HOCHSCHULE
MITTWEIDA
University of Applied Sciences

MASTERARBEIT

Herr
Tom Thoralf Fritzsche, B.Sc.

Hybride Genom-Assemblierung des altdeutschen Schafpudels

Mittweida, November 2022

Fakultät **Angewandte Computer- und Biowissenschaften**

MASTERARBEIT

Hybride Genom-Assemblierung des altdeutschen Schafpudels

Autor:
Tom Thoralf Fritzsche

Studiengang:
Genomische Biotechnologie

Seminargruppe:
GB20-wM-M

Erstprüfer:
Prof. Dr. rer. nat. habil. Röbbe Wünschiers

Zweitprüfer:
M.Sc. Nils Schön

Einreichung:
Mittweida, 07.11.2022

Faculty of **Applied Computer Sciences and Biosciences**

MASTER THESIS

Hybrid genome assembly of the old german sheep poodle

Author:
Tom Thoralf Fritzsche

Course of Study:
Applied Computer Science

Seminar Group:
GB20-wM-M

First Examiner:
Prof. Dr. rer. nat. habil. Röbbe Wünschiers

Second Examiner:
M.Sc. Nils Schön

Submission:
Mittweida, 07.11.2022

Bibliografische Beschreibung:

Fritzsche, Tom Thoralf:
Hybride Genom-Assemblierung des altdeutschen Schafpudels. – 2022. – 91 Seiten, 26 Abbildungen
Mittweida, Hochschule Mittweida – University of Applied Sciences, Fakultät Angewandte Computer- und Biowissenschaften, Masterarbeit

Englischer Titel: Hybrid genome assembly of the old german sheep poodle

Masterarbeit, 2022

Referat:

Für die genetische Untersuchung eines Organismus ist es heutzutage unerlässlich dessen Genom vorliegen zu haben. Diese Arbeit schafft diese essentielle Grundlage für die weiteren Untersuchungen des altdeutschen Schafpudels. Die Assemblierung des Genoms wird dabei mit Illumina-Short-Reads und Nanopore-Long-Reads durchgeführt. Es werden drei verschiedene Ansätze getestet: das Short-Read-Assembly, das Long-Read-Assembly und das Hybrid-Assembly. Die Assemblies aller drei Ansätze werden im Folgenden auf ihre Qualität überprüft und miteinander verglichen, um das Assembly schrittweise zu verbessern. Das Ziel ist, dass das Assembly ähnliche Qualitäts-Metriken wie die Chromosomen-Level-Assemblies der bisher schon assemblierten Genome anderer Hundearten aufweist.

Abstract:

It is essential for the genetic study of an organism to have its genome available. This work provides this basis for further investigations of the old German sheep poodle. The assembly of the genome is performed with Illumina-short-reads and Nanopore-long-reads. Three different approaches will be tested: short-read-assembly, long-read-assembly, and hybrid-assembly. The assemblies of all three approaches will be quality tested and compared with each other to improve the assembly step by step. The goal for the assembly is to have similar quality metrics as the chromosome-level-assemblies of other canine species' genomes, that have already been assembled.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

bp ..... Basenpaare

Busco ..... Benchmarking universal single copy orthologs

BWT ..... Burrows-Wheeler Transform

DNA ..... Desoxiribonukleinsäure

Gbp ..... Gigabasenpaare

InDel ..... Insertion/Deletion

INSDC ..... International Nucleotide Sequence Database Collaboration

LINE ..... Long interspersed nuclear element

Mbp ..... Megabasenpaare

mRNA ..... messenger RNA

NCBI ..... National Center for Biotechnology Information

NGS ..... Next Generation Sequecing

N's ..... unbestimmte Basen

ONT ..... Oxford Nanopore Technologies

PCR ..... Polymerasekettenreaktion

QC ..... Qualitätskontrolle

RAM ..... Random access memory

RNA ..... Ribonukleinsäure

RNA-Seq ..... RNA-Sequencing

SINE ..... Short interspersed nuclear element

SMRT ..... Single-molecule real-time

STR ..... short tandem repeats

TB ..... Terabyte

TGS ..... Third Generation Sequencing

tRNA ..... transfer RNA

1 Einleitung

Ein Genom zu sequenzieren, ist ein Schritt, um es zu verstehen. Die Genomsequenz eines Organismus vorliegen zu haben, erleichtert dessen genetische Untersuchung beträchtlich. Zumindest ist es eine wertvolle Abkürzung, um Gene und andere konservierte Regionen zu finden, die wissenschaftliche Signifikanz haben. Welche Funktion haben die Gene? Wie arbeiten sie zusammen? Wie werden sie reguliert? Letztendlich will man verstehen, wie ein Organismus dazu in der Lage ist, das zu tun, was er kann. Und hierfür ist die Untersuchung dessen Genoms essentiell. Folglich ist es notwendig das Genom zunächst erst einmal „zusammenzubauen".

1.1 Der altdeutsche Schafpudel

Der Hund ist der beste Freund des Menschen und diese Freundschaft geht weit zurück. Botigué et al., 2017 grenzen den Startpunkt der Hundedomestizierung auf 20.000 bis 40.000 Jahre ein. Der Grund für die Domestizierung ist jedoch nicht bekannt. Die beiden führenden Theorien sind die „Commensal Scavenger Theorie" und die „Cross-Species Adoption" [Serpell, 2021]. Erstere geht davon aus, dass „freundliche" Wölfe als Aasfresser den Abfall der Menschen fraßen, was von den Menschen geduldet wurde [Hare und Woods, 2013]. Dadurch kamen sich die Menschen und Wölfe mit der Zeit näher und die Menschen begannen die Vorteile des Zusammenlebens zu sehen [Coppinger, 2001]. Das bewusste Züchten von Wölfen, um sie als Haustiere zu halten, wird von der zweiten Theorie impliziert [Serpell, 2021]. Beide Theorien halben jedoch Kritikpunkte. Die Aasfresser Theorie impliziert, dass Menschen zur Zeit der Domestizierung von Hunde in größeren Gemeinschaften zusammengelebt haben. Die Epoche der „sesshaften Landwirtschaft" begann jedoch erst vor circa 12000 Jahren, wohingegen erste zuverlässige Beweise für die Domestizierung von Hunden bereits 15000 Jahre alt sind (Balter, 2005, Benecke, 1987). Zudem war die Duldung der Wölfe nahe den Siedlungen ein großer Risikofaktor, was dies unwahrscheinlich macht [Serpell, 2021]. Zudem würde sich die Aasfresser-Theorie nicht nur auf Wölfe anwenden lassen. Füchse, Bären, Hyänen, Säbelzahnkatzen und andere Raubtiere dieser Zeitperiode wären auch an den Rückständen interessiert. Warum sollte sich also der Wolf für die Domestizierung herauskristallisieren, die anderen Tiere aber weiterhin gejagt werden [Bekoff, 2018]. Die bewusste Züchtung hat den Kritikpunkt, dass Wölfe schreckliche Haustiere sind [Serpell, 2021]. Die Zähmung von Wölfen ist nahezu unmöglich, da Wölfe unbekannte Individuen vermeiden [Serpell, 2021]. Soziale Zuneigung zu unbekannten Individuen werden nur in den ersten zwei bis drei Wochen nach der Geburt zugelassen. Und dass Menschen freiwillig und bewusst neugeborene Wölfe aufnehmen,

ist in dieser Zeitepoche unwahrscheinlich [Serpell, 2021].
Das Wölfe domestiziert wurden, ist jedoch unbestreitbar. Und aus diesem Zusammenleben von Mensch und Hund über die Jahrtausende entwickelten sich Mensch und Hund auch in ähnlicher Weise. Es wurde durch den Vergleich des Wolf- und Hundegenoms deutlich, dass durch die Domestizierung einige signifikante Unterschiede entstanden sind. Einerseits durch die Adaption von Hunden an eine stärkereiche Ernährung [Axelsson et al., 2013]. Zudem werden regulierende Gene für Serotonin und Oxytocin, welche eine beruhigende Wirkung haben, stärker exprimiert als beim Wolf, welche das Stresslevel senken und somit die soziale Annäherung des Hundes zum Menschen möglich machte [Bekoff, 2018].
Diese Nähe zwischen Mensch und Hund macht domestizierte Hunde zu einem exzellenten Model, um die genetischen Ursachen von Krankheiten zu untersuchen. Mehr als 400 vererbbare Merkmale, die analog zum Menschen sind, wurden bereits beschrieben [Jagannathan et al., 2019]. Ebenso können genetische Krankheiten, die beim Menschen vorkommen auf Krankheiten adaptiert werden, die auch beim Hund vorkommen.
Und eine dieser genetischen Krankheiten wird auch in der Forschungsgruppe von Herrn Professor Wünschiers an der Hochschule Mittweida untersucht. Dabei handelt es sich um den Kryptorchismus, den Hodenhochstand, welcher die häufigste Störung der Geschlechtsentwicklung bei Hunden darstellt [Krzeminska et al., 2020]. Diese Untersuchung wurde von Rebecca Prause begonnen in Zusammenarbeit mit der Züchterin Mechthild Jennisen-Tibbe (Institut für ganzheitliche Schafpudelzucht).
Um die Untersuchung weiter zu verbessern, soll das Genom eines männlichen Schafpudels vollständig sequenziert und assembliert werden. Dieses Genom soll als Referenz für weitere Untersuchungen dienen. Die Genome einiger anderer Hunderassen wurden bereits aufgeklärt. Begonnen hat dies mit Shadow, dem neunjähigen Pudel von Craig Venter. Jedoch konnte hier keine vollständige Genomsequenz erzeugt werden. Es wurde nur eine Genom-Coverage von etwa 80 % erreicht [Whitfield, 2003]. Das erste vollständig sequenzierte Hundegenom gehört dem Boxer Tasha und ist als CanFam1.0 bezeichnet [Lindblad-Toh et al., 2005]. Das Genom wurde über die Jahre hinweg immer weiter verbessert und ist aktuell als Dog10K_Boxer_Tasha auf dem NCBI zu finden [URL-04, 2020]. Die aktuelle Referenz für das Hundegenom ist allerdings ROS_Cfam1.0, da es sich hier um einen Rüden handelt [URL-01, 2020]. Tasha ist eine Hündin und folglich war die Sequenz des Y Chromosoms bisher unbekannt. Das Hundegenom besteht aus einem diploiden Chromosomensatz mit 38 Autosomen, sowie den Geschlechtschromosomen [Lindblad-Toh et al., 2005]. Folglich sind durch das Assembly 40 Chromosomensequenzen zu erwarten. Die Genomgrö-

ße beträgt in etwa 2,4 Gbp mit einem durchschnittlichen GC-Gehalt von 41 % [Lindblad-Toh et al., 2005].

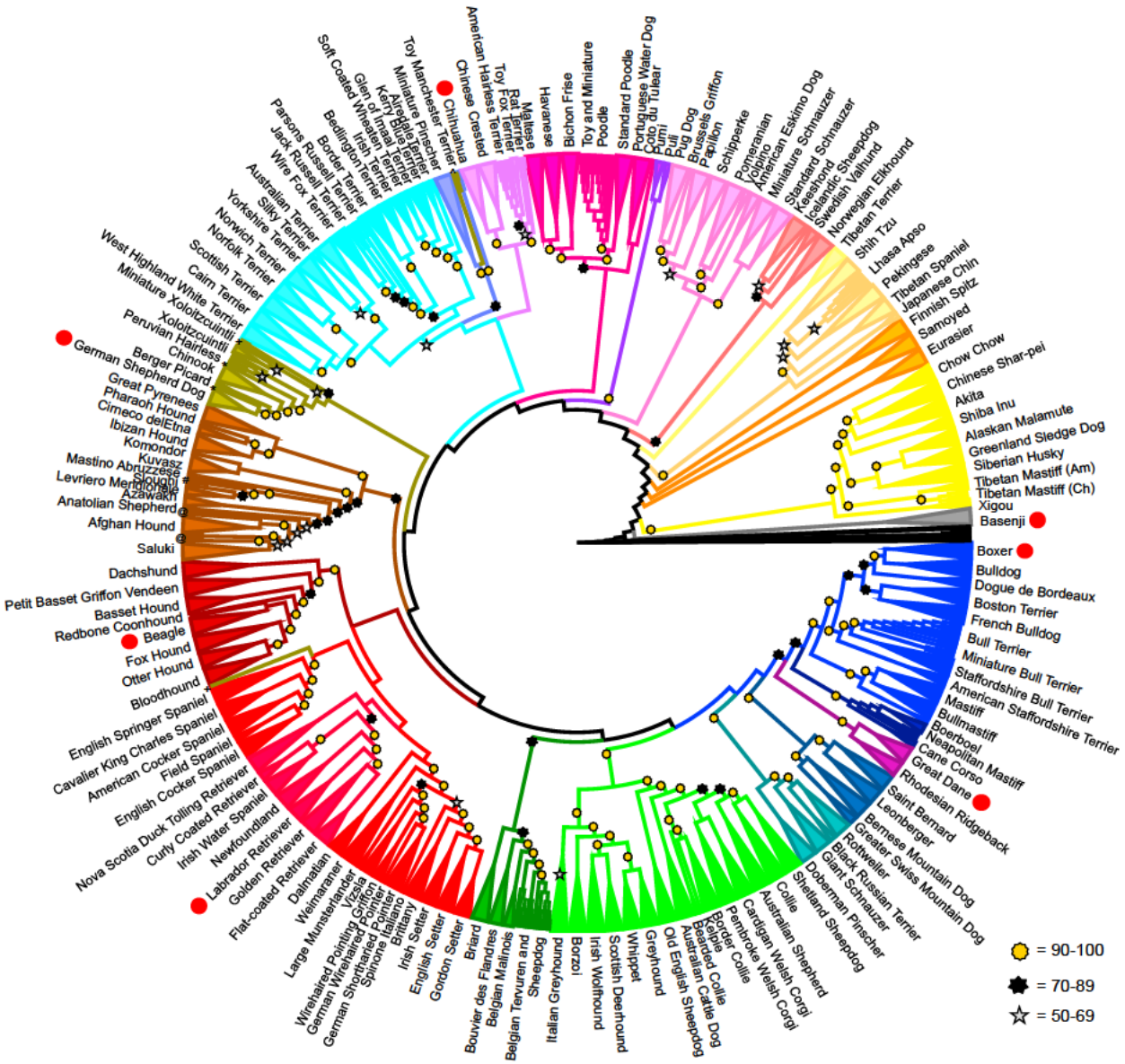

Abbildung 1.1: Kladogramm der Haushunderassen, modifiziert nach [Parker et al., 2017]

Lindblad-Toh et al., 2005 berichten, dass domestizierte Hunde sich zwar stark in ihren Aussehen unterscheiden, deren Genome jedoch zu 99,85 % identisch sind. Sie unterstreichen dies mit der Aussage: *„A dog is a dog in a genomic sense"* [Hooper, 2005]. Die geringen genetischen Unterschiede trotz der großen Unterschiede im Aussehen sind auch bei der Anzahl der Gene zu beobachten, die für ein bestimmtes Merkmal verantwortlich sind. Beim Menschen sind über 100 Gene bekannt, welche die Größe beeinflussen können, wohingegen laut Plassais et al., 2019 nur etwa 15 bis 20

Gene Einfluss auf die Größe des Hundes haben. Diese geringen genetischen Unterschiede machen im evolutionären Kontext aber Sinn. Einer der frühesten bekannten Menschen (*Homo habilis*) lebte vor etwa 2,4 Millionen Jahren und die ersten modernen Menschen (*Homo sapiens*) entwickelten sich vor etwa 300.000 Jahren. Wie bereits erwähnt, startete die Hundedomestizierung vor etwa 20.00 – 40.000 Jahren [Botigué et al., 2017]. Die meisten modernen Hundearten wurden sogar erst Mitte des 19. Jahrhunderts gezüchtet [Olmert, 2018]. Elaine Ostrander sagt: *„That's an evolutionary drop in the bucket"* [Neff, 2019]. Beim Menschen beeinflussen folglich viele Gene die Größe ein bisschen, wohingegen beim Hund wenige Gene die Größe stark beeinflussen [Plassais et al., 2019]. Dennoch ist die genetische Variation zwischen verschiedenen Hunderassen mit 27,5 % größer als zwischen verschiedenen Populationen beim Menschen mit nur 5,4 % [Ostrander, 2017].
In Abbildung 1.1 ist das Kladogramm der Hunderassen dargestellt [Parker et al., 2017]. Die sieben roten Punkte geben dabei die Rassen an, die als Referenzen in dieser Arbeit verwendet werden. Dabei wird deutlich, dass mit den bisher erstellten Genom-Assemblies des Hundes das gesamte Kladogramm abgedeckt wird. Somit ist die Wahrscheinlichkeit hoch, dass auch der Schafpudel mit einer der Referenzen nahe verwandt ist, auch wenn dieser nicht im Kladogramm vorkommt.

1.2 Sequenziermethoden

Die Entdeckung der DNA [Miescher, 1869], deren Funktion als Träger der Erbinformation [Avery et al., 1944] und deren strukturelle Aufklärung [Watson und Crick, 1953] legten den Grundstein für die Möglichkeit des Ablesens der Informationen, die einen Organismus ausmachen. Dies ermöglichte die erste Sequenzierung, wobei es sich um die Aufklärung des tRNA Moleküls für die Aminosäure Alanin handelte [Holly et al., 1965]. Die erste vollständige Sequenzierung eines Genoms erfolgte bei der Bakteriophage PhiX174 [Sanger et al., 1977]. Mittels Sanger Sequenzierung war es, durch das Lesen einzelner Sequenzabschnitte und dem Überlappen dieser zu einer langen ununterbrochenen Sequenz erstmals möglich, ein Genom vollständig zu entschlüsseln (Sanger et al., 1977; URL-02, 2021).
Diese Sequenziermethoden der ersten Generation legten den Grundstein für zahlreiche Anwendungen, unter anderen die Hochdurchsatzmethoden, die aktuell verwendet werden. Die darauf basierenden Sequenziermethoden des Next Generation Sequecing (NGS) bieten massive parallele Sequenzierungstechnologien, die einen hohen Durchsatz, hohe Skalierbarkeit und hohe Geschwindigkeit erbringen. [URL-03, 2021].
NGS hat die biologischen Wissenschaften revolutioniert und es Laboren ermöglicht, eine Vielzahl von Anwendungen durchzuführen und biologische Systeme auf einem nie zuvor möglichen Niveau

zu untersuchen. Mit NGS können zum Beispiel Genome schnell und bestimmte Zielregionen tief sequenziert werden. Zudem ist es mittels RNA-Sequenzierung (RNA-Seq) möglich mRNA in Genexpressionsanalysen zu quantifizieren [URL-03, 2021].
Trotz des hohen Durchsatzes und der weit verbreiteten Anwendungen des NGS wurden Limitierungen dieser Technologien deutlich. Vor allem die kurze Länge der Reads erschwert das Assembly von Genomen [Earl et al., 2011]. Dadurch entstanden die Sequenziermethoden des Single-molecule real-time (SMRT) Sequencing, welche schnell qualitativ hochwertige Long-Reads erstellen können. Mit Oxford Nanopore können Reads im Mbp Bereich erstellt werden. Der längste Read, der bisher erstellt wurde, ist 2.272.580 bp lang [Payne et al., 2018]. Diese Technologien benötigen keine PCR-Amplifikation während der Präparation der Library, was die Fehlerquote verringert und eine gleichmäßigere Verteilung bei der Abdeckung des Genoms ermöglicht [van Dijk et al., 2018]. Die Auswahl der zu verwendeten Sequenziertechnologie ist ein wichtiger Schritt, um ein vollständiges und sinnvolles Genom zu erhalten. Der entscheidende Faktor hier ist die vorhandene Zeit und das Budget. Hierbei sollten die später verwendeten Tools für das Assembly auch schon mit in Betracht gezogen werden, da die meisten Assembler nur mit spezifischen Sequenzdaten umgehen können. Die Illumina und Oxford Nanopore Sequenzierung werden in den folgenden Kapiteln kurz beschrieben.

1.2.1 Illumina Sequenzierung

Die Illumina Technologien basieren auf dem Prinzip der Sequenzierung durch Synthese [URL-19, 2017]. Sie laufen im Wesentlichen in vier Schritten ab. Nach der Extraktion der genomischen DNA wird diese fragmentiert. In der Library Preparation werden Adapter an die Fragmente ligiert und diese klonal amplifiziert. Diese mit Adaptern versehenen Fragmente werden anschließend in Clustern durch Hybridisierung an einen Gegenstrang komplementär zum Adapter, der an der Oberfläche der Flow Cell im Sequencer gebunden ist, immobilisiert. Die immobilisierten Fragmente werden anschließend sequenziert. Im Detail bedeutet dies, dass von einer Polymerase unterschiedlich fluoreszenzmarkierte Nukleotide den Gegenstrang des Fragments ergänzen. Dabei wird zyklenweise nur eines der fluoreszenzmarkierten Nukleotide zur Verfügung gestellt. Am Ende jedes Zyklus werden Bilder bei den jeweiligen optimalen Anregungswellenlängen der Fluorophore gemacht. Die Fluoreszenzsignale werden anschließend in Intensitäten umgewandelt und den jeweiligen Clustern zugeordnet. Die höchste Intensität in einem Cluster in einem Zyklus wird in die dazugehörige Base umgewandelt. Dies wird in Echtzeit von Software in den Sequencern durchgeführt [Cacho et al., 2016]. Dieses sogenannte Basecalling erfolgt direkt nach dem Einbau des Nukleotids. Das Fluorophor wird abgespalten und ausgewaschen, sodass das nächste Nukleotid eingebaut werden

kann. Es werden Milliarden von Nukleotiden zur gleichen Zeit, parallel sequenziert [URL-19, 2017]. Der NovaSeq 6000 kann bis zu 6 TB bei einer Maximallänge von 2 x 250 bp in 13 bis 44 Stunden Laufzeit erzeugen [URL-07, 2022]. Je länger die Reads, desto einfacher ist das Assembly, da längere sich überlappende Regionen entstehen. Je länger die Reads sind, desto wahrscheinlicher ist es, dass Fehler in den Reads vorkommen [Pfeiffer et al., 2018]. Laut Stoler und Nekrutenko, 2021 wurde für den NovaSeq bei 239 Proben eine Fehlerrate mit einem Median von 0.109 % und einer Standardabweichung von 0,35 % bestimmt.

1.2.2 Oxford Nanopore Sequenzierung

Als Teil des Third Generation Sequencing (TGS) ist die Oxford Nanopore Sequenzierung eine recht neue Technologie. Der Ansatz hinter der Technologie wurde aber bereits 1989 von David Deamer beschrieben [Deamer et al., 2016]. Er beschreibt, dass mittels eines elektrischen Feldes ein Einzelstrang RNA oder DNA durch einen 2,6 nm großen Ionenkanal einer Doppellipidschicht gezogen werden kann. Aufgrund des Durchmessers der Pore kann kein Doppelstrang durch die Pore wandern. Dadurch entsteht ein Ionengradient an der Membran. Während des Durchwanderns wird die Pore teilweise blockiert. Diese teilweise Blockade verringert das Durchschreiten von anderen Ionen durch die Pore, was den Ionenstrom verringert. Diese Änderung ist für jede Basenabfolge, die gerade die Pore blockieren, anders [Kasianowicz et al., 1996]. Diese Stromflussänderung wird von der MinKNOW-Software aufgezeichnet. Die prozessive Bewegung der Basen durch die Pore führt zu einer kontinuierlichen Stromänderung, die als „Squiggle“ bekannt ist. Die Software MinKNOW verarbeitet den Squiggle in Echtzeit in Reads, wobei jeder Read einem einzelnen DNA/RNA-Strang entspricht. Diese werden in Fast5-Dateien geschrieben [URL-08, 2021]. Das in Fast5-Dateien gespeicherte Signal wird von Basecalling-Algorithmen verarbeitet, um die Basensequenz in Fastq-Dateien zu decodieren. Guppy, der in MinKNOW integrierte Basecaller, führt Basecalling live während des Laufs durch, nachdem ein Lauf beendet wurde, oder einer Kombination aus beidem [URL-08, 2021]. Das Resultat ist eine Datei im Fastq-Format, welche sowohl die Nukleotid-Sequenz als auch Qualitätswerte für jedes Nukleotid enthält. Einige Firmen haben Nanoporen-basierte Sequenzier Strategien vorgeschlagen. Oxford Nanopore Sequencing war jedoch die erste Strang-basierte Sequenziermethode, die erfolgreich von unabhängigen genomischen Laboratorien angewandt werden konnte [Jain et al., 2016]. Jedoch ist die Fehlerrate beim Basecalling, also die Übertragung der Änderung des ionischen Stromflusses zu der entsprechenden Base, deutlich höher als beim NGS. Delahaye und Nicolas, 2021 berichten von Fehlerraten von

1,6 - 2,7 % für Deletionen, 1,2 - 2,2 % bei Mismatches und 1,1 - 2,4 % bei Insertionen. Da jedoch das gesamte DNA-Molekül durch die Pore gezogen wird, können lange Reads mit mehreren Millionen Basen entstehen [Payne et al., 2018].

1.3 Herausforderungen des *de novo* Assemblies

Beim *de novo* Genom-Assembly wird ein Genom ohne eine Vorlage, bzw. Referenz von Grund auf neu zusammengebaut. Im Gegensatz dazu steht das Referenz basierte Genom-Assembly, wobei eine Vorlage notwendig ist. Das *de novo* Assembly ist ein nicht triviales Problem, welches viele Rechenressourcen und hohe Standards erfordert. Hier werden vier wesentliche Probleme des *de novo* Assemblies zusammengefasst.

1.3.1 Fehlerkorrektur

Die erste Herausforderung besteht in der Korrektur von Sequenzierfehlern, da diese die Verlängerung der Contig und Scaffolds behindern [Sohn und Nam, 2016]. Je nachdem welche Sequenziertechnologie Anwendung findet, ist mit unterschiedlichen Fehlerarten und Fehlerhäufigkeiten zu rechnen, wodurch unterschiedliche Ansätze nötig sind, um die Fehler zu korrigieren. In dieser Arbeit ist dabei zwischen den Long-Reads von Oxford Nanopore Technologies (ONT) und den Short-Reads von Illumina zu unterscheiden.
Als Grundsatz in der Sequenzierung gilt: Je länger die zu sequenzierenden Moleküle sind, desto höher ist die Wahrscheinlichkeit, dass Fehler eingebaut werden [Pfeiffer et al., 2018]. Bei den Long-Reads ist dies besonders problematisch, da hier Read-Längen von mehreren Millionen Basen entstehen können [Delahaye und Nicolas, 2021]. Aufgrund dessen verhindert diese hohe Fehlerrate eine genaue Analyse einzelner Nukleotide der Sequenzen. Aus diesem Grund ist die Fehlerkorrektur beim Long-Read-Assembly essentiell [Zhang et al., 2020]. Bei ONT Reads sind die Fehler allerdings nicht zufällig, sondern sind in gewisser Weise verzerrt. Jain et al., 2016 berichten, dass Substitutionen von A zu T und umgekehrt weniger häufig auftreten als die anderen Substitutionen und dass InDels tendenziell häufiger in Homopolymerregionen auftreten. Diese Fehlereigenschaften stellen eine Herausforderung für Long-Read-Datenanalysen dar, insbesondere für den Nachweis korrekter Read-Überlappungen während des Genom-Assemblies. Fehlerkorrektur Algorithmen wurden entwickelt, um Sequenzierungsfehler zu identifizieren und zu beheben oder zu entfernen, wodurch *de novo* Sequenzierung begünstigt wird. [Zhang et al., 2020]. Da die Fehlerkorrektur beim Long-Read-Assembly essentiell ist, sind viele Long-Read-Assembler entwickelt wurden, die einen Error Correction Schritt bereits mit integriert haben (Canu und HGAP) [Zhang et

al., 2020]. Ansätze hierfür sind die Selbstkorrektur anhand von Informationen der Überlappungssequenzen oder die Korrektur anhand von zusätzlichen Short-Reads (Hybridmethoden), was jedoch eine Sequenzierung mit zwei Sequenziermethoden voraussetzt. Die bestperformanten Hybridmethoden (z.B. FMLRC) erreichen eine Genauigkeit von mehr als 99 % [Wang et al., 2018]. Jedoch muss dabei beachtet werden, dass diese hohe Genauigkeit auch mit einer großen Read-Depth und einem hohen N50 einhergeht. Dies bedeutet, dass die Tools nicht nur die fehlerhaften Reads trimmen oder komplett entfernen, sondern Fehler beheben. Ansonsten könnte es zu der Entfernung vieler Sequenzen einer bestimmten Genomregion kommen, was die Assembly fragmentieren würde [Zhang et al., 2020]. Bei der nicht-hybriden Korrektur besteht die Herausforderung, dass sie ausschließlich auf Überlappungen zwischen fehlerhaften Long-Reads beruht, dennoch können diese Tools auch eine hohe Genauigkeit liefern. Nicht-Hybrid-Verfahren können jedoch eine geringere Alignmentidentität erreichen, wenn die Long-Reads fehlerhafter sind [Zhang et al., 2020].
Ein weiteres Problem besteht bei der Evaluierung der Error-Correction Tools. Die Tools werden fast ausschließlich nach ihrer Fähigkeit zur Behebung von Fehlern bewertet. Das Ziel ist es aber, dass durch die Fehlerbehebung das Assembly verbessert werden soll. Folglich wäre eine Bewertung nach diesem Kriterium angebrachter.
Heydari et al., 2017 berichten, dass eine geringere Fehlerrate in den Short-Reads nicht unbedingt zu besseren Assembly Ergebnissen führt, da die meisten Fehler einmalig sind und von den Assemblern aufgrund des dahinter liegenden Assembly Prozesses (z.B. De Bruijn Graph) auch ohne Error Correction entfernt werden. Die wirklich problematischen Fehler, welche zur Verbindung zweier verschiedener Genomregionen führen, werden nur selten richtig korrigiert. Diese Fehler führen daher zu unechten „chimären“ Verbindungen zwischen Knoten im De Bruijn-Graphen, die ansonsten in der ursprünglichen Genom-Sequenz weit entfernt liegen. Weitere Probleme treten bei Bereichen mit geringer Abdeckung auf, sodass die Assembler keine Überlappungsbereiche erkennen können. Gleiches gilt für Bereiche mit einem besonders hohen oder niedrigen GC-Gehalt [Sohn und Nam, 2016].
Zusammengefasst kann man sagen, dass, wenn ein Error Correction Tool zwar alle einzigartigen Fehler beheben kann, aber keine der problematischen Fehler, die Fehlerbehebungsrate in der Evaluation zwar hoch ist, der Assembly Prozess aber nicht verbessert wird [Heydari et al., 2017]. Es besteht sogar die Möglichkeit, dass neue Fehler in die Sequenzdaten eingebracht werden. Besonders dann, wenn systematisch der gleiche Fehler in einem bestimmten Kontext gemacht wird, kann des Assembly fragmentiert oder sogar fehlerhaft sein [Heydari et al., 2017]. Heydari et al., 2017 berichten zudem, dass die Error Correction Tools zwar einen erheblichen Teil der Sequenzierungsfehler von Illumina Daten korrigieren, die Assembler allerdings nicht immer davon profitieren, da

zum Teil kürzere Contig-Längen nach dem Assembly mit vorheriger Error Correction entstanden sind. Dies bezieht sich vor allem auf die Tools die mit k-meren arbeiten. Tools die auf Multiplen Sequenz Alignments basieren, können in dieser Hinsicht bessere Ergebnisse liefern, allerdings benötigen diese signifikant mehr Rechen-Ressourcen [Heydari et al., 2017]. Die wesentliche Frage bei der Korrektur von Sequenzfehlern besteht darin, Sequenzfehler zu identifizieren und von den heterozygoten Allelen zu unterscheiden [Heydari et al., 2017].

1.3.2 Repetitive Strukturen

Die Nicht-Zufälligkeit eines Genoms durch die repetitiven Elemente (STR, LINE, SINE oder hochrepetitive Bereiche nahe den Zentromeren und Telomeren) erschwert den Assembly-Prozess signifikant [Sohn und Nam, 2016]. Diese hochrepititiven Regionen erzeugen oft zahlreiche mehrdeutige Pfade und Lücken im sich ergebenden Assembly. Einige dieser Probleme können durch eine hohe Read-Depth oder durch die Verwendung von Long-Reads gelöst werden. Wenn dies nicht der Fall ist, kann durch die Anzahl der Reads, die eine bestimmte Art an Repeats enthält, grob geschätzt werden, wie groß die Repeat-Region sein sollte. Dieser Ansatz ist aber nicht ausreichend, um eine genaue Wiederholungszahl zu bestimmen [Sohn und Nam, 2016].

1.3.3 Ungleichmäßige Read-Depth

Ungleichmäßige Read-Depth entstehen durch Limitationen in den Sequenziertechnologien, vor allem des NGS. Diese führen zu Lücken im Draft-Assembly. Eine Lösung dieses Problems ist jedoch aufgrund fehlender Informationen in diesen Bereichen nur schwer möglich. Nur mithilfe des Multiplen k-mer Ansatzes (siehe Kapitel 1.4.3), wobei sowohl kurze als auch lange k-mere genutzt werden, konnten teilweise dieses Problem gelöst werden, jedoch nicht vollständig. Zudem kann die Verwendung von kurzen k-meren zu unvorhergesehenen Misassemblies führen [Sohn und Nam, 2016].

1.3.4 Rechen Ressourcen

Je größer das Genom des zu untersuchenden Organismus ist, desto größer sind die Rechenkosten. Zum einen bezieht sich das auf die benötigte RAM und zum anderen auf die Rechenzeit. Bei Bakteriengenomen ist dies überschaubar, da diese von der Genomgröße her gesehen sehr klein sind [Chapman et al., 2011]. Der benötigte Arbeitsspeicher für das Speichern einer der k-mer Tabellen beträgt laut Chapman et al., 2011 für das Cereibactergenom zwischen 250 und 1000 Mbp. Ein menschliches Genom würde aber bereits mindestens 500 Gb RAM benötigen. Die Rechenzeit

ist Assembler- und Ansatz-abhängig. Auch hier ist die Genomgröße des Organismus entscheiden. Für ein menschliches Genom kann von einer Assembly Dauer von einigen Tagen bis zu mehreren Wochen ausgegangen werden [Luo et al., 2012]. Da der Schafpudel ein vergleichbar großes Genom besitzt, kann von ähnlichen Werten ausgegangen werden [Lindblad-Toh et al., 2005].

1.4 Assembly-Methoden

Die Auswahl des Assemblers ist abhängig von der vorhandenen Zeit, dem Budget und den Sequenzdaten. Bei der Genom Assemblierung ist es zudem wichtig zwischen *de novo* und Referenz-basierten Ansätzen zu unterscheiden [Pop, 2009]. Erstere zielen darauf ab Genome zu rekonstruieren, die keinem zuvor sequenzierten Organismus ähnlich sind, und letztere, auch Re-Sequenzierungs-Ansätze genannt, nutzen die Sequenz eines eng verwandten Organismus als Orientierungshilfe während des Assemblies.
De novo Ansätze sind deutlich schwieriger zu lösen, da keine effiziente rechnerische Lösung bekannt ist. Referenz basierte Ansätze sind einfacher zu lösen, da es im Wesentlichen ausreicht den Datensatz an Reads auf das Referenzgenom zu alignieren. Wenn es allerdings Bereiche im Genom gibt, die sich deutlich von der Referenz unterscheiden, dann können diese Regionen nur mithilfe des *de novo* Assemblies rekonstruiert werden [Pop, 2009]. Im Folgenden werden drei der Ansätze für die Assemblierung kurz erklärt.

1.4.1 Greedy Approach

Die Greedy-Algorithmen sind die einfachste und intuitive Lösung eines Assemblyproblems. Dabei werden die einzelnen Reads iterativ zu Contigs zusammengesetzt, beginnend mit den Reads, die sich am Besten überschneiden. Sie enden, wenn keine Überlappungen zwischen Reads mehr festgestellt werden kann. Das große Problem ist, dass das Prinzip auf einer lokalen Optimierung der Zielfunktion beruht. Das bedeutet, dass die zwei sich am besten überlappenden Reads verbunden werden. Dies verhindert, dass es zu einer globalen optimalen Lösung kommt, da vor allem Repeat-Regionen falsch zusammengefügt werden [Pop, 2009]. Ein weiteres Problem ist, dass das paarweise Alignment zwischen einem Satz an Reads berechnet werden muss, was sehr zeitintensiv ist. Vor allem in den Repeat-Regionen ist die benötigte Zeit proportional zum Quadrat der Anzahl der Reads. Folglich gilt: Je mehr Reads, desto länger dauert das Assembly und desto mehr Arbeitsspeicher wird benötigt. Die Länge der Reads hat allerdings nur eine geringe Wirkung auf Zeit und Arbeitsspeicher. Folglich wäre es sinnvoller Sequenziermethoden zu wählen, die längere Reads produzieren, wenn ein Assembler benutzt wird, der mit dem Greedy-Algorithmus arbeitet [Pop, 2009].

1.4.2 Overlap-Layout-Consensus Approach

Dieser Ansatz beginnt ebenfalls mit der Erstellung der paarweisen Alignments zwischen den Reads. Mithilfe dieser Informationen wird ein Überlappungsgraph erstellt. Dabei ist jeder Read als Knoten dargestellt und eine Kante verbindet zwei Knoten, wenn eine Überlappung festgestellt wurde. Ein solcher Graph ist beispielhaft in Abbildung 1.2 dargestellt. Während der Layout Phase wird dieser Graph analysiert. Das Ziel ist es, dass ein Weg durch den Graphen gefunden wird, der jeden Knoten genau einmal durchquert, was der Rekonstruktion des Genoms unter Verwendung aller Reads entsprechen würde [Compeau et al., 2011]. Der letzte Schritt ist die Erzeugung einer Konsensus-Sequenz. Anhand der Wahrscheinlichkeit des Auftretens einer Base an einer bestimmten Position in verschiedenen Reads wird entschieden, welche Base in der Konsensus-Sequenz eingebaut wird [Pop, 2009]. Ebenso ist die Verwendung von längeren Reads hier vorteilhaft, aus gleichem Grund wie beim Greedy-Algorithmus [Sohn und Nam, 2016].

1.4.3 De Bruijn Graph Approach

Die bisher angesprochenen Ansätze haben als großen Nachteil, dass sie nur schwer mit einer großen Anzahl an Reads umgehen können. Der Ansatz der auf den De Bruijn Graphen beruht, wurde speziell dafür entwickelt mit einer großen Anzahl an kurzen Reads umzugehen. Da Illumina eine der weitverbreitetsten Methoden zur Sequenzierung ist und da diese Methode eine sehr hohe Anzahl an kurzen Reads in kurzer Zeit produzieren kann, werden Assembler, die auf De Bruijn Graphen basieren, am häufigsten angewandt, besonders bei der Assemblierung von humanen Genomen [Lander et al., 2001].

Ebenso wie der Overlap-Layout-Consensus Ansatz ist dieser Ansatz auf Graphen basierend. Der Unterschied ist jedoch, dass nicht die gesamte Länge der Reads als ein Knoten dargestellt wird, sondern die Reads in k-mere unterteilt werden [Compeau et al., 2011]. Dies ist in Abbildung 1.2 dargestellt. Ein Read wird dabei so unterteilt, dass von links beginnend das erste k-mer einer bestimmten Länge erzeugt wird. Das nächste k-mer, ausgehend von dem davor erzeugten k-mer, ist um eine Position nach rechts verschoben, sodass das erste Nukleotid entfernt wird und am Ende ein anderes angehangen wird. Es gibt nun zwei Ansätze, wie ein Weg durch den Graphen gefunden werden kann. Der Erste, unter Abbildung 1.2c dargestellt, ist der Hamitonische Zyklus, bei welchem jeder Knoten nur einmal durchquert wird und der Startpunkt auch der Endpunkt ist, wodurch ein Zyklus entsteht. Dieser Ansatz stellt zwar das Genom wieder her, lässt sich allerdings nicht auf große Graphen skalieren [Compeau et al., 2011]. Moderne Short-Read-Assembly-Algorithmen basieren zumeist auf Eulerschen Zyklen (Abb. 1.2d). Dabei werden die Präfixe und Suffixe der k-mere

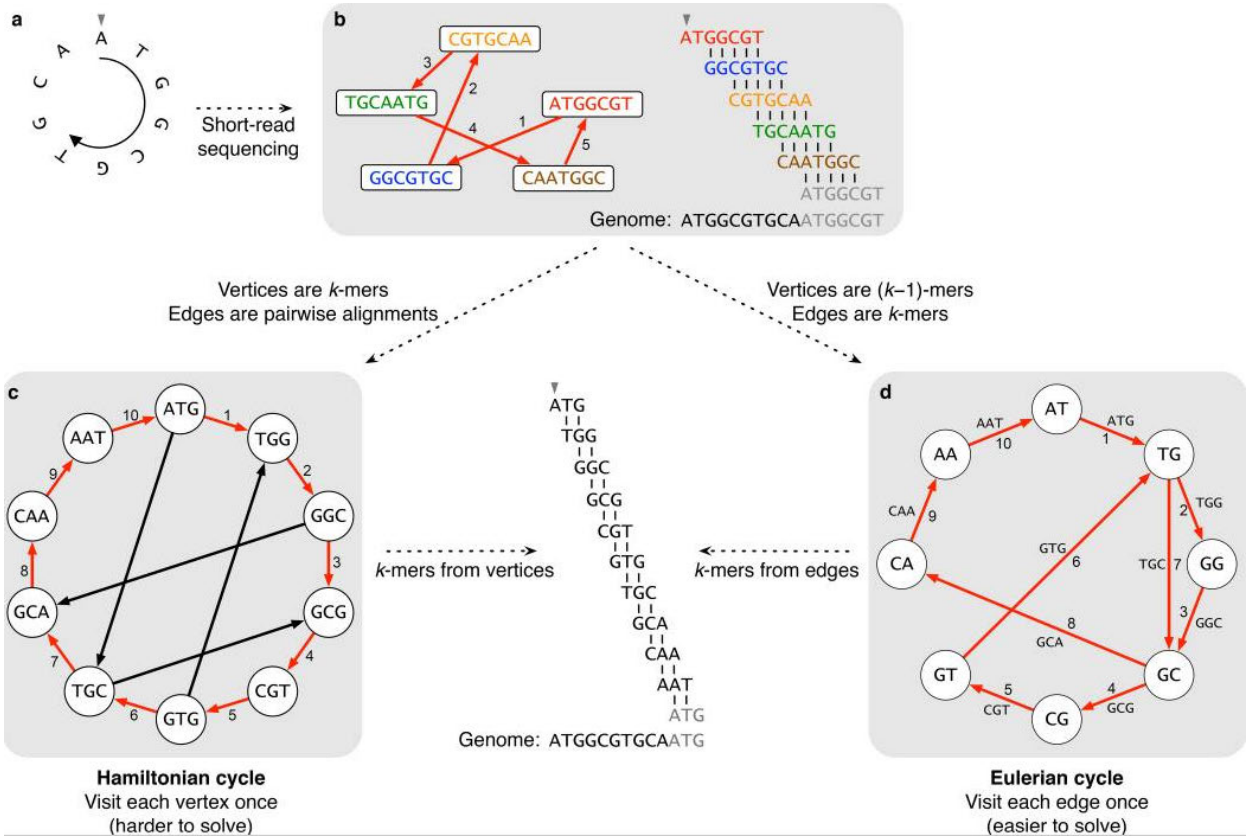

Abbildung 1.2: De Bruijn Graph Assembly Ansätze Hamiltonscher und Eulerischer Zyklus [Compeau et al., 2011]

als Knoten dargestellt. Die Kanten sind die k-mere, die diese Prä- und Suffixe als Knoten beinhalten. Das Auffinden eines Eulerschen Zyklus ermöglicht es, das Genom zu rekonstruieren, indem ein Alignment gebildet wird, indem jedes aufeinanderfolgende k-mer (von aufeinander folgenden Kanten), um eine Position verschoben wird. Dies erzeugt dieselbe zyklische Genomsequenz ohne den Rechenaufwand einen Hamilton-Zyklus zu finden [Compeau et al., 2011]. Jedoch bestehen auch bei den De Bruijn Graph Ansatz Probleme beim Assembly. Vor allem da vorher vier Annahmen getroffen werden müssen. Es wird angenommen, dass alle k-mere, die im Genom vorkommen, auch generiert werden, dass diese k-mere fehlerfrei sind, dass jedes k-mer maximal einmal im Genom vorkommt und das ein Genom nur aus einem einzigen zirkulären Chromosom besteht [Compeau et al., 2011]. Wenn beispielsweise die 100 bp langen Reads auch als 100-mers verwendet werden, dann kann nur ein kleiner Teil aller 100-mere, die im Genom zu finden sind, auch generiert werden, da nicht von jeder Basenpaarposition im Genom auch ein Read erzeugt wird. Wenn jedoch die k-mere zu kurz gewählt werden, dann repräsentieren die entstehenden k-mere nahezu alle k-mere, die im Genom zu finden sind. Ein Mittelweg ist also optimal (z.B 55-mere), damit zum einen abgesichert wird, dass auch, wenn einige 100-mers nicht generiert wurden, fast alle 55-mere im Genom erkannt werden und noch lang genug sind, dass sie nicht mehrfach vorkommen [Compeau

et al., 2011]. Um die auftretenden Fehler in den k-meren zu berichtigen, ist es Standard eine Error Correction vor dem Assembly durchzuführen, bzw. haben einige Assembler auch ein Modul zur Fehlerkorrektur mit eingebaut. Aufgrund dessen, dass die schon kurzen Reads in noch kürzere k-mere unterteilt werden, ist es schwierig Repeat-Bereiche aufzuklären, da es vorkommen kann, dass k-mere mehr als einmal auftreten. Aus diesem Grund ist die k-mer Multiplicity ein wesentlicher Bestandteil vieler De Bruijn Graph Assembler. Wenn Organismen sequenziert werden, die mehr als ein Replikon besitzen, würden Graphen erzeugt werden, im welchem nicht alle Knoten miteinander verbunden sind. Der Hamiltonsche Zyklus ist folglich nicht anwendbar, jedoch kann der Eulersche Zyklus auf dieses Problem adaptiert werden [Compeau et al., 2011]. Aus diesem Grund ist dies der Ansatz den die meisten Assembler als Grundlage haben.

1.5 MaSuRCA

Das MaSuRCA (Maryland Super Read Cabog Assembler) Genom-Assembler- und Analyse-Toolkit enthält den MaSuRCA-Genom-Assembler, den QuORUM-Fehlerkorrektor für Illumina-Reads, die POLCA-Genom-Polishing-Software, den Samba-Scaffolder, den Chromosomen-Scaffolder, den Jellyfish-Mer-Counter und den MUMmer-Aligner. Der MaSuRCA-Assembler kombiniert die Vorteile des DeBruijn-Graphen und des Overlap-Layout-Consensus-Assembler-Ansatzes. Er unterstützt das Hybrid-Assembly mit Short-Illumina-Reads und Long-PacBio/MinION-Reads [URL-09, 2022]. Er basiert laut Zimin et al., 2013 auf dem Konzept der Super-Reads. Dabei werden die Reads an beiden Enden Base für Base verlängert, solange die Verlängerung einmalig ist. Es wird ein *„k-mer count look-up table“* erstellt, um zu bestimmen, wie oft jedes k-mer in den Reads vorkommt. Da es nur vier Basen gibt, gibt es nur vier Möglichkeiten, wie ein Read an einem Ende verlängert werden kann. Folglich gibt es auch nur vier k-mere, die einen Read an dieser Stelle verlängern können. Zur Erweiterung wird also im *k-mer count look-up table* geschaut, welcher der vier k-mere in der Tabelle vorkommt. Entsprechend des k-mers wird die vorkommende Base an den Read angehangen. Diese Verlängerung wird fortgesetzt, bis kein einmaliger k-mer für eine bestimmte Position zur Verlängerung des Reads bestimmt werden kann. Die Verlängerung eines Reads wird abgebrochen, wenn an beiden Enden der Read nicht mehr verlängert werden kann. Diese verlängerten Reads werden als Super-Reads bezeichnet. Anschließend wird eine modifizierte Version des Cabog Assemblers genutzt, um die Super-Reads zu assemblieren. Es werden dabei nur die „maximalen“ Reads benutzt, als nur die Reads, die kein Substring größerer Reads sind. Zusammen mit den Super-Reads werden an dieser Stelle auch Long-Reads mit für das Assembly genutzt, wenn diese mit angegeben werden [Zimin et al., 2013].

1.6 Quality Control Assembly

Auch wenn eine sorgfältige Planung aller Teilschritte bis zum fertigen Assembly durchgeführt wurde, kann man nie davon ausgehen, dass auch ein qualitativ hochwertiges Assembly entsteht. Folglich muss, bevor die nächsten Schritte in der Pipeline angegangen werden können (Genom Annotation), die Qualität des Assemblies geprüft werden. Typischerweise erfolgt die Qualitätsbewertung bei Draft-Assemblies über statistische Messungen und dem Abgleich mit einem Referenzgenom, sofern dieses vorhanden ist. Dazu gehörend sind die Assemblygröße (meist im Vergleich zur geschätzten Genomgröße), das Maß der Contiguity, Completeness und Correctness [Jung et al., 2020]. Die Contiguity wird oft als Contig N50 gemessen, wobei 50 % aller Basen des Assemblies in Contigs mindestens der angegebenen Länge enthalten sind. Für eukaryotische Genome ist ein Contig N50 von über 1 Mbp allgemein als gut angesehen [Gurevich et al., 2013]. Letztendlich wird mit diesem Messwert und der Contiganzahl beschrieben, wie stark fragmentiert ein Assembly ist. Die Completeness wird oft anhand des *Benchmarking universal single copy orthologs (Busco)*-Score gemessen. Dabei wird auf das Vorhandensein oder Fehlens von hochkonservierten Genen im Assembly geprüft. Es wird der Prozentsatz an identifizierten Genen ausgegeben, wobei eine Bewertung von über 95 % als gut gilt [Simão et al., 2015]. Ein letzter Messwert ist die Correctness, also die Genauigkeit jedes Basenpaares im Assembly. Dieser ist jedoch nur schwer zu bestimmen und zudem nur, wenn eine qualitativ hochwertige Referenz vorliegt. Wenn die Referenz fehlerhaft ist, wird das Assembly schlecht bewertet, auch wenn das Assembly deutlich besser als die Referenz ist [PacBio, 2020]. Für diese Untersuchungen können zum einen Transkriptomdaten und zum anderen Referenzgenome und Short-Reads verwendet werden. Mithilfe der Transkriptomdaten können Frameshifts in den codierenden Genen erkannt werden [PacBio, 2020]. Insertion/Deletion (InDel)-Fehler sind in den hochkonservierten Genombereichen selten anzutreffen, da diese die Genprodukte fehlerhaft machen. Aus diesem Grund sind InDel-Fehler im Assembly in den meisten Fällen auf Fehler während des Assembly Prozesses zurückzuführen [PacBio, 2020]. Ähnlich wie bei Busco wird hiermit jedoch nur ein kleiner Teil des Genoms bewertet [PacBio, 2020]. Aus diesem Grund ist es gut, wenn ein Referenzgenom vorliegt, da somit ein größerer Teil des Assemblies überprüft werden kann. Dabei werden Hochkonfidenzbereiche festgelegt, in denen Referenz und Assembly gut miteinander übereinstimmen und diese miteinander verglichen [PacBio, 2020]. Eine weitere Möglichkeit ist die Verwendung von Short-Reads eines weiteren Sequenzierlaufs. Dabei wird kein Referenzgenom benötigt. Es werden k-mere des Assemblies mit k-meren der Short-Reads verglichen, wobei sich dies nicht nur auf die codierenden Bereiche beschränkt. Es wird nach k-meren in den Short-Reads gesucht, die nicht im Assembly vorkommen [Rhie et al., 2020].

2 Zielstellung

Das Ziel dieser Arbeit ist es aus den vorliegenden Illumina-Short-Read und Oxford-Nanopore-Long-Reads eines altdeutschen Schafpudels das bestmögliche *de novo* Assembly zu erstellen. Im besten Fall sollten dabei die Chromosomen vollständig zusammengebaut werden, wie es beispielsweise bei ROS_Cfam1.0 [URL-01, 2020] der Fall ist.
Es sollen drei verschiedene Ansätze getestet werden: das Short-Read-Assembly mit den Illumina-Short-Reads, das Long-Reads-Assembly mit den Oxford-Nanopore-Long-Reads und das Hybrid-Assembly mit beiden Sequenzdatenarten. Anschließend wird versucht jedes dieser initialen Assemblies zu optimieren.
Der erste Ansatz umfasst ein initiales Short-Read-Assembly mit anschließenden Merging verschiedener erzeugter Assemblies und dem Scaffolding dieser Assemblies. Der zweite Ansatz ist ein Long-Read-Assembly mit Polishing und Scaffolding und der dritte Ansatz ist ein Hybrid-Assembly mit Scaffolding.
Es werden dabei verschiedene Assembler, Merging-, Error-Correction-, Scaffolding- und Polishing-Tools getestet. Zudem wird jedes Assembly mit verschiedenen Tools auf die Qualität überprüft und zusätzlich mit externen Referenzen verglichen.

3 Material

3.1 Sequenzdaten

Illumina-Sequenzdaten eines altdeutschen Schafpudel-Rüdens (Sequenziert von GENEWIZ Germany GmbH)

Oxford-Nanopore-Sequenzdaten eines altdeutschen Schafpudel-Rüdens (Sequenziert von M.Sc. Nils Schön; Hochschule Mittweida)

3.2 Referenzen

GCA_000002285.4_Dog10K_Boxer_Tasha
https://ftp.ncbi.nlm.nih.gov/genomes/genbank/vertebrate_mammalian/
Canis_lupus/latest_assembly_versions/
GCA_000002285.4_Dog10K_Boxer_Tasha/GCA_000002285.4_Dog10K_Boxer_Tasha_genomic.fna.gz
In dieser Arbeit als Referenz Boxer bezeichnet.

GCA_004027395.1_CanFam_VD_v1_BIUU
https://ftp.ncbi.nlm.nih.gov/genomes/genbank/vertebrate_mammalian/
Canis_lupus/latest_assembly_versions/
GCA_004027395.1_CanFam_VD_v1_BIUU/GCA_004027395.1_CanFam_VD_v1_BIUU_genomic.fna.gz
In dieser Arbeit als Referenz BS72 bezeichnet.

GCA_005444595.1_UMICH_Zoey_3.1
https://ftp.ncbi.nlm.nih.gov/genomes/genbank/vertebrate_mammalian/
Canis_lupus/latest_assembly_versions/
GCA_005444595.1_UMICH_Zoey_3.1/GCA_005444595.1_UMICH_Zoey_3.1_genomic.fna.gz
In dieser Arbeit als Referenz Dogge bezeichnet.

GCA_011100685.1_UU_Cfam_GSD_1.0
https://ftp.ncbi.nlm.nih.gov/genomes/genbank/vertebrate_mammalian/
Canis_lupus/latest_assembly_versions/
GCA_011100685.1_UU_Cfam_GSD_1.0/GCA_011100685.1_UU_Cfam_GSD_1.0_genomic.fna.gz
In dieser Arbeit als Referenz Schäferhund bezeichnet.

GCA_011634725.1_ASM1163472v1
https://ftp.ncbi.nlm.nih.gov/genomes/genbank/vertebrate_mammalian/
Canis_lupus/latest_assembly_versions/
GCA_011634725.1_ASM1163472v1/GCA_011634725.1_ASM1163472v1_genomic.fna.gz
In dieser Arbeit als Referenz Chihuahua bezeichnet.

GCA_008641055.3_ASM864105v3
https://ftp.ncbi.nlm.nih.gov/genomes/genbank/vertebrate_mammalian/
Canis_lupus/latest_assembly_versions/
GCA_008641055.3_ASM864105v3/GCA_008641055.3_ASM864105v3_genomic.fna.gz
In dieser Arbeit als Referenz Basenji bezeichnet.

GCA_014441545.1_ROS_Cfam_1.0
https://ftp.ncbi.nlm.nih.gov/genomes/genbank/vertebrate_mammalian/
Canis_lupus/latest_assembly_versions/
GCA_014441545.1_ROS_Cfam_1.0/GCA_014441545.1_ROS_Cfam_1.0_genomic.fna.gz
In dieser Arbeit als Referenz Labrador bezeichnet.

GCA_905319855.2_mCanLor1.2
https://ftp.ncbi.nlm.nih.gov/genomes/genbank/vertebrate_mammalian/
Canis_lupus/latest_assembly_versions/
GCA_905319855.2_mCanLor1.2/GCA_905319855.2_mCanLor1.2_genomic.fna.gz
In dieser Arbeit als Referenz Wolf bezeichnet.

3.3 Software

Tabelle 3.1: Verwendete Software

Software	Quelle
Abyss v2.3.4	[Jackman et al., 2017]
Bbmap v38.96	[Bushnell, Brian, 2014]
Bcftools v1.9	[Li, 2011]
Bowtie2 v2.3.5.1	[Langmead und Salzberg, 2012]
Busco v5.2.2	[Simão et al., 2015]
Fastqc v0.11.9	[Andrews et al., 2012]
Flye v2.9.b1774	[Kolmogorov et al., 2019]
Fmlrc v1.0.0	[Wang et al., 2018]
GAM-NGS v1.1b	[Vicedomini et al., 2013]
Guppy v6.0.1	[Wick et al., 2019]
MAC v2.0	[Tang et al., 2019]
MaSuRCA v4.0.8	[Zimin et al., 2013]
Nanoplot v1.33.0	[Coster et al., 2018]
PDR v0.6.4e	[Xie und Wong, 2021]
Pilon v1.24	[Walker et al., 2014]
Polca (MaSuRCA v4.0.9)	[Zimin und Salzberg, 2020]
Quast v5.1.0rc1	[Gurevich et al., 2013]
R v4.0.4	[R Core Team, 2013]
Rstudio 2022.02.3 Build 492	[RStudio Team, 2020]

Tabelle 3.1: Verwendete Software

Software	Quelle
RopeBTW2-r187	[Li, 2014]
Samba (MaSuRCA v4.0.9)	[Zimin und Salzberg, 2022]
Samtools v1.13-44-g260b6b8	[Li, 2011]
SOAPdenovo2 v2.04-r241	[Luo et al., 2012]
SparseAssembler v1.0	[Ye et al., 2012]
Trinity v2.13.2	[Grabherr et al., 2011]

Tabelle 3.2: R Packages

Software	Quelle
dplyr v1.0.9	[Wickham et al., 2022]
ggplot v3.3.6	[Wickham, 2016]
patchwork v1.1.0.900C	[Pedersen, 2022]
scales v1.2.0	[Wickham und Seidel, 2022]
tidyr v1.2.0	[Wickham und Girlich, 2022]
tidyverse v1.3.1	[Wickham et al., 2019]

3.4 Server

Tabelle 3.3: Server Spezifikationen

Detail	Spezifikation
Architecture	x86_64
CPU op-mode(s)	32-bit, 64-bit
Byte Order	Little Endian
Address sizes	46 bits physical, 48 bits virtual
CPU(s)	40
CPU Model name	Intel(R) Xeon(R) CPU E5-2690 v2 @ 3.00GHz
Memory	251 GiB
Distribution	5.3.6-arch1-1-ARCH

4 Methoden

4.1 Sequenzierung

Die Illumina-Sequenzierung wurde von GENEWIZ Germany GmbH durchgeführt. Dafür wurde der NovaSeq 5000 von Illumina, Inc. verwendet, wobei Paired-End-Reads mit einer Länge von 150 bp erzeugt wurden. Im Anschluss wurde ein Adapter Trimming und ein Quality Filtering (Phred-Score >10) durchgeführt.

Die Nanopore-Sequenzierung wurde von Nils Schön an der Hochschule Mittweida durchgeführt. Die DNA für die Sequenzierung wurde aus einer Vollblutprobe eines altdeutschen Schafpudel-Rüdens gewonnen. Die Library Preparation wurde mit dem Ligation Sequencing Kit SQK-LSK112 durchgeführt. Zur Sequenzierung wurde der MinION Mk1C mit FLO Min 112 Flow Cells verwendet. Es wurde drei Sequenzierläufe durchgeführt. Das Basecalling der erhaltenen Sequenzdaten wurden mit Guppy v6.0.1 mit dem Modus 'super high accuracy' durchgeführt, wobei Read Splitting, Calibration Strand Detection und Adapter Trimming aktiviert wurden. Die Qualitätskontrolle der Nanopore-Sequenzdaten wurde mit NanoPlot v1.33.0 durchgeführt.

4.2 Vergleich der Referenzen

Es wurden neun Referenzen von Hundegenomen ausgewählt, die mit den erzeugten Schafpudel-Assemblies verglichen wurden. Diese sind in Kapitel 3.2 aufgelistet. Die Fasta-Dateien der Assemblies der Referenzen wurden heruntergeladen und die Qualitäts-Metriken, wie in Kapitel 4.8 beschrieben, bestimmt, um die Referenzen untereinander und später mit den erzeugten Assemblies vergleichen zu können. Die Read-Coverage wurde nicht bestimmt, da die Referenzgenome mit anderen Sequenzdaten erzeugt wurden. Bei der Bestimmung des PDR-Scores wurde jedes der neun Referenz-Assemblies mit den anderen acht Referenz-Assemblies verglichen.

4.3 Short-Read-Assembly

4.3.1 Initiale Assemblies

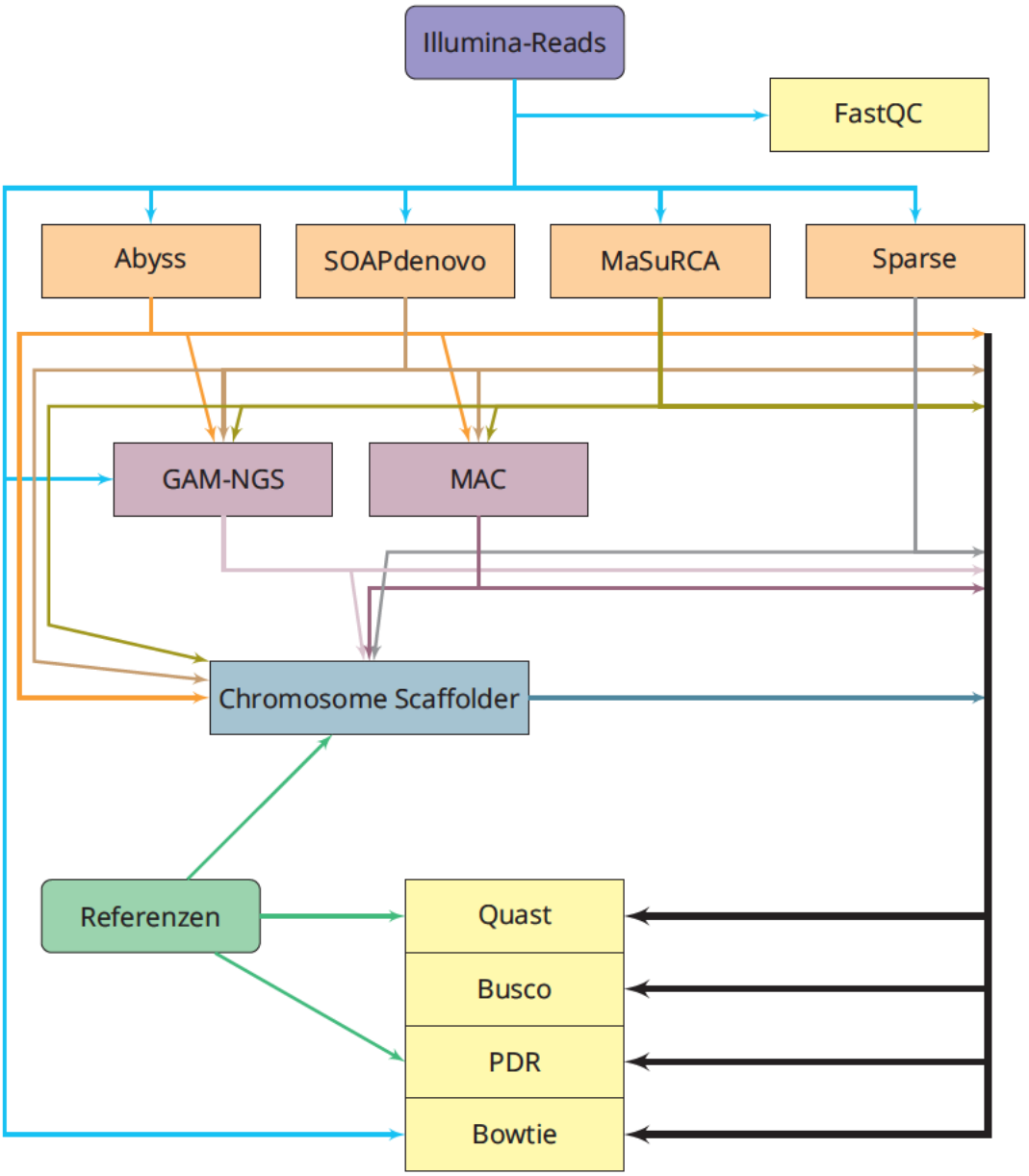

Abbildung 4.1: Flussdiagramm der Short-Read-Assemblies

Das Vorgehen beim Short-Read-Assembly ist in Abbildung 4.1 dargestellt. Es wurden vier Short-Read-Assembler getestet. Bei diesen Assemblern handelt es sich um Abyss (v2.3.4), SOAPdenovo2 (v2.04-r241), MaSuRCA (v4.0.8) und SparseAssembler (v1.0). Es wurden bei allen Assemblern die Paired-End-Reads (`sheep_poodle_reads_1.fastq` und `sheep_poodle_reads_2.fastq`) benutzt, ohne dass diese vorher einer Fehlerkorrektur unterzogen wurden. Für Abyss wurden die

Optionen `np=30 name=sheep_poodle k=96 B=150G in='Read-Files'` verwendet (siehe Anhang A1.1). Das SOAPdenovo-Assembly wurde mit den Optionen `-R -K 63` und der Angabe des Pfades zum Config-File SOAPdenovo.config durchgeführt (siehe Anhang A1.2). Für das MaSuRCA-Assembly musste ein Config-File erstellt werden, in welchem alle Parameter definiert sind (Config.txt). Mit `./assemble` im entsprechenden Verzeichnis wird das Assembly gestartet (siehe Anhang A1.3). Für den SparseAssembler wurden die Optionen `g 15 k 53 LD 0 GS 2400000000 f <Read-file_1.fastq> f <Read-file_2.fastq> NodeCovTh 1 EdgeCovTh 0` verwendet (siehe Anhang A1.4).

4.3.2 Assembly-Merging

Für das Merging der Assemblies wurden die Tools GAM-NGS (v1.1b) und MAC (v2.0) verwendet. Es wurden das Abyss-, SOAPdenovo- und MaSuRCA-Assmebly gemerged. Es konnten immer nur zwei Assemblies miteinander gemerged werden. Zunächst wurde das Abyss-Assembly (A) und das SOAP-Assembly (S) gemerged (GAM-NGS_A-S und MAC_A-S) und anschließend das erhaltene Merged-Assembly mit dem MaSuRCA-Assembly (M) gemerged (GAM-NGS_A-S-M und MAC_A-S-M). Alle Befehle für das Merging sind im Anhang A2.1 und A2.2 zu finden.

4.3.3 Chromosom-Scaffolding

Für das Chromosom-Scaffolding mit Chromosome-Scaffolder (Bestandteil von MaSuRCA v4.0.9) wurden die Optionen `-r <Referenz.fasta> -q <Assembly.fasta> -t 32 -nb` verwendet. Als Referenzgenom wurde Referenz Labrador verwendet. Die Befehle sind im Anhang A3.1 zu finden.

4.4 Long-Read-Assemblies

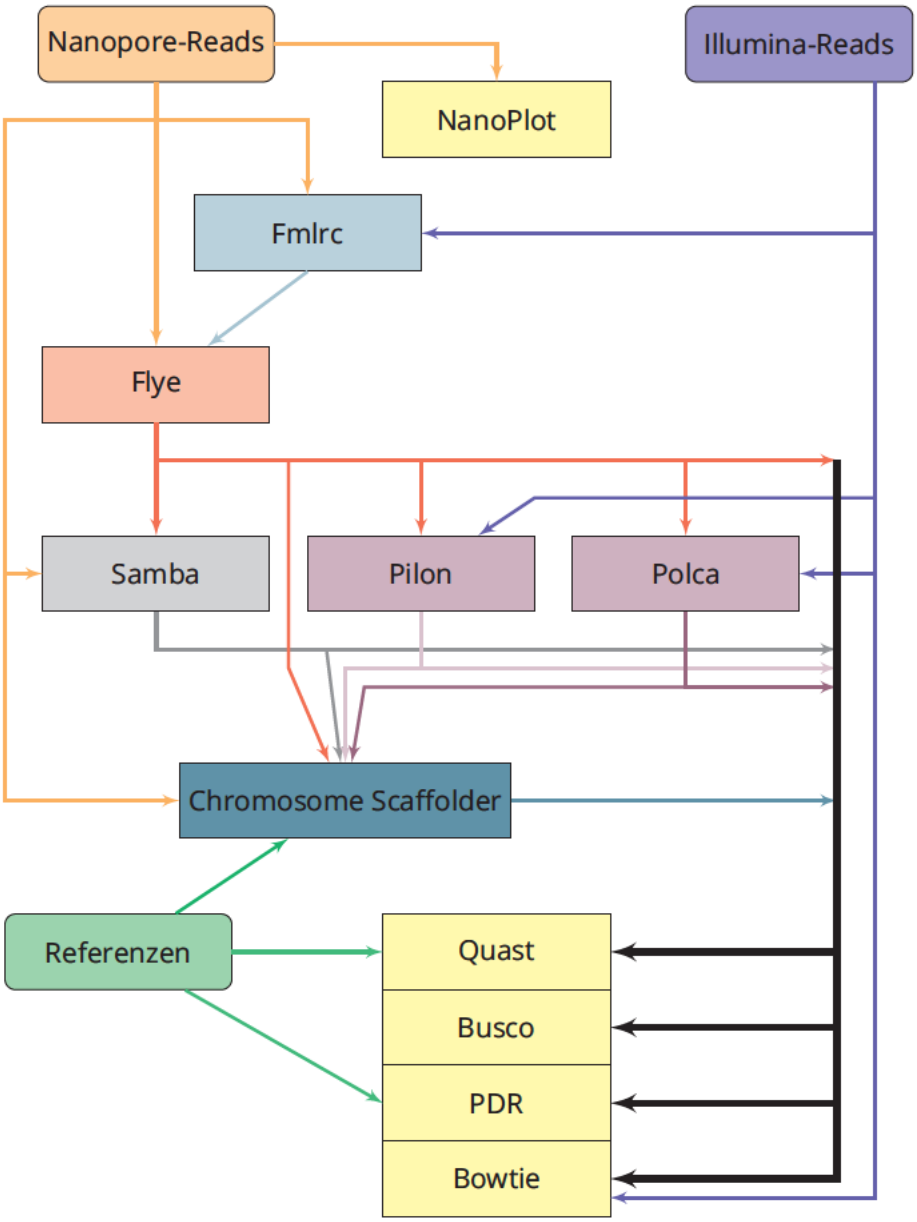

Abbildung 4.2: Flussdiagramm der Long-Read-Assemblies

Der Ablauf des Long-Read-Assemblies ist in Abbildung 4.2 dargestellt. Es wurde eine Error-Correction der Long-Reads mit Fmlrc (v1.0.0) durchgeführt. Dafür wurde zunächst ein Burrows-Wheeler Transform (BWT) der Paired-End-Reads mit RopeBWT2 erstellt. Unter Angabe des erhaltenen BWT, dem Assembly-File und dem Long-Read-File wurden die Long-Reads mit Fmlrc korrigiert (siehe Anhang A4.1). Das Long-Read-Assembly wurde mit Flye (v2.9.b1774) durchgeführt. Hierfür wurden die unkorrigierten Reads (im Folgenden nur als Flye bezeichnet) als auch die mit fmlrc-korrigierte

Reads (im Folgenden als Flye_fmlrc bezeichnet) assembliert (siehe Anhang A4.2). Es wurden Pilon und Polca als Polishing-Tools getestet (gekennzeichnet durch das anhängen von _pilon, bzw. _polca) (siehe Kapitel 4.7). Wenn ein Polishing-Schritt mehrmals durchgeführt wurde wird die Anzahl mit angegeben (1x, 2x). Zudem wurden die Scaffolding-Tools Samba (Long-Read-Scaffolding) und Chromosome-Scaffolder (Referenz-basiertes Long-Read-Scaffolding) verwendet (gekennzeichnet durch Präfix Samba_, bzw Mscaf_) (siehe Kapitel 4.6). Bei Samba wurden zum einen die unkorrigierten Nanopore-Reads (Samba_) und zum anderen die fmlrc-korrigierten Nanpore-Reads (Samba_fmlrc_) verwendet.

4.5 Hybrid-Assembly

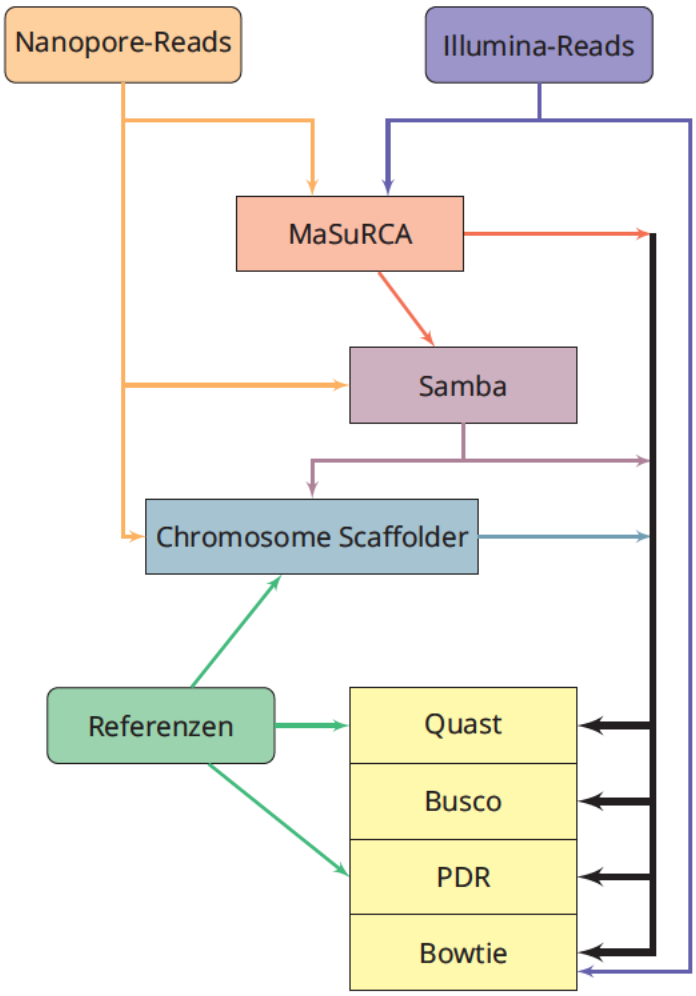

Abbildung 4.3: Flussdiagramm der Hybrid-Assemblies

Der Ablauf des Hybrid-Assemblies ist in Abbildung 4.3 dargestellt. Für das Hybrid-Assembly wurde der MaSuRCA-Assembler verwendet. Um ein Hybrid-Assembly mit MaSuRCA durchzuführen muss in der Config-Datei der Parameter `NANOPORE=<path/to/Nanopore.fastq>` angegeben werden.

4.6 Assembly-Scaffolding

Es wurden zwei Scaffolding-Tools in dieser Arbeit verwendet. Beide sind Bestandteil des MaSuRCA-Assemblers. Samba-Scaffolder ermöglicht das Scaffolding von Assemblies mit Long-Reads. Dabei muss das Assembly mit der Option `-r <Assembly.fasta>` und und die Long-Reads mit `-q <Reads.fastq>` angegeben werden (siehe Anhang A3.2). Das Chromosome-Scaffolding wurde, wie bereits in Kapitel 4.3.3 beschreiben, durchgeführt.

4.7 Assembly-Polishing

Für das Polishing der Assemblies wurden Pilon (v1.24) und Polca (Bestandteil von MaSuRCA v4.0.9) verwendet. Für das Polishing mit Pilon mussten zunächst die Paired-End-Reads mit Bowtie2 (v2.3.5.1) an das Assembly aligniert werden. Die Befehle sind im Anhang A6.4 zu finden. Das entstandene Alignment-File wurde mit Samtools (v1.13-44-g260b6b8) indexiert. Das eigentliche Pilon-Polishing wurde mit den Optionen `--genome <Assembly.fasta> --fix all --changes --frags mapping.sorted.bam --outdir <path/to/outdir> --output <prefix>` durchgeführt (siehe Anhang A5.1). Das Polishing mit Polca wurde mit den Parametern `-a <Assembly.fasta> -t 10 -r '</Path/to/Paired-End-Reads>'` durchgeführt (siehe Anhang A5.2).

4.8 Assembly Quality Control

Die Metriken für die Qualitätskontrolle der erstellten Assemblies wurde mit Quast (v5.1.0rc1) und Busco (v5.2.2) erstellt. Für Quast wurden die Optionen `-r <Referenz.fasta> -1 <Reads_1.fastq> -2 <Reads_2.fastq>` verwendet (siehe Anhang A6.1). Als Referenz wurde Referenz Boxer verwendet. Bei der Bestimmung der Busco Completeness wurde als Abstammungsdatensatz *Laurasiatheria_odb10* gewählt und der Modus auf *„genome“* gesetzt. Das vollständige Config-File ist im Anhang A6.2 busco_config.ini zu finden. PDR wurde verwendet, um zwei Assemblies direkt miteinander zu vergleichen. Es wurden alle Referenzen miteinander verglichen. Alle Assemblies wurden mit den Referenzgenomen Boxer,

Labrador, Basenji, Dogge, Schäferhund und Wolf verglichen. Der Befehl für einen dieser Vergleiche ist im Anhang A6.3 zu finden. Bowtie2 wurde verwendet um die Read-Alignment-Rate zu bestimmen, indem die Paired-End-Reads an das Assembly aligniert wurden. Hierfür wurden die Optionen `-q -p 40 -k 20 -x <Assembly.fasta> -1 </path/to/to/Reads_1.fastq> -2 </path/to/to/Reads_2.fastq>` verwendet (siehe Anhang A6.4).

5 Ergebnisse und Diskussion

5.1 Read QC

5.1.1 Illumina Reads

Die Qualitätskontrolle der Illumina Paired-End-Reads wurde mit FastQC durchgeführt. Acht der elf Module waren erfolgreich (Abb. 5.1). Bei den Forward-Reads gab es eine Warnung und zwei Fehlschläge, wohingegen die Reverse-Reads zwei Warnungen und einen Fehlschlag aufzeigten.

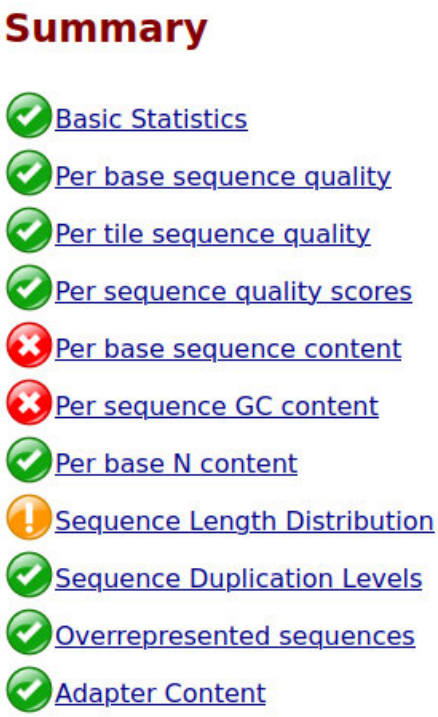

Abbildung 5.1: FastQC Summary der Illumina-Sequenzdaten

Da Paired-End-Reads vorliegen, sind keine großen Unterschiede zwischen den Forward- und Reverse-Reads zu erwarten. Einige auffällige Module werden im Folgenden genauer betrachtet. Wie in Abbildung 5.2 zu sehen, sind insgesamt 231.463.053 Reads in jeder der beiden Dateien enthalten. Anhand der Zeilen *Sequences flagged as poor quality* und *Sequence length* ist erkennbar, dass bei den Reads, die von GENEWIZ Germany GmbH erstellt worden sind, schon ein Pre-Processing durchgeführt wurde, da keine Reads von schlechter Qualität enthalten sind und Sequenzlängen von 35-151 bp vorkommen. Die unprozessierten Reads hätten ansonsten alle eine Länge von 150 bp und die Qualität wäre nicht bei allen Reads über dem Qualitätsschwellwert [URL-12, 2018].

Basic Statistics

Measure	Value
Filename	55-1_S5_R1_001.fastq.gz
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	231463053
Sequences flagged as poor quality	0
Sequence length	35-151
%GC	41

Abbildung 5.2: Basic Statistics der Illumina-Sequenzdaten

Per base sequence quality

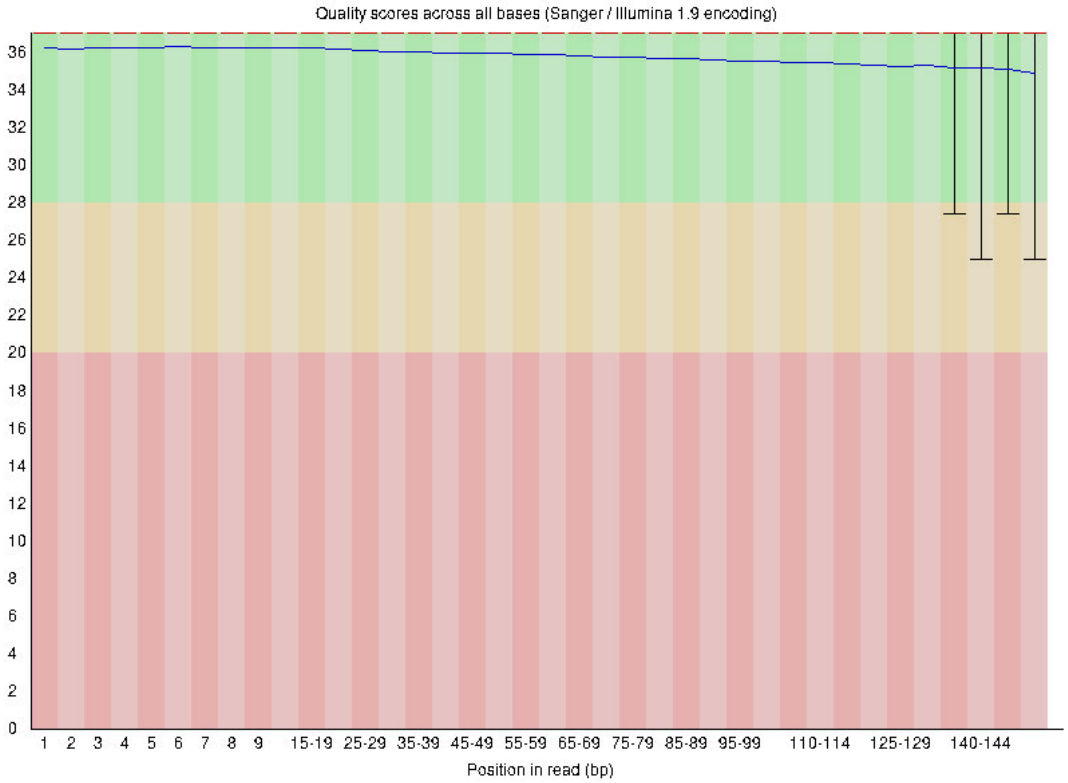

Abbildung 5.3: *Per Base Sequence Quality* der Illumina-Sequenzdaten

Das wichtigste Modul von FastQC ist die *per base sequence quality* (Abb. 5.3). Diese gibt Positionsweise die durchschnittliche Qualität der einzelnen Basen in den Reads an. Die Reads haben einen Q-Score im Bereich von 34 bis 36, was einer Fehlerrate von etwa 0,025 % entspricht. Da die

Qualität im Durchschnitt sehr hoch ist, wirken sich die Unterschiede nicht negativ aus. Der leichte Abfall der Qualität am Ende der Reads durch das Phasing ist nicht relevant (URL-13, 2022; URL-14, 2019). Insgesamt kann gesagt werden, dass die Illumina-Paired-End-Reads von hoher Qualität sind [URL-05, 2011].

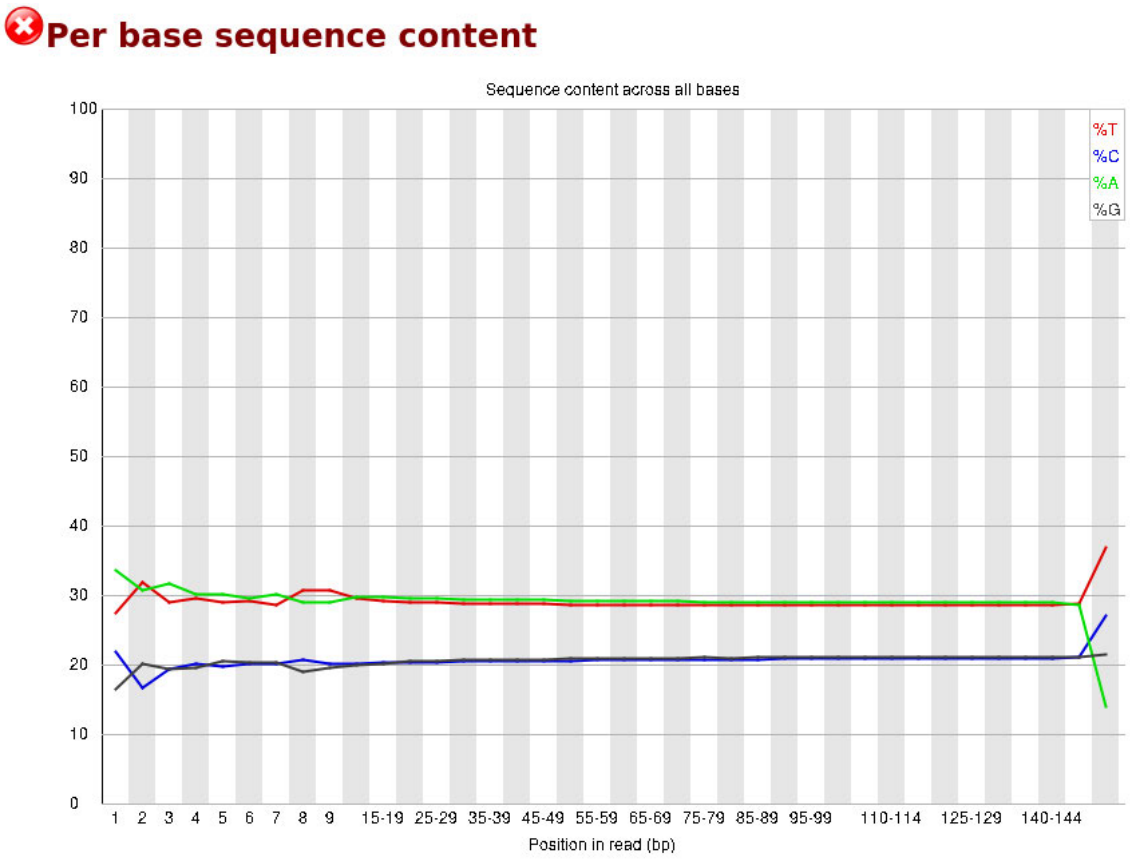

Abbildung 5.4: *Per Base Sequence Content* der Illumina-Sequenzdaten

Eines der beiden fehlgeschlagenen Module ist das Modul *per base sequence content* (Abb. 5.4). Hier ist der durchschnittliche GC-Gehalt der einzelnen Basen über die einzelen Positionen in den Reads angegeben. Wie zu erkennen, ist dieser für A und T bei etwa 29-30 % und für C und G bei 20-21 %. Dies würde für einen durchschnittlichen GC-Gehalt von 40-41 % sprechen, was den Literaturangaben für das Hundegenom entspricht (Lindblad-Toh et al., 2005; Edwards et al., 2021; Wang et al., 2021a). Der Grund für das Fehlschlagen dieses Moduls sind die Abweichungen am Ende der Reads. Diese Unterschiede entstehen durch ein aggressives Adapter Trimming [URL-12, 2018]. Da Sequenzen, die mit kurzen Bereichen der Adapter übereinstimmen, durch das Adapter Trimming auch mit entfernt werden, verbleiben nur Sequenzen, die nicht mit dem Adapter übereinstimmen, was zu Unterschieden im Gehalt der einzelnen Basen führen kann [URL-12, 2018].

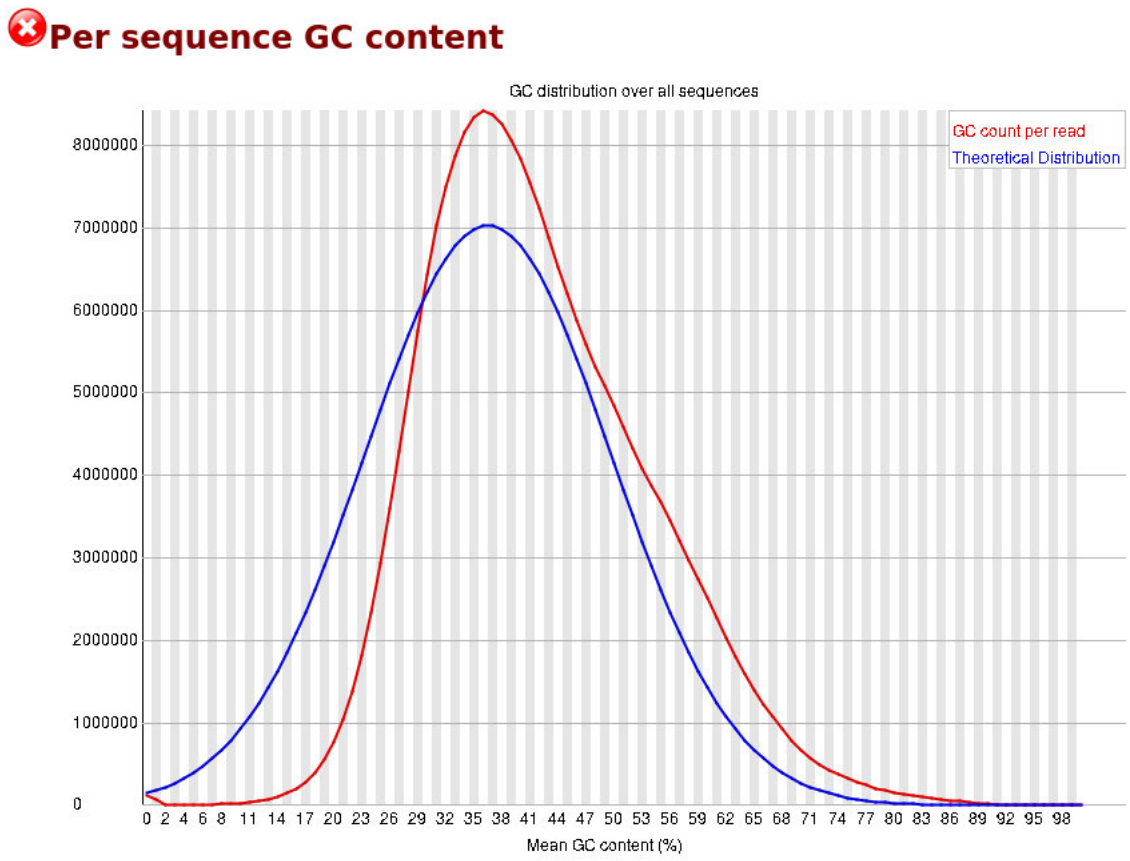

Abbildung 5.5: *Per Sequence GC Content* der Illumina-Sequenzdaten

Das Modul *per sequence GC Content* (Abb. 5.5) gibt bei den Forward Reads einen Fehler und bei den Reverse Reads eine Warnung. Es sind allerdings keine großen Unterschiede im Verlauf der Kurven zu erkennen, weshalb nur eines der beiden Diagramme dargestellt ist. Die Verteilung des durchschnittlichen GC-Gehaltes der Reads wird als Normalverteilung angenommen [URL-12, 2018]. Wie allerdings in Abbildung 5.5 zu sehen, ist keine perfekte Normalverteilung bei den Reads erkennbar. Der Peak ist deutlich höher als bei der angenommenen Normalverteilung. Es sind deutlich weniger Reads mit geringem GC-Gehalt als bei der Normalverteilung vorhanden. Allerdings ist ein fast linearer Abfall der Verteilung von 35 % bis etwa 70 % GC-Gehalt zu erkennen. Diese Abweichungen von der Normalverteilung haben jedoch keinen Einfluss auf die Qualität der Reads, da kein ungewöhnlicher Kurvenverlauf, wie mehrere Peaks oder Ähnliches, auftritt [URL-12, 2018].

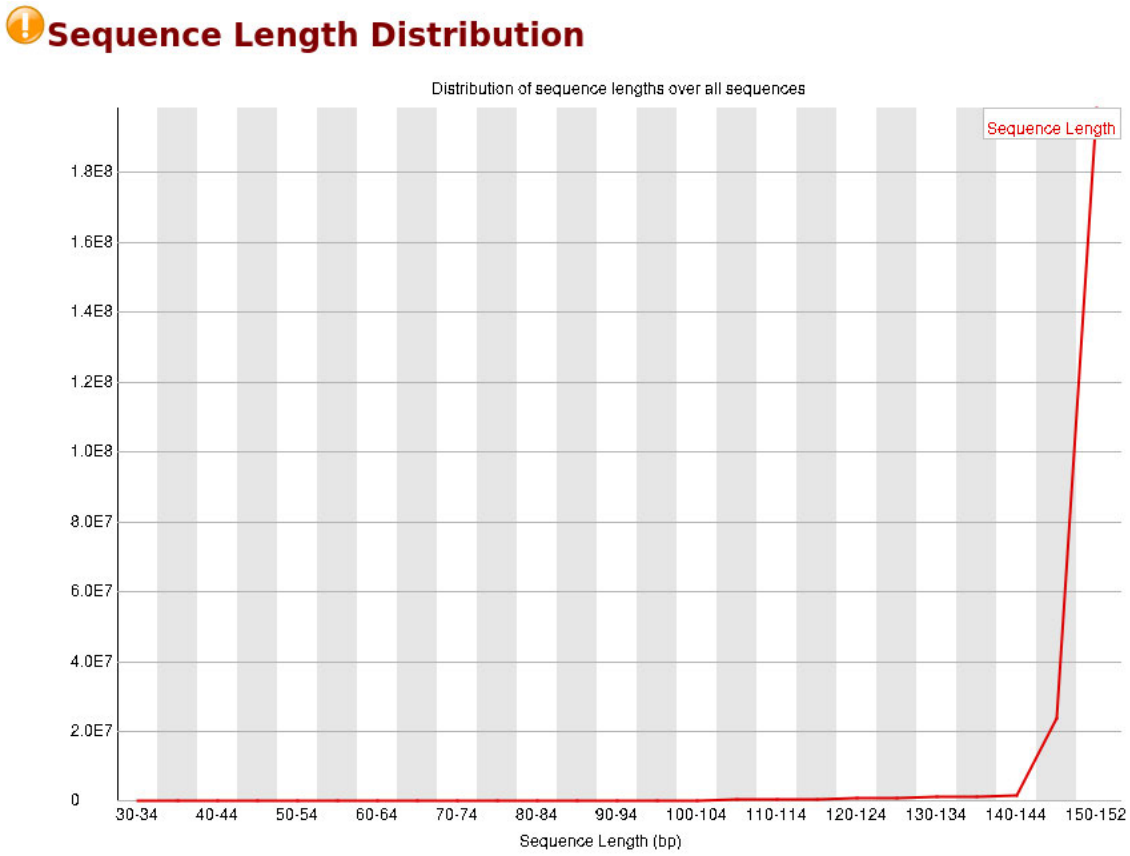

Abbildung 5.6: *Sequence Length Distribution* der Illumina-Sequenzdaten

Die *Sequence length Distribution* gibt eine Warnung, wenn nicht alle Reads die gleiche Länge haben [URL-12, 2018]. Da aber ein Adapter Trimming durchgeführt wurde, kann diese Warnung ignoriert werden.

5.1.2 Nanopore Reads

Bei der Qualitätskontrolle der Nanopore-Sequenzdaten mittels NanoPlot sind die Werte entstanden, wie sie in Tabelle 5.1 dargestellt sind. Die durchschnittliche Länge der Sequenzdaten beträgt 4.167 bp, wohingegen der Median mit 2.185 bp nur bei etwa der Hälfte liegt. Folglich ist davon auszugehen, dass in den Sequenzdaten viele kurze Sequenzen enthalten sind, aber nur wenige lange Sequenzen. Bei der Betrachtung des Q-Score wird deutlich, dass hier der Median leicht über dem Durchschnittswert liegt, was darauf hindeutet, dass die kürzeren Sequenzen von leicht besserer Qualität sind. Eine Standardabweichung von 6.263 bp zeugt von großen Unterschieden in der Read-Länge, was auf eine qualitativ schlechte Probe (starke Fragmentierung der DNA in der Probe) oder

Tabelle 5.1: NanoPlot Summary der Nanopore-Sequenzdaten

Länge (Durchschnitt)	4.167 bp
Q-Score (Durchschnitt)	15,5
Länge (Median)	2.185 bp
Q-Score (Median)	16,6
Standardabweichung (Länge)	6.263 bp
Read-Anzahl	4.227.338
N50 (Länge)	8.058
Nukleotidanzahl	17.616.964.864

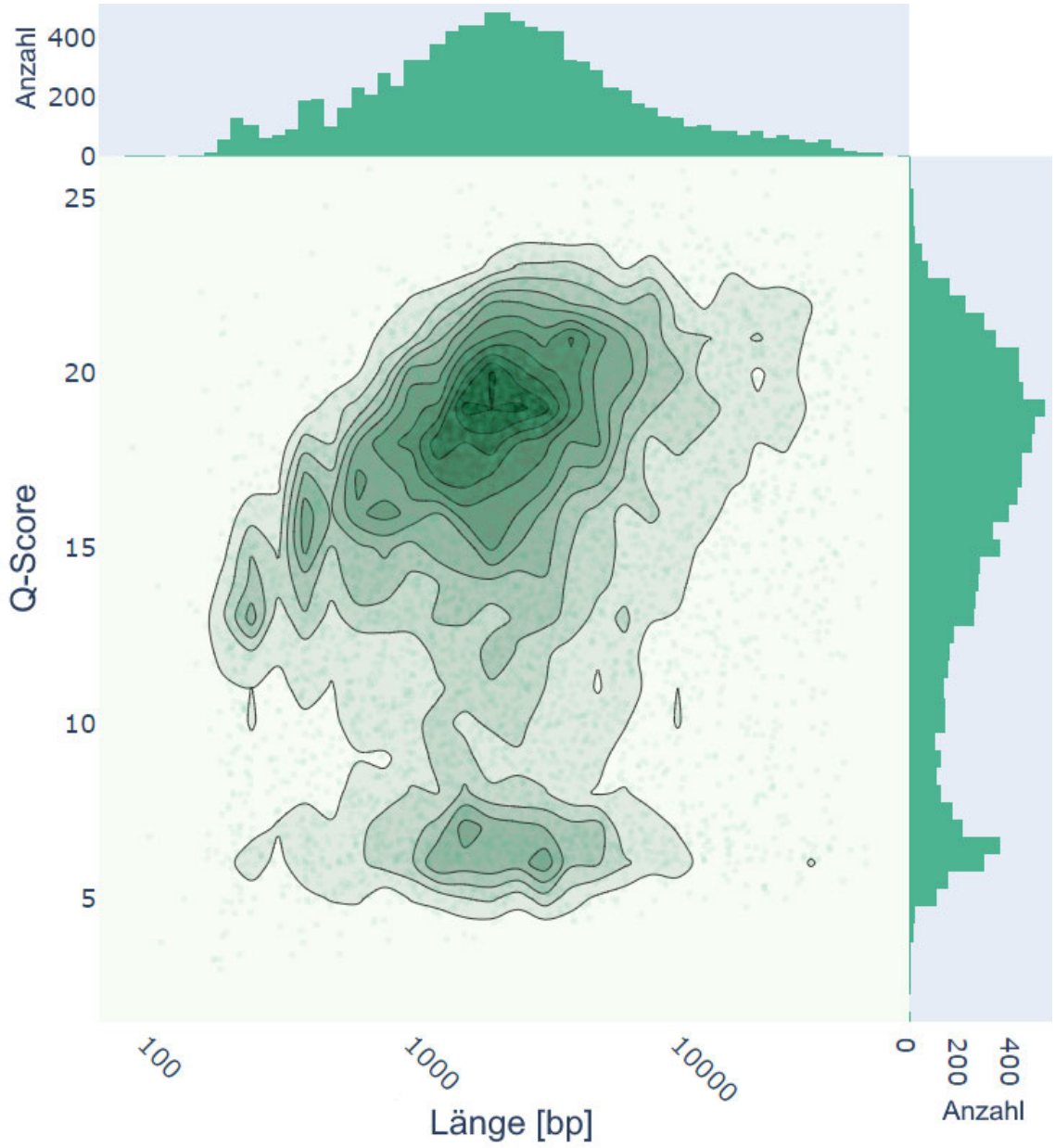

Abbildung 5.7: Kerndichteschätzung der Nanopore-Sequenzdaten mittels NanoPlot

auf das Einwirken hoher mechanischer Kräfte auf die DNA während der Libary Preparation hindeutet [Klingström et al., 2018]. Insgesamt wurden etwa 4,2 Millionen Reads mit einem N50 von 8.058 und einer Gesamtnukleotidanzahl von 17,6 Gbp erzeugt. Dies würde, wenn ein Genom von etwa 2,4 Gbp erwartet wird [Lindblad-Toh et al., 2005], eine Sequencing-Depth von 7,34 x ergeben. Der Long-Read-Assembler Flye, der in dieser Arbeit Verwendung findet, gibt als Voraussetzung eine Sequenzing-Depth von 40 x an, damit gute Assemblies entstehen (URL-10, 2022; Kolmogorov et al., 2019).
Abbildung 5.7 zeigt das Diagramm der Kerndichteschätzung der Nanopore-Sequenzdaten, wobei die Read-Länge und der Q-Score aufgetragen sind. Der Bereich mit der größten Dichte liegt bei einer Read-Länge von 3.000 bis 3.500 bp und einem Q-Score von 19 bis 20. Zusätzlich setzt sich ein Bereich ab, von ähnlicher Länge, aber deutlich geringerem Q-Score (5-7). Dieser Bereich ist aber nicht in einem einzelnen Sequenzierlauf entstanden, sondern ist in jedem der Sequenzierläufe erkennbar. Es bestände die Möglichkeit die Sequenzdaten nach deren Qualität zu filtern, sodass beispielsweise nur die Reads mit einem Q-Score von über 10 verwendet werden. Dies würde jedoch die Sequenzing-Depth noch weiter verringern, welche ohnehin schon sehr gering für ein Long-Read-Assmelby ist. Aus diesem Grund wird wie in Kapitel 5.4 beschrieben, eine Error-Correction der Nanopore-Sequenzdaten durchgeführt, die Sequenzdaten aber nicht nach Qualität gefiltert.

5.2 Vergleich der Referenzen

Eine Untersuchung verschiedener Genom-Assemblies des Hundegenoms ist angebracht, um Vergleichswerte für das Schafpudelgenom zu erhalten, woran die Qualität des Assemblies überprüft werden kann. Im Folgenden wurden neun verschiedene Assemblies als Referenzen herangezogen, wobei acht dieser Assemblies von unterschiedlichen Hunderassen und eines vom Grauwolf stammt. Bei der Betrachtung der Verwandtschaftsverhältnisse der Hunderassen nach Parker et al., 2017 (siehe Abb. 1.1) kann gesagt werden, dass sieben der Hunderassen (mit Ausnahme der Referenzen Wolf und BS72) aus verschiedenen phänotypischen und historischen Gruppen stammen. Somit ist es wahrscheinlich, dass das Schafpudelgenom ähnliche Werte erzeugt, wie die Referenzen.
Die Qualitäts-Metriken der Referenz-Assemblies sind in Abbildung 5.8 visualisiert. Von keinem Assembly kann behauptet werden, dass es zu 100 % richtig oder vollständig ist, da es keine Möglichkeit gibt dies eindeutig nachzuweisen [Bradnam et al., 2013]. Sie enthalten unbestimmte Nukleotide und nicht zugeordnete Contigs und Scaffolds [Gurevich et al., 2013]. Folglich ist es sinnvoll

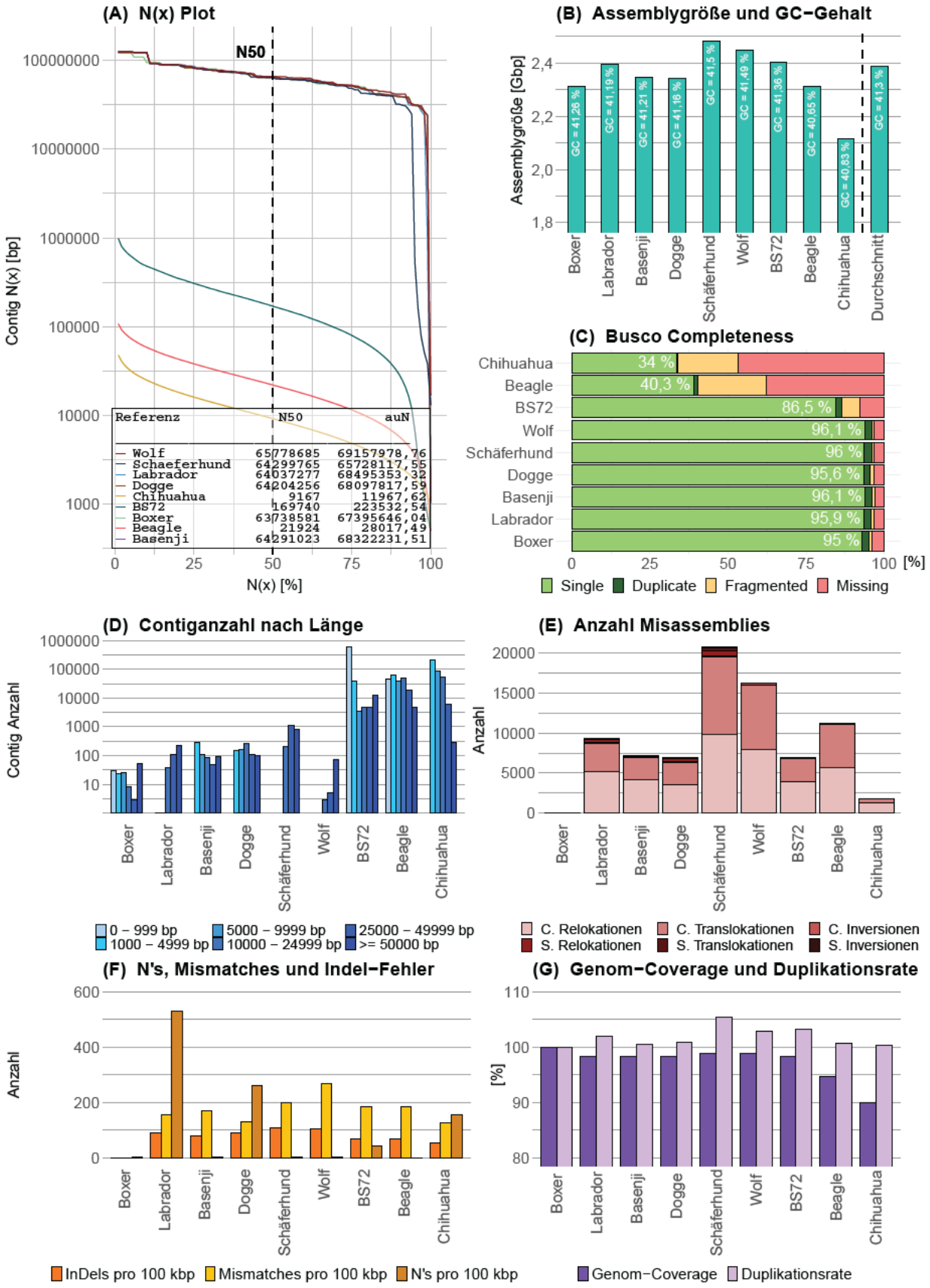

Abbildung 5.8: Qualitäts-Metriken der Referenz-Assemblies visualisiert

die Qualität der Assemblies miteinander zu vergleichen, um später bessere Erkenntnisse beim Vergleich der Referenzen mit den Assemblies des Schafpudelgenoms zu ermöglichen [Bradnam et al., 2013]. Im Nachfolgenden wird trotzdem von vollständigen Assemblies gesprochen, jedoch immer unter Beachtung dieser Limitierung.

Der N50 gibt die minimale Länge von Contigs an, in denen mindestens 50 % aller Nukleotide vorkommen [Earl et al., 2011]. In Abbildung 5.8 A ist diese Länge für alle Perzentile von N1 bis N100 aufgetragen und der N50 noch einmal extra markiert. Zudem ist in der Tabelle in Abbildung 5.8 A der N50 und der auN für alle Referenzen angegeben. Der auN gibt die Fläche unter der Kurve des N(x)-Contiglängen-Graphen an. Je größer die Fläche unter der Kurve, desto besser die Contiguity des Assemblies [Li, 2020]. Laut Li, 2020 ist der auN ein besserer Messwert, um die Contiguity eines Assemblies zu messen, als der N50, da er nicht nur auf einen bestimmten Punkt in der Nx-Kurve bezieht, sondern die gesamte Kurve mit einbezieht. Für die Bestimmung des N50 und auN werden nur Contigs mit einer Länge von mindestens 500 bp verwendet. Wie zu sehen, ist der Verlauf bei den Chromosom-Level-Assemblies (Boxer, Labrador, Basenji, Dogge, Schäferhund, Wolf) nahezu identisch. Daraus lässt sich schließen, dass, wenn ein N50 und auN von etwa 65.000.000 und 70.000.000 vorliegen, vollständig assemblierte Chromosomen im Assembly erzeugt wurden. Der N50 und auN der Scaffold- und Contig-Level-Assemblies (Beagle, BS72 und Chihuahua) liegen deutlich unter den Werten der Chromosomen-Level-Assemblies. Ein vollständiges Assembly des Schafpudels hätte folglich einen N50 und auN vergleichbar mit dem der Chromosomen-Level-Assemblies.

Die Assemblygröße und der GC-Gehalt dieser neun Referenz-Assemblies sind in Abbildung 5.8 B dargestellt. Zudem ist der Durchschnitt der Assemblygröße (2,387 Gbp), des GC-Gehalts (41,3 %) und der Standardabweichung (66 Mbp) im letzten Balken des Diagramms angegeben. Der Durchschnitt bezieht sich jedoch nur auf die Chromosomen-Level-Assemblies, also die Assemblies bei denen ununterbrochene Chromosomensequenzen entstanden sind. Da die Assemblies von den Referenzen Chihuahua, BS72 und Beagle nicht zu vollständigen Chromosomen zusammengesetzt werden konnten, (Scaffold- oder Contig-Level-Assemblies), ist die Nukleotidanzahl dieser Assemblies nicht mit der Genomgröße gleichzusetzen. Wie in Abbildung 5.8 B zu sehen, liegt die Assembly-, bzw. Genomgröße von Hundegenomen im Bereich von 2,3 bis 2,5 Gbp. Somit ist es wahrscheinlich, dass auch das Genom des Schafpudels sich in diesem Bereich befindet. Gleiches gilt für den GC-Gehalt, der zwischen 40 und 42 % liegt. Wenn einer der Werte deutlich erhöht oder erniedrigt ist, muss dieses Assembly kritisch betrachtet werden.

In Abbildung 5.8 C ist die Busco Completeness, also das Vorhandensein essentieller Gene auf dem Assembly, angegeben. Die Ausgabe von Busco ist in vier Kategorien unterteilt: Vollständig

and Einzelkopie (Single), Vollständig und Dupliziert (Duplicate), Fragmentiert (Fragmented) und Fehlend (Missing) [URL-18, 2022]. Die Busco Completeness ist die Summe von Single und Duplicate. Diese beträgt bei den Chromosom-Level-Assemblies für den Datensatz *Laurasiatheria_odb10* zwischen 95 und 96 %. Folglich sollte eine Busco Completeness von über 95 % auch für ein vollständiges Assembly des Schafpudelgenoms sprechen (Player et al., 2021, Edwards et al., 2021, Wang et al., 2021a). Die Busco Completeness der Scaffold- und Contig-Level-Assemblies beträgt 34 % (Chihuahua), 40,3 % (Beagle) und 86,5 % (BS72). Das Assembly des Isolats BS72 liegt noch in einem akzeptablen Bereich, hingegen Referenz Chihuahua und Beagle haben eine hohe Anzahl an fragmentierten und fehlenden BUSCOs (Benchmarking Universal Single-Copy Orthologs).
In Abbildung 5.8 D ist die Anzahl der Contigs bestimmter Größen dargestellt. Im besten Fall ist die Anzahl der Contigs gleich der Anzahl an Chromosomen im haploiden Chromosomensatz des Organismus. Bei Referenz Labrador (Rüde) sollten folglich 40 Chromosomen assembliert werden [URL-01, 2020]. Zusätzlich zu den 40 Chromosomen sind allerdings noch 334 Sequenzen entstanden, die nicht zu den Chromosomen zugeordnet werden konnten. Die Contiganzahl beträgt bei Referenz Labrador folglich 374. Die Chromosomenlängen für Hundegenome liegen deutlich über den in Abbildung 5.8 D unterschiedenen Contiglängen. Die Länge des kürzesten Chromosoms (Chr38) bei Referenz Labrador ist mit 24,1 Mbp und das Y Chromosom mit 3,9 Mbp angegeben, was deutlich über den 50.000 bp liegt, die im Diagramm angegeben werden [URL-01, 2020]. Bei Referenz Labrador wird zudem deutlich, dass die Länge bei fast allen Contigs über den 50.000 bp liegt und nur wenige Contigs kürzer sind. Referenz Boxer und Wolf haben die wenigsten Contigs und sind aus diesem Grund die qualitativ hochwertigsten Assemblies in Bezug auf die Contiganzahl. Die Referenzen Beagle, BS72 und Chihuahua haben mit Abstand die meisten Contigs. Zudem sind starke Unterschiede in der Länge der Contigs erkennbar. Beispielsweise sind 90,4 % der Contigs bei BS72 kürzer als 1.000 bp, was auf ein stark fragmentiertes Assembly hindeutet. Die Chromosomen-Level-Assemblies (Referenz Schäferhund, Basenji, Labrador, Boxer, Wolf und Dogge) haben eine geringe Contiganzahl und sind folglich alle qualitativ hochwertig [Wang et al., 2021a].
In Abbildung 5.8 E ist die Anzahl der Contig-Misassemblies im Vergleich zu Referenz Boxer aufgetragen. Die Contig-Misassemblies können in drei Arten unterteilt werden: Relokationen, Translokationen und Inversionen. In der Summe erhält man die Gesamtzahl der Contig-Misassemblies. Gleiches wird für die Scaffold-Misassemblies getan. Die Referenz Boxer ist der Vergleich des Assemblies mit sich selbst und folglich ist die Anzahl der Misassemblies Null. Die Referenz Schäferhund hat die meisten Misassemblies, was vermutlich auf die Unterschiede zwischen den Rassen (Vergleich mit Boxergenom) zurückzuführen ist. Schon bei der Assemblygröße können die größeren Unterschiede zwischen den Rassen abgeleitet werden, da Referenz Schäferhund mit 2,482 Gbp das größte

Assembly aufwies. Die geringe Anzahl an Misassemblies bei der Referenz Chihuahua ist auf die starke Fragmentierung des Assemblies zurückzuführen. Da der N50 bei Referenz Chihuahua nur 9.167 bp, im Vergleich zu den 63.738.581 bp bei Referenz Boxer, beträgt, ist es unwahrscheinlicher, dass Contigs fehlerhaft assembliert wurden [Wang et al., 2021b]. Erst beim Scaffolding würden Misassemblies deutlich werden, wenn die Lücken zwischen den Contigs aufgefüllt werden [Gurevich et al., 2013].

In Abbildung 5.8 F werden die unbestimmten Basen (N's), Mismatches und InDels pro 100 kbp im Vergleich zu Referenz Boxer betrachtet. Da bei Referenz Boxer wieder das Assembly mit sich selbst verglichen wird, ist hier je ein Wert von Null zu erwarten. Erstaunlicherweise liegen die Werte leicht über Null. Der Grund hierfür kann nur auf den Prozess zur Berechnung dieser Werte mit Quast zurückzuführen sein, da keine Unterschiede vorhanden sind, wenn ein Assembly mit sich selbst verglichen wird. Wie bereits beschrieben sind die Unterschiede bei den Chromosomen-Level-Assemblies zu einem großen Teil auf die Unterschiede in den Rassen zurückzuführen. Die Mismatches und InDel-Fehler können hiermit bei den Chromosomen-Level Assemblies erklärt werden. Die unbestimmten Basen sind allerdings ausschließlich durch Fehler während des Assembly-Prozesses zu erklären, da diese unabhängig von der Referenz sind. Besonders die Referenz Labrador hat mit 528,3 N's pro 100 kbp viele unbestimmte Basen. Dies lässt sich auf ein schlechtes Assembly von Chromosom X zurückführen, da hier die Anzahl der unbestimmten Basen 3.589 N's pro 100 kbp beträgt.

In Abbildung 5.8 G ist die Genom-Coverage und die Duplikationsrate im Vergleich zu Referenz Boxer angegeben. Die Genom-Coverage ist der Prozentteil an alignierten Basen des Assemblies im Referenzgenom [URL-06, 2022]. Die Duplikationsrate ist die Anzahl der alignierten Basen im Assembly geteilt durch die Anzahl der alignierten Basen im Referenzgenom [URL-06, 2022]. Wenn ein Assembly mit sich selbst verglichen werden würde, wie es bei Referenz Boxer getan wurde, dann wären beide Werte genau 100 %. In der Realität liegt die Genom-Coverage unter 100 % und die Duplikationsrate über 100 %, entweder aufgrund der Unterschiede zwischen den Rassen oder durch Fehler beim Assembly. Die Unterschiede in den Chromosomen-Level-Assemblies beziehen sich wahrscheinlich fast ausschließlich auf die Unterschiede in den Rassen, aufgrund der hohen Qualität dieser. Hier wären beispielsweise die Unterschiede zwischen Schäferhund und Boxer zu nennen, wobei der Schäferhund ein deutlich größeres Genom aufweist (2,482 Gbp im Vergleich zu den 2,312 Gbp bei Referenz Boxer) (URL-04, 2020, URL-11, 2020). Aufgrund dessen ist es möglich, dass im Genom des Schäferhundes einige Sequenzbereiche häufiger vorkommen als im Genom des Boxers, was die Duplikationsrate deutlich erhöht. Die deutlich geringere Genom-Coverage bei der Referenz Chihuahua ist wahrscheinlich auf das stark fragmentierte Assembly zurückzuführen,

wodurch einige Sequenzbereiche im diesem Assembly nicht vorhanden sind. Gleiches kann bei der Referenz Beagle gesagt werden. Jedoch ist von der Genom-Coverage von Referenz Beagle ausgehend ein nicht so stark fragmentiertes Assembly zu erwarten, wie bei Referenz Chihuahua. Dies kann mit der Contiganzahl bestätigt werden, welche bei Referenz Beagle 195.936 und bei Referenz Chihuahua 354.927 Contigs beträgt.

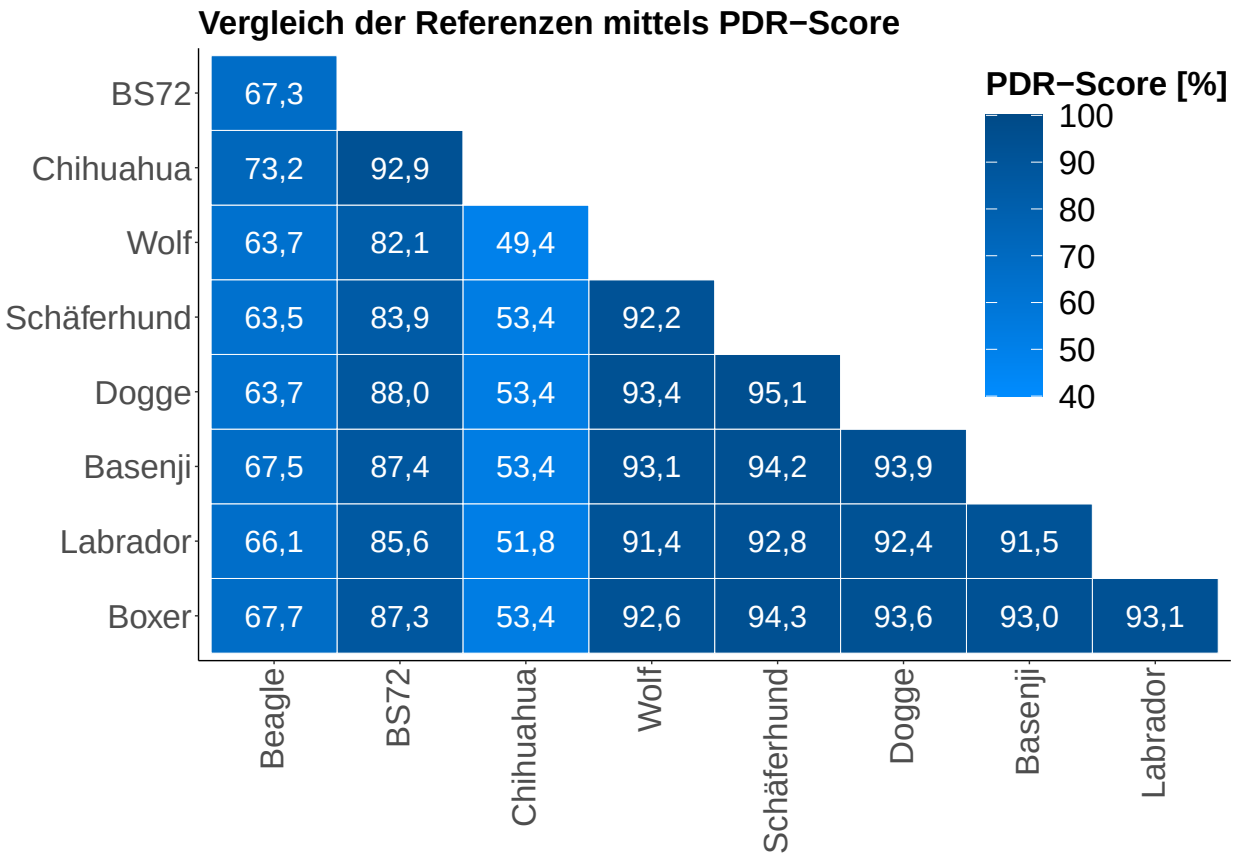

Abbildung 5.9: PDR-Score der Referenzen

In Abbildung 5.9 ist der PDR-Score beim Vergleich zweier Referenzen miteinander dargestellt. Der *Pairwise-Distance-Reconstruction*-Score wird berechnet, indem zufällig zwei Sequenzabschnitte auf einem Contig der Referenz ausgewählt werden und deren Distanz bestimmt wird [Xie und Wong, 2021]. Diese beiden Sequenzabschnitte werden im Folgenden auch auf dem Assembly lokalisiert und deren Distanz berechnet. Der PDR-Score ist der Anteil an Sequenzpaaren mit gleicher Distanz in Referenz und Assembly [Xie und Wong, 2021]. Der PDR-Score berücksichtigt sowohl die Contiguity, Completeness als auch Correctness und fasst diese drei Aspekte in einem Wert zusammen [Xie und Wong, 2021]. Die Chromosomen-Level-Assemblies haben alle einen PDR-Score von über 90 %, wenn sie miteinander vergleichen werden. Folglich sollte auch das Schafpudel-Assembly, wenn dies vollständig assembliert wurde, einen PDR-Score von über 90 % beim Vergleich mit den

Chromosomen-Level-Assemblies erreichen. Der PDR-Score der Referenz BS72 liegt zwischen 80 und 90 %, bei Referenzen Beagle zwischen 60 und 70 % und bei Referenz Chihuahua zwischen 50 und 60 %. Referenz Chihuahua ist laut dem PDR-Score das schlechteste Assembly der Referenzen.

5.3 Short-Read-Assembly

5.3.1 Initiale Assemblies

Die ersten initialen Assemblies wurden nur mit den Illumina-Short-Reads durchgeführt. Hierfür wurden die Assembler Abyss, SOAPdenovo2, MaSuRCA und SparseAssembler ausgewählt. In Abbildung 5.10 sind die Qualitäts-Metriken der initialen Assemblies visualisiert.

Der N(x)-Plot in Abbildung 5.10 A zeigt, dass das MaSuRCA-Assembly den höchsten N50 (11.224) als auch auN (14.440) erzeugt hat. Die Assemblies von Abyss, SOAP und Sparse sind im N50 und auN ähnlich. N50 und auN sind insgesamt bei den Short-Read-Assemblies sehr gering. Ein vollständiges Assembly, wie die Chromosomen-Level-Assemblies der Referenzen, würde Werte von 60.000.000 bis 70.000.000 für N50 und auN erzeugen (vgl. Abb. 5.8 A). Ein niedriger N50 und auN weist auf eine starke Fragmentierung bei den initialen Assemblies hin [URL-06, 2022]. Diese starke Fragmentierung kann mit der hohen Contiganzahl in Abbildung 5.10 D bestätigt werden. Das Abyss-Assembly hat insgesamt 2.377.250, SOAP 3.390.680, MaSuRCA 373.584 und Sparse 5.472.759 Contigs. Im Vergleich hierzu hat Referenz Schäferhund die höchste Anzahl an Contigs der Chromosomen-Level-Assemblies mit 2.198, was deutlich unter der Contiganzahl der initialen Assemblies liegt (vgl. Abb. 5.8 D).

Die erwartete Assemblygröße von 2,3 bis 2,5 Gbp ist bei keinem der Assemblies erreicht worden (Abb. 5.10 B). Die Werte liegen mit 2,231 Gbp (Abyss), 2,162 Gbp (SOAP), 2,245 Gbp (MaSuRCA) und 2,139 Gbp (Sparse) leicht unter der zu erwartenden Genomgröße. Die von MaSuRCA vor dem Assembly geschätzte Genomgröße des Schafpudels beträgt 2,306 Gbp.

Die Busco Completeness (Abb. 5.10 C) ist mit 27,1 % (Abyss), 30,4 % (SOAP), 41,3 % (MaSuRCA) und 26,4 % (Sparse) vergleichbar mit der Busco Completeness von Referenz Beagle (40,3 %) und Chihuahua (34,0 %). Die geringe Busco Completeness ist auf die starke Fragmentierung zurückzuführen, welche auch bei den beiden genannten Referenzen vorhanden ist (vgl. Abb. 5.8 C) [Simão et al., 2015].

Die Anzahl an Misassemblies (Abb. 5.10 E) unterscheidet sich stark. Sparse erzeugte die mit Abstand meisten. Aufgrund dieser hohen Anzahl an Misasemblies wird das Sparse-Assembly nicht weiter verwendet. SOAP hat die wenigsten Misassemblies. Dies kann durch die starke Fragmentierung entstanden sein. Zudem ist wie in Abb. 5.10 F eine große Anzahl an unbestimmten Basen im

Abbildung 5.10: Qualitäts-Metriken der initialen Assemblies visualisiert

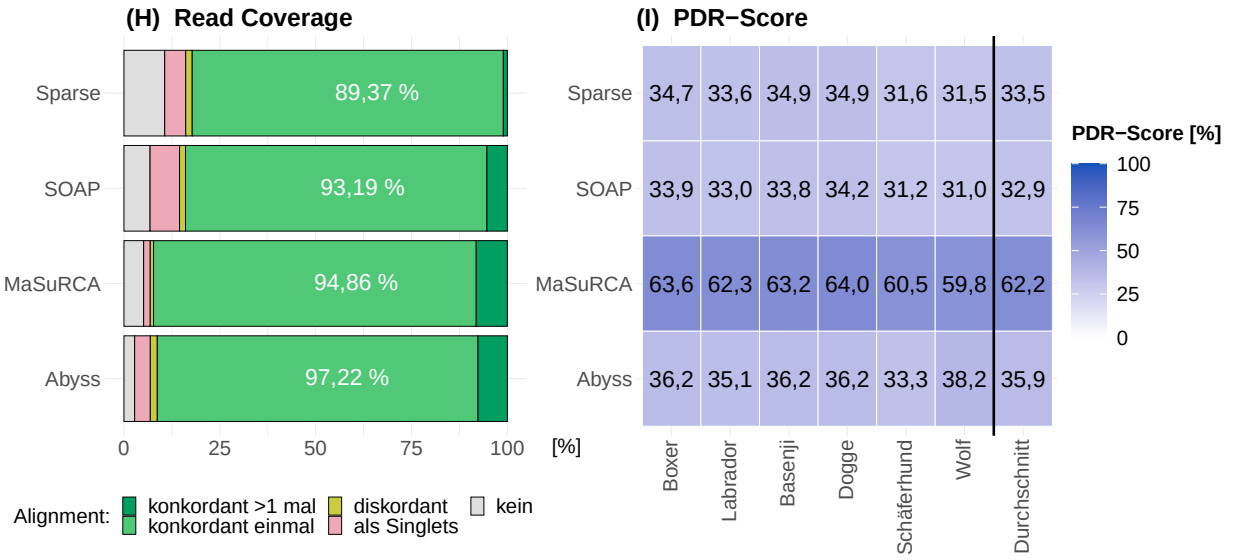

Abbildung 5.11: Qualitäts-Metriken der initialen Assemblies visualisiert (fortgesetzt)

SOAP-Assembly vorhanden. Da es unbestimmte Basen sind zählen diese nicht als Misassemblies [URL-06, 2022]. Die geringere Anzahl an unbestimmten Basen beim Abyss-Assembly führt zu der höheren Anzahl an Misassemblies im Vergleich zum SOAP-Assembly. Das MaSuRCA-Assembly hat eine größere Anzahl an Misassemblies, da längere Contigs vorliegen und folglich mehr Relokationen, Translokationen und Inversionen auf den Contigs vorkommen können [URL-06, 2022].

Die Genom-Converage (Abb. 5.10 G) ist beim SOAP- und Sparse-Assembly mit 90,4 % und 90,7 % etwas geringer als die anderen initialen Assemblies. Dies ist auf die starke Fragmentierung und auf die hohe Anzahl an unbestimmten Basen zurückzuführen [URL-06, 2022]. Die Genom-Coverage beim Abyss- und MaSuRCA-Assembly liegt bei 93 und 94 %. Ein qualitativ hochwertiges Assembly sollte Werte von 98 % und höher erreichen, wie es bei den Chromosom-Level-Assemblies der Fall ist (vgl. 5.8 C) [Gurevich et al., 2013]. Die Duplikationsrate liegt bei allen initialen Assemblies bei leicht über 100 %. Die Werte sind vergleichbar mit den Referenzen.

Die Read-Coverage (Abb. 5.10 H), welche mit einem Bowtie2-Alignment der Illumina-Reads bestimmt wurde, kann auch als Qualitäts-Metrik für ein Assembly genutzt werden. Die Alignment-Rate, welche als Wert in jedem der Balken als Prozentsatz dargestellt ist, ist die Summe aller Reads, die in einer der vier aufgeführten Arten mit dem Assembly alignieren. Konkordant bezeichnet ein Read-Alignment, bei welchem die Paired-End-Reads auch als Paare auf dem Assembly auftreten [URL-17, 2022]. „Konkordant >1 mal“ bezeichnet die Paired-End-Reads die mehr als einmal auf dem Assembly alignieren. Diskordant bezeichnet ein Read-Alignment, bei dem die Paired-End-Reads zwar auf dem Assembly aligniert wurden, aber nicht als Paare auftreten [URL-17, 2022]. Singlets bezeichnen das Read-Alignment, bei dem nur einer der Forward- oder Reverse-Reads

auf dem Assembly aligniert und nicht beide [URL-17, 2022]. Wenn die Paired-End-Reads nicht alignieren wird hier von „Kein Alignment“ gesprochen. Bei der Betrachtung der Read-Coverage ist erkennbar, dass Abyss mit 97,22 % die höchste Read-Coverage hat. Der Prozentteil von konkordant alignierten Reads ist jedoch beim MaSuRCA-Assembly am höchsten. Das Abyss-, SOAP-, und Sparse-Assembly haben einen beträchtlichen Teil an Reads, die als Singlets oder diskordant aligniert sind. Im besten Fall wären bei einem Assembly 100 % der Reads konkordant einmal aligniert. Das MaSuRCA-Assembly liefert folglich anhand der Read-Coverage das qualitativ beste Assembly der vier durchgeführten initialen Assemblies.
Der PDR-Score (Abb. 5.10 I) liegt beim Abyss-, SOAP- und Sparse-Assembly bei 30 bis 40 %. Das MaSuRCA-Assembly hingegen hat einen PDR-Score von 62,2 % erreicht. Das MaSuRCA-Assembly ist folglich das qualitativ hochwertigste der vier initialen Assemblies. Werte von über 90 % sind das Ziel, da die Chromosomen-Level-Assemblies beim Vergleich miteinander PDR-Scores von über 90 % erreichen (siehe Abb. 5.9).
Bei der Betrachtung aller Qualitäts-Metriken kann geschlossen werden, dass MaSuRCA das mit Abstand beste initiale Assembly erstellt hat. Jedoch ist auch dieses noch sehr fragmentiert und unvollständig im Vergleich zu dem Chromosomen-Level-Assemblies der Referenzen. Folglich müssen die Assemblies noch weiter verbessert werden.

5.3.2 Assembly Merging

Im nächsten Schritt des Assembly-Prozesses wurden drei der initialen Assemblies gemerged. Die dazu verwendete initialen Assemblies wurden mit A (Abyss-Assembly), S (SOAP-Assembly) und M (MaSuRCA-Assembly) bezeichnet.
In den Diagrammen in Abbildung 5.12 sind diese vier Merged-Assemblies dargestellt. Im N(x)-Plot (Abb. 5.12 A) wird deutlich, dass mit dem Tool GAM-NGS ein geringerer N50 und auN erreicht wird, als bei MAC. Eine weitere Schlussfolgerung aus dem Diagramm ist, dass das Merging aller drei initialen Assemblies einen höheren N50 und auN erzeugt, als nur das Merging des Abyss- und SOAP-Assemblies. Dies ist darauf zurückzuführen, dass das MaSuRCA-Assembly einen höheren N50 und auN als das Abyss- und SOAP-Assembly hat und folglich beim Merging die längeren Contigsequenzen mit übernommen werden. Dabei ist jedoch zu beachten, dass der N50 bei GAM-NGS_A-S geringer ist als bei den initialen Assemblies (Abyss und SOAP) und bei GAM-NGS_A-S-M ein geringerer N50 vorliegt als beim MaSuRCA-Assembly. Es ist jedoch nicht klar, weshalb das Merging den N50 verringert, da GAM-NGS speziell dafür entwickelt wurde, zwei Assemblies aus NGS-Sequenzdaten, also Short-Read-Assemblies zu mergen [Vicedomini et al, 2013]. Es ist zu vermuten, dass die Insert-Größe, also die Anzahl an Basen zwischen Forward- und Reverse-Read der

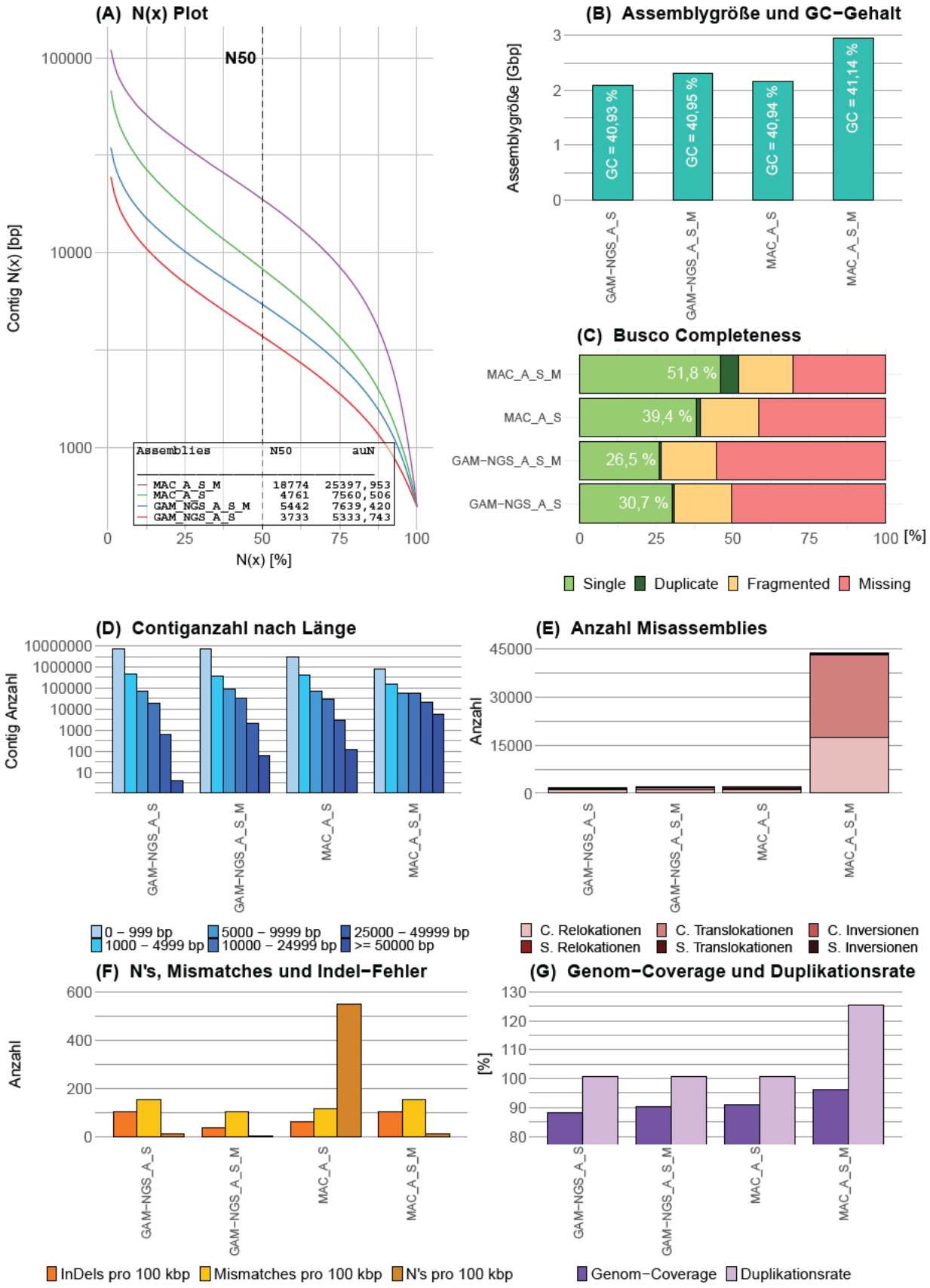

Abbildung 5.12: Qualitäts-Metriken der Merged-Assemblies visualisiert

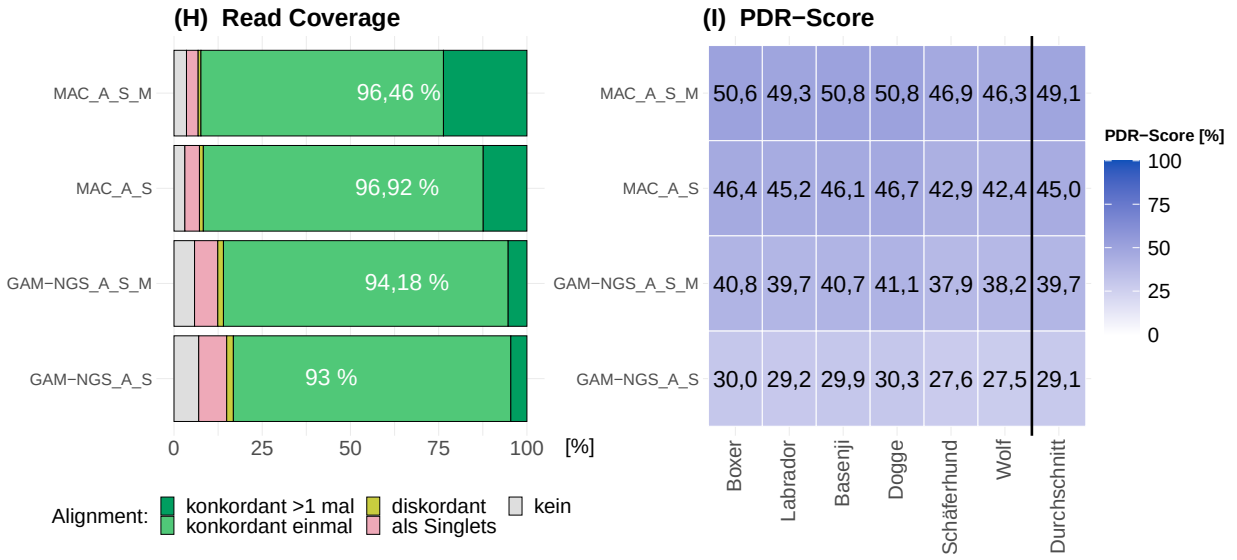

Abbildung 5.13: Qualitäts-Metriken der Merged-Assemblies visualisiert (fortgesetzt)

Paired-End-Reads zu gering ist, sodass das Scaffolding nicht effektiv war [Vicedomini et al, 2013]. Beim MAC-Merging hat sich der N50 und auN im Vergleich zu den initialen Assemblies erhöht (MaSuRCA: 11.224, GAM-NGS_A-S-M: 18.774). Das Merging aller initialen Assemblies mit MAC (MAC_A-S-M) führt jedoch zu einer deutlich erhöhten Assemblygröße (Abb. 5.12 B), Misassembly-Anzahl (Abb. 5.12 E) und Duplikationsrate (Abb. 5.12 G). Dies stammt daher, dass Contigs, die zwar ähnlich sind, aber nicht gemerged wurden, einfach mit übernommen werden und folglich Duplikationen im Assembly-File vorhanden sind [Tang et al., 2020]. Dies ist ein großes Problem, da dies das Assembly verfälscht.

Diese mehrfach vorkommenden Sequenzen sind auch bei der Busco Completeness erkennbar, da hier der Prozentteil der Duplicates 5,8 % beträgt (Abb. 5.12 C). Durch diese mehrfach vorkommenden Sequenzen erhöht sich beim MAC Merging die Busco Completeness insgesamt. Im Gegensatz dazu verringert sich die Busco Completeness beim Merging mit GAM-NGS. Die Read-Coverage bestätigt die mehrfach vorkommenden Sequenzen in den MAC-Assemblies (Abb. 5.13 H). Der Prozentsatz konkordant mehr als einmal alignierter Reads beträgt bei MAC_A-S 12,37 % und bei MAC_A-S-M sogar 26,65 %. Bei dem GAM-NGS-Assemblies ist die Read-Coverage vergleichbar mit den initialen Assemblies (Abb. 5.11 H).

Jedoch liegen alle anderen in Abbildung 5.12 dargestellten Metriken im Erwartungsbereich. Insgesamt kann gesagt werden, dass das Merging der initialen Assemblies mit GAM-NGS und MAC keine uneingeschränkte Verbesserung der Assembly-Qualität mit sich gebracht hat.

5.3.3 Chromosome-Scaffolding

Die Qualitäts-Metriken der Scaffolded-Assemblies sind in Abbildung 5.14 visualisiert. Alle mit Chromosome-Scaffolder erstellten Assemblies werden mit „Mscaf“ bezeichnet. Durch das Scaffolding wurden die Contigs an die Chromosomenstruktur von Referenz Labrador aligniert, sodass die 40 Chromosomen des Hundegenoms entstanden sein sollten. Anhand des N(x)-Plots (Abb. 5.14 A) wird deutlich, dass die Scaffolded-Assemblies, aufgrund des Alignierens an die Referenz, vom Verlauf her alle ähnlich sind. Der N50 und auN liegen im Bereich von 60 bis 70 Mbp. Diese Werte sind gleich zu dem Bereich der bei den Chromosomen-Level-Assemblies der Referenzen erhalten wurde (vgl. Abb. 5.8 A). Folglich sollten die Chromosomen erfolgreich zusammengebaut worden sein.

Bei der Betrachtung der Assembly-Größe sollten Werte im Bereich von 2,3 bis 2,5 Gbp erreicht werden. Bei allen Scaffolded-Assemblies mit Ausnahme von Mscaf_MAC_A-S-M liegt die Genomgröße in Erwartungsbereich von 2,3 bis 2,5 Gbp. Mscaf_MAC_A-S-M erzeugte eine Assemblygröße von fast 3,0 Gbp, was deutlich über der Genomgröße aller Referenzen liegt (vgl. Abb. 5.14 B). Diese erhöhte Genomgröße ist jedoch bereits beim Merging aufgetreten und wurde folglich auch beim Scaffolding mit übernommen (vgl. Abb. 5.12 B).

Die Busco Completeness (Abb. 5.14 C) liegt bei allen Scaffolded-Assemblies über 90 %. Dies ist ein akzeptabler Bereich aber trotzdem noch verbesserungswürdig (vgl. 5.10 C). Das Mscaf_GAM-NGS_A-S-M-Assembly lieferte die beste Busco Completeness mit 95 %. Das Mscaf_MaSuRCA-Assembly hat jedoch nur eine Busco Completeness von 91 % und Mscaf_MAC_A-S-M-Assembly nur von 90 %.

Die Contiganzahl ist bei diesen beiden Scaffolded-Assemblies im Vergleich zu den anderen Assemblies leicht erhöht. Daraus lässt sich schließen, dass kürzere Contigs besser an die Referenz aligniert werden können, da nicht alignierte Contigs nicht aus der fasta-Datei des Assemblies entfernt, sondern am Ende angehängt werden. Insgesamt ist die Contiganzahl bei allen Assemblies im Vergleich zu den initialen und Merged-Assemblies jedoch deutlich verringert worden. Die Contiganzahl (Abb. 5.14 D) liegt aber trotzdem über der Contiganzahl von Referenz Labrador, welche als Referenz für das Scaffolding genutzt wurde (vgl. Abb 5.8 D Referenz Boxer).

Bei der Anzahl der Missassemblies sticht Mscaf_MAC_A-S-M hervor. Die vielen Missassemblies sind beim Merging entstanden (vgl. Abb. 5.12 E MAC_A-S-M) und wurden beim Scaffolding mit übernommen.

Durch das Scaffolding werden viele der Lücken mit unbestimmten Basen gefüllt, was anhand von Abbildung 5.14 F deutlich wird. Die wenigsten N’s sind bei Mscaf_MaSuRCA-Assembly mit knapp 3000 N’s pro 100 kbp und die meisten bei Mscaf_MAC_A-S-M-Assembly entstanden. Bei

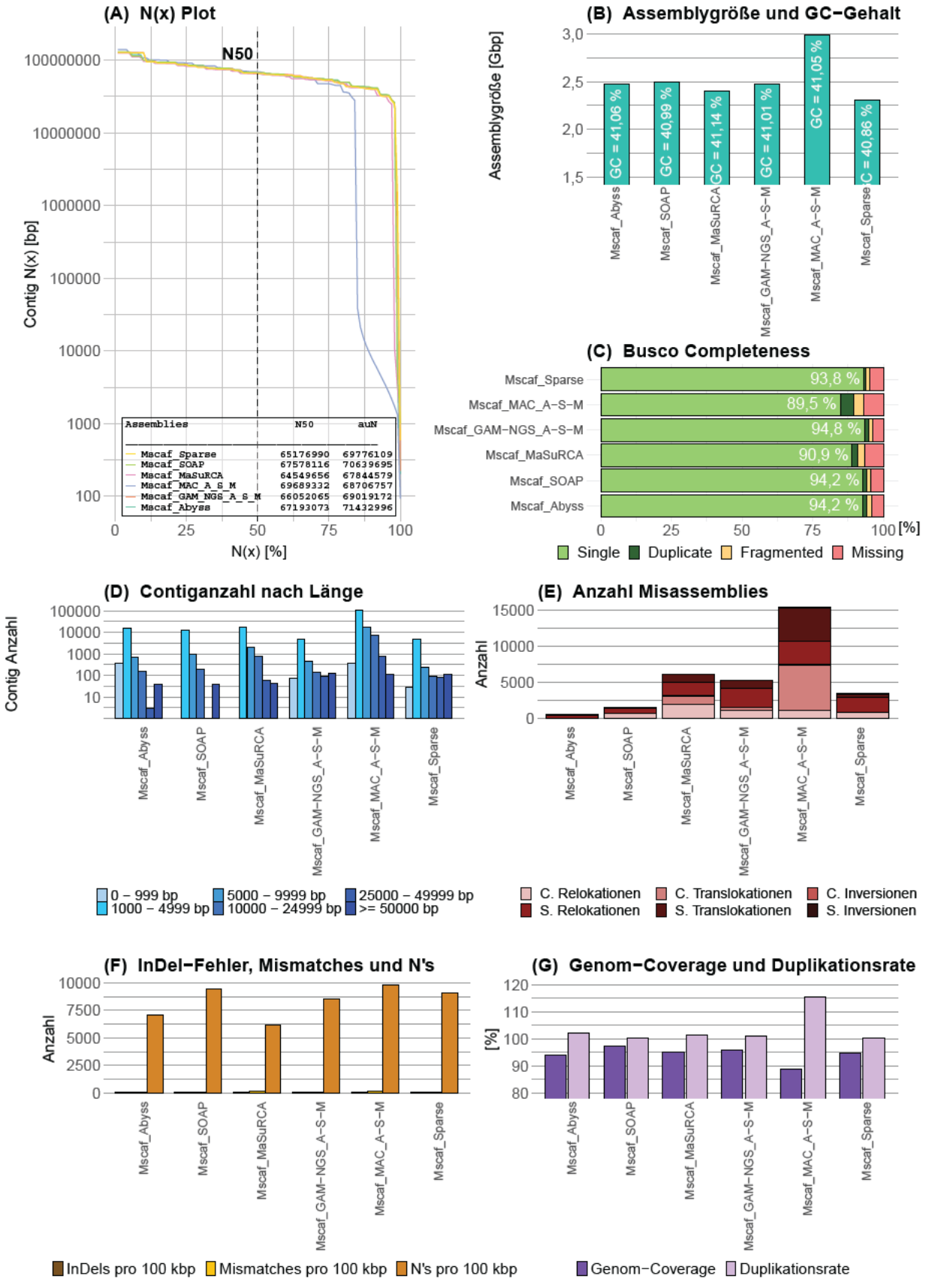

Abbildung 5.14: Qualitäts-Metriken der Scaffolded-Assemblies visualisiert

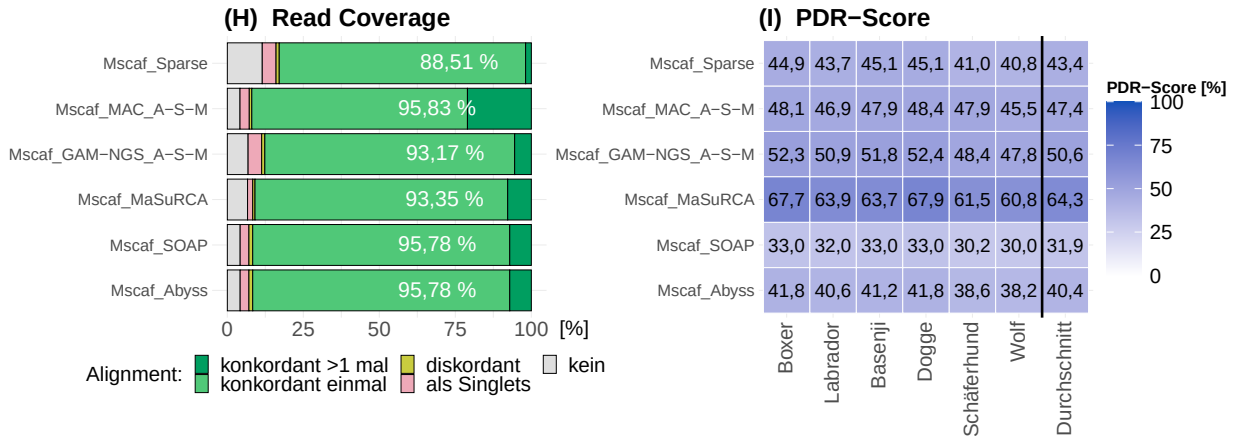

Abbildung 5.15: Qualitäts-Metriken der Scaffolded-Assemblies visualisiert (fortgesetzt)

der Genom-Coverage und Duplikationsrate (Abb. 5.14 G) fällt ebenfalls das Mscaf_MAC_A-S-M-Assembly negativ auf, da hier die Genom-Coverage deutlich verringert und die Duplikationsrate erhöht ist. Die anderen Scaffolded-Assemblies liegen alle nahe bei 100 %, was vergleichbar mit den Referenzen ist (siehe Abb. 5.8 G).

Die Read-Coverage (Abb. 5.15 H) ist bei allen Scaffolded-Assemblies, mit Ausnahme von Sparse sehr ähnlich, mit geringen Unterschieden bei nicht alinierten Reads und Singlets. Beim Mscaf_MAC_A-S-M-Assembly wurde die hohe Anzahl an konkordant mehr als einmal alignierter Reads aus dem Merging mit übernommen (siehe Abb. 5.12 H). Der PDR-Score (Abb. 5.15 I) wurde durch das Scaffolding bei allen Assemblies leicht erhöht. Auch hier hat das Mscaf_MaSuRCA-Assembly den höchsten PDR-Score.

Insgesamt kann geschlossen werden, dass das Scaffolded-MaSuRCA-Hybrid-Assembly das beste Assembly geliefert hat, da die Chromosomen vollständig zusammengebaut wurden, die Assemblygröße im Erwartungsbereich liegt, die Busco Completeness mit den Referenzen vergleichbar ist, die Contiganzahl verhältnismäßig gering ist, mäßig viele Misassemblies vorhanden sind, die wenigsten unbestimmten Basen zum Gap-Filling verwendet wurden, das Genom fast vollständig abgedeckt wurde und nicht viele duplizierte Sequenzen vorhanden sind.

5.4 Long-Read-Assembly

Das Flye-Assembly, welches mit den unkorrigierten Nanopore-Reads erzeugt wurde, wird in Abbildung 5.16 mit „Flye“ und das Assembly aus den fmlrc-korrigierten Nanopore-Reads mit „Flye_fmlrc“ bezeichnet. Das Pilon- und Polca-Polishing ist gekennzeichnet durch das anhängen von „_pilon“, bzw. „_polca“. Wenn ein Polishing-Schritt mehrmals durchgeführt wurde, wird die Anzahl mit angegeben (1x, 2x). Das Scaffolding mit Samba und Chromosome-Scaffolder ist gekennzeichnet durch

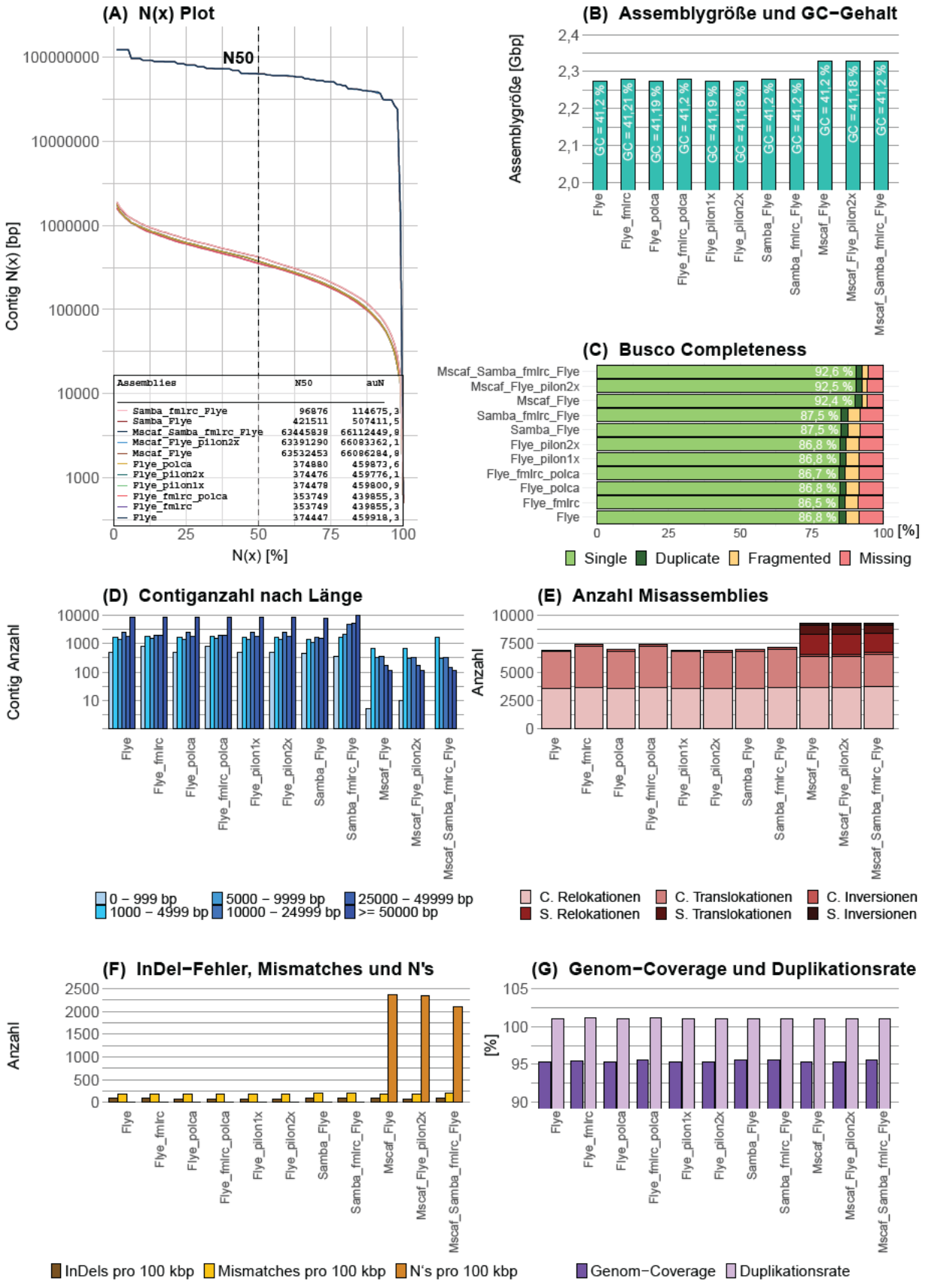

Abbildung 5.16: Qualitäts-Metriken der Long-Read-Assemblies visualisiert

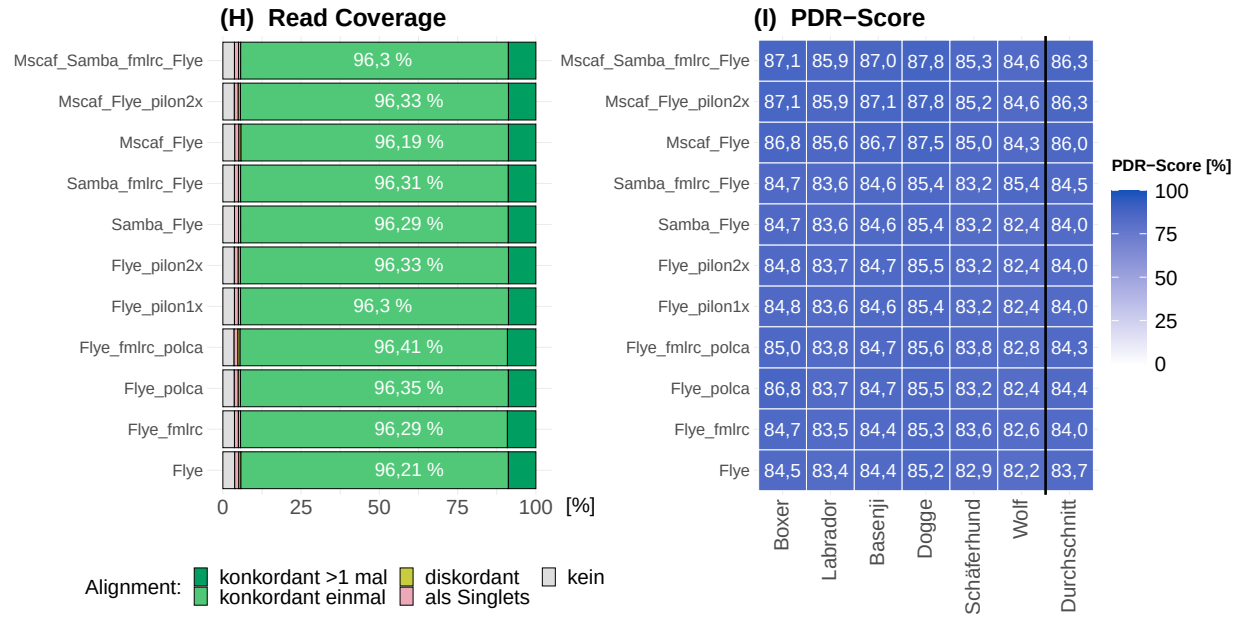

Abbildung 5.17: Qualitäts-Metriken der Long-Read-Assemblies visualisiert (fortgesetzt)

die Präfixe „Samba_“, bzw. „Mscaf_“. Bei Samba wurden zum einen die unkorrigierten Nanopore-Reads (Samba_) und zum anderen die fmlrc-korrigierten Nanpore-Reads (Samba_fmlrc_) verwendet.

Die Qualitäts-Metriken der Long-Read-Assemblies sind in Abbildung 5.16 dargestellt. Das Long-Read-Assembly mit Flye hat trotz der geringen Sequencing-Depth ein qualitativ gutes Assembly geliefert. Im Vergleich zu den initialen Short-Read-Assemblies (als Vergleich wird hier das MaSuRCA-Assembly genutzt) hat das Long-Read-Assembly einen 20 mal so hohen N50, die Busco Completeness ist um 40 % höher, enthält 95 % weniger Contigs und hat einen um etwa 20 % höheren PDR-Score. Der einzige größere Nachteil im Vergleich zum MaSuRCA-Assembly ist die höhere Anzahl an Misasemblies (Flye: 6.946, MaSuRCA: 4.978). Insgesamt kann gesagt werden, dass das Long-Read-Assembly im Vergleich zu den Short-Read-Assemblies ein deutlich besseres Assembly geliefert hat . Da die Sequencing-Depth mit 7x deutlich unter den vom Flye-Assembler empfohlenen 40x liegt (URL-10, 2022; Kolmogorov et al., 2019), ist es höchst wahrscheinlich, dass das Long-Read-Assembly mit zusätzlichen Nanopore-Reads noch stark verbessert werden kann.

Das initiale Long-Read-Assembly mit den fmlrc-korrigierten Nanopore-Reads führt zu einer minimal schlechteren Qualität im Vergleich zum Assembly mit den unkorrigierten Nanopore-Reads. Der N50 ist um 700 geringer, die Contiganzahl um 700 Contigs höher, die Busco Completeness um 0,3 % geringer und die Misassembly-Anzahl um 500 Misasemblies größer. Der Grund hierfür könnte der in Flye eingebaute Error-Correction Schritt sein, der schlechter mit den korrigierten Reads umgehen kann, oder sogar eine fehlerhafte Error-Correction der Nanopore-Reads von fmlrc

[Kolmogorov et al., 2019]. Aufgrund dessen, dass, wenn die fmlrc-korrigierten Nanopore-Reads verwendet werden, es nicht zu einer Verbesserung des Assemblies kommt, ist Letzteres wahrscheinlicher.
Das Pilon-Polishing des Long-Read-Assemblies führte zu einer minimalen Verbesserung des Assemblies. Das Pilon-Polishing sollte folglich nur als letzter Schritt im Assembly-Prozess verwendet werden, wenn das Assembly nicht weiter von anderen Tools verbessert werden kann. Gleiches gilt für das Polca-Polishing.
Das Samba-Scaffolding führte zu leichten Verbesserungen des Assemblies. Die Contiganzahl wurde durch das Scaffolding um etwa 1.700 gesenkt, der N50 und auN um etwa 4.000 und die Busco Completeness um 0,7 % erhöht. Der einzige negative Faktor ist die Erhöhung der Misassembly-Anzahl um etwa 200. Diese Verbesserung ist möglich durch das Schließen der Lücken zwischen den Contigs mit den Nanopore-Reads. Folglich werden hier nicht wie beim Chromosome-Scaffolder unbestimmte Basen, sondern Nukleotide in das Assembly eingebaut [Zimin und Salzberg, 2022]. Dies lässt sich durch die Anzahl an unbestimmten Basen, welche auch nach dem Samba-Scaffolding Null beträgt, bestätigen (Abb. 5.16 F).
Nur beim Referenz-basierten-Scaffolding mit Chromosome-Scaffolder sind beträchtliche Unterschiede auf den ersten Blick erkennbar. Da die entstandenen Contigs an das Referenzgenom aligniert und die fehlenden Bereiche mit unbestimmten Basen ergänzt werden, entstehen weniger, aber deutlich längere Sequenzen (Chromosomen), was den N50, den auN, die Assemblygröße, die Busco Completeness, die Anzahl an Scaffold-Missassemblies und die Anzahl an unbestimmten Basen erhöht und die Contiganzahl im Vergleich zu den anderen Long-Read-Assemblies senkt. Bei der Read-Coverage treten ähnliche Werte auf, wie bei den anderen Long-Read-Assemblies. Der PDR-Score ist um etwa 2 % erhöht. Besonders beim PDR-Score wird die hohe Qualität der Long-Read-Assemblies deutlich, da dieser im Vergleich zu dem Short-Read-Asemblies stark erhöht werden konnte. Jedoch wurden auch hier kein PDR-Score von über 90 % erreicht, wie es bei den Chromosomen-Level-Assemblies der Fall ist (vgl. Abb. 5.9).

5.5 Hybrid-Assembly

Mit dem MaSuRCA-Assembler ist es möglich ein Hybrid-Assembly durchzuführen, bei dem sowohl Short-, als auch Long-Reads im Assembly-Prozess verwendet werden. Des Weiteren wird dieses Assembly mit dem Samba-Scaffolder und dem Chromosomen-Scaffolder weiter verbessert.
Das Hybrid-Assembly mit MaSuRCA führte zu einem in jeder Hinsicht besseren Assembly im Vergleich mit dem Short-Read-Assembly mit MaSuRCA. Die Contiganzahl konnte im Vergleich mit

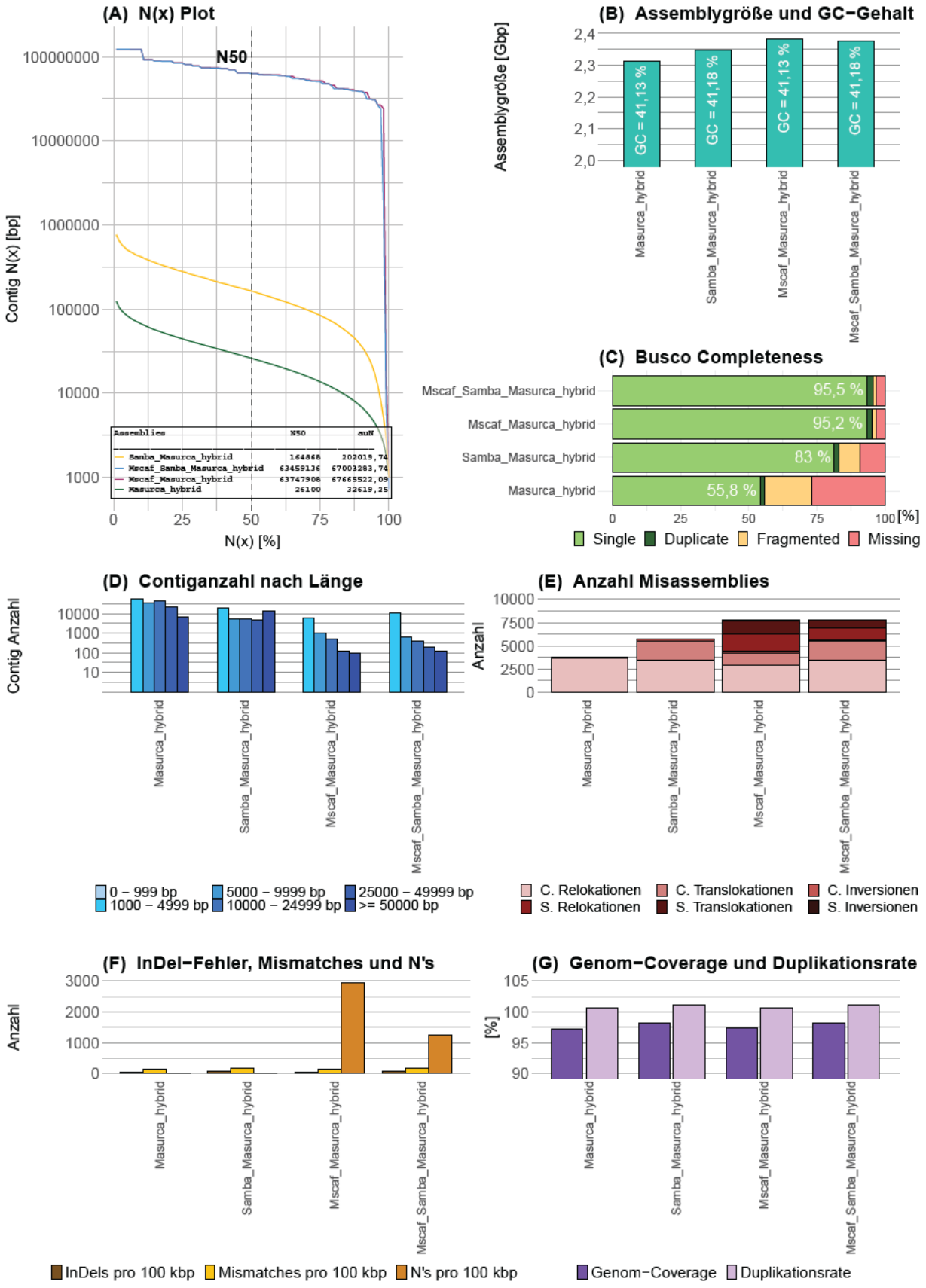

Abbildung 5.18: Qualitäts-Metriken der Hybrid-Assemblies visualisiert

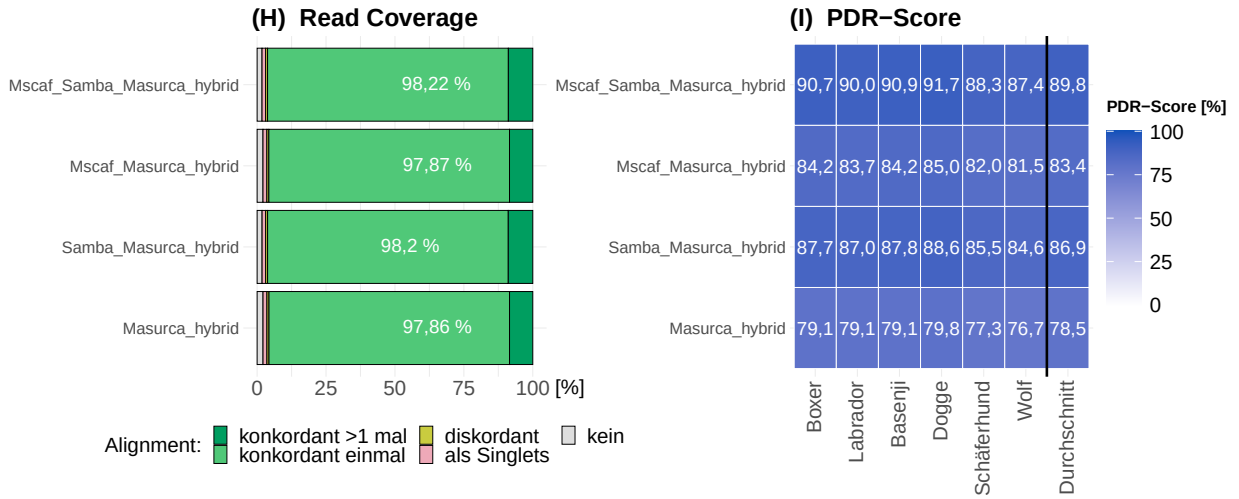

Abbildung 5.19: Qualitäts-Metriken der Hybrid-Assemblies visualisiert (fortgesetzt)

dem Short-Read-Assembly um 57 % reduziert und der N50 um das 2,36-fache erhöht werden. Das erhaltene Assembly ist dabei 2,31 Gbp groß, also im unteren Bereich des Erwartungsbereiches. Auch die Busco Completeness konnte um 14,7 % erhöht und die Missassembly-Anzahl um fast 2.000 gesenkt werden. Die Genom-Coverage ist im Vergleich zum Short-Read-Assembly erhöht mit 97,29 % um mehr als 3 % und der PDR-Score um 16,3 %.

Mit dem Scaffolding des Hybrid-Assemblies kann ein noch besseres Assembly erzeugt werden. Durch das Scaffolding mit Samba konnte die Contiganzahl um weitere 70,4 % verringert und der N50 um das 6,3-fache erhöht werden. Die Busco Completeness ist um 27,2 %, die Genome-Coverage um 0,9 %, der PDR Score um 8,4 % und die Misassembly-Anzahl ist um 41,3 % angestiegen. Dieses Scaffolded-Hybrid-Assembly ist von den Qualitäts-Metriken in etwa auf dem Level wie das Long-Read-Assembly mit Flye (vgl. Abb. 5.16). Folglich ist es möglich, dass dieses Assembly mit zusätzlichen Long-Reads noch verbessert werden kann und dies sowohl beim initialen Hybrid-Assembly als auch beim Scaffolding. Für den MaSuRCA-Assembler wird eine Sequencing-Depth von 20x empfohlen, wenn ein qualitativ gutes Hybrid-Assembly erzeugt werden soll [Zimin et al., 2013].

Das Chromosom-Scaffolding wurde sowohl mit dem Hybrid-Assembly als auch mit dem Scaffolded-Hybrid-Assembly durchgeführt. Das Chromosom-Scaffolding des Scaffolded-Hybrid-Assemblies erzeugt das in dieser Arbeit qualitativ beste Assembly. Bei diesem ist die Anzahl an unbestimmten Basen von allen Chromosom-Scaffolded-Assemblies mit nur 1.244 N's pro 100 kbp am geringsten. Das Chromosom-Scaffolded-MaSuRCA-Short-Read-Assembly hat 6.172 N's pro 100 kbp und das Chromosome-Scaffolded-Flye-Long-Read-Assembly enthält 2.359 N's pro 100 kbp. Der PDR-Score ist mit 89,8 % auch der Höchste von allen Assemblies.

5.6 Vergleich der Assembly-Ansätze

In Abbildung 5.20 wurden das MaSuRCA-Short-Read-Assembly (MaSuRCA_short), das Flye-Long-Read-Assembly (Flye), das MaSuRCA-Hybrid-Assembly (MaSuRCA_hybrid), das Samba-Scaffolded-Long-Read-Assembly (Samba_Flye), das Samba-Scaffolded-Hybrid-Assembly (Samba_MaSuRCA_hybrid) und die Chromosome-Scaffolded-Assemblies der Samba-Scaffolded-Assemblies (Mscaf_Flye und Mscaf_MaSuRCA) und des MaSuRCA-Short-Read-Assemblies (Mscaf_MaSuRCA_short) aufgetragen. Als Vergleich wurde zudem eine Referenz aufgetragen, wobei es sich um die Referenz Basenji handelt. Referenz Basenji wurde gewählt, da Referenz Boxer aufgrund der Verwendung bei Quast als Referenz nicht verwendet werden kann. Referenz Labrador entfällt ebenfalls, da es beim Chromosom-Scaffolding als Referenz genutzt wurde.

Das Short-Read-Assembly (MaSuRCA_short), welches nur mit den Illumina-Short-Reads erstellt wurde, erzeugte kein qualitativ hochwertiges Assembly. Es ist sehr stark fragmentiert (niedriger N50 bzw. auN und hohe Contiganzahl), hat eine niedrige Anzahl essentieller Gene (Busco Completeness) und den geringsten PDR-Score (direkter Vergleich eines Assemblies mit einer Referenz). Folglich ist es sinnvoll Long-Reads (z.B. Nanopore-Reads) mit im Assembly-Prozess zu verwenden [Amarasinghe et al., 2020]. Hierfür gibt es zwei Ansätze: Long-Read-Assembly und Hybrid-Assembly. Das Long-Read-Assembly erzeugte ein deutlich besseres Assembly im Vergleich zum Short-Read-Assembly. Der N50 und auN ist dabei 20x höher und die Contiganzahl ist um 95 % geringer. Die Busco Completeness ist um 45 % und der PDR-Score um 21 % höher. Beim Hybrid-Assembly konnte im Vergleich zum Short-Read-Assembly eine Verbesserung aller Qualitäts-Metriken festgestellt werden. Jedoch ist dieses Assembly schlechter als das Long-Read-Assembly. Das Scaffolding mit Samba (Samba_MaSuRCA_hybrid) konnte die Qualität des Hybrid-Assemblies deutlich erhöhen, wohingegen das Scaffolding des Long-Read-Assemblies zu keiner deutlichen Verbesserung geführt hat. Nach dem Scaffolding ist das Hybrid-Assembly von den Qualitäts-Metriken her ähnlich zum Long-Read-Assembly. Der größte Unterschied besteht in der Assemblygröße (Samba_Flye: 2,278 Gbp, und Samba_MaSuRCA_hybrid: 2,347 Gbp). Das Hybrid-Assembly ist dabei näher an der Assemblygröße der Referenz (2,329 Gbp). Die geringere Assemblygröße des Long-Read-Assemblies entsteht aufgrund der geringen Sequencing-Depth der Nanopore-Reads. Diese beträgt nur etwa 7x und ist somit deutlich geringer als die 40 x, welche von Flye für ein Long-Read-Assembly empfohlen werden (URL-10, 2022; Kolmogorov et al., 2019). Die bisher beschriebenen Assemblies zählen zum *de novo* Assembly. Ein letzter Schritt ist das Scaffolding mit dem Chromosome-Scaffolder. Dies ist ein Referenz-basiertes Assembly und kann folglich die vollständigen 40 Chomosomen des Hundegenoms erzeugen, wenn die Referenz ein Chromosomen-Level-Assembly ist. Aus diesem Grund ist der N50 und auN dieser Chromosom-Scaffolded-Assemblies auf dem Level der Referenz. Dabei

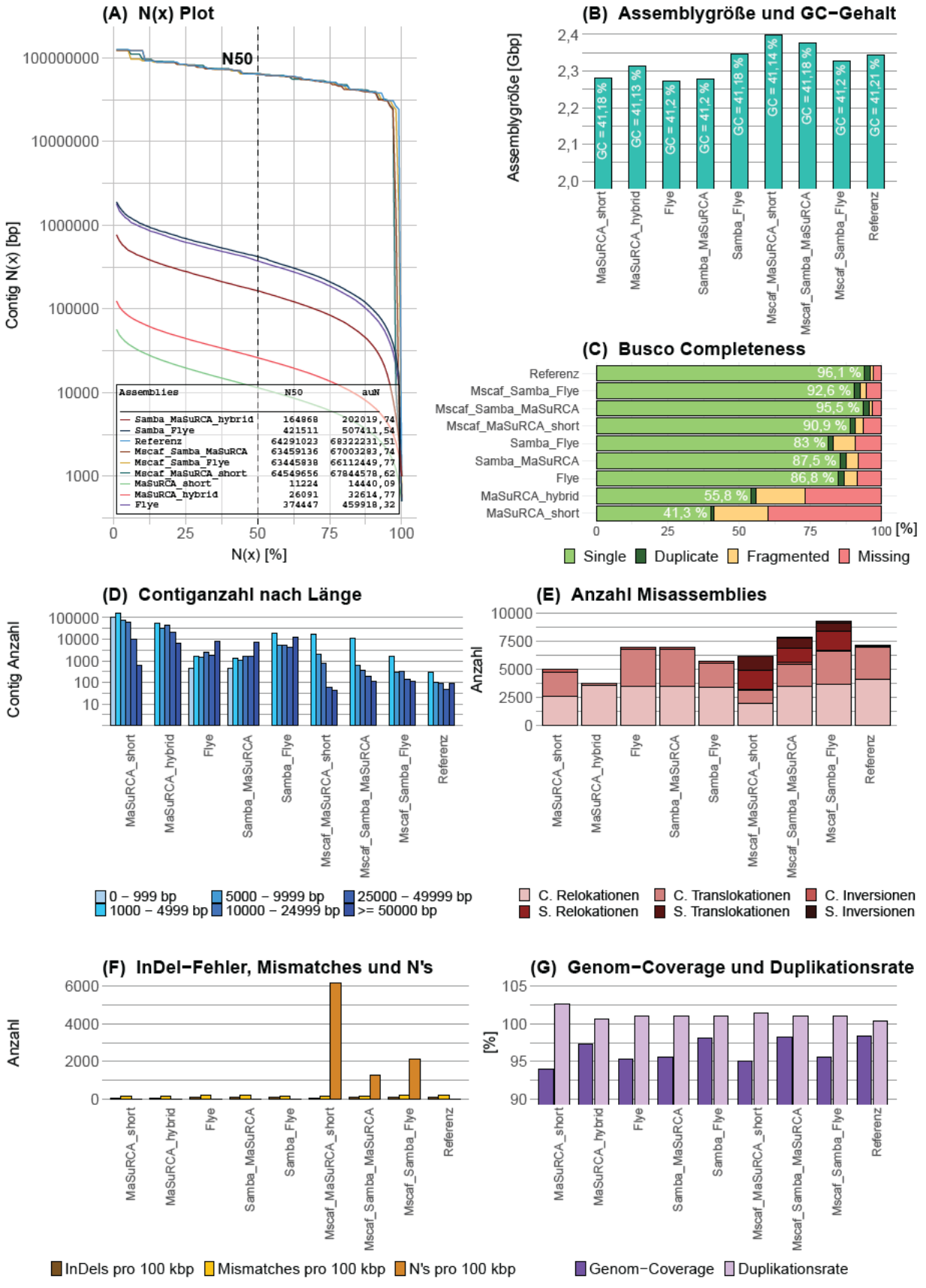

Abbildung 5.20: Qualitäts-Metriken der Hybrid-Assemblies visualisiert

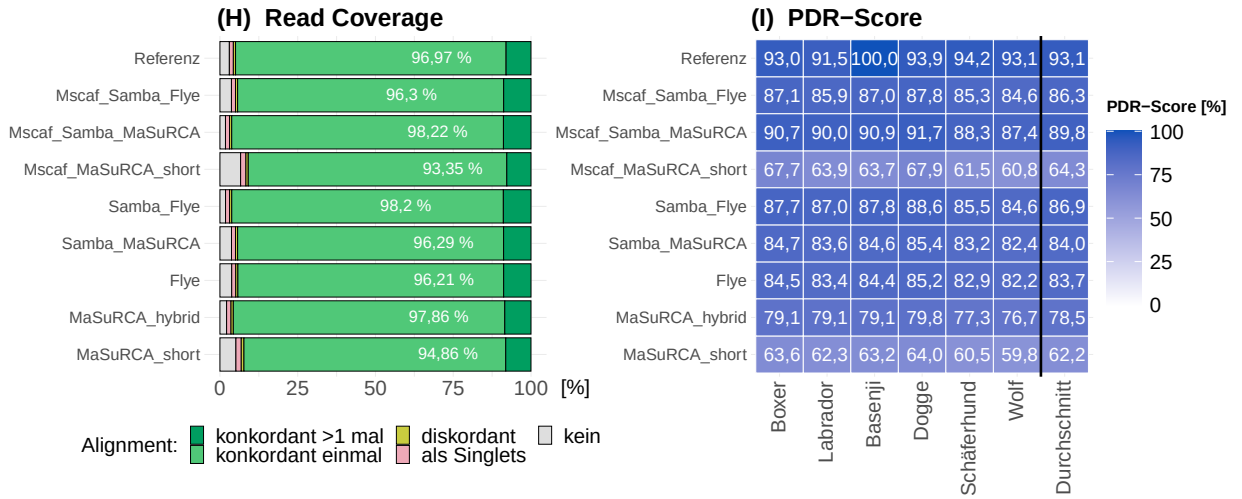

Abbildung 5.21: Qualitäts-Metriken der Hybrid-Assemblies visualisiert (fortgesetzt)

werden die Lücken zwischen den Contigs mit unbestimmten Basen (N's) gefüllt. Die Menge an N's ist beim Short-Read-Assembly mit Abstand am höchsten (6.000 N's pro 100 kbp). Beim Long-Read-Assembly ist die Anzahl mit etwa 2.000 N's pro 100 kbp deutlich geringer. Das Hybrid-Assembly hat mit etwa 1.600 N's pro 100 kbp die geringste Anzahl an unbestimmten Basen. Insgesamt kann gesagt werden, dass das Long-Read- und Samba-Scaffolded-Hybrid-Assembly die besten *de novo* Assemblies erzeugt haben, da sie beide von den Qualitäts-Metriken sehr ähnlich sind. Beim Vergleich mit den Chromosomen-Level-Assemblies der Referenzen wird allerdings deutlich, dass vor allem bei der Contiganzahl und dem N50 noch Verbesserungspotential besteht (vgl. Abb. 5.8).

6 Ausblick

Eine Grundlage für die Untersuchung eines Organismus ist die Annotation dessen Genoms, welche meist direkt in Folge einer erfolgreichen Genom-Assemblierung durchgeführt wird. Die Annotation wurde in dieser Arbeit nicht durchgeführt, da das Schafpudelgenom noch nicht vollständig assembliert wurde. Die Annotation könnte mit der Eukaryotic Genome Annotation Pipeline des NCBI durchgeführt werden. Diese Pipeline ist jedoch nicht frei verfügbar. Die Annotation muss beim NCBI beantragt werden, wenn das Genom bei der International Nucleotide Sequence Database Collaboration (INSDC) (z.B. GenBank) öffentlich verfügbar ist. Erst dann wird das Genom überhaupt in Betracht gezogen annotiert zu werden. Die Richtlinien sind unter URL-15, 2022 zu finden. Alternativ können Open-Source Annotations Pipelines wie BRAKER2 [Brůna et al., 2021] oder Funannotate [URL-16, 2022] verwendet werden. Der Goldstandard für die Annotation ist aber eine manuelle Annotation, da die automatischen Pipelines zwar leistungsfähig sind, aber trotzdem Gene übersehen, falsche Vorhersagen treffen, Gene zusammenfügen oder fragmentieren [McDonnell et al., 2018]. Eine manuelle Annotation ist allerdings extrem zeitintensiv, jedoch notwendig, wenn ein genauer und zuverlässiger Datensatz erstellt werden soll [Ejigu und Jung, 2020]. Eine Arbeitskette für die manuelle Gen-Annotation, welche verwendet werden könnte, wird von McDonnell et al., 2018 beschrieben.

Die Annotation ist aber nur sinnvoll, wenn ein hochwertiges Genom-Assembly vorliegt [Del Dominguez Angel et al., 2018]. Folglich ist es notwendig das Schafpudel-Assembly noch zu verbessern. Wie sich in der Arbeit angedeutet und auch anhand der Assemblies von anderen Hundegenomen gezeigt hat (Edwards et al., 2021; Player et al., 2021; Wang et al., 2021a), führt ein initiales Long-Read-Assembly mit anschließendem Polishing und Scaffolding zu den besten Assemblies. Bisher waren allerdings nicht ausreichend Nanopore-Sequenzdaten vorhanden, um ein gutes initiales Long-Read-Assembly durchzuführen. Der Long-Read-Assembler Flye gibt eine Sequenzing-Depth von 40x an, dass gute Assemblies enstehen (URL-10, 2022; Kolmogorov et al., 2019). MaSuRCA, welches als Hybrid-Assembler verwendet werden kann, gibt eine Sequencing-Depth von 20x an, dass der Hybrid-Ansatz hochwertige Assemblies liefert [Zimin et al., 2013]. Folglich sind weitere Sequenzierläufe mit ONT notwendig, um das Schafpudel-Assembly zu verbessern.

7 Zusammenfassung

Das Genom eines Organismus vorliegen zu haben, erleichtert dessen Untersuchung beträchtlich. Ebenso soll die Assemblierung des Genoms des altdeutschen Schafpudels dazu beitragen genetische bedingte Krankheiten besser zu verstehen. So zum Beispiel der Kryptochismus, welcher an der Hochschule Mittweida von der Arbeitsgruppe von Professor Wünschiers untersucht wird. Das Ziel ist folglich ein qualitativ hochwertiges Assembly des Genoms des altdeutschen Schafpudels zu erzeugen.

Die dafür notwendigen Sequenzdaten wurden zum einen von GENEWIZ Germany GmbH mit dem Illumina NovaSeq 5000 (Illumina-Paired-End-Reads) und von Nils Schön an der Hochschule Mittweida mit dem MinION Mk1c (Nanopore-Reads) erzeugt. Diese wurden mit FastQC und NanoPlot auf ihre Qualität überprüft. Dabei ergab sich eine sehr hohe Qualität der Illumina-Reads mit einem durchschnittlichen Q-Score von etwa 36 und einer Sequencing-Depth von etwa 30x. Die Qualität der Nanopore-Reads ist mit einem durchschnittlichen Q-Score von 15,5 und einer Sequencing-Depth von etwa 7x deutlich geringer.

Um die Qualität der erstellten Assemblies zu überprüfen und um einen guten Vergleich durchführen zu können, wurden zudem acht Genom-Assemblies von anderen Hundearten und ein Assembly des Grauwolfs herangezogen und die Qualitäts-Metriken wie bei den Schafpudel-Assemblies bestimmt.

Es wurden drei Assembly-Ansätze getestet: Short-Read-Assembly, Long-Reads-Assembly und Hybrid-Assembly. Für das Short-Read-Assembly mit den Illumina-Reads wurden vier verschiedene Assembler getestet. Alle vier Short-Read-Assemblies hatten eine starke Fragmentierung, also einen niedrigen N50 und eine hohe Contiganzahl. Zudem war die Busco Completeness und der PDR-Score gering. Mit Ausnahme der Anzahl an Misassemblies erbrachte das MaSuRCA-Assembly die besten Qualitäts-Metriken der Short-Read-Assemblies. Diese liegen aber deutlich unter den Werten der Referenzen. Folglich ist eine Verbesserung dieser Assemblies notwendig. Hierfür wurde ein Assembly-Merging mit GAM-NGS und MAC getestet. Das Merging der initialen Assemblies erbrachte jedoch keine uneingeschränkte Verbesserung der Assembly-Qualität. Das Chromosom-Scaffolding mit dem Chromosome-Scaffolder von MaSuRCA erzeugte Assemblies, die von deutlich höherer Qualität sind. Diese Assemblies sind von der Qualität her vergleichbar mit den Chromosomen-Level-Assemblies der untersuchten Referenzen. Für das Chromosom-Scaffolding wurde eine Referenz verwendet, an welche die Contigs der Assemblies aligniert wurden. Es handelt sich also nicht, um ein *de novo* Assembly sondern, um ein Referenz-basiertes Assembly. Folglich können Eigenschaften der Referenz in das Assembly eingebaut werden, die im Schafpudelgenom nicht vorkommen. Es

ist somit besser ein hochwertiges Assembly zu erzeugen, ohne eine Referenz im Assembly-Prozess zu verwenden. Die Verwendung von Long-Reads ist folglich essentiell.

Das Long-Read-Assembly mit den Nanopore-Reads wurde mit Flye durchgeführt. Hierbei wurde ein deutlich besseres Assembly im Vergleich zu den Short-Read-Assemblies erzeugt. Der N50 konnte um das 20-fache erhöht und die Contiganzahl um 95 % gesenkt werden. Die Busco Completeness konnte um 45 % erhöht werden. Das Scaffolding und Polishing des Long-Read-Assemblies führte jedoch nur zu einer minimalen Verbesserung des Assemblies. Nur das Chromosom-Scaffolding konnte die Qualität des Assemblies deutlich erhöhen.

Das Hybrid-Assembly mit MaSuRCA erzeugte ein deutlich besseres Assembly als das Short-Read-Assembly mit MaSuRCA. Der N50 konnte um das 2,36-fache erhöht und die Contiganzahl um 57 % reduziert werden. Die Busco Completeness konnte um 14,7 % un der PDR-Score um 16,3 % erhöht werden. Jedoch ist dieses Hybrid-Assembly von der Qualität her schlechter als das Long-Read-Assembly. Jedoch konnte hier mit dem Samba-Scaffolding, im Gegensatz zum Long-Read-Assembly, das Assembly noch stark verbessert werden. Nach dem Scaffolding sind das Long-Read- und Hybrid-Assembly von den Qualitäts-Metriken her gesehen sehr ähnlich. Beim N50, der Contiganzahl und der Busco Completeness hat das Long-Read-Assembly bessere Werte. Hingegen ist das Hybrid-Assembly bei der Assemblygröße, der Misassembly-Anzahl, der Anzahl an unbestimmten Basen und dem PDR-Score qualitativ besser.

Eine Verbesserung sowohl des Long-Read-, als auch des Hybrid-Assemblies ist durch zusätzliche Sequenzdaten möglich. Eine Sequencing-Depth der Long-Reads von 40x für das Long-Read-Assembly und von 20x für das Hybrid-Assembly wird von den Assemblern für die Erzeugung eines qualitativ guten Assemblies empfohlen.

Nachdem ein qualitativ hochwertiges Assembly erzeugt wurde, ist eine Annotation dieses Assemblies angedacht. Hierfür ist es beispielsweise möglich die Eukaryotic Genome Annotation Pipeline des NCBI oder Open-Source Annotation-Pipelines wie BRAKER2 oder Funannotate zu verwenden.

8 Summary

Having the genome of an organism available greatly facilitates its study. Likewise, the assembly of the genome of the old German sheep poodle should help to better understand genetic diseases, for instance, cryptochism, which is being studied by Professor Wünschiers' research group at the University of Applied Sciences Mittweida. The aim is to produce a high-quality genome-assembly of the old German sheep poodle.

The necessary sequencing data were generated by GENEWIZ Germany GmbH with the Illumina NovaSeq 5000 (Illumina paired-end-reads) and by Nils Schön at the University of Applied Sciences Mittweida with the MinION Mk1c (Nanopore reads). These were checked for quality using FastQC and NanoPlot. This resulted in a very high quality of the Illumina reads with a mean Q-score of about 36 and a sequencing depth of about 30x. The quality of the Nanopore reads is significantly lower with a mean Q-score of 15.5 and a sequencing depth of about 7x.

To check the quality of the generated assemblies and to make a good comparison, eight genome assemblies from other dog species and one assembly from the graywolf genome were also used and quality metrics were determined as it was done for the sheep poodle assemblies.

Three assembly approaches were tested: short-read-assembly, long-reads-assembly and hybrid-assembly. Four different assemblers were tested for the short-read-assembly with the Illumina reads. All four short-read-assemblies were highly fragmented, i.e., low N50 and a high contig count. In addition, the Busco Completeness and the PDR score were low. With the exception of the number of misassemblies, the MaSuRCA assembly yielded the best quality metrics for the short-read-assemblies. However, these are significantly lower than the scores of the references. Consequently, an improvement of these assemblies is necessary. For this purpose, an assembly-merging with GAM-NGS and MAC was tested. However, merging the initial assemblies did not yield an unrestricted improvement in assembly quality. Chromosome-scaffolding with the Chromosome-scaffolder from MaSuRCA produced assemblies that are of significantly higher quality. These assemblies are comparable in quality to the chromosome-level-assemblies of the references studied. However, for chromosome-scaffolding, a reference was used, to which the contigs of the assemblies were aligned. Thus, it is not a *de novo* assembly but a reference-based assembly. Consequently, properties of the reference can be incorporated into the assembly that do not occur in the sheep poodle genome. Thus, it is better to generate a high-quality assembly without using a reference in the assembly process. Consequently, the use of long reads is essential.

The long read assembly with Nanopore reads was performed with Flye. This produced a significantly better assembly compared to the short-read-assemblies. The N50 was increased by a factor

of 20 and the contig count was reduced by 95 %. The Busco Completeness could be increased by 45 %. However, scaffolding and polishing of the long-read-assembly resulted only in a minimal improvement. Only chromosome-scaffolding could significantly increase the quality of the assembly. The hybrid-assembly with MaSuRCA produced a significantly better assembly than the short-read-assembly with MaSuRCA. The N50 could be reduced by 2.36 times and the contig count by 57 %. The Busco Completeness could be increased by 14.7 % and the PDR score by 16.3 %. However, the quality of this hybrid-assembly is inferior to that of the long-read-assembly. In contrast to the long-read-assembly, the hybrid-assembly could still be greatly improved with the Samba-scaffolding. After scaffolding, the quality metrics of the long-read- and hybrid-assemblies are very similar. The long read assembly has better values for the N50, the contig count and the Busco Completeness. In contrast, the hybrid-assembly is better in terms of assembly size, misassembly count, number of undeterminate bases, and PDR score.

Improvement of both the long-read- and hybrid-assemblies is possible with additional sequencing data. A sequencing depth of the long-reads of 40x for the long-read-assembly and of 20x for the hybrid-assembly is recommended by the assemblers for the generation of a good quality assembly. After a high-quality assembly has been generated, an annotation of this assembly is considered. For this purpose, it is possible to use the Eukaryotic Genome Annotation Pipeline of the NCBI or open-source annotation pipelines such as BRAKER2 or Funannotate.

Anhang

A1 Befehle der Short-Read-Assembler

A1.1 Abyss

```
/usr/local/hdd2/tfritzs3/Tools/abyss-2.2.5/bin/abyss-pe np=30
name=sheep_poodle k=96 B=150G
in='/usr/local/hdd2/tfritzs3/raw_data/sheep_poodle_reads_1.fastq.gz
/usr/local/hdd2/tfritzs3/raw_data/sheep_poodle_reads_2.fastq.gz'
```

A1.2 SOAPdenovo2

```
/usr/local/hdd2/tfritzs3/Tools/SOAPdenovo2-r242/SOAPdenovo-63mer all
-s /usr/local/hdd2/tfritzs3/Tools/SP_v1_soapdenovo.config -p 30
-o 02_SOAPdenovo_k63_fastp_reads -R -K 63
```

SOAPdenovo.config

```
#maximal read length
max_rd_len=150
[LIB]
#average insert size
avg_ins=216.4
#if sequence needs to be reversed
reverse_seq=0
#in which part(s) the reads are used
asm_flags=3
#use only first 100 bps of each read
rd_len_cutoff=150
#in which order the reads are used while scaffolding
rank=1
# cutoff of pair number for a reliable connection (at least 3 for
short insert size)
pair_num_cutoff=3
#minimum aligned length to contigs for a reliable read location (at least
```

```
32 for short insert size)
map_len=32
#a pair of fastq file, read 1 file should always be followed by read 2 file
q1=/usr/local/hdd2/tfritzs3/raw_data/sheep_poodle_reads_1.fastq
q2=/usr/local/hdd2/tfritzs3/raw_data/sheep_poodle_reads_2.fastq
#another pair of fastq file, read 1 file should always be followed by read
2 file
#q1=/path/**LIBNAMEA**/fastq2_read_1.fq
#q2=/path/**LIBNAMEA**/fastq2_read_2.fq
#a pair of fasta file, read 1 file should always be followed by read 2 file
#f1=/path/**LIBNAMEA**/fasta1_read_1.fa
#f2=/path/**LIBNAMEA**/fasta1_read_2.fa
#another pair of fasta file, read 1 file should always be followed by read
2 file
#f1=/path/**LIBNAMEA**/fasta2_read_1.fa
#f2=/path/**LIBNAMEA**/fasta2_read_2.fa
#fastq file for single reads
#q=/path/**LIBNAMEA**/fastq1_read_single.fq
#another fastq file for single reads
#q=/path/**LIBNAMEA**/fastq2_read_single.fq
#fasta file for single reads
#f=/path/**LIBNAMEA**/fasta1_read_single.fa
#another fasta file for single reads
#f=/path/**LIBNAMEA**/fasta2_read_single.fa
#a single fasta file for paired reads
#p=/path/**LIBNAMEA**/pairs1_in_one_file.fa
#another single fasta file for paired reads
#p=/path/**LIBNAMEA**/pairs2_in_one_file.fa
#bam file for single or paired reads, reads 1 in paired reads file should
always be followed by reads 2
# NOTE: If a read in bam file fails platform/vendor quality checks(the flag
field 0x0200 is set), itself and it's paired read would be ignored.
#b=/path/**LIBNAMEA**/reads1_in_file.bam
#another bam file for single or paired reads
```

```
#b=/path/**LIBNAMEA**/reads2_in_file.bam
#[LIB]
#avg_ins=2000
#reverse_seq=1
#asm_flags=2
#rank=2
# cutoff of pair number for a reliable connection (at least 5 for
large insert size)
#pair_num_cutoff=5
#minimum aligned length to contigs for a reliable read location (at least
35 for large insert size)
#map_len=35
#q1=/path/**LIBNAMEB**/fastq_read_1.fq
#q2=/path/**LIBNAMEB**/fastq_read_2.fq
#f1=/path/**LIBNAMEA**/fasta_read_1.fa
#f2=/path/**LIBNAMEA**/fasta_read_2.fa
#p=/path/**LIBNAMEA**/pairs_in_one_file.fa
#b=/path/**LIBNAMEA**/reads_in_file.bam
```

A1.3 MaSURCA

```
cd /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/03_MaSuRCA_assembly

/usr/local/hdd2/tfritzs3/Tools/MaSuRCA-4.0.8/bin/masurca
/usr/local/hdd2/tfritzs3/Tools/MaSuRCA-4.0.8/Config_hybrid_all_reads_mixed.txt

./assemble.sh
```

Config_hybrid_all_reads_mixed.txt

```
# example configuration file
# DATA is specified as type PE,JUMP,OTHER,PACBIO and 5 fields:
# 1)two_letter_prefix 2)mean 3)stdev 4)fastq(.gz)_fwd_reads
# 5)fastq(.gz)_rev_reads. The PE reads are always assumed to be
# innies, i.e. --->.<---, and JUMP are assumed to be outties
# <---.--->. If there are any jump libraries that are innies, such as
```

```
# longjump, specify them as JUMP and specify NEGATIVE mean. Reverse reads
# are optional for PE libraries and mandatory for JUMP libraries. Any
# OTHER sequence data (454, Sanger, Ion torrent, etc) must be first
# converted into Celera Assembler compatible .frg files (see
# http://wgs-assembler.sourceforge.com)
DATA
PE= SP 216.4 41.7
/usr/local/hdd2/tfritzs3/raw_data/sheep_poodle_reads_1.fastq.gz
/usr/local/hdd2/tfritzs3/raw_data/sheep_poodle_reads_2.fastq.gz
#Illumina paired end reads supplied as <two-character prefix>
<fragment mean> <fragment stdev> <forward_reads> <reverse_reads>
#if single-end, do not specify <reverse_reads>
#If mean/stdev are unknown use 500 and 50 -- these are safe values
that will work for most runs
#MUST HAVE Illumina paired end reads to use MaSuRCA
#Illumina mate pair reads supplied as <two-character prefix>
<fragment mean> <fragment stdev> <forward_reads> <reverse_reads>
#JUMP= sh 3600 200 /FULL_PATH/short_1.fastq /FULL_PATH/short_2.fastq
#pacbio OR nanopore reads must be in a single fasta or fastq file with
absolute path, can be gzipped
#if you have both types of reads supply them both as NANOPORE type
#PACBIO=/FULL_PATH/pacbio.fa
NANOPORE=/usr/local/hdd2/tfritzs3/raw_data/Nanopore_Schafpudel/sup_reads/
poodle55_all_sup_reads.fastq
#Legacy reads (Sanger, 454, etc) in one frg file, concatenate your
frg files into one if you have many
#OTHER=/FULL_PATH/file.frg
#synteny-assisted assembly, concatenate all reference genomes into one
reference.fa; works for Illumina-only data
#REFERENCE=/FULL_PATH/nanopore.fa
END
PARAMETERS
#PLEASE READ all comments to essential parameters below, and set the
parameters according to your project
```

```
#set this to 1 if your Illumina mate pair (jumping) library reads are
shorter than 100bp
EXTEND_JUMP_READS=0
#this is k-mer size for deBruijn graph values between 25 and 127 are
supported, auto will compute the optimal size based on the read data and
GC content
GRAPH_KMER_SIZE = auto
#set this to 1 for all Illumina-only assemblies
#set this to 0 if you have more than 15x coverage by long reads
(Pacbio or Nanopore) or any other long reads/mate pairs
(Illumina MP, Sanger, 454, etc)
USE_LINKING_MATES = 0
#specifies whether to run the assembly on the grid
USE_GRID=0
#specifies grid engine to use SGE or SLURM
GRID_ENGINE=SGE
#specifies queue (for SGE) or partition (for SLURM) to use when
running on the grid MANDATORY
GRID_QUEUE=all.q
#batch size in the amount of long read sequence for each batch on the grid
GRID_BATCH_SIZE=500000000
#use at most this much coverage by the longest Pacbio or Nanopore reads,
discard the rest of the reads
#can increase this to 30 or 35 if your long reads reads have N50<7000bp
LHE_COVERAGE=25
#this parameter is useful if you have too many Illumina jumping library
reads. Typically set it to 60 for bacteria and 300 for the other organisms
LIMIT_JUMP_COVERAGE = 300
#these are the additional parameters to Celera Assembler; do not worry
about performance, number or processors or batch sizes
-- these are computed automatically.
#CABOG ASSEMBLY ONLY: set cgwErrorRate=0.25 for bacteria and
0.1<=cgwErrorRate<=0.15 for other organisms.
CA_PARAMETERS = cgwErrorRate=0.15
```

```
#CABOG ASSEMBLY ONLY: whether to attempt to close gaps in
scaffolds with Illumina or long read data
CLOSE_GAPS=1
#number of cpus to use, set this to the number of CPUs/threads per
node you will be using
NUM_THREADS = 32
#this is mandatory jellyfish hash size -- a safe value is
estimated_genome_size*20
JF_SIZE = 24000000000
#ILLUMINA ONLY. Set this to 1 to use SOAPdenovo contigging/scaffolding
module.
#Assembly will be worse but will run faster. Useful for very large
(>=8Gbp) genomes from Illumina-only data
SOAP_ASSEMBLY=0
#If you are doing Hybrid Illumina paired end + Nanopore/PacBio assembly
ONLY (no Illumina mate pairs or OTHER frg files).
#Set this to 1 to use Flye assembler for final assembly of corrected
mega-reads.
#A lot faster than CABOG, AND QUALITY IS THE SAME OR BETTER.
#DO NOT use if you have less than 20x coverage by long reads.
FLYE_ASSEMBLY=0
END
```

A1.4 SparseAssembler

```
/usr/local/hdd2/tfritzs3/Tools/SparseAssembler/SparseAssebmler
g 15 k 53 LD 0 GS 2400000000 f /usr/local/hdd2/tfritzs3/raw_data/
sheep_poodle_reads_1.fastq f /usr/local/hdd2/tfritzs3/raw_data/
sheep_poodle_reads_2.fastq NodeCovTh 1 EdgeCovTh 0
```

A2 Befehle der Assembly-Merger

A2.1 GAM-NGS

```
bwa index -p bwa_[01/02/03/GAM_01_02]_assembly -a bwtsw
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/[01/02/03/GAM_01_02]
_reads[01/02/03/GAM_01_02]_assembly/assembly.fasta

bwa mem -P -t 15 /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]_assembly/bwa_mapping/
bwa_[01/02/03/GAM_01_02]_assembly /usr/local/hdd2/tfritzs3/raw_data/
Processed_data/fastp/sp_reads_1.fastq > /usr/local/hdd2/tfritzs3/
Schafpudel/Genome_Assembly/[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]
_assembly/bwa_mapping/bwa_mapping_pe1_[01/02/03/GAM_01_02].sam

bwa mem -P -t 15 /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]_assembly/bwa_mapping/
bwa_[01/02/03/GAM_01_02]_assembly /usr/local/hdd2/tfritzs3/raw_data/
Processed_data/fastp/sp_reads_2.fastq > /usr/local/hdd2/tfritzs3/
Schafpudel/Genome_Assembly/[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]
_assembly/bwa_mapping/bwa_mapping_pe2_[01/02/03/GAM_01_02].sam

samtools view -bS -@14 /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]_assembly/bwa_mapping/
bwa_mapping_pe1_[01/02/03/GAM_01_02].sam > /usr/local/hdd2/tfritzs3/
Schafpudel/Genome_Assembly/[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]
_assembly/bwa_mapping/bwa_mapping_pe1_[01/02/03/GAM_01_02].bam

samtools view -bS -@14 /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]_assembly/bwa_mapping/
bwa_mapping_pe2_[01/02/03/GAM_01_02].sam > /usr/local/hdd2/tfritzs3/
Schafpudel/Genome_Assembly/[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]
_assembly/bwa_mapping/bwa_mapping_pe2_[01/02/03/GAM_01_02].bam

samtools sort -@14 /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
```

```
01_reads[01/02/03/GAM_01_02]_assembly/bwa_mapping/
bwa_mapping_pe1_[01/02/03/GAM_01_02].bam > /usr/local/hdd2/tfritzs3/
Schafpudel/Genome_Assembly/[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]
_assembly/bwa_mapping/bwa_mapping_pe1_[01/02/03/GAM_01_02].sorted.bam

samtools sort -@14 /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]_assembly/bwa_mapping/
bwa_mapping_pe2_[01/02/03/GAM_01_02].bam > /usr/local/hdd2/tfritzs3/
Schafpudel/Genome_Assembly/[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]
_assembly/bwa_mapping/bwa_mapping_pe2_[01/02/03/GAM_01_02].sorted.bam

samtools index /usr/local/hdd2/tfritzs3/Schafpudel/GenomevAssembly/
[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]_assembly/bwa_mapping/
bwa_mapping_pe1v[01/02/03/GAM_01_02].sorted.bam

samtools index /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
[01/02/03/GAM_01_02]_reads[01/02/03/GAM_01_02]_assembly/bwa_mapping/
bwa_mapping_pe2_[01/02/03/GAM_01_02]_.sorted.bam

/usr/local/hdd2/tfritzs3/Tools/gam-ngs/bin/gam-create --master-bam
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/02_SOAPdenovo_k63
_fastp_reads/ bwa_mapping_Contigs/gam-ngs_02_SOAP_k63_master.PE.bams.txt
--slave-bam /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
01_abyss_reads01_assembly/bwa_mapping/gam-ngs_01_abyss_master.PE.bams.txt
--min-block-size 50 --output /usr/local/hdd2/tfritzs3/Schafpudel/
Genome_Assembly/GAM-NGS_merge_01_02

/usr/local/hdd2/tfritzs3/Tools/gam-ngs/bin/gam-merge --master-bam
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/02_SOAP
_k63_fastp_reads/ bwa_mapping_Contigs/gam-ngs_02_SOAP_k63
_master.PE.bams.txt --slave-bam /usr/local/hdd2/tfritzs3/
Schafpudel/Genome_Assembly/01_abyss_reads01_assembly/bwa_mapping/
gam-ngs_01_abyss_master.PE.bams.txt --master-fasta
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/02_SOAP_k63
```

```
_fastp_reads/02_SOAPdenovo_k63_fastp_reads.contig --slave-fasta
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/01_abyss_reads01
_assembly/ sheep_poodle-6.fa --threads 30 --blocks-file
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/GAM-NGS_merge_01_02/
GAM-NGS_merge_01_02.blocks --output
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/GAM-NGS_merge_01_02/
GAM-NGS_merge_01_02

/usr/local/hdd2/tfritzs3/Tools/gam-ngs/bin/gam-create --master-bam
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/GAM_01_02
_fastp_reads/ bwa_mapping_Contigs/gam-ngs_GAM_01_02_master.PE.bams.txt
--slave-bam /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
03_MASuRCA_reads03_assembly/bwa_mapping/gam-ngs_03_MaSuRCA
_master.PE.bams.txt --min-block-size 50 --output /usr/local/hdd2/
tfritzs3/Schafpudel/Genome_Assembly/GAM-NGS_merge_01_02_03

/usr/local/hdd2/tfritzs3/Tools/gam-ngs/bin/gam-merge --master-bam
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/GAM_01_02
_fastp_reads/ bwa_mapping_Contigs/gam-ngs_GAM_01_02
_master.PE.bams.txt --slave-bam /usr/local/hdd2/tfritzs3/
Schafpudel/Genome_Assembly/03_MaSuRCA_reads03_assembly/bwa_mapping/
gam-ngs_03_MaSuRCA_master.PE.bams.txt --master-fasta
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/GAM_01_02
_fastp_readsGAM_01_02_fastp_reads.contig --slave-fasta
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/03_MaSuRCA_reads03
_assembly/primary_assembly_scf.fasta --threads 30 --blocks-file
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
GAM-NGS_merge_01_02_03/ GAM-NGS_merge_01_02_03.blocks
--output /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
GAM-NGS_merge_01_02_03
```

gam-ngs_01_abyss_master.PE.bams.txt

```
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/01_abyss_reads01
_assembly/bwa_mapping/bwa_mapping_pe1_01_abyss.sorted.bam
2 5947
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/01_abyss_reads01
_assembly/bwa_mapping/bwa_mapping_pe2_01_abyss.sorted.bam
2 5947
```

gam-ngs_02_SOAP_k63_master.PE.bams.txt

```
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/02_SOAPdenovo_k63
_fastp_reads/bwa_mapping_Contigs/bwa_mapping_pe1_02_SOAP_k63.sorted.bam
2 3974
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/02_SOAPdenovo_k63
_fastp_reads/bwa_mapping_Contigs/bwa_mapping_pe2_02_SOAP_k63.sorted.bam
2 3974
```

gam-ngs_03_MaSuRCA_master.PE.bams.txt

```
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/03_MaSuRCA
_fastp_reads/bwa_mapping_Contigs/bwa_mapping_pe1_03_MaSuRCA.sorted.bam
2 22927
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/03_MaSuRCA_fastp
_reads/bwa_mapping_Contigs/bwa_mapping_pe2_03_MaSuRCA.sorted.bam
2 22927
```

gam-ngs_GAM_01_02_master.PE.bams.txt

```
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/03_MaSuRCA
_fastp_reads/bwa_mapping_Contigs/bwa_mapping_pe1_GAM_01_02.sorted.bam
2 22927
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/03_MaSuRCA_fastp
_reads/bwa_mapping_Contigs/bwa_mapping_pe2_GAM_01_02.sorted.bam
2 22927
```

A2.2 MAC

```
export PATH=$PATH:/usr/local/hdd2/tfritzs3/Tools/mummer-4.0.0rc1

cd /usr/local/hdd2/tfritzs3/Tools/MAC

./MAC2.0 01_abyss.fasta 02_SOAPdenovo.fa
./MAC2.0 MAC_01_02.fa 03_MaSuRCA.fa
```

A3 Befehle der Assembly-Scaffolder

A3.1 Chromosome_Scaffolder

```
/usr/local/hdd2/tfritzs3/Tools/MaSuRCA-4.0.9/bin/chromosome_scaffolder.sh
-r /usr/local/hdd2/tfritzs3/Schafpudel/Referenzen/Tasha/tasha_genome.fasta
-q /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/03_MaSuRCA_assembly/
CA/primary.genome.scf.fasta -t 32 -nb
```

A3.2 Samba_Scaffolder

```
/usr/local/hdd2/tfritzs3/Tools/MaSuRCA-4.0.9/bin/samba.sh -r
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/Flye/all_sup_reads
_fmlrc_corr/assembly.fasta -t 10 -q /usr/local/hdd2/tfritzs3/raw_data/
Nanopore_Schafpudel/sup_reads/poodle55_all_sup_reads.fastq
```

A4 Befehle der Long-Read-Assembler

A4.1 fmlrc

```
cat /usr/local/hdd2/tfritzs3/raw_data/sheep_poodle_reads_1.fastq
/usr/local/hdd2/tfritzs3/raw_data/sheep_poodle_reads_2.fastq |
awk 'NR % 4 == 2' | sort | tr NT TN | /usr/local/hdd2/tfritzs3/Tools/
ropebwt2/ropebwt2 -LR | tr NT TN | /usr/local/hdd2/tfritzs3/Tools/fmlrc/
fmlrc-convert -f comp_msbwt.npy


/usr/local/hdd2/tfritzs3/Tools/fmlrc/fmlrc comp_msbwt.npy
/usr/local/hdd2/tfritzs3/raw_data/Nanopore_Schafpudel/sup_reads/
poodle55_all_sup_reads.fastq poodle55_all_sup_reads_Illumina
_reads_fmlrc_corr.fasta -p 30
```

A4.2 Flye

```
/usr/local/hdd2/tfritzs3/Tools/Flye/bin/flye
--nano-raw /usr/local/hdd2/tfritzs3/raw_reads/reads.fq
--genome-size 2.4G --threads 25
-o /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/Fyle
```

A5 Befehle der Assembly-Polisher

A5.1 Pilon-Polishing

```
samtools index mapping.sorted.bam


java -Xmx200G -jar /usr/local/hdd2/tfritzs3/Tools/pilon-1.24.jar
-genome assembly.fasta -fix all -changes -frags mapping.sorted.bam -outdir
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/Pilon_Polish/Flye_pilon1x
-output Flye_pilon1x
```

A5.2 Polca-Polishing

```
/usr/local/hdd2/tfritzs3/Tools/MaSuRCA-4.0.9/bin/polca.sh -a
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/Flye/all_sup_reads
_fmlrc_corr/assembly.fasta -t 10 -r '/home5/share/hund/ngs_reads/
RW01/delivery/RW01_55-1/55-1_S5_R1_001.fastq.gz /home5/share/hund/ngs_reads/
RW01/delivery/RW01_55-1/55-1_S5_R2_001.fastq.gz'
```

A6 Befehle der Qualitätskontrolle der Assemblies

A6.1 Quast

```
/usr/local/hdd2/tfritzs3/Tools/quast/quast.py /usr/local/hdd2/tfritzs3/
Schafpudel/Genome_Assembly/03_MaSuRCA_assembly/CA/9-terminator/
primary.genome.scf.fasta -r /usr/local/hdd2/tfritzs3/Schafpudel/
Referenzen/Tasha/tasha_genome.fasta -1 /usr/local/hdd2/tfritzs3/
raw_data/sheep_poodle_reads_1.fastq.gz -2 /usr/local/hdd2/tfritzs3/
raw_data/sheep_poodle_reads_2.fastq.gz -o /usr/local/hdd2/tfritzs3/
Schafpudel/QC/quast/03_MaSuRCA_Ref_Tasha -t 30
```

A6.2 Busco

```
/home/tfritzs3/.local/bin/busco --config  /Forschungsprojekt/Config_Files/
busco_config.ini
```

busco_config.ini

```
# This is the BUSCOv5 configuration file template.
# It is not necessary to use this, as BUSCO will use the dependencies
available on your PATH by default.
# The busco run parameters can all be set on the command line.
See the help prompt (busco -h) for details.
#
# To use this file for an alternative configuration, or to specify
particular versions of dependencies:
# 1) edit the path and command values to match your desired dependency
versions.
# WARNING: passing a parameter through the command line overrides the
value specified in this file.
#
# 2) Enable a parameter by removing ";"
#
# 3) Make this config file available to BUSCO either by setting an
environment variable
#
# export BUSCO_CONFIG_FILE=/path/to/myconfig.ini"
#
# or by passing it as a command line argument
#
# busco <args> --config /path/to/config.ini
#
[busco_run]
# Input file
in = /usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/
03_MaSuRCA_assembly/CA/primary.genome.scf.fasta
# Run name, used in output files and folder
```

```
out = BUSCO_05_MaSuRCA_primary
# Where to store the output directory
out_path = /usr/local/hdd2/tfritzs3/Schafpudel/QC/BUSCO
# Path to the BUSCO dataset
lineage_dataset = laurasiatheria_odb10
# Which mode to run (genome / proteins / transcriptome)
mode = genome
# Run lineage auto selector
;auto-lineage = True
# Run auto selector only for non-eukaryote datasets
;auto-lineage-prok = True
# Run auto selector only for eukaryote datasets
;auto-lineage-euk = True
# How many threads to use for multithreaded steps
cpu = 30
# Force rewrite if files already exist (True/False)
force = True
# Restart a previous BUSCO run (True/False)
;restart = False
# Blast e-value
;evalue = 1e-3
# How many candidate regions (contigs, scaffolds) to consider for
each BUSCO
;limit = 3
# Metaeuk parameters for initial run
;metaeuk_parameters='--param1=value1,--param2=value2'
# Metaeuk parameters for rerun
;metaeuk_rerun_parameters=
# Augustus parameters
;augustus_parameters='--param1=value1,--param2=value2'
# Quiet mode (True/False)
;quiet = False
# Local destination path for downloaded lineage datasets
download_path = /usr/local/hdd2/tfritzs3/raw_data/busco_downloads
```

```
# Run offline
offline = True
# Ortho DB Datasets version
;datasets_version = odb10
# URL to BUSCO datasets
;download_base_url = https://busco-data.ezlab.org/v4/data/
# Download most recent BUSCO data and files
;update-data = True
# Use Augustus gene predictor instead of metaeuk
;use_augustus = True [tblastn]
path = /home/tfritzs3/Forschungsprojekt/Tools/ncbi-blast-2.12.0+/bin/
command = tblastn


[makeblastdb]
path = /home/tfritzs3/Forschungsprojekt/Tools/ncbi-blast-2.12.0+/bin/
command = makeblastdb

[metaeuk]
path = /home/tfritzs3/Forschungsprojekt/Tools/metaeuk/bin/
command = metaeuk

[augustus]
path = /home/tfritzs3/Forschungsprojekt/Tools/augustus.2.5.5/bin/
command = augustus

[etraining]
path = /home/tfritzs3/Forschungsprojekt/Tools/augustus.2.5.5/bin/
command = etraining

[gff2gbSmallDNA.pl]
path = /home/tfritzs3/Forschungsprojekt/Tools/augustus.2.5.5/scripts/
command = gff2gbSmallDNA.pl
```

```
[new_species.pl]
path = /home/tfritzs3/Forschungsprojekt/Tools/augustus.2.5.5/scripts/
command = new_species.pl

[optimize_augustus.pl]
path = /home/tfritzs3/Forschungsprojekt/Tools/augustus.2.5.5/scripts/
command = optimize_augustus.pl

[hmmsearch]
path = /home/tfritzs3/Forschungsprojekt/Tools/hmmer-3.3.2/src/
command = hmmsearch

[sepp]
path = /home/tfritzs3/Forschungsprojekt/Tools/sepp/bin/
command = run_sepp.py

[prodigal]
path = /home/tfritzs3/Forschungsprojekt/Tools/Prodigal/
command = prodigal
```

A6.3 PDR

```
java -jar  /Forschungsprojekt/Tools/PDRi.jar /usr/local/hdd2/
tfritzs3/Schafpudel/Referenzen/Tasha/GCA_000002285.4_Dog10K
_Boxer_Tasha/GCA_000002285.4_Dog10K_Boxer_Tasha_genomic.fna.gz
/usr/local/hdd2/tfritzs3/Schafpudel/Genome_Assembly/03_MaSuRCA/CA/
primary.genome.scf.fasta --threads 10 > /usr/local/hdd2/
tfritzs3/Schafpudel/QC/PDR/03_MaSuRCA_Ref_Tasha.txt
```

A6.4 Bowtie2

```
bowtie2-build assembly.fasta assembly.fasta

bowtie2 -q -p 40 -k 20 -x assembly.fasta -1 /home5/share/hund/ngs_reads/
RW01/delivery/RW01_55-1/55-1_S5_R1_001.fastq.gz -2 /home5/share/hund/
ngs_reads/RW01/delivery/RW01_55-1/55-1_S5_R2_001.fastq.gz | samtools view
-@39
-Sb | samtools sort -o mapping.sorted.bam
```

A7 Skripte

Trinity_stats.pl

```
#!/usr/bin/env perl
use strict;
use warnings;
use FindBin;
use lib ("$FindBin::RealBin/../PerlLib");
use Fasta_reader;
use BHStats;
my $usage = "\n\nusage: $0 transcripts.fasta\n\n";
my $fasta_file = $ARGV[0] or die $usage;
main: {
my $fasta_reader = new Fasta_reader($fasta_file);
my @all_seq_lengths;
my $number_transcripts = 0;
my $num_GC = 0;
my %component_to_longest_isoform;
my $tot_seq_len = 0;
my $missing_gene_ids_flag = 0;
while (my $seq_obj = $fasta_reader->next()) {
my $acc = $seq_obj->get_accession();
$number_transcripts++;
my $comp_id = $acc;
if (! $missing_gene_ids_flag) {
if ($acc =~ /^(.*c\d+_g\d+)/) {
```

```
$comp_id = $1;
}
elsif ($acc =~ /^(.*comp\d+_c\d+)/) {
$comp_id = $1;
}
else {
print STDERR "Error, cannot decipher gene identifier from acc: $acc";
$missing_gene_ids_flag = 1;
}
}
my $sequence = $seq_obj->get_sequence();
my $seq_len = length($sequence);
$tot_seq_len += $seq_len;
if ( (! exists $component_to_longest_isoform{$comp_id})
||
$component_to_longest_isoform{$comp_id} < $seq_len) {
$component_to_longest_isoform{$comp_id} = $seq_len;
}
push (@all_seq_lengths, $seq_len);
while ($sequence =~ /[gc]/ig) {
$num_GC++;
}
}
&report_stats(@all_seq_lengths);
print "";
if ($missing_gene_ids_flag) {
print "";
}
else {
print "#######################################################\n";
print "## Stats based on ONLY LONGEST ISOFORM per 'GENE':\n";
print "#######################################################\n\n";
&report_stats(values %component_to_longest_isoform);
print "\n\n\n";
```

```
}
exit(0);
}
####
sub report_stats {
my (@seq_lengths) = @_;
@seq_lengths = reverse sort {$a<=>$b} @seq_lengths;
my $cum_seq_len = 0;
foreach my $len (@seq_lengths) {
$cum_seq_len += $len;
}
for (my $i = 1; $i <= 100; $i += 1) {
my $cum_len_needed = $cum_seq_len * $i/100;
my $N_val = &get_contigNvalue($cum_len_needed, \@seq_lengths);
print "$N_val\n";
}
return;
}
sub get_contigNvalue {
my ($cum_len_needed, $seq_lengths_aref) = @_;
my $partial_sum_len = 0;
foreach my $len (@$seq_lengths_aref) {
$partial_sum_len += $len;
if ($partial_sum_len >= $cum_len_needed) {
return($len);}}
```

Literaturverzeichnis

Amarasinghe, Shanika L., Shian Su, Xueyi Dong, Luke Zappia, Matthew E. Ritchie und Quentin Gouil (2020). „Opportunities and challenges in long-read sequencing data analysis“. In: *Genome Biology* 21.1, S. 30. ISSN: 1474-760X. DOI: 10.1186/s13059-020-1935-5. URL: https://genomebiology.biomedcentral.com/articles/10.1186/s13059-020-1935-5.

Andrews, Simon, Felix Krueger, Anne Segonds-Pichon, Laura Biggins, Christel Krueger und Steven Wingett (Jan. 2012). *FastQC*. Babraham Institute. Babraham, UK.

Avery, O. T., C. M. Macleod und M. McCarty (1944). „STUDIES ON THE CHEMICAL NATURE OF THE SUBSTANCE INDUCING TRANSFORMATION OF PNEUMOCOCCAL TYPES : INDUCTION OF TRANSFORMATION BY A DESOXYRIBONUCLEIC ACID FRACTION ISOLATED FROM PNEUMOCOCCUS TYPE III“. In: *Journal of Experimental Medicine* 79.2, S. 137–158. ISSN: 0022-1007. DOI: 10.1084/jem.79.2.137. URL: https://rupress.org/jem/article/79/2/137/4753/STUDIES-ON-THE-CHEMICAL-NATURE-OF-THE-SUBSTANCE.

Axelsson, Erik, Abhirami Ratnakumar, Maja-Louise Arendt, Khurram Maqbool, Matthew T. Webster, Michele Perloski, Olof Liberg, Jon M. Arnemo, Ake Hedhammar und Kerstin Lindblad-Toh (2013). „The genomic signature of dog domestication reveals adaptation to a starch-rich diet“. In: *Nature* 495.7441, S. 360–364. ISSN: 1476-4687. DOI: 10.1038/nature11837. URL: https://www.nature.com/articles/nature11837.

Balter, Michael (2005). „The Seeds of Civilization“. In: *Smithsonian Magazine*. URL: https://www.smithsonianmag.com/history/the-seeds-of-civilization-78015429/.

Bekoff, M. (2018). „Dumping the Dog Domestication Dump Theory Once and For All“. In: *Psychology Today*. URL: https://www.psychologytoday.com/au/blog/animal-emotions/201811/dumping-the-dog-domestication-dump-theory-once-and-all.

Benecke, Norbert (1987). „Studies on early dog remains from Northern Europe“. In: *Journal of Archaeological Science* 14.1, S. 31–49. ISSN: 0305-4403. DOI: 10.1016/S0305-4403(87)80004-3. URL: https://www.sciencedirect.com/science/article/pii/S0305440387800043.

Botigué, Laura R. et al. (2017). „Ancient European dog genomes reveal continuity since the Early Neolithic“. In: *Nature Communications* 8, S. 16082. ISSN: 2041-1723. DOI: 10.1038/ncomms16082. URL: https://pubmed.ncbi.nlm.nih.gov/28719574/.

Bradnam, Keith R. et al. (2013). „Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species“. In: *GigaScience* 2.1, S. 10. DOI: 10.1186/2047-217X-2-10. URL: https://academic.oup.com/gigascience/article/2/1/2047-217X-2-10/2656129.

Brůna, Tomáš, Katharina J. Hoff, Alexandre Lomsadze, Mario Stanke und Mark Borodovsky (2021). „BRAKER2: automatic eukaryotic genome annotation with GeneMark-EP+ and AUGUSTUS supported by a protein database“. In: *NAR Genomics and Bioinformatics* 3.1, lqaa108. DOI: 10.1093/nargab/lqaa108. URL: https://academic.oup.com/nargab/article/3/1/lqaa108/6066535.

Bushnell, Brian (2014). „BBMap: A Fast, Accurate, Splice-Aware Aligner“. In: *California Digital Library*. URL: https://escholarship.org/uc/item/1h3515gn.

Cacho, Ashley, Ekaterina Smirnova, Snehalata Huzurbazar und Xinping Cui (2016). „A Comparison of Base-calling Algorithms for Illumina Sequencing Technology“. In: *Briefings in Bioinformatics* 17.5, S. 786–795. ISSN: 1467-5463. DOI: 10.1093/bib/bbv088. URL: https://academic.oup.com/bib/article/17/5/786/2262186.

Chapman, Jarrod A., Isaac Ho, Sirisha Sunkara, Shujun Luo, Gary P. Schroth und Daniel S. Rokhsar (2011). „Meraculous: De Novo Genome Assembly with Short Paired-End Reads“. In: *PLOS ONE* 6.8, e23501. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0023501. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0023501.

Compeau, Phillip E. C., Pavel A. Pevzner und Glenn Tesler (2011). „How to apply de Bruijn graphs to genome assembly“. In: *Nature biotechnology* 29.11, S. 987–991. ISSN: 1087-0156. DOI: 10.1038/nbt.2023.

Coppinger, Raymond (2001). *Dogs: A startling new understanding of canine origin, behavior, and evolution*. New York: Scribner. ISBN: 9780684855301.

Coster, Wouter de, Svenn D'Hert, Darrin T. Schultz, Marc Cruts und Christine van Broeckhoven (2018). „NanoPack: visualizing and processing long-read sequencing data“. In: *Bioinformatics* 34.15, S. 2666–2669. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bty149. URL: https://academic.oup.com/bioinformatics/article/34/15/2666/4934939.

Deamer, David, Mark Akeson und Daniel Branton (2016). „Three decades of nanopore sequencing“. In: *Nature Biotechnology* 34.5, S. 518–524. ISSN: 1546-1696. DOI: 10.1038/nbt.3423. URL: https://www.nature.com/articles/nbt.3423.

Del Dominguez Angel, Victoria et al. (2018). „Ten steps to get started in Genome Assembly and Annotation“. In: *F1000Research* 7. ISSN: 2046-1402. DOI: 10.12688/f1000research.13598.1. URL: https://pubmed.ncbi.nlm.nih.gov/29568489/.

Delahaye, Clara und Jacques Nicolas (2021). „Sequencing DNA with nanopores: Troubles and biases“. In: *PLOS ONE* 16.10, e0257521. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0257521. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0257521.

Earl, Dent et al. (2011). „Assemblathon 1: a competitive assessment of de novo short read assembly methods“. In: *Genome Research* 21.12, S. 2224–2241. ISSN: 1549-5469. DOI: 10.1101/gr.126599.111. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3227110/.

Edwards, Richard J. et al. (2021). „Chromosome-length genome assembly and structural variations of the primal Basenji dog (Canis lupus familiaris) genome“. In: *BMC Genomics* 22.1, S. 188. ISSN: 1471-2164. DOI: 10.1186/s12864-021-07493-6. URL: https://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-021-07493-6.

Ejigu, Girum Fitihamlak und Jaehee Jung (2020). „Review on the Computational Genome Annotation of Sequences Obtained by Next-Generation Sequencing“. In: *Biology* 9.9. DOI: 10.3390/biology9090295.

Grabherr, Manfred G. et al. (2011). „Full-length transcriptome assembly from RNA-Seq data without a reference genome“. In: *Nature Biotechnology* 29.7, S. 644–652. ISSN: 1546-1696. DOI: 10.1038/nbt.1883.

Gurevich, Alexey, Vladislav Saveliev, Nikolay Vyahhi und Glenn Tesler (2013). „QUAST: quality assessment tool for genome assemblies“. In: *Bioinformatics (Oxford, England)* 29.8, S. 1072–1075. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btt086. URL: https://pubmed.ncbi.nlm.nih.gov/23422339/.

Hare, Brian und Vanessa Woods (2013). „Opinion: We Didn’t Domesticate Dogs. They Domesticated Us“. In: *National Geographic*. URL: https://www.nationalgeographic.com/animals/article/130302-dog-domestic-evolution-science-wolf-wolves-human.

Heydari, Mahdi, Giles Miclotte, Piet Demeester, Yves van de Peer und Jan Fostier (2017). „Evaluation of the impact of Illumina error correction tools on de novo genome assembly“. In: *BMC Bioinformatics* 18.1, S. 374. ISSN: 1471-2105. DOI: 10.1186/s12859-017-1784-8. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-017-1784-8.

Holly, R. W., J. APGAR, G. A. EVERETT, J. T. MADISON, M. Marquisse, S. H. Merill, J. R. PENSWICK und A. ZAMIR (1965). „Structure of ribonuceic acid“. In: *Science (New York, N.Y.)* 147.3664, S. 1462–1465. DOI: 10.1126/science.147.3664.1462. URL: https://pubmed.ncbi.nlm.nih.gov/14263761/.

Hooper, Rowan (2005). „Boxer is first dog to have full genome revealed“. In: *New Scientist*. URL: https://www.newscientist.com/article/dn8430-boxer-is-first-dog-to-have-full-genome-revealed/.

Jackman, Shaun D. et al. (2017). „ABySS 2.0: resource-efficient assembly of large genomes using a Bloom filter“. In: *Genome Research* 27.5, S. 768–777. ISSN: 1549-5469. DOI: 10.1101/gr.214346.116. URL: https://genome.cshlp.org/content/27/5/768.

Jagannathan, V., C. Drögemüller und T. Leeb (2019). „A comprehensive biomedical variant catalogue based on whole genome sequences of 582 dogs and eight wolves“. In: *Animal Genetics* 50.6, S. 695–704. ISSN: 1365-2052. DOI: 10.1111/age.12834. URL: https://onlinelibrary.wiley.com/doi/full/10.1111/age.12834.

Jain, Miten, Hugh E. Olsen, Benedict Paten und Mark Akeson (2016). „The Oxford Nanopore MinION: delivery of nanopore sequencing to the genomics community“. In: *Genome Biology* 17.1, S. 239. ISSN: 1474-760X. DOI: 10.1186/s13059-016-1103-0. URL: https://link.springer.com/article/10.1186/s13059-016-1103-0.

Jung, Hyungtaek, Tomer Ventura, J. Sook Chung, Woo-Jin Kim, Bo-Hye Nam, Hee Jeong Kong, Young-Ok Kim, Min-Seung Jeon und Seong-Il Eyun (2020). „Twelve quick steps for genome assembly and annotation in the classroom“. In: *PLoS computational biology* 16.11, e1008325. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1008325. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1008325.

Kasianowicz, J. J., E. Brandin, D. Branton und D. W. Deamer (1996). „Characterization of individual polynucleotide molecules using a membrane channel“. In: *Proceedings of the National Academy of Sciences of the United States of America* 93.24, S. 13770–13773. ISSN: 0027-8424. DOI: 10.1073/pnas.93.24.13770. URL: https://www.pnas.org/content/93/24/13770.

Klingström, Tomas, Erik Bongcam-Rudloff und Olga Vinnere Pettersson (2018). „A comprehensive model of DNA fragmentation for the preservation of High Molecular Weight DNA“. In: *bioRxiv*, S. 254276. DOI: 10.1101/254276. URL: https://www.biorxiv.org/content/10.1101/254276v3.

Kolmogorov, Mikhail, Jeffrey Yuan, Yu Lin und Pavel A. Pevzner (2019). „Assembly of long, error-prone reads using repeat graphs“. In: *Nature Biotechnology* 37.5, S. 540–546. ISSN: 1546-1696. DOI: 10.1038/s41587-019-0072-8. URL: https://www.nature.com/articles/s41587-019-0072-8.

Krzeminska, P., M. Stachowiak, M. Skrzypski, T. Nowak, A. Maslak und M. Switonski (2020). „Altered expression of CYP17A1 and CYP19A1 in undescended testes of dogs with unilateral cryptorchidism“. In: *Animal Genetics* 51.5, S. 763–771. ISSN: 1365-2052. DOI: 10.1111/age.12977. URL: https://onlinelibrary.wiley.com/doi/10.1111/age.12977.

Lander, E. S. et al. (2001). „Initial sequencing and analysis of the human genome“. In: *0028-0836* 409.6822, S. 860–921. ISSN: 0028-0836. DOI: 10.1038/35057062. URL: https://deepblue.lib.umich.edu/handle/2027.42/62798.

Langmead, Ben und Steven L. Salzberg (2012). „Fast gapped-read alignment with Bowtie 2“. In: *Nature Methods* 9.4, S. 357–359. ISSN: 1548-7105. DOI: 10.1038/nmeth.1923. URL: https://pubmed.ncbi.nlm.nih.gov/22388286/.

Li, Heng (2011). „A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data“. In: *Bioinformatics (Oxford, England)* 27.21, S. 2987–2993. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btr509. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3198575/.

Li, Heng (2014). „Fast construction of FM-index for long sequence reads“. In: *Bioinformatics (Oxford, England)* 30.22, S. 3274–3275. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btu541. URL: https://pubmed.ncbi.nlm.nih.gov/25107872/.

Li, Heng (2020). *auN: a new metric to measure assembly contiguity*. URL: https://lh3.github.io/2020/04/08/a-new-metric-on-assembly-contiguity.

Lindblad-Toh, Kerstin et al. (2005). „Genome sequence, comparative analysis and haplotype structure of the domestic dog“. In: *Nature* 438.7069, S. 803–819. ISSN: 1476-4687. DOI: 10.1038/nature04338. URL: https://www.nature.com/articles/nature04338.

Luo, Ruibang et al. (2012). „SOAPdenovo2: an empirically improved memory-efficient short-read de novo assembler“. In: *GigaScience* 1.1. DOI: 10.1186/2047-217X-1-18. URL: https://academic.oup.com/gigascience/article/1/1/2047-217X-1-18/2656146.

McDonnell, Erin, Kimchi Strasser und Adrian Tsang (2018). „Manual Gene Curation and Functional Annotation“. In: *Fungal Genomics*. Hrsg. von Ronald P. de Vries, Adrian Tsang und Igor V. Grigoriev. SpringerLink Bücher. New York, NY: Humana Press, S. 185–208. ISBN: 978-1-4939-7804-5. DOI: 10.1007/978-1-4939-7804-5{\textunderscore}16.

Miescher, F. (1869). „Die Histochemischen und Physi-ologischen Arbeiten von Friedrich Miescher - Aus dem wissenschaft-lichen Briefwechsel von F. Miesche: Letter I; to Wilhelm His“. In: *His, W., et al. (Eds.), vol. 1. F.C.W. Vogel, Leipzig,pp. 33–38.*

Neff, Ellen P. (2019). „Whole genome sequencing has gone to the dogs“. In: *Lab Animal* 48.6, S. 166. ISSN: 1548-4475. DOI: 10.1038/s41684-019-0315-9. URL: https://www.nature.com/articles/s41684-019-0315-9.

Olmert, Meg Daley (2018). „Genes unleashed: how the Victorians engineered our dogs“. In: *Nature* 562.7727, S. 336–337. ISSN: 1476-4687. DOI: 10.1038/d41586-018-07039-z. URL: https://www.nature.com/articles/d41586-018-07039-z.

Ostrander, E. (2017). *Genetics and the Shape of Dogs: Studying the new sequence of the canine genome shows how tiny genetic changes can create enormous variation within a single species*. URL: https://www.americanscientist.org/article/genetics-and-the-shape-of-dogs.

PacBio (2020). „Beyond Contiguity – Assessing the Quality of Genome Assemblies with the 3 C's“. In: *PacBio*. URL: https://www.pacb.com/blog/beyond-contiguity/.

Parker, Heidi G., Dayna L. Dreger, Maud Rimbault, Brian W. Davis, Alexandra B. Mullen, Gretchen Carpintero-Ramirez und Elaine A. Ostrander (2017). „Genomic Analyses Reveal the Influence of Geographic Origin, Migration, and Hybridization on Modern Dog Breed Development“. In: *Cell reports* 19.4, S. 697–708. ISSN: 2211-1247. DOI: 10.1016/j.celrep.2017.03.079. URL: https://pubmed.ncbi.nlm.nih.gov/28445722/.

Payne, Alexander, Nadine Holmes, Vardhman Rakyan und Matthew Loose (2018). „Whale watching with BulkVis: A graphical viewer for Oxford Nanopore bulk fast5 files“. In: *bioRxiv*, S. 312256. DOI: 10.1101/312256. URL: https://www.biorxiv.org/content/10.1101/312256v1.

Pfeiffer, Franziska, Carsten Gröber, Michael Blank, Kristian Händler, Marc Beyer, Joachim L. Schultze und Günter Mayer (2018). „Systematic evaluation of error rates and causes in short samples in next-generation sequencing“. In: *Scientific Reports* 8.1, S. 10950. ISSN: 2045-2322. DOI: 10.1038/s41598-018-29325-6. URL: https://www.nature.com/articles/s41598-018-29325-6.

Plassais, Jocelyn, Jaemin Kim, Brian W. Davis, Danielle M. Karyadi, Andrew N. Hogan, Alex C. Harris, Brennan Decker, Heidi G. Parker und Elaine A. Ostrander (2019). „Whole genome sequencing of canids reveals genomic regions under selection and variants influencing morphology“. In: *Nature Communications* 10.1, S. 1489. ISSN: 2041-1723. DOI: 10.1038/s41467-019-09373-w. URL: https://www.nature.com/articles/s41467-019-09373-w.

Player, Robert A., Ellen R. Forsyth, Kathleen J. Verratti, David W. Mohr, Alan F. Scott und Christopher E. Bradburne (2021). „A novel canis lupus familiaris reference genome improves variant resolution for use in breed-specific GWAS“. In: *Life science alliance* 4.4. ISSN: 2575-1077. DOI: 10.26508/lsa.202000902. URL: https://pubmed.ncbi.nlm.nih.gov/33514656/.

Pop, M. (2009). „Genome assembly reborn: recent computational challenges“. In: *Briefings in Bioinformatics* 10.4, S. 354–366. ISSN: 1467-5463. DOI: 10.1093/bib/bbp026. URL: https://academic.oup.com/bib/article/10/4/354/299108?login=true.

Rhie, Arang, Brian P. Walenz, Sergey Koren und Adam M. Phillippy (2020). „Merqury: reference-free quality, completeness, and phasing assessment for genome assemblies“. In: *Genome Biology* 21.1, S. 245. ISSN: 1474-760X. DOI: 10.1186/s13059-020-02134-9. URL: https://genomebiology.biomedcentral.com/articles/10.1186/s13059-020-02134-9.

Sanger, F., G. M. Air, B. G. Barrell, N. L. Brown, A. R. Coulson, C. A. Fiddes, C. A. Hutchison, P. M. Slocombe und M. Smith (1977). „Nucleotide sequence of bacteriophage phi X174 DNA“. In: *Nature* 265.5596, S. 687–695. ISSN: 1476-4687. DOI: 10.1038/265687a0. URL: https://www.nature.com/articles/265687a0.

Serpell, James A. (2021). „Commensalism or Cross-Species Adoption? A Critical Review of Theories of Wolf Domestication“. In: *Frontiers in Veterinary Science* 8, S. 662370. ISSN: 2297-1769. DOI: 10.3389/fvets.2021.662370. URL: https://www.frontiersin.org/articles/10.3389/fvets.2021.662370/full.

Simão, Felipe A., Robert M. Waterhouse, Panagiotis Ioannidis, Evgenia V. Kriventseva und Evgeny M. Zdobnov (2015). „BUSCO: assessing genome assembly and annotation completeness with single-copy orthologs“. In: *Bioinformatics (Oxford, England)* 31.19, S. 3210–3212. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btv351. URL: https://academic.oup.com/bioinformatics/article/31/19/3210/211866.

Sohn, Jang-il und Jin-Wu Nam (2016). „The present and future of de novo whole-genome assembly“. In: *Briefings in Bioinformatics* 19.1, bbw096. ISSN: 1467-5463. DOI: 10.1093/bib/bbw096. URL: https://academic.oup.com/bib/article/19/1/23/2339783?login=true.

Stoler, Nicholas und Anton Nekrutenko (2021). „Sequencing error profiles of Illumina sequencing instruments“. In: *NAR Genomics and Bioinformatics* 3.1, lqab019. DOI: 10.1093/nargab/lqab019. URL: https://academic.oup.com/nargab/article/3/1/lqab019/6193612.

Tang, Li, Min Li, Fang-Xiang Wu, Yi Pan und Jianxin Wang (2019). „MAC: Merging Assemblies by Using Adjacency Algebraic Model and Classification“. In: *Frontiers in Genetics* 10, S. 1396. ISSN: 1664-8021. DOI: 10.3389/fgene.2019.01396. URL: https://www.frontiersin.org/articles/10.3389/fgene.2019.01396/full.

van Dijk, Erwin L., Yan Jaszczyszyn, Delphine Naquin und Claude Thermes (2018). „The Third Revolution in Sequencing Technology“. In: *Trends in genetics : TIG* 34.9, S. 666–681. ISSN: 0168-9525. DOI: 10.1016/j.tig.2018.05.008. URL: https://pubmed.ncbi.nlm.nih.gov/29941292/.

Vicedomini, Riccardo, Francesco Vezzi, Simone Scalabrin, Lars Arvestad und Alberto Policriti (2013). „GAM-NGS: genomic assemblies merger for next generation sequencing“. In: *BMC Bioinformatics* 14 Suppl 7.7, S6. ISSN: 1471-2105. DOI: 10.1186/1471-2105-14-S7-S6. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-14-S7-S6.

Walker, Bruce J. et al. (2014). „Pilon: an integrated tool for comprehensive microbial variant detection and genome assembly improvement“. In: *PLOS ONE* 9.11, e112963. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0112963. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0112963.

Wang, Chao et al. (2021a). „A novel canine reference genome resolves genomic architecture and uncovers transcript complexity“. In: *Communications Biology* 4.1, S. 185. ISSN: 2399-3642. DOI: 10.1038/s42003-021-01698-x. URL: https://www.nature.com/articles/s42003-021-01698-x.

Wang, Jeremy R., James Holt, Leonard McMillan und Corbin D. Jones (2018). „FMLRC: Hybrid long read error correction using an FM-index“. In: *BMC Bioinformatics* 19.1, S. 50. ISSN: 1471-2105. DOI: 10.1186/s12859-018-2051-3. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-018-2051-3.

Wang, Peipei, Fanrui Meng, Bethany M. Moore und Shin-Han Shiu (2021b). „Impact of short-read sequencing on the misassembly of a plant genome“. In: *BMC Genomics* 22.1, S. 99. ISSN: 1471-2164. DOI: 10.1186/s12864-021-07397-5. URL: https://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-021-07397-5.

Watson, J. D. und F. H. C. Crick (1953). „The structure of DNA“. In: *Cold Spring Harbor Symposia on Quantitative Biology* 18, S. 123–131. ISSN: 1943-4456. DOI: 10.1101/SQB.1953.018.01.020. URL: http://symposium.cshlp.org/content/18/123.short.

Whitfield, John (2003). „Dog genome unveiled“. In: *Nature*. ISSN: 1476-4687. DOI: 10.1038/news030922-17. URL: https://www.nature.com/articles/news030922-17.

Wick, Ryan R., Louise M. Judd und Kathryn E. Holt (2019). „Performance of neural network basecalling tools for Oxford Nanopore sequencing“. In: *Genome Biology* 20.1, S. 129. ISSN: 1474-760X. DOI: 10.1186/s13059-019-1727-y. URL: https://genomebiology.biomedcentral.com/articles/10.1186/s13059-019-1727-y.

Wickham, Hadley (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN: 978-3-319-24277-4. URL: https://ggplot2.tidyverse.org.

Wickham, Hadley et al. (2019). „Welcome to the tidyverse“. In: *Journal of Open Source Software* 4.43, S. 1686. DOI: 10.21105/joss.01686.

Xie, Luyu und Limsoon Wong (2021). „PDR: a new genome assembly evaluation metric based on genetics concerns“. In: *Bioinformatics (Oxford, England)* 37.3, S. 289–295. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btaa704. URL: https://pubmed.ncbi.nlm.nih.gov/32761066/.

Ye, Chengxi, Zhanshan Sam Ma, Charles H. Cannon, Mihai Pop und Douglas W. Yu (2012). „Exploiting sparseness in de novo genome assembly“. In: *BMC Bioinformatics* 13 Suppl 6.6, S1. ISSN: 1471-2105. DOI: 10.1186/1471-2105-13-S6-S1. URL: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-13-S6-S1.

Zhang, Haowen, Chirag Jain und Srinivas Aluru (2020). „A comprehensive evaluation of long read error correction methods“. In: *BMC Genomics* 21.Suppl 6, S. 889. ISSN: 1471-2164. DOI: 10.1186/s12864-020-07227-0. URL: https://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-020-07227-0.

Zimin, Aleksey V., Guillaume Marçais, Daniela Puiu, Michael Roberts, Steven L. Salzberg und James A. Yorke (2013). „The MaSuRCA genome assembler“. In: *Bioinformatics (Oxford, England)* 29.21, S. 2669–2677. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btt476. URL: https://academic.oup.com/bioinformatics/article/29/21/2669/195975.

Zimin, Aleksey V. und Steven L. Salzberg (2020). „The genome polishing tool POLCA makes fast and accurate corrections in genome assemblies“. In: *PLoS computational biology* 16.6, e1007981. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1007981. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1007981.

Zimin, Aleksey V. und Steven L. Salzberg (2022). „The SAMBA tool uses long reads to improve the contiguity of genome assemblies“. In: *PLoS computational biology* 18.2, e1009860. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1009860. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1009860.

Internetdokumente

URL-01 (2020). *ROS_Cfam_1.0 (letzter Zugriff am: 11.08.2022)*. URL: https://www.ncbi.nlm.nih.gov/assembly/GCF_014441545.1/.

URL-02 (2021). *DNA Sequencing | Understanding the genetic code (letzter Zugriff am: 15.12.2021)*. URL: https://emea.illumina.com/techniques/sequencing/dna-sequencing.html.

URL-03 (2021). *RNA Sequencing | RNA-Seq methods and workflows (letzter Zugriff am: 15.12.2021)*. URL: https://emea.illumina.com/techniques/sequencing/rna-sequencing.html.

URL-04 (2020). *Dog10K_Boxer_Tasha (letzter Zugriff am: 27.10.2022)*. URL: https://www.ncbi.nlm.nih.gov/assembly/GCF_000002285.5/.

URL-05 (2011). *Quality Scores for Next-Generation Sequencing: Assessing sequencing accuracy using Phred quality scoring: Technical Note: Sequencing (letzter Zugriff am: 02.11.2022)*. URL: https://www.illumina.com/documents/products/technotes/technote_Q-Scores.pdf.

URL-06 (2022). *QUAST 5.2.0 manual (letzter Zugriff am: 27.10.2022)*. URL: https://quast.sourceforge.net/docs/manual.html#sec3.

URL-07 (2022). *Welcome to immense discovery power (letzter Zugriff am: 11.08.2022)*. URL: https://emea.illumina.com/systems/sequencing-platforms/novaseq.html.

URL-08 (2021). *How basecalling works (letzter Zugriff am: 11.08.2022)*. URL: https://nanoporetech.com/how-it-works/basecalling.

URL-09 (2022). *MaSuRCA Genome Assembly and Analysis Toolkit Quick Start Guide (letzter Zugriff am: 11.08.2022)*. URL: https://github.com/alekseyzimin/masurca.

URL-10 (2022). *Flye-USAGE.md (letzter Zugriff am: 28.09.2022)*. URL: https://github.com/fenderglass/Flye/blob/flye/docs/USAGE.md.

URL-11 (2020). *UU_Cfam_GSD_1.0 (letzter Zugriff am: 27.10.2022)*. URL: https://www.ncbi.nlm.nih.gov/assembly/GCF_011100685.1/.

URL-12 (2018). *FastQC_Manual (letzter Zugriff am: 06.11.2022)*. URL: https://dnacore.missouri.edu/PDF/FastQC_Manual.pdf.

URL-13 (2022). *Why does the per base sequence quality decrease over the read in Illumina? (letzter Zugriff am: 28.09.2022)*. URL: https://www.ecseq.com/support/ngs/why-does-the-sequence-quality-decrease-over-the-read-in-illumina.

URL-14 (2019). *Diagnosing problems with phasing and pre-phasing on Illumina platforms (letzter Zugriff am: 28.09.2022)*. URL: http://lab.loman.net/high-throughput%20sequencing/2013/11/21/diagnosing-problems-with-phasing-and-pre-phasing-on-illumina-platforms/.

URL-15 (2022). *NCBI Eukaryotic Genome Annotation Policy On Which Genomes Are Annotated (letzter Zugriff am: 07.10.2022)*. URL: https://www.ncbi.nlm.nih.gov/genome/annotation_euk/policy/.

URL-16 (2022). *Funannotate 1.7.0 documentation (letzter Zugriff am: 07.10.2022)*. URL: https://funannotate.readthedocs.io/en/latest/.

URL-17 (2022). *Bowtie 2: Manual (letzter Zugriff am: 13.10.2022)*. URL: https://bowtie-bio.sourceforge.net/bowtie2/manual.shtml.

URL-18 (2022). *User guide BUSCO v5.4.2 (letzter Zugriff am: 27.10.2022)*. URL: https://busco.ezlab.org/busco_userguide.html#interpreting-the-results.

URL-19 (2017). *An Introduction to Next-Generation Sequencing Technology*. URL: https://www.illumina.com/content/dam/illumina-marketing/documents/products/illumina_sequencing_introduction.pdf.

Glossar

auN beschreibt die Fläche unter der Kurve des N(x)-Plots. N(x) beschreibt, alle Perzentile von N1 bis N100.

Busco Completeness beschreibt das Vorhandensein/Fehlen essentieller Gene. Anhand von Busco interenen Datenbanken (verschiedene DB können ausgewählt werden entsprechend dem verwendeten Organismus) werden die hinterlegten Sequenzen auf dem Assembly gesucht. Dabei wird zwischen Single, Duplicate, Fragmented und Missing BUSCOs unterschieden. Die Busco Completeness ist die Summe aus Single und Duplicate BUSCOs.

Contig ist ein Satz sich überlappender Reads, die von der selben Quelle stammen.

Contiguity ist eine Metrik für die Qualität eines Assemblies. Diese beschreibt die Länge und Anzahl an Contigs in einem Assembly. In den meisten Fällen wird sie als N50 angegeben.

Coverage beschreibt das Verhältnis zwischen Sequenzen und Referenz. Die Read- oder Sequencing-Coverage beschreibt dabei die das Verhältnis der Gesamtzahl an Nukleotiden in den Reads geteilt durch die Anzahl der Nukleotide in der Referenz. Die Genom-Coverage beschreibt das Verhältnis der Anzahl an Nukleotiden im Assembly geteilt durch die Anzahl der Nukleotide in der Referenz.

diskordant bezeichnet ein Read-Alignment, bei dem die Paired-End-Reads zwar auf dem Assembly alignieren, aber nicht als Paar auftreten.

konkordant bezeichnet ein Read-Alignment, bei welchem die Paired-End-Reads auch als Paare auf dem Assembly auftreten.

N50 beschreibt die Qualität eines Assemblies im Bezug auf die Contiguity. Der N50 ist definiert als die Länge des kürzesten Contigs in einem der Größe nach sortierten Genom, welches 50 % aller Basen des Genom enthält.

Scaffold ist die Verknüpfung einer Reihe von nicht zusammenhängender genomischer Sequenzen (Contigs) durch das Einfügen von Lücken bekannter Länge.

Singlets bezeichnen ein Read-Alignment, bei dem nur der Forward- oder Reverse-Read der Paired-End-Reads auf dem Assembly aligniert und nicht beide.

Eidesstattliche Erklärung

Hiermit versichere ich – Tom Thoralf Fritzsche – an Eides statt, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Sämtliche Stellen der Arbeit, die im Wortlaut oder dem Sinn nach Publikationen oder Vorträgen anderer Autoren entnommen sind, habe ich als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt oder anderweitig veröffentlicht.

Mittweida, 29. November 2022	
Ort, Datum	Tom Thoralf Fritzsche, B.Sc.