (Burton, 2008) and relative GC content. Libraries are grouped into five batches according to their delivery date, with sums of reads, library size and estimated coverage given in highlighted rows.

IID	reads	size [bp]	rl [bp]	N50 [bp]	GC [%]
dm-pb-001	19875	23234523	50- 7589	2547	49.74
dm-pb-002	51707	84566278	50- 7314	2414	43.96
dm-pb-003	21644	31238647	50- 6946	2078	43.2
dm-pb-004	19012	27057062	50- 7208	2065	43.22
dm-pb-005	24931	35413617	50- 6830	2051	43.08
dm-pb-006	28493	41225571	50- 7515	2109	43.37
dm-pb-007	24484	34450208	50- 6896	2047	43.27
dm-pb-008	16535	26839813	50-11511	2314	43.33
dm-pb-009	19439	32209205	50-11472	2414	43.19
dm-pb-010	21639	36174357	50-10784	2420	43.29
dm-pb-011	35786	61414464	50-11140	2491	43.31
dm-pb-012	24742	42348909	50-10444	2486	43.24
dm-pb-013	27253	80506605	50-17694	4238	44.65
dm-pb-014	13602	46170831	50-19784	4767	44.62
dm-pb-015	17386	51909465	50-21238	4167	44.41
dm-pb-016	19763	56131560	50-16360	3959	44.44
dm-pb-017	18900	52312015	50-16900	3829	44.52
dm-pb-018	19220	54898925	50-18176	4002	44.54
dm-pb-019	22735	71267776	50-17743	4421	44.55
dm-pb-020	22079	56937835	50-18965	3398	44.48
dm-pb-021	28137	74795732	50-17080	3613	44.63
dm-pb-022	31098	80019155	50-17284	3487	44.58
dm-pb-023	49282	133189141	50-17880	3793	45.06
dm-pb-024	42245	114090557	50-19218	3756	44.82
dm-pb-025	22651	58097830	50-17411	3420	44.74
dm-pb-026	28261	75054941	50-18873	3586	45.02
dm-pb-027	26395	69767206	50-18327	3575	44.84
dm-pb-028	21774	60361915	50-16357	3879	44.58
batch 1	719068	1611684143	0.5 X		
dm-pb-029	45916	128383379	50-17971	3802	44.31
dm-pb-030	52729	145693516	50-19716	3815	44.17
dm-pb-031	47661	129369474	50-20675	3698	44.23
dm-pb-032	58967	138878634	50-18583	3123	44.18
dm-pb-033	66222	156215795	50-18863	3088	44.2
dm-pb-034	60675	142391718	50-19306	3090	44.28
dm-pb-035	60368	140907192	50-17717	3060	44.31
dm-pb-036	58985	139631578	50-18190	3090	44.31
dm-pb-037	62602	145582433	50-19103	3002	44.33

Continued on next page

2.3. PacBio data

Table 2.8.: Inventory of raw PacBio sequencing libraries showing number of reads, total size of the library, read length range (rl), N50 (Burton, 2008) and relative GC content. Libraries are grouped into five batches according to their delivery date, with sums of reads, library size and estimated coverage given in highlighted rows.

IID	reads	size [bp]	rl [bp]	N50 [bp]	GC [%]
dm-pb-038	60063	140215114	50-19139	3023	44.31
dm-pb-039	61036	141259560	50-18764	3010	44.43
dm-pb-040	56821	145864585	50-18324	3493	44.49
dm-pb-041	63018	160209182	50-17068	3424	44.58
dm-pb-042	46321	126047840	50-19441	3694	44.66
dm-pb-043	67954	179300595	50-19285	3617	44.73
dm-pb-044	54814	144339645	50-18741	3630	44.37
dm-pb-045	58450	145343059	50-19021	3312	44.34
dm-pb-046	49026	124580239	50-20801	3412	44.45
dm-pb-047	60853	152609840	50-17235	3410	44.44
dm-pb-048	70638	162451560	50-15705	3034	44.63
dm-pb-049	71124	163719032	50-18197	3055	44.52
dm-pb-050	77576	177855053	50-18242	3027	44.5
dm-pb-051	77332	190024586	50-19045	3272	44.54
dm-pb-052	73592	174524235	50-23733	3146	44.5
dm-pb-053	75789	176808047	50-18584	3038	44.66
dm-pb-054	73347	171031969	50-20069	3047	44.79
dm-pb-055	70513	161715621	50-17128	3015	44.64
dm-pb-056	58096	129420197	50-17891	2885	44.34
batch 2	1740488	4234373678	1.4 X		
dm-pb-057	54270	132363604	50-17141	3196	44.45
dm-pb-058	48915	127402612	50-17802	3492	44.4
dm-pb-059	54519	127907670	50-17280	3042	44.39
dm-pb-060	54590	130075405	50-18739	3091	44.68
dm-pb-061	64289	152037939	50-18557	3114	44.67
dm-pb-062	69879	167466560	50-16948	3177	44.77
dm-pb-063	59812	145985395	50-22073	3237	44.56
dm-pb-064	61387	147498698	50-21659	3218	44.53
dm-pb-065	71415	168423532	50-20458	3129	44.52
dm-pb-066	63585	146725019	50-15300	3047	44.38
dm-pb-067	59821	136229580	50-17220	2948	44.4
dm-pb-068	56856	142570662	50-19006	3361	44.37
dm-pb-069	46823	113838183	50-18500	3207	44.04
dm-pb-070	46320	114599515	50-20225	3281	44.21
dm-pb-071	42177	110120483	50-17298	3524	44.16
dm-pb-072	39550	102055387	50-15653	3517	44.26
dm-pb-073	16474	48056257	50-16860	4080	44.5
dm-pb-074	32302	85820936	50-16579	3649	44.39
dm-pb-075	38881	99298426	50-17611	3485	44.32
dm-pb-076	49368	114760797	50-18050	2958	44.6
dm-pb-077	45345	102712412	50-19337	2873	44.43
dm-pb-078	5589	13854669	50-15840	3346	44.11
dm-pb-079	51091	118728700	50-17306	2983	44.55

Continued on next page

2. Sequencing data inventory

Table 2.8.: Inventory of raw PacBio sequencing libraries showing number of reads, total size of the library, read length range (rl), N50 (Burton, 2008) and relative GC content. Libraries are grouped into five batches according to their delivery date, with sums of reads, library size and estimated coverage given in highlighted rows.

IID	reads	size [bp]	rl [bp]	N50 [bp]	GC [%]
dm-pb-080	35737	80930917	50-15548	2875	44.5
dm-pb-081	35392	79704437	50-15284	2801	44.69
dm-pb-082	46830	105249187	50-18167	2854	44.46
dm-pb-083	47366	107318144	50-19006	2877	44.52
dm-pb-084	49563	113799849	50-19741	2984	44.45
batch 3	1348146	3235534975	1.1 X		
dm-pb-085	68016	183880771	50-19273	3693	43.93
dm-pb-086	66431	175750220	50-16365	3475	43.83
dm-pb-087	88276	288051497	50-21703	4399	43.15
dm-pb-088	83200	263473446	50-19218	4291	43.5
dm-pb-089	83995	264854848	50-20788	4233	42.76
dm-pb-090	79245	243822139	50-22057	4116	42.9
dm-pb-091	86507	271530860	50-20840	4195	43.14
dm-pb-092	43037	125615937	50-17740	3984	42.74
dm-pb-093	33634	98510838	50-17843	3971	42.77
dm-pb-094	38528	111749230	50-18569	3951	42.74
dm-pb-095	38042	113262863	50-17065	4050	42.8
dm-pb-096	43230	125502585	50-16804	3917	42.69
dm-pb-097	43710	129385731	50-17783	4011	42.74
dm-pb-098	36353	106056303	50-18534	3914	42.69
dm-pb-099	93237	300988254	50-19811	4341	42.78
dm-pb-100	90480	288496554	50-20021	4275	42.91
dm-pb-101	86933	278851762	50-21098	4280	42.79
dm-pb-102	76488	237751508	50-19321	4142	42.76
dm-pb-103	79697	239296138	50-20515	4018	42.7
dm-pb-104	101696	321321447	50-18631	4266	42.99
dm-pb-105	93084	280983857	50-19668	4057	42.98
dm-pb-106	73252	251259901	50-19158	4516	43.36
dm-pb-107	99480	314909274	50-19799	4261	42.99
dm-pb-108	80235	262669108	50-21535	4320	43.25
dm-pb-109	101129	311528268	50-20298	4149	42.91
dm-pb-110	71572	240959466	50-18979	4416	43.39
batch 4	1879487	5830462805	1.9 X		
dm-pb-111	97267	267880376	500-23536	3227	42.62
dm-pb-112	110121	315311416	500-25666	3350	42.00
dm-pb-113	77034	216630483	500-25404	3253	41.48
dm-pb-114	85905	227264202	500-21024	3074	41.36
dm-pb-115	91222	245976451	500-25982	3153	41.79
dm-pb-116	91836	238411148	500-25242	3004	41.25
dm-pb-117	89632	230817139	500-24002	2980	41.22
dm-pb-118	89519	234314025	500-24013	3028	41.55
dm-pb-119	27704	73992409	500-27649	3128	41.51
batch 5	760240	2050597649	0.7 X		

Continued on next page

2.3. PacBio data

Table 2.8.: Inventory of raw PacBio sequencing libraries showing number of reads, total size of the library, read length range (rl), N50 (Burton, 2008) and relative GC content. Libraries are grouped into five batches according to their delivery date, with sums of reads, library size and estimated coverage given in highlighted rows.

IID	reads	size [bp]	rl [bp]	N50 [bp]	GC [%]
total	6447429	16962653250	5.7 X		

2.3.2. *In silico* mate-pair libraries from corrected PacBio reads

Table 2.9.: Inventory of *in silico* mate-pair libraries generated with pb2il from corrected PacBio reads, with number of reads, total size of the library, read length (rl), insert size (is) and relative GC. Libraries are grouped into subsets according to insert size, with sums of reads, library size and estimated coverage given in highlighted rows.

IID	reads	size [bp]	rl [bp]	is [bp]	GC [%]
dm-pb2il-2000	68209189	6795619379	100	2000	43.22
	68209189	6809720662	100	2000	43.19
total (is=2000)	136418378	13605340041	4.5 X		
dm-pb2il-3000	38940574	3878469696	100	3000	43.21
	38940574	3887657710	100	3000	43.16
total (is=3000)	77881148	7766127406	2.6 X		
dm-pb2il-5000	12349202	1229364230	100	5000	43.26
	12349202	1232859951	100	5000	43.15
total (is=5000)	24698404	2462224181	0.82 X		
dm-pb2il-10000	381572	37916474	100	10000	43.21
	381572	38077716	100	10000	43.05
dm-pb2il-15000	6157	611203	100	15000	43.11
	6157	613715	100	15000	42.82
total (is>=10000)	775458	77219108	0.03 X		
	239773388	23910910736	8.0 X		

3. Software and parameters

3.1. Computing resources

Many of the computational tasks performed during the project required specific architectures and extensive amounts of computational capacities with respect to storage, CPU and memory. Primary analyses were run on the grid architecture maintained at the university of Wuerzburg. The grid comprises three HPC (High Performance Computing) nodes with 30 processors and 192 GB RAM (node IDs: *r5n02*, *r5n03*, *r5n05*) and one large memory HPC node with 40 hyper-threaded processors and 512 GB RAM (node ID: *r5n01*).

External computing resources on low-memory node cluster architectures with a total of several hundred processors were procured through grant applications at the LRZ (Leibniz-Rechenzentrum) and the DIAG (Data Intensive Academic Grid). Further, temporary access to high memory nodes with >1 TB of RAM for individual assembly runs was available through cooperation with Prof. Dr. Chris-Carolin Schön at the Technische Universität München (node ID: *mammoth*) and through cooperation with Prof. Dr. Bernd Weisshaar at the CeBiTec (Center for Biotechnology), Bielefeld University, Department of Biology (node ID: *robin*).

3.2. Sequencing data quality control

Technical integrity and sequencing quality of data were verified with the FastQC (Andrews, 2015) pipeline (v0.10.1, v0.11.2). The tool computes a series of basic statistics from FASTQ (Cock et al., 2009) files. The results are evaluated against internal standards and potential problems are indicated by warnings or critical exceptions. FastQC was applied to (a) Illumina data before and after trimming and (b) PacBio data before and after correction.

3. Software and parameters

3.3. Illumina data processing

3.3.1. Illumina adapter and quality trimming

During sample preparation, DNA fragments for Illumina high-throughput sequencing machines are enclosed in adapters mediating flow cell binding, amplification and sample identification. After sequencing the adapter information needs to be clipped. Similarly regions of low sequencing quality have to be trimmed in order to prevent sequencing errors from affecting downstream analyses (Minoche et al., 2011).

Clipping and trimming of 100 bp and 150 bp reads was performed with `trimmomatic` (Bolger et al., 2014) v0.3. Read pairs with at least one read below a minimum length of 75 bp and 100 bp, respectively, after trimming were removed. Quality was evaluated in 10 bp window requiring a minimum mean Phred quality score (Ewing et al., 1998; Ewing and Green, 1998) of 20 (\cong 99 % base call accuracy).

Mate-pair reads with a length of 49 bp were trimmed with `sickle` v1.200, with a quality cutoff of 25, a length cutoff of 40 and N containing reads removed entirely.

3.3.2. Insert size estimation

option	value	is [bp]
<code>-minins</code>	30	<i>any</i>
<code>-score-min</code>	C,-12,0	<i>any</i>
<code>-N</code>	0	<i>any</i>
<code>-L</code>	32	<i>any</i>
<code>-i</code>	C,20,1	<i>any</i>
<code>-R</code>	1	<i>any</i>
<code>-n-ceil</code>	C,0,0	<i>any</i>
<code>-maxins</code>	1000	< 250
<code>-maxins</code>	2000	> 250

Table 3.1.: Bowtie2 parameters for estimation of insert size (is) distribution

Illumina reads were sequenced in paired-end configuration. Fragments of a targeted insert size were generated by size exclusion during library preparation. The effective insert sizes of the sequenced libraries were evaluated through the following approach:

Reads of each library (100 % of the LGC set, 5 % of the BGI set, sampled with `SeqChunker`) were mapped using `bowtie2` (Langmead and Salzberg, 2012) v2.2.2 in paired-end mode and with strict settings (table 3.1) onto a 10 % sample of proovread corrected PacBio reads. SAM \leftrightarrow BAM conversion was performed with `samtools` (H. Li et al., 2009) v0.1.19 and insert sizes parsed into a table with a custom script. The tables were processed with R (Statistical programming language) and plotted with `ggplot2` (R plotting library) grouped by expected insert size.

3.3.3. Overlap merging

Overlapping libraries are designed in a way that the ends of the two reads of an insert overlap each other. The insert size (S_i) of overlapping libraries needs to be smaller than twice the sequenced read length (L_r). The expected overlap length is given by $2L_r - S_i$. See table 3.2 for typical combinations of read lengths and insert sizes on current Illumina instruments.

The overlap information of read and mate can be used to merge both sequences into one longer fragment. Merging reads reduces sequencing error rates, and downstream processing benefits from increased length. During library preparation, the insert size of a library is determined by fragmentation and subsequent size exclusion procedures (Magoc and Salzberg, 2011); therefore, resulting insert and overlap lengths vary. Accurate overlaps need to be computed based on alignments of the read ends.

The *D. muscipula* BGI read set comprises five overlapping libraries (table 3.3). Merging of read data was performed with FLASH (Fast Length Adjustment of Short reads, Magoc and Salzberg, 2011) v1.2.10. Phred offset was set to 64, read length and insert size were set according to library specifications. A standard deviation of 20 bp and 30 bp were specified for 100 bp and 150 bp libraries, respectively. Maximum mismatch density was set to 0.10.

The LGC set consists entirely of overlapping read data. Merging of read data was performed with FLASH v1.2.10 and default parameters.

3.3.4. Read Error correction

Error correction of Illumina data was performed with the error correction module of ALLPATHS. The ALLPATHS error correction is based on a k -mer approach first described by Pevzner et al., 2001. It utilizes 24-mer stacks to identify reads with underrepresented k -mers and modifies affected reads to only comprise non-erroneous k -mers.

BGI paired-end libraries have been corrected multiple times in different configurations. A full correction of the entire raw read data set requires more than 1 TB of RAM, and thus execution on the r5n01 node with 512 GB failed. The first corrected sequence set was generated by subsetting the given libraries into three batches of similar size and running the ALLPATHS correction on each batch individually. While this way, memory limits are matched, each set is also reduced to about one third of the initial coverage, resulting in subsets of less than $20x$. At this coverage level and, in particular, in the presence of high levels of heterozygosity, the quality of the error correction impaired.

An all-at-once correction of the BGI set was subsequently achieved by utilizing normalized

Instrument	r1 [bp]	is [bp]
HiSeq	100	170-180
HiSeq	150	200-250
HiSeq, MiSeq	250	450
MiSeq	300	550

Table 3.2.: Typical read lengths (r1) and insert sizes (is) of overlapping libraries sequenced on Illumina HiSeq and MiSeq platforms

IID	ol [bp]
Dm_GenI1_[019-021]	30
Dm_GenI1_033	150
Dm_GenI1_034	50
dm-i1-[01-04]	20

Table 3.3.: Overlap lengths (ol) of sequenced *D. muscipula* libraries

3. Software and parameters

libraries as input (see section 3.3.5 for details on normalization). Two corrections were performed, the first with standard settings, the second with `HAPLOIDIFY=TRUE`. In this mode, `ALLPATHS` tries to identify heterozygous sites in addition to errors, and to remove those by consistently choosing k -mers of the same allele for all input reads.

A full correction of the entire BGI set with `ALLPATHS` was generated as soon as the *mammoth* machine at the LRZ with 2TB RAM became available. These reads were further filtered by running the `ALLPATHS` deduplication module before correction. The procedure included a deduplication step before and after correction.

3.3.5. Digital normalization

Digital normalization is a procedure to decrease sequencing data set size and redundancy by identification and filtering of coverage threshold exceeding reads, thus creating a read set with a systematically limited target coverage. If applied to genomic data of a single organism, particularly unique genomic regions are unaffected, while removal of repeat reads largely reduces RAM requirements and speeds up computing in subsequent processing and assembly with little impact on the information content of the generated contigs. (Brown and Crusoe, 2014).

For efficient implementation of digital normalization, reads are assessed by k -mer coverage. The `normalize-by-median.py` script of the `khmer` (Brown and Crusoe, 2014) package uses a single pass approach that filters each read according to its median k -mer coverage. Reads are serially processed and k -mer counts are assessed on-the-fly. Therefore, at the beginning of a run, each k -mer and thus the reads median k -mer coverage is below the target threshold. While running through the reads, counts of abundant k -mers will increase faster than those of k -mers of non-repetitive regions and at some point exceed the specified threshold. If the majority of k -mers of a read, and thus the coverage median, exceeds the threshold, the corresponding read pair will be removed from the set. This can be understood as: If the information encoded in a read was seen often enough, adding more reads of the same composition is omitted, as it only would increase redundancy and not information content.

The `normalize-by-median.py` (khmer-20120916) script was used to generate the read set for the `Dm-gen-ap-6` assembly. As input for the normalization process, error corrected overlapping libraries `Dm_GenI1_[019-021]` and `Dm_GenI1_[033-034]` as well as the 500 bp insert size library `Dm_GenI1_027`, obtained from the `Dm-gen-ap-5` assembly run with the entire BGI short read data set, were used. Digital normalization with a target coverage of 100X was applied. The estimated coverage for the raw reads set is $45x$. $100x$ was chosen in order to reduce excessively repetitive reads but at the same time keep them at a higher coverage than reads derived from unique regions to allow the assembler to still make the distinction of repetitive and non-repetitive regions.

A newer version of `normalize-by-median.py` (khmer-20130520) was applied to the LGC raw libraries `dm-i1-[01-04]`. Target coverage was set to $200x$ given a $80x$ coverage for

the raw set. The resulting reads were used for assembly `Dm-gen-ap-10`.

In 2014, improved digital normalization software packages - `bbnorm.sh` (Bushnell, 2014) and `NeatFreq` (McCorrison et al., 2014) - were published. `bbnorm.sh` uses a faster and less memory demanding k -mer hash and a two-pass approach with bloom filter making it less sensitive to sequencing errors. This is of particular importance as normalization tends to accumulate erroneous k -mers due to their low coverages.

`bbnorm.sh` of the `BBDMap` (Bushnell, 2014) v34.86 release was used to obtain an improved normalized set of the LGC raw libraries. The script requires coverage to be specified as k -mer coverage, rather than expected per-base coverage. A target k -mer coverage of 140 was used, which given 31-mers and 100 bp reads is equivalent to a $200x$ per-base coverage (see section 8.1.1 for details on conversion). The resulting read set was used for assembly `Dm-gen-ap-13`.

3.4. PacBio data processing

3.4.1. Preparation

Sequencing of PacBio read data was performed at the FGCZ (Functional Genomics Center Zurich) on a PacBio RS II platform. Two initial test cells were sequenced in July and September 2012. The additional 117 cells were sequenced in five batches between October 2012 and August 2014. The pre-processed data was sent from FGCZ on hard-disk drives and upon receipt transferred onto the local computing infrastructure.

Processing of the data was carried out with a locally installed and hosted instance of PacBio's web-based SMRT-Portal analysis suite. Annotation of adapters and export of subreads in FASTQ format from PacBio's native HDF5 (Hierarchical Data Format 5) compressed storage format was performed with default settings. For the latest batch, SMRT-Portal was replaced by the faster `dextract` (G. Myers, 2014) module from the `DEXTRACTOR` (G. Myers, 2014) package. Since the tool also provides a command line interface, hosting and maintenance of the SMRT-Portal web-application was discontinued. Application of `dextract` is limited to the recent `bax.h5` data format generated by PacBio instruments since a software update in 2014. Older `bas.h5` encoded data still needs to be converted with SMRT-Portal derived scripts.

3.4.2. Hybrid correction

To prepare the highly erroneous PacBio reads for assembly and downstream processing, the reads were corrected using an Illumina-PacBio hybrid correction approach implemented in the software `proofread`. The development and functionality of `proofread` are described in details in chapter 6 `proofread` – large-scale hybrid PacBio correction through iterative consensus.

3. Software and parameters

PacBio hybrid correction on data sets of gigabase-sized genomes is computationally highly demanding. External computing time on large cluster architectures was procured through successful applications for run-time at the LRZ and the DIAG. Both facilities offer project-specific computing time and architecture free-of-charge to non-profit academic research. Due to software incompatibilities and issues related to data transfer, only the Linux cluster at the LRZ was actually used for correction of *D. muscipula* data. The computing time at the DIAG was used with other data sets during the development of the proovread software.

The entire sequence set of Illumina and PacBio data required for hybrid correction comprises several hundred GB of raw data. Efficient transfer of the large amounts of data was achieved through usage of the globusconnect system, which is specifically designed to facilitate fast and secure transfer of large data set between research infrastructures. A globusconnect client was set-up, configured and used to transfer bzip2 compressed archives of the raw and corrected files to and from the LRZ servers.

Initial corrections of subsets of the first two batches were run with developmental versions of proovread (< v1.00) to evaluate and optimize parameters.

A first full correction was run on all batches available at the time (batches 1-4) with proovread v1.01 and digitally normalized BGI Illumina data. PacBio reads were split with SeqChunker into 10MB chunks and jobs were submitted to the *serial_long* queue with 4 CPUs, 8GB RAM and a 480h job run-time limit per node.

A second full correction was performed on batches 1 to 5 with proovread v2.12 and digitally normalized LGC Illumina reads. PacBio reads were split with SeqChunker into 10MB chunks. Improved speed of the newer proovread version allowed to submit jobs to the *serial* queue with 4 CPUs, 8GB RAM and a 240h job run-time limit per node.

3.5. *k*-mer analysis, genome size estimation and *k*-mer-derived plots

k-mer abundance based analyses, plots and assessments of derived data were performed with custom scripts described in detail in chapter 8 Analysis and visualization of *k*-mer distributions and derived data. Counting of *k*-mers was performed with Jellyfish (Marçais and Kingsford, 2011) v2.2.3 and v2.2.4.

3.6. Assembly and scaffolding software

3.6.1. SOAP-denovo 2

SOAP was developed for human-sized de-novo short read assemblies based on a De Bruijn graph algorithm. The initial *D. muscipula* draft assembly delivered by the BGI was generated with SOAP using merged overlapping reads and a final k -mer size of 127. However, particular details of the assembly process have not been disclosed further, rendering it impossible to reproduce and assess the exact assembly procedure.

First assemblies with the purpose of testing our newly set up computational architecture for the use with SOAP have been run by me and Felix Bemm with SOAP version r239 or older. The runs were executed with mostly default settings, all available libraries and k -mer sizes <100 . In all cases, the resulting assemblies were highly fragmented with contig N50 <1 kbp. Further analysis of the assemblies have been omitted. All assemblies described in the following have been generated with SOAP r240.

The `Dm-gen-so-4` assembly was run with a k -mer size of 127. Input for the initial assembly graph construction were digitally normalized versions of the the BGI libraries with a read length of 150 bp (`Dm_GenI1_[033-035]`). For scaffolding all mate-pair libraries were used, after read pairs comprising repetitive k -mers were removed.

The `Dm-gen-so-5` contig assembly was generated with a k -mer size of 127 and from merged overlapping libraries of the BGI read set (`Dm_GenI1_[019-021,033,034]`).

The initial graph assembly `Dm-gen-so-7` was constructed from all paired-end libraries with 100 bp read length (`Dm_Gen_I1[019-021,027-032]`) and with a k -mer size of 49. Scaffolds were generated with BGI mate-pair libraries of 2 kbp and 5 kbp insert size, filtered for read pairs containing repetitive k -mers, as well as artificial mate-pair libraries generated from corrected PacBio reads (`pb2i1_2000`, `pb2i1_3000`, `pb2i1_4000`, `pb2i15000`). The PacBio mate-pair libraries were generated with the `pb2il` module, which has been implemented by Markus Ankenbrand as an extension to the `perl5lib-Fastq` library (see section 7.1).

`Dm-gen-so-9` was constructed with a k -mer size of 127 and from merged LGC overlapping libraries (`dm-il-[01-04]`). BGI paired-end reads with 500 bp and 800 bp insert size and mate-pair libraries with 2 kbp and 5 kbp insert size were used for scaffolding.

Assemblies `Dm-gen-so-[10-12]` have been generated with k -mer sizes of 67, 77 and 87, respectively. Corrected LGC libraries were used as input. Scaffolding was performed with all non-overlapping trimmed BGI libraries (`Dm_GenI1_[023-032]`).

3. Software and parameters

3.6.2. ALLPATHS-LG

The ALLPATHS (Butler et al., 2008) assembler was originally developed as a De Bruijn graph assembler for bacterial and small eukaryotic genomes. Later, the algorithm was adapted specifically for the use with large, repeat-rich vertebrate genome data and released as ALLPATHS-LG, with *LG* denoting its usability for large genomes. Since then, ALLPATHS was utilized to reconstruct a multitude of large eukaryotic genomes, including human, mouse, rhino, manatee, hedgehogs, a series of rodents and more (The ALLPATHS-LG developmental team, 2016).

Preliminary ALLPATHS assemblies were run by Felix Bemm and me on r5n01 with ALLPATHS R41592, R42682 and R42811. Early runs failed due to technical issues with the newly set-up computing grid and the underlying SuSE Enterprise OS. These problems were fixed by switching to an Ubuntu OS and optimizing the configuration of the system.

On the stable system assemblies continued to fail as the full BGI data set when used with ALLPATHS exceeded the 512 GB available memory. Successfully finishing runs were obtained with reduced data sets. *Dm-gen-ap-3* (s170_f100_c100k, Apr 27 2012) comprised BGI paired-end libraries with an insert-size of 170 bp (*Dm_GenI1_[019-021]*) resulting in an estimated coverage of 15 x . *Dm-gen-ap-4* was run with 25 % of all BGI libraries at an estimated coverage of 29 x .

Assembly *Dm-gen-ap-5* (Blauwal, Dezember 2013) was run with ALLPATHS R47673 on *Mammuth* at the LRZ with 2 TB of RAM. Access to the machine was granted to us by Prof. Chris-Carolin Schön, Department of Plant Breeding, TU Munich. For this assembly the entire BGI read data set was used, and read pre-processing was run independently. First, reads were deduplicated using the `RemoveDodgyReads` module of ALLPATHS, then reads were error-corrected with ALLPATHS followed by a second round of deduplication. The thus obtained read set was used to run the assembly with settings `HAPLOIDIFY=true`, `REMOVE_DODGY_READS_FRAG=false`, `PRE_CORRECT=false` and `ERROR_CORRECT_FRAGS=false`. The HAPLOIDIFY option modifies the behavior of ALLPATHS's error correction module; each heterozygous SNV (Small nucleotide variation) in the data set leads to the occurrence of two populations of k -mers, both at the heterozygous rather than the homozygous coverage. During error correction, these SNV overlapping k -mers are identified by coverage, and consistently corrected to either one of the to possible variants observed for the SNV, thus removing heterozygous noise from the data.

Dm-gen-ap-6 (Wiesel, Oktober 2013) was run with ALLPATHS R47609 on r5n01 using a digitally normalized version of the BGI data set derived from reads corrected during the *Dm-gen-ap-5* assembly procedure. The assembly was run with `HAPLOIDIFY=true`. All raw BGI mate pair libraries were supplied for scaffolding.

Dm-gen-ap-8 (Kuckkuck, October 2014) was a run with the same settings as *Dm-gen-ap-6* and an additional artificial overlapping library of reads with a length of 250 bp and an insert size of 400 bp. These reads have been generated from corrected PacBio long reads

3.6. Assembly and scaffolding software

with the `pb2il` module to add strong overlapping data aiding in ALLPATHS backbone construction. However, the assembly was run without scaffolding libraries.

`Dm-gen-ap-10` (Nerz, September 2014) is the first assembly generated from LGC sequenced overlapping libraries. The assembly was run with ALLPATHS R50599 on `r5n01` using digitally normalized LGC libraries (`dm-il-[01-04]`) and `HAPLOIDIFY=true`. All raw BGI mate pair libraries were supplied for scaffolding. `Dm-gen-ap-12` was obtained from the same data and with the same settings, yet with `HAPLOIDIFY=false`.

`Dm-gen-ap-11` (Orca, November 2014) was generated with ALLPATHS R50599 on *Mammuth* with the full LGC library set, all BGI scaffolding libraries and `HAPLOIDIFY=true`.

`Dm-gen-ap-13` (Honeyguide, December 2014) was constructed from LGC libraries, digitally normalized with `bbnorm.sh`. For scaffolding, all BGI mate pair libraries and libraries with insert sizes of 500 bp and 800 bp (`Dm_GenI1_[027-032]`) as well as *in silico* mate pair libraries generated from corrected PacBio reads (`dm-pb2il-[1235]000-01`, `dm-pb2il-15000-01`) were used; non-default parameters were `HAPLOIDIFY=true` and `MIN_CONTIG=200`. Note that the `MIN_CONTIG` parameter not only affects the length of the reported contigs, but is already applied at graph level and also excludes shorter fragments from the scaffolding process.

3.6.3. ABySS

ABySS belongs to the first generation of De Bruijn graph based short read assemblers capable of handling gigabase-sized genomes. Originally, it has been tested on a full human Illumina data set, later Bombarely et al. (2012) used a combination of ABySS and SOAP to assemble the 3 Gbp genome of *Nicotiana benthamiana*. Birol et al. (2013) used it for the reconstruction of the 20 Gbp genome of the white spruce, *Picea glauca*.

Test runs for *D. muscipula* with ABySS have been performed by Felix Bemm. A proper assembly, however, could not be obtained due to software problems and memory limits.

3.6.4. MaSuRCA

The MaSuRCA (Zimin et al., 2013) genome assembler uses a hybrid strategy, combining advantages of De Bruijn graph and OLC strategies. Illumina short reads are first pre-assembled into *super-reads* and subsequently assembled into contigs, optimally in combination with additional long read data from 454 (Roche 454 Pyro-sequencing) or Sanger sequencing. The method has been shown to outperform SOAP and ALLPATHS on *Rhodobacter sphaeroides* and chromosome 16 of the mouse. In 2014, MaSuRCA was used to assemble the 22 Gbp genome of the loblolly pine (Zimin et al., 2014; Hamilton and Buell, 2014).

`Dm-gen-ms-0` was run with MaSuRCA v2.3.2 on `r5n01` with default settings, trimmed

3. Software and parameters

paired-end LGC reads (`dm-il-[01-04]`) and 2 kbp and 5 kbp raw mate-pair libraries.

`Dm-gen-ms-1` was run with untrimmed paired-end LGC reads (`dm-il-[01-04]`), 2 kbp and 5 kbp raw mate-pair libraries and `GRAPH_KMER_SIZE=69`.

3.6.5. Platanus

Platanus (Kajitani et al., 2014) has been designed particularly to address gigabase-sized heterozygous genome assemblies. It is a De Bruijn graph based assembler and employs successively increasing k -mer sizes and graph simplifications to resolve heterozygous structures during contig and scaffold assembly. Kajitani et al. (2014) showed it to work well on the Assemblathon 2 (Bradnam et al., 2013) data set as well as the highly heterozygous Oyster genome. Further, Roullier et al. (2013) reported notable results for the assembly of different cultivated sweet potato strains.

The `Dm-gen-pt-0` assembly has been generated with Platanus v1.2.1, default settings and all trimmed paired-end LGC reads on `r5n01`.

3.6.6. Meraculous 2

Meraculous2 (Chapman et al., 2015) is a De Bruijn graph assembler developed for large and potentially polyploid genomes. It has been successfully used in the assembly of the 16 Gbp hexaploid wheat genome (Chapman et al., 2015).

`Dm-gen-mc-0` and `Dm-gen-mc-1` have been run with all trimmed paired-end LGC reads. The first assembly was run at a k -mer size of 47 and also included 2 kbp, 5 kbp and 10 kbp scaffolding libraries. The second run was executed with k -mer size 67, which excluded 49 bp mate-pair reads from the assembly.

3.6.7. Minia

The `minia` (Chikhi and Rizk, 2013) assembler employs a highly storage efficient, bloom filter based De Bruijn graph representation. The implementation allows to run gigabase-sized genome assemblies with less than 10 GB of RAM.

The initial run `Dm-gen-mn-0` was performed with `minia` v1.0.3 but did not finish due to program crashes in the k -mer counting step.

Assemblies `Dm-gen-mn-1` to `Dm-gen-mn-5` were run with `minia` v1.0.4 and k -mer sizes 31, 51, 71, 91 and 111. All trimmed and merged reads of the LGC read set have been used.

3.6.8. Discover de novo

Discover de novo (Weisenfeld et al., 2014) is a De Bruijn graph based assembler designed for gigabase-sized genomes sequenced at low cost using a PCR-free HiSeq 2000 (HiSeq-2000 instrument) 250 bp library. The software is heterozygosity-aware and represents assemblies as networks of haplotype contigs rather than as a single haplotype consensus.

`Dm-gen-dc-0` and `Dm-gen-dc-1` were run with Discover de novo R51564 on `r5n01`, trimmed LGC reads with a minimum length of 96 bp and with libraries `dm-il-[01-04]` and `dm-il-[01-03]`, respectively.

`Dm-gen-dc-2` was run with Discover de novo R51828 on `r5n01` and a digitally normalized set of reads from libraries `dm-il-[01-04]`.

3.6.9. MIRA

MIRA is a rather old assembly program, originally designed for Sanger and 454 sequences. Recent versions have been upgraded to offer support for CCS (Circular Consensus Sequencing) as well as corrected PacBio reads.

A test run with proofread corrected *D. muscipula* PacBio reads was performed with MIRA v4.0rc1 and customized settings (listing 1).

3.6.10. Celera

The Celera assembler, also known as wgs-assembler, has initially been designed as a whole-genome shotgun assembler for Sanger read data, and since has been successively upgraded to work with newer technologies, such as 454, Illumina and, since 2013, high coverage raw or hybrid corrected PacBio reads.

The first successful assembly runs with *D. muscipula* data were performed by Felix Bemm with Celera v7.0. `Dm-gen-cl-1` was generated from PacBio reads corrected with proofread 1.01, comprising cells `dm-pb-[001-110]`. For `Dm-gen-cl-2` merged BGI overlapping libraries (`Dm_Gen_I1[019-021,027,033,034]`) were added. For details on parameters see listing 2 `Dm-gen-cl-1` assembly parameter and listing 3 `Dm-gen-cl-1` assembly parameter.

Markus Ankenbrand ran `Dm-gen-cl-3` and `Dm-gen-cl-4` with Celera v8.0, using the same corrected cells `dm-pb-[001-110]`, and for the second assembly, additionally merged BGI libraries `Dm_Gen_I1[019-021,027,033,034]`. For details on parameters see listing 4 `Dm-gen-cl-1` assembly parameter and listing 5 `Dm-gen-cl-1` assembly parameter.

3. Software and parameters

3.6.11. DBG2OLC

The DBG2OLC assembler combines the advantage of Illumina-based De Bruijn graph and PacBio-based OLC assemblies. Next to raw PacBio reads, it uses precomputed Illumina contigs as input. The contigs are used to construct a compressed overlap graph from the PacBio reads with all regions already present in an Illumina contig being removed. This largely simplifies the underlying graph and renders the software highly efficient in terms of speed and computational requirements even for mammalian sized genomes.

A series of assemblies has been generated in order to assess applicability and to optimize the assembler for *D. muscipula* data. All assemblies have been constructed from untrimmed `proofread` corrected reads of the second correction run. In particular, the usage of corrected PacBio reads rather than raw reads has been accounted for in the custom parameter settings.

Assembly	k	KmerCovTh	MinOverlap	AdaptiveTh	LD1	RemoveChimera	Contigs
Dm-gen-do-1	21	20	20	0.2	0	0	Dm-gen-ap-6.1
Dm-gen-do-2	17	2	20	0.02	0	0	Dm-gen-ap-6.1
Dm-gen-do-3	17	1	20	0.001	0	0	Dm-gen-ap-6.1
Dm-gen-do-4	31	1	20	0.001	0	0	Dm-gen-ap-6.1
Dm-gen-do-5	17	2	20	0.02	0	0	Dm-gen-ap-10.1
Dm-gen-do-6	17	2	20	0.02	0	0	Dm-gen-ap-11.1
Dm-gen-do-7	31	1	20	0.001	0	0	Dm-gen-ap-10.1
Dm-gen-do-8	31	1	20	0.001	0	0	Dm-gen-ap-11.1
Dm-gen-do-9	31	1	20	0.001	0	0	Dm-gen-ap-13.1

Table 3.4.: Parameter settings of DBG2OLC runs

3.6.12. PBJelly

PBJelly (Worley, 2014) is a automated pipeline for scaffolding and gap closing of draft genomes with PacBio data. PBJelly was used on assembly `Dm-gen-ap-13` after removal of heterozygous contigs with `proofread` corrected PacBio reads and the following optimized parameters: `minSubreadLength 3000`, `-minMatch 17`, `-minPctIdentity 95`, `-aggressiveIntervalCut`, `-bestn 1`, `-nCandidates 20`, `-maxScore -750`.

3.7. Assembly analysis

3.7.1. Assembly metrics with SeqFilter and QUAST

Assembly metrics, including number and size of contigs / scaffolds, N50 values and base composition were generated with `SeqFilter` v2.1.4. For details on `SeqFilter` see section 7.2 `SeqFilter` – versatile manipulation of sequence files.

Raw assembly statistics were further assessed with **QUAST** (A. Gurevich et al., 2013). The software produces a comparative table including contig numbers, total assembly length and longest contigs. As single value measurement for contig size distribution, the N50 length statistic is used.

Base composition is given by percentage GC and the fraction of N's per 100 kbp. Additional plots provide details on the distributions of specified statistics over the range of the given contigs.

Unless otherwise stated, all **QUAST** derived statistics were obtained with a minimum contig length of 1,000 bp.

3.7.2. Completeness of gene content with CEGMA and BUSCO

CEGMA (Parra et al., 2007) and **BUSCO** (Simão et al., 2015) are programs designed to assess the completeness and quality of gene models present in genome assemblies. Both programs rely on reference gene sets comprising near-universal single-copy orthologs. In comparison to **CEGMA**, **BUSCO** uses larger gene sets specifically designed for different kingdoms, such as arthropods, vertebrates, fungi or plants. Further, **BUSCO** employs a more complex gene detection procedure increasing its sensitive and predictive power.

Gene content analysis in this work were performed with default settings for both programs and with plant specific reference profiles in the case of **BUSCO**.

3.7.3. Transcriptome coverage

Coverage of the *D. muscipula* reference transcriptome `DM_tra_qt1.03.RefSeq` by analyzed genome assemblies was assessed using a custom mapping based approach. To simplify statistical evaluation only the longest isoform of each unigene was used as single representative for the respective cluster. The resulting set comprised 42,653 transcripts in the range of 145 bp to 15,692 bp in length and with a total size of 46,601,336 bp. The transcripts were mapped onto genome assemblies and corrected PacBio reads with **BWA-MEM** (H. Li, 2013) v0.7.10-r868-dirty and converted to **BAM** (Binary SAM, SAM/BAM Format Specification Working Group, 2015) format with **samtools** v1.2. Using **BWA-MEM** as mapper rather than an aligner specifically designed for transcripts was based on practical considerations, in particular computational feasibility. Although, the resolution **BWA-MEM** mappings is not sufficient to capture the entire set of exons, in particular exons shorter than 75 bp, the approach provides a reasonable approximation to infer the coverage of coding regions by an assembly.

The horizontal coverage of each transcript was estimated with a custom **Perl** script. For each transcript, the relative amount of bases mapped at least once in an alignment with a normalized score of 0.9 (section 6.4.1 **Sam::Alignment**) were computed for (a) alignments on the same reference sequence and (b) alignments on any reference sequence.

3. Software and parameters

For the stacked histograms generated, alignments with an overall coverage of at least 90 % were considered *full-length*. Full-length mappings were further classified as *split* if alignments were distributed across more than one reference sequence. For partially covered transcripts, distinction between split and unsplit was omitted because an unaligned regions needs to be consider as not mapped to the same reference.

4. Assembling the genome of *D. muscipula*

The overall goal of the Venus flytrap genome project is to gain comprehensive insights into the genetic mark-up of the non-model organism, prototype carnivorous plant *D. muscipula*. The construction of a high quality draft assembly as a reference for downstream analyses is an essential first step in this quest. The generated assembly needs to provide a basis for gene prediction and annotation, based on which functional predictions and interpretation, particularly with respect to the carnivorous life style of the plant can be made. Additionally, we aim to explore regulatory features associated with specific genes, such as upstream promoters or enhancers, particularly if involved in carnivory related processes. Next to the individual gene-based analysis, comparative analysis of genes and genomic regions to other carnivorous and non-carnivorous plant are of high importance to derive new insights into the differences, novelties and adaptations that the *D. muscipula* genome acquired on its evolutionary path to a flesh eating plant.

In order to be able to perform the aforementioned analyses, the generated draft genome must meet the following criteria: The assembly needs to be complete in a sense that, in particular, the entire gene set and other elements of interest are represented. The contiguity of the assembly must at least allow the identification of full-length gene models, ideally, it should comprise entire gene clusters including up -and downstream regulatory elements. Moreover, the amount of assembly errors should be kept to a minimum and artifacts, such as redundancy of contigs due to heterozygosity need to be reduced as far as possible.

4.1. Evaluation of the BGI draft assembly reveals substantial deficiencies

4.1.1. Basic metrics

The initial *D. muscipula* draft genome assembly was generated by the BGI and delivered at the end of 2011. The BGI employed an Illumina-based paired-end and mate-pair sequencing strategy, tailored to work well with their in-house developed large genome assembly software SOAP-denovo2. In total, the assembly is based on 35 libraries, 17 in paired-end configuration with reads of 100 bp or 150 bp in length and insert-sizes ranging from 180 bp to 800 bp, and 18 mate-pair libraries with 49 bp reads and inserts ranging from 2 kbp to 20 kbp. For the draft assembly, a customized strategy was employed by

4. Assembling the genome of *D. muscipula*

the BGI, details have only partially been disclosed. To my knowledge, first, overlapping reads were merged in a way similar to the approach described in section 4.5.5 Merging of overlapping libraries. The resulting fragments were assembled with SOAP and a k -mer size of 127 bp. Subsequently the assembly was scaffolded and gap-closed using paired-end and mate-pair libraries.

Metrics	Dm-gen-so-1.1	Dm-gen-so-1.2
Contigs \geq 0 bp	8470,199	6571,343
Contigs \geq 1 kbp	476,500	337,902
Length \geq 0 [bp]	3167,489,463	3752,455,584
Length \geq 1 k[bp]	784,079,898	2287,688,755
Longest [bp]	16,745	355,463
N50 [bp]	514	3350
GC [%]	42.4	42.4
N [%]	0.0	21.9

Table 4.1.: Metrics of assembly Dm-gen-so-1. Contigs: number of Contigs or Scaffolds; Length: length of the total assembly; Longest: length of longest contig/scaffold; N50: length of the sequence that together with all longer sequences adds up to $\geq 50\%$ of the total assembly length; GC: percentage of nucleotides G and C relative to A,T,G and C (N-nucleotides are ignored); N: percentage of N-nucleotides in the total assembly.

35.8%.

The basic statistics of the assembly, on contig as well as scaffold level, show that the *D. muscipula* draft assembly generated by the BGI is highly fragmented and discontinuous. The majority of contigs is less than 1 kbp in length, contigs equal or longer than 1 kbp comprise 784 Mbp, contigs larger than 10 kbp less than 1.2 Mbp. The estimated haploid genome size for the Venus flytrap is 2.8 Gbp. The total contig assembly length is in congruence with this estimate. However, when looking at longer contigs of reasonable size for downstream analysis (< 1 kbp), a substantial part of the genome is missing.

Scaffolding substantially improves the contiguity of the assembly, resulting in about 2.3 Gbp of contigs equal to or longer than 1 kbp and 1.6 Gbp longer than 10 kbp. However, with 35.8%, a significant fraction of the scaffolds consists of uninformative N's, and when looking at the total assembly size of 3.75 Gbp including short contigs, the assembly exceeds the expected genome size by almost 1 Gbp.

4.1.2. Conserved gene content

The completeness of an assembly on gene level can be inferred by annotation of conserved ubiquitous single copy genes. I performed conserved gene annotation on for the BGI assembly with two tools, CEGMA (on contigs only) and BUSCO. Respective results are

The complete draft assembly comprises 8.4×10^6 contigs with a minimum length of 128 bp and a total assembly length of 3.2 Gbp. The longest contig has a length of 17 kbp. If one ignores contigs shorter than 1 kbp, a recommended practice, given that very small contigs are of little value to – and at the same time a complicating factor for – downstream analysis and annotation, the number of contigs decreases to 4.7×10^5 and a total assembly length of 784 Mbp. The N50 for the minimum 1 kbp contig set is 1.6 kbp.

After scaffolding, the assembly comprises 6.6×10^6 scaffolds with a total length of 3.7 Gbp. Omitting scaffolds shorter than 1,000 bp, the number of scaffolds is 337,902 and the total scaffold length is 2.3 Gbp. The scaffold N50 is 26 kbp. The fraction of N's in the scaffold assembly is

4.1. Evaluation of the BGI draft assembly reveals substantial deficiencies

shown in table 4.2 and table 4.3.

		Proteins	Completeness [%]	Total	Average	Orthologs [%]
contigs	Complete	47	18.95	78	1.66	46.81
	Partial	126	50.81	271	2.15	63.49

Table 4.2.: CEGMA analysis of assembly Dm-gen-sp-1.1.

CEGMA analysis of the assembly Dm-gen-so-1.1, based on a total of 248 conserved protein templates, revealed 47 complete (18.95 %) and 126 partial hits (50.81 %). On average, for each annotated complete protein, 1.66 copies and for each partial hit, 2.15 copies were found.

The very low number of complete genes can mainly be attributed to the high level of fragmentation of the assembly; the large amount of partial hits corroborates this interpretation. Nevertheless, even if partials and completes are added up, a substantial fraction (30.2 %) of expected conserved genes is missing from the assembly. Furthermore, the high average number of hits per gene indicate a high level of redundancy in the assembly for gene containing contigs, congruent with alleles being assembled into separated sequences.

Core gene annotation with BUSCO based on 956 near-universal single-copy orthologs resulted in 6.1 % of complete and 2 % of partial genes found in the contig assembly, and 32 % of complete and 3.5 % of partial genes in the scaffold assembly. Duplication rates were 0.2 % and 0.8 % for contig and scaffold analysis, respectively.

	in [%]:	compl.	dupl.	part.
Dm-gen-so-1.1		6.1	0.2	2.0
Dm-gen-so-1.2		32	0.8	3.5

Table 4.3.: BUSCO analysis of assembly Dm-gen-sp-1 based on 956 near-universal single-copy orthologs

The amount of detected core genes with BUSCO is even lower than the results obtained with CEGMA, although scaffolding substantially increases the number of annotated genes. Overall, the CEGMA and BUSCO analyses supports the general assumption that the BGI draft assembly is fragmented beyond the level of genes. This renders the assembly incomplete at least in the sense that contigs of length <1 kbp, even if they contain parts of genes, due to the lack of context cannot be annotated. Moreover, duplication levels reported by CEGMA suggest a considerable degree of redundancy, potentially caused by heterozygosity.

4.1.3. Transcript coverage

While CEGMA and BUSCO analyses provide an indirect completeness estimate based on homology of universal conserved core genes, a picture more specific to *D. muscipula* can be obtained by assessing the coverage of the *D. muscipula* transcriptome by the genome assembly.

Figure 4.1 shows the results of the transcriptome coverage analysis for the draft assembly Dm-gen-so-1. The stacked histogram bars indicate the amount of transcripts that

4. Assembling the genome of *D. muscipula*

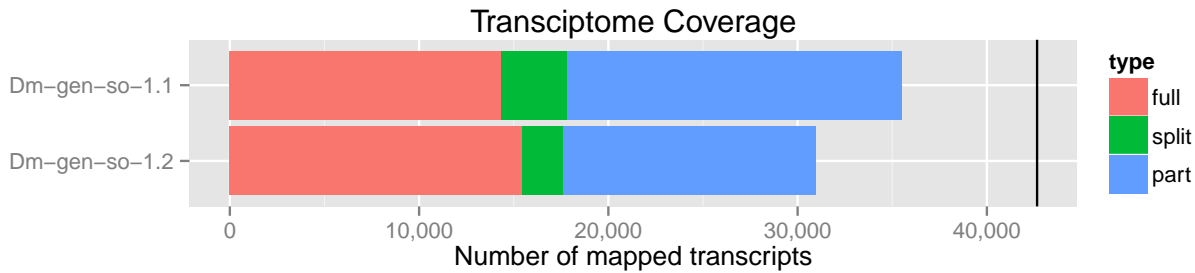


Figure 4.1.: Transcriptome coverage of the BGI assemblies `Dm-gen-so-1.1` (contigs) and `Dm-gen-so-1.2` (scaffolds). Bars of the stacked histogram represent the amount of full-length (red), full-length but split across reference sequences (green) and partially (blue) mapped longest isoforms with maximum 42,653 (black line) derived from transcriptome assembly `DM_tra_qt1.03`.

mapped to contigs and scaffolds, respectively. About 42% (approx. 18×10^3) of the available transcripts were mapped in full length, either on the same or split across multiple sequences of the assemblies. For the scaffold assembly, the number of split mappings was slightly lower than for the contig assembly. A similar fraction of transcripts (18×10^3) aligned partially to the contig assembly, yet only 30% (13×10^3) to the scaffold assembly.

In particular, the low fractions of full-length mappings support the argument that the assembly generated by the BGI is fragmented and that for a considerable fraction at least parts of the genes are missing.

4.1.4. Contig coverage and GC landscape

To further explore the underlying reasons for the high level of fragmentation of the BGI assembly, contigs and scaffolds were analyzed in the context of their coverage and GC content. Existing tools for this type of analysis, such as *Blobology* (Kumar et al., 2013), require read mappings in order to obtain per sequence coverage information. This approach is impractical for the large *D. muscipula* genome, as it would require several days per analysis as well as extensive hard-disk space.

Therefore, I utilized a much more efficient k -mer count based approach, which I specifically developed for this purpose (see chapter 8 *Analysis and visualization of k -mer distributions and derived data for details*). k -mer coverages can easily be obtained in less than an hour for an entire *D. muscipula* assembly. This allows to repeatedly run the procedure on different assemblies, and thus to use it as a practical tool for comparative evaluation.

Raw k -mer coverage can be used directly to assess the repetitive nature of the analyzed sequences. However, for read mapping based coverage approaches, it is common practice to use random placement for multi-mappings, thus encoding relative usage information for certain sequence region in the coverage values: Assuming ideal data, a sequencing depth of $100x$ and a perfect assembly, a unique region will have a coverage of $100x$ and

4.1. Evaluation of the BGI draft assembly reveals substantial deficiencies

will occur once in the assembly, i.e. have a copy number (CN) of 1. A repeat with 5 copies will gather reads depending on how much it collapsed. If only one copy of the repeat is present in the assembly, all reads will be assigned to it and the coverage will be around $500x$. If all five copies are present in the assembly, reads will be randomly assigned to either one copy and the coverage for each copy will be approximately $100x$.

The same effect can be obtained using k -mers by computing an *adjusted coverage* (section 8.4 *kmer-coverage – representative, frequency adjusted k -mer coverage*). Analogous to the above example the adjusted coverage is given by the ratio of coverage to copy number. In case of k -mers, the k -mer copy number can be obtained by counting the occurrence of the k -mer in the assembly. The resulting coverages encode k -mer and, by extrapolation, sequence over- and underrepresentation in a given assembly. For collapsed repeats the resulting adjusted coverage for the corresponding k -mers will be higher than the sequencing depth. Similarly, if heterozygosity leads to separate assembly of alleles, non-allele-specific k -mers will be overused, giving them higher copy numbers, which results in reduced adjusted coverages. Overall, adjusted k -mer coverages are largely equivalent to mapping derived coverages with random placement of multi-mapping, but can be computed far more efficiently.

Moreover, while existing approaches use either mean or median coverage to characterize the coverage of a sequence by a single value, my implementation employs an algorithm that computes another so called *representative coverage*, which is given by the peak coverage of the largest population of k -mers with similar coverage in the sequence (section 8.4 *kmer-coverage – representative, frequency adjusted k -mer coverage*). This approach is particularly robust on sequence sets comprising regions with different copy numbers, e.g. contigs containing different repeat species as well as unique and low copy number regions.

The following plots describe the coverage landscape of *D. muscipula* assemblies based on either raw median or representative adjusted k -mer coverages. The k -mer coverages used for the computation of the raw distribution and the assembly coverages derive from the LGC Illumina read data set and not the BGI read set that was actually used to generate the assemblies. While this introduces a slight bias due to k -mers of the BGI set missing in the LGC set, the analysis greatly benefits from the less biased, higher covered and more robust LGC k -mer set (see section 4.2 for details on BGI data set issues).

Analysis of cumulative contig lengths with respect to raw median coverage for contigs and scaffolds (fig. 4.2) shows that the majority of sequences in the assembly is short and of repetitive character while only a minor fraction of assembled sequences (<300 Mbp) is non-repetitive. The amount of longer contigs is much greater for lower coverages. This is in congruence with assembly theory, which only guarantees unambiguously resolvable graphs for unique sequences.

If one employs representative adjusted coverage instead of raw median coverage, the topology of the distribution changes: The long tail of high coverage sequences disappears (fig. 7) and the data accumulates at coverage between $40x$ and $200x$ for contigs as well

4. Assembling the genome of *D. muscipula*

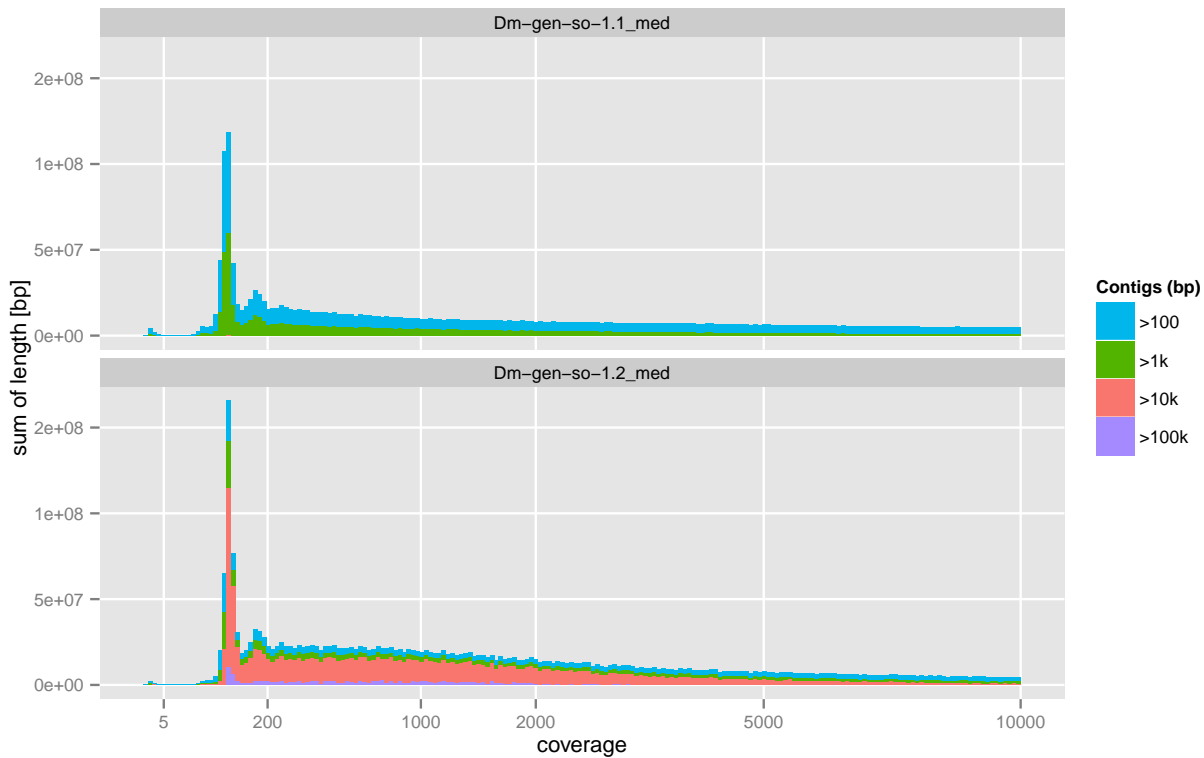


Figure 4.2.: Sequence length distribution of assembly `Dm-gen-so-1` as a function of raw median k -mer coverage. Stacked histogram bars depict cumulative length of contigs (upper panel) and scaffolds (lower panel) in different length intervals (color-coded, largest at the bottom). Coverages are scaled by Anscombe transformation (section 8.2) and limited to a maximum of $10,000x$.

as scaffolds, as shown in fig. 4.3 with a maximum coverages $>300x$, missing less than 1% of data points due to the cut-off.

The majority of assembled sequences is comprised of sequences between 100 bp and 999 bp, and a minor part is comprised of contigs >1 kbp. Contigs greater than 10 kbp are missing. The distributions exhibit a single prominent peak with a maximum between $50x$ and $60x$. The three largest bars in total comprise about 2 Gbp of the total 3.2 Gbp assembly. For scaffolds, sequence length is shifted towards sequences >10 kbp with a small fraction of greater than 100 kbp (<200 Mbp). The two major bars comprise ~ 3 Gbp of sequences of the total 3.8 Gbp assembly.

The raw k -mer distribution (dotted black line) obtained from the Illumina read data is plotted on top of the histogram. It exhibits a bimodal distribution with a major peak at $\sim 80x$ corresponding to k -mers shared between alleles of the diploid genome and a minor peak representing k -mers unique to a haploid copy of the genome.

As expected given the large assembly sizes close to the expected 3GB, the vast majority

4.1. Evaluation of the BGI draft assembly reveals substantial deficiencies

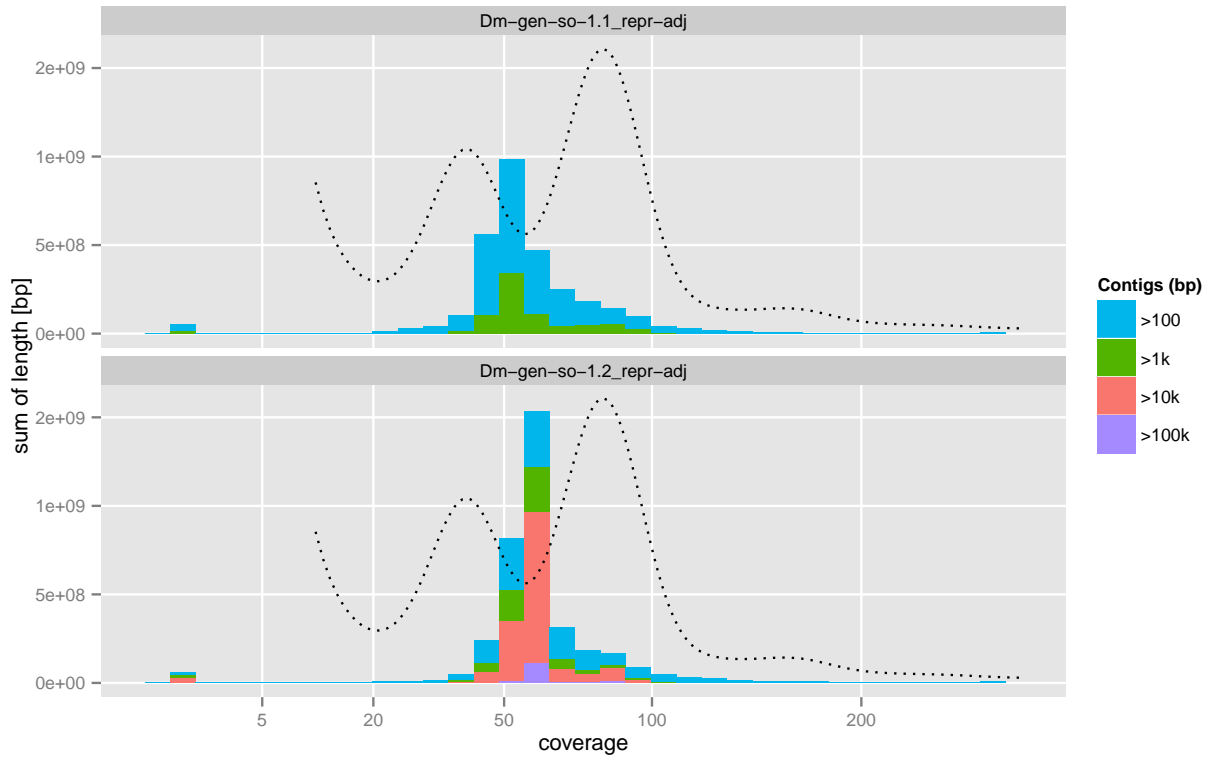


Figure 4.3.: Sequence length distribution of assembly `Dm-gen-so-1` as a function of representative adjusted k -mer coverage. Stacked histogram bars depict cumulative length of contigs (upper panel) and scaffolds (lower panel) in different length intervals (color-coded, largest at the bottom). Coverages are scaled by Anscombe transformation (section 8.2) and limited to a maximum of $300x$. The raw k -mer distribution obtained from LGC Illumina reads (black dotted line, section 4.2 Assessment of the BGI read data shows technical bias and high level of heterogeneity) and with frequencies scaled to fit the size of the plot, exhibits a bimodal distributions with peaks corresponding to haplotype-specific ($40x$) and haplotype-shared k -mer populations ($80x$)

of contigs is represented by coverages close to the sequencing depth, indicating a overall low level of collapsed repeats. Ideally, one would expect the contig coverage peak to coincide with the diploid k -mer peak. For the BGI assembly, however, the peak does not fit the diploid peak nor does it match the haploid peak. It majority of sequences is represented by a coverage in between diploid and haploid. At a first glance, this result is surprising. For unique regions, one would expect that the bimodal distribution of the underlying k -mer to be retained. A k -mer can be used once, which would not change the adjusted coverage, twice, which would half the coverage, shifting it to the haploid peak and indicating allele specific assembly, or several times, indicating great overuse and resulting in below haplo-peak adjusted coverage.

In the case of the flytrap, however, the majority of sequences consists of repetitive regions with high copy numbers and hence high raw coverages. The adjusted coverages of these k -mers are generated by dividing the raw coverage with the observed occurrence in the

4. Assembling the genome of *D. muscipula*

assembly. Proper amount of usage would put the k -mer coverage in the range of the diploid peak. In contrast to unique k -mers, overuse of repetitive k -mers can be more gradual. While a unique k -mers can be used either once or twice, which already means overuse by a factor of 2, a k -mer with a CN of 10, can for example be used 15 times. This also means overuse but only by a factor of 1.5. Based on this fact the obtained representative coverages for the BGI assembly sequences can be interpreted as follows:

The majority of sequences in the assembly is dominated by repeats with a relative overuse factor of 1.5. The overuse of these sequences adds redundant information and is the primary reason for the overall oversized assembly. There are several contributing factors potentially causing relative overuse of repetitive sequences: (1) allele specific assembly of unique regions driving overuse of adjacent repetitive regions, (2) increased level of heterozygosity in repeats due to pooling and (3) an assembly strategy *optimized* to generate a large assembly, e. g. by fusing different draft assemblies.

4.1. Evaluation of the BGI draft assembly reveals substantial deficiencies

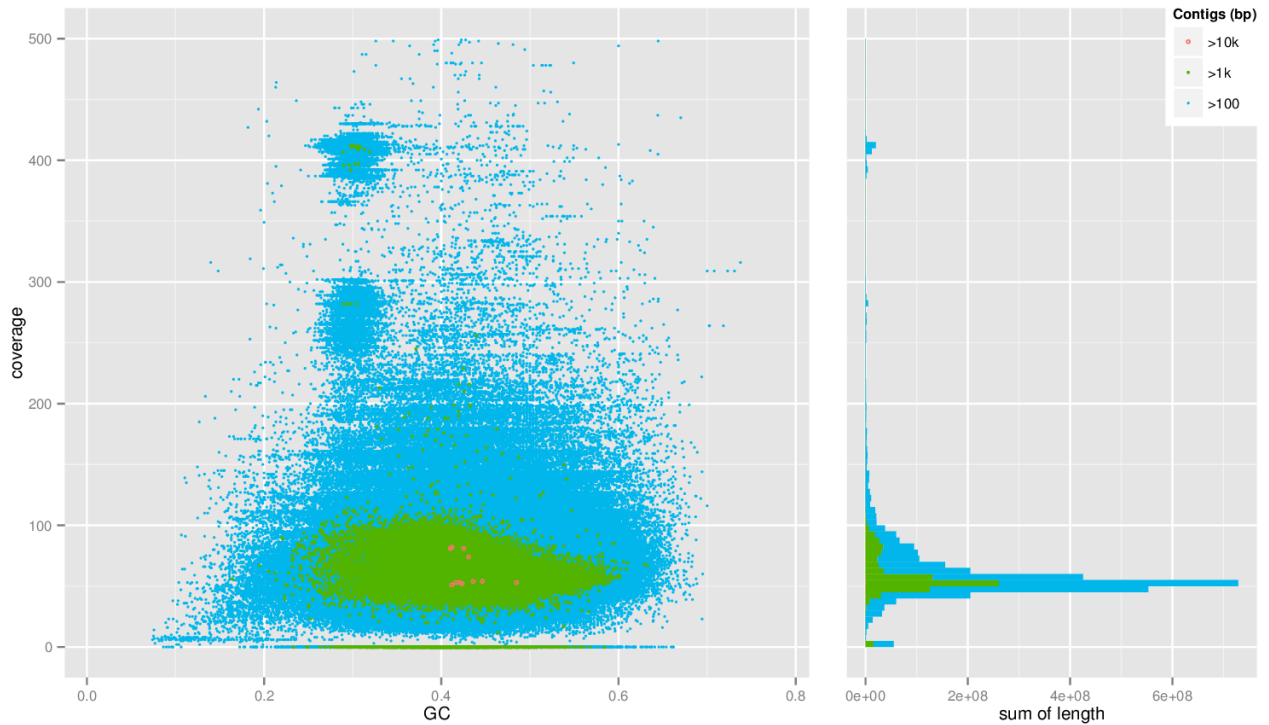


Figure 4.4.: GC-content and sequence length distribution of assembly `Dm-gen-so-1.1` as a function of representative adjusted k -mer coverage. **Left Panel:** Circles represent individual sequences of a 1% subsample with length encoded by size and color. Subsampling increases readability while the overall topology of the scatter plot is preserved. **Right Panel:** Histogram bars depict cumulative lengths of the entire sequence set binned according to length intervals, color coded and stacked in decreasing order.

To further resolve the structure of the assembly I included GC composition as an additional parameter next to length and coverage. Figure 4.4 and fig. 4.5 show the resulting GC-coverage composite plots for the BGI contig and scaffold assembly. The right panel is a stacked histogram of cumulative lengths per bin. In the left panel, each circle represents a sequence in the assembly, with relative GC content on the x-axis and representative adjusted coverage on the y-axis. Color and size of the circles represent the length of the sequence. To allow the graphs to be properly readable, only a 1% subsample of the all sequences is displayed in the scatter plot. The overall structure and topology of the graph is well preserved by this down-sampling procedure, however differences and clustering behavior are much easier to recognize in the thinned out plot.

The vast majority of sequences in both plot are sequences of <1 kbp (blue). These sequences also exhibit the largest variance in terms of coverage as well as GC content. Larger sequences (green, red, purple) are mostly located between $40x$ and $100x$ and 30% to 50% GC. Within this accumulation, two clusters can be distinguished: a larger cluster at $60x$ and $>40\%$ GC and a smaller cluster at $80x$ and $<40\%$ GC. Additionally, at least two clusters can be identified at 35% GC and coverage $>200x$. These two clusters

4. Assembling the genome of *D. muscipula*

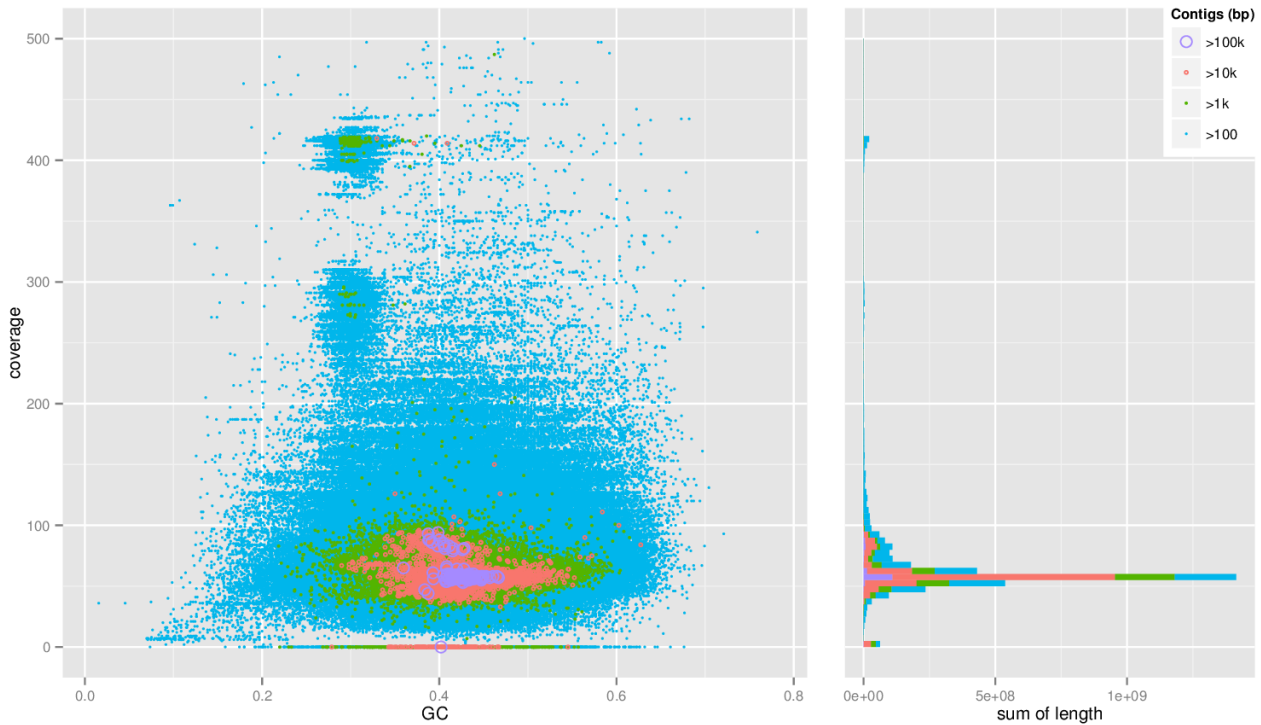


Figure 4.5.: GC-content and sequence length distribution of assembly *Dm-gen-so-1.2* as a function of representative adjusted k -mer coverage. See fig. 4.4 for detailed plot description.

comprise roughly 50 Mbp of sequences, however, the high coverages are indicative of a collapsed nature under the assumption that the sequences derive from the nuclear genome and not a high coverage contamination or organelle genome. Properly resolved, these clusters could make up several hundred Mbp and hence represent a significant proportion of the overall assembly.

The analysis of sequence length and GC-coverage composition of the BGI assembly further corroborate the findings of the previous analyses: the BGI assembly is highly fragmented. The fragmentation, on the one hand is related to the high amount of repeats present in the data, on the other hand seems to be equally driven by a high level of heterozygosity. Although the assembly roughly fits the expected size of about 3 Gbp, GC-coverage analysis revealed that this is an artifact caused by having both, overrepresented repeats and missing data. Downstream analyses performed on the BGI assembly and aimed at identifying genes etc. would therefore fail to produce proper and conclusive results.

4.2. Assessment of the BGI read data shows technical bias and high level of heterogeneity

4.2.1. Quality control of BGI libraries

The general quality metrics of the BGI sequencing libraries were compiled with FastQC. A summary of the results for raw paired-end and mate-pair as well as trimmed paired-end libraries is shown in fig. 4.6.

More than half of the raw paired-end libraries failed the test on *Per base sequence quality*. The reason for this is the typical 5' quality bias, an issue that is solved by trimming of low quality tails. As a result all trimmed libraries pass the test but at the same time receive warnings for *Sequence Length Distribution*, as trimmed reads differ in length. The warning issued across all libraries for *Per base sequence content* can be attributed to a minor compositional bias at the 3' end of the reads.

A more complex picture emerges from the analysis of *Per sequence GC content*. Figure 4.7 shows plots of the number of observed reads versus the relative GC content for raw paired-end (upper panel) and raw mate-pair libraries (lower panel). The expected shape for this distribution under generic conditions is a single Gaussian peak. However, for genomic DNA of *D. muscipula*, a bimodal distribution with a large peak at 44% and a smaller peak at 31% relative GC content is observed. The deviation from the expected Gauss distribution is the reason for the warn/fail states in the FastQC tests. The consistency of the bimodal distribution across libraries suggests an underlying biological phenomenon. The second, smaller peak comprises a subpopulation of reads of low GC content distinctly different from that of the majority of reads. Given the repetitive character of the flytrap genome, a likely explanation is that these reads derive from a species of highly abundant elements that comprise at least large portions of low GC regions. However, solely based on this analysis, a definitive assessment cannot be made. Potential alternative explanations include systematic contamination, e.g. by associated symbionts / parasites or an extreme technical bias.

Furthermore, warnings and failures are reported for *Sequence Duplication Level*, *Overrepresented Sequences* and *k-mer Content* for different libraries. The issues implied by all of the aforementioned categories need to be interpreted in the context of the repetitive nature of the flytrap genome, by which they are affected in similar fashion. The underlying assumption for the applied models are diverse, unenriched libraries (Andrews, 2015). The presence of high copy repeat species, however, contradicts this assumption, because highly abundant repeat derived reads behave similar to highly enriched reads resulting from a particular technical or experimental setup. Therefore, the poor performance in the categories in question can be attributed to a particular biological scenario rather than technical issues related to library preparation or sequencing. The difference in the results for mate-pair and paired-end libraries can be explained by a) the substantial difference in read length (100 bp vs. 49 bp) and b) the difference in library preparation,

4. Assembling the genome of *D. muscipula*

with mate-pair libraries undergoing a more complex procedure that introduces additional biases.

4.2.2. Insert-size distribution of BGI libraries

Typically, Illumina sequencing generates reads in paired-end conformation as a result of the *bridging* based amplification procedure for the DNA on the sequencing chip (Quail et al., 2012). The two reads of a pair are generated from both ends of a longer DNA fragment, called *insert*. For paired-end data, the distance of the reads is given by the *initial size* of the sequenced fragment. Orientation wise, the reads face each other. Typical distances are 180 bp to <1 kbp. Mate-pair reads with larger insert-sizes (1 kbp - 20 kbp) are generated with a more complex protocol, utilizing a circularized intermediate, resulting in a reversed orientation of the reads of a pair.

The information about the insert-size / distance of the reads in a pair is used in assembly to connect contigs into scaffolds. The initial fragments are generated by size exclusion during library preparation. In order to use read pair information properly during assembly, it is important that the resulting sizes for each library are of similar (and desired) length.

Paired-end libraries

Figure 4.8 show the distribution of insert-sizes across all BGI libraries sequenced in paired-end confirmation. All libraries exhibit a single peak in the distribution of sizes coinciding with the expected insert-size within an acceptable margin of error. For the majority of libraries, the peak is narrow and sharp, indicating unbiased library preparation. Lower peaks can be observed of libraries of 200 bp and 800 bp insert-size, as well as libraries Dm_GenI1_024 (350 bp) and Dm_GenI1_035 (500 bp). These libraries are of lower quality with respect to insert-size, however, still can be used for scaffolding.

Mate-pair libraries

The procedure for the estimation for insert-sizes for mate-pair libraries is the same as for paired-end libraries. However, obtaining conclusive results is more difficult. Since the reads of a pair need to be aligned to a single reference sequence in order to compute their distance reliably, having reference sequence larger than the insert-size of the tested library is a requirement. In the case of the flytrap, however, the vast majority of contigs generated in assemblies is shorter than the mate-pair insert-sizes and, in particular, for the 10 kbp and 20 kbp libraries, practically no potential reference sequence for insert-size assessment exist in the available contig assemblies. This problem is illustrated by the results obtained from ALLPATHS insert-size size estimation module for mate-pair libraries shown in table 4.4. For the 2 kbp and 5 kbp libraries, the results match the

4.2. Assessment of the BGI read data shows technical bias and high level of heterogeneity

targeted sizes, and are consistent between different assemblies. Therefore, these libraries are of good quality.

For the 10 kbp library, estimated sizes are far off. However, when comparing alignments `Dm-gen-ap-6` and `Dm-gen-ap-13`, a clear trend can be observed. The estimated sizes for the second assembly are much larger and much closer to the expected size. The reason for that is the increased contiguity of the underlying `Dm-gen-ap-13` contig assembly. While the on `Dm-gen-ap-6`, the estimates are almost entirely driven by improperly mapped pairs, a much larger fraction of pairs actually mapping with larger distances can be observed for `Dm-gen-ap-13`. In case of the 20 kbp libraries, `ALLPATHS` did not report any estimate in the first place due to a lack of large enough reference contigs. Therefore, we can conclude that the mapping-based insert-size estimation for the large insert-size mate-pair libraries are inconclusive and a reliable assessment regarding the quality of the libraries cannot be made.

library	target	Dm-gen-ap-6	Dm-gen-ap-13
IID	is	is \pm sd [bp]	is \pm sd [bp]
<code>Dm_GenIl_001</code>	2000	2017 \pm 148	2016 \pm 149
<code>Dm_GenIl_002</code>	2000	2035 \pm 227	2042 \pm 228
<code>Dm_GenIl_003</code>	2000	2030 \pm 141	2022 \pm 146
<code>Dm_GenIl_004</code>	2000	2053 \pm 228	2044 \pm 234
<code>Dm_GenIl_005</code>	5000	4936 \pm 185	4912 \pm 184
<code>Dm_GenIl_006</code>	5000	4697 \pm 194	4722 \pm 210
<code>Dm_GenIl_007</code>	5000	4926 \pm 176	4909 \pm 176
<code>Dm_GenIl_008</code>	5000	4702 \pm 188	4726 \pm 216
<code>Dm_GenIl_009</code>	10000	1858 \pm 1664	3850 \pm 1330
<code>Dm_GenIl_010</code>	10000	400 \pm 393	2910 \pm 2449
<code>Dm_GenIl_011</code>	10000	414 \pm 400	2830 \pm 2473
<code>Dm_GenIl_012</code>	10000	412 \pm 396	2769 \pm 2444
<code>Dm_GenIl_013</code>	10000	1856 \pm 1653	3717 \pm 1331

Table 4.4.: Insert-size distribution of different BGI mate-pair libraries as estimated by `ALLPATHS` during assembly runs `Dm-gen-ap-6` and `Dm-gen-ap-13`. Library sizes are estimated based on alignments of pairs to previously generated contigs. Lack of sequences larger than the insert size in the initial contig set render the estimation faulty and inconclusive.

4. Assembling the genome of *D. muscipula*

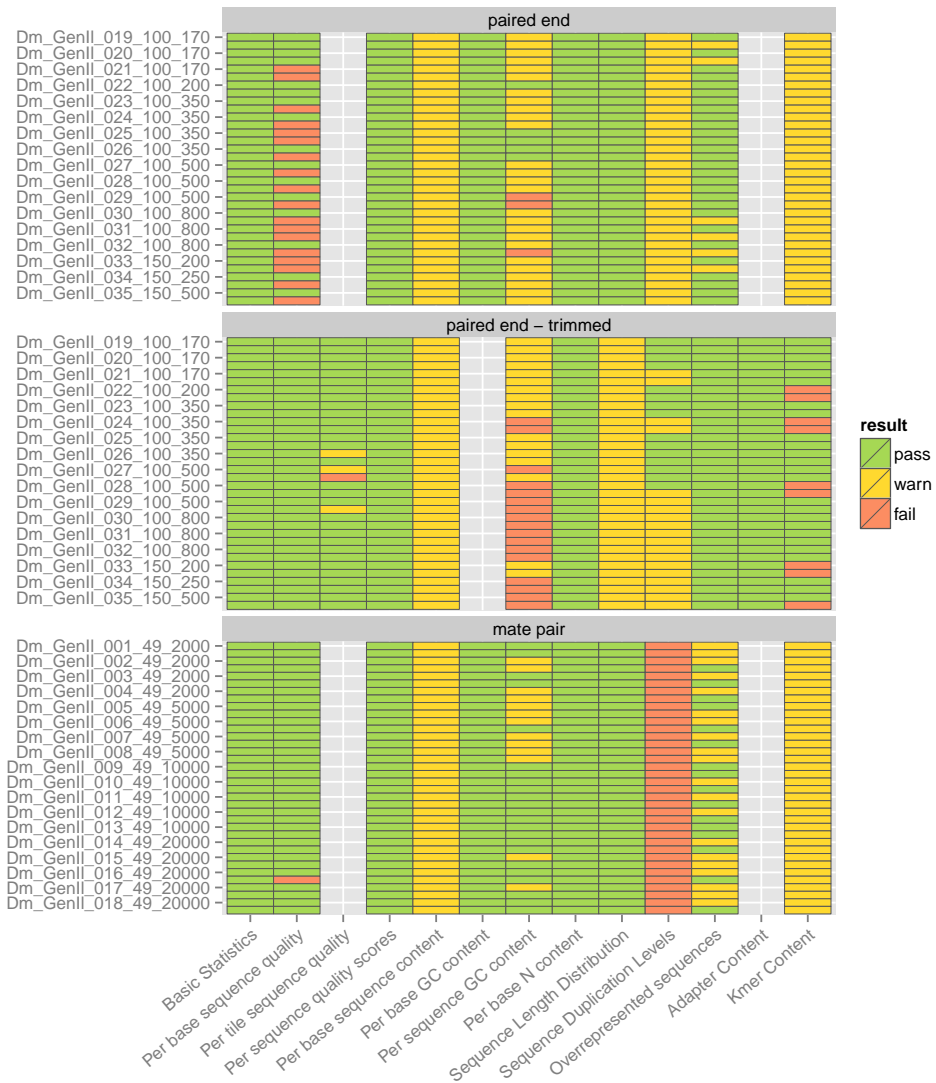


Figure 4.6.: Summary of quality control tests performed on BGI raw and trimmed paired-end and raw mate-pair libraries with FastQC. Test outcomes are colour coded (*pass*: green, *warn*: yellow, *fail*: red), empty columns indicate missing data due to differences between versions of FastQC utilized for raw and trimmed data.

4.2. Assessment of the BGI read data shows technical bias and high level of heterogeneity

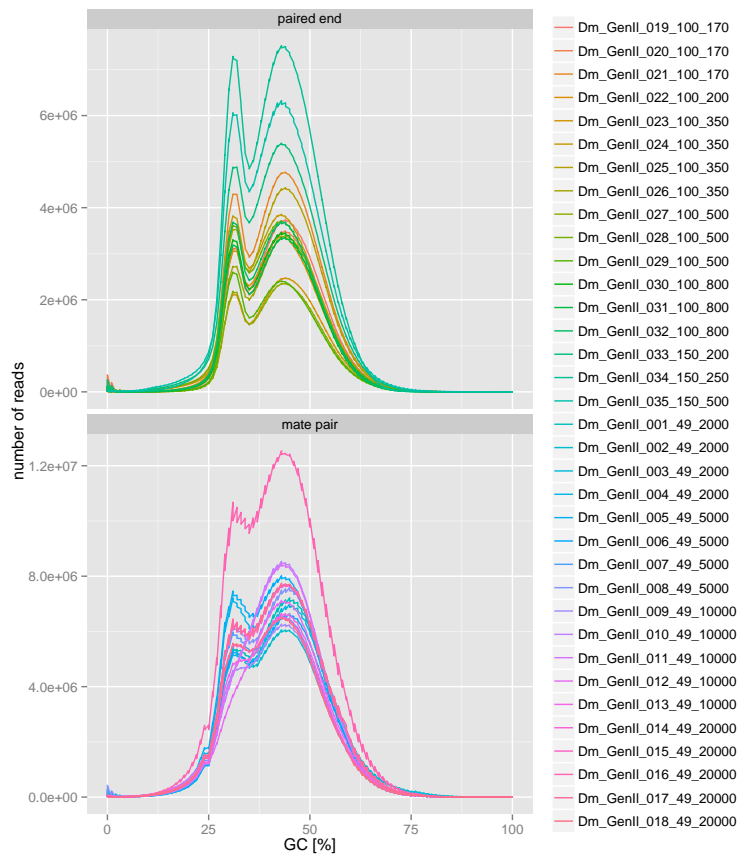


Figure 4.7.: Per sequence GC-content for different BGI paired-end (top) and mate-pair libraries (bottom).

4. Assembling the genome of *D. muscipula*

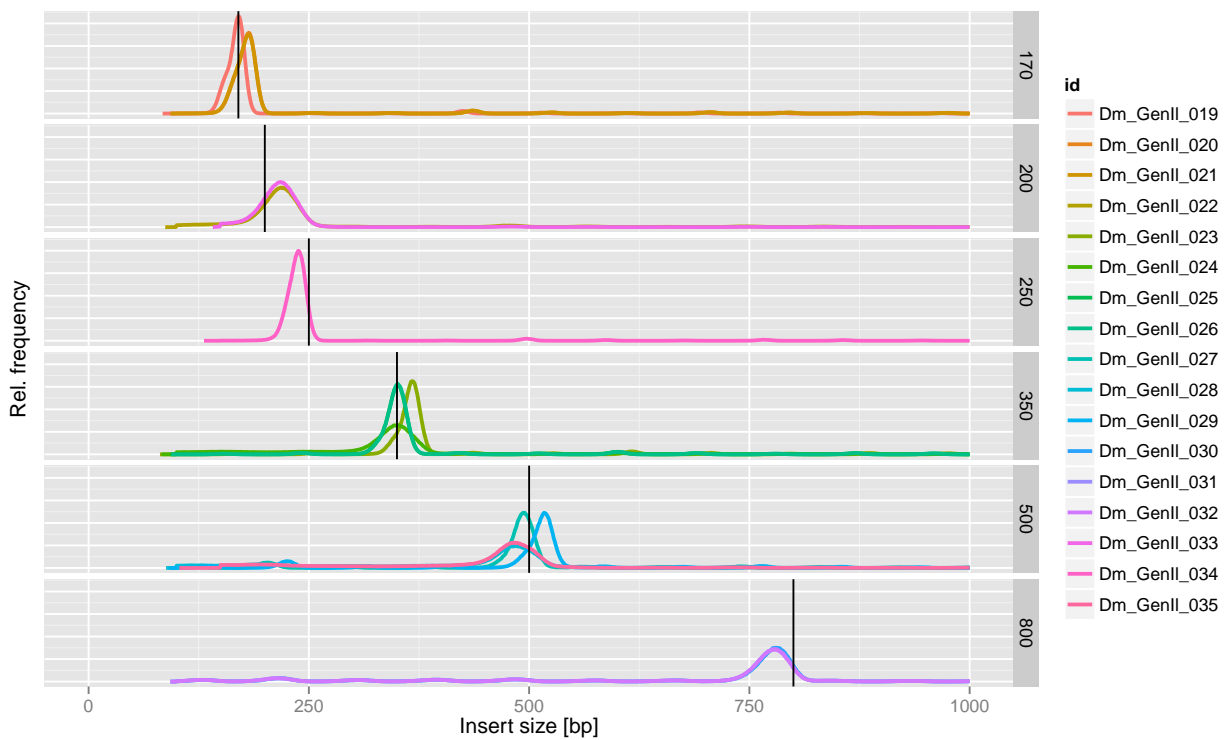


Figure 4.8.: Insert-size distribution of different BGI paired-end libraries, grouped by target insert-size (black vertical bars). Distributions were computed based on 5% of randomly sampled pairs of each library, mapped to corrected pacbio reads (see section 4.7.2 PacBio hybrid correction with proovread).

4.2. Assessment of the BGI read data shows technical bias and high level of heterogeneity

4.2.3. *k*-mer analysis, genome size estimation and repeat content

The analysis of the distribution of *k*-mer coverages in a NGS (Next Generation Sequencing, Pettersson et al., 2009) read data set allows for the assessment of actual sequencing depth, potential biases and heterozygosity as well as an estimation of genome size and repeat content. Figure 4.9 shows an Anscombe transformed histogram (red bars) of the coverage-frequency distribution of 19-mers in the full set of BGI paired-end libraries (See section 8.2 for details on and significance of Anscombe transformation.). The maximum displayed coverage is $1 \times 10^5 x$. The Maximum frequency was set 7×10^7 . The majority of *k*-mers is concentrated at coverages below $250x$ with a prominent peak at about $100x$. This peak corresponds to the unique portion of the sequenced diploid genome and indicates the sequencing depth of the template DNA. *k*-mers with very low coverages represent contaminations, sequencing errors and reads derived from low copy templates, such as rare alleles in the pooled sample. *k*-mers of high coverages derive from repetitive regions. In addition to the primary peak at $100x$, a second small peak at $200x$ can be identified in the histogram. This peak comprises duplicated genomic regions, which could include large segmental duplications as well as duplicated genes with no or little differences. At higher coverages, no other population of reads with a an accumulation of *k*-mers at a specific range can be identified.

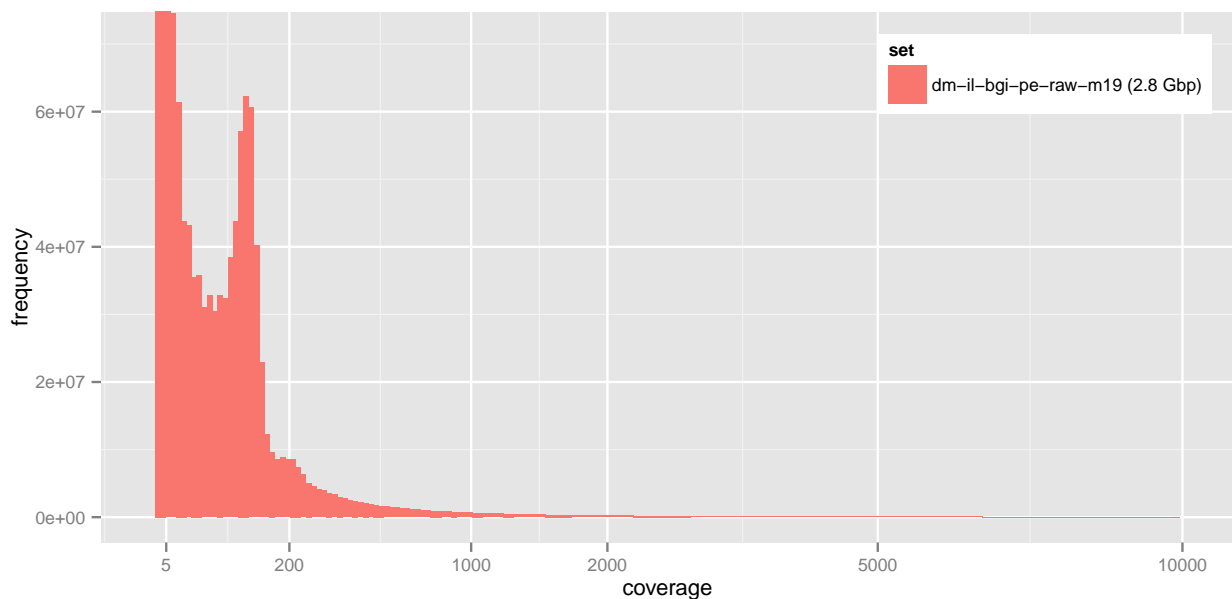


Figure 4.9.: Frequency distribution of 19-mers in all BGI paired-end reads plotted as Anscombe¹ transformed histogram with maximum displayed coverage of $1 \times 10^4 x$ and maximum displayed frequency of 7×10^7 . The estimated genome size is 2.8 .

Figure 4.10 shows the *k*-mer distribution of BGI paired-end libraries without transfor-

¹See section 8.2 for details on and significance of Anscombe transformation.

4. Assembling the genome of *D. muscipula*

mation and only for coverages up to $300x$. The distribution exhibits a primary peak at $105x$ representing homozygous k -mers with an estimated length of 240 Mbp. A distinct peak for heterozygous k -mers at half the coverage of the homozygous peak cannot be identified, yet generally rather high frequencies for k -mers with coverages of $30x$ to $80x$ are observed. A second, smaller peak, corresponding to duplicated regions, is observed at $196x$ comprising 37 Mbp. The expected total genome size based on the all k -mers with coverages $\geq 10x$ and a coverage standard of $105x$ for unique sequences is 2.8 Gbp.

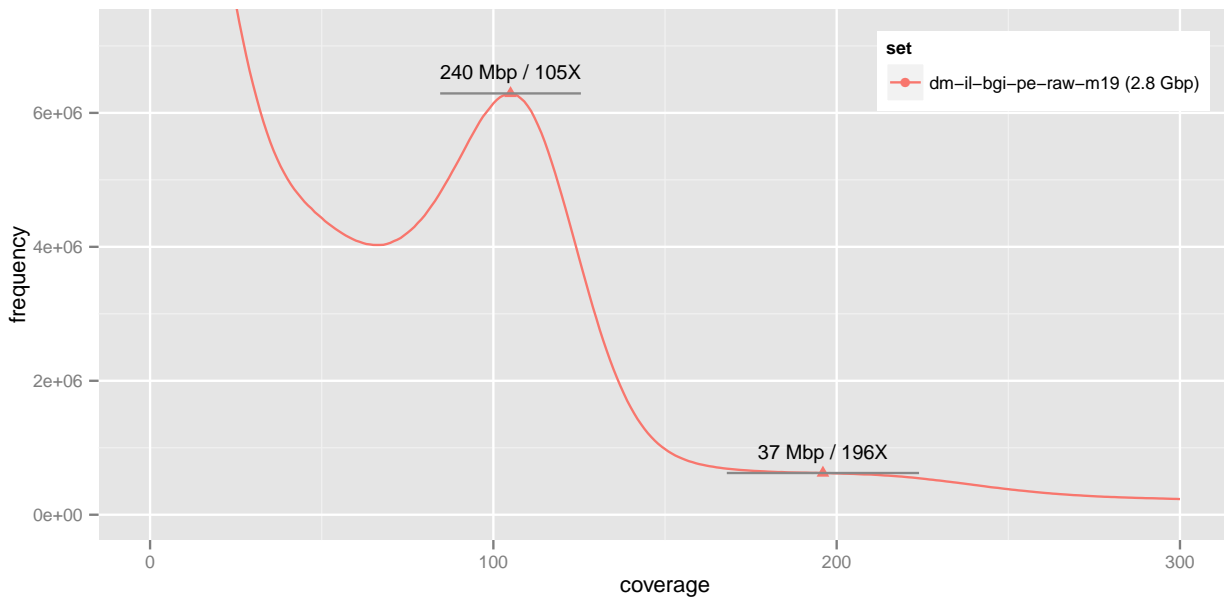


Figure 4.10.: Frequency distribution of 19-mers in all BGI paired-end reads with maximum displayed coverage of $300x$ and maximum displayed frequency of 7×10^7 . Triangles indicate peaks detected by the peak-calling algorithm, horizontal bars indicate the range used for peak size computation. The estimated genome size is 2.8 Gbp.

Based on this k -mer distribution the overall coverage of the *D. muscipula* genome in the pooled set of all raw BGI Illumina libraries is slightly more than $100x$. The large amount of k -mers at coverages between $2x$ to $10x$ indicate a high level of low copy templates, either due to high levels of contaminations or as a consequence of the pooling of a highly heterogenous set of plants. On top, high frequency levels at coverages around $50x$ indicate a high level of heterozygosity in individual plants. Accurate estimates on heterozygosity rate, however, cannot be made, because the haploid peak is obscured by the overlap with both, the unusual wide low coverage peak as well as wide main peak.

The amount of regions in the genome with unique character can be estimated from all k -mers in the range of $10\ 300$ – it roughly comprises 500 Mbp. Consequently regions of repetitive character add up to at least 2 Gbp ($>70\%$) of the flytrap genome.

4.2. Assessment of the BGI read data shows technical bias and high level of heterogeneity

4.2.4. GC bias in BGI sequencing data

The chemical difference in the interaction of A-T and G-C DNA base pairs (3 versus 2 hydrophobic bonds) causes a bias in polymerase-dependent ligation and amplification steps as well as the actual synthesis process monitored during sequencing. In general, Illumina HiSeq sequencing is known to work less efficiently on higher (>70%) as well as low GC levels (<30%) (Quail et al., 2012). The severity of the bias depends on sequencing chemicals and other experimental parameters and can vary between different samples.

Figure 4.11 shows a heatmap of k -mer coverages with respect to GC content of the individual k -mers. The 19 possible discrete GC counts for 19-mers are given on the y-axis, coverage is given on the x-axis. Relative k -mer counts are encoded by color, ranging from low (blue) to high (red) coverage. Black dots indicate the main peaks for each GC count level.

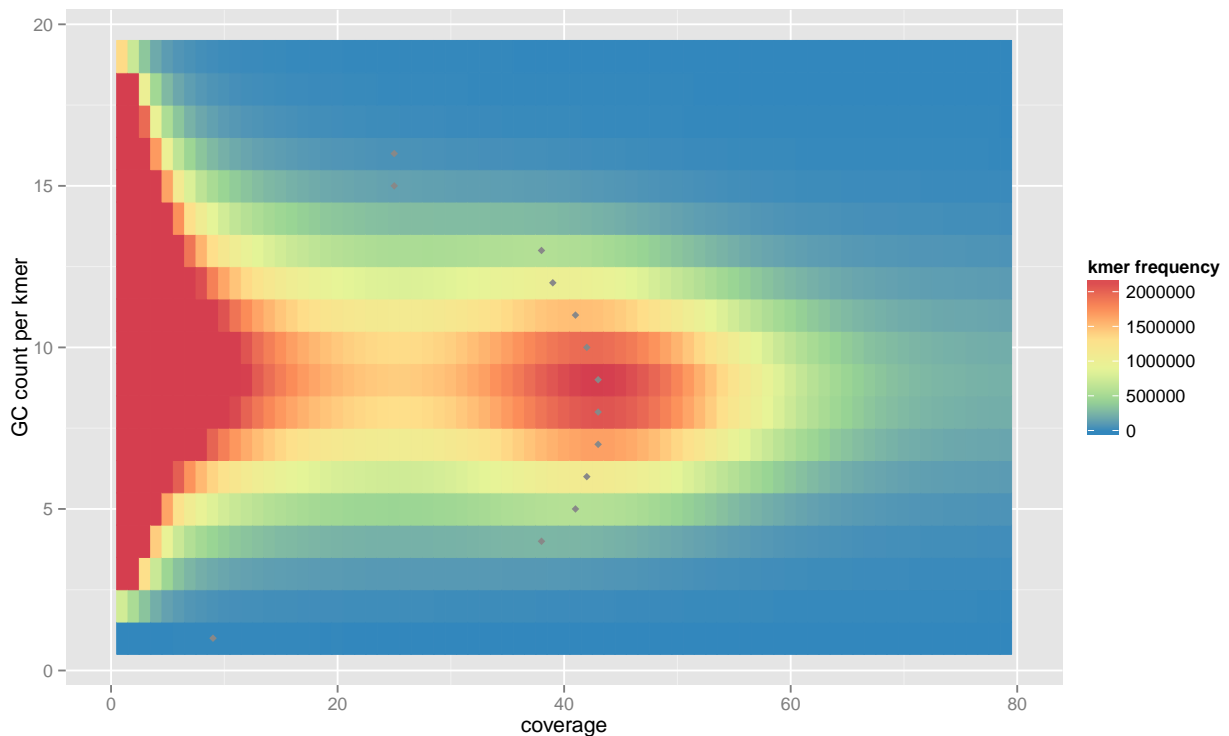


Figure 4.11.: Heatmap of 19-mer frequencies in BGI overlapping libraries with respect to coverage and GC-content. Frequencies are encoded by color ranging from blue (low) to red (high). Grey diamonds indicate individual peaks for each GC count as called by `kmer-plot gcmx`.

For the GC count range of 4 to 13 (21%-68%), the corresponding peaks range from coverages $37x$ to $44x$. Without bias, one would expect all peaks to line up on a vertical line. For the BGI Illumina read set, medium GC levels (37%-47%) exhibit maximum

4. Assembling the genome of *D. muscipula*

peak coverages, while higher /lower GC levels show decreasing coverage levels. The relative shift of the minimum compared to the maximum is at about 15%. This GC-correlated shift in coverage is one of the contributing factors causing the broad left-hand side flank of the main peak in the 2D- k -mer distribution, which blurs the distinction of haploid and diploid k -mer populations.

This GC-related bias can also be observed when comparing per base quality-scores of different trimmed read data sets. The Phred quality scores analyzed during trimming represent the level of confidence for the base call made by the sequencer. The decreased accuracy of the Illumina sequencing technology in low /high GC regions of reads is directly reflected in the reported quality scores. During trimming, tails of reads with quality scores below a certain threshold are removed. ?? shows the change in relative GC content (y-axis) in trimmed data set with respect to the number of accepted maximum of trimmed bases per read.

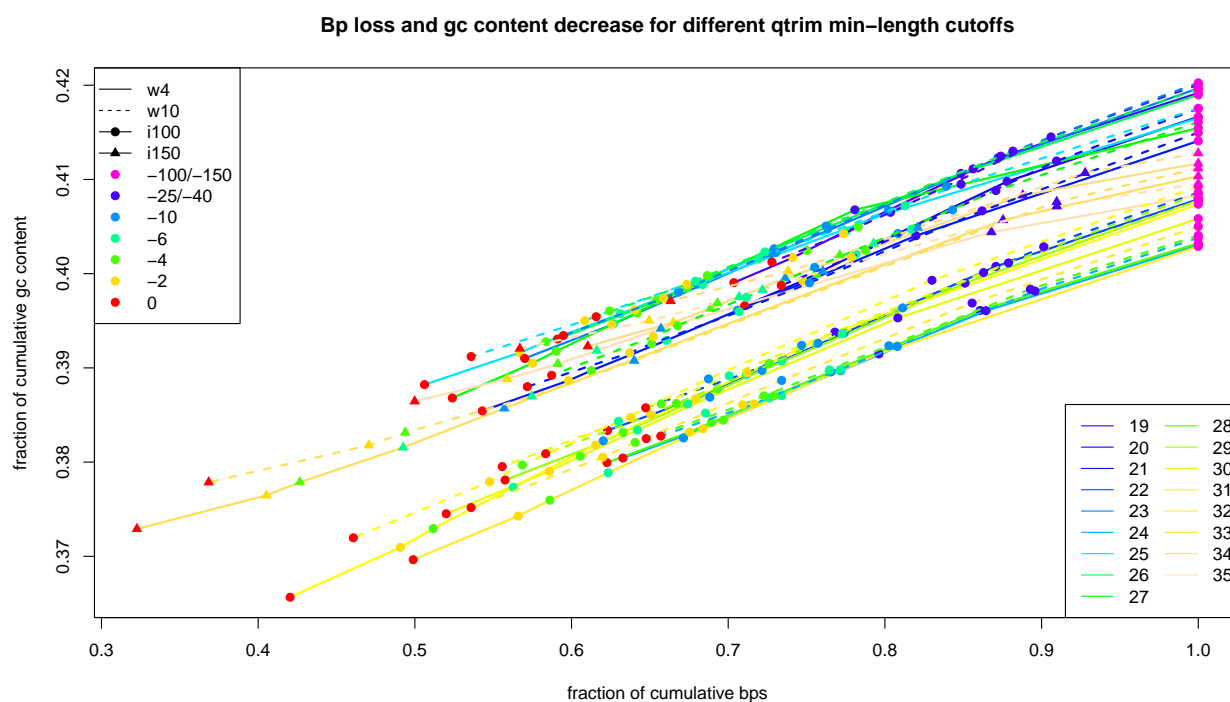


Figure 4.12.: GC-content of BGI paired-end libraries as function of varying library size caused by different minimum read length cutoffs during trimming. Libraries *Dm_GenI1_[019-035]* are indicated by line color, line type encodes window sizes 4 (continuous) and 10 (dashed), symbol shape represents different original read lengths of 100 bp (bullet) and 150 bp (triangle) and symbol color encodes maximum number of allowed bases missing after trimming, ranging from 0 bp to 100 bp or 150 bp, respectively.

GC-rich regions are particularly overrepresented in trimmed read regions resulting in an overall drop of GC level in trimmed data sets. The bias is linearly correlated with the strictness of trimming. If reads of arbitrary length are allowed in the trimmed read

4.2. Assessment of the BGI read data shows technical bias and high level of heterogeneity

set, the GC content ranges between 40% and 42%. If only full-length reads are allowed in the trimmed set, GC content for all libraries drops by 2% to 3.5%. Therefore, in trimmed data sets, the shift in coverage for GC-rich regions is even higher and the overall coverage distribution of the set is skewed even more.

In summary, the BGI data suffers from a combination of technical and biological problems resulting in non-optimal quality of libraries, which render the reconstruction of a high quality assembly infeasible. The only viable option to overcome the problems inherent to the data, is the sequencing of new libraries with optimized protocols and an improved strategy.

4.3. Various SOAP and ALLPATHS assemblies shed light on specific challenges

4.3.1. SOAP assemblies

To put the overall quality of the BGI assembly into context and as a starting point for further improving assembly quality, in cooperation with Felix Bemm, I performed and analyzed assemblies built from the BGI read data with SOAP on our in-house computing infrastructure.

Initial trials with straight-forward application and mostly default parameters generated assemblies of very low quality and contiguity. The primary conclusion was that given the particular genomic structure and the variety of shortcomings in the design and execution of the sequencing, generation of a high quality draft assembly requires extensive optimization of the default procedure. For the sake of brevity, detailed descriptions of the results of these initial runs were omitted here.

Metrics	Dm-gen-so-5.1
Contigs ≥ 0 bp	17,834,956
Contigs ≥ 1 kbp	349,327
Length ≥ 0 [bp]	4779,791,269
Length ≥ 1 k[bp]	546,533,018
Longest [bp]	30,558
N50 [bp]	260
GC [%]	42.4
N [%]	0.0

Table 4.5.: Metrics of assembly Dm-gen-so-5. See table 4.1 for detailed description of metrics.

The first noteworthy assembly (Dm-gen-so-5) was obtained with SOAP and a strategy similar to the one presumably used by the BGI. The assembly was constructed with a k -mer size of 127 from 446×10^6 single-end reads obtained through merging of all available overlapping paired-end libraries Dm_GenI1_[019-021,033,034].

The resulting contig assembly (table 4.5) comprised 17.8×10^6 contigs with an minimum length of 128 bp and 350×10^3 longer than or equal to 1 kbp with a total length of 4.8 Gbp and 550 Mbp, respectively. The N50 for the entire set is 260 bp, the longest contig has 30,558 bp. A scaffold assembly for this data set is not available as all reads were single-end, and hence could not be used for scaffolding. Stand-alone scaffolding with additional

libraries was not performed at the time.

With more than 90% of contigs shorter than 1 kbp, the assembly is highly fragmented. However, in contrast to assemblies previously computed with smaller kmer sizes, which always were much smaller size than the Dm-gen-so-1 assembly, the assembly Dm-gen-so-5 surpasses the BGI assembly in total size and number of contigs with a minimum length of 128 bp. The number and total length of contigs ≥ 1 kbp is slightly lower, yet of comparable magnitude.

The observed trend of increasing assembly size with increased kmer size fits well with previously described characteristics of the *D. muscipula* genome and the sequenced samples in particular. Especially the high level of heterogeneity in combination with a suspected high level of heterozygosity can generate this effect. Heterogeneity in general introduces ambiguities in the constructed assembly graph. With increasing kmer size (and

4.3. Various SOAP and ALLPATHS assemblies shed light on specific challenges

given sufficient coverage), initially intertwined bubbly paths, which cannot be resolved and, hence, are removed from the assembly, start to separate into longer independent stretches, which eventually will be reported as individual short contigs.

4.3.2. ALLPATHS assemblies from low coverage data sets

While SOAP is capable of constructing assemblies of giga-byte sized genomes with reasonable computational demands, other assembly programs apt for the task were developed. To date, ALLPATHS is considered one of the best De Bruijn graph based assemblers available for large eukaryotic genomes (Bradnam et al., 2013). It was reported to outperform assemblers like SOAP and ABySS in terms of contiguity and accuracy, particularly on complex data sets. Application to the *D. muscipula* read data thus has the potential to generate assemblies of improved quality in comparison to the assembly constructed by the BGI with SOAP.

The main disadvantage of ALLPATHS is its comparatively large demand on computational resources and in particular memory. Initial assembly trial runs utilizing the full BGI read data failed on a machine with 500 GB RAM (*r5n01*) due to insufficient memory.

The first successful ALLPATHS assembly (Dm-gen-ap-3) was obtained from a largely reduced read set comprising only overlapping paired-end libraries with an insert-size of 170 bp in combination with the full set of mate pair data. Theoretical coverage based on the total size of the supplied libraries was $19.4x$. k -mer analysis by ALLPATHS put coverage at $15x$, estimated genome size at 1.7 Gbp with a 70 %

kmer analysis	Dm-gen-ap-3	Dm-gen-ap-4
Genome size [bp]	1669,924,270	2549,272,689
\perp CN ≤ 1	499,304,323	605,928,211
Repetitive [%]	70.1	76.2
Coverage [x]	15	29
SNP rate	1/254	1/274

Table 4.6.: ALLPATHS kmer analysis of Dm-gen-ap-3

proportion of repetitive sequences, and a SNP rate of 1 in 254 bp (table 4.6). The assembly took 100 h and required 431 GB of peak memory. The obtained assembly comprised 75×10^3 scaffolds of at least 1 kbp in length and with a total size of 353 Mbp (table 4.7).

A second assembly (Dm-gen-ap-4) was obtained from a 25 % subsample of all BGI paired-end libraries. Theoretical coverage was $32x$. ALLPATHS estimated a coverage of $29x$, a genome size of 2.5 Gbp with 76 % repetitive content and an average rate of 1 SNP per 274 bp (Tab. 4.6). The final scaffold assembly comprised 67k contigs longer than 1 kbp and had a total size of 419 Mbp (Tab. 4.7).

With less than 350 Mbp and 420 Mbp of total scaffold length, both assemblies are substantially smaller than the expected genome size of 2.8 Gbp as well as the respective estimated genome sizes. One obvious explanation is the low coverage of the utilized read sets that directly resulted in increased number of insufficiently supported or entirely missing connections in the underlying assembly graph.

However, one key setting of ALLPATHS, especially in comparison to SOAP assemblies,

4. Assembling the genome of *D. muscipula*

has to be considered. **ALLPATHS** applies a minimum length cutoff of 1 kbp for contigs; minor fractions of smaller contigs observed in the final assembly result from the application of the cutoff prior to refinement and scaffolding. In contrast, the minimum contig length for **SOAP** assemblies is determined by the utilized k -mer size and computes to k -mer size + 1. Given the large number of short contigs in the **SOAP** assemblies, it is reasonable to assume that also for the **ALLPATHS** assembly, initially a large fraction of short contigs was constructed. Yet, due to the minimum length cutoff these contigs were reported in the final assembly. The fact that contigs are also already removed prior to the scaffolding step also causes the substantial difference in size of the scaffold assemblies with minimum sequence length of 1 kbp. While **SOAP** scaffolds a large portion of <1 kbp contigs into >1 kbp scaffolds and hence greatly increases the proportion of larger scaffolds, **ALLPATHS** lacks this vast pool of short contigs and only can scaffold together the limited number of contigs already larger than 1 kbp.

Metrics	Dm-gen-ap-3.1	Dm-gen-ap-3.2
Contigs \geq 0 bp	130,532	79,444
Contigs \geq 1 kbp	121,946	75,237
Length \geq 0 [bp]	319,449,775	356,891,523
Length \geq 1k[bp]	311,310,950	352,864,087
Longest [bp]	44,489	112,772
N50 [bp]	3299	7640
GC [%]	42.2	42.2
N [%]	0.0	10.5
Metrics	Dm-gen-ap-4.1	Dm-gen-ap-4.2
Contigs	131,259	72,656
\geq 1 kbp	121,899	67,321
Length [bp]	323,302,065	424,312,843
\geq 1 kbp	314,411,581	419,215,516
Longest [bp]	55,979	157,218
N50 [bp]	3344	12,445
GC [%]	42.4	42.4
N [%]	0.0	23.8

Table 4.7.: Metrics of assemblies **Dm-gen-ap-3** and **Dm-gen-ap-4**. See table 4.1 for detailed description of metrics.

for subsets of full-length covered (red), full-length covered, but split across multiple contigs (green) and partially covered transcripts. Of a total number of $>4.2 \times 10^4$ transcripts, 3.1×10^4 are covered by **Dm-gen-so-1**. **Dm-gen-ap-3** and **Dm-gen-ap-4** cover $\sim 2.6 \times 10^4$ and $\sim 2.2 \times 10^4$, respectively. The same trend can also be observed for the individual subsets. The largest amount of covered transcripts in each category are observed for the **SOAP** assembly, slightly lower numbers for full-length hits for **Dm-gen-ap-4** and

Due to these conceptual differences between the two assemblers, the only metric that can be compared directly is the number and size of contigs larger than 1 kbp. Here, the **BGI** assembly has a total size of 780 Mbp, while both **ALLPATHS** assemblies have about 310 Mbp. The analysis of the coverage landscape of the **BGI** assembly has revealed a large overuse for the majority of sequences, i.e. the assembly comprises redundant information and its true size is more likely to be somewhere close to 520 Mbp. Compared to this value, **Dm-gen-ap-3** and **Dm-gen-ap-4** still fall short by 200 Mbp.

4.3.3. Transcriptome coverage and comparison of **SOAP** and **ALLPATHS** assemblies

Figure 4.13 shows the results of the comparative analysis of transcriptome coverage by the **BGI** and **ALLPATHS** assemblies. The stacked histogram bars represent the amount of transcripts that could be aligned to an assembly and color encode

4.3. Various SOAP and ALLPATHS assemblies shed light on specific challenges

about 80 % for Dm-gen-ap-3. For both ALLPATHS assemblies, about 3/4 of partial hits compared the BGI assembly have mappings.

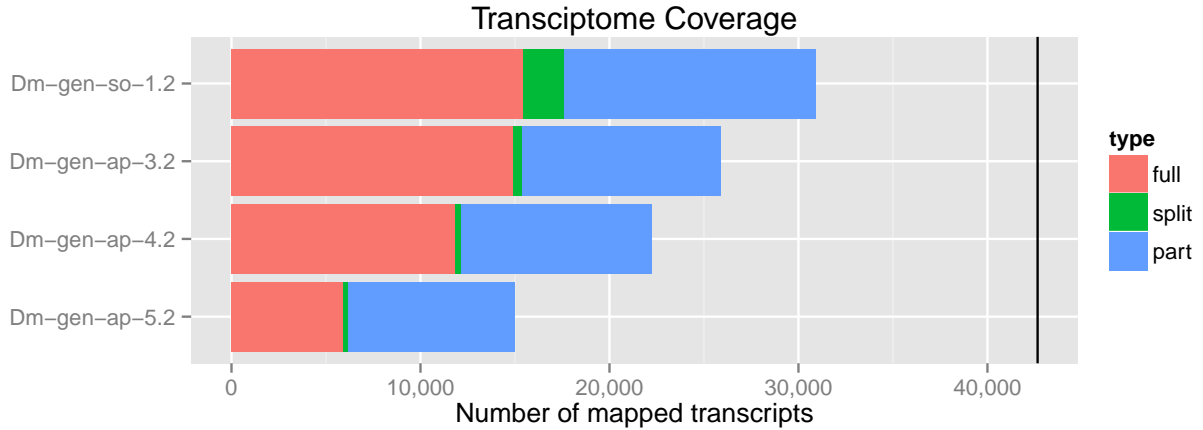


Figure 4.13.: Transcriptome content of the ALLPATHS assemblies Dm-gen-ap-3.2, Dm-gen-ap-4.2 and Dm-gen-ap-5.2 including comparison to Dm-gen-so-1. Bars of the stacked histogram represent the amount of full-length (red), full-length but split across reference sequences (green) and partially (blue) mapped longest isoforms with maximum 42,653 (black line) derived from transcriptome assembly DM_tra_qt1.03.

Therefore, the conclusion is that although the SOAP metrics are overestimates, the smaller ALLPATHS assemblies are incomplete to a larger degree in plain size as well as on the level of coding potential and hence are inferior to the BGI assembly.

4.3.4. ALLPATHS assembly with a high coverage data set

To explore the impact of not having to use low coverage data sets due to hardware restraints, we procured computing time for an additional assembly run on a 2 TB RAM machine at the LRZ. After configuration of the machine, data transfer and installation of software, read correction and assembly of the full data set were performed under my supervision by Markus Ankenbrand (Master student). The total runtime of the assembly (Dm-gen-ap-5) was 473 h, peak memory consumption was 1.98 TB. The resulting assembly (Tab. 4.8) comprised 6.8×10^4 scaffolds with at least 1 kbp in length and had a total size of 115 Mbp.

Metrics	Dm-gen-ap-5.1	Dm-gen-ap-5.2
Contigs ≥ 0 bp	82,977	81,326
Contigs ≥ 1 kbp	69,319	67,968
Length ≥ 0 [bp]	125,963,777	128,132,705
Length ≥ 1 k[bp]	112,970,769	115,421,489
Longest [bp]	30,348	51,692
N50 [bp]	1454	1495
GC [%]	43.0	43.0
N [%]	0.0	1.7

Table 4.8.: Metrics of assembly Dm-gen-ap-5. See table 4.1 for detailed description of metrics.

At a first glance, the very low size of the assembly and its high level of incompleteness, as corroborated by transcriptome coverage (Fig. 4.13), comes as a surprise, given that

4. Assembling the genome of *D. muscipula*

the assembly was constructed from the data set with a theoretical coverage of $52x$ – the maximum coverage in available in overlapping libraries. On close inspection, however, this result matches a trend already indicated by the two previous **ALLPATHS** assemblies. With an increase in the coverage of overlapping reads (**Dm-gen-ap-3**: $15x$, **Dm-gen-ap-3**: $29x$, **Dm-gen-ap-3**: $52x$), the completeness of the obtained assembly decreases. A potential explanation is the high heterogeneity of the different sequenced samples and libraries. Rather than increasing overall support of the underlying assembly graph, additional data increase the level of noise and introduce additional paths and connections and thus reduce overall length of well-supported unambiguously resolvable regions.

4.4. Redundancy reduction through digital normalization boosts assembly computation and quality

Given the highly repetitive nature of the flytrap genome in combination with the fact that repetitive reads contribute overproportionally to computational demands, digital normalization is a particular promising approach in order to target repeat-related assembly issues. Digital normalization reduces redundancy and hence size of Illumina read sets. Using k -mer coverage abundant reads are identified and only a fraction up to a specified target coverage is kept in the sample. In the filtered set coverage of unique and low coverage region remains mostly unaffected, while excessive coverage of repetitive regions is decreased.

Digital normalization is prone to accumulate sequencing errors in the data set, as errors generate low coverage k -mers, which are not recognized as potentially redundant regions and hence always make it into the filtered set. Initially, I experimented with normalized raw read data for assemblies (data not shown), but based on the above observation, decided to run error correction on the reads prior to normalization in order to mitigate the effect. This approach, however, required a full corrected set of short reads, which was only available after the generation of ALLPATHS assembly `Dm-gen-ap-5` on a machine with 2 TB RAM.

For assembly `Dm-gen-ap-6` all available overlapping libraries (`Dm_GenI1_[019-021]`, `Dm_GenI1_[033-034]`) and one library with insert-size 500 bp (`Dm_GenI1_027`) were used. Through digital normalization, the read set was reduced to 43.6% of its original size. k -mer spectrum analysis by ALLPATHS (table 4.9) shows that peak coverage was retained at $45x$, as compared to the expected $24x$ for a non-normalized set of the same size. Estimated genome size, however, was reduced to 1.2 Gbp, as a result of lowered coverage of repetitive regions.

With the normalized read set, it was possible to successfully run an ALLPATHS assembly within the limits of our hardware. Peak memory consumption was 429.8 Gbp. Total runtime was reduced substantially as well, the complete run took 100.5 h.

The resulting assembly (table 4.10) comprised 16×10^4 contigs and 7.1×10^4 scaffolds with 450 kbp and 619 kbp in total length, respectively. The N50 increased from 3.9 kbp to 19.7 kbp through scaffolding. 25.6% of the scaffold assembly are made up of N filled gaps.

<i>k</i> -mer analysis	Dm-gen-ap-6
Genome size [bp]	1,251,048,130
\perp CN ≤ 1	671,203,606
Repetitive [%]	46.3
Coverage [x]	45
SNP rate	1/91

Table 4.9.: ALLPATHS k -mer analysis of `Dm-gen-ap-6`

Metrics	Dm-gen-ap-6.1	Dm-gen-ap-6.2
Contigs ≥ 0 bp	173,104	75,466
Contigs ≥ 1 kbp	157,636	71,438
Length ≥ 0 [bp]	464,029,794	623,269,190
Length ≥ 1 k [bp]	449,801,249	619,404,082
Longest [bp]	91,443	496,750
N50 [bp]	3946	19,699
GC [%]	42.2	42.2
N [%]	0.0	25.6

Table 4.10.: Metrics of assembly `Dm-gen-ap-6`. See table 4.1 for detailed description of metrics.

4. Assembling the genome of *D. muscipula*

Even after scaffolding, the assembly is much smaller than the expected 2.8 Gbp, as well as the 1.2 Gbp estimated by ALLPATHS for the normalized set. To assess the completeness of the assembly in terms of coding potential, I mapped *D. muscipula* transcripts and computed the corresponding coverage. Results are shown in fig. 4.14. More than 20×10^3 transcripts can be mapped full-length, about 1×10^3 split over different scaffolds and about 9×10^3 can be mapped partially. Compared to the BGI SOAP assembly (Dm-gen-so-1), the total number of mapped transcripts is with 30×10^3 and 31×10^3 comparable. However, out of these mappable transcripts, 25% more ($\sim 5 \times 10^3$) can be mapped to the ALLPATHS assembly in full-length. This indicates that in spite of the small size, the ALLPATHS assembly is with respect to coding regions similarly complete as Dm-gen-so-1. At the same time, the increased contiguity of the assembly allows for a much higher fraction of full-length mappings, indicating a higher fraction of full-length gene models in the Dm-gen-ap-6.2 assembly.

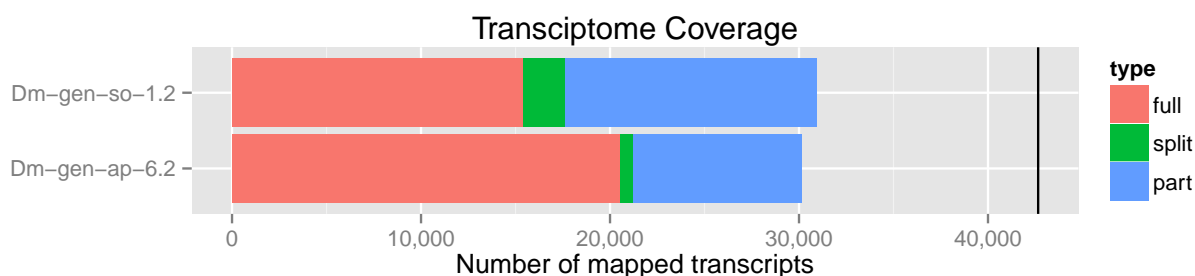


Figure 4.14: Transcriptome content of the assembly Dm-gen-ap-6 in comparison to Dm-gen-so-1. Bars of the stacked histogram represent the amount of full-length (red), full-length but split across reference sequences (green) and partially (blue) mapped longest isoforms with maximum 42,653 (black line) derived from transcriptome assembly DM_tra_qt1.03.

Figure 4.15 shows the GC-coverage landscape and cumulative contig length distribution of the Dm-gen-ap-6.2 assembly. In the left panel each circle represents a scaffold with relative GC in the x-axis and representative adjusted coverage on the y-axis; color and size correspond to its length. For better readability, only a 1% subsample of scaffolds is actually displayed in the scatterplot. The overall topology of the plot is not affected by this down-sampling procedure. The panel on the left is a stacked histogram of the unsampled cumulative scaffold length.

The majority of scaffolds in terms of number as well as total length is concentrated in a cluster between $70x$ to $90x$ coverage and between 35% and 50% GC. The trend of lower coverages at higher GC levels is consistent with the GC bias observed in the raw read data. The total length of all sequences in the cluster makes up for a total of ~ 450 Mbp. The second cluster at about half the coverage of the primary cluster and at a similar GC range comprises 50 Mbp. The two clusters correspond to the expected coverage for scaffolds present in either one or two allelic copies. Another 50 Mbp fraction of scaffolds exist at zero coverage. These scaffolds either derive from contaminations or sequences

4.4. Redundancy reduction through digital normalization boosts assembly computation and quality

that are by majority specific to the BGI read data set. Given a total assembly size of 620 Mbp, less than 70 Mbp are to be attributed to sequences predominantly consisting of collapsed repeats.



Figure 4.15.: GC-content and length distribution of assembly `Dm-gen-ap-6.2` as a function of representative adjusted k -mer coverage. See fig. 4.4 for detailed plot description.

Compared to the BGI SOAP assembly `Dm-gen-so-1`, which shows a strong bias towards relative overuse of repetitive sequences and lacks proper distinction of homozygous and heterozygous sequences, the `Dm-gen-ap-6` assembly exhibits a well structured sequence coverage landscape. The majority of bases is assembled and scaffolded into sequences of at least 10 kbp length and at the expected coverage level. Separate assembly of allelic regions due to heterozygosity affects a minor proportion of the scaffold set. The amount of collapsed, unconnected repeat sequences is negligible.

In addition to ALLPATHS, I also tested the performance of SOAP on digitally normalized data. From different runs with varying library compositions (data not shown), the best results in terms of contig assembly statistics were obtained from assembly `Dm-gen-so-4`. The assembly was constructed with a k -mer size of 127 and from a digitally normalized set of all BGI 150 bp libraries `Dm_GenI1_[033-035]`. The coverage of $48x$ of the set is comparable to the set used for the ALLPATHS assembly. Scaffolding of the obtained contigs was not performed.

4. Assembling the genome of *D. muscipula*

Metrics	Dm-gen-so-4.1
Contigs ≥ 0 bp	1177,868
Contigs ≥ 1 kbp	107,022
Length ≥ 0 [bp]	671,847,278
Length ≥ 1 k[bp]	157,492,207
Longest [bp]	24,098
N50 [bp]	615
GC [%]	41.2
N [%]	0.0

Table 4.11.: Metrics of assembly Dm-gen-so-4. See table 4.1 for detailed description of metrics.

The assembly comprises 107×10^3 contigs of a length ≥ 1 kbp with a total length of 157 Mbp (table 4.11). The N50 of contigs of all lengths is 615 bp. Compared to other SOAP assemblies as well as the ALLPATHS assemblies generated from normalized data, assembly Dm-gen-so-4 is very small and incomplete. This result supports the interpretation that obtaining large assemblies from running SOAP with large k -mer sizes can mostly be attributed to the overuse of heterogeneous repetitive regions. Through normalization and the resulting reduction in coverage, the support for individual paths of different repeat species drops below threshold and thus repetitive regions are removed from the final assembly.

4.5. Resequencing solves heterogeneity issue and mitigates technical bias

The analysis of the BGI Illumina read data and the evaluation of the derived assemblies clearly indicate that the BGI paired-end read set suffers from technical (GC-bias) and experimental problems (heterogeneity due to pooling), which are likely to contribute substantially to the poor quality of the obtained assemblies.

In addition, while at least with normalized read data, **ALLPATHS** seems to be the the assembler most suitable to generate an assembly of proper quality for *D. muscipula*, the design of the libraries generated by the BGI is less than ideal for the use with **ALLPATHS**. The initial assembly graph of **ALLPATHS** is solely based on overlapping paired-end libraries. fig. 4.16 shows the comparison of the k -mer distribution of the complete paired-end set (red line) and the overlap only set (blue line). The actual sequencing depth of all raw overlapping libraries combined is $42x$. The estimate for the main peak size changes slightly from 240 Mbp to 320 Mbp, the overall genome size estimate of 2.9 Gbp is quite similar. Due to the shift in coverage and the resulting compression of the peaks, resolution of erroneous, contaminant, homozygous and heterozygous k -mer populations is complicated further.

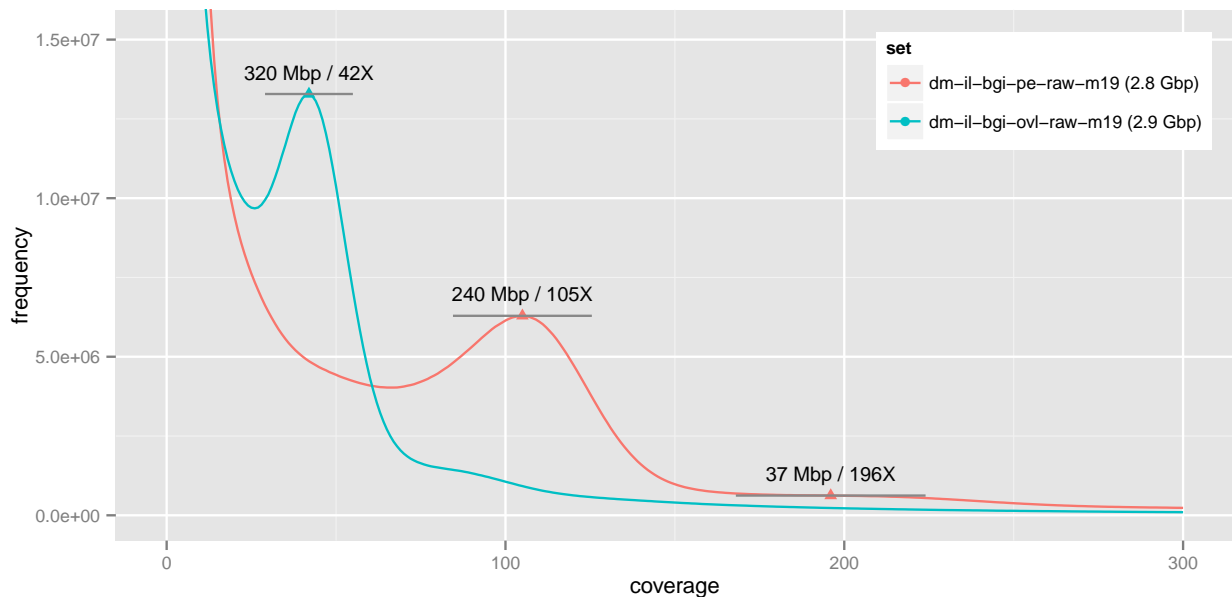


Figure 4.16.: Frequency distribution of 19-mers in BGI overlapping (blue line) and all BGI paired-end reads (red line) with maximum displayed coverage of $300x$ and maximum displayed frequency of 1.5×10^7 . Triangles indicate peaks detected by the peak-calling algorithm, horizontal bars indicate the range used for peak size computation. The estimated genome sizes are 2.9 Gbp and 2.8 Gbp for the overlapping and the full paired-end set, respectively.

4. Assembling the genome of *D. muscipula*

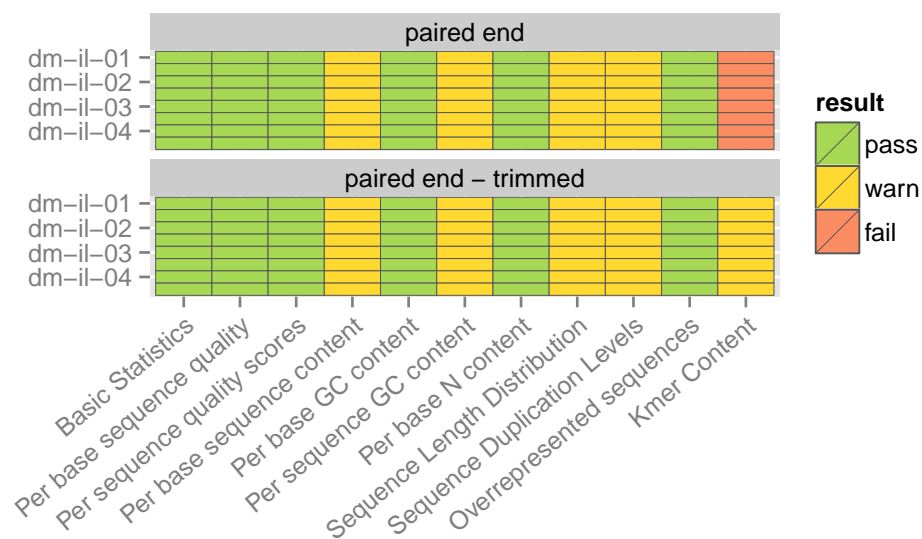
To obtain high quality results, the recommendations for ALLPATHS assemblies are $100x$ coverage in overlapping libraries of high quality, i.e. little sequencing bias, and with low heterogeneity and only little heterozygosity. The BGI library set with less than $50x$ coverage in overlapping libraries, high heterogeneity, heterozygosity and the GC-dependent coverage bias clearly contrasts these recommendations.

In order to address these non-biological issues identified for the BGI read set, four new paired-end Illumina libraries in overlapping configuration with read length 100 bp and insert-size of 180 bp were sequenced at LGC. The DNA was prepared in-house with an optimized protocol from a small batch of most likely mono-clonal plants to reduce heterogeneity. Newer sequencing chemicals with a potentially lower composition based sequencing bias were used.

After adapter clipping the four libraries (dm-il-[01-04]) comprised 1.4×10^9 read pairs with a total size of 294 Gbp and relative GC content between 41.3% and 41.6% (table 2.5).

4.5.1. Quality control of LGC libraries

The quality of the libraries obtained from LGC was assessed with FastQC. fig:dm-il-lgc-fastqc provides a schematic overview of the basic quality metrics on both, the raw and trimmed LGC read set. The results for each test are color-coded (*pass*: green, *warn*: yellow, *fail*: red). Warnings are issued for raw and trimmed reads in the categories: *Per base sequence content*, *Per base GC content*, *Sequence Length Distribution* and *Sequence Duplication Levels*. The test on *k-mer Content* failed for raw reads and produced a warning after trimming.



4.5. Resequencing solves heterogeneity issue and mitigates technical bias

The warning on *Per base sequence content* can be attributed to slight compositional bias at the 5' of the reads. The signal, however, is quite weak and was also already observed for the BGI reads.

The reason for the *Per sequence GC content* warning is a particular feature of the *D. muscipula* genome and was already observed and described for the BGI data (see section 4.2.1). The bimodal distribution in relative GC content per read derives from a subpopulation of reads with significantly lower relative GC content, which most likely are part of an abundant class of repetitive elements in the genome.

The warning on *Sequence Length Distribution* can simply be attributed to the fact that the test expects all sequences to be of identical length, while the program was run on clipped / trimmed read sets comprising already shortened reads.

Sequence Duplication Levels and *k-mer Content* warnings / failure were also already observed for the BGI set. While these results can indicate biases in library preparation or amplification, these test are only conclusive under the assumption of homogeneously distributed coverage levels as obtained from smaller, unenriched samples with high complexity. In the case of *D. muscipula* it is most likely that the warnings are triggered by the high content of repetitive sequences rather than technical artifacts.

Overall, the evaluation of the **FastQC** results on the libraries sequenced by LGC indicate properly generated libraries without obvious technical issues.

4.5.2. Insert-size distribution of LGC Illumina libraries

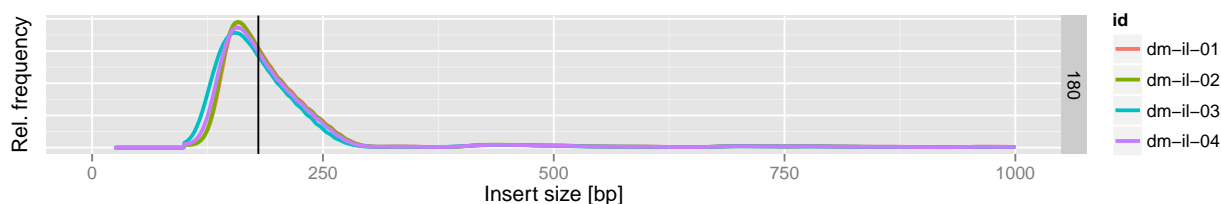


Figure 4.17.: Insert-size distribution of LGC paired-end libraries with target insert-size of 180 bp (black vertical bars). Distributions were computed based on 5% of randomly sampled pairs of each library, mapped to corrected PacBio reads (see section 4.7.2)

The libraries sequenced by LGC were designed as overlapping libraries with an insert-size of 180 bp. The actual distribution of insert-sizes obtained through mapping using corrected PacBio reads as reference are consistent among the four libraries and match the target size of 180 bp (fig. 4.17).

4. Assembling the genome of *D. muscipula*

4.5.3. *k*-mer-analysis, genome size estimation and repeat content

To assess actual sequencing depth, error rate, potential biases and heterozygosity as well as confirm the estimates on genome size and repeat content derived from the BGI read set, I performed a *k*-mer analysis of the LGC read set. Figure 4.18 shows an Anscombe transformed histogram (red bars) of the coverage-frequency distribution of 19-mers in the full LGC library set (See section 8.2 for details on and significance of Anscombe transformation.). The maximum displayed coverage is $1 \times 10^4 x$, maximum frequency was automatically set to 1×10^8 by the plot script based on the size of the primary peak. The majority of *k*-mers in the plot has coverages below $200x$. Next to the half peak at $1x$ to $5x$ coverage comprising mostly *k*-mers derived from sequencing errors, one major and two minor peaks can be distinguished. The main peak corresponds to the unique regions of the diploid genome shared between alleles (homozygous peak), the first minor peak at half the coverage of the main peak comprises haplotype specific *k*-mers (heterozygous peak). The second minor peak at twice the main peak coverage derives from duplicated genomic regions. At higher coverage, no accumulation or *k*-mers corresponding to specific subpopulation of reads can be observed.

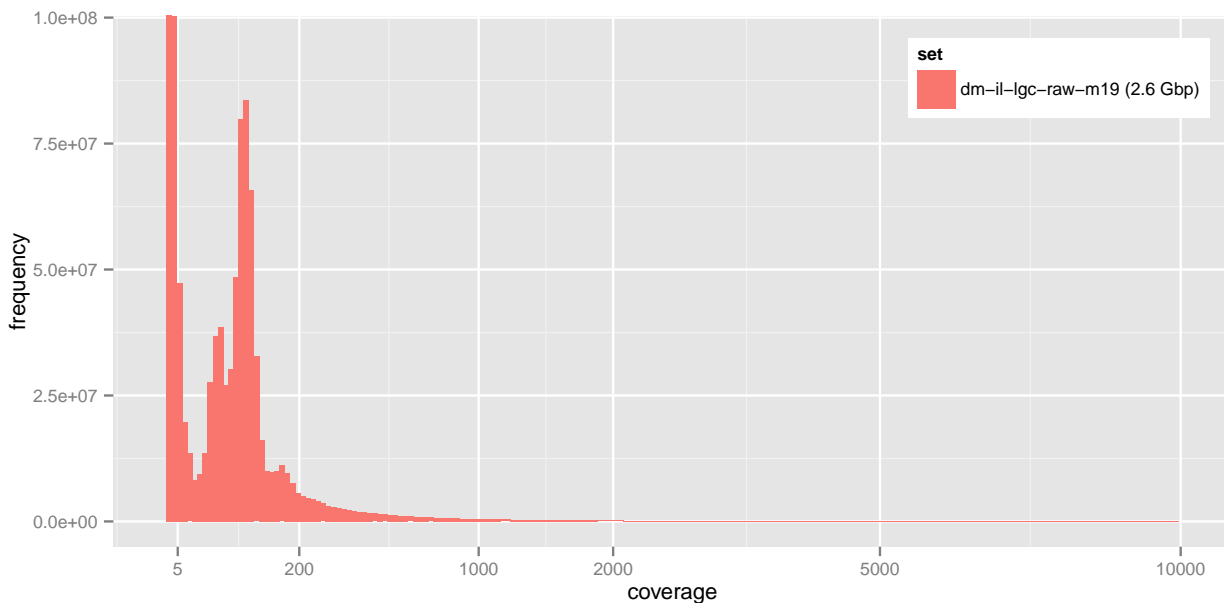


Figure 4.18.: Frequency distribution of 19-mers in LGC reads plotted as Anscombe transformed histogram with maximum displayed coverage of $10,000x$ and maximum displayed frequency of 1.0×10^8 . The estimated genome size is 2.6 Gbp.

Figure 4.19 provides a closer look at the $\geq 300x$ part of the *k*-mer distribution of the LGC set (red line) and, for comparison, the spectrum of BGI *k*-mers (blue line) in untransformed space. Peaks are indicated by triangles and labels comprising estimated size and coverage, peak width by gray horizontal bars. The estimated genome size for

4.5. Resequencing solves heterogeneity issue and mitigates technical bias

D. muscipula based on the LGC data is 2.6 Gbp. In close-up as well, the distribution exhibits three distinct peaks: A major peak at $82x$ with an estimated size of 300 Mbp comprising unique haplotype shared k -mers, a clearly distinct peak at half of the main peak coverage ($40x$) comprising haplotype specific k -mers with an estimated size of 130 Mbp and a minor peak at twice the main peak coverage ($158x$) with an estimated size of 43 Mbp.

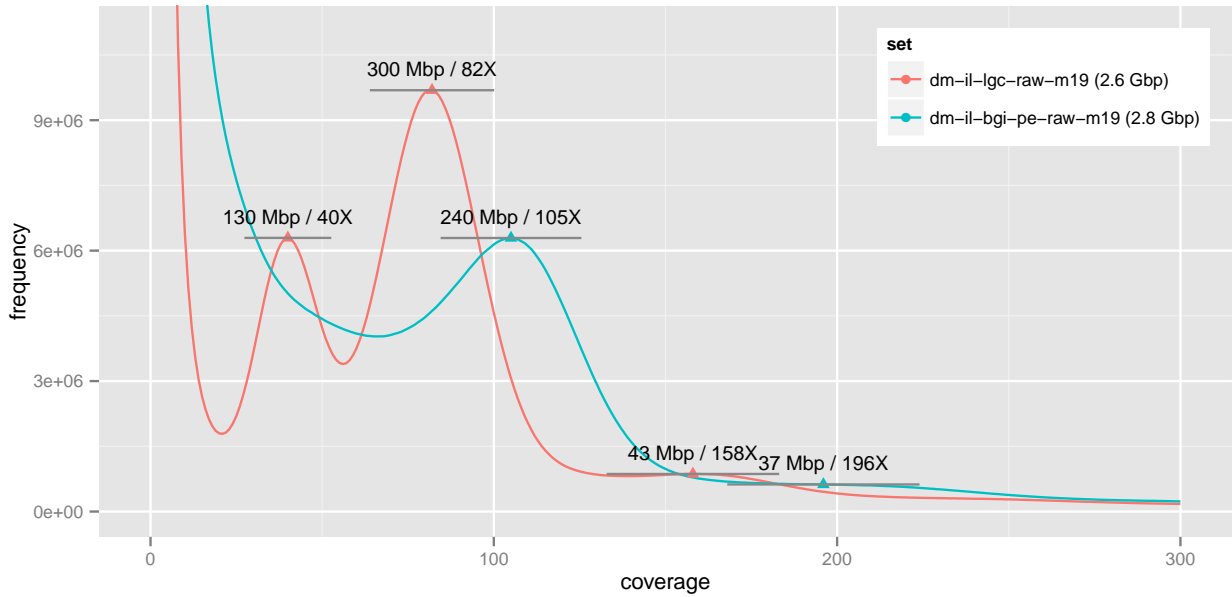


Figure 4.19.: Frequency distribution of 19-mers in LGC (blue) and BGI paired-end (red) reads with maximum displayed coverage of $300x$ and maximum displayed frequency of 1.1×10^7 . Triangles indicate peaks detected by the peak-calling algorithm, horizontal bars indicate the data range used for peak size computation. Estimated genome sizes are 2.6 Gbp and 2.8 Gbp for the LGC and BGI data set, respectively.

In comparison to the BGI spectrum with overlapping and due to biases indistinguishable peaks, the k -mer spectrum of the LGC library set exhibits distinct and sharp peaks with a Poisson-like shape for homo- and heterozygous k -mer populations. The half peak comprising erroneous k -mers is narrow and clearly separated from non-erroneous data at higher coverages. This on the one hand indicates a much lower level of technical bias than for the BGI data, on the other hand also is a result of the strongly reduced heterogeneity due to the limited pooling of individuals. The slightly reduced genome size estimate - 2.6 Gbp versus 2.8 Gbp - as a result of reduced diversity in the sequenced sample further supports this interpretation. The overall estimate of $>70\%$ of repetitive sequence content remains unchanged.

The distinct haploid peak also allows for an estimate of the SNV rate in the sequence *D. muscipula* sample. Under the simplified assumption that heterozygosity is primarily caused by randomly distributed SNVs (comprising mostly), an approximate rate of 1 SNV in 147bp can be estimated according to eq. (8.3) and eq. (8.4), given a heterozygous

4. Assembling the genome of *D. muscipula*

peak size of 130 Mbp and a non-repetitive genome size of 500 Mbp. This, however, is only a rough estimate due to two factors:

First, the model assumes that each SNV affects the maximum number of k -mers, however, if two SNVs are closer together than the utilized k -mer size, the effective number of affected k -mers is reduced, as k -mers affected by both variants still only occur once. In natural samples, clustering of polymorphisms is common. The effect is further enhanced by an overall high SNV rate, which already puts SNVs closer together by chance. Overall, this leads to an underestimate of the SNV rate.

Second, SNVs introduced in repeat copies put affected k -mers within the heterozygous peak as long as the SNV is present in a single allelic copy. The corresponding unaffected k -mers, however, due to their repetitive nature, remain at a high coverage levels and are not considered during heterozygosity rate estimation. Thus, the number of heterozygous k -mers in the non-repetitive genome is overestimated, resulting in an overestimate of the total SNV rate.

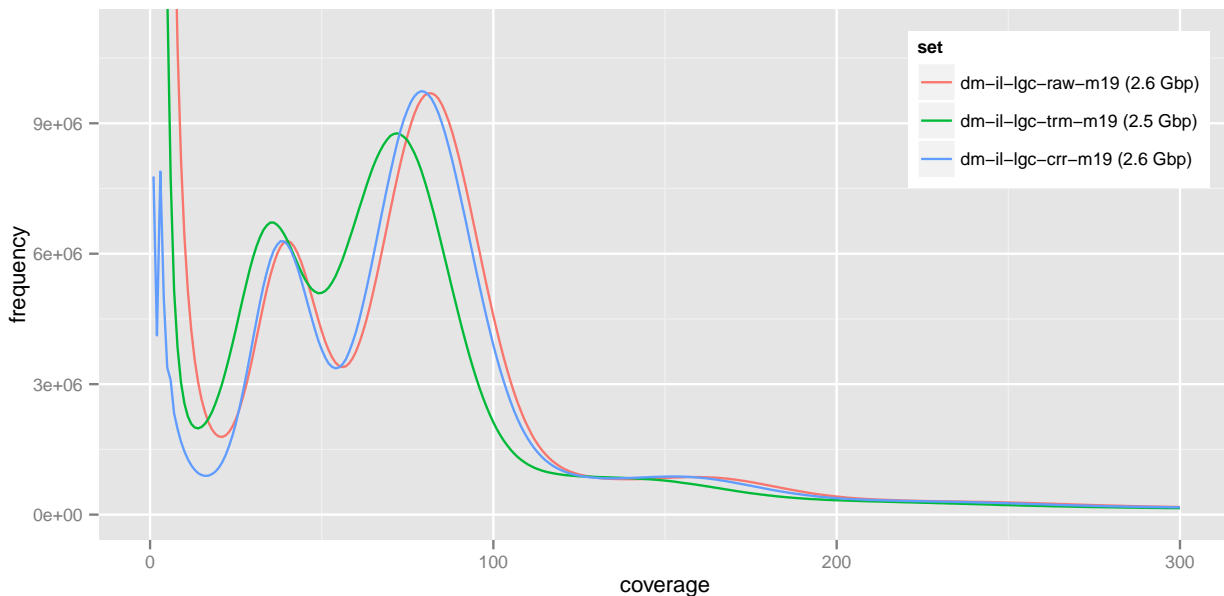


Figure 4.20.: Frequency distribution of 19-mers in raw (red), trimmed (green) and corrected (blue) LGC read sets with maximum displayed coverage of $300x$ and maximum displayed frequency of 1.1×10^7 . Estimated genome sizes are 2.6 Gbp, 2.5 Gbp and 2.6 Gbp, respectively.

Figure 4.20 comprises a comparison of k -mer spectra obtained from the LGC reads without processing (red), after trimming (green) and after ALLPATHS error correction of the raw reads (blue). Through trimming, the overall coverage of the set is reduced by about 10%. Thus, homozygous and heterozygous peak are shifted to lower coverages and brought closer together. The increased overlap between the peaks partially accounts for the less distinct valley. However, in combination with a slightly increased peak width, a

4.5. Resequencing solves heterogeneity issue and mitigates technical bias

minor systematic bias seems to be introduced most likely similar to the GC-bias observed during trimming of the BGI set, yet much less severe. Error correction of the raw read data only has a minor impact on the shape of the two main peaks, however, erroneous k -mers with coverages close to $1x$ are substantially reduced.

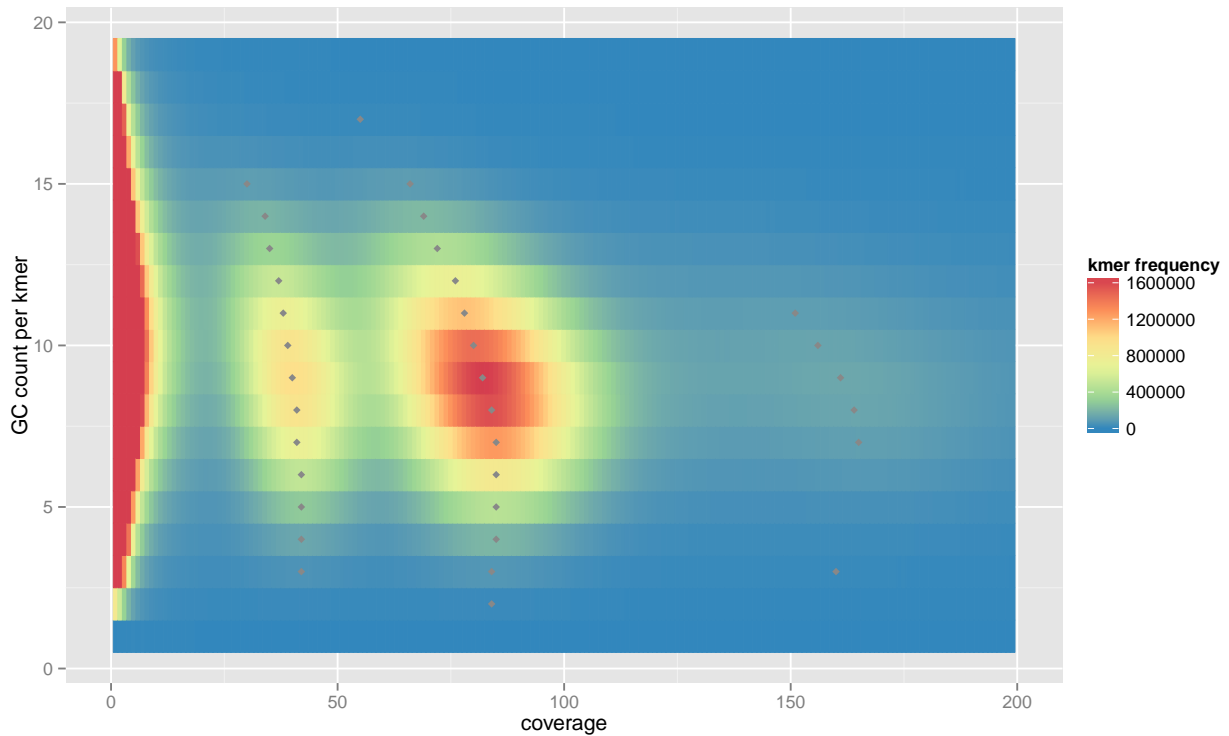


Figure 4.21.: Heatmap of 19-mer frequencies in LGC overlapping libraries with respect to coverage and GC-content. Frequencies are encoded by color ranging from blue (low) to red (high). Grey diamonds indicate individual peaks for each GC count, as called by *kmer-plot*.

To assess a potentially GC-related bias in sequencing depth, I analyzed the k -mer distribution with respect to the GC composition of the quantified k -mers. Figure 4.21 shows a plot of the k -mer coverage against the GC-count per k -mer. Frequency is color coded by a spectrum ranging from blue (low) to red (high). The population of erroneous, heterozygous and homozygous k -mers form distinct clusters. Grey dots indicate the individual peak values for each GC level. Under unbiased conditions one would expect all peaks to line up vertically. For the GC range of the main peak at 4 to 13 (21%-68%), peak coverages range from $72x$ to $85x$, which computes to a relative shift of 15%. This is equivalent to the GC-correlated shift in the BGI sequencing data. This suggests that a GC-related sequencing bias at this level is a general trend for *D. muscipula* sequencing samples on Illumina platforms, rather than an issue specific to the sequencing procedure used by the BGI.

4. Assembling the genome of *D. muscipula*

4.5.4. Digital normalization of LGC sequencing data

IID	reads	size [bp]	kept [%]
dm-il-nkh	518238768	51702495021	35.1
	518238768	51701548333	
dm-il-nbb	575343861	57404351676	39.0
	575343861	57401075176	

Table 4.12.: Metrics of LGC libraries digitally normalized with `normalize-by-median.py` (Brown and Crusoe, 2014) (`dm-il-nkh`) and `bbnorm.sh` (`dm-il-nbb`).

of 103 Gbp (35 % of the raw set, table 4.12).

In addition to `normalize-by-median.py`, digital normalization was also performed with `bbnorm.sh` of the `BBDMap` package. This program uses a more sophisticated two-pass approach, which makes it less sensitive to sequencing errors and reduces the bias introduced to the non-repetitive fraction of the reads. Using a target k -mer coverage of $140x$ (equivalent to $200x$ per-base coverage), the set was reduced 575×10^6 read pairs with a total size of 115 Gbp (39 %, table 4.12).

4.5.5. Merging of overlapping libraries

IID	merged [%]	size [bp]	[x]
dm-il-01	59.5	29525041403	9.8
dm-il-02	60.0	29994918620	10.0
dm-il-03	65.4	30207905589	10.1
dm-il-04	61.6	38110998025	12.7
dm-il-nkh	54.3	40156843701	-

Table 4.13.: Metrics of raw (`dm-il-[01-04]`) and normalized (`dm-il-nkh`) LGC overlapping libraries merged with `FLASH`.

Given the success of applying digital normalization to the BGI read set in order to facilitate the assembly within limit computational resources and at the same time to improve the overall assembly quality, digital normalization was also run on the LGC reads prior to assembly. Normalization of the four libraries combined using `normalize-by-median.py` with a threshold coverage of $200x$ produced a library of 518×10^6 read pairs and a total size

In order to obtain read fragments of maximum length, the two reads of a pair in an overlap library can be combined into a single fragment by aligning and merging them at their overlapping ends. Using `FLASH` merging was successful for 60 % to 65 % of the pairs per library in case of the raw LGC libraries, creating a set of reads with an approximate length of 180 bp and a combined coverage of $43x$ (table 4.13).

For the combined, `khmer` normalized library of the LGC set, merging was successful for 54 % of the read pairs, producing a library with a total size of 40 Gbp.

4.6. LGC reads further improve assembly contiguity and completeness

4.6.1. SOAP assemblies corroborate higher quality of LGC reads

To assess the behavior of the newly sequenced libraries in assembly, a series of SOAP assemblies was performed using strategies analogous to the ones previously applied to the BGI data.

Assembly `Dm-gen-so-9` was constructed in a way similar to `Dm-gen-so-1` and `Dm-gen-so-5`, using merged overlapping libraries and a k -mer size of 127. The resulting total assembly size was 1.2 Gbp, yet only 37×10^3 contigs were at least 1 kbp long, resulting in total length of 47 Mbp for large contigs (table 4.14). Scaffolding was not performed. The high level of fragmentation of the assembly is somewhat surprising, given that `Dm-gen-so-5` has a total length of 550 Mbp for contigs ≥ 1 kbp and was constructed from BGI data with a much higher level of heterogeneity. However, it has to be noted that while the level of heterogeneity was greatly reduced, the read set is still highly heterozygous and the underlying assembly graph remains highly ambiguous. In addition, the average length of merged fragments from the LGC set is 180 bp, whereas due to two libraries with 150 bp reads and insert-sizes of 200 bp and 250 bp, the average read length for the BGI set is about 220 bp. This roughly 20% difference in read length has disproportionate impact on the coverage of the underlying k -mer graph. A read of 180 bp comprises 54 127-mer, while a read of 220 bp comprises 94 127-mers. With both sets of similar size and estimated coverages close to $40x$, the overall support of the BGI data derived k -mer graph is substantially higher. The poor contiguity of the LGC based assembly can be attributed to regions of low k -mer support trimmed from the graph.

Assemblies `Dm-gen-so-10` to `Dm-gen-so-12` were constructed from ALLPATHS corrected LGC libraries with k -mer sizes 67, 77 and 87, respectively. Obtained assembly metrics are shown in table 4.14. `Dm-gen-so-10` consists of 46×10^6 contigs with a total size of 5 Gbp. When looking at contigs of at least 1 kbp, these numbers decrease substantially to 270×10^3 contigs and a total size of 349 Mbp. A similar trend can be observed for assembly size and contig numbers of the two assemblies, with raw counts of 35×10^6

Metrics	Dm-gen-so-9.1	Dm-gen-so-10.1
Contigs ≥ 0 bp	4678,921	46,280,104
Contigs ≥ 1 kbp	36,686	136,524
Length ≥ 0 [bp]	1210,730,901	5007,245,923
Length ≥ 1 k[bp]	46,632,545	269,553,853
Longest [bp]	12,606	31,576
N50 [bp]	388	101
GC [%]	40.0	43.1
N [%]	0.0	0.0
Metrics	Dm-gen-so-11.1	Dm-gen-so-12.1
Contigs ≥ 0 bp	35,100,734	15,706,681
Contigs ≥ 1 kbp	176,794	104,239
Length ≥ 0 [bp]	4591,817,142	2449,395,592
Length ≥ 1 k[bp]	348,549,133	160,261,473
Longest [bp]	36,450	14,004
N50 [bp]	129	170
GC [%]	42.6	41.5
N [%]	0.0	0.0

Table 4.14.: Metrics of assemblies `Dm-gen-so-9`, `Dm-gen-so-10`, `Dm-gen-so-11` and `Dm-gen-so-12`. See table 4.1 for detailed description of metrics.

4. Assembling the genome of *D. muscipula*

(4.6 Gbp) and 16×10^6 contigs (2.5 Gbp) compared to 177×10^3 (349 Mbp) and 104×10^3 (160 Mbp) contigs ≥ 1 kbp.

Scaffolding of the assemblies did generate extremely poor results (data not shown), and may have been caused by an application error. Further analysis of the data was therefore omitted.

All three assemblies are highly fragmented with less than 8% of their respective total sizes made up by contigs of at least 1 kbp. The assembly with the highest contiguity could be obtained for a k -mer size of 77, while a smaller as well as a larger k -mer size performed worse. While larger k -mer size in general tend to mitigate graph ambiguities and hence further assembly contiguity and size, k -mer sizes too close to the length of the utilized reads decrease support for individual paths (as elaborated on for 127-mers and different read length above). This leads to an increased level of disruption of the graph and thus a more fragmented and at the same time smaller assembly. This series of assemblies illustrates that there exists local optima for different assembly parameters, in this case k -mer size, and that these optima often can only be determined empirically.

4.6.2. Alternative assemblers fail on the *D. muscipula* data

In the beginning of the flytrap genome projects, [ALLPATHS](#) and [SOAP](#) were the only assembly programs with the out-of-the-box capability of properly handling a 3 Gbp genome with reasonable hardware demands. In the course of the project, however, a handful of software was either upgraded or newly developed to take on this challenge. In order to verify the usability of these assemblers for the flytrap genome, the majority of these programs was tested on *D. muscipula* data.

ABYSS was the first new assembler reportedly applied to large plant genomes, such as the 3 Gbp *N. benthamiana* genome (Bombarely et al., 2012) and the 20 Gbp *P. glauca* genome (Birol et al., 2013). Tests on *D. muscipula* data were performed by Felix Bemm. A proper assembly, however, could not be obtained due to problems with the software as well as computational requirements.

MaSuRCA, an assembler build on the Celera assembly framework, was used to assembly the 22 Gbp genome of *Pinus taeda* (Zimin et al., 2014; Hamilton and Buell, 2014). An assembly of *D. muscipula* data, however, failed. During error correction and with different settings and input data, the program required more than the available 512GB of RAM. While the author of **MaSuRCA** agreed that in theory, the program should be able to handle the data set within the available memory, the issue could not be resolved.

Platanus was used for the assembly of the hexaploid *Ipomoea batatas* (Roullier et al., 2013). Assemblies for *D. muscipula* did compute, yet, even the best contig assembly comprised only 7.3 Mbp of contigs with at least 1 kbp. Due to the high fragmentation and low quality, further analyses of the assemblies were omitted.

Meraculous2 was used to generate an assembly of the 6 Gbp hexaploid *Triticum aestivum*

4.6. LGC reads further improve assembly contiguity and completeness

genome by Chapman et al. (2015). With the flytrap data, however, the software repeatedly crashed during its unitig construction phase. Attempts to fix the issue in cooperation with the authors did not succeed.

minia is an assembler with an extremely resource efficient implementation for De Bruijn graph construction and processing. **minia** was tested with k -mer sizes of 31, 51, 71, 91 and 111 on *D. muscipula* data. Only for a k -mer size of 51, a contig assembly comprising 6.9×10^6 with a total size of 1.2 Gbp and an N50 of 177 bp could be obtained. Given the high level of fragmentation of the assembly, further analysis was omitted. Noteworthy, however, is that the assembly was constructed with a peak memory demand of less than 10 Gbp.

Discover de novo, an assembler specifically developed for the assembly of large heterozygous data sets with the primary goal of identifying SNVs, was also tested on *D. muscipula* data. However, **Discover de novo** is designed to work with Illumina paired-end libraries generated on a HiSeq 2000 with a specific protocol generating overlapping read with a length of 250 bp. When provided with the 100 bp reads of *D. muscipula*, the program quickly ran out of memory and hence no assembly was obtained.

4.6.3. Combination of ALLPATHS and digital normalization is the superior strategy

The corrected LGC read data utilized in the SOAP assemblies described above, were generated from a full-data ALLPATHS assembly, similar to Dm-gen-ap-5. Again, external computation time on the 2 TB RAM node *mammoth* at the LRZ had to be procured in order to match the computational demands of ALLPATHS for reads sets of this size. The obtained assembly comprised 73×10^3 scaffolds of at least 1 kbp and a total scaffold length of 660 Mbp (table 4.15). With these metrics, the assembly is clearly superior to the full-data ALLPATHS assembly constructed from the BGI reads (68k contigs, 115 Mbp). Yet, with an N50 of 17kbp contiguity is lower than for assembly Dm-gen-ap-6 generated from normalized data and with 40% N-content, a considerable fraction of the assembly is comprised of uninformative gaps. Nevertheless, the obtained assembly indicates that the new read data when used with ALLPATHS produces better assemblies.

Metrics	Dm-gen-ap-11.1	Dm-gen-ap-11.2
Contigs ≥ 0 bp	253,506	76,042
Contigs ≥ 1 kbp	111,335	73,052
Length ≥ 0 [bp]	398,543,028	661,574,300
Length ≥ 1 k[bp]	304,128,659	658,714,918
Longest [bp]	65,605	602,575
N50 [bp]	2492	17,051
GC [%]	42.6	42.6
N [%]	0.0	39.8

Table 4.15.: Metrics of assembly Dm-gen-ap-11. See table 4.1 for detailed description of metrics.

4. Assembling the genome of *D. muscipula*

<i>k</i> -mer analysis	Dm-gen-ap-10/12
Genome size [bp]	952,546,772
↳ CN ≤ 1	562,815,26
Repetitive [%]	40.9
Coverage [<i>x</i>]	82
SNP rate	1/107

Table 4.16.: ALLPATHS *k*-mer analysis of Dm-gen-ap-10 / Dm-gen-ap-12

Assemblies Dm-gen-ap-10 and Dm-gen-ap-12 were constructed from digitally normalized read sets generated from raw data with *khmer*. Dm-gen-ap-10 was run with haploidification, whereas Dm-gen-ap-12 was run without allowing to directly assess the impact of the procedure. *k*-mer analysis resulted in an estimated genome size for the normalized set of 953 Mbp and a coverage of single copy regions of 82 *x* (table 4.16).

Assembly Dm-gen-ap-10 finished in 334 h and with a peak memory consumption of 453 Gbp. The assembly comprised 69×10^3 scaffolds of at least 1 kbp with a total size of 1,104 Mbp (Tab 4.17). Without haploidification, the assembly took with 588 h with peak memory usage of 453 GB. The resulting scaffold assembly was with 72×10^3 scaffolds >1 kbp and a total size of 1,007 Mbp slightly smaller and less contiguous. This supports the assumption that haploidification of the heterozygous data set increases assembly quality. Overall, the obtained assembly are of high contiguity outperforming all previously produced assemblies, with the exception of the BGI assembly.

Metrics	Dm-gen-ap-10.1	Dm-gen-ap-10.2
Contigs ≥ 0 bp	271,863	70,029
Contigs ≥ 1 kbp	177,093	68,704
Length ≥ 0 [bp]	718,849,301	1015,218,851
Length ≥ 1 k[bp]	657,422,302	1013,945,149
Longest [bp]	81,999	999,304
N50 [bp]	5407	31,814
GC [%]	42.5	42.5
N [%]	0.0	29.2
Metrics	Dm-gen-ap-12.1	Dm-gen-ap-12.2
Contigs ≥ 0 bp	288,147	73,733
Contigs ≥ 1 kbp	186,445	72,327
Length ≥ 0 [bp]	696,714,361	1008,033,297
Length ≥ 1 k[bp]	630,192,449	1006,682,621
Longest [bp]	78,053	996,199
N50 [bp]	4508	29,570
GC [%]	42.6	42.6
N [%]	0.0	30.9

Table 4.17.: Metrics of assembly Dm-gen-ap-10 and Dm-gen-ap-12. See table 4.1 for detailed description of metrics.

a total length of at least 100 Mbp. Taken together, both clusters represent more than 95 % of the total assembly size (1 Gbp), indicating that only a small fraction of sequences dominated by collapsed repeats is present in the assembly. Sequences in the cluster around 50 *x* coverage are predominantly composed of *k*-mers used twice as often as their number of occurrence in the read set indicates. This is the result of assembling

Figure 4.22 shows the topology of the assembly in the context of GC-content versus *k*-mer-coverage, and its cumulative contig length distribution. Color and size encoded circles on the left panel represent scaffolds with relative GC-content on the x-axis and representative adjusted *k*-mer-coverage on the y-axis. For better readability, only a 1 % subsample of the total number of scaffolds is actually displayed in the scatterplot. Overall topology of the plot is not affected by this procedure. The panel on the left holds a stacked histogram of the unsampled cumulative scaffold lengths at the respective *k*-mer-coverage bin.

The majority of scaffolds in terms of number as well as total length is found in a cluster between 80 *x* and 100 *x* and in a relative GC range of 35 % to 55 %. The cluster comprises sequences with a total length of at least 800 Mbp. A second, smaller cluster at half the coverage and with similar GC-content consists of sequences with

4.6. LGC reads further improve assembly contiguity and completeness

heterozygous regions into separate contigs and scaffolds. With 10% of the total assembly size, the fraction of these heterozygous contigs is still quite high.



Figure 4.22.: GC-content and length distribution of assembly *Dm-gen-ap-10.2* as a function of representative adjusted *k*-mer coverage. See fig. 4.4 for detailed plot description.

4.7. PacBio sequencing, correction and assembly

4.7.1. Low coverage PacBio sequencing

While digital normalization can mitigate the challenge of repeats in the assembly process by removing repetitive data and making computation feasible, it cannot solve the actual underlying problem: paths connecting genomic regions interspersed by repeats can only be resolved unambiguously by sequencing data directly tying together the flanks at either side of the repetitive region. In classic Illumina based sequencing, this is achieved by sequencing large insert paired-end and mate-pair libraries. The linkage information stored in the pairs is used to scaffold across ambiguities, filling the gaps with Ns, and in turn to largely increase contiguity.

For this form of scaffolding to work, the type of repeats in the genome at hand is of high importance. If the size of these repeats is larger than that of the inserts of the sequencing library, the repeats cannot be bridged. Preliminary studies by me (data not shown) followed by an in-depth analysis by Niklas Terhoeven in his master thesis showed that the *D. muscipula* genome predominantly comprises LTR repeats of the Ty1-copia and Ty3-gypsy class. With lengths of up to 20 kbp, these transposable elements are comparatively long. The BGI mate pair reads designed for scaffolding consist of libraries with insert-sizes of 2 kbp, 5 kbp, 10 kbp and 20 kbp. Therefore, only a part of the available read pairs is suited to close LTR-related assembly gaps.

Next to the length of the insert, also the length of the actually sequenced read matters. The concept of repetitiveness is relative with respect to the scope and scale one applies during the assessment. At the extremes, within a genome each single nucleotide is repetitive, as it occurs multiple times, whereas the entire genome is always unique. Repetitiveness can only be used in the context of smaller units occurring once or multiple times within a larger set. Applied to the scaffolding procedure, this means that contigs which are to be scaffolded together across a gap need to be non-repetitive in the sense that reads mapped to the corresponding flanks can be placed there uniquely. The mate-pair libraries generated by the BGI have a read length of only 49 bp. Compared to 100 bp reads obtained from paired-end sequencing, the short mate-pair reads are less likely to generate unique mappings, which considerably reduces their applicability and efficiency.

Recently emerging long reads sequencing technologies, such as PacBio's SMRT (Single molecule realtime, Eid et al., 2009; Levene et al., 2003) sequencing or Oxford Nanopore Technologies' nanopore sequencing are promising alternatives to mate-pair sequencing (and related technologies encoding long range information by linkage of multiple short fragments, e.g. phosmids). Nanopore sequencing, however, only became broadly available in late stages of the flytrap genome project and therefore was not considered.

The major advantages of PacBio reads in comparison to mate-pair libraries are that PacBio libraries can be constructed from smaller amounts of extracted DNA and at lower costs. Further, the technology exhibits no coverage-affecting compositional bias and the

4.7. PacBio sequencing, correction and assembly

information obtained from the read is not only long-ranged but continuous along the entire length of the read (Quail et al., 2012). This is particularly noteworthy in the context of the previously described relativity of repetitiveness, with longer reads having considerable higher likelihoods of being uniquely mappable.

The main disadvantage of PacBio reads is the high error rate of 15% and more. These errors, mostly comprising insertions and deletions, impede usage of raw PacBio data for assembly as well as scaffolding (Koren et al., 2012; C.-S. Chin et al., 2013; Hackl et al., 2014). Two general strategies for the assembly of PacBio read data were developed (see section 6.1 for details): approaches solely relying on PacBio data require high sequencing depth for the PacBio data to overcome the inherent high error rate. With respect to the difficulties in DNA extraction and preparation and the cost of sequencing, this rendered an application to *D. muscipula* infeasible. The alternative hybrid correction-based approach, on the contrary, utilizes high coverage Illumina short reads to construct highly accurate consensus sequences for PacBio long reads, thus integrating the advantages of both technologies. High coverage of PacBio data are not required. With Illumina data already available and motivated by the potential of the PacBio data to improve *D. muscipula* assembly contiguity, low coverage PacBio sequencing with subsequent hybrid correction and assembly was envisaged to further push the quality of the flytrap draft genome.

Including initial test cells, a total of 119 PacBio SMRT cells were sequenced at the FGCZ. After processing, the cells yielded a total amount of 6.5×10^9 reads with an overall length of 17 Gbp, equivalent to an estimated coverage of the 3 Gbp flytrap genome of 5.7x.

4.7.2. PacBio hybrid correction with proofread

Initial attempts to correct *D. muscipula* PacBio data with an early version of the PBCR (Koren et al., 2012) pipeline failed. Execution of the pipeline on in-house HPC nodes is infeasible due to excessive runtime requirements. Distribution on low memory cluster architectures was not possible due to incompatibilities with the scheduling software and the limited capabilities of the software to process subsets of data independently and at low computational demands.

These issues motivated the development of the PacBio hybrid correction pipeline proofread. proofread is designed on the *divide and conquer* principle and is capable of handling large-scale data by distribution of arbitrary sized subsets on any given hardware architecture.

Nevertheless, hybrid correction is computationally highly demanding. In order to complete the task for the flytrap data in reasonable time, computing time on the Linux cluster at the LRZ was procured through a project specific proposal. Additionally, digitally normalized reads Illumina read set were used to speed up correction.

4. Assembling the genome of *D. muscipula*

read set		reads	size [bp]	rl [bp]	N50 [bp]
Dm_GenPb	raw	5687189	14912055601	50-23733	3560
	untrimmed	4932208	13348560023	500-23548	3392
	trimmed	6173046	8676296048	500-16381	1708
dm-pb	raw	6447429	16962653250	50-26971	3485
	untrimmed	3884492	11954357927	500-26971	3874
	trimmed	4617106	9158050856	500-19366	2580

Table 4.18.: Metrics of PacBio libraries corrected with `proofread` v1.01 and normalized BGI Illumina reads (Dm_GenPb) / and `proofread` v2.12 and normalized LGC Illumina reads (dm-pb). Shown are number of reads, total library size, read length range and N50 for raw, corrected untrimmed and corrected trimmed data.

Two independent corrections were performed on the *D. muscipula* PacBio data. The first run was executed with v1.01 on PacBio batches 1 to 4 with digitally normalized BGI Illumina data. The generated set (Dm_GenPb) comprised 5.3×10^6 untrimmed reads with a total length of 13.5 Gbp and 10.5 Gbp respectively (Table 4.18). Trimming increased the number of reads to 14×10^6 and reduced the total length to 10.5 Gbp.

The second run was performed with `proofread` v2.12 on batches 1 to 5 with digitally normalized LGC Illumina data. The obtained set dm-pb consisted of 3.9×10^6 untrimmed reads with a total length of 12 Gbp. Trimming resulted in 4.6×10^6 reads with a total length of 9.2 Gbp.

Compared to the raw data, the untrimmed sets comprise 90 % and 71 %, and the trimmed sets 58 % and 54 % of the total amount of bases, respectively. Read counts for the trimmed data sets are higher than for untrimmed set, while their N50s drop considerably. In order to interpret the output and efficiency of the correction procedure properly, however, the following considerations have to be taken into account: `proofread` applies a dynamic minimum length cutoff of about two times the mean length of the provided Illumina data set. Further, given an error profile consisting of 10 % insertions and 5 % deletion, the net length of a fully corrected read on average is reduced by 5 %. Untrimmed reads comprise both, corrected and uncorrected regions, which puts them somewhere in between. Taken together, these reasons account for the difference of 10 % between raw and untrimmed corrected data in the case of the first correction run.

For the second run with `proofread` v2.12, an additional mechanism has to be considered. Due to the sequencing of circularized templates, a single DNA template can give rise to more than one PacBio subread. Essentially, all subreads of the same template encode identical and hence redundant information. To speed up correction and improve quality, `proofread` utilizes this information and collapses redundant subreads into a single representative consensus. Together with a minimum length the general reduction in size of corrected reads, this accounts for the decrease of the data set size by 29 %. However, as in the first case, these data cannot truly be considered lost, as they either derived from errors, reads too short to be useful or redundant information.

Trimming removes parts of the reads that obtained low quality scores during correction due to missing or insufficient per base support. Because mappings cannot extend across

PacBio read ends, terminal regions (approx. 1 bp-75 bp) exhibit low coverages. As a consequence, terminal regions are trimmed in the majority of reads. In addition, low quality regions can occur mid-read, splitting the read into multiple, independent parts. This increases the overall number of reads and has great impact on the N50, as cutting a read in the middle immediately halves its N50, even if only a small fraction of data is actually lost.

A more robust estimation of the through-put efficiency of the correction can be obtained from the direct comparison of untrimmed and trimmed reads. For the first run this ratio is 65 %, for the second run 77 %. The difference can be attributed to the optimizations implemented in *proofread* between the two releases as well as improved quality of the utilized reads data. Nevertheless, both values are at the lower end of output rates observed for *proofread* on other data sets. The efficiency of the correction directly depends on the quality of the sequencing of the PacBio data. In particular the first batches of *D. muscipula* data are inferior due to a less optimized sequencing chemistry and DNA of lower quality. In addition, the high level of heterozygosity makes the construction of robust consensus sequences more difficult and affects overall correction efficiency.

4.7.3. Transcriptome coverage of the PacBio data

To estimate completeness of the PacBio read set, transcriptome coverage was assessed. For the sake of comparability, the same methodology as for Illumina based assemblies was used. Figure 4.23 shows a stacked histogram comparing the results obtained for the BGI assembly *Dm-gen-so-1*, the trimmed PacBio reads and PacBio-based assemblies described later in this chapter. In total, slightly less than 35×10^3 transcripts could be mapped to the PacBio reads set. Of these, approximately 23×10^3 were aligned full-length (red), 3×10^3 aligned full-length but split over different reads and about 8.5×10^3 transcripts only aligned partially.

Compared to *Dm-gen-so-1*, the amount of alignments mapped in full-length is considerably larger, indicating that the high level of fragmentation of the assembly is a major issue with respect to the identification of complete gene models. However, the total amount of mappable transcripts is with 3×10^3 additional transcripts only slightly larger. This result can be interpreted in two ways: On the one hand, PacBio reads are, in contrast to contigs, not affected by any assembly graph related trimming. Given sufficient coverage, the PacBio read set should comprise the complete transcriptome. Therefore, the missing alignments could imply incompleteness of the PacBio read set due to insufficient coverage. On the other hand, if one were to assume completeness, the result would indicate an upper boundary for the resolution of the employed method. Transcripts without mapping would either not actually derive from the flytrap genome or are not mappable with the utilized method. The first case would point towards contaminations, the latter holds true for transcripts, for example composed entirely of very short (<100 bp) exons, which can easily be missed by .

4. Assembling the genome of *D. muscipula*

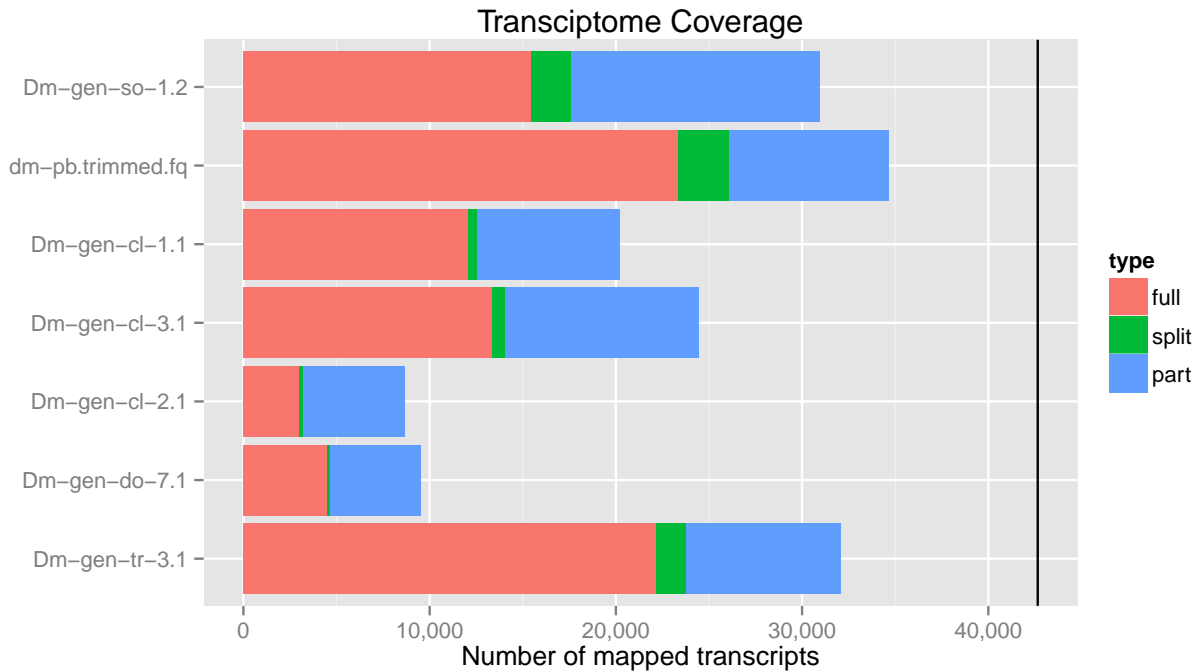


Figure 4.23.: Transcriptome content of corrected PacBio reads and PacBio based assemblies *Dm-gen-cl-1*, *Dm-gen-cl-2*, *Dm-gen-cl-3*, *Dm-gen-do-7* and *Dm-gen-tr-3* in comparison to *Dm-gen-so-1*. Bars of the stacked histogram represent the amount of full-length (red), full-length but split across reference sequences (green) and partially (blue) mapped longest isoforms with maximum 42,653 (black line) derived from transcriptome assembly *DM_tra_qt1.03*.

For the flytrap PacBio data, incompleteness due to insufficient coverage cannot be excluded. However, given a coverage of $5x$, at least from a statistical point of view, the effect should be minor. Consequently, it can be concluded, that in fact, the applied method for the estimation of transcriptome coverage is limited, and that the results obtained for the PacBio read set indicate an upper boundary of resolution for the method.

4.7.4. PacBio-only and PacBio-Illumina hybrid assemblies

Assemblies *Dm-gen-cl-1* and *Dm-gen-cl-3* were generated with Celera v7 and v8, respectively, from corrected PacBio reads of the first proofread run. The resulting assemblies comprised 248×10^3 and 543×10^3 contigs with a total lengths of 1 Gbp and 1.3 Gbp. With N50s of 4.9 kbp and 2.8 kbp, both assemblies are fragmented. With sizes above 1 Gbp, the assemblies are larger than the majority of assemblies obtained from Illumina data, however, compared the an estimated genome size of 2.8 Gbp, the assemblies appear incomplete. This is further corroborated by analysis of transcriptome coverage (fig. 4.23). With approximately 20×10^3 and 24×10^3 of total mapped transcripts

4.7. PacBio sequencing, correction and assembly

(12×10^3 and 14×10^3 full-length) both assemblies are inferior to the BGI assembly and lack a substantial portion of gene coding regions present in the PacBio read data.



Figure 4.24.: GC-content and length distribution of assembly *Dm-gen-cl-3.1* as a function of representative adjusted k -mer coverage. See fig. 4.4 for detailed plot description.

Analysis of the GC-coverage and contig length landscape (fig. 4.24) revealed that a considerable fraction of contigs is comprised of collapsed repeats. At the same time, the obtained coverages are more scattered across a range of coverages rather than concentrated at either the haploid or the diploid genome coverage, as observed for Illumina-based ALLPATHS assemblies. This shows that the assembly comprises both, over- and underrepresented sequences and that the assembly is of overall low quality.

Attempts to assemble the corrected low coverage PacBio read sets with MIRA did not succeed. The best assembly obtained comprised less than 5×10^3 contigs with a total length of 10 Mbp (data not shown). Further exploration of MIRA assemblies hence was omitted.

Metrics	<i>Dm-gen-cl-1.1</i>	<i>Dm-gen-cl-3.1</i>
Contigs ≥ 0 bp	248,345	542,858
Contigs ≥ 1 kbp	248,345	542,857
Length ≥ 0 [bp]	1083,811,606	1325,334,332
Length ≥ 1 k [bp]	1083,811,606	1325,333,333
Longest [bp]	30,443	24,132
N50 [bp]	4857	2762
GC [%]	43.4	43.6
N [%]	0.0	0.0

Table 4.19.: Metrics of assembly *Dm-gen-cl-1* and *Dm-gen-cl-3*. See table 4.1 for detailed description of metrics.

4. Assembling the genome of *D. muscipula*

While *Celera* and *MIRA* are suited (at least in theory) to work with corrected PacBio data, assemblers developed for raw PacBio data was also considered. The HGAP (Hierarchical Genome Assembly Process, C.-S. Chin et al., 2013) pipeline uses a hierarchical approach to pre-correct PacBio reads by PacBio-PacBio consensus and ultimately assembles the resulting reads with *Celera*. Given that the recommendation for the HGAP pre-correction is at least 20x PacBio coverage and that the procedure itself is likely to be computationally infeasible for the flytrap genome, no test runs with this software were performed.

A similar reasoning holds true for the assembler. While this assembler was shown to be capable of assembling large diploid genomes from raw PacBio data, the required PacBio coverage is 20x or more. Additionally, the software is still in developmental stage, and any attempts to properly install and execute it, failed.

Metrics	Dm-gen-cl-2.1
Contigs \geq 0 bp	754,009
Contigs \geq 1 kbp	754,008
Length \geq 0 [bp]	2027,352,664
Length \geq 1k[bp]	2027,351,665
Longest [bp]	25,605
N50 [bp]	3204
GC [%]	44.0
N [%]	0.0

Table 4.20.: Metrics of assembly Dm-gen-cl-2. See table 4.1 for detailed description of metrics.

In addition to PacBio-only assemblies, also the performance of various hybrid approaches was evaluated. For assembly Dm-gen-cl-2, generated with *Celera* v7, in addition to the corrected PacBio reads, merged BGI overlapping libraries were supplied. The obtained assembly yielded 754k contigs with total length of 2 Gbp and an N50 of 3.2 kbp (Table 4.20). The assembly is considerably larger than the assemblies obtained with *Celera* and PacBio data only, while of similar contiguity. However, the amount of mappable transcript dropped substantially (Figure 4.23) implying that the assembly is highly incomplete with respect to non-repetitive and coding regions. A comparable run with *Celera* v8 was terminated after having taken more than four weeks to get to the buildUnitigs step.

Next to OLC approaches, also the applicability of PacBio data for De Bruijn graph based assemblies was analyzed. For ALLPATHS assembly Dm-gen-ap-8, additional artificial overlapping libraries generated with pb2il (section 7.1) from corrected PacBio reads were utilized. The obtained assembly, however, was of smaller size and contiguity than comparable Illumina-only assemblies.

4.7.5. PacBio hybrid assemblies with DBG2OLC and corrected PacBio reads

The recently published DBG2OLC assembler employs a novel approach that tries to integrate advantages of Illumina and PacBio-based assembly. DBG2OLC uses PacBio reads and in addition precomputed unitigs or contigs obtained from an Illumina De Bruijn graph-based assembly as input. The assembly sequences are used as anchors and to construct a compressed overlap graph for the PacBio reads. This makes DBG2OLC computationally highly efficient in terms of speed as well as resource requirements. However, Similar to *FALCON*, the software is in a developmental stage. I was able to fix problems occurring during the initial assembly step by modifying the source code, yet the second, consensus refinement stage continued to fail. Nevertheless, a series of

4.7. PacBio sequencing, correction and assembly

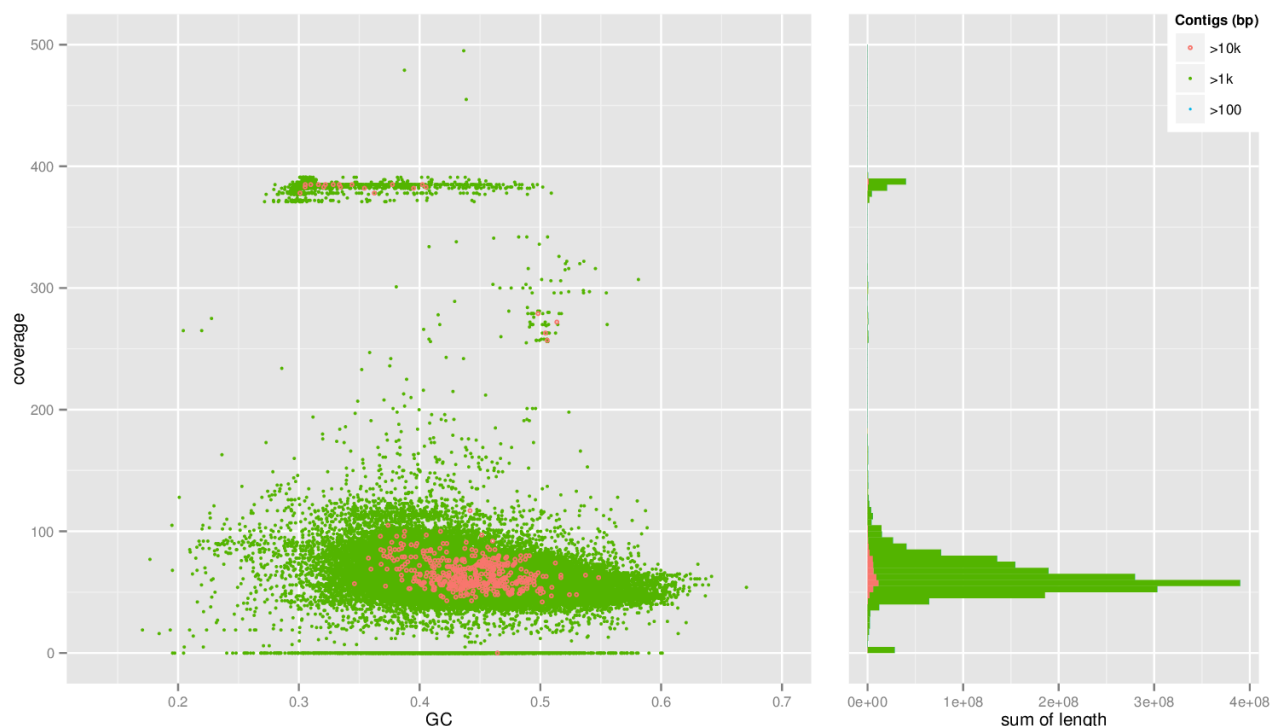


Figure 4.25. GC-content and length distribution of assembly `Dm-gen-cl-2.1` as a function of representative adjusted k -mer coverage. See fig. 4.4 for detailed plot description.

preliminary assemblies for *D. muscipula* was computed.

In contrast to the published approach, the assemblies were constructed from proovread corrected PacBio reads and with customized parameter settings. Next to faster computation this has the advantage that the unrefined assemblies have lower per base error rates allowing for qualitative assessment despite the failing of the consensus step.

Overall, the assemblies generated (section 8.6) with DBG2OLC were of mediocre contiguity with N50s ranging from 11 kbp to 16 kbp. Total sizes, however, varied substantially with parameter settings and the utilized contig set. Contig sets obtained from non-normalized Illumina reads in general performed worse. The largest assembly (`Dm-gen-so-7`) was obtained with `Dm-gen-ap-10.1` contigs (table 4.21). It comprises 84×10^3 contigs with a total length of 940 Mbp and an N50 of 13 kbp.

The structure of the assembly was further assessed by GC-coverage and contig length distribution (fig. 4.26). Due to the utilization of untrimmed PacBio reads, which include

Metrics	Dm-gen-do-7.1
Contigs ≥ 0 bp	84,310
Contigs ≥ 1 kbp	84,230
Length ≥ 0 [bp]	940,091,810
Length ≥ 1 k [bp]	940,037,689
Longest [bp]	77,231
N50 [bp]	13,049
GC [%]	43.8
N [%]	0.0

Table 4.21. Metrics of assembly `Dm-gen-do-7`. See table 4.1 for detailed description of metrics.

4. Assembling the genome of *D. muscipula*



Figure 4.26.: GC-content and length distribution of assembly `Dm-gen-do-7.1` as a function of representative adjusted k -mer coverage. See fig. 4.4 for detailed plot description.

erroneous regions, a large fraction (400 Mbp) of the contigs is dominated by k -mers absent from the Illumina derived k -mer hash. The majority of non-zero coverage contigs is shifted from the expected coverages for haploid and diploid sequences towards higher coverage, indicating the predominant presence of k -mers derived from collapsed repeats in the constructed contigs.

Using `DBG2OLC` with `ALLPATHS`-derived contigs does not produce assemblies of higher contiguity and quality than scaffolded `ALLPATHS` assemblies. By design, `DBG2OLC` does not work with scaffolds, because the N-gaps do not provide any additional information that could be used to further compress or simplify the overlap graph. With stand-alone scaffolding being inferior to `ALLPATHS` and the overall lower contiguity of the `DBG2OLC` assemblies, in combination with potential issues related to the presence of collapsed repeats and a high level of erroneous bases, an actual improvement of the assembly quality through the use of `DBG2OLC` could not be established.

4.7.6. Custom approaches towards a PacBio assembly

The overall unsatisfying results obtained from PacBio based assemblies motivated the exploration of customized approaches towards PacBio based assemblies. With the major goal of deciphering the gene content and coding potential of the flytrap, completeness of individual genes as well as the entire gene set captured in the assemblies is a primary concern.

With the *transcript guided* assembly pipeline, gene containing regions are specifically targeted for assembly. Exon containing long reads (or contigs) are identified by alignment to available transcriptomic data. For each transcript, reads are collected and assembled individually in order to obtain full length gene models.

Assembly `Dm-gen-tr-3` was generated from proofread corrected PacBio reads and the reference transcriptome assembly `DM_tra_qt1.03`. The assembly resulted in a total number of 42k contigs with total assembly size of 189 Mbp and an N50 of 6 kbp (table 4.22). The very small size of the assembly makes sense, given that the assembly mostly comprises exons and introns as well as some up- and downstream regions of targeted genes. Analysis of the transcriptome content (fig. 4.23 shows very high amounts of mappable transcripts and, in particular, full-length hits, only surpassed by the entire corrected PacBio reads set and the ALLPATHS assembly `Dm-gen-ap-10`).

Metrics	Dm-gen-tr-3.1
Contigs \geq 0 bp	42,120
Contigs \geq 1 kbp	38,900
Length \geq 0 [bp]	188,765,959
Length \geq 1 k[bp]	186,836,727
Longest [bp]	41,097
N50 [bp]	5925
GC [%]	39.6
N [%]	0.0

Table 4.22.: Metrics of assembly `Dm-gen-tr-3`. See table 4.1 for detailed description of metrics.

In terms of k -mer coverage, the generated contigs exhibit a pattern very similar to ALLPATHS assemblies, with the majority of contigs belonging to either one of two distinct populations of heterozygous and homozygous sequences (fig. 4.27). In addition, a minor population of contigs was found at about one third of the expected diploid genome coverage, representing sequences with a relative overuse factor of three. This effect can probably be attributed to either redundant seeds (isoforms) or nested gene models, which due to independent assembly of the recruited reads generate multiple contigs for overlapping regions.

The obtained results show that *transcript guided* assembly is a viable approach towards a targeted assembly of *D. muscipula* gene models. However, the fact that the amount of full-length transcripts as well as the overall amount of mapped transcripts in the assembled reads slightly decreases compared to the corrected PacBio reads, also demonstrates the limitations of the current data set and implementation. The micro assembly of recruited PacBio reads only in very few cases led to the completion of gene models only partially present in the unassembled reads, while at the same time some regions containing transcripts were lost. This could either be caused by the parts of the targeted genes not being covered by PacBio data (or by reads not overlapping enough to actually be assembled together) or by exons being too far apart from each other to be connected by any PacBio read present in the data set. Nevertheless, the *transcript guided* assembly

4. Assembling the genome of *D. muscipula*

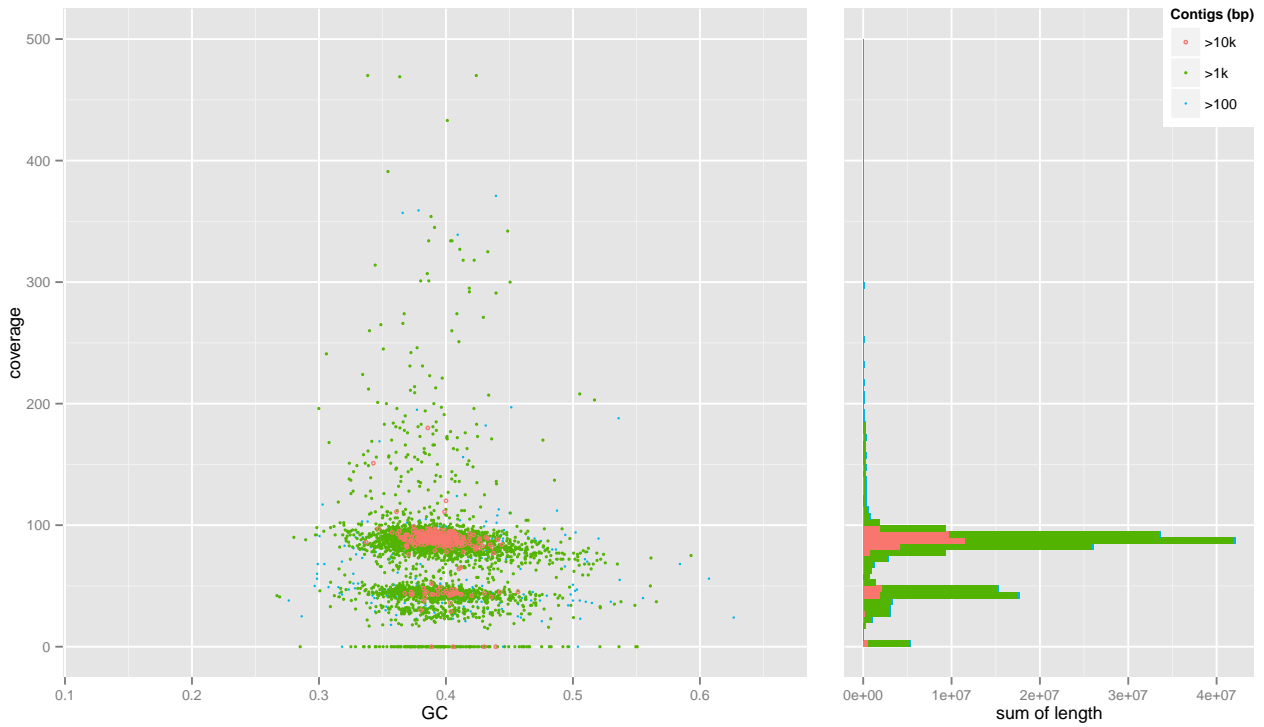


Figure 4.27.: GC-content and length distribution of assembly `Dm-gen-tr-3.1` as a function of representative adjusted k -mer coverage. See fig. 4.4 for detailed plot description.

serves as valuable case study on how available transcriptome data can be used to further the *D. muscipula* genome assembly.

4.8. The *D. muscipula* draft genome assembly

To produce a final, high quality draft assembly for the Venus flytrap, I combined the findings described in previous chapters regarding assembly optimization into a single strategy. The assembly `Dm-gen-ap-13` was generated using `ALLPATHS` with a minimum contig size of 200 bp and haploidification enabled. As overlapping paired-end libraries I used digitally normalized LGC reads, obtained through an improved normalization run with `bbnorm.sh`. For scaffolding, the entire set of BGI mate-pair libraries plus the BGI paired-end libraries with insert sizes of 500 bp and 800 bp as well as artificial mate-pair generated from corrected PacBio reads were supplied. `ALLPATHS` *k*-mer analysis resulted in an estimated genome size of 1.1 Gbp for the normalized read set and a coverage of single copy regions of 78 *x* (table 4.23).

Assembly `Dm-gen-ap-13` finished in 1,400 h and with a peak memory consumption of 453 GB. The resulting contig assembly comprises 247.8×10^3 contigs ≥ 1 kbp with a total length of 1.08 Gbp, an N50 of 5.0 kbp and a longest sequence of 77 kbp (table 4.24). The scaffold set has 104×10^3 sequences ≥ 1 kbp, with a total length of 1.45 Gbp, an N50 of 34.7 kbp and a longest scaffold of >1 Mbp. With these statistics, `Dm-gen-ap-13` is the most contiguous of all available flytrap assemblies, and at the same time the largest assembly in comparison to all previously obtained `ALLPATHS` assemblies. This suggests that the assembly is of high quality and completeness.

Two additional post-processing steps were performed to further improve the assembly. First, `Redundans` (Pryszcz and Gabaldón, 2016) was used to identify and merge redundant scaffolds that resulted from the assembly of different heterozygous regions into separate contigs. The procedure reduced the number of scaffolds to 87.3×10^3 and the overall assembly size to 1.42 Gbp, with an N50 of 35.9 kbp (table 4.24). Subsequently, I used `PBJelly` to further increase the contiguity of the assembly by scaffolding and extending the available sequences and filling N containing gaps. `PBJelly` returned an assembly comprising 86.6×10^3 sequences ≥ 1 kbp with a total assembly size of 1.45 Gbp and an N50 of 36.4 kbp (table 4.24).

<i>k</i> -mer analysis	Dm-gen-ap-13
Genome size [bp]	1,105,865,866
\perp CN ≤ 1	570,805,138
Repetitive [%]	48.4
Coverage [<i>x</i>]	78
SNP rate	1/163

Table 4.23.: `ALLPATHS` *k*-mer analysis of `Dm-gen-ap-13`

Metrics	Dm-gen-ap-13.1	Dm-gen-ap-13.2
Contigs ≥ 0 bp	533,480	104,847
Contigs ≥ 1 kbp	246,538	104,089
Length ≥ 0 [bp]	1077,631,653	1454,767,257
Length ≥ 1 k[bp]	961,303,673	1454,046,846
Longest [bp]	77,013	1004,118
N50 [bp]	5014	34,655
GC [%]	42.8	42.8
N [%]	0.0	25.9
Metrics	Dm-gen-ap-13.3	Dm-gen-ap-13.4
Contigs ≥ 0 bp	87,488	87,002
Contigs ≥ 1 kbp	87,285	86,832
Length ≥ 0 [bp]	1420,612,294	1445,729,224
Length ≥ 1 k[bp]	1420,417,640	1445,566,239
Longest [bp]	1004,118	1010,313
N50 [bp]	35,927	36,397
GC [%]	42.7	42.7
N [%]	26.5	23.4

Table 4.24.: Metrics of assembly `Dm-gen-ap-13`. See table 4.1 for detailed description of metrics.

4. Assembling the genome of *D. muscipula*

Figure 4.28 shows the topology of the refined assembly `Dm-gen-ap-13.4` in the context of GC-content versus k -mer-coverage, and its cumulative contig length distribution. Color and size coded circles on the left panel represent scaffolds with relative GC-content on the x-axis and representative adjusted k -mer-coverage on the y-axis. For better readability, only a 1% subsample of the total number of scaffolds is actually displayed in the scatterplot. Overall topology of the plot is not affected by this procedure. The panel on the left holds a stacked histogram of the unsampled cumulative scaffold lengths at the respective k -mer-coverage bin.

The majority of scaffolds in terms of number as well as total length is found in a cluster between $80x$ and $100x$ and in a relative GC range of 35% to 55%. The cluster comprises sequences with a total length of about 1.3 Gbp. A second, smaller cluster at half the coverage and with similar GC-content consists of sequences with a total length of less than 80 Mbp. Taken together, both clusters represent more than 95% of the total assembly size (1.45 Gbp), indicating that only a small fraction of sequences dominated by collapsed repeats is present in the assembly. Sequences in the cluster around $50x$ coverage are represent either heterozygous contigs that could not be merged, or sequences actually present only in one of the two chromosomal copies. With less than 5% of the of the total assembly size, the fraction of these potentially heterozygous contigs is lower than for any of the previously generated assemblies further corroborating the comparatively high quality of this assembly.

4.8. The *D. muscipula* draft genome assembly



Figure 4.28.: GC-content and length distribution of assembly `Dm-gen-ap-13.4` as a function of representative adjusted k -mer coverage. See fig. 4.4 for detailed plot description.

Further evidence for the high level of completeness for the assembly is given by the results of the BUSCO analysis (table 4.25) and transcriptome coverage. Scaffold as well as refined assemblies contain 79% of full-length and 2.7% to 2.9% of partially annotated single-copy core orthologs. The duplication level for the initial scaffold assembly is 5.3% and decreases slightly to 4.9% after the application of `Redundans` to the data. These values are in congruence with the observations of the GC-coverage plot, which also suggests that a portion of $\sim 5\%$ of the scaffolds seem to be redundant due to separate assembly of heterozygous regions. In terms of transcriptome coverage (data shown in fig. 5.1, chapter 5), `Dm-gen-ap-13` is superior to all other assemblies and even outmatches the transcriptome coverages observed for the corrected PacBio reads set. In summary, it can be concluded that the assembly at hand is *healthy* with respect to coverage and heterozygous contigs and that it shows a high level of completeness and contiguity with respect to potential coding regions.

	in [%]:	compl.	dupl.	part.
<code>Dm-gen-ap-13.2</code>		79	5.3	2.7
<code>Dm-gen-ap-13.3</code>		79	4.9	2.7
<code>Dm-gen-ap-13.4</code>		79	4.9	2.9

Table 4.25.: BUSCO analysis of assembly `Dm-gen-ap-13` based on 956 near-universal single-copy orthologs for scaffolds (`Dm-gen-ap-13.2`), after removal of heterozygous contigs with `Redundans` (`Dm-gen-ap-13.3`) and after additional scaffolding and gapclosing with corrected PacBio reads using `PBJelly`.

5. Conclusion

The de novo assembly of large plant genomes in general, and of *D. muscipula*, in particular, is a highly challenging endeavor. The size of the diploid flytrap genome was estimated to a range of 2.6 Gbp to 3.0 Gbp and the relative amount of repetitive sequences to at least 70%. The initial draft assembly was sequenced and assembled by the BGI in 2011. With a size of 3.7 Gbp and more than 6.5×10^6 scaffolds (2.3 Gbp and $355 \times 10^3 \geq 1$ kbp) the assembly appears to be rather complete, however, with the huge number of sequences and an N50 of for the entire set, highly fragmented as well. Assessment of the assembly with respect to gene content showed that only about one third of the expected genes is represented in the assembly in full-length, while the rest is either only present partially or missing entirely. Moreover, coverage analyses revealed that due to an overassembly of repetitive regions the sequence set is artificially inflated, enhancing the illusion of completeness.

The assembly was generated from a series of Illumina HiSeq paired-end and mate-pair libraries constructed and sequenced at the BGI. In-depth analysis of the read data revealed technical biases with respect to GC-content and error rate, as well as high levels of heterozygosity and heterogeneity, caused by the pooling of samples from different, non-monoclonal plants. In combination with the high level of repetitiveness of the genome, these issues pose major challenges for the construction of a high quality draft assembly and are the reasons for the low quality of the BGI assembly.

For the computation of the assembly, the BGI utilized the assembly software SOAP. While SOAP is capable of constructing assemblies of gigabase-sized genomes within reasonable time and computational demands, other assemblers, such as ALLPATHS, have been shown to produce superior assemblies from challenging eukaryotic data sets, however at the expense of substantially higher hardware requirements. Through application of digital normalization, a computational technique to reduce the amount of repetitive data in sequencing sets while maintaining the overall information content, I was able to overcome these computational challenges. The resulting ALLPATHS assemblies were of higher completeness with respect to gene content and showed a much better resolution of heterozygous and repeat regions. However, the overall quality of the assembly remained poor and did not allow for proper downstream analyses.

While it was possible to address the issue of repeats and heterogeneity to some extent computationally, further improvement of the assembly, at this point, could only be achieved by the generation new data of improved quality. The issue of heterogeneity was solved by the sequencing of four new Illumina overlapping libraries from non-polymorphic

5. Conclusion

high-quality DNA samples at LGC. Moreover, to support scaffolding and resolution of repeats, a 5x set of PacBio reads was generated. Correction of the large PacBio data set was performed with , a hybrid correction pipeline, which I developed specifically for this purpose. While different assembly approaches directly based on the low-coverage PacBio read data did not produce assemblies of high quality, artificial mate-pair libraries generated from corrected PacBio libraries were successfully used to improve the scaffolding process of Illumina-based assemblies.

Ultimately, I was able to combine all of the aforementioned findings into a unified assembly strategy. Using LGC libraries with low heterogeneity, normalized with an optimized procedure, BGI and PacBio-derived scaffolding libraries and customized assembler settings accounting for heterozygosity and graph complexity, I generated a preliminary *D. muscipula* draft genome assembly. The assembly was further refined by removing redundant sequences constructed due to heterozygosity, and by scaffolding of the remaining sequence set with corrected PacBio reads. The final assembly is 1.45 Gbp in size and comprises 86×10^3 scaffolds ≥ 1 kbp. Figure 5.1 shows a comparison of different milestone assembly with respect to assembly size and contig / scaffold length distribution and transcriptome coverage. The latter serves as simple measurement representing the level of completeness of the respective assembly. Through a series of optimizations to the assembly strategy, addressing all the aforementioned issues, I was able to consequently push the quality of the draft assemblies as indicated by the gradual increase of transcriptome coverage.

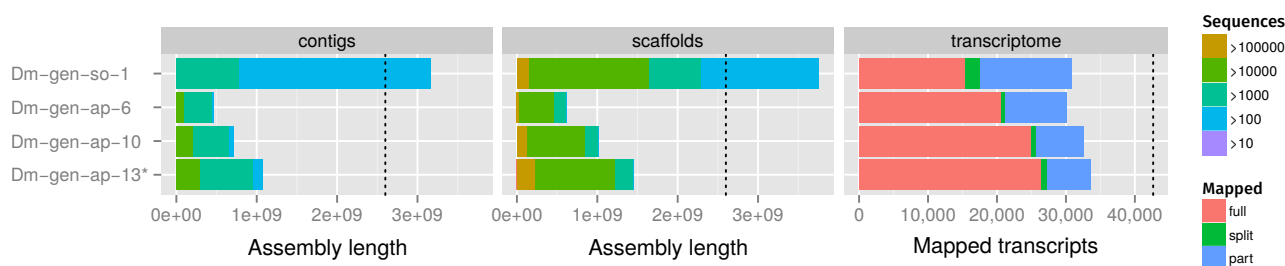


Figure 5.1.: Comparison of different milestone assemblies with respect to assembly size and contig / scaffold length distribution and transcriptome coverage

While with a total size of 1.45 Gbp, the draft genome, at the first glance, still seems to be incomplete, transcriptome coverage and conserved gene analysis show otherwise. The absolute difference in size between the assembly and the estimate for the flytrap genome, can be explained by accounting for the low level of complexity within the vast amount repetitive elements present in the genome. From the k -mer analysis, it is possible to compute the amount of the unique sequence information represented by all repetitive k -mers present in *D. muscipula*, simply by counting the number of k -mers and ignoring their individual abundances: it adds up to less than 40 Mbp. That means, that the flytrap only comprises a very limited set of different repeat species, yet the copies of these repeats are of very low diversity and at the same time extremely abundant. During assembly, the vast majority of these repeat sequence is collapsed. This is illustrated particularly

well by the case of a ~128 bp tandem repeat, most likely the generic repetitive unit of the centromeres of *D. muscipula*, that based on *k*-mer coverage estimates, makes up for about 300 Mbp of the entire genome, yet only a couple of hundred bases of the assembly. This interpretation is also corroborated by the results obtained from mapping available genomic read data to the assembly, for which mapping rates of >99 % were observed (data not shown). Moreover, preliminary gene prediction and annotation, currently performed by colleague Niklas Terhoeven, indicate that a comparatively complete set of gene models can be compiled from the draft assembly. In summary, it can be concluded that the assembly generated for the *D. muscipula* genome constitutes a draft assembly of high quality. It represents the genetic mark-up of Venus flytrap to a high degree of completeness and, therefore, serves as a proper reference in our quest of deepening the understanding of carnivorous plant in general and this *most wonderful* plant, in particular.

Part II.

Development of bioinformatics tools and libraries

6. proovread – large-scale hybrid PacBio correction through iterative consensus

6.1. Introduction

The development of the Illumina high-throughput next generation sequencing platforms gave rise to a new era in sequencing-based research fields, such as genomics, transcriptomics and metagenomics. The massive amounts of reads that can be generated rapidly, at low costs and with low error rates revolutionized molecular biology (Quail et al., 2012). However, Illumina reads have a particular drawback: with typical read lengths of 100 bp to 300 bp, depending on platform and chemistry (Illumina, 2016), the reads are very short. Although, paired-end sequencing allows to co-encode limited long-range information, the assembly of complex genomes comprising repeats, conserved motives and domains poses a challenge, unresolvable with short read data alone (Gnerre et al., 2011).

More recently, this particular problem was addressed by emerging third generation real-time single molecule sequencing technologies. SMRT (Single molecule realtime, Eid et al., 2009; Levene et al., 2003) sequencing developed by PacBio (Pacific Biosciences), monitors the incorporation of fluorescently labelled nucleotides into a newly synthesized DNA strand by an immobilized polymerase, thus indirectly determining the sequence of the template. The MinION (Kasianowicz et al., 1996) system developed by Oxford Nanopore Technologies infers nucleotide directly from analysis of electric properties of individual bases, which are measured during the translocation of a DNA molecule through a protein nanopore. Both technologies generate long reads of several kilobases. However, in both cases, the gain in read length comes at the price of a largely reduced accuracy. The high error rates of 15 % to 20 % for PacBio (Ono et al., 2013; Ross et al., 2013) and 5 % to 40 % for Nanopore sequencing (Goodwin et al., 2015), greatly hamper the usability of the long read data in assembly and downstream analysis. Further, production ready MinION systems for large scale projects, have only become available very recently.

For PacBio data, different approaches for the assembly of the erroneous reads have been developed. The HGAP (Hierarchical Genome Assembly Process, C.-S. Chin et al., 2013) pipeline, released by PacBio, uses alignments of shorter to longer PacBio reads of the same sample to generate long consensus sequences of high accuracy. These pre-corrected reads are further constructed into an assembly with OLC (Overlap-Layout-Consensus) assemblers suited for long accurate reads, such as Celera (E. Myers et al., 2000; Berlin et al., 2015) and MIRA (Chevreux et al., 1999).

6. *proofread* – large-scale hybrid PacBio correction through iterative consensus

For HGAP and similar PacBio-only assembly approaches currently in development, such as DAZZLER (G. Myers, 2014) and FALCON (J. Chin, 2016), high PacBio coverage is required in order to allow for efficient pre-correction of the longest reads subset. If generation of high PacBio coverage is problematic, Illumina-PacBio *hybrid* procedures offer a PacBio-coverage independent alternative. Here, only a low coverage set of PacBio reads is sequenced, and in addition, a high coverage set of Illumina reads. Using computational methods, the Illumina data can be used to identify and correct errors in the complementary PacBio read set. The tools PBcR (Koren et al., 2012) and LSC (Au et al., 2012) use Illumina-PacBio alignments as basis for consensus computation. The more recently developed LoRDEC (Salmela and Rivals, 2014) first constructs a De Bruijn graph (Bruijn, 1946) from provided Illumina data, then matches PacBio reads based on k -mers to error-free paths in the graph.

The development of the new hybrid correction software *proofread* (Hackl et al., 2014) was motivated by the challenges encountered during the Venus flytrap genome assembly. For the highly repetitive genome, in addition to Illumina data, PacBio reads with a coverage of $5x$ were sequenced to support the assembly procedure. The large size of the genome rendered sequencing of PacBio data with higher coverages, as required for a PacBio-only correction and assembly strategy, intractable. Hybrid correction, however, was a promising alternative given that Illumina data was already available. Yet, PacBioToCA (an early version of PBcR), the only hybrid correction software available at the time, was incapable of handling the large amount of *D. muscipula* read data on available hardware infrastructures. *proofread*, thus, was designed as a mapping and consensus bases hybrid correction pipeline with particular focus on being applicable to large-scale projects, both in terms of hardware as well as run-time demands.

6.2. proofread 1.0 – legacy implementation and evaluation

6.2.1. Hybrid scoring model and suitable mappers

The first step toward a consensus based Illumina-hybrid correction is the successful and efficient generation of alignments of Illumina and PacBio reads referred to as *hybrid* alignments. The high error rate and distinctive nature of the SMRT sequencing, however, renders this task challenging (Chaisson and Tesler, 2012). Common alignment scoring models are designed to either account for evolutionary differences, mostly comprising nucleotide exchanges and a moderate amount of gaps, or in the case of most mappers for the very low error rates of Illumina data and naturally occurring nucleotide variations. The alignment of Illumina reads onto erroneous PacBio reads with default settings, as tested on sample data, results in no or at best spurious alignments (data not shown).

However, based on the specific error profile of PacBio reads with roughly 10% insertions, 5% deletions and 1-2% substitutions (Ono et al., 2013; Ross et al., 2013), it is possible to formulate a new, generalized model with the following constraints: Gaps caused by indels (Insertion / Deletions) appear frequent in both, query (Illumina read) and reference (PacBio read) and hence need to have low costs as compared to mismatches attributed to substitutions. Insertions (gaps in query) occur twice as often as deletions (gaps in reference), which should be reflected by halved costs. Substitutions are rare, which should result in penalties of about 10 times the gap costs. Contrasting to biological scenarios, for which multi-base deletions and insertions are likely and which is accounted for by affine gap cost models, the distribution of errors in PacBio reads is random (Quail et al., 2012). Hence, costs for gap extensions need to be at least as high as gap opening costs.

The implementation of the hybrid model in practice strongly depends on the utilized mapping software. The alignment of high-coverage Illumina short read data to large reference genomes is computationally challenging and gave rise to a series of highly optimized mapping algorithms (Langmead et al., 2009; H. Li and Durbin, 2009; David et al., 2011; H. Li, 2013; Bushnell, 2014). For Illumina-PacBio hybrid alignment, the scenario is further complicated: PacBio read sets, even if sequenced at low coverage, will have a total size several times larger than the actual sequenced genome. Mapping to a $10x$ read set is similar to mapping to a ten times larger genome. On top, the main factor determining alignment speed (and a key target of algorithm optimizations), is the amount of tolerated differences, which in this case is equal to the amount of tolerated errors. While Illumina mappers are designed to operate at accuracies close to 100% (Langmead et al., 2009), hybrid correction requires alignments of reads with as little as 80% accurate bases. Hence, a combination of high speed and high sensitivity is essential for mappers suited for usage with hybrid correction. In addition,

parameter	score
match	5
mismatch	-11
gap open reference	-2
gap extension reference	-1
gap open query	-4
gap extension query	-3

Table 6.1.: Hybrid alignment scoring scheme used by proofread with SHRiMP-2.0 and BWA-MEM. Penalty costs are denoted as negative values, as provided to the SHRiMP-2.0 command line interface. For BWA-MEM costs are provided as positive values.

6. *proofread* – large-scale hybrid PacBio correction through iterative consensus

the constraints regarding the scoring scheme are complex. Thus, only mapping software providing comprehensive customization options for alignment parameters can be used.

Originally, two mappers matching the general requirements were considered for use with *proofread*: *bowtie2* (Langmead and Salzberg, 2012) and *SHRiMP-2.0*. While both mappers work with Illumina and PacBio data, only *SHRiMP-2.0* (SHort Read Mapping Package 2.0, Rumble et al., 2009; David et al., 2011) allows for the full implementation of the hybrid scoring scheme (table 6.1) and thus was chosen as default software.

Later, *proofread* was adapted for use of the more recently published *BWA-MEM* (see section 6.3.2 *bwa-proofread* – a modified *BWA-MEM* implementation). *BWA-MEM* (H. Li, 2013) offers comparable sensitivity but considerably outperforms *SHRiMP-2.0* in terms of speed. *BMap* (Bushnell, 2014) was tested as well, yet no improvement over *BWA-MEM* was observed and full implementation was omitted.

6.2.2. Localized score comparison and trusted alignments

Given the highly sensitive mapping parameters, which are necessary to handle PacBio error rates, additional evaluation of the generated alignments is required in order to distinguish valid alignments from spurious ones. Due to sequence similarity, particularly repetitive regions and conserved domains recruit Illumina reads deriving from genomic locations other than the PacBio reads true origin, as observed in preliminary experiments. If mapped to an error free reference, these alignments could easily be detected and filtered based on their non-optimal scores caused by minor differences in the sequences. However, when aligned to PacBio reads with high error rates, the minor differences of similar genomic regions often are lower than local fluctuations of the PacBio error rate.

For example, consider two locations L_a and L_b of different, yet highly similar origin (>97% identity), and with two PacBio reads p_a and p_b obtained from those locations. Let p_a have an error rate of 10% and p_b an error rate of 20%. If short reads from both locations are aligned to both PacBio reads, even alignments of Illumina reads derived from L_b are likely to have higher scoring alignments with the p_a than with p_b , as the total sum of errors plus biological differences to p_a is smaller than the amount of differences caused by errors in p_b . The local error rate of the PacBio read reshapes local score optima and, thus, obscures biological differences if alignments are assessed in a global context. Nevertheless, if both locations are evaluated independently, the highest scoring alignments for p_a and p_b consistently are obtained from L_a and L_b derived Illumina reads, respectively. In the local context, the biological differences are sufficient to allow the distinction of valid and spurious alignments.

To facilitate the comparison and evaluation of alignments in local context, PacBio reads are represented by *proofread* as a series of consecutive bins (section 6.4.3). For the common case of 100 bp reads, the default bin size is 20 bp. Alignments are allocated to a bin by the coordinates of their center. Alignments within the same bin are aligned in the same local context and hence, their scores can be compared directly. Only the

best scoring alignments within a bin, up to a coverage cutoff chosen with respect to the expected sample coverage, are considered for the consensus computation.

6.2.3. **Optimized speed and sensitivity through an iterative mapping procedure**

Initially, *proofread* was implemented as an iterative correction procedure primarily for the sake of optimizing run-time while maintaining high overall sensitivity. The underlying math is closely connected to the utilized mapper *SHRiMP-2.0*. The distribution of errors along PacBio reads varies a lot between windows as small as the 100 bp Illumina reads used in alignments. Maximum sensitivity of alignments, however, is only required for a minor fraction of the full read set, yet sensitivity exponentially increases run-time. The overall alignment process was sped up substantially by implementing an iterative approach with increasing sensitivity with each cycle and in combination with masking of already processed regions. The following reasoning applies: Initially, Illumina reads are aligned at medium sensitivity, sufficient for the majority of PacBio read regions, yet several times faster than with sensitivity settings required for erroneous regions. After the iteration, regions with alignments are masked by replacing respective nucleotides in the sequence with ‘N’s. Essentially, this reduces the size of the reference prior to the next iteration significantly (depending on the actual sensitivity settings and the particular sample). Subsequently, a more sensitive, slower iteration is run on the pre-masked reference to address regions of higher error rates. Since the run-time for *SHRiMP-2.0* scales rather linearly with the size of the reference, but exponentially with sensitivity, running the entire set on medium sensitivity plus a small fraction at high sensitivity in an second iteration, is faster than running everything within a single high sensitivity cycle. The process can be further optimized by using multiple iterations with a step wise increase in sensitivity. An optimal setup based on three iterations has been implemented as default.

Next to the speed-up, the iterative procedure has an additional benefit. Actual alignment computation for most mappers including the aforementioned ones, is triggered by seed matches - usually one to several small regions of absolute identity between query and reference. With the random distribution and local accumulation of errors, seed generation can be impeded at some locations along the PacBio read. By leaving terminal bases of pre-corrected regions unmasked, these regions facilitate seeding, thus increasing the chance of generating alignments across regions of high error rate.

With the introduction of *BWA-MEM* as default mapper and the recommended utilization of longer short reads (*MiSeq* or merged overlapping *HiSeq* reads), the main purpose of the iterative procedure shifted from a means to increase speed to a means for optimizing sensitivity and contiguity. The gain in speed obtained through masking as well as the loss due to increased sensitivity are much smaller for *BWA-MEM* than *SHRiMP-2.0*. Running *BWA-MEM* at high sensitivity from the start is faster than a step-wise increase of sensitivity. However, in particular with longer Illumina reads, (which are handled much more efficiently by *BWA-MEM* than *SHRiMP-2.0*) the closing of gaps by seed-walking

6. *proofread* – large-scale hybrid PacBio correction through iterative consensus

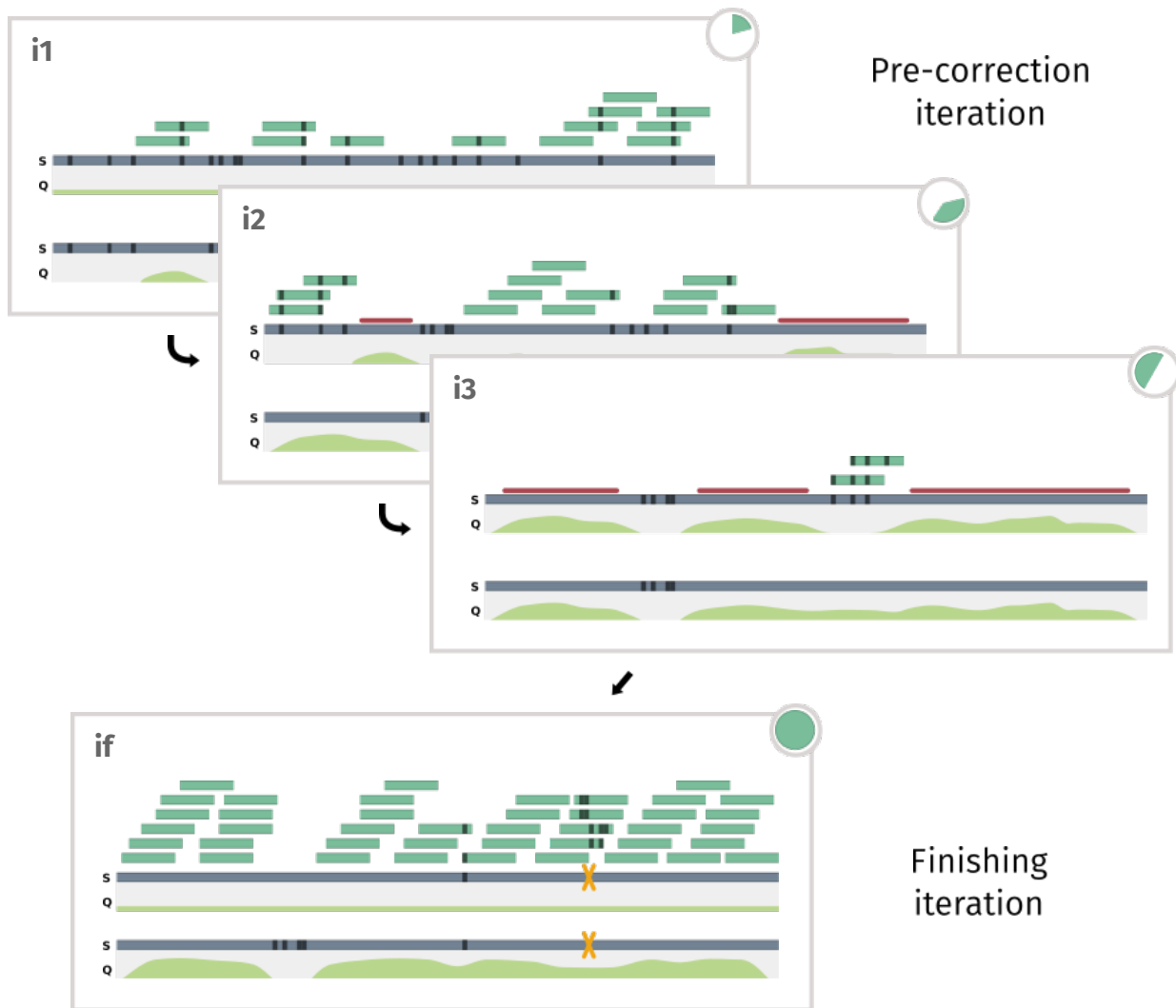


Figure 6.1.: Iterative correction procedure of proofread by short read consensus (Hackl et al., 2014). Complementing subsets of Illumina reads (green bars) with varying coverage (green pies) are mapped onto erroneous PacBio reads (blue bars) in four iterations (i1-i3, if). During initial iterations, reads are mapped with high sensitivity, and sufficiently covered regions of high quality (light green areas) are masked (red bars) prior to the next iteration. Mappings for the final iteration are generated on entirely unmasked sequences with strict settings, to obtain a high-accuracy consensus and annotation of chimera break points (yellow X).

from pre-corrected regions is a much more rewarding procedure. Even with [BWA-MEM](#), running proofread in iterative mode is highly recommended as it produces superior results.

As an additional run-time reduction mechanism, [proofread](#) applies complementary subsampling of the provided Illumina reads for each iteration. In general, coverages of

6.2. *proofread 1.0 – legacy implementation and evaluation*

40 x to 50 x are necessary to ensure that each region of the sample, despite biases and stochastic fluctuations is sufficiently covered. However, even at coverages of 15 x , the majority of the underlying genome is covered. *proofread* by default uses 15 x subsamples for initial iterations. The samples are generated with **SeqChunker** (section 7.3) such that after three iterations, every read of a 45 x set has been mapped at least once. That way, each mapping step can be run three times faster, while potential coverage gaps present in any one of the individual subsets are complemented by previous / subsequent iterations.

In addition to the initial iterations, which are run with sensitive settings, a final iteration on the mostly pre-corrected, unmasked reads and with at least 30 x Illumina coverage is performed with strict and fast settings. The reads obtained from this final consensus call are for the most parts of high quality, however, still comprise regions for which no correction occurred, e.g. due to insufficient short read support or very high error rates. Using a window-based quality filter implemented in **SeqFilter**, these low-quality regions are detected and trimmed from the reads. Both, trimmed and untrimmed reads are reported by *proofread* as output, and can be utilized for further analyses as the user sees fit.

6.2.4. High flexibility through scalability and parallelization

Due to the binning of alignments and filtering based on local score context, the correction of individual PacBio reads with *proofread* is independent. Therefore, PacBio data sets can be split into chunks of arbitrary size without affecting correction quality. At the same time the amount of required memory and the total run-time for a particular chunk of long read data is directly determined by its size. By adjusting chunk size and threads, run time and hardware requirements of individual *proofread* jobs can be matched to a large spectrum of different machines, including desktop PCs, cheap low memory nodes as well as powerful HPC nodes. This allows for an effective distribution of *proofread* jobs on different cluster architectures and a high level of parallelization.

6.2.5. Benchmark of *proofread* and comparison to other tools

An extensive assessment of correction quality indicating criteria of *proofread* v1.01 as well as the previously released hybrid correction tools PacBioToCA (included in Celera v7.0) and LSC (v0.3.1) was generated for and analyzed in the *proofread* publication (Hackl et al., 2014). The obtained results are shown in fig. 6.2. Performance of the tools was evaluated in four categories – accuracy, output, N50 and run-time – on three genomic sets with varying size (*E. coli*, *A. thaliana* and *H. sapiens*) and one transcriptomic read set (*H. sapiens*). For the large genomic *H. sapiens* as well as the transcriptomic set, next to a default *proofread* run, also a run with a custom-tailored setup was included in the study. In case of the genomic read set, the performance of *proofread* with a digitally normalized (section 4.4) short read data set was tested. On the transcriptomic data

6. proovread – large-scale hybrid PacBio correction through iterative consensus

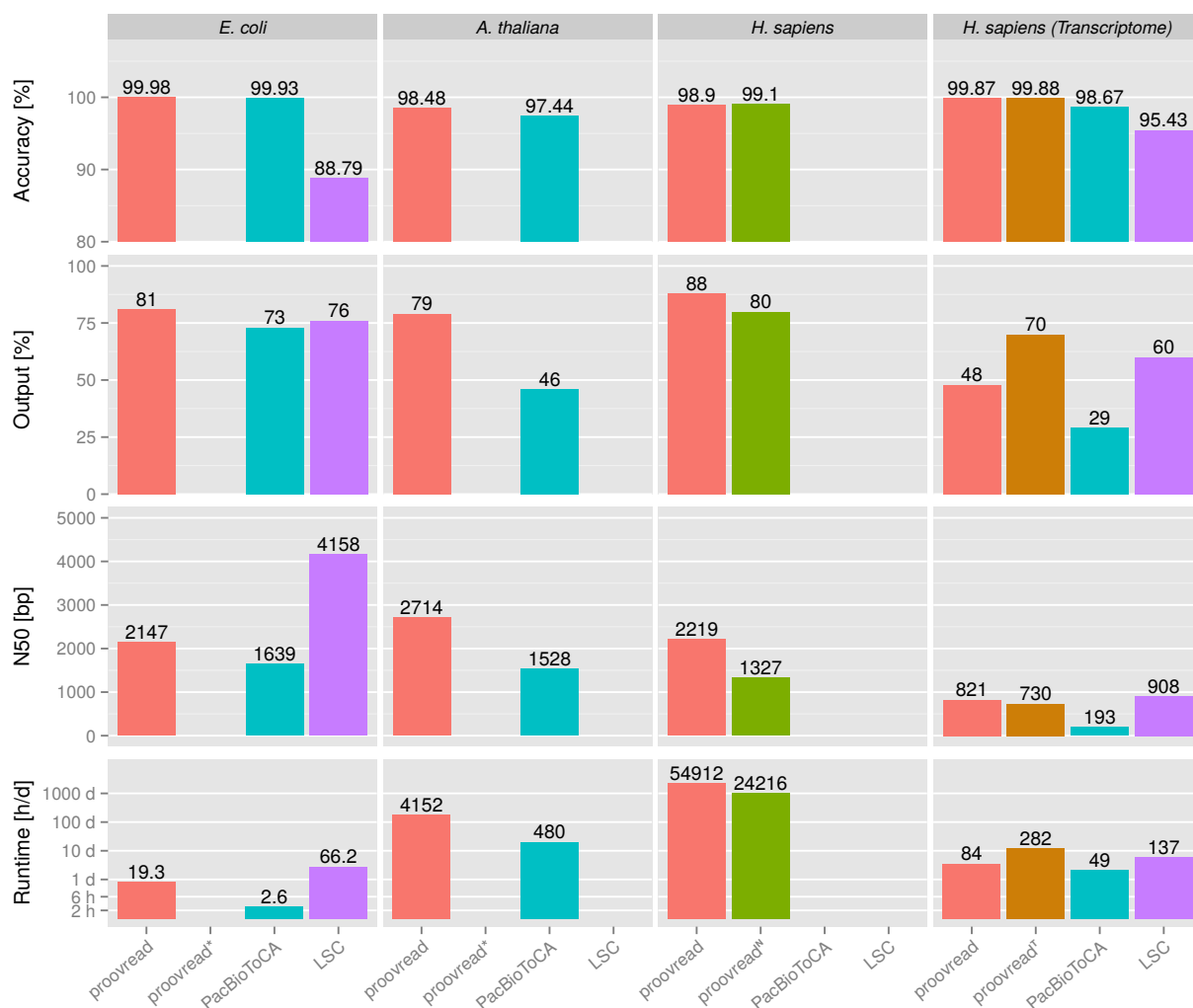


Figure 6.2.: Benchmark of hybrid corrections performed on three genomic (*Escherichia coli*, *Arabidopsis thaliana*, *Homo sapiens*) and one transcriptomic read set (*H. sapiens* Transcriptome) with proovread (red), PacBioToCA (blue) and LSC (purple). For the large human genomic set an additional run with normalized short read data (green) and for the transcriptomic set a run with increased sensitivity for low abundance data (brown) were analyzed. Results were assessed in categories *accuracy*, *relative output*, *N50* (Burton, 2008) and *runtime* and are indicated by the height of the respective bar with exact values plotted on top.

set, proovread was reconfigured to obtain higher sensitivity for reads derived from low abundance transcripts.

The correction of the genomic *H. sapiens* set was carried out on a cluster infrastructure to acquire sufficient computational power. Unfortunately, neither PacBioToCA (software incompatibility with SLURM (Simple Linux Utility for Resource Management) queuing engine) nor LSC (excessive memory requirements) could be used on this infrastructure and hence, no correction results for these tools could be obtained for comparison to the

6.2. *proofread 1.0* – legacy implementation and evaluation

performance of *proofread*.

Across all test data sets, *proofread* achieved highest accuracies, ranging from 99.48 % (*A. thaliana*) to 99.98 % (*E. coli*). *PacBioToCA* obtained slightly lower values, ranging from 97.44 % (*A. thaliana*) to 99.93 % (*E. coli*). Accuracies of the two read sets corrected with *LSC* were substantially lower (88.79 %, 95.43 %). However, the correction process of *LSC* differs from *proofread* and *PacBioToCA* in one important aspect: *proofread* as well as *PacBioToCA* apply a post-correction trimming to the reads thereby removing un- / poorly corrected regions from the data. This trimming step is missing in the *LSC* procedure. Therefore, *LSC* corrected reads comprise both, corrected as well as uncorrected regions affecting overall read accuracies.

proofread also retained the highest fraction of input data after correction for all genomic data sets (79 % to 88 %) and – after parameter customization accounting for low abundance transcripts – also for the transcriptomic set. On the *E. coli* set, *PacBioToCA* and *LSC* performed similar, on the *A. thaliana* set, *PacBioToCA* only recovered 46 % of the input data. On the transcriptomic set, *PacBioToCA* performed worst (29 %), while *LSC* with 60 % output outperformed *proofread* with default settings.

Similar to accuracy, also the N50 length of the corrected reads need to be interpreted with respect to the missing trimming step in *LSC*. For *LSC*, differences in N50 derive from the general shrinkage of corrected reads compared to uncorrected ones due to removal of insertions, as well as the removal of complete reads, which potentially skews the overall read length distribution. Therefore, *LSC* N50 values are close to the raw N50s and provide limited information regarding correction quality. For *proofread* and *PacBioToCA*, on the other hand, low N50 values indicate a higher level of fragmentation of the read data caused by trimming of internal read regions. With *proofread* and default settings, comparably high N50s ranging from 2,147 bp to 2,714 bp on genomic and 821 bp on transcriptomic reads were obtained. The utilization of normalized read data, however, resulted in a much higher level of fragmentation indicated by a strong decrease of the N50 (1,327 bp). With *PacBioToCA*, substantially lower values than with default *proofread*, ranging from 1,528 bp to 1,639 bp for genomic data and 193 bp for transcriptomic reads were observed.

For all sets but the genomic *H. sapiens* set, *PacBioToCA* was the fastest tool. *LSC* was the slowest program on the *E. coli* set, slower than *proofread* on the transcriptomic set, yet faster than the transcriptome-optimized *proofread*. On small sets *proofread* was between two and four times slower than *PacBioToCA*, on the larger *A. thaliana* set almost ten times. The utilization of normalized reads on for the genomic *H. sapiens* set reduced the overall run-time by more than two fold.

Overall, the benchmark revealed that *proofread* achieves high accuracy as well as sensitivity on the entire range of assessed data sets. Further, *proofread* outperforms existing software in terms of accuracy, as well as output and contiguity. Decreased speed compared to *PacBioToCA* is compensated for by the more flexible implementation that allows processing of large data sets on cluster architectures.

6.3. Upgrades and extensions to the proofread pipeline

6.3.1. Storage-efficient best alignment filter through progressive minimum score adjustment

The reads obtained from a sequencing run are because of the random shot-gun sequencing procedure entirely unordered with respect to their true location on the sequenced reference genome. Yet, in order to determine the highest scoring alignments of a bin, all alignments of a bin have to be compared against each other. For unsorted input, this is only guaranteed, if the entire read set has been analyzed. For large data sets, the total amount of alignments cannot be kept in memory, therefore the most straight-forward approach is to first, compute and store all alignments to disk, than sort the data and in a second pass process each bin and apply the filter. The major disadvantage of this exhaustive approach is that also the majority of sub-optimal and spurious alignments, which are exponentially more for repetitive data sets, have to be saved to disk and on top sorted. For gigabase-sized projects, this is impracticable or even infeasible.

Therefore, for proofread I developed a *progressive* two pass filtering approach that is particularly effective in reducing sub-optimal alignments caused by repetitive reads. Here, alignments are already assessed on-the-fly during generation. In the beginning, the scores of all alignments are memorized for each bin, and the alignments are written to disk. Once the maximum amount of alignments for a particular bin given by the coverage cutoff is reached, new alignments are only memorized and written to disk if they score better than the alignments already present in the bin. Every time a higher scoring alignment is added to a full bin, the lowest scoring alignment is removed. This, however, is only possible for the scores, which are kept in memory. The actual alignment already written to disk remains. Nevertheless, with this procedure, the required minimum score for each bin is progressively raised, converging to its true value. While in the beginning all alignments are kept, later only alignments closed to the local optimum are considered. This pre-filtered and thus smaller set can be sorted and cleaned of sub-threshold alignments much faster.

$$(6.1) \quad E[N] = n_c + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} \quad \begin{array}{ll} n: & \text{Number of observed scores} \\ n_c: & \text{Number of best scores} \\ E[N]: & \text{Expected number of stored scores} \end{array}$$

$$(6.2) \quad E[N] < n_c + \ln(n)$$

The convergence of the progressively adjusted minimum score cutoff can be modeled as stochastic process. The expected number of stored scores can be computed according to equation 6.1, or for large numbers of observed scores approximated according to equation 6.2¹. Figure 6.3 shows the number of stored scores as function of the number of observed

¹Special thanks to Johannes H. for helping me figuring out the math for this elegant solution

6.3. Upgrades and extensions to the proofread pipeline

scores, for exhaustive as well as the precise and approximated progressive approaches, in the simplest case of one alignment per bin. While for the the exhaustive approach, the expected number of stored scores is correlated linearly with the number of observed scores, for the progressive method, the correlation is logarithmic, resulting in a drastic decrease of the relative amount of actually stored scores for high numbers of observed scores.

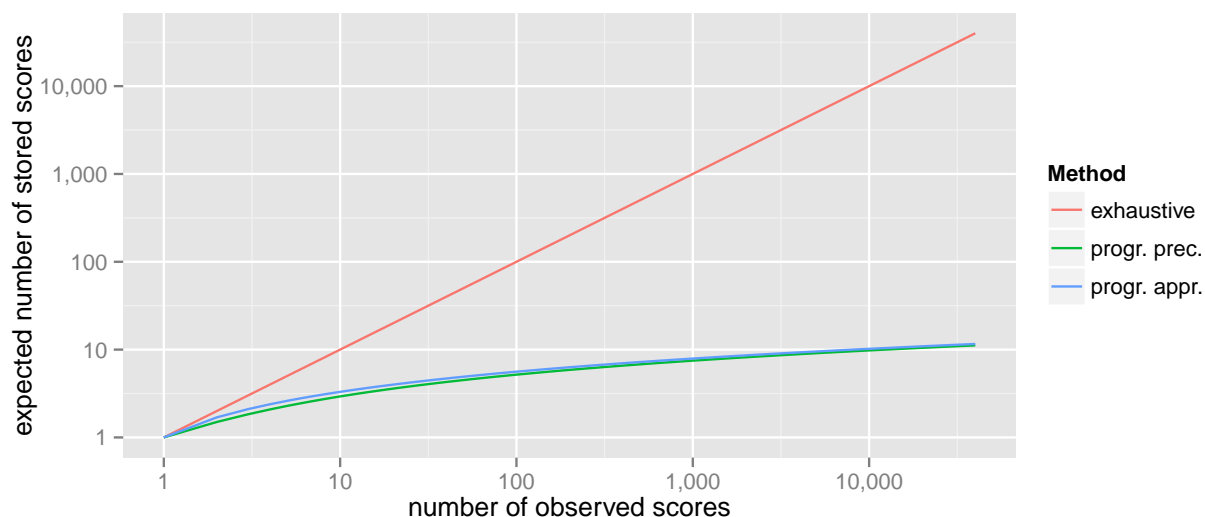


Figure 6.3.: Storage-efficiency of progressive minimum score adjustment alignment filter. The number of expected scores required to be stored are plotted as a function of the number of observed scores, without (red) and with progressive adjustment computed precisely (Eq. 6.1) and approximately (Eq. 6.2).

The impact of the progressive filtering was also assessed directly on *D. muscipula* data. Merged LGC (LGC Genomics) Illumina libraries were mapped onto a 5 Mbp sample of *D. muscipula* raw PacBio reads, with and without filter. Without filter, the obtained BAM (Binary SAM, SAM/BAM Format Specification Working Group, 2015) file containing the alignments had an approximated size of 300 GB (the run was cancelled after one fourth of the data had been mapped with a file size of 77 GB), with filter the file had a size of only 367 MB. These results suggest that for the repetitive *D. muscipula* read set, about 99.9% of required disk space can be saved through progressive minimum score adjustment and filtering.

Initially, the procedure was implemented as a Perl-based parser, through which plain mapper output was piped and selectively stored to disk. This implementation, however, was inefficient and produced a bottleneck, particularly when used with multi-threaded mappers. With the introduction of BWA-MEM, I thus implemented the progressive filtering directly into the C (Programming language) source code of the mapper itself (see section 6.3.2 bwa-proofread – a modified BWA-MEM implementation), resulting in a multi-thread ready, highly efficient filter procedure.

6.3.2. *bwa-proofread* – a modified BWA-MEM implementation

In order to optimize BWA-MEM for the use with *proofread*, two key modification of the published source code have been introduced. First, to facilitate score assessment, the default reporting and filtering mode for alignment scores was changed from an absolute to a length-relative system. That way, reads of different length can be filtered with a single, comparable cutoff.

I/O	
-T INT	PER BASE minimum score 0.50
Binning	
-b	bin size 20
-l	max basepairs per bin 100

Table 6.2.: Customizations of the BWA-MEM command line interface in *bwa-proofread*.

Second, the basic structure holding the compressed nucleotide reference (`bnt_seq_t`) was extended to be able to also hold the best scores for each bin in a sorted, memory efficient and quickly accessible form. The extension was implemented as an array of double linked lists using `TAIL_QUEUES` macros provided by the FreeBSD `QUEUE` library (The FreeBSD Project, 2016). Thread safety, was achieved by declaring mutually exclusive access to individual queues (Dijkstra, 1965; Taubenfeld, 2004). The

memory footprint of the implementation scales linear with both, size of the indexed genome as well as the maximum number of stored alignments per bin. Given 20 bp bins, 100 bp reads and a target coverage of $15 \times$ (300 bp per bin), length, score and two pointers of the double linked list need to be stored for three alignments in each bin. Embedded in an array structure and accounting for mutex flags, this computes to roughly 20 B per bin, which is equal to 1 B per 1 bp. This ratio is similar to that of the regular *bwa* memory structure, which comprises a 2-bit compressed representation of the genome and the index obtained through *Burrows-Wheeler transform* (Langmead et al., 2009; H. Li and Durbin, 2009; R. Li et al., 2009). Thus, the overall memory footprint of *bwa-proofread* is in the order of two times the footprint of the standard BWA-MEM implementation. Customization options for the binning algorithm were added to BWA-MEM’s command line interface (table 6.2).

6.3.3. Redundancy reduction through circular consensus computation

PacBio SMRT sequencing uses circularized DNA templates branded SMRT-bells. These templates are generated from linear double stranded molecules through ligation of a single-stranded adapters to both ends of both strands at each side of the molecule (fig. 6.4). Due to the circularization, sequencing is not terminated once the polymerase has processed a template one time, but will continue until either the polymerase stops working due to laser-induced damage or until the run itself is terminated. Therefore, a read can contain one or multiple copies of its template, referred to as subreads, but present in alternating orientation and interspersed with adapter sequences (`forward-adapter-reverse-adapter-...`). The number of subreads per read is primarily determined by the initial size of the template. Shorter templates can be sequenced more often within the same time.

6.3. Upgrades and extensions to the proovread pipeline

Processing of raw PacBio read data including adapter clipping and splitting into subreads is usually performed along side base calling by the software of the sequencing platform. The subsequent analysis of PacBio read data in general is based on subreads, which for the sake of simplicity are in this work commonly referred to as PacBio reads.

With respect to correction, the existence of multiple subreads has two key implications: subreads of the same template comprise redundant information and independent correction of the copies unnecessarily increases run-time. At the same time, because of the random distribution or errors, alignment of multiple subreads can be exploited to uncover errors in individual copies and compute a more accurate representative consensus.

The `ccseq` module of proovread has been designed to address both of these issues. It identifies subreads derived from the same template by parsing the corresponding information from the sequence header, aligns the fragments against each other using `BWA-MEM`, and generates a single representative consensus for each processed template. In contrast to the circular consensus algorithm provided with the `SMRT-Portal` software, the `ccseq` module also infers consensus from two-fold covered and only partially overlapping reads, facilitated by a quality-weighted consensus procedure (section 6.4.3 State matrix). The module was introduced to the proovread pipeline with release v2.01 and is run as first step.

6.3.4. Advanced trimming of undetected SMRT-bell adapters

In rare cases, the proper detection and clipping of adapters fails during initial processing, e.g. if adapter sequences are obscured by very high error rates. As a result, PacBio subread data can contain a small fraction (<1%) of reads actually comprising multiple copies of the template in the same alternating, adapter-interspersed conformation as observed for unprocessed data. In typical samples, the vast majority of these reads will comprise less than two complete copies, as DNA templates are selected to be as long as possible and having multiple copies of adapters would facilitate detection by the initial processing software.

Although, the fraction of these chimeric reads is low, their negative impact on assembly quality can be drastic. In the test assembly generated from proovread corrected PacBio data of the giant Virus CroV, without SMRT-bell-chimera trimming, more than 20 additional small contigs and at least 4 additional breaks to large contigs can be observed (fig. 6.5 A). Contig ends at those breaks as well as the small contigs exhibit the typical structure observed for improperly processed PacBio reads with the terminal region being

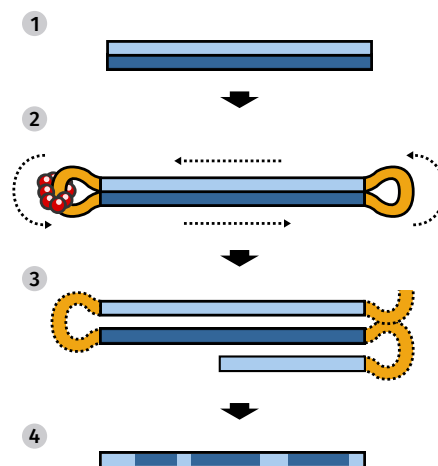


Figure 6.4.: Sequencing of double-stranded DNA template (1) after SMRT-bell circularization (2), resulting in a read with multiple subreads in alternating, adapter interspersed conformation (3), merged into a representative subread consensus (4).

6. proofread – large-scale hybrid PacBio correction through iterative consensus

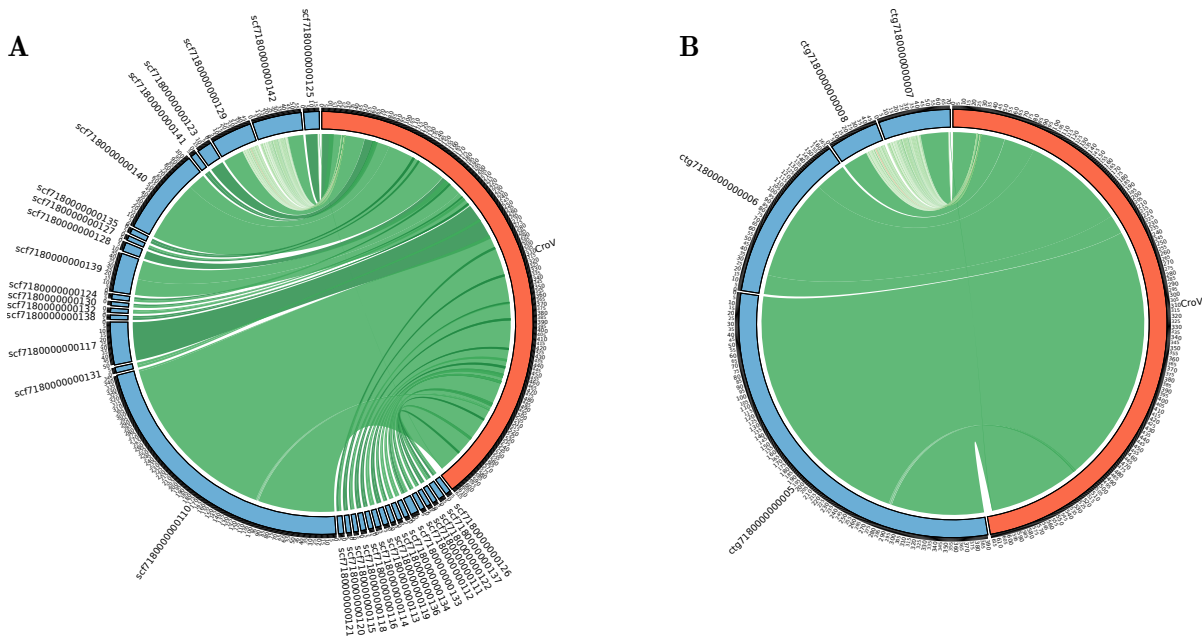


Figure 6.5.: Whole genome alignments of the giant virus CroV (Garza and Suttle, 1995) reference genome (red) against assemblies (blue) generated with Celera from proofread corrected PacBio reads, without (A) and with (B) SMRT-bell-chimera trimming. Colored areas connecting contigs indicate aligned regions with color encoding for the level of identity (dark green: 100 % to red: <70 %).

an inverted copy of its respective upstream / downstream region.

proofread’s SMRT-bell-chimera module (*siamera*) specifically targets this disrupted palindromic structure in order to identify improperly trimmed reads. After the iterative hybrid correction, each read is aligned to itself with BLAST (Altschul et al., 1990) using custom settings (table 4). Only hits on the reverse strand are considered, as these hits derive from palindromic regions. Depending on the level of degradation of the adapter, either a single alignment covering the template, its (partial) reverse complement counter part and the adapter, or two adjacent alignment of the same length, covering reverse complement regions at either side of the non-palindromic adapter can be observed. In case of multiple untrimmed adapters or poor correction, also a more complex layout of alignments is possible. The alignment structure is further assessed and if it can be simplified to a structure with a single adapter the shorter part of the sequence, either up- or downstream of the adapter, as well as the adapter itself are trimmed. If alignments cannot be resolved, by default the entire read is discarded.

By detecting the palindromic signature in corrected reads rather than highly degraded adapters, much higher sensitivity is achieved. However, the approach will also trim genuine palindromic motives if they occur at the terminal region of a read. Yet, with respect to assembly, this is a negligibly small disadvantage. Just like the SMRT-bell-chimeras, also genuine palindromes introduce potentially unresolvable structures in the

6.3. Upgrades and extensions to the proovread pipeline

assembly graph that can result in an assembly break. Only if reads exist that span the entire palindromic structure and link their respective up- and downstream flanks a break can be prevented. Those reads, however, by design do not qualify as potential SMRT-bell-chimeras, as the palindrome they carry, is not terminal. Whether or not genuine palindromes can be resolved in a certain read set is determined by the existence of the respective spanning reads rather than the degree to which partially overlapping reads are trimmed. Hence, trimming of genuine palindromic motifs that fit the structure of SMRT-bell-chimeras does not impair the assembly potential of the processed read data. At the same time, the removal of true positive SMRT-bell-chimeras is highly beneficial (fig. 6.5): After application of SMRT-bell-chimera trimming to the reads set used for the CroV assembly, all small chimeric contigs as well as chimera-related break points disappeared from the assembly (fig. 6.5 B).

The `siamera` filter is executed as part of the proovread pipeline along with quality based trimming as last step of the correction process.

6.3.5. Usability of pre-computed unitigs for correction

The vast amount of high coverage Illumina data and the computational power required for their alignment are a key challenge of hybrid correction approaches. The information contained in the reads, due to their overlaps is highly redundant. This fact can be exploited to reduce the amount of data required to be mapped and thus substantially speed up the entire process: Rather than aligning individual reads to erroneous PacBio reads, the reads can be assembled into unitigs first, creating non-redundant and longer sequences of high accuracy. Then, these sequences are mapped to PacBio reads at largely reduced computational costs.

While in theory, the usage of contigs or scaffolds derived from short read assembly, is possible, the utilization of unitigs is recommended. Unitigs represent the most complete, non-redundant sequence set that unambiguously can be constructed from short read data, while contigs and scaffolds usually derive from processed graphs, with potentially collapsed and trimmed regions.

For the alignment of sequences longer than typical Illumina reads, proovread uses BLASR (Chaisson and Tesler, 2012) or DALIGNER (experimentally) instead of less sensitive BWA-MEM. While BLASR directly outputs alignments in SAM (Sequence Alignment/Map, SAM/BAM Format Specification Working Group, 2015) format, DALIGNER output is converted with a custom script (`dazz2sam`).

The unitig correction is implemented in proovread as independent, optional step, which by default is run prior to read based correction using previously generated unitigs as additional input. The consensus procedure for aligned unitigs is similar to the one used for short reads, however, differs in some key aspects: By definition, for every region of a read, there can only exist one correct unitig representing the same genomic location. Hence, the maximum coverage for unitig consensus is $1x$. While short reads are assessed and

6. *proofread* – large-scale hybrid PacBio correction through iterative consensus

filtered based on scores, the best unitig for each region is determined by a more complex approach that takes accounts for their longer and varying lengths (see section 6.4.3, section 6.4.3) as well as the low coverage using a quality-weighted consensus matrix (section 6.4.3).

Test runs generated with the unitig mode on different sample data sets (data not shown) revealed that in the case of small genomes (<100 MB) results of higher quality could be generated in shorter time. On more complex genomes, however, the increased amount of ambiguities resulted in fractions lower than <50 % of pre-masked read regions after unitig correction. The speed-up of the subsequent read-based iterations was, thus, not as high as for smaller genomes, and taken together with the unitig iteration, did not result in significant overall improvement of the run-time of the entire correction procedure. On top, results obtained with unitigs were in terms of quality and contiguity at best comparable to short read only corrections, in some cases even worse. Therefore, the *proofread*'s unitig mode is a useful extension to the pipeline for the correction of small genomes, however, it is not recommended for large data sets.

6.3.6. Optimized correction for transcriptomic and metagenomic data

proofread has been designed with primarily genomic sequencing data in mind. A key assumption for the coverage based best alignment filtering is, that the coverage distribution of the sample is even, or that at least every location has been sequenced with the minimum depth of coverage. While this assumption in general holds true for genomic samples (biases can disturb coverage distribution), it fails on transcriptomic and metagenomic data sets. The particular problem of correction of low coverage regions, besides a genuine lack of alignments, is the distinction of valid and spurious alignments if both are present in the below-threshold coverage alignment fraction. As soon as spurious alignments are in the majority, the consensus will be driven towards a false result. This effect, for example, can be observed for low abundance transcripts sharing conserved regions with other high abundant transcripts. Similarly, correction of low abundance species in metagenomic samples is affected by the presence of closely related species with higher abundances.

proofread-meta is an experimental proof-of-concept implementation that takes on the challenge of correcting data that includes low coverage read populations. In contrast to *proofread* with a universal coverage cutoff, *proofread-meta* tries to determine a reads true coverage and dynamically adjusts the cutoff accordingly. The estimation of a reads true coverage is based on discriminating SNPs (Single nucleotide polymorphisms) as described in section 6.4.3 Haplotype coverage.

6.3. Upgrades and extensions to the proovread pipeline

To benchmark the performance of `proovread-meta`, the program was run with a mixed set of simulated Illumina reads generated from *E. coli* strains K-12 and REL. Illumina coverage was set to 10x for *E. coli* REL and 30x for *E. coli* K-12. Correction was only applied to PacBio reads simulated from *E. coli* REL. After correction with `proovread-meta`, the reads could mapped back with very high identity of 99.8% to the *E. coli* REL genome, but only with an identity of 97.0% to *E. coli* K-12. This is in congruence with the average distance between both strains. In contrast, processing of the reads with `LoRDEC` results in a much lower identity of 96.2% for alignments to the *E. coli* REL genome, as well as a poor rate of 97.1% for alignments to the *E. coli* K-12 genome. While `proovread-meta` is able to properly distinguish both species and produce an unbiased consensus reflecting the true origin of the erroneous PacBio reads, `lordec` corrected reads are hybrid consensus sequences that neither match the original nor the higher covered genome at high accuracy.

The applicability of `proovread` to metagenomic data sets was further explored in a collaboration with Beate Slaby² for metagenomic study on the demosponge *A. aerophoba*. Originally, samples were sequenced and assembled by the JGI. Later complementing PacBio data was generated with the goal to improve assembly quality. Preliminary test for `proovread` correction and assembly of metagenomic PacBio data were carried out on simulated data (section 8.6) generated based on the organismic composition used for the evaluation of the metagenome assembler `Omega` (Haider et al., 2014). Correction with `proovread-meta` resulted in an overall read accuracy >99.99% for the 200 MB set according to the assessment of `BBMap` alignments.

In the following, the *A. aerophoba* data set was corrected with the same setup of `proovread-meta` used for the `Omega` set. After correction an Illumina-only and a Illumina-PacBio hybrid assembly were constructed with `SPAdes` and assessed in comparison to the assembly

	Ec-REL [%]	Ec-K-12 [%]
<code>proovread-meta</code>	99.8	97.0
<code>LoRDEC</code>	96.2	97.1

Table 6.3.: Accuracies of corrected PacBio reads simulated from *E. coli* REL. The Illumina reads were provided as a mixed set of 10x *E. coli* REL and 30x *E. coli* K-12. PacBio reads were corrected with `proovread-meta` and `LoRDEC` and aligned with `BBMap`.

Metrics	aa-so-jgi	aa-sp	aa-sp-hybrid
Contigs ≥ 0 bp	662,400	262,763	59,859
Contigs ≥ 1 kbp	125,969	104,868	29,744
Length ≥ 0 [bp]	657,215,853	584,430,936	317,514,008
Length ≥ 1 k[bp]	419,980,954	490,586,800	301,207,972
Longest [bp]	334,550	1120,532	1117,843
N50 [bp]	1916	6417	32,477
GC [%]	57.2	57.8	60.5
N [%]	0.0	0.1	0.1

Table 6.4.: Assembly metrics of *Aplysina aerophoba* metagenomes generated with SOAP (SOAP-denovo2, R. Li et al., 2009) and Illumina reads by the JGI (Jount Genome Institute), generated with `SPAdes` (Bankevich et al., 2012) and Illumina read and generated from Illumina in combination with `proovread-meta` corrected PacBio data. Contigs: number of Contigs or Scaffolds; Length: length of the total assembly; Longest: length of longest contig/scaffold; N50: length of the sequence that together with all longer sequences adds up to 50% of the total assembly length; GC: percentage of nucleotides G and C relative to A,T,G and C (N-nucleotides are ignored); N: percentage of N-nucleotides in the total assembly.

²GEOMAR Helmholtz Centre for Ocean Research, Kiel; University of Wuerzburg, Botany II at the start of the collaboration

6. *proofread* – large-scale hybrid PacBio correction through iterative consensus

generated by the JGI (table 6.4) using SOAP. With an N50 of 32.4 kbp, compared to 6.4 kbp for the Illumina-only SPAdes assembly and 1.9 kbp for the JGI assembly, the hybrid strategy proved to be highly superior with respect to contiguity of the produced assembly. Whether or not the decrease in total assembly size (-28% / -39%) can be attributed to a loss of data or rather a better resolution of heterogeneity is currently under investigation. Nevertheless, the results obtained in this study so far, strongly suggest that *proofread*-meta can be used to efficiently perform hybrid correction for metagenomic data, and that utilization of such data for assembly has the potential to greatly improve assembly contiguity.

6.3.7. Haplotype awareness through variant calling and read-backed phasing

Heterozygosity is a common issue when dealing with eukaryotic sequence data that adds an additional level of complexity to hybrid correction. Variable sites introduce noise, which complicate accurate consensus computation. Further, SNVs (Small nucleotide variations) can be present in both Illumina and PacBio read sets, and depending on the experimental design, these sets not necessarily match each other.

In general, there are two different desired behaviors for how SNVs should be handled during correction: Depending on the goals of a particular project, variant sites present in the data after correction either need to be considered detrimental, e.g. during construction of an assembly graph, or they are of special interest, e.g. in haplotype specific trait analysis, such as reconstruction of a di- / polyploid genome structure or GWAS (Genome-wide association study). In the first case, hybrid correction ideally should remove ambiguous sites consistently among all reads by always using the same base(s) for a particular locus. In the latter case, variable sites need to be preserved exactly as present in PacBio read.

With the *proofread* default correction procedure, the fate of SNVs is not explicitly addressed, yet the outcome can be steered by the setup of the correction run. *proofread* consensus is based on majority vote, therefore if different variants are present for a specific locus in the best alignment set, the most abundant will be chosen. By lowering the coverage cutoff for the correction to a value close the coverage of a single haplotype (or by increasing the total amount of provided short reads), the set of best alignments will be driven towards comprising mostly reads with the same variant as the underlying PacBio reference. Thus, applying individual haplotype-coverage as cutoff the correction will retain most of the heterozygosity shared between the Illumina and PacBio set. Raising the coverage cutoff, on the other hand, will have the opposite effect. If all available variants are considered at each locus, the most abundant variant will dominate the consensus calls regardless of the state observed in the PacBio read and heterozygosity in the corrected read set will be reduced. Nevertheless, both approaches only allow to set a general trend for individual SNV processing. Evaluation of different test runs (e.g. fig. 6.7) indicates that only in 75% to 85% of the cases the desired variant state is chosen for a specific locus. In order to obtain more consistent results additional mechanisms

6.3. Upgrades and extensions to the proofread pipeline

had to be introduced.

However, before dealing with the actual inference of proper SNV states, a more general problem related to SNVs, and also observed in the test data, needed to be addressed. After proofread correction, for most variable sites, one of the states present in the short reads set is also chosen in the consensus. For a significant fraction, however, a new false state is reported. This is caused by the particular consensus implementation of `proofread`, which assesses each position of the reference independently and in case of equal frequencies, chooses one state randomly³. The effect is further enhanced by the high level of insertions/deletions present in raw PacBio reads. The two most common cases potentially leading to erroneous consensus sequences are described in fig. 6.6. In the case of multi-base deletions, even without errors in the PacBio sequence, `proofread`'s consensus algorithm does not guarantee a consistent inference of either one of the true states. Taking into account PacBio errors, false consensus sequences also can result from insertions directly adjacent to SNPs loci. Given an average insertion rate of 10%, two potential target sites (upstream and downstream) and a chance of 25% for each base, the likelihood for a detrimental insertion next to a SNP can be estimated at about 5%. Propagation of the ambiguity along homopolymers further increases the overall likelihood.

To mitigate the effect of heterozygosity-derived consensus errors, an improved refinement step (`refine`) with a SNV-robust consensus mechanism was developed (fig. 6.7-1). The module replaces the default finishing iteration and is also based on an additional mapping iteration with strict settings onto pre-corrected reads. Prior to consensus computation, variants present in the data are called and subsequently stabilized by transforming adjacent sites from independent into combined multi-base compound states of actually observed permutations covering the entire locus (?? ??). That way, introduction of new, false states during consensus computation, as observed for the legacy consensus routine, is prevented.

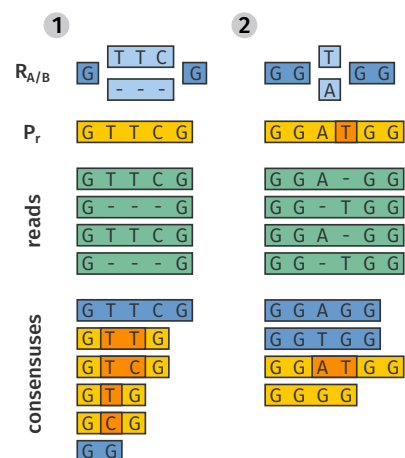


Figure 6.6.: Potential consensus errors introduced by (1) a 3-base indel without adjacent PacBio sequencing error and (2) a PacBio insertion error next to a SNP matching the alternative variant. R_{A/B}: Reference with variants; P_r: PacBio read without and with error (orange); reads: Illumina read alignments; Consensuses: possible correct (blue) and erroneous (yellow) consensus sequences called with equal likelihoods.

³Note that with low default sub-sample coverage of 15x, ties are likely

6. proofread – large-scale hybrid PacBio correction through iterative consensus

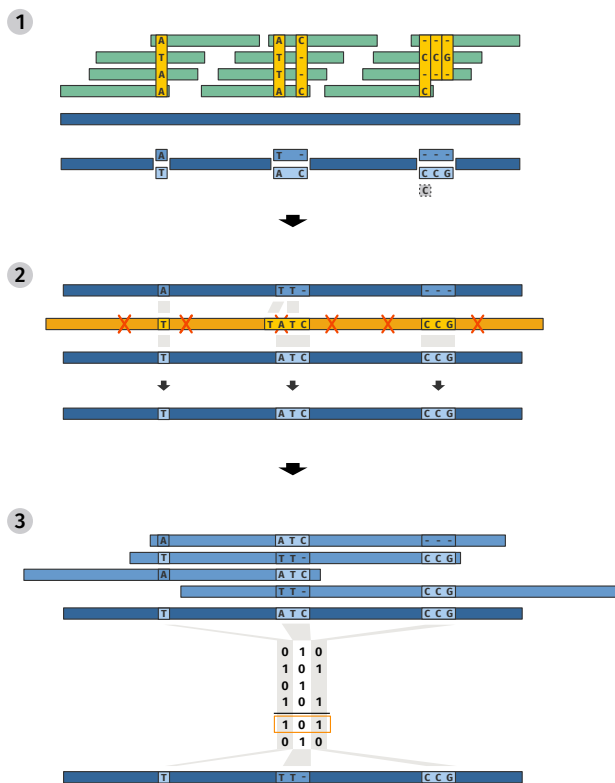


Figure 6.7.: Haplotype aware correction procedure implemented in `proofread` modules `refine` and `polish`. (1) Variants (yellow) are called and stabilized based on Illumina alignments (green) onto the pre-corrected PacBio read (blue). (2) Variants are scored according to alignment of preliminary, complementary consensus sequences back to the raw PacBio read. (3) Other refined reads of the same sample are aligned to the refined PacBio read and miscalled variants are corrected by read-backed phasing.

Further, to facilitate inference of variant states in accordance with the underlying PacBio read a new mechanism for the assessment of SNVs was introduced (fig. 6.7-2). After construction of the consensus matrix and stabilization of variants, two preliminary consensus sequences are generated: one comprising the most abundant states and one containing the second most abundant variants for each variable site. These sequences are then aligned back to the original raw PacBio read. Based on these alignments, possible variants can be directly compared to the respective base(s) in the raw read. In the best case, one of the possible variants will match the respective raw read locus exactly and thus can directly be considered as the mostly correct state. Due to the sequencing errors in PacBio reads, many sites however, will not produce exact matches. In these cases, variants are scored by string similarity based on the algorithm described by Ukkonen (1985) and E. W. Myers (1986) and implemented in the `String::Similarity` module⁴. If scoring is indecisive, and coverage for the variants differ, the variant with a coverage closer to the coverage of the majority of unambiguously scored variants, is chosen. As a last resort, variants are assigned randomly. With this procedure, accurate determination of variants is achieved on test data for 90 % to 95 % of variable sites (fig. 6.8-2). The remaining miscalled sites can be attributed to positions obscured by PacBio sequencing errors and unresolvable by only considering Illumina data and

the individual PacBio read alone. The most common cases include SNPs sites either coinciding with deletions or detrimental insertions, as described above.

Having established an average accuracy of 90 % for variable sites on PacBio reads, further improvement of haplotype resolution can be achieved by exploring the pre-corrected PacBio reads set as a whole (fig. 6.7-3). `proofread`'s `polish` module runs an all-versus-all alignment of corrected PacBio reads and identifies variable sites by superimposition of variant annotations generated from Illumina data. In this setup, miscalled variants then can be identified and corrected by read-back phasing. To that end, a greedy, heuristic, read-backed phasing algorithm inspired by Levy et al. (2007) and further optimized for the use with long read data was implemented (section 6.4.4). During phasing, the likelihood

⁴Marc Lehmann, <http://search.cpan.org/~mlehmann/String-Similarity-1.04/Similarity.pm>

6.3. Upgrades and extensions to the proovread pipeline

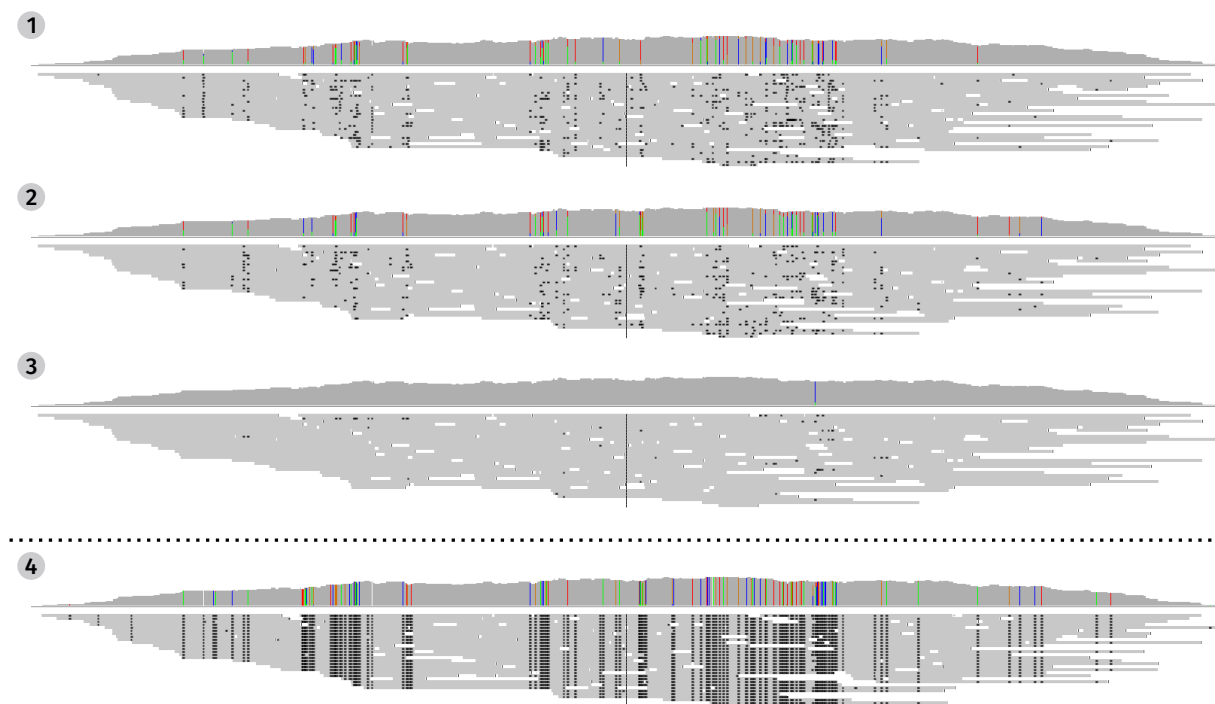


Figure 6.8.: Alignments of simulated PacBio reads to a 30 kbp *A. thaliana* reference genome segment corrected with a heterozygous Illumina set with $10x$ coverage for the correct and $30x$ coverage for the complementary haplotype. Corrections have been carried out with (1) proovread-legacy, (2) proovread + refine, (3) proovread + refine + polish and (4) low coverage variant insensitive PBcR. Visualization was created with IGV (Integrative Genomics Viewer, Robinson et al., 2011). For each panel, the top histogram represents per base coverage with variable sites and frequencies of the observed variants indicated by colored columns. Grey bars in the lower part represent corrected PacBio reads with mismatches and indels indicated by black dots.

of a sequence of variants occurring together as observed in the read at hand, is assessed by computing a Boolean support matrix for each position according to other aligned refined PacBio reads. On perfect data, with two haplotypes present, the alignment of a read of the same haplotype will consistently result in matching variants, while the alignment of a read of the complement haplotype results in mismatches for each variable site. If, however, a miscalled variant is present in the reference read, matches and mismatches with respect to other reads will be inverted at this position. The noise caused by miscalled variants present in the query reads is compensated for by having multiple reads aligned at each location.

On test data with high PacBio coverage, accuracies of $>99.9\%$ for variable sites were observed for corrected reads after refinement and subsequent polishing (fig. 6.8-3). However, the current read-backed phasing `polish` module is an experimental proof-of-concept implementation. In particular, the all-versus-all alignment procedure in its current form is inefficient with respect to run-time and renders application of the program to larger data sets infeasible. Nevertheless, the results obtained so far demonstrates the high

6. *proovread – large-scale hybrid PacBio correction through iterative consensus*

potential of the approach towards haplotype-aware and -consistent hybrid correction.

6.4. perl5lib-Sam – a perl API to read alignment data

The basis of *proofread*'s correction procedure is the evaluation of sequence alignments generated by different mapping programs. Most mappings programs including the aligners used within *proofread*, except for *DALIGNER*, report result in the de-facto standard SAM format. Thus, the ability to parse and process SAM data efficiently is a key prerequisite to the implementation of consensus-based read correction.

The SAM format is text-based TAB-separated file format designed for storage of large numbers of nucleotide alignments against a set of reference sequences. Optional header lines are identified by an '@' as first line character. The header section is further structured by a combination of line tags and tagged fields, comprising information about reference sequences, sample groups, programs used to produce the alignments and other general information. The header section is followed by the alignment section. An alignment is represented by a single line comprising eleven mandatory (see table 3) and an arbitrary number of optional fields further characterized by tags.

Listing 6.1: sample SAM file⁵

```

1 @HD VN:1.5 S0:coordinate
2 @SQ SN:ref LN:45
3 r001 99 ref 7 30 8M2I4M1D3M = 37 39 TTAGATAAAGGATACTG *
4 r002 0 ref 9 30 3S6M1P1I4M * 0 0 AAAAGATAAAGGATA *
5 r003 0 ref 9 30 5S6M * 0 0 GCCTAAGCTAA * SA:Z:ref,29,-,6H5M,17,0;
6 r004 0 ref 16 30 6M14N5M * 0 0 ATAGCTTCAGC *
7 r003 2064 ref 29 17 6H5M * 0 0 TAGGC * SA:Z:ref,9,+,5S6M,30,1;
8 r001 147 ref 37 30 9M = 7 -39 CAGCGGCAT * NM:i:1

```

The BAM format is the compressed, binary equivalent of the SAM format. SAM to BAM conversion and vice versa are provided by the *samtools* (H. Li et al., 2009) and *sambamba* (Tarasov et al., 2015) utilities. Further, sorting and indexing of BAM files allows random access to alignment subsets by reference coordinates. Thus, BAM files can be used as a simple database-like solution for alignment information storage and retrieval.

To allow efficient processing of SAM data, I developed the Perl-based *perl5lib-Sam* library as an object-oriented API (Application programming interface) to SAM/BAM encoded mapping data. The library provides basic parsing and filtering functionalities as well as complex methods required for *proofread*'s binning-based filter algorithm and consensus computation.

⁵modified from the *Sequence Alignment/Map Format Specification*, SAM/BAM Format Specification Working Group, 2015

6.4.1. Sam::Alignment

The `Sam::Alignment` class provides a flexible interface to individual SAM encoded alignment records. In addition to generic accessor methods (retrieval and manipulation of primary field data) and Boolean tests against the bit-wise flag (see table 3), the interface offers access to derived information. These include alignment length, full length of the (partially) aligned sequence, a representation of the read sequence transformed into reference coordinate space and different alignment score derivatives.

Score, nscore and ncscore

Alignment scores are the fundamental units of measurement for alignment quality assessment. While the score of an alignment is not a mandatory attribute in SAM format, most mappers report it as optional value in the AS tagged field. The reported score S is a raw score derived from the mappers scoring scheme. In addition to this raw score the `Sam::Alignment` class provides access to two derived scores: S_n and S_{nc} .

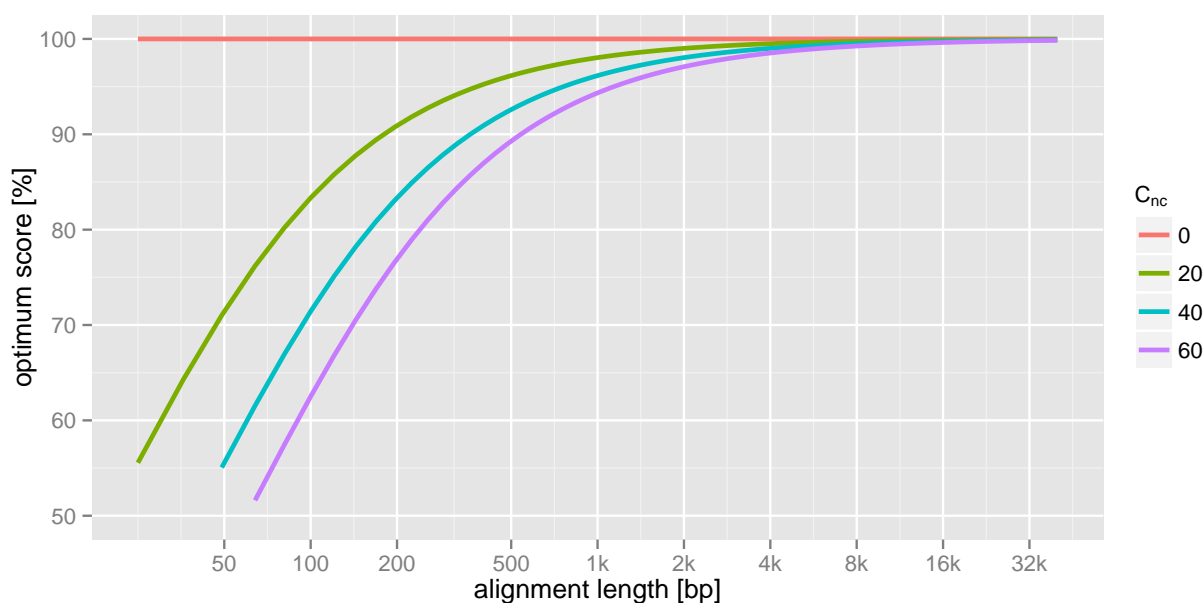


Figure 6.9.: Length-corrected length-normalized score computed according to eq. 6.4. Scores are plotted as a percentage of the respective uncorrected score and as function of the length of the alignment for different score correction constants C_{nc} .

S_n is the length-normalized version of the raw score (Eq. 6.3). S_{nc} is the length-normalized score corrected for the increasing level of uncertainty associated with very short alignments (Eq. 6.4). S_{nc} asymptotically approaches S_n for long alignments (fig. 6.9). The degree to which shorter alignments are penalized is controlled via a special constant C_{nc} . A

default value of 40 has been empirically determined to work well with typical short and long read data.

$$(6.3) \quad S_n = \frac{S}{l}$$

$$(6.4) \quad S_{nc} = S_n \times \frac{l}{C_{nc} + l}$$

l :	length
S :	raw score
S_n :	length-normalized score
S_{nc} :	corrected length-normalized score

6.4.2. Sam::Parser

The `Sam::Parser` class provides an interface to `SAM` / `BAM` files and data streams. For both, header and alignment section, parser methods returning single record entries are available. Header entries are split in a hash structure using tags as keys, alignment records are returned as `Sam::Alignment` objects. Advanced filter options restricting the stream to a specific subset, e.g. using `SAM` binary flags, can be configured. The class also provides methods for creating and writing to `SAM` files.

6.4.3. Sam::Seq

The `Sam::Seq` class is an auxiliary class, providing access to individual reference sequences and associated alignment records (`Sam::Alignment` objects). Its design focuses on the efficient computation of a consensus sequences from the given alignment data.

Typically, a `Sam::Seq` object is initiated from the reference sequence information (ID, length) provided in the header section of the underlying `SAM` file. The actual reference sequence data, if available, is stored as a `Fasta::Seq` / `Fastq::Seq` (section 7.1) object. Alignments are stored in a plain hash structure using unique internal IDs. This otherwise unordered association allows to quickly add, access and remove alignments and is sufficient to execute all basic operations.

Score based filter

Basic functionality of the `Sam::Seq` class comprises generic filters for alignments based on scores. The three defined score derivatives S , S_n and S_{nc} can be used. Filters can be executed during initialization, which can help to keep the memory footprint low, or run independently.

Repeat region filter

Handling of repeats and repetitive sequence motives is a common challenge when dealing with biological alignment data. Given a sufficient similarity, each existing copy of the repeat is likely to contribute alignments to a region even if it only comprises a single copy. This introduces noise through variations in different copies and at the same time is highly impractical from a computational point of view due to the multiplication of data. The `Sam::Seq` repeat region filter is designed to address local alignments of repeat core regions observed with long read and sequence data. The filter itself is based on the idea that correct placement of an alignment can be inferred if the alignment is placed at least partially outside the repetitive region.

In the `Sam::Seq` object, a repetitive region is defined as a region exceeding a certain repeat coverage threshold. This threshold needs to be set with respect to the given experimental background. Applying the repeat region filter will remove all alignments entirely placed within a repeat region. Sensitivity of the filter can be increased by configuring initially annotated repeat regions for symmetric extension, either by an absolute or relative length.

Containee filter

In some cases it is required to only keep one optimal alignment for each location in the reference sequence. Deciding on the best alignment, however, is difficult if long alignments of different length overlap each other. In this scenario, the containee filter handles the very common case of shorter alignments being contained within longer ones.

In order to detect contained alignments efficiently, alignments are first sorted by length in descending order. Starting from the second longest alignment, each alignment is compared against all available, longer alignments. Alignments are considered contained if at least 90% of the shorter alignment is located within start and end of the longer alignment. Each pair of alignments fulfilling the containment criteria is then processed according to a following rules: (1) Given alignments of similar length ($\Delta l \leq 40$ bp) the alignment with the higher length-normalized, corrected score (S_{nc}) is kept. (2) Given alignments of different length ($\Delta l > 40$ bp), the longer alignment is kept, even if it scores worse (fig. 6.10).

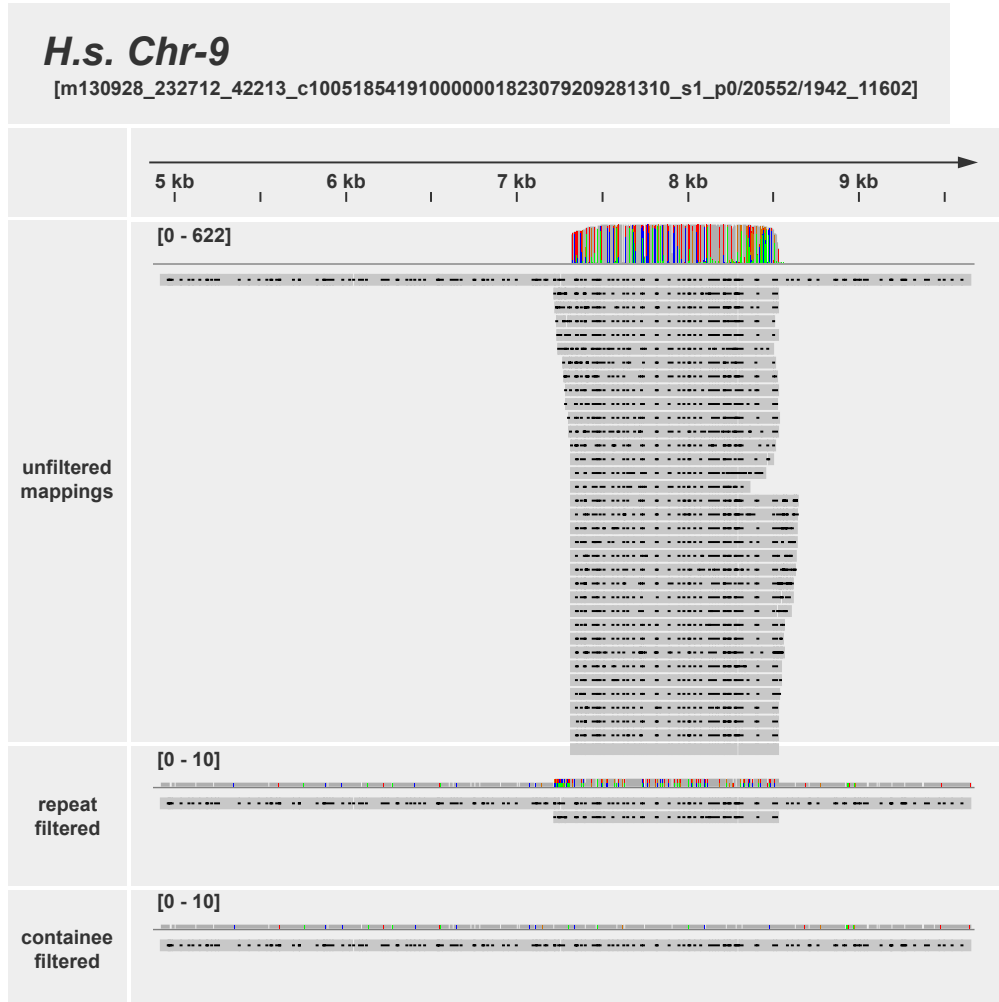


Figure 6.10.: Repeat and containee filters provided by the `Sam::Seq` class. Local alignments of uncorrected PacBio reads (grey bars) with alignment mismatches and indels (black dots) to a 5 kbp region of the *H. sapiens* Chromosome 9 comprising a repeat (7.3 kbp to 8.6 kbp). In the unfiltered output (panel 3) 622 alignments to the repeat regions are reported (only a subset of 35 is shown) including one alignment spanning across the repeat into adjacent regions. After application of the repeat region filter (panel 4), all but one partial alignment are removed. The remaining alignment is not considered repetitive as it extends to far outside the determined repeat region. The successive application of the containee filter (panel 5) further simplifies the scenario and reduces the alignment set to the only alignment covering the repeat as well as adjacent regions.

Binning based filter

Typically, genomic short read data is operated at coverages of tens to several hundred x . In the simple case of 1-to-1 relations of short reads and reference locations and high mapping specificity, this produces stable and manageable quantities of alignments for each `Sam::Seq` object. However, when working with repetitive or unevenly covered (transcriptomic, metagenomic) data sets and in combination with high mapping sensitivity and the possibility of multi-mappings, the number of alignments per reference sequence multiplies exponentially.

In this scenario, two major challenges need to be addressed to allow efficient handling of alignment data and consensus computation in particular: (a) Redundancy needs to be kept or reduced to a minimum sufficient level to allow the generation of a reliable signal from low resources. (b) Non-optimal alignments need to be identified and filtered in an efficient way to remove noise and further minimize computational demands (??). Both challenges can be met by assuming the following principles: (a) A sufficient minimum subset for consensus calling comprises a subset of all optimal alignments sampled in a way that all regions of the reference sequence are covered up to a certain maximum. (b) Optimal alignments can be identified by local score assessment.

In order to efficiently implement these principles, the `Sam::Seq` class is equipped with an additional auxiliary structure providing localized context for associated alignments. In this structure, the reference sequence is represented as a series of consecutive bins of defined length (default 20 bp). Alignments are assigned to their respective bin based on the center coordinate of their aligned sequence. Each bin holds ID, S_{nc} and length for each of its alignments, ranked by S_{nc} . Using this structure, an *optimal alignment subset* is given by the highest scoring alignments of each bin up to a customizable maximum per bin base pair threshold.

Binning based filtering can either be applied after loading all alignments of a `Sam::Seq` or already during loading. In the second case, each new alignment is checked against its respective bin. The alignment is added if the bin's total length plus the length of the new alignment does not exceed the bin's maximum base pair capacity. Otherwise, only alignments with scores higher than the lowest scoring alignment already in the bin are added while lowest scoring alignments are removed to satisfy the maximum base criteria. This way a fixed maximum memory footprint is guaranteed, given by *number of bins \times maximum bin coverage*. At the same time, the resulting `Sam::Seq` object holds a subset of optimal alignments for each location of the reference with evenly distributed coverage.

Binning and bin based alignment filtering make a powerful tool for the efficient handling of high coverage short read alignment data. However, parameters such as bin size and the maximum base pair threshold need to be set with care and require tuning with respect to technological and biological background.

State matrix

The basic structure for consensus computation in a `Sam::Seq` object is the *state matrix* (see Fig. 6.12). In this two-dimensional representation of the available alignment information, each position in the reference sequence is represented by one column. In this reference based coordinate system two different types of states are required to cover all possible relations: *Simple states* represent either a match, mismatch or insertion to the reference, holding a single nucleotide or a gap (A,T,G,C,-,N). *Compound states* originate from one or multiple successive deletions in the reference. The resulting orphaned nucleotides in the query are appended to their preceding state, creating multi-nucleotide states of arbitrary length within columns directly preceding deletions (e.g. AT,GGC,CCGAT).

The rows of the matrix hold the sum of weights of a specific state observed in all alignments at a particular position. In default mode, the weight of a state is modeled by its frequency. In the advanced, quality weighted mode, the weight of a state is inferred from its ASCII (American Standard Code for Information Interchange) encoded Phred quality score (Ewing et al., 1998; Ewing and Green, 1998). For details on conversion (section 6.4.3 proovread coefficient} . This approach introduces base-call confidence information to the matrix and the consensus model, increasing sensitivity on low quality / coverage data, as for example observed in unitig-based correction (section 6.3.5 Usability of pre-computed unitigs for correcion) or circular consensus computation (section 6.3.3 Redundancy reduction through circular consensus computation).

Within the matrix, state-weight association is provided by a semi-dynamic index. For the highly abundant simple states the index is static (A:0,T:1,C:2,G:3,-:4,N:5). For compound states, the index is dynamically extended on as-need basis for each individual `Sam::Seq`. This approach is both, memory efficient and fast.

The state matrix of a `Sam::Seq` is initialized by serially traversing all associated `Sam::Alignments`. Individual states are parsed from the sequence through transformation into reference coordinate space using the CIGAR string annotation. The initialization of the matrix can be restricted locally by (a) providing a subset of available alignments and (b) specifying coordinate ranges to be ignored.

In addition to state information derived from the alignments, also the states of the reference sequences can be added to the matrix. This feature, however, usually only make sense in quality weighted mode, as the frequencies for the reference states are all one.

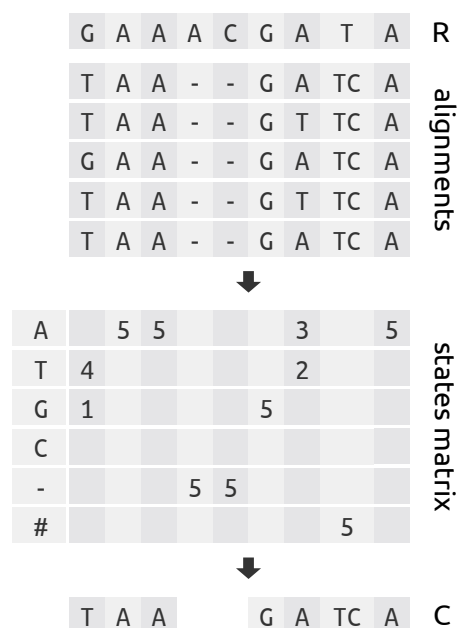


Figure 6.11.: proovread state matrix and consensus. Reference (R) and query states (alignments) are parsed into a state frequency matrix with reference based coordinates. The consensus (C) is generated by concatenation of the most abundant state for each column, ignoring gaps in the final sequence.

Trimming and Consensus

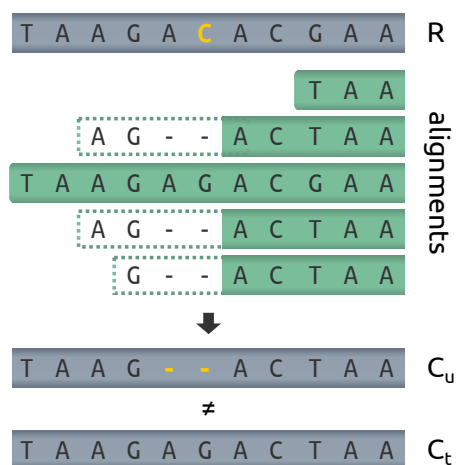


Figure 6.12.: Pre-consensus trimming of alignment ends. A sequencing error (yellow) in the reference sequence (R) shifts terminal bases (dotted parts) of aligned reads (green bars) in case that the introduction of two gaps is cheaper than the mismatch at the erroneous position. This introduces an artifact into the consensus (u), which can be avoided by trimming of gap containing alignment ends prior to consensus computation (t).

After the matrix has been constructed and alignment information has been loaded, a consensus can be determined. The consensus sequence is generated by concatenation of the highest weighted state in each column, ignoring gaps. In addition, a confidence score is calculated for each position based on the weight of the most abundant state. The weight is converted into a Phred quality score and encoded in ASCII to produce a quality string. That way, consensus sequences can be reported in common FASTQ (Cock et al., 2009) format, facilitating downstream processing with other software.

The accuracy of a consensus call from state matrix highly depends on the accuracy of the initial alignment information. Apart from misplaced reads a common source of errors are poorly aligned read ends. In particular gap favoring scoring schemes, as used with PacBio data, are prone to introduce cheap gaps at the ends of alignments instead of actually correct mismatches.

To avoid introducing these errors into the state matrix, the `Sam::Seq` object offers customizable trimming options for alignment ends, which are executed during matrix construction. Within a specified absolute or relative window at the beginning and the end the alignment is scanned for gaps. If gaps are found, the alignments are trimmed to remove all positions up to and including the most inner gap. This way only the more reliable core of the alignment is utilized increasing overall accuracy of

subsequent operations such as consensus calling.

proofread coefficient

The conversion of weights (or frequencies) into Phred quality scores and vice versa is implemented according to equations 6.5 and 6.6. C_{pr} is an empirically determined conversion coefficient. It effectively models the amount of trust put into weight / frequency when calling bases from a state matrix and at the same time the amount of trust that is put into Phred quality scores when comparing nucleotides between alignments. Using the conversion in both directions with the same coefficient ensures mathematical symmetry for weights and Phred quality scores ≤ 40 . The upper limit of 40 has been introduced for consistency with FASTQ Phred quality score specifications.

$C_{pr} = 120$ is used as default setting since `Sam::Seq-0.11`. Earlier versions, including the one used in the original proofread publication, employed a less generalized and less

$$(6.5) \quad w = \frac{p^2}{C_{pr}} \quad \begin{array}{ll} w: & \text{weight} \\ p: & \text{phred} \end{array}$$

$$(6.6) \quad p = \max\left(\sqrt{w \times C_{pr}}, 40\right)$$

flexible precursor of the described conversion system, roughly equivalent to $C_{pr} = 50$.

The generic two-way conversion model is an essential feature of the `Sam::Seq` class and its consensus calling capabilities. It facilitates the integration of sequence information derived from fundamentally different backgrounds – high quality / coverage short reads, low quality / coverage long reads as well as synthetic data, such as unitigs and CCS (Circular Consensus Sequencing) reads – in a straightforward and consistent manner.

Chimera detection

Aside from true sequence errors affecting regions of one to several adjacent bases, structural aberrations are a common source of error in PacBio sequence data. These aberrations manifest in fusion sequences, composed of two (or more) sequences from different origin, referred to as *chimeras*.

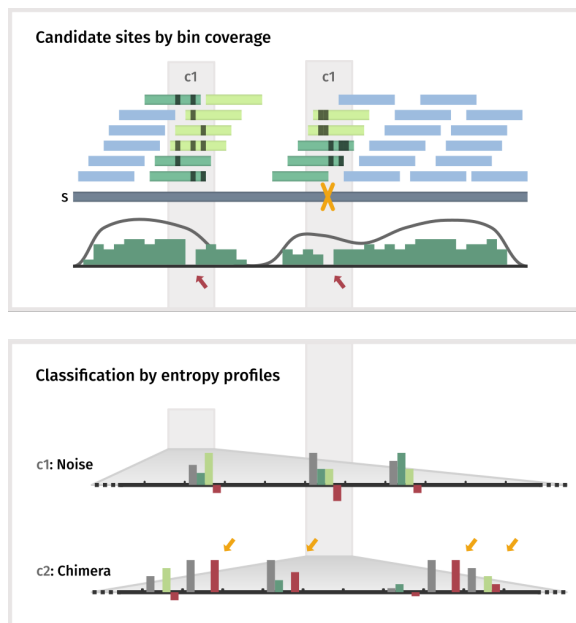


Figure 6.13.: Detection of chimera break points (Hackl et al., 2014). (1) Mapping of Illumina (light blue and green bars) onto pre-corrected PacBio read (blue bar) containing a chimeric break point (yellow X); black strokes indicate non-matching alignment positions. While not detectable by per-base coverage (gray curve), break point candidate sites (c1, c2) can be inferred from drops in bin coverage (green histogram, red arrows). Reads overlapping candidate sites from the right (dark green bars) and the left (light green bars) are considered for classification of each a site. (2) Entropy profiles derived from the alignment matrix at two candidate sites, comprising four entropies each: H_t calculated from the entire read set (gray bar), H_u for reads placed upstream (dark green bar), H_d for reads placed downstream (light green bar) and H_Δ (red bar), which is the difference of H_t and the greater value of H_u and H_d . An accumulation of $H_\Delta > 0$ (yellow arrows) is observed at true chimera locations.

The `Sam::Seq` class chimera detection is designed to identify chimeric break points based on the associated alignments. Two general cases of how alignments look like at break point can be distinguished: (a) If the fused sequence differs from the original sequence (and alignments settings are strict), alignments will not extend across a break point. (b)

6. *proofread – large-scale hybrid PacBio correction through iterative consensus*

If original and fused sequence are similar (and alignment settings are sensitive) reads will extend across the break point.

In accordance with the first case, a lack of coverage can indicate a break point. However, generating no alignment data for a location can have different reasons including genuine lack of data or a locally high error rate, exceeding alignment sensitivity. Therefore, lack of coverage does not qualify as criterion for positive break point identification. Removal of uncovered break points needs to be handled through alternative mechanisms, e. g. trimming of uncovered regions from the consensus sequence.

In the second scenario, break points cannot be identified by coverage. On the contrary, if coverage lies within expected limits, break points will go undetected in subsequent quality based trimming. Thus, advanced detection for covered break points is essential to prevent the generation of structurally inconsistent consensus sequences. The basis for detections of covered break points relies on differences between the expected and the actually present sequence on the other side of a break point. Break points at locations without differences in original and actual sequence cannot be detected based on alignment data. The detection process (Figure 6.13) is carried out in two steps.

First, candidate sites are gathered based on *bin coverage*. In contrast to per base coverage, bin coverage is affected in a more distinct manner at break points. Bin coverage is determined by placement of the alignment center coordinates. Only alignments located directly on top of a break point will contribute to the coverage of the according bin. Placement directly on top of the break point, however, requires about 50% of the read to be aligned to a false reference. Given differences in the underlying sequence, this is highly unlikely. Most reads overlapping a break point will be positioned largely at either side and only extend across the break point with a couple of bases. Thus, bin coverage at a break point is expected to be close to 0. Chimera candidate sites are given as regions of 2 to 5 consecutive bins below a customizable bin coverage threshold.

$$(6.7) \quad H(w) = - \sum_{i=1}^n p(w_i) \times \log_2 p(w_i)$$

H :	Shannon entropy
p :	probability
w :	state weight
i :	state iterator
n :	number of states

In a second step, candidate sites are verified through analysis of error patterns in the alignments surrounding the potential break point. An increase of non-matching bases alone can be attributed to noisy data, e.g. due to heterozygosity. However, at a true break point, upstream overlapping reads produce a consistent, yet distinctly different error pattern than downstream placed reads due to their separate origins. These differences can be detected from the alignment data by comparison of flank-specific local state matrices and the according overall matrix. For each column of the matrices, the *Shannon entropy* (Eq. 6.7) as a measurement for information content (Shannon, 1948) is computed. This produces three values per site: upstream entropy H_u , downstream entropy H_d and

the total entropy H_t , comprising both read groups. For comparison, the difference of H_t and the greater value of H_u and H_d , H_Δ is utilized. Positive values indicate an increase in information content per site in case flanking reads are assessed separately. A true chimera break point, in contrast to noisy spots introduced by other effects, is characterized by an accumulation of positive H_Δ values. The `Sam::Seq` chimera detection transforms the ratio of $H_\Delta > 0.7$ to the total number of signal containing columns into a chimera score χ . By choosing an adequate threshold for χ with respect to the given biological and technological background, `Sam::Seq` objects can be identified as chimeric and processed further accordingly.

Haplotype coverage

An important factor for the accuracy of consensus computation is the composition of the actually used optimal alignment set. Usually, this is controlled by a global coverage cutoff applied through binning. Binning based filtering ensures that the consensus is driven by reads actually derived from the reference sequence, with similar but lower scoring alignments being ignored. A global cutoff, however, is problematic if a data set comprises low coverage sequences, as for example transcriptomic or metagenomic samples do. Here, sequences with low coverage alignment support are driven towards similar sequences of higher coverage, because sup-optimal alignments contribute in higher numbers to the state matrix. Low coverage SNP positions in heterozygous genome data are affected in a similar manner.

Calling haplotype coverage on a `Sam::Seq` object will run an analysis aimed to identifying the true coverage of the underlying reference sequence. The analysis is based on the concept of *discriminating SNPs*: At a discriminating SNP position, true positive alignments uniformly carry a nucleotide different from alignments of other origins. These sites can be identified by first calling potential SNP positions, and second by assessing the score distribution of the alignments associated with the SNP. If within the highest scoring alignments, the distribution of states coincides with a state different from the overall most abundant state, then this state is considered the true state despite its low support. By computing the coverages for all most likely true states of each SNP positions along a read, the true overall coverage to the read can be inferred. This newly determined read-specific coverage can be used to further filter alignments and thus increase haplotype resolution.

6.4.4. `Sam::Phase`

The `Sam::Phase` class provides methods pertaining to the experimental read-back phasing algorithm described in section 6.3.7 [Haplotype awareness through variant calling and read-backed phasing](#). The implementation of the phasing is based on the algorithm introduced by Levy et al. (2007), yet with significant modifications to optimize performance for longer

6. *proovread – large-scale hybrid PacBio correction through iterative consensus*

and with respect to variable sites less accurate pre-corrected long reads. Further, not only SNPs, but also insertions, deletions and more complex small variants are supported.

The approach itself uses 0/1 encoding to represent reference matching and mismatching variant states for each SNV position. The obtained states for each read are concatenated forming short binary sequences. Using a greedy heuristic, the layout of the binary sequences of aligned reads is determined and sequences are subsequently merged into a consensus covering all variable sites present in the reference read. Finally, the states of the reference read are matched to the obtained consensus and corrected if necessary.

6.5. Conclusion

PacBio reads, due to their length, are a promising means in the quest towards the perfect assembly. The high sequencing error rate of the technology, however, impairs the usefulness of the generated data. Correction is a viable option, yet PacBio-only strategies are limited to small scale projects because of computational and economic constraints. Hybrid correction strategies, which integrate cheap and accurate Illumina with low coverage, long PacBio reads, are from a sequencing point of view suited for large genome assemblies. However, available software can only be applied to a very limited degree to large-scale projects. The `proofread` hybrid correction pipeline has been designed to fill this gap and enable correction of PacBio data generated for gigabase-sized genomes.

`proofread` uses a short read mapping based consensus approach, that assesses optimal alignments by localized score comparison. This renders the correction of individual PacBio reads independent of each other, facilitating distribution of computing jobs on arbitrary hardware architectures. As a result `proofread` scales well on data sets ranging in size from viral and bacterial to large eukaryotic genomes.

On benchmark data, `proofread`'s legacy implementation outperformed competing software in terms of correction accuracy as well as sensitivity, as indicated by highest total output and contiguity. The pipeline has since been further improved by the introduction of a faster, and specifically modified mapper (`bwa-proofread`), the addition of a pre-correction step utilizing precomputed unitigs and the implementation of processing steps addressing SMRT sequencing specific issues. Ultimately, `proofread` was used to successfully correct more than 15 Gbp of PacBio reads sequenced from Venus flytrap samples (section 4.7 PacBio sequencing, correction and assembly), demonstrating its applicability to large-scale projects.

Further, the usability of `proofread` for metagenomic data sets was explored. First, based on simulated data, the correction procedure was optimized with respect to low abundance read data. Subsequently, the resulting `proofread-meta` program was used for the correction of a marine sponge metagenome data set. Analysis of assemblies generated from only Illumina and from Illumina plus corrected PacBio reads, revealed substantial improvements to assembly quality for the hybrid approach.

With the latest upgrades to the pipeline, the challenge of haplotype aware hybrid correction was addressed. The basic Illumina-to-PacBio alignment and consensus approach was extended to include identification and stabilization of variable sites, coverage independent inference of the most likely state from the raw read and a final polishing step performing read-backed phasing. Preliminary test runs showed an increase in haplotype consistent variant calling from less than 80 % to over more than 99.9 %. Although still experimental, this proof-of-concept implementation demonstrates, that haplotype resolution on single read level during correction with `proofread` is achievable.

7. Efficient processing of sequence data

7.1. perl5lib-Fasta/-Fastq – a Perl API to sequence data

FASTA (Lipman and Pearson, 1985) and FASTQ (Cock et al., 2009) are the two most common text-based formats for storage of sequence data. Efficient processing of data in these formats is therefore a key prerequisite to the development of bioinformatic applications.

In the FASTA format (listing 7.1) each record is identified by a ‘>’ directly followed by a unique sequence identifier without white-space and an optional white-space delimited description. Subsequent lines hold the actual sequence information.

The FASTQ format (listing 7.2) is a de facto extension of the FASTA format for storing NGS (Next Generation Sequencing, Pettersson et al., 2009) sequence information and corresponding quality scores. A record is identified by a ‘@’, followed by a sequence identifier and an optional description. The second line holds the sequence information. The third line starts with ‘+’, optionally followed by additional descriptive information. Line four holds the ASCII encoded per base quality scores, usually Phred quality scores, representing estimated base call error probabilities during sequencing.

Listing 7.1: sample FASTA record

```
1 >ID [DESCRIPTION]
2 GTCGTACGTATATCGCTGCTACGTAACCACACAGACCGACTGCTAGCTATTTGACCAACGCTGCTAGC
3 ACTGTTACGTACTGACTGGA
```

Listing 7.2: sample FASTQ record

```
1 @ID [DESCRIPTION]
2 GTCGTACGTATATCGCTGCTACGTAACCACACAGACCGACTGCTAGCTAGAAATCGGTGGCTACGTAA
3 +
4 ADACEE9GDIIIJJIJGIFHIFGIIBFJKIBFKFKKIKJJFKBKHFEJFGKJGJ=K5CCJ7JKEFEJA@K
```

The Perl-based perl5lib-Fasta and perl5lib-Fastq libraries were developed as object-oriented APIs to sequence data stored in the respective formats. The libraries provide basic parsing functionalities as well as the more complex analysis, modification and conversion routines. This includes the masking/unmasking operations and window-based quality trimming employed by `proovread`. The libraries are publicly hosted at the GitHub repositories <https://github.com/BioInf-Wuerzburg/perl5lib-Fasta> and <https://github.com/BioInf-Wuerzburg/perl5lib-Fastq>.

7.2. SeqFilter – versatile manipulation of sequence files

SeqFilter is a Perl based tool for analysis and manipulation of sequence data stored in FASTA or FASTQ format. The program is implemented using the perl5lib-Fasta/q libraries (section 7.1). Its key functionalities comprise the compilation of basic sequence statistics (sequence count, length, shortest, longest, N50 and others Nxx values, sequence length distribution as histogram if filters are applied, statistics are provided for input and output), manipulation of sequence headers and sequences (modify ID and description, lowercase/uppercase conversion, mask irregular characters, reverse complement, extract or modify parts of sequences based on IDs and coordinates), extraction, removal or distribution of sequences based on length, ID lists or pattern matches and quality based manipulation of sequences (convert to FASTA, transform phred-offsets, trim/extract/-masked regions of low/high quality). SeqFilter is publicly available for download at <https://github.com/BioInf-Wuerzburg/SeqFilter>.

SeqFilter is a core component of the proovread correction pipeline. It was also used to generate the assembly metrics tables of the *D. muscipula* assemblies presented in this work.

7.3. SeqChunker – efficient subset generation and reproducible sampling for sequence data

Two important methods for computationally efficient processing of NGS sequencing data are *parallelization* and *sub-sample analysis*. Both approaches require that sequencing data is split into subsets of defined size and composition. SeqChunker has been developed in cooperation with Frank Förster and Simon Pfaff to provide an efficient solution for both of these tasks. The tool digests FASTA and FASTQ data with sequences of arbitrary length. The data are split either into chunks of a specified size (s) or alternatively into a specified number of chunks (n). Optimal performance in comparison to exhaustively parsing each sequence individually, is achieved by limiting parsing to chunk boundaries, while chunk cores are copied en bloc.

Systematic sampling (fig. 7.1) is realized by skipping defined subsets of chunks in the output. Skipping rules are defined through low-level integer arithmetic. Available parameters are chunk-step (x), chunks-per-step (y), first-chunk (f) and last-chunk (l). For example, an approximately 25% en bloc subset covering the second quarter of the input data set can be generated with $n = 4$, $f = 2$ and $l = 2$. A 25% subset systematically sampled across the entire data set is generated by using $x = 4$ and $y = 1$ and $n \gg x$. Its 75% complementary subset, comprising all sequences not yet utilized is produced with $x = 4$, $y = 3$ and $f = 2$. For convenience it is also possible to specify the desired subset size as percentage. In this case, SeqChunker will internally compute appropriate values for x and y .

7.3. SeqChunker – efficient subset generation and reproducible sampling for sequence data

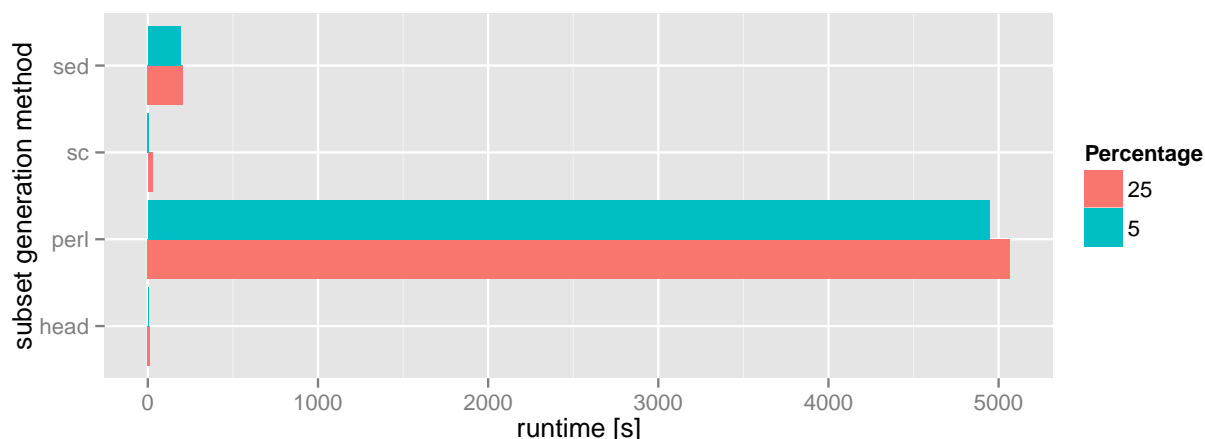


Figure 7.2.: Benchmark of SeqChunker sub-sampling procedure. Bars indicate the required run-time of different methods for sampling sets of 5% (blue) and 25% (red) from a large FASTQ file. SeqChunker was used with default settings and with respective `-percentage` values. `sed` was set to extract four lines every 16 and 80 lines, respectively. The Perl approach is based on parsing each individual sequence and reporting every fourth / twentieth record. With `head`, the first 5% / 25% of bytes present in the file were extracted.

The chunk-based systematic sub-sampling approach employed by SeqChunker is similarly robust as an entirely random sampling procedure with respect to introducing / obscuring biases. At the same time, benchmarks on test data (fig. 7.2) show that the implementation is 50 to 100 times faster than approaches requiring parsing of individual sequences.

Further, the procedure can be applied directly to a stream of data without requiring temporary storage or noteworthy processing resources. The method is deterministic and hence, results are fully reproducible. Contrasting to random sampling, it is also possible, to generate entirely complementary subsets.

SeqChunker is available as stand-alone tool at <https://github.com/BioInf-Wuerzburg/SeqChunker>. Additionally, it is distributed along with `proovread`, which utilizes SeqChunker for short reads sampling.

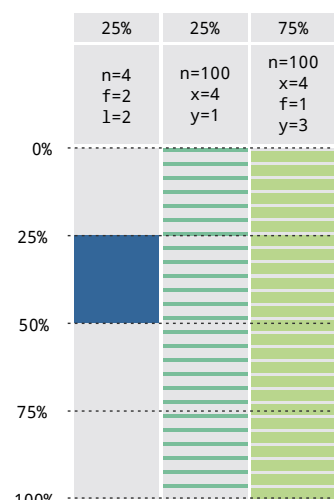


Figure 7.1.: Systematic sampling with SeqChunker. Colored bars represent extracted file regions for different sampling parameters (top panels)

8. Analysis and visualization of k -mer distributions and derived data

8.1. Introduction

8.1.1. The concept of k -mers in sequence analysis

k -mers, next to their key role in short read assembly (see section 1.3), are a particularly useful tool for the inference of sequencing sample characteristics, such as genome size, repeat content and ploidy. The major advantages of the technique is that the initial analysis is directly performed on sequencing data, thus rendering the process independent of the potentially complex assembly procedure (B. Liu et al., 2013).

k -mers – referred to as l -tuples in early publications – were first introduced to “estimat[e] the repeat structure and length of DNA sequences” by X. Li and Waterman (2003). Since then, k -mer-frequency analysis has become a commonly used method for assessing characteristics of genomic sequencing data sets (Chor et al., 2009; Huang et al., 2009; R. Li et al., 2010; Andrews, 2015). Further, k -mer-based approaches have been explored to facilitate similarity independent identification and assembly of repetitive elements (Gu et al., 2008; Koch et al., 2014; Nicolas et al., 2016). In combination with assemblies, k -mers can be used to aid in assessment of copy number variations (Shen and Kidd, 2015; Sinha and Kundu, 2015), classification of metagenomic sequences (Schmieder and R. Edwards, 2011; R. A. Edwards et al., 2012) or quantification of gene expression from RNA-Seq data (Patro et al., 2014).

Fast and efficient counting of k -mers on large NGS data sets is computationally challenging (Marçais and Kingsford, 2011). Different software, such as Meryl (E. Myers et al., 2000), Tallymer (Raphael et al., 2004), Jellyfish (Marçais and Kingsford, 2011), BFcounter (Melsted and Pritchard, 2011) and KMC 2 (Deorowicz et al., 2014), have been implemented to address the task.

In sequence analysis, the term k -mer refers to all possible substrings of a sequence of the length k . The total number of k -mers of a sequence is given by (Eq. 8.1). The *abundance* (a_k) of a k -mer is the number of times a distinct k -mer occurs within a sequence / a set of sequences: The sequence ATGGC, for example, comprises three distinct 3-mers ATG, TGG, GGC, each with an abundance of 1. If regions of a template sequence occur multiple times within the sequence, this is reflected in the number of distinct

8. Analysis and visualization of k -mer distributions and derived data

k -mers and the corresponding abundances: **ATGGATGC** comprises a total of six 3-mers, yet only a set of five distinct ones (**ATG**, **TGG**, **GGA**, **GAT**, **TGC**) with **ATG** occurring twice, thus having an abundance of 2. In contrast to abundance, the term *frequency* (f_a) does not apply to individual k -mers and their rate of occurrence, but is used to describe the distribution of k -mer abundances in a set. k -mer frequency refers to the number of times k -mers with the same abundance are observed; it therefore denotes the frequency of kmer abundances. In the example above, 3-mers with the abundance of 1 have a frequency of 4 (**TGG**, **GGA**, **GAT**, **TGC** occur once) and 3-mers with the abundance of 2 have a frequency of 1 (**ATG** occurs twice). Note that the definitions of k -mer abundance and frequency given here is widely accepted, yet can differ in other publications (team, 2016). The distribution of k -mer frequencies with respect to abundance, referred to as *k -mer distribution* or *k -mer spectrum* (Chor et al., 2009), is typically represented as smoothed curve connecting the discrete values of the distribution or as *k -mer histogram* with a bin size of 1 (Chikhi and Medvedev, 2014).

$$(8.1) \quad n_k = l_s - k + 1$$

n_k :	Number of kmers
l_s :	Sequence length
k :	Size of k -mer

The k -mer abundances obtained from sequencing data are directly related to the depth of coverage (c) of the sequenced sample. If the template sequence **ATGGC** is covered by 50 reads from end to end, this read set comprises the same three distinct 3-mers (**ATG**, **TGG**, **GCC**) as observed for the reference, however, each with an abundance of 50. Yet, the coverage values obtained from k -mer abundances, due to edge effects, differ from physical per-base coverage on actual data: Consider the reads **ATGC** and **GCGA**, which together cover the template **ATGGCGA**, resulting in an average per-base coverage of >1 (six bases with $c = 1$ plus one base with $c = 2$). On k -mer level, both reads each comprise two of the total of five 3-mers present in the template. The average c_k thus computes to 0.8. Because the number of k -mers in a read always is lower than the number of bases (Eq. 8.1), kmer-based coverages always fall short of actual per-base coverages. For typical Illumina data sets with 100 bp reads and typical k -mer sizes (19 to 31 used in analyses in this work), the maximum obtainable kmer-coverage is in the range of 70 % to 82 % of the true sequencing coverage.

In contrast to abundance, which refers to the actual count of a k -mer in the read set, the term *copy number* (CN) refers the number of times the k -mer is present in the sequenced template. In the aforementioned simple example with $50x$ read coverage, the abundance of each k -mer is 50, but the copy number is 1. Duplicated or repeated regions within a template by definition have higher copy numbers (2 to n), and derived k -mers proportionally higher abundances ($a_k \times n$). Due to the random generation of sequencing fragments, the coverage of the underlying template within a set varies, and in turn, so do the observed abundances. Because of the random, independent nature of the events, the k -mer frequency distributions for each population of k -mers with identical copy number can be modelled as Poisson distributions, and for high coverages (>1000)

approximated with the Gauss distribution (Chikhi and Medvedev, 2014). On real data, biases associated with PCR (Polymerase Chain Reaction, Bartlett and Stirling, 2003) and amplification, which are part of the sequencing process (Quail et al., 2012), constitute a non-independent component that can skew the distribution. Depending on the severity of the effects, the negative binomial distribution, which allows for larger variances than Poisson, can provide an alternative, more accurate model (Anders and Huber, 2010).

A particular strength of the Illumina sequencing technology is its high rate of accuracy of >99% – nevertheless errors, mostly substitutions, do occur (Morozova and Marra, 2008; Dohm and Lottaz, 2008). From a k -mer perspective, an error in a sequencing read is an alteration of all k -mers within the read that overlap with the erroneous position. For large enough k -mers, the vast majority of the erroneous k -mers are novel distinct k -mers that do not match the set of k -mers present in the template (Kelley et al., 2010). Further, the introduction of an individual error to a read, usually represents an event unique to a single read. Thus, the abundance of erroneous k -mers is 1, or slightly higher in rare cases of coinciding errors.

Typically, sequencing is performed from non-strand specific sequencing libraries (Levin et al., 2010). Thus, sequencing read sets comprise reads and consequently k -mers from both strands conveying the same information. This is accounted for in k -mer analysis by using canonical k -mers (Y. Liu et al., 2013; Kundeti et al., 2010). A canonical k -mer is defined as the lexicographically smaller representation of a k -mer and its reverse complement. The sequence ATGGC comprises three distinct pairs of forward / reverse complement 3-mers, ATG/CAT, TGG/CCA/ and GGC/GCC, which correspond to the three canonical 3-mers ATG, CCA and GCC. Reads generated from the template, by definition, comprise the same three canonical 3-mers regardless of the strand they were sequenced from.

Unless otherwise stated, all analysis in this work were carried out based on canonical k -mer representation. Furthermore, counting of k -mers was performed with Jellyfish.

8.1.2. Interpretation of k -mer distributions

The typical k -mer distribution of a haploid genomic sample, as plotted in fig. 8.1 exhibits a unimodal k -mer frequency distribution. The main peak represents k -mers with a copy number of 1 and with abundances corresponding to the depth of sequencing of the analyzed library. The k -mer coverage of a sample is determined by the center of the main peak, here $78x$. Low abundance (also referred to as low coverage) k -mers at high frequencies correspond to k -mers resulting from sequencing errors. k -mers with abundances greater than the main peak correspond to regions of higher copy numbers, i. e. repetitive k -mers. The k -mer coverage observed in the example is in congruence with the expected per-base coverage of the sample: The size of the *E. coli* K-12 substr. MG1655 reference genome is 4.64 Mbp (Accession NC_000913); the analyzed sample comprises 5×10^6 reads at 100 bp and thus a total of 500 Mbp. This computes to a theoretical maximum per-base sequencing depth of $109x$. The true coverage needs to be assumed

8. Analysis and visualization of k -mer distributions and derived data

lower, as the read set usually contains minor cross and RNA contaminations as well as adapter sequences and regions comprising uninformative Ns (Quail et al., 2012). The theoretical maximum for the k -mer coverage for 19-mers on 100 bp due to edge-effects, is 82 % of the true per-base coverage. This maximum, however is shifted to lower values because of sequencing errors, which remove k -mers from the main distribution. Thus, assuming a reduced maximum per-base coverage of $100x$, $78x$ k -mer coverage fits the expected range of less than 82 % of the per-base coverage well.

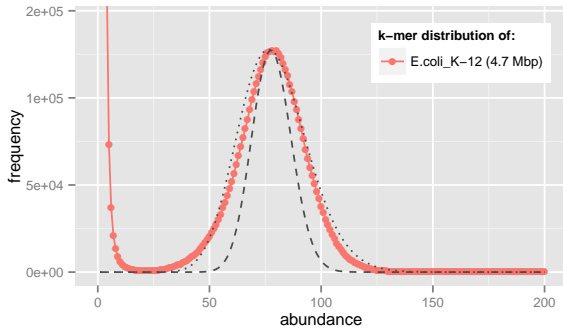


Figure 8.1.: Distribution of 19-mers in 5×10^6 reads of *E. coli* K-12 library ERR008613; red points connected by a line represent frequencies of the observed k -mer abundances. The dashed and dotted lines represent Poisson and negative binomial fits, respectively.

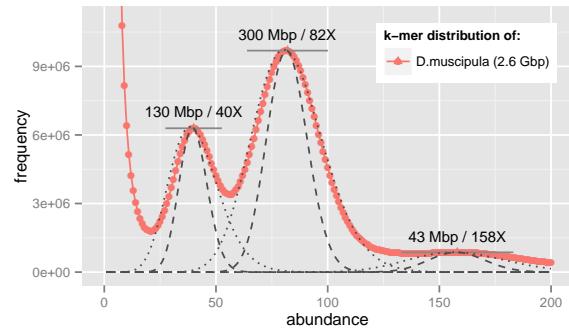


Figure 8.2.: Bimodal distribution of 19-mers for heterozygous *D. muscipula* libraries; red points connected by a line represent frequencies of the observed k -mer abundances. The dashed and dotted lines represent Poisson and negative binomial fits, respectively.

For diploid, heterozygous samples, the k -mer distribution typically shows a bimodal frequency distribution (fig. 8.2). The first, usually smaller peak corresponds to k -mers present only in one allele, and thus is referred to as *heterozygous peak* comprising heterozygous k -mers. The second, *homozygous peak*, at twice the coverage of the heterozygous peak represents the majority of k -mers, which are shared between the two alleles. Note, that the term copy number in di- and polyploid scenarios is not applied at the level of alleles, but on the polyploid genome as a whole, meaning, a region present in all homologous chromosomes has a copy number of 1.

In general, peaks in a k -mer distribution represent read populations and consequently regions of the sequenced sample with a distinct copy numbers. In the *D. muscipula* example (fig. 8.2), a minor, third peak at $160x$, which is twice the homozygous coverage indicates regions with a copy number of 2. Depending on the structure of the genome, this can either be attributed to several smaller duplicated regions (e.g. paralogous genes) or to larger segmental duplications. Further, peaks at coverages lower than the heterozygous peak often correspond to contaminations, while peaks at higher coverages can represent genomes of plastids or mitochondria, which are present in the sequencing set in much higher abundances than the nuclear genome.

8.1.3. Estimation of genome size, repeat content and heterozygosity

The size of a sequenced genome can directly be inferred from the k -mer distribution according to eq. 8.2 (X. Li and Waterman, 2003). The assumption is that all non-erroneous (and non-contamination derived) k -mers belong to the genome and that abundances directly convey the underlying copy numbers. For long sequences, such as chromosomes, the sequence length can be approximated by the total number of k -mers contained in the sequence. If the set of k -mers is represented by distinct k -mers with abundances, each k -mer with copy number > 1 needs to be multiplied by its abundance to obtain the size of the entire set. This logic also applies to read data derived from genomes, however, with an additional normalization to transform abundances into copy numbers.

$$(8.2) \quad s_g = \frac{\sum_{i=a_t}^{\infty} a_i \times f_i}{c_k}$$

a :	abundance
f :	frequency
a_t :	threshold for non-erroneous k -mers
c_k :	k -mer coverage (CN== 1)
s_g :	size of the genome

Similarly, also the sizes of subsets of the genome for different ranges of copy numbers can be estimated. The total length of repetitive sequences in a set can be inferred using eq. 8.2 with an abundance threshold a_t corresponding to copy numbers > 1 . SNV counts and SNV rates can be estimated based on the size of the heterozygous peak and the non-repetitive fraction of the genome using eq. 8.3 and 8.4, respectively (B. Liu et al., 2013). Here the reasoning is that, similar to sequencing errors, each SNV ideally affects k SNV-overlapping k -mers. Therefore, as an approximation, the amount of SNVs in a diploid genome is proportional to the amount of reference k -mers affected by SNVs. However, if a SNV is introduced, the set of overlapping k -mers is split in two subsets, each representing one of the two variants with halved abundances. Therefore, the heterozygous peak comprises $2 \times k$ k -mers per SNV. It should be noted, though, that this approximation does not generally hold true for repeats. Here, SNVs often affect only one allele of a single copy of the repeat. Therefore, k -mers derived from this particular copy will contribute to the heterozygous peak, while the corresponding k -mers of the other copies will remain in the repetitive fraction of the distribution. Therefore, the ratio of heterozygous to homozygous k -mers can be shifted towards the heterozygous peak and lead to an overestimation of the heterozygosity rate deduced from the k -mer distribution.

$$(8.3) \quad n_v = \frac{s_{het}}{2 \times k}$$

k :	kmer size
s_{het} :	size of the heterozygous peak
n_v :	number of SNVs

$$(8.4) \quad r_v = 1 \text{ in } \frac{s_u}{n_v}$$

s_u :	size of the non-repetitive genome
r_v :	SNV rate

8.2. Anscombe transformation facilitates computational analysis and visualization of k -mer distributions

k -mer distributions and plots carry valuable information about the underlying sequenced sample. Yet, inference of underlying properties, visualization and annotation of the distribution often require manual investigation or alternatively the fitting of specialized models (Chaisson and Pevzner, 2008; Kelley et al., 2010; Chikhi and Medvedev, 2014). Although, these models, which comprise mixtures of binomial, Poisson or Gauss distributions, are not necessarily complex from a mathematical point of view, their implementation in non-statistical languages and the computations required for fit optimizations can be challenging. Further, these models are designed for particular scenarios, e.g. haploid or diploid genomes at coverage ranges from $20x$ to $200x$, and, based on experience, perform rather poorly on other data sets.

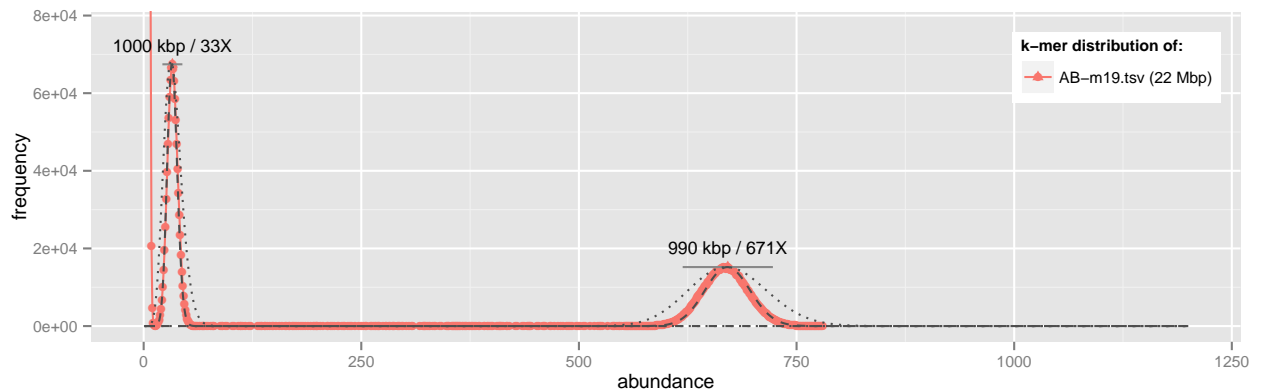
The proper (automated) visualization of k -mer distributions is closely related to the ability of inferring basic characteristics directly from the raw abundance–frequency data. While usually plots are drawn in a way that the entire data set is displayed, for k -mer distributions, this does not make a lot of sense. The maximum frequencies, observed for low coverage erroneous k -mers, can be several orders of magnitude larger than the frequencies of the peaks of interest. The same holds true for high abundances observed for repetitive k -mers. Therefore, plot regions need to be restricted with respect to location and size of the signal containing peaks, rather than the full data set. This, however, requires that size and location of potential peaks can be inferred directly from the data prior to the generation of the actual plot.

The key problem for peak detection is that the width and height of a peak directly depend on the position of the peak on the x-axis: the higher the abundances, the broader and lower the peaks (fig. 8.3 A) – or in mathematical terms: For Poissonian data, the variance of the data is equal to the mean of the distribution ($\sigma^2 = \mu$). As a result of this dependency, peaks have to be assessed with respect to their abundances, which complicates regressions-based fits, but also graphical data exploration (Everitt, 2002). Consider the two peaks in fig. 8.3 A at mean abundances of $33x$ and $671x$. Because of the difference in shape, it is difficult to compare the two peaks by eye, for example to tell, which peak is larger and represents a larger fraction of the sequenced sample.

The challenge of mean–variance dependency in general, can be met with variance stabilizing transformation (Everitt, 2002). In this procedure, the raw values of the data set are transformed by a function in a way that in the resulting data set, the variance and mean are independent. For Poissonian data, for example, variance stabilization can be achieved by a simple square root transformation. A more practical transformation was introduced first by Anscombe (1948): The *Anscombe transformation* (Eq. 8.5) turns Poissonian into approximately Gaussian data with a fixed standard deviation (σ) of 1. While utilization of Anscombe transformation has been proposed for RNA-Seq (RNA sequencing, Chu and Corey, 2012) data analysis (Kulinskaya et al., 2008; Rau et al., 2014), to my knowledge,

8.2. Anscombe transformation facilitates computational analysis and visualization of k -mer distributions

A



B

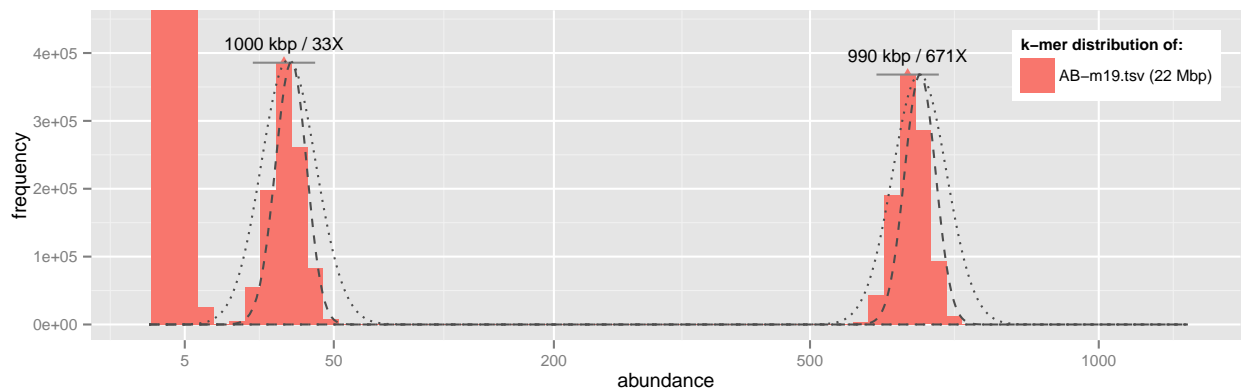


Figure 8.3.: Frequency distribution of 19-mers in an artificial read set comprising to populations of reads with different coverage levels, generated with `kmer-plot kcov`. The distribution is shown as continuous smoothed curve in linear space (A) and as stacked histogram in Anscombe transformed space. Peak sizes and coverages are indicated by the respective labels. Dashed and dotted lines represent Poisson and negative binomial fits, respectively. While in linear space the dependency of the variance of Poisson distributed data from the mean of the distribution renders direct comparison of the peaks in terms of size and shape difficult, transformation into Anscombe space allows for an intuitive interpretation.

this work is the first exploring and demonstrating its applicability for the improvement of k -mer distribution analysis.

Figure 8.3 A and B both show the k -mer distribution of a simulated read set generated from two artificial 1 Mbp templates, with per-base coverage of $50x$ and $1,000x$, respectively. While plot A was directly generated from the raw abundance-frequency data, plot B shows the distribution as histogram in Anscombe transformed space with transformed abundances cumulatively binned in intervals of 1. x-axis labels of the plot were retransformed to regular space values for better comparability. In terms of readability, the Anscombe transformed plot has two distinct advantages: First, the size and shape (Gauss) of the peaks is independent of their position on the x-axis, and thus peaks can be directly

8. Analysis and visualization of k -mer distributions and derived data

$$(8.5) \quad x_a = 2\sqrt{x + \frac{3}{8}} \quad x_a: \quad \text{Anscombe transformed value}$$

compared. While in the non-transformed plot, the difference in size of the two peaks is difficult to determine, in the Anscombe transformed histogram, it is evident that both peaks are of the same size and thus represent genomic regions of similar size. Second, the non-linear scaling of the x-axis increases the resolution on low abundance data while compressing distances at high abundances. Therefore, Anscombe transformation facilitates the graphical exploration of signals ranging over several orders of magnitude.

From a computational point of view, Anscombe transformation is also highly beneficial. For Gaussian data, the relative amount of data points within a certain range around the mean (μ) of the function can be directly expressed as function of the standard deviation of the distribution. A range of one standard deviation ($\mu \pm \sigma$) accounts for 68%, a range of two standard deviations ($\mu \pm 2\sigma$) for 95% and a range of three standard deviations ($\mu \pm 3\sigma$) for 99.7% of the data. Given that Anscombe transformation turns Poissonian into Gaussian data with a fixed standard deviation of 1, and given a representation of the transformed data in a discrete Anscombe transformed histogram with an interval size of 1, which is equal to σ , the 5 bins closest to the peaks maximum represent 97.6% to 98.8% of the total data of a single peak. This generalized assumption allows for the formulation of a simple and efficient peak calling heuristic: Peaks are given by local maxima obtained from a sliding window analysis with window size 5 on aggregated Anscombe transformed data. This heuristic is straight-forward in its implementation, does not require computationally intensive fit optimization, works on any scale of abundances, and imposes no additional assumptions on the data, such as state of ploidy, rendering it universally applicable.

This important concept new forms the basis for the k -mer based analytical tools described in the following sections, and which were essential to the assessment of the *D. muscipula* draft assemblies presented in the Part ?? ??f this work.

8.3. *kmer-plot kcov* – visualization and automated annotation of *k*-mer distributions

The `kcov` command of the `kmer-plot` toolkit provides a versatile interface for the generation and automated annotation of *k*-mer distribution plots. The primary interface is written in Bash (Unix shell and command language), data processing and plotting are implemented in R (Statistical programming language) and using `ggplot2` (R plotting library). The program is able to process multiple input files, either provided as tab-separated plain files or as binary Jellyfish hashes. Initial analysis of the raw data and inference of plot-relevant metrics, such as location and size of peaks rely on Anscombe transformation based heuristics, as described in the previous section (section 8.2), and are executed automatically during each run. The generated plots are scaled accordingly, peak locations and sizes are indicated by labels, genome size estimates are added to the legend. Distributions can either be drawn in regular or Anscombe space representation.

Figure 8.4 and fig. 8.5 show sample plots generated from two real experiments and demonstrate the range of applicability of the program. Figure 8.4 was generated from different *D. muscipula* data sets with estimated genome sizes of 2.6 Gbp and 2.7 Gbp, respectively; it is displayed in regular space and focuses on size and topology of the annotated main peaks at abundances between $1x$ and $300x$. fig. 8.5 depicts distributions of high coverage sequencing samples in Anscombe space obtained from viral samples, before and after quality trimming. Although the samples exhibit extreme sequencing biases – indicated by very broad peaks – the annotation heuristic employed by `kcov` is capable of recognizing the signals of interest and scale the plot accordingly.

8. Analysis and visualization of k -mer distributions and derived data

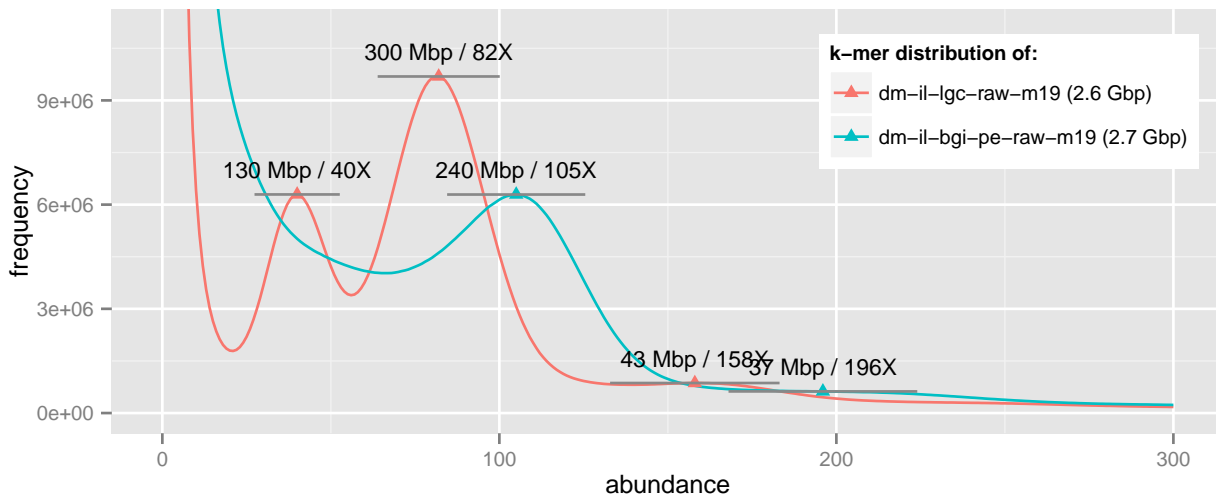


Figure 8.4.: 19-mer distribution plot of sequencing data of a large diploid genome generated with `kmer-plot` `kcov`. The two lines represent two *D. muscipula* data set from different sequencing experiments. Axis scaling, genome sizes indicated in the legend (2.6 Gbp and 2.7 Gbp) and peak annotations indicating size and coverage have been automatically determined by the script with default settings.

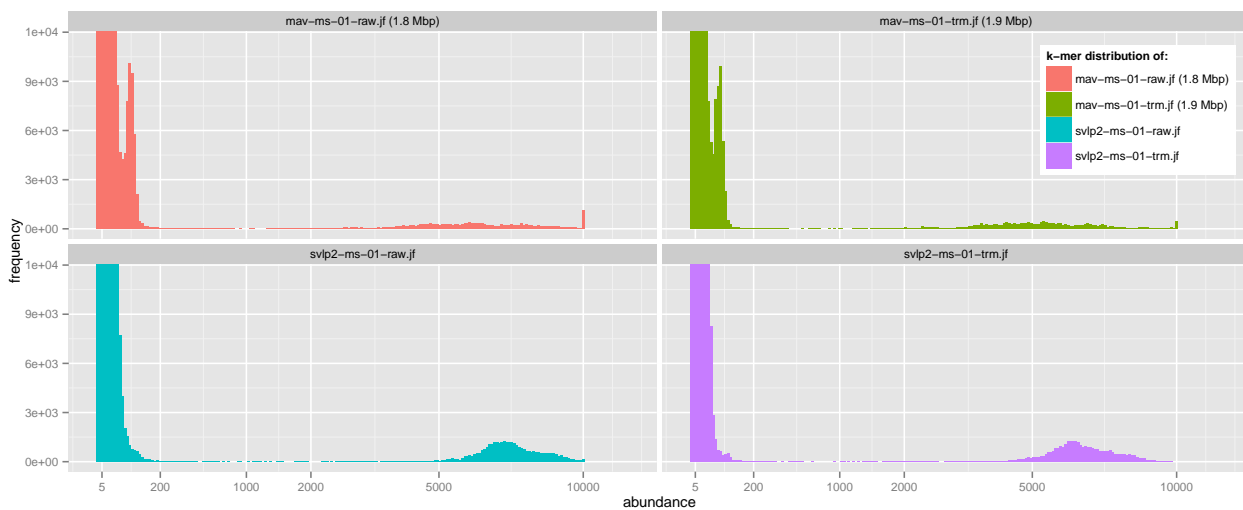


Figure 8.5.: 19-mer distribution of different viral, high coverage sequencing samples plotted with `kmer-plot` `kcov` as histograms in Anscombe space and oriented in multiple panels.

8.4. *k*-mer-coverage – representative, frequency adjusted *k*-mer coverage

While *k*-mer abundances obtained from high coverage read data inherently carry information about general characteristics of the sequenced sample, *k*-mer abundances can also be used to specifically assess reference sequences, such as assembly contigs. For this approach, instead of computing the abundances of *k*-mers present in the reference directly, the abundance of each reference *k*-mer is determined based on its abundance in the high coverage read data. This transfer of abundances from read data onto reference sequences results in sequence specific *k*-mer coverage profiles, or short *k*-mer profiles, which convey sequence specific characteristics, such as copy numbers, zygoty and repetitiveness of contained regions as well as potential assembly errors.

For example, given a read set with $100x$ *k*-mer coverage, regions comprising mostly *k*-mers with abundances close to $100x$ correspond to regions with a copy number of 1. Regions with higher abundances correspond to higher copy numbers and hence indicate repeats. *k*-mer profiles offer insights similar to per-base coverage obtained from analysis of mapped reads. However, the *k*-mer-based approach is computationally much more efficient as neither the computation nor the processing of alignments to obtain actual per-base coverages, is required. Moreover, once the *k*-mer-abundances have been computed for the read set, these precomputed abundances can be used to analyze an arbitrary number of sequence sets. There is, however, a distinct difference between read-based and *k*-mer-based coverages: Mapping-derived per-base coverages are context dependent, meaning the results obtained for a specific sequence can differ depending on which other sequences are present in the analyzed data set. For example, reads from repetitive regions potentially map to multiple locations in the reference. To avoid multi-mappings in the output, only the best mapping for each read is reported. If several alignments score equally, the mapping location is determined by random choice. Therefore, if the genome comprises a repeat with 5 copies, the genome is sequenced and assembled, and the assembly also comprises all 5 copies, the corresponding reads will during mapping be evenly distributed among those copies. Ultimately, the per-base coverages also for the repeats will be in the range of the sequencing depth. However, if during the assembly, the repeat sequences are collapsed and only a single copy is reported in the contigs, this regions will gather all reads matching the repeat, and thus get per-base coverages 5 times higher than the sequencing depth. The same applies if only single sequences or subsets of sequences of an assembly are analyzed. A decrease of the copy number of a particular region in a sequence set will cause an increase in observed read-based coverages for the remaining copies of said region. *k*-mer profiles, in contrast to mapping-derived per-base coverages, are independent of the context the sequence occurs in. A repetitive *k*-mer will always have the abundance with which it occurs in the read set, regardless of the number of times, the repeat is present in the assembly. Therefore, *k*-mer coverages directly, and even out of context, indicate repetitive regions within a sequence.

Adjusted k -mer coverage

Carrying context information, as per-base coverage does, however, can be of high value in the analysis of assembly data as well. As mentioned before, collapsed repeats can be identified using per-base coverage, as these regions exhibit higher per-base coverages than non-collapsed regions. Similar, if for a heterozygous data set, alleles are assembled into separate contigs, the coverage will be split between these two copies and thus only be half as high as for regular regions. To introduce context dependency into k -mer abundance data, I devised a new approach that produces k -mer profiles of *adjusted coverage*. The adjusted coverage of a k -mer is given by the ratio of its abundance in the read set and the number of its occurrence in the analyzed reference sequence set. Again, consider the example of the assembly of a 5 copy repeat from a $100x$ sequencing sample. If the repeat is fully resolved in the assembly and present in 5 copies, its raw k -mer coverages are approx. $500x$, its adjusted coverages, however, compute to $100x$. This is equal to the coverages expected for regions with a copy number of 1 and present once in the assembly. If, however, the repeat would have been collapsed during assembly the adjusted coverages would increase proportionally. Therefore, adjusted k -mer coverage can be interpreted analogous to mapping-derived coverages, but at the same time generated much faster.

Representative k -mer coverage

While the detailed analysis of k -mer profiles allows to assess particular regions of a sequence individually, it often is useful to make more generalized assertion on the overall nature of a complete sequence, such as: “contig A is repetitive, contig B has low coverage, ...”. Typically, this is done by either using the mean or the median of the observed coverages to describe the overall tendency in the data. However, mean as well as median only provide a robust measure for unimodal distributions. k -mer distribution as well as profiles often are multimodal mixtures, with different peaks corresponding to distinct populations of reads and by extension to regions of distinct copy numbers. To better reflect this multimodal topology representing distinct levels of copy numbers, I devised an alternative approach that defines the *representative coverage* of a sequence. The concept of representative coverage is based on the notion that the global nature of a sequence is more adequately given by the coverage corresponding to the distinct copy number of the largest portion of the analysed sequence, rather than mere mean / median. Representative coverage, thus, is given by the coverage determined for the largest peak within the distribution. In contrast to mean and median, representative coverage is a robust measure in the sense that (a) only coverages actually representing a significant fraction of the entire data set are considered in the first place and that (b) the measure is independent of size and position of any additional peaks corresponding to other, minor populations of k -mers.

Implementation

The *kmer-coverage* tool is a multi-thread ready, Perl-based script that allows the computation of the raw and adjusted *k*-mer coverages for all positions in a set of given reference sequences. Based on these profiles, median and representative coverage for both, raw and adjusted coverages of each analyzed sequence are reported. Internally, the *Jellyfish* software is employed for *k*-mer counting. *k*-mer abundances in the reference sequence set required for adjusted coverages are computed on the fly. Peak calling, required for the computation of the representative coverage, is premised on the Anscombe transformation based heuristic described in section 8.2. The *kmer-coverage* tool was used for the generation of all *k*-mer coverage dependent analyses presented in this work.

8.5. *kmer-plot gccov* – automated visualization of GC-coverage plots

GC-coverage plots were introduced to characterize bacterial contaminations in assemblies of nonaxenic cultures by Nederbragt and Rounge (2010). Later, they were adopted for the analysis of metagenome assemblies of bacterial consortia (D’haeseleer et al., 2013) and simultaneous sequencing experiments of host-symbiont systems (Kumar and Blaxter, 2011; Kumar et al., 2013). GC-coverage plots are scatter-plots, in which individual contigs are represented by a single data point with their relative GC content given on the x-axis and their mean / median per-base coverage on the y-axis. Because of small intra-species GC content variations but often large inter-species differences, and because of the proportionality of coverage and abundance of different templates in the initial sample, contigs derived from the same source cluster together in GC-coverage plots. GC-coverage plots, in general, provide a mean for the graphical exploration of an assembly with respect to the composition of the underlying sequencing sample. This not only holds true for contigs derived from independent templates, but also for assemblies of complex genomes, for which, especially if the level of fragmentation is high, different contigs can represent different subsets of the total genome with diverse properties, such as high copy number genes, transposable elements or viral integrations. Usually, GC-coverage plots are generated from mapping-derived per-base coverages. This, however, renders the approach computationally impracticable on large data sets.

To allow the efficient generation of the GC-coverage plots for large data sets, I adopted the underlying idea, however used it with *k*-mer-derived coverages rather than mapping-based ones. As described in the previous section (section 8.4), representative adjusted *k*-mer coverages can be utilized as an alternative to mapping-derived coverages and interpreted in a similar manner, yet can be computed much more efficiently. To facilitate the automatic generation of GC-coverage plots I developed the *kmer-plot gccov* tool. It extends the R / *ggplot2* based *kmer-plot* implementation for processing and plotting GC-coverage and associated taxonomic information.

8. Analysis and visualization of k -mer distributions and derived data

To illustrate the utility of the `gccov` plots, let me introduce a short background story. In December 2015, Boothby and Tenlen (2015) published the draft genome of the tardigrade *Hypsibius dujardini*. Based on the identification of high levels of non-tardigrade derived genes in the assembly, the authors concluded that the assembly is proof of a high level of horizontal gene transfer between tardigrades and bacteria. Soon after the publication of the study, that claim became the subject of a heated debate among scientist, mostly communicated through social media channels. Less than two weeks later, Koutsovoulos et al., 2015 submitted a paper to BioRxiv entitled Koutsovoulos2015. The obvious question is: Who is right?

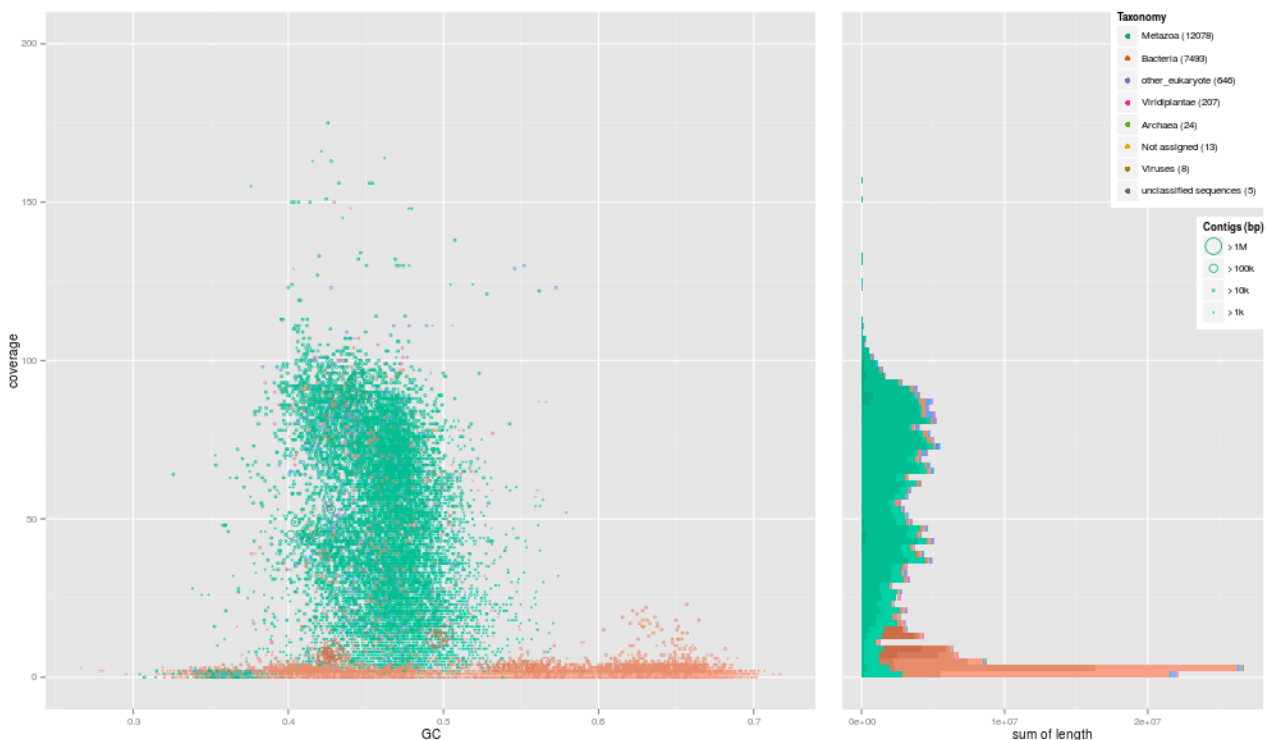


Figure 8.6.: GC-content and sequence length distribution as a function of representative adjusted k -mer coverage for the draft genome of *Hypsibius dujardini*. Left Panel: Circles represent individual sequences with length encoded by size and taxonomic classification indicated by color. Right Panel: Histogram bars depict cumulative lengths of the entire sequence set binned according to length intervals, color coded and stacked in decreasing order.

Figure 8.6 shows the GC-coverage for the genome assembly of *Hypsibius dujardini* generated with `gccov` from the published data. Each spot in the left panel represents a contigs plotted with respect to its relative GC-content and representative adjusted k -mer coverage. Dot / circle size indicate contig length. Different colors correspond to taxonomic classifications assigned to each sequence by my colleague Frank Förster using a homology-based approach. One major fraction of contigs clusters within a GC-range of 40 % to 50 % and a coverage range of $20x$ to $100x$. The vast majority of these sequences

is characterized as metazoan and can be attributed to the genome of *Hypsibius*. The second major fraction comprises sequences of bacterial origin and show coverages of $>20x$ and are distributed over a large GC-range. The difference in coverage levels for genomic and bacterial sequences clearly indicates that the bacterial sequences in the sample are not integrated into the tardigrade genome, but but constitute independent contaminations. Contrasting, only a very small fraction of bacterial sequences shows coverages similar to those of the tardigrade sequence. This clearly contradicts the claim of excessive levels of horizontal gene transfer made by Boothby and Tenlen (2015).

8.6. Conclusion

k -mer distributions and derived k -mer profiles of assembly sequences are powerful means for the exploratory analysis of genomic data as well as generated assemblies. Compared to mapping-derived coverage, the concept of k -mer coverage provides a computationally advantageous approach, that with the introduction of adjusted and representative coverages also is superior in terms range of application and interpretation of results. The `kmer-coverage` and `kmer-plot` scripts facilitate the automated generation of k -mer based plots and, thus, are valuable tools for analysis of, in particular, large assembly data sets.

Back matter

Glossary

454	Roche 454 pyro-sequencing technology. 31, 33, 159
API	Application programming interface. 123, 137, 159
ASCII	Generic character-encoding scheme. 129, 130, 137, 159
Assemblathon 2	Genome assembly competition (http://assemblathon.org/assemblathon2). 32, 159
BAM	Compressed, binary representation of SAM format (http://samtools.github.io/hts-specs/SAMv1.pdf). 24, 35, 111, 123, 125, 159
BGI	Beijing Genomics Institute (http://www.genomics.cn/en/index). ix, xi, 11–16, 24–26, 28–31, 33, 37, 38, 40, 46, 47, 49–56, 58–61, 65, 67, 68, 70, 71, 73–75, 77, 78, 80, 82, 83, 85, 86, 91, 95, 96, 159, 163,
BioRxiv	Free online archive for unpublished preprints in the life science (http://biorxiv.org/) 154, 159
CCS	Circular Consensus Sequencing. 33, 131, 159
De Bruijn graph	Directed graph representing sequence overlaps of constant width. 10, 29–34, 59, 77, 86, 102, 159
DIAG	Data Intensive Academic Grid. 23, 28, 159
DNA	Deoxyribonucleic acid. 9, 11, 24, 96, 159
FASTA	Text-based format for sequence information (http://blast.ncbi.nlm.nih.gov/blastcgihelp.shtml). 137, 138, 159
FASTQ	Text-based format for sequence and quality information. 23, 27, 130, 137–139, 159
FGCZ	Functional Genomics Center Zurich (http://www.fgcz.ch/). 27, 81, 159
GA	Illumina short reads sequencing platform. 159
GAIIX	Illumina short reads sequencing platform. 159
GWAS	(https://en.wikipedia.org/wiki/Genome-wide_association_study). 118, 159
HDF5	Portable, binary storage format for large and complex data; raw output format of PacBio sequencing instruments (http://www.hdfgroup.org/HDF5). 27, 159
HiSeq	Illumina short reads sequencing platform. 11, 25, 55, 95, 105, 159
HiSeq 2000	Illumina short reads sequencing platform. 33, 77, 159
HiSeq 2500	Illumina short reads sequencing platform. 159
HPC	High Performance Computing. 23, 159
Illumina	Biotechnological company developing sequencing instruments 23–25, 31, 33, 159
indel	Insertion / Deletion of bases in comparison of DNA sequences. 103, 119, 121, 159
insert	DNA template fragment used for Illumina sequencing 159
insert size	Length of insert fragment 159
JGI	Joint Genome Institute (http://jgi.doe.gov/). 117, 118, 159
LGC	LGC Genomics (http://www.lgcgroup.com/our-science/genomics-solutions). 16, 17, 24–29, 31–33, 41, 43, 68–75, 77, 82, 91, 96, 111, 159, 165
LRZ	Leibniz-Rechenzentrum (http://www.lrz.de/). 23, 26, 28, 30, 61, 77, 81, 159
MinION	Oxford Nanopore long read sequencing platform. 101, 159
MiSeq	Illumina short reads sequencing platform. 25, 105, 159
N50	Simple statistical measurement for assembly contiguity. 18–21, 34, 35, 38, 58, 63, 66, 77, 82–84, 86, 87, 89, 91, 95, 108, 159
NGS	Next Generation Sequencing. 53, 137, 138, 141, 159
object-oriented	Programming paradigm based on encapsulation of data in objects, attributes and object associated methods 123, 159
OLC	Overlap-Layout-Consensus assembly method based on variable width overlaps. 10, 31, 34, 86, 101, 159
Oxford Nanopore Technologies	Biotechnological company developing sequencing instruments 10, 80, 101, 159
PacBio	Biotechnological company developing sequencing instruments. 10, 101, 159
PacBio RS II	PacBio long read sequencing platform 27, 159
PCR	Polymerase Chain Reaction. 143, 159
Phred quality score	Phred quality score used to encode base call confidence in sequencing. 24, 56, 129, 130, 137, 159, 188
RNA-Seq	RNA sequencing. 141, 146, 159
SAM	Text-based TAB delimited format for reference based alignment information, produced by most current mapping programs (http://samtools.github.io/hts-specs/SAMv1.pdf). 24, 115, 123–125, 159
Sanger	Sequencing technology 31, 33, 159
SMRT	PacBio long read sequencing technology. 80, 81, 101, 103, 112, 135, 159
SMRT-bell	PacBio SMRT sequencing DNA template vii, 112–115, 159, 164, 166, 188
SNP	Single nucleotide polymorphism. 116, 119, 120, 133, 134, 159
SNV	Small nucleotide variation. 30, 71, 72, 77, 118–120, 134, 145, 159

List of Software Packages

ABySS	De Bruijn graph based short read assembler (http://www.bcgsc.ca/platform/bioinfo/software/abyss). 10, 31, 59, 76, 159, 185
ALLPATHS	De Bruijn graph based short read assembler. 30, 159
ALLPATHS	De Bruijn graph based short read assembler for large genomes (http://www.broadinstitute.org/software/allpaths-lg/blog/). ix, xi, 10, 25, 26, 30, 31, 48, 49, 59–68, 72, 75–78, 85, 86, 88, 89, 91, 95, 159, 163,
ARACHNE	Overlap-Layout-Consensus assembler for Sanger reads (https://www.broadinstitute.org/crd/wiki/index.php/Arachne_Main_Page). 10, 159
Bash	Unix shell and command language (https://www.gnu.org/software/bash/). 149, 159
BBMap	Read mapper for short reads and PacBio reads (https://sourceforge.net/projects/bbmap/). 27, 74, 104, 117, 159
bbnorm.sh	Digital normalization tool for short read data (https://sourceforge.net/projects/bbmap/). 27, 31, 74, 91, 159
BLASR	PacBio read mapper (https://github.com/PacificBiosciences/blasr). 115, 159
BLAST	Basic Local Alignment Search Tool (http://blast.ncbi.nlm.nih.gov/Blast.cgi). 114, 159
Blobology	Tool for generating taxon-annotated GC-coverage plots (https://github.com/blaxterlab/blobology). 40, 159
bowtie	Short read mapper (http://bowtie-bio.sourceforge.net/index.shtml). 159
bowtie2	Short read mapper (http://bowtie-bio.sourceforge.net/bowtie2/index.shtml). 24, 104, 159
BUSCO	Tool for completeness assessment of assemblies with respect to gene content. 35, 38, 39, 93, 159
BWA-MEM	Burrows-Wheeler-based multi-purpose mapper for short reads and longer sequences (https://github.com/lh3/bwa). 35, 103–106, 111–113, 115, 159
C	General-purpose, imperative computer programming language. 111, 159
CEGMA	Tool for completeness assessment of assemblies with respect to gene content. 35, 38, 39, 159
Celera	Software package providing De Bruijn graph as well as Overlap-Layout-Consensus assembly modules (http://wgs-assembler.sourceforge.net/). 10, 33, 76, 84, 86, 101, 107, 114, 159, 185
DALIGNER	PacBio read mapper (https://github.com/thegenemyers/DALIGNER) 115, 123, 159
DAZZLER	Overlap-Layout-Consensus assembler for PacBio reads (https://dazzlerblog.wordpress.com/). 10, 102, 159
DBG2OLC	Overlap-Layout-Consensus assembler for PacBio reads in combination with De Bruijn graph derived short read contigs (http://sourceforge.net/projects/dbg2olc/). 10, 34, 86–88, 159, 163
dextract	Tool for extracting PacBio data from HDF5 encoded binary data files. 27, 159
DEXTRACTOR	Software package for processing of HDF5 encoded PacBio data (https://github.com/thegenemyers/DEXTRACTOR). 27, 159
dipSPAdes	De Bruijn graph based short read assembler for heterozygous data (http://bioinf.spbau.ru/spades). 159, 185
Discover de novo	De Bruijn graph based short read assembler for heterozygous data (http://www.broadinstitute.org/software/discover/blog/). 33, 77, 159, 185
FALCON	Overlap-Layout-Consensus assembler for PacBio reads (https://github.com/PacificBiosciences/FALCON). 10, 86, 102, 159
FastQC	Software for quality assessment of sequencing data (http://www.bioinformatics.babraham.ac.uk/projects/fastqc/). 23, 47, 50, 68, 69, 159
FLASH	Software for merging of overlapping paired end reads (http://ccb.jhu.edu/software/FLASH/). 25, 74, 159
ggplot2	Data visualization library for R (http://ggplot2.org/). 24, 149, 153, 159
GitHub	Web-based Git repository hosting service (https://github.com/) 137, 159
globusconnect	Data transfer software for large data sets 28, 159
gmap	Splice-aware mapper for reads, ESTs and transcriptome data (http://www.molecularevolution.org/software/genomics/gmap). 159
head	Unix tool for displaying the beginning of files and data streams 139, 159
HGAP	PacBio-only assembly software with PacBio self correction (https://github.com/PacificBiosciences/Bioinformatics-Training/wiki/HGAP). 86, 101, 102, 159
IGV	Interactive visualisation tool for mapped read data and genome annotations (https://www.broadinstitute.org/igv/). 121, 159
Jellyfish	<i>k</i> -mer counting software (http://www.genome.umd.edu/jellyfish.html). 28, 141, 143, 149, 153, 159
khmer	Software package for <i>k</i> -mer based data analysis (https://pypi.python.org/pypi/khmer). 26, 74, 78, 159
kmer-coverage	Tool for generation of per-sequence <i>k</i> -mer profiles (https://github.com/thackl/kmer-scripts) 153, 155, 159
kmer-plot	Tool for visualisation of <i>k</i> -mer distributions and derived data (https://github.com/thackl/kmer-scripts) 73, 147, 149, 150, 153, 155, 159
LorDEC	<i>k</i> -mer-based Illumina-PacBio hybrid correction pipeline (http://www.atgc-montpellier.fr/lordec/). 102, 117, 159
LSC	Mapping-based Illumina-PacBio hybrid correction pipeline (http://www.healthcare.uiowa.edu/labs/au/LSC/). 102, 107–109, 159
MaSuRCA	De Bruijn graph based short read assembler (http://www.genome.umd.edu/masurca.html). 31, 76, 159, 185
Meraculous2	De Bruijn graph based short read assembler (http://jgi.doe.gov/data-and-tools/meraculous/). 32, 76, 159, 185
Meryl	<i>k</i> -mer counting software. 159
minia	De Bruijn graph based short read assembler (http://minia.genouest.org/). 32, 77, 159, 185
miniasm	De Bruijn graph based, memory efficient short read assembler (https://github.com/lh3/miniasm). 10, 159
MIRA	Overlap-Layout-Consensus assembler for accurate first and third generation reads (http://sourceforge.net/projects/mira-assembler/). 10, 33, 85, 86, 101, 159, 182, 185
normalize-by-median.py	Digital normalization tool for short read data. 74, 159
Omega	Overlap-Layout-Consensus short read assembler for metagenomes (http://omega.omicsbio.org/). 117, 159
PBcR	Mapping-based Illumina-PacBio hybrid correction pipeline (http://www.cccb.umd.edu/software/PBcR/). 81, 102, 121, 159
PBJelly	Software for scaffolding and gapclosing of assemblies with PacBio reads (https://sourceforge.net/p/pb-jelly/wiki/Home/). 34, 91, 93, 159
Perl	High-level, interpreted programming language (https://www.perl.org/) viii, 35, 111, 123, 137–139, 153, 159

Phrap Overlap-Layout-Consensus assembler for Sanger reads (<http://www.phrap.org>). 10, 159

Platanus De Bruijn graph based short read assembler for heterozygous data (<http://platanus.bio.titech.ac.jp/platanus-assembler/>). 32, 76, 159, 185

proovread Mapping-based Illumina-PacBio hybrid correction software (<https://github.com/BioInf-Wuerzburg/proovread>). ix, xi, 24, 27, 28, 33, 34, 81–83, 87, 102–110, 112–120, 123, 130, 135, 137, 139, 159, 164, 165

QUAST QUality Assessment Tool for genome assemblies (<http://quast.bioinf.spbau.ru/>). 35, 159

R Statistical programming language (<https://www.r-project.org/>). 24, 149, 153, 159

Redundans Pipeline that assists an assembly of heterozygous/polymorphic genomes. (<https://github.com/lpryszcz/redundans>). 91, 93, 159

sambamba Processing software for alignment data in BAM/SAM format (<http://lomereiter.github.io/sambamba/>). 123, 159

samtools Processing software for alignment data in BAM/SAM format (<http://www.htslib.org/>). 24, 35, 123, 159

sed Unix stream editor for parsing and transforming text data 139, 159

SeqChunker Sequence data subsetting and sampling tool (<https://github.com/BioInf-Wuerzburg/SeqChunker>) 24, 28, 138, 139, 159

SeqFilter Sequence data analysis and manipulation tool (<https://github.com/BioInf-Wuerzburg/SeqFilter>) 34, 107, 138, 159

SGE Job scheduling software for computing clusters. 159

SHRIMP-2.0 Short read mapper (<http://compbio.cs.toronto.edu/shrimp/>). 103–105, 159

sickle Quality-based short read trimming tool (<https://github.com/najoshi/sickle>) 24, 159

SLURM Job scheduling software for computing clusters. 108, 159

SMRT-Portal Analysis suite for PacBio read data (<https://github.com/PacificBiosciences/SMRT-Analysis>) 27, 113, 159

SOAP De Bruijn graph based short read assembler (<https://github.com/aquaskyline/SOAPdenovo2>). 10, 29, 31, 38, 58–61, 64–66, 75–77, 95, 117, 118, 159, 185

SPAdes De Bruijn graph based short read assembler (<http://bioinf.spbau.ru/spades>). 10, 117, 118, 159, 185

Tallymer *k*-mer counting software (<http://www.zbh.uni-hamburg.de/?id=211>). 159

trimmomatic Quality-based short read trimming tool (<http://www.usadellab.org/cms/?page=trimmomatic>). 24, 159

Velvet (<https://www.ebi.ac.uk/~zerbino/velvet/>). 10, 159

List of Tables

2.1. Raw BGI paired-end libraries	11
2.1. Raw BGI paired-end libraries	12
2.2. Raw BGI mate-pair libraries	12
2.2. Raw BGI mate-pair libraries	13
2.3. Trimmed BGI paired-end libraries	13
2.3. Trimmed BGI paired-end libraries	14
2.4. Normalized BGI paired-end libraries	15
2.5. Clipped LGC paired-end libraries	16
2.6. Trimmed LGC paired-end libraries	16
2.7. Normalized LGC paired-end libraries	17
2.8. Raw PacBio libraries	18
2.8. Raw PacBio libraries	19
2.8. Raw PacBio libraries	20
2.8. Raw PacBio libraries	21
2.9. <i>In silico</i> PacBio-derived mate-pair libraries	21
3.1. Bowtie2 parameters for estimation of insert size (is) distribution	24
3.2. Read lengths and insert sizes of overlapping libraries	25
3.3. Overlap lengths of sequenced libraries	25
3.4. Parameter settings of DBG2OLC runs	34
4.1. Metrics of assembly <i>Dm-gen-so-1</i>	38
4.2. CEGMA analysis of <i>Dm-gen-sp-1.1</i>	39
4.3. BUSCO analysis of <i>Dm-gen-sp-1</i>	39
4.4. Insert-size distribution of BGI (Beijing Genomics Institute) mate-pair libraries	49
4.5. Metrics of assembly <i>Dm-gen-so-5</i>	58
4.6. ALLPATHS (ALLPATHS-LG, Gnerre et al., 2011) kmer analysis of <i>Dm-gen-ap-3</i>	59
4.7. Metrics of assemblies <i>Dm-gen-ap-3</i> and <i>Dm-gen-ap-4</i>	60
4.8. Metrics of assembly <i>Dm-gen-ap-5</i>	61
4.9. ALLPATHS <i>k</i> -mer analysis of <i>Dm-gen-ap-6</i>	63
4.10. Metrics of assembly <i>Dm-gen-ap-6</i>	63
4.11. Metrics of assembly <i>Dm-gen-so-4</i>	66
4.12. Normalized LGC libraries	74
4.13. Metrics of merged LGC overlapping libraries	74
4.14. Metrics of assemblies <i>Dm-gen-so-9</i> to <i>Dm-gen-so-12</i>	75

4.15. Metrics of assembly <code>Dm-gen-ap-11</code>	77
4.16. ALLPATHS <i>k</i> -mer analysis of <code>Dm-gen-ap-10</code> / <code>Dm-gen-ap-12</code>	78
4.17. Metrics of assembly <code>Dm-gen-ap-10</code> and <code>Dm-gen-ap-12</code>	78
4.18. Metrics of corrected PacBio libraries	82
4.19. Metrics of assembly <code>Dm-gen-cl-1</code> and <code>Dm-gen-cl-3</code>	85
4.20. Metrics of assembly <code>Dm-gen-cl-2</code>	86
4.21. Metrics of assembly <code>Dm-gen-do-7</code>	87
4.22. Metrics of assembly <code>Dm-gen-tr-3</code>	89
4.23. ALLPATHS <i>k</i> -mer analysis of <code>Dm-gen-ap-13</code>	91
4.24. Metrics of assembly <code>Dm-gen-ap-13</code>	91
4.25. BUSCO analysis of <code>Dm-gen-ap-13</code>	93
6.1. proofread hybrid alignment scoring scheme	103
6.2. bwa-proofread command line interface	112
6.3. Accuracy of proofread-meta	117
6.4. Assembly metrics of <i>A. aerophoba</i> metagenome data	117
1. Abbreviations of assembly software used in assembly naming schemes	185
3. SAM alignment column specification, modified from SAM/BAM Format Specification Working Group, 2015	188
4. blastn parameter used for annotation of SMRT-bell chimeras by <code>siamera</code> module.	188

List of Figures

1.1. Open trap with trigger hairs and glands	6
1.2. Trap with prey after rapid closing phase	7
4.1. Transcriptome coverage of <i>Dm-gen-so-1</i>	40
4.2. Length-coverage distribution of assembly <i>Dm-gen-so-1</i>	42
4.3. Length-coverage distribution of <i>Dm-gen-so-1</i>	43
4.4. GC-coverage and length-coverage distribution of <i>=Dm-gen-so-1.1</i>	45
4.5. GC-coverage and length distribution of <i>Dm-gen-so-1.2</i>	46
4.6. Quality control of BGI paired-end and mate-pair reads	50
4.7. Per sequence GC-content of BGI read data	51
4.8. Insert-size distribution of BGI paired-end libraries	52
4.9. <i>k</i> -mer distribution of BGI paired-end reads	53
4.10. <i>k</i> -mer distribution of BGI paired-end reads	54
4.11. Heatmap of <i>k</i> -mer frequencies in BGI overlapping libraries	55
4.12. GC-bias in quality trimmed BGI libraries	56
4.13. Transcriptome coverage of assemblies <i>Dm-gen-ap-3.2</i> to <i>Dm-gen-ap-5.2</i>	61
4.14. Transcriptome content of the assembly <i>Dm-gen-ap-6</i>	64
4.15. GC-coverage and length-coverage distribution of <i>Dm-gen-ap-6.2</i>	65
4.16. <i>k</i> -mer distribution of BGI paired-end and overlapping read sets	67
4.17. Insert-size distribution of paired-end libraries	69
4.18. <i>k</i> -mer distribution of LGC reads	70
4.19. <i>k</i> -mer distribution of LGC paired-end reads	71
4.20. <i>k</i> -mer distributions of processed LGC paired-end reads	72
4.21. Heatmap of <i>k</i> -mer frequencies in LGC overlapping libraries	73
4.22. GC-coverage and length-coverage distribution of <i>Dm-gen-ap-10.2</i>	79
4.23. Transcriptome content of PacBio reads and assemblies	84
4.24. GC-coverage and length-coverage distribution of <i>Dm-gen-cl-3.1</i>	85
4.25. GC-coverage and length-coverage distribution of <i>Dm-gen-cl-2.1</i>	87
4.26. GC-coverage and length-coverage distribution of <i>Dm-gen-do-7.1</i>	88
4.27. GC-coverage and length-coverage distribution of <i>Dm-gen-tr-3.1</i>	90
4.28. GC-coverage and length-coverage distribution of <i>Dm-gen-ap-13.4</i>	93
5.1. Comparison of Milestone assemblies	96
6.1. Iterative correction procedure	106
6.2. proofread benchmark	108
6.3. Storage-efficiency of progressive best alignment filter	111

6.4.	SMRT-bell, subreads and circular consensus	113
6.5.	Effect of siamera filter trimming on assembly	114
6.6.	Consensus errors introduced by *snv	119
6.7.	Haplotype aware correction procedure	120
6.8.	proofread refine and polish	121
6.9.	Length-corrected length-normalized score	124
6.10.	Repeat and containee filters	127
6.11.	proofread state matrix and consensus	129
6.12.	Pre-consensus trimming of alignment ends	130
6.13.	Detection of chimera break points	131
7.2.	Benchmark of SeqChunker subsampling	139
7.1.	Systematic sampling with SeqChunker	139
8.1.	k -mer distribution of a <i>E. coli</i> K-12 sample	144
8.2.	k -mer distribution of a <i>E. coli</i> K-12 sample	144
8.3.	Anscombe transformation for k -mer distributions	147
8.4.	kmer-plot kcov: k -mer distribtion of data of a large diploid genome .	150
8.5.	kmer-plot kcov: Anscombe transformed k -mer distribution of different viral, high coverage sequencing samples	150
8.6.	GC-coverage plot of the <i>Hypsibius dujardini</i> draft genome	154
7.	Length-coverage distribution of Dm-gen-so-1	181

Bibliography

- Adamec, L. (1997). “Mineral nutrition of carnivorous plants: a review”. *The Botanical Review*.
- Adlassnig, W. and M. KollerPeroutka (2012). “Endocytotic uptake of nutrients in carnivorous plants”. *The Plant journal*.
- Albert, V. et al. (1992). “Carnivorous plants: phylogeny and structural evolution”. *Science*.
- Altschul, S. F. et al. (1990). “Basic local alignment search tool.” *Journal of Molecular Biology* 215, pp. 403–410.
- Anders, S. and W. Huber (2010). “Differential expression analysis for sequence count data”. *Genome Biology*.
- Andrews, S. (2015). *FastQC: A quality control tool for high throughput sequence data*. url: <http://www.bioinformatics.babraham.ac.uk/projects/fastqc/> (visited on 09/29/2015).
- Anscombe, F. (1948). “The transformation of Poisson, binomial and negative-binomial data”. *Biometrika* 35.3/4, pp. 246–254.
- Au, K. F. et al. (2012). “Improving PacBio Long Read Accuracy by Short Read Alignment”. *PLoS ONE* 7.10, e46679.
- Avery, O. T. (1946). “Studies on the chemical nature of the substance inducing transformation of Pneumococcal types”. *The Journal of experimental medicine*, pp. 89–96.
- Bailey, T. and S. McPherson (2012). *Dionaea: The Venus’s Flytrap*.
- Bankevich, A. et al. (2012). “SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing”. *Journal of Computational Biology* 19.5, pp. 455–477.
- Bartlett, J. and D. Stirling (2003). “A short history of the polymerase chain reaction”. *PCR protocols*.
- Batzoglou, S. et al. (2002). “ARACHNE: a whole-genome shotgun assembler”. *Genome Research*.
- Berlin, K. et al. (2015). “Assembling large genomes with single-molecule sequencing and locality-sensitive hashing”. *Nature Biotechnology* 33.6, pp. 623–630.

- Birol, I. et al. (2013). “Assembling the 20 Gb white spruce (*Picea glauca*) genome from whole-genome shotgun sequencing data”. *Bioinformatics* 29.12, pp. 1492–1497.
- Böhm, J. et al. (2016). “The Venus Flytrap *Dionaea muscipula* Counts Prey-Induced Action Potentials to Induce Sodium Uptake”. *Current Biology*, pp. 1–10.
- Böhm, J. et al. (2015). “Venus flytrap HKT1-type channel provides for prey sodium uptake into carnivorous plant without conflicting with electrical excitability”. *Molecular Plant*.
- Bolger, A. M. et al. (2014). “Trimmomatic: A flexible trimmer for Illumina sequence data”. *Bioinformatics* 30.15, pp. 2114–2120.
- Bombarely, A. et al. (2012). “A draft genome sequence of *Nicotiana benthamiana* to enhance molecular plant-microbe biology research”. *Molecular Plant-Microbe Interactions* 25.12, p. 120809142432005.
- Boothby, T. and J. Tenlen (2015). “Evidence for extensive horizontal gene transfer from the draft genome of a tardigrade”. *Proceedings of the . . .*
- Bradnam, K. R. et al. (2013). “Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species.” *GigaScience* 2.1, p. 10. arXiv: [1301.5406](https://arxiv.org/abs/1301.5406).
- Brown, C. T. and M. R. Crusoe (2014). “The khmer software package: enabling efficient sequence analysis”. *Figshare.Com*, pp. 2–4.
- Bruijn, N. G. de (1946). “A combinatorial problem”. *Proc. Koninklijke Nederlandse Akademie van Wetenschappen*. Vol. 49, pp. 758–764.
- Burdon-Sanderson, J. (1873). *Letter to Charles Darwin*.
- Burton, J. (2008). *N50 — Computational Research and Development Wiki at The Broad Institute*. url: <https://www.broad.harvard.edu/crd/wiki/index.php/N50> (visited on 09/27/2015).
- Bushnell, B. J. R. (2014). *BBMap short read aligner, and other bioinformatic tools*. url: <http://sourceforge.net/projects/bbmap/> (visited on 01/02/2016).
- Butler, J. et al. (2008). “ALLPATHS: De novo assembly of whole-genome shotgun microreads”. *Genome Research* 18.5, pp. 810–820.
- Cameron, K. M. et al. (2002). “Molecular evidence for the common origin of snaptraps among carnivorous plants”. *American Journal of Botany* 89.9, pp. 1503–1509.
- Chaisson, M. J. and P. A. Pevzner (2008). “Short read fragment assembly of bacterial genomes.” *Genome Research* 18.2, pp. 324–30.
- Chaisson, M. J. and G. Tesler (2012). “Mapping single molecule sequencing reads using basic local alignment with successive refinement (BLASR): application and theory.” *BMC Bioinformatics* 13.1, p. 238.
- Chapman, J. A. et al. (2015). “A whole-genome shotgun approach for assembling and anchoring the hexaploid bread wheat genome”. *Genome Biology* 16, p. 26.

- Chase, M. (1952). “Independent Functions of Viral Protein and Nucleic Acid in Growth of Bacteriophage”. *The Journal of General Physiology*.
- Chevreur, B. et al. (1999). “Genome Sequence Assembly Using Trace Signals and Additional Sequence Information”. *Proceedings of the German Conference on Bioinformatics*, 99:45–56.
- Chikhi, R. and P. Medvedev (2014). “Informed and automated k-mer size selection for genome assembly”. *Bioinformatics* 30.1, pp. 31–37. arXiv: [1304.5665](https://arxiv.org/abs/1304.5665).
- Chikhi, R. and G. Rizk (2013). “Space-efficient and exact de Bruijn graph representation based on a Bloom filter.” En. *Algorithms for molecular biology : AMB* 8.1, p. 22.
- Chin, C.-S. et al. (2013). “Nonhybrid, finished microbial genome assemblies from long-read SMRT sequencing data”. *Nature Methods* may, pp. 1–9.
- Chin, J. (2016). *FALCON*. url: <https://github.com/PacificBiosciences/FALCON> (visited on 03/09/2016).
- Chor, B. et al. (2009). “Genomic DNA k-mer spectra: models and modalities.” *Genome Biology* 10.10, R108.
- Chu, Y. and D. R. Corey (2012). “RNA sequencing: platform selection, experimental design, and data interpretation.” *Nucleic acid therapeutics* 22.4, pp. 271–4.
- Cock, P. J. A. et al. (2009). “The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants”. *Nucleic Acids Research* 38.6, pp. 1767–1771.
- Crick, F. et al. (1961). *General nature of the genetic code for proteins*.
- Darwin, C. (1875). “Insectivorous Plants Table of Contents”.
- David, M. et al. (2011). “SHRiMP2: sensitive yet practical SHort Read Mapping.” *Bioinformatics (Oxford, England)* 27.7, pp. 1011–2.
- Deorowicz, S. et al. (2014). “KMC 2: Fast and resource-frugal k-mer counting”. *Bioinformatics* 31.10, pp. 1569–1576. arXiv: [arXiv:1407.1507v1](https://arxiv.org/abs/1407.1507v1).
- D’haeseleer, P. et al. (2013). “Proteogenomic Analysis of a Thermophilic Bacterial Consortium Adapted to Deconstruct Switchgrass”. *PLoS ONE* 8.7, e68465.
- Dijkstra, E. W. (1965). “Solution of a problem in concurrent programming control”. *Communications of the ACM* 8.9, p. 569.
- Division, P. et al. (2005). “How the Venus flytrap snaps”. *Nature* 433.January, pp. 421–425.
- Dohm, J. and C. Lottaz (2008). “Substantial biases in ultra-short read data sets from high-throughput DNA sequencing”. *Nucleic Acids Research*.
- Edwards, R. A. et al. (2012). “Real Time Metagenomics: Using k-mers to annotate metagenomes”. *Bioinformatics* 28.24, pp. 3316–3317.

- Eid, J. et al. (2009). “Real-time DNA sequencing from single polymerase molecules.” en. *Science (New York, N.Y.)* 323.5910, pp. 133–8.
- Ellison, A. M. and N. J. Gotelli (2009). “Energetics and the evolution of carnivorous plants - Darwin’s ’most wonderful plants in the world’”. *Journal of Experimental Botany* 60.1, pp. 19–42.
- Ellison, A. (2006). “Nutrient limitation and stoichiometry of carnivorous plants”. *Plant Biology*.
- Ellison, A. and N. Gotelli (2001). “Evolutionary ecology of carnivorous plants”. *Trends in ecology & evolution*.
- Escalante-Pérez, M. and S. Scherzer (2014). “Mechano-Stimulation Triggers Turgor Changes Associated with Trap Closure in the Darwin Plant *Dionaea muscipula*”. *Molecular Plant*.
- Escalante-Pérez, M. et al. (2011). “A special pair of phytohormones controls excitability, slow closure, and external stomach formation in the Venus flytrap.” *Proceedings of the National Academy of Sciences of the United States of America* 108.37, pp. 15492–7.
- Everitt, B. (2002). *The Cambridge dictionary of statistics/BS Everitt*.
- Ewing, B. and P. Green (1998). “Base-calling of automated sequencer traces using phred. II. Error probabilities”. *Genome Research* 8, pp. 186–194.
- Ewing, B. et al. (1998). “Base-calling of automated sequencer traces using Phred. I. Accuracy assessment”. *Genome Research*, pp. 175–185.
- Fleischmann, R. and M. Adams (1995). “Whole-genome random sequencing and assembly of *Haemophilus influenzae* Rd”. *Science*.
- Gao, P. et al. (2015). “Integration of trap and root derived nitrogen nutrition of carnivorous *Dionaea muscipula*”. *New Phytologist*.
- Garza, D. R. and C. A. Suttle (1995). “Large double-stranded DNA viruses which cause the lysis of a marine heterotrophic nanoflagellate (*Bodo* sp.) occur in natural marine viral communities”. *Aquatic Microbial Ecology* 9.3, pp. 203–210.
- Gibson, T. C. and D. M. Waller (2009). “Evolving Darwin’s ’most wonderful’ plant: Ecological steps to a snap-trap”. *New Phytologist* 183.3, pp. 575–587.
- Givnish, T. J. (2015). “New evidence on the origin of carnivorous plants”. *Proceedings of the National Academy of Sciences of the United States of America* 112.1, pp. 10–11.
- Givnish, T. et al. (1984). “Carnivory in the bromeliad *Brocchinia reducta*, with a cost/benefit model for the general restriction of carnivorous plants to sunny, moist, nutrient-poor habitats”. *American Naturalist*.
- Gnerre, S. et al. (2011). “High-quality draft assemblies of mammalian genomes from massively parallel sequence data.” *Proceedings of the National Academy of Sciences of the United States of America* 108.4, pp. 1513–8.

- Goebel, K. (1893). *Pflanzenbiologische Schilderungen, Part 2*, pp. 161–214.
- Goodwin, S. et al. (2015). “Oxford Nanopore Sequencing and de novo Assembly of a Eukaryotic Genome”. en. *bioRxiv*, p. 013490.
- Govaerts, R. (2016). *World Checklist of Selected Plant Families - Dionaea muscipula*. url: http://apps.kew.org/wcsp/namedetail.do?name%5C_id=62109 (visited on 02/22/2016).
- Green, P. (2016). *Phrap*. url: <http://www.phrap.org> (visited on 03/09/2016).
- Gu, W. et al. (2008). “Identification of repeat structure in large genomes using repeat probability clouds”. *Analytical Biochemistry* 380.1, pp. 77–83.
- Gurevich, A. et al. (2013). “QUAST: Quality assessment tool for genome assemblies”. *Bioinformatics* 29.8, pp. 1072–1075.
- Hackl, T. et al. (2014). “proovread: large-scale high-accuracy PacBio correction through iterative short read consensus”. *Bioinformatics* 30.21, pp. 3004–3011.
- Haider, B. et al. (2014). “Omega: An Overlap-graph de novo Assembler for Metagenomics”. *Bioinformatics* 30.19, pp. 2717–2722.
- Hamilton, J. P. and C. R. Buell (2014). “Timber! Felling the loblolly pine genome.” *Genome Biology* 15.3, p. 111.
- Hodick, D. and A. Sievers (1988). “The action potential of *Dionaea muscipula* Ellis”. *Planta* 174.1, pp. 8–18.
- Huang, S. et al. (2009). “The genome of the cucumber, *Cucumis sativus* L.” *Nature Genetics* 41.12, pp. 1275–1281.
- Illumina (2016). *Illumina*. url: http://www.illumina.com/content/illumina-marketing/amr/en%5C_US/systems/sequencing.html (visited on 02/09/2016).
- Kajitani, R. et al. (2014). “Efficient de novo assembly of highly heterozygous genomes from whole-genome shotgun short reads.” *Genome Research* 24.8, pp. 1384–95.
- Kasianowicz, J. J. et al. (1996). “Characterization of individual polynucleotide molecules using a membrane channel.” *Proceedings of the National Academy of Sciences of the United States of America* 93.24, pp. 13770–13773.
- Kelley, D. R. et al. (2010). “Quake: quality-aware detection and correction of sequencing errors.” *Genome biology* 11.11, R116.
- Koch, P. et al. (2014). “RepARK - de novo creation of repeat libraries from whole-genome NGS reads”. *Nucleic Acids Research* 42.9, e80.
- Koren, S. et al. (2012). “Hybrid error correction and de novo assembly of single-molecule sequencing reads.” *Nature Biotechnology* 30.7, pp. 693–700.
- Koutsovoulos, G. et al. (2015). *The genome of the tardigrade *Hypsibius dujardini**. en. Tech. rep., p. 033464.

- Kreuzwieser, J. et al. (2014). “The Venus flytrap attracts insects by the release of volatile organic compounds”. en. *Journal of Experimental Botany* 65.2, pp. 755–766.
- Krol, E. et al. (2006). “Effects of ion channel inhibitors on cold-and electrically-induced action potentials in *Dionaea muscipula*”. *Biologia Plantarum*.
- Kulinskaya, E. et al. (2008). *Meta analysis : a guide to calibrating and combining statistical evidence*. Vol. 14. 8, pp. 560–567.
- Kumar, S. and M. L. Blaxter (2011). “Simultaneous genome sequencing of symbionts and their hosts”. *Symbiosis* 55.3, pp. 119–126.
- Kumar, S. et al. (2013). “Blobology: exploring raw genome data for contaminants, symbionts and parasites using taxon-annotated GC-coverage plots.” *Frontiers in Genetics* 4.November, p. 237.
- Kundeti, V. K. et al. (2010). “Efficient parallel and out of core algorithms for constructing large bi-directed de Bruijn graphs.” En. *BMC Bioinformatics* 11.1, p. 560.
- Lander, E. et al. (2001). “Initial sequencing and analysis of the human genome”. *Nature*.
- Langmead, B. and S. L. Salzberg (2012). “Nmeth.1923”. *Nature Methods* 9.4, pp. 357–360. arXiv: [\#14603](#).
- Langmead, B. et al. (2009). “Ultrafast and memory-efficient alignment of short DNA sequences to the human genome”. *Genome Biology* 10.3, R25.
- Levene, M. J. et al. (2003). “Zero-mode waveguides for single-molecule analysis at high concentrations.” en. *Science (New York, N.Y.)* 299.5607, pp. 682–6.
- Levin, J. Z. et al. (2010). “Comprehensive comparative analysis of strand-specific RNA sequencing methods.” *Nature Methods* 7.9, pp. 709–15.
- Levy, S. et al. (2007). “The diploid genome sequence of an individual human”. *PLoS Biology* 5.10, pp. 2113–2144.
- Li, H. (2013). “Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM”. *arXiv preprint arXiv* 00.00, p. 3. arXiv: [1303.3997](#).
- Li, H. (2015). “Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences”. *arXiv*, pp. 1–7. arXiv: [1512.01801](#).
- Li, H. and R. Durbin (2009). “Fast and accurate short read alignment with Burrows-Wheeler transform”. *Bioinformatics* 25.14, pp. 1754–1760.
- Li, H. et al. (2009). “The Sequence Alignment / Map (SAM) Format and SAM-tools 1000 Genome Project Data Processing Subgroup”. *Bioinformatics (Oxford, England)* 25.16, pp. 1–2.
- Li, R. et al. (2009). “SOAP2: An improved ultrafast tool for short read alignment”. *Bioinformatics* 25.15, pp. 1966–1967.

- Li, R. et al. (2010). “The sequence and de novo assembly of the giant panda genome.” *Nature* 463.7279, pp. 311–7.
- Li, X. and M. Waterman (2003). “Estimating the repeat structure and length of DNA sequences using -tuples”. *Genome Research*.
- Lipman, D. J. and W. R. Pearson (1985). “Rapid and sensitive protein similarity searches.” en. *Science* 227.4693, pp. 1435–1441.
- Liu, B. et al. (2013). “Estimation of genomic characteristics by analyzing k-mer frequency in de novo genome projects”. *arXiv*, p. 1308.2012. arXiv: [1308.2012](https://arxiv.org/abs/1308.2012).
- Liu, Y. et al. (2013). “Musket: A multistage k-mer spectrum-based error corrector for Illumina sequence data”. *Bioinformatics* 29.3, pp. 308–315.
- Lloyd, F. E. (1942). *The carnivorous plants*. Vol. IX, pp. 115–168.
- Lüttge, U. (1963). “Die Bedeutung des chemischen Reizes bei der Resorption von 14 C-Glutaminsäure, 35 SO 4- und 45 Ca++ durch *Dionaea muscipula*”. *Naturwissenschaften*.
- Lüttge, U. (1964). “Untersuchungen zur Physiologie der Carnivoren-Drüsen”. *Planta* 63, pp. 103–117.
- Magoc, T. and S. L. Salzberg (2011). “FLASH: Fast length adjustment of short reads to improve genome assemblies”. *Bioinformatics* 27.21, pp. 2957–2963.
- Marçais, G. and C. Kingsford (2011). “A fast, lock-free approach for efficient parallel counting of occurrences of k-mers”. *Bioinformatics* 27.6, pp. 764–770.
- Maxam, a. M. and W. Gilbert (1977). “A new method for sequencing DNA.” *Proceedings of the National Academy of Sciences of the United States of America* 74.2, pp. 560–564.
- McCorrison, J. M. et al. (2014). “NeatFreq: reference-free data reduction and coverage normalization for De Novo sequence assembly.” *BMC Bioinformatics* 15.1, p. 357.
- Melsted, P. and J. Pritchard (2011). “Efficient counting of k-mers in DNA sequences using a bloom filter”. *BMC Bioinformatics*.
- Meselson, M. and F. W. Stahl (1958). “The replication of DNA in *Escherichia coli*”. *Proceedings of the National Academy of Sciences of the United States of America* 44, pp. 671–682.
- Miescher F (1871). “Ueber die chemische Zusammensetzung der Eiterzellen”. *Medizinisch-chemische Untersuchungen* 4, pp. 441–460.
- Minoche, A. E. et al. (2011). *Evaluation of genomic high-throughput sequencing data generated on Illumina HiSeq and Genome Analyzer systems*.
- Morozova, O. and M. A. Marra (2008). “Applications of next-generation sequencing technologies in functional genomics”. *Genomics* 92, pp. 255–264.

- Myers, E. W. (1995). "Toward simplifying and accurately formulating fragment assembly." *Journal of computational biology : a journal of computational molecular cell biology* 2, pp. 275–290.
- Myers, E. W. (1986). "An O(ND) difference algorithm and its variations". *Algorithmica* 1.1-4, pp. 251–266.
- Myers, E. et al. (2000). "A Whole-Genome Assembly of Drosophila." *Science*.
- Myers, G. (2014). "Efficient local alignment discovery amongst noisy long reads". *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 8701 LNBI, pp. 52–67.
- Nagarajan, N. and M. Pop (2013). "Sequence assembly demystified." *Nature Reviews Genetics* 14.3, pp. 157–67.
- Nederbragt, A. and T. Rounge (2010). "Identification and quantification of genomic repeats and sample contamination in assemblies of 454 pyrosequencing reads". *Sequencing*.
- Needleman, S. B. and C. D. Wunsch (1970). "A general method applicable to the search for similarities in the amino acid sequence of two proteins". *Journal of Molecular Biology* 48.3, pp. 443–453.
- Nicolas, J. et al. (2016). "Finding and Characterizing Repeats in Plant Genomes." *Methods in Molecular Biology* 1374, pp. 293–337.
- Nirenberg, M. W. and J. H. Matthaei (1961). "The dependence of cell-free protein synthesis in E. coli upon naturally occurring or synthetic polyribonucleotides". *Proceedings of the National Academy of Sciences of the United States of America* 47, pp. 1588–1602.
- Nirenberg, M. et al. (1965). "RNA codewords and protein synthesis, VII. On the general nature of the RNA code." *Proceedings of the National Academy of Sciences of the United States of America* 53.5, pp. 1161–8.
- Ono, Y. et al. (2013). "PBSIM: PacBio reads simulator - Toward accurate genome assembly". *Bioinformatics* 29.1, pp. 119–121.
- Oosterhuis, J. (1927). "Over de invloed van insectenvoeding op Drosera".
- Parra, G. et al. (2007). "CEGMA: A pipeline to accurately annotate core genes in eukaryotic genomes". *Bioinformatics* 23.9, pp. 1061–1067.
- Patro, R. et al. (2014). "Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms". *Nature Biotechnology*.
- Pavlovi, A. and M. Saganová (2015). *A novel insight into the cost-benefit model for the evolution of botanical carnivory*.
- Pettersson, E. et al. (2009). "Generations of sequencing technologies." *Genomics* 93.2, pp. 105–11.

- Pevzner, P. A. et al. (2001). “An Eulerian path approach to DNA fragment assembly.” *Proceedings of the National Academy of Sciences of the United States of America* 98.17, pp. 9748–53.
- Pryszcz, L. P. and T. Gabaldón (2016). “Redundans: an assembly pipeline for highly heterozygous genomes”. *Nucleic Acids Research* submitted.
- Quail, M. et al. (2012). “A tale of three next generation sequencing platforms: comparison of Ion torrent, pacific biosciences and illumina MiSeq sequencers”. *BMC Genomics* 13.1, p. 1.
- Raphael, B. et al. (2004). “A novel method for multiple alignment of sequences with repeated and shuffled elements.” *Genome Research* 14.11, pp. 2336–46.
- Rau, A. et al. (2014). “Differential meta-analysis of RNA-seq data from multiple studies.” En. *BMC Bioinformatics* 15.1, p. 91.
- Rea, P. (1982). “Fluid composition and factors that elicit secretion by the trap lobes of *Dionaea muscipula* Ellis”. *Zeitschrift für Pflanzenphysiologie*.
- Rivadavia, F. et al. (2003). “Phylogeny of the sundews, *Drosera* (Droseraceae), based on chloroplast *rbcL* and nuclear 18S ribosomal DNA sequences”. *American Journal of Botany* 90.1, pp. 123–130.
- Robins, R. and B. Juniper (1980). “The secretory cycle of *Dionaea muscipula* Ellis”. *New Phytologist*.
- Robinson, J. T. et al. (2011). “Integrative genomics viewer”. *Nature Biotechnology* 29.1, pp. 24–26. arXiv: [ro](https://arxiv.org/abs/1011.2518).
- Ross, M. G. et al. (2013). “Characterizing and measuring bias in sequence data.” *Genome Biology* 14.5, R51.
- Roullier, C. et al. (2013). “Disentangling the Origins of Cultivated Sweet Potato (*Ipomoea batatas* (L.) Lam.)” *PLoS ONE* 8.5, e62707.
- Rumble, S. M. et al. (2009). “SHRiMP: Accurate mapping of short color-space reads”. *PLoS Computational Biology* 5.5, e1000386.
- Safonova, Y. et al. (2014). “dipSPAdes: Assembler for Highly Polymorphic Diploid Genomes”. *Research in Computational Molecular Biology* 8394.00, pp. 265–279.
- Salmela, L. and E. Rivals (2014). “LoRDEC: Accurate and efficient long read error correction”. *Bioinformatics* 30.24, pp. 3506–3514.
- SAM/BAM Format Specification Working Group (2015). “Sequence Alignment/Map Format Specification”. May, pp. 1–16.
- Sanger, F., G. M. Air, et al. (1977). “Nucleotide sequence of bacteriophage phi X174 DNA.” *Nature* 265.5596, pp. 687–695.
- Sanger, F. and A. Coulson (1975). “A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase”. *Journal of Molecular Biology*.

- Sanger, F., S. Nicklen, et al. (1977). “DNA sequencing with chain-terminating inhibitors.” *Proceedings of the National Academy of Sciences of the United States of America* 74.12, pp. 5463–7.
- Scala, J. et al. (1968). “Plastids of the Venus Flytrap (*Dionaea muscipula*)”. *Nature* 219, pp. 1183–1184.
- Scherzer, S. et al. (2013). “The *dionaea muscipula* ammonium channel DmAMT1 provides NH₄⁺ uptake associated with venus flytrap’s prey digestion”. *Current Biology* 23.17, pp. 1649–1657.
- Schmieder, R. and R. Edwards (2011). “Quality control and preprocessing of metagenomic datasets”. *Bioinformatics* 27.6, pp. 863–864.
- Schulze, W. X. et al. (2012). “The protein composition of the digestive fluid from the Venus flytrap sheds light on prey digestion mechanisms”. *Molecular & Cellular Proteomics* 11.10, pp. 1306–1319.
- Shannon, C. E. (1948). “A mathematical theory of communication”. *The Bell System Technical Journal* 27.July 1928, pp. 379–423. arXiv: [9411012](https://arxiv.org/abs/9411012) [[chao-dyn](#)].
- Shen, F. and J. Kidd (2015). *QuicK-mer: A rapid paralog sensitive CNV detection pipeline*. Tech. rep., p. 028225.
- Simão, F. A. et al. (2015). “BUSCO: Assessing genome assembly and annotation completeness with single-copy orthologs”. *Bioinformatics* 31.19, pp. 3210–3212.
- Simpson, J. T. et al. (2009). “ABySS: a parallel assembler for short read sequence data.” *Genome Research* 19.6, pp. 1117–23.
- Sinha, R. and P. Kundu (2015). “Detection of Copy Number Variations in Genomic regions”. *International Journal of Innovative Science Engineering and Technology*.
- Smith, T. and M. Waterman (1981). “Identification of common molecular subsequences”. *Journal of Molecular Biology*.
- Staden, R. (1979). “A strategy of DNA sequencing employing computer programs.” *Nucleic Acids Research* 6.7, pp. 2601–10.
- Takahashi, K. et al. (2009). “Comparative studies on the acid proteinase activities in the digestive fluids of *Nepenthes*, *Cephalotus*, *Dionaea*, and *Drosera*”. *Carnivorous Plant Newsletter* 38.September, pp. 75–82.
- Tarasov, A. et al. (2015). “Sambamba: Fast processing of NGS alignment formats”. *Bioinformatics* 31.12, pp. 2032–2034.
- Taubenfeld, G. (2004). “The Black-White Bakery Algorithm and related bounded-space, adaptive, local-spinning and FIFO algorithms”. *Distributed Computing*.
- team, A. developmental (2016). *Kmer Spectrum Primer*. url: <http://www.broadinstitute.org/software/allpaths-lg/blog/wp-content/uploads/2014/05/KmerSpectrumPrimer.pdf> (visited on 10/02/2016).

- The ALLPATHS-LG developmental team (2016). *List of publicly available ALLPATHS-LG assemblies*. url: <http://www.broadinstitute.org/software/allpaths-lg/blog/?cat=5> (visited on 03/05/2016).
- The FreeBSD Project (2016). *FreeBSD QUEUE*. url: <https://www.freebsd.org/cgi/man.cgi?query=queue%5C&sektion=3%5C&manpath=FreeBSD+10.2-RELEASE> (visited on 02/09/2016).
- Ukkonen, E. (1985). “Algorithms for approximate string matching”. *Information and control*.
- Venter, C. J. et al. (2001). “The Sequence of the Human Genome”. *Science* 291.5507, pp. 1304–1351.
- Volkov, A. G. et al. (2008). “Kinetics and mechanism of *Dionaea muscipula* trap closing.” *Plant physiology* 146.2, pp. 694–702.
- Volkov, A. G. et al. (2011). “Complete hunting cycle of *Dionaea muscipula*: Consecutive steps and their electrical properties”. *Journal of Plant Physiology* 168.2, pp. 109–120.
- Wahba, A. J. et al. (1963). “Synthetic polynucleotides and the amino acid code. VIII.” *Proceedings of the National Academy of Sciences of the United States of America* 49, pp. 116–22.
- Watson, J. and F. Crick (1953). “Molecular structure of nucleic acids”. *Nature*.
- Weisenfeld, N. I. et al. (2014). “Comprehensive variation discovery in single human genomes.” *Nature Genetics* 46.12, pp. 1350–5.
- Williams, S. and A. Bennett (1982). “Leaf closure in the Venus flytrap: an acid growth response”. *Science*.
- Worley, K. C. (2014). “Improving Genomes Using Long Reads and PBJelly 2”. English. *Plant and Animal Genome XXII Conference*. Plant and Animal Genome.
- Ye, C. et al. (2014). “DBG2OLC: Efficient Assembly of Large Genomes Using the Compressed Overlap Graph”. arXiv: [1410.2801](https://arxiv.org/abs/1410.2801).
- Zerbino, D. R. and E. Birney (2008). “Velvet: algorithms for de novo short read assembly using de Bruijn graphs.” *Genome Research* 18, pp. 821–829.
- Zimin, A. V. et al. (2013). “The MaSuRCA genome assembler.” en. *Bioinformatics (Oxford, England)* 29.21, pp. 2669–77.
- Zimin, A. V. et al. (2014). “Sequencing and assembly of the 22-gb loblolly pine genome.” *Genetics* 196.3, pp. 875–90.

APPENDIX

Length-coverage distribution of Dm-gen-so-1

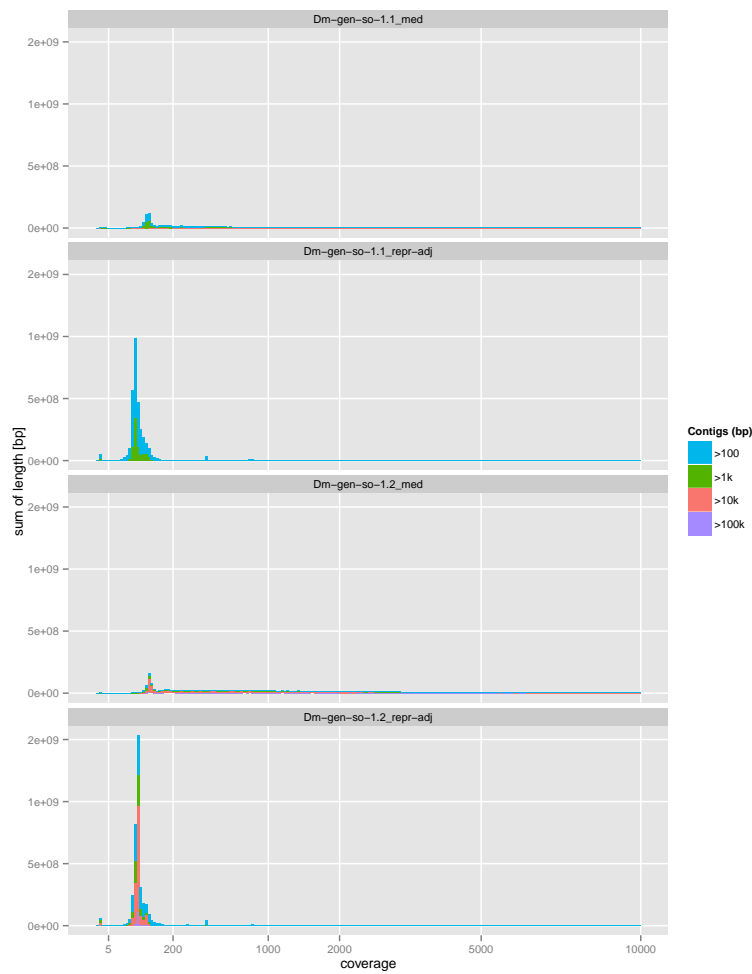


Figure 7.: Sequence length distribution of assembly *Dm-gen-so-1* as a function of raw median (panel 1 and 3) and representative adjusted k -mer coverage (panel 2 and 4). Stacked histogram bars depict cumulative length of contigs (panel 1 and 2) and scaffolds (panel 3 and 4) in different length intervals (color-coded, largest at the bottom). Coverages are scaled by Anscombe transformation (section 8.2) and limited to a maximum of $10,000x$.

Assembly parameter specifications

Listing 1: MIRA parameters

```
job = genome,denovo,draft
parameters = -GE:not=20
parameters = -NW:check_nfs=no
parameters = -HS:mnr=yes
parameters = -HS:nrc=15
parameters = PCBIOHQ_SETTINGS -CL:pec=yes
readgroup = PCBIOHQ
data = Dm_GenPb_fil15.renamed.fastq
technology = PCBIOHQ
```

Listing 2: Dm-gen-cl-1 assembly parameter

```
ovlErrorRate = 0.06
cnsErrorRate = 0.06
cgwErrorRate = 0.10
utgErrorRate = 0.015
utgErrorLimit = 2.5
doFragmentCorrection = 1
merylMemory = 100000
merylThreads = 24
mbtBatchSize = 10000000
mbtConcurrency = 24
mbtThreads = 24
ovlStoreMemory = 2048
ovlThreads = 12
ovlHashBlockLength = 100000000
ovlRefBlockSize = 5000000
ovlHashBits = 24
ovlConcurrency = 24
merOverlapperThreads = 24
merOverlapperSeedBatchSize = 100000
merOverlapperExtendBatchSize = 75000
merOverlapperSeedConcurrency = 24
frgCorrBatchSize = 2000000
frgCorrThreads = 24
frgCorrConcurrency = 24
ovlCorrBatchSize = 1000000
ovlCorrConcurrency = 24
cnsMinFrgs = 10000
cnsConcurrency = 24
doExtendClearRanges = 2
```

Listing 3: Dm-gen-cl-1 assembly parameter

```
ovlErrorRate = 0.06
cnsErrorRate = 0.06
cgwErrorRate = 0.10
```

```

utgErrorRate = 0.015
utgErrorLimit = 2.5
merylMemory = 100000
merylThreads = 40
mbtBatchSize = 10000000
mbtConcurrency = 24
mbtThreads = 2
ovlStoreMemory = 2048
ovlThreads = 2
ovlHashBlockLength = 100000000
ovlRefBlockSize = 5000000
ovlHashBits = 24
ovlConcurrency = 24
merOverlapperThreads = 2
merOverlapperSeedBatchSize = 100000
merOverlapperExtendBatchSize = 75000
merOverlapperSeedConcurrency = 24
frgCorrBatchSize = 2000000
frgCorrThreads = 2
frgCorrConcurrency = 24
ovlCorrBatchSize = 1000000
ovlCorrConcurrency = 24
cnsMinFrag = 10000
cnsConcurrency = 24
doExtendClearRanges = 2

```

Listing 4: Dm-gen-cl-1 assembly parameter

```

ovlErrorRate = 0.06
cnsErrorRate = 0.06
cgwErrorRate = 0.10
utgErrorRate = 0.015
utgErrorLimit = 2.5
merylMemory = 100000
merylThreads = 40
mbtBatchSize = 10000000
mbtConcurrency = 24
mbtThreads = 2
ovlStoreMemory = 2048
ovlThreads = 2
ovlHashBlockLength = 100000000
ovlRefBlockSize = 5000000
ovlHashBits = 24
ovlConcurrency = 24
merOverlapperThreads = 2
merOverlapperSeedBatchSize = 100000
merOverlapperExtendBatchSize = 75000
merOverlapperSeedConcurrency = 24
frgCorrBatchSize = 500000
frgCorrThreads = 2
frgCorrConcurrency = 4
ovlCorrBatchSize = 800000
ovlCorrConcurrency = 4

```

```

doExtendClearRanges = 2
cnsMinFragms = 10000
cnsConcurrency = 8
merQC=1 # compute a mer based QC report (default=0)
cleanup=none # or light or heavy or aggressive (default=none)
doToggle=1

```

Listing 5: Dm-gen-cl-1 assembly parameter

```

ovlErrorRate = 0.06
cnsErrorRate = 0.06
cgwErrorRate = 0.10
utgErrorRate = 0.015
utgErrorLimit = 2.5
gkpAllowInefficientStorage = 1
merylMemory = 100000
merylThreads = 40
mbtBatchSize = 10000000
mbtConcurrency = 24
mbtThreads = 2
ovlStoreMemory = 10000
ovlThreads = 2
ovlHashBlockLength = 100000000
ovlRefBlockSize = 5000000
ovlHashBits = 24
ovlConcurrency = 24
merOverlapperThreads = 2
merOverlapperSeedBatchSize = 100000
merOverlapperExtendBatchSize = 75000
merOverlapperSeedConcurrency = 24
frgCorrBatchSize = 500000
frgCorrThreads = 2
frgCorrConcurrency = 4
ovlCorrBatchSize = 800000
ovlCorrConcurrency = 4
doExtendClearRanges = 2
cnsMinFragms = 10000
cnsConcurrency = 8
merQC=1 # compute a mer based QC report (default=0)
cleanup=none # or light or heavy or aggressive (default=none)
doToggle=1

```


List of Assembly software abbreviations

Table 1.: Abbreviations of assembly software used in assembly naming schemes

Assembly software	Abbrev.
ABySS (Simpson et al., 2009)	ab
ALLPATHS	ap
Celera	cl
Discover de novo (Weisenfeld et al., 2014)	dc
dipSPAdes (Safonova et al., 2014)	dp
minia (Chikhi and Rizk, 2013)	mn
MIRA	mr
MaSuRCA (Zimin et al., 2013)	ms
Meraculous2 (Chapman et al., 2015)	mc
Platanus (Kajitani et al., 2014)	pt
SOAP	so
SPAdes	sp
backbone	bb
tr-guided	tr

Assembly metrics inventory

Assembly	reads	bases	max	min	N50	%N
Dm-gen-so-1.1.fa	8470199	3167489463	16745	128	514	0.00
Dm-gen-so-1.1.fa	476500	784079898	16745	1000	1626	0.00
Dm-gen-so-1.2.fa	6571343	3752455584	355463	128	3350	21.92
Dm-gen-so-1.2.fa	337902	2287688755	355463	1000	26023	35.79
Dm-gen-so-4.1.fa	1177868	671847278	24098	128	615	0.00
Dm-gen-so-4.1.fa	107022	157492207	24098	1000	1422	0.00
Dm-gen-so-5.1.fa	17834956	4779791269	30558	128	260	0.00
Dm-gen-so-5.1.fa	349327	546533018	30558	1000	1522	0.00
Dm-gen-so-7.1.fa	75538900	5632532763	16136	50	69	0.00
Dm-gen-so-7.1.fa	51753	85530928	16136	1000	1628	0.00
Dm-gen-so-7.2.fa	8204536	1479523081	16136	100	173	0.00
Dm-gen-so-7.2.fa	51753	85530928	16136	1000	1628	0.00
Dm-gen-so-9.1.fa	4678921	1210730901	12606	128	388	0.00
Dm-gen-so-9.1.fa	36686	46632545	12606	1000	1210	0.00
Dm-gen-so-10.1.fa	46280104	5007245923	31576	68	101	0.00
Dm-gen-so-10.1.fa	136524	269553853	31576	1000	2063	0.00
Dm-gen-so-10.2.fa	11651795	2545846025	33678	100	237	2.61
Dm-gen-so-10.2.fa	173965	330777880	33678	1000	1950	2.58
Dm-gen-so-11.1.fa	35100734	4591817142	36450	78	129	0.00
Dm-gen-so-11.1.fa	176794	348549133	36450	1000	2057	0.00

Dm-gen-so-11.2.fa	11365857	2723441110	36450	100	283	2.00
Dm-gen-so-11.2.fa	245254	458348838	36450	1000	1886	2.72
Dm-gen-so-12.1.fa	15706681	2449395592	14004	88	170	0.00
Dm-gen-so-12.1.fa	104239	160261473	14004	1000	1487	0.00
Dm-gen-so-12.2.fa	6725843	1716157617	15435	100	372	1.32
Dm-gen-so-12.2.fa	225751	334044066	15435	1000	1414	2.89
Dm-gen-ap-3.1.fa	130532	319449775	44489	125	3299	0.00
Dm-gen-ap-3.1.fa	121946	311310950	44489	1000	3404	0.00
Dm-gen-ap-3.2.fa	79444	356891523	112772	879	7640	10.49
Dm-gen-ap-3.2.fa	75237	352864087	112772	1000	7747	10.61
Dm-gen-ap-4.1.fa	131259	323302065	55979	244	3344	0.00
Dm-gen-ap-4.1.fa	121899	314411581	55979	1000	3462	0.00
Dm-gen-ap-4.2.fa	72656	424312843	157218	788	12445	23.81
Dm-gen-ap-4.2.fa	67321	419215516	157218	1000	12623	24.10
Dm-gen-ap-5.1.fa	82977	125963777	30348	503	1454	0.00
Dm-gen-ap-5.1.fa	69319	112970769	30348	1000	1558	0.00
Dm-gen-ap-6.1.fa	173104	464029794	91443	84	3946	0.00
Dm-gen-ap-6.1.fa	157636	449801249	91443	1000	4125	0.00
Dm-gen-ap-6.2.fa	75466	623269190	496750	880	19699	25.55
Dm-gen-ap-6.2.fa	71438	619404082	496750	1000	19814	25.71
Dm-gen-ap-8.1.fa	143703	398042183	63064	158	4237	0.00
Dm-gen-ap-8.1.fa	129479	385036546	63064	1000	4472	0.00
Dm-gen-ap-8.2.fa	70988	521916307	339789	880	17764	23.73
Dm-gen-ap-8.2.fa	66855	517959941	339789	1000	17932	23.92
Dm-gen-ap-10.1.fa	271863	718849301	81999	125	5407	0.00
Dm-gen-ap-10.1.fa	177093	657422302	81999	1000	6194	0.00
Dm-gen-ap-10.2.fa	70029	1015218851	999304	845	31814	29.19
Dm-gen-ap-10.2.fa	68704	1013945149	999304	1000	31886	29.23
Dm-gen-ap-10.3.fa	130130	838025396	312707	125	16871	14.48
Dm-gen-ap-10.3.fa	93295	814802398	312707	1000	17497	14.89
Dm-gen-ap-10.4.fa	130012	987731519	280514	125	26837	27.38
Dm-gen-ap-10.4.fa	91519	963349224	280514	1000	27735	28.07
Dm-gen-ap-11.1.fa	253506	398543028	65605	239	2492	0.00
Dm-gen-ap-11.1.fa	111335	304128659	65605	1000	3644	0.00
Dm-gen-ap-11.2.fa	76042	661574300	602575	845	17051	39.76
Dm-gen-ap-11.2.fa	73052	658714918	602575	1000	17145	39.93
Dm-gen-ap-11.4.fa	120427	517610805	193891	259	10894	23.49
Dm-gen-ap-11.4.fa	73897	487266012	193891	1000	11726	24.95
Dm-gen-ap-12.1.fa	288147	696714361	78053	78	4508	0.00
Dm-gen-ap-12.1.fa	186445	630192449	78053	1000	5307	0.00

Dm-gen-ap-12.2.fa	73733	1008033297	996199	845	29570	30.88
Dm-gen-ap-12.2.fa	72327	1006682621	996199	1000	29628	30.93
Dm-gen-ap-13.1.fa	533480	1077631653	77013	27	5014	0.00
Dm-gen-ap-13.1.fa	246538	961303673	77013	1000	5974	0.00
Dm-gen-ap-13.2.fa	104847	1454767257	1004118	809	34655	25.92
Dm-gen-ap-13.2.fa	104089	1454046846	1004118	1000	34677	25.94
Dm-gen-mn-2.1.fa	6698143	1272448859	20463	103	177	0.00
Dm-gen-mn-2.1.fa	77811	154980348	20463	1000	2117	0.00
Dm-gen-cl-2.1.fa	754009	2027352664	25605	999	3204	0.00
Dm-gen-cl-2.1.fa	754008	2027351665	25605	1000	3204	0.00
Dm-gen-do-1.1.fa	1096	11129563	31505	2204	10866	0.00
Dm-gen-do-1.1.fa	1096	11129563	31505	2204	10866	0.00
Dm-gen-do-2.1.fa	26754	315532794	54506	307	13320	0.00
Dm-gen-do-2.1.fa	26743	315525165	54506	1014	13320	0.00
Dm-gen-do-3.1.fa	56937	648708028	79214	83	13181	0.00
Dm-gen-do-3.1.fa	56885	648672206	79214	1003	13181	0.00
Dm-gen-do-4.1.fa	70532	742309688	64927	70	12340	0.00
Dm-gen-do-4.1.fa	70434	742242856	64927	1001	12340	0.00
Dm-gen-do-5.1.fa	14064	188282016	63626	533	15423	0.00
Dm-gen-do-5.1.fa	14063	188281483	63626	1000	15423	0.00
Dm-gen-do-6.1.fa	44155	573840283	83510	157	14770	0.00
Dm-gen-do-6.1.fa	44140	573830821	83510	1013	14770	0.00
Dm-gen-do-7.1.fa	84310	940091810	77231	209	13049	0.00
Dm-gen-do-7.1.fa	84230	940037689	77231	1015	13050	0.00
Dm-gen-do-8.1.fa	5960	82819863	65737	416	16845	0.00
Dm-gen-do-8.1.fa	5952	82813182	65737	1005	16845	0.00
Dm-gen-do-9.1.fa	96678	1076755135	100397	210	13127	0.00
Dm-gen-do-9.1.fa	96596	1076695894	100397	1007	13127	0.00
Dm-gen-tr-3.1.fa	42120	188765959	41097	100	5925	0.00
Dm-gen-tr-3.1.fa	38900	186836727	41097	1000	5978	0.00

SAM alignment column specification

Table 3.: SAM alignment column specification, modified from SAM/BAM Format Specification Working Group, 2015

Col	Field	Type	Brief description
1	QNAME	String	Query template NAME
2	FLAG	Int	bitwise FLAG
3	RNAME	String	Reference sequence NAME
4	POS	Int	1-based leftmost mapping POSition
5	MAPQ	Int	MAPping Quality
6	CIGAR	String	CIGAR string
7	RNEXT	String	Ref. name of the mate/next read
8	PNEXT	Int	Position of the mate/next read
9	TLEN	Int	observed Template LENgth
10	SEQ	String	segment SEQUENCE
11	QUAL	String	ASCII of Phred quality score-scaled base QUALity+33

siamera blastn parameter

Table 4.: blastn parameter used for annotation of SMRT-bell chimeras by siamera module.

parameter	initial blast	refinement of multi-HSP alignments
-dust	no	no
-strand	'minus'	'minus'
-xdrop_gap_final	25	100
-soft_masking	false	false
-gapopen	3	<i>default</i>
-gapextend	2	<i>default</i>
-penalty	-4	<i>default</i>
-culling_limit	1	1
-perc_identity	97.5	95.5

proovread-meta Omega test set

idx	Genome Name	accession	il-cov	diff	rel-cov	omega-cov
1	<i>Acidobacterium capsulatum</i> ATCC 51196	NC_012483.1	200	60	43.6	20
2	<i>Bacteroides vulgatus</i> ATCC 8482	NC_009614.1	140	40	21.8	17
3	<i>Clostridium thermocellum</i> ATCC 27405	NC_009012.1	100	30	13.1	15
4	<i>Desulfovibrio vulgaris</i> DP4	NC_008751.1	70	20	8.7	13
5	<i>Fusobacterium nucleatum</i> ATCC 25586	NC_003454.1	50	18	5.4	11
6	<i>Nitrosomonas europaea</i> ATCC 19718	NC_004757.1	32	12	3.3	9
7	<i>Porphyromonas gingivalis</i> ATCC 33277	NC_010729.1	20	8	2.2	7
8	<i>Sulfolobus tokodaii</i> 7	NC_003106.2	12	4	1.3	5
9	<i>Thermoanaerobacter pseudethanolicus</i> ATCC 33223	NC_010321.1	8	8	0.7	3
					1	