Methods to improve short fragment NGS analysis - with a focus on ancient DNA

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

MSc. Alexander Seitz

aus Weingarten

Tübingen

2019

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	29.05.2019
Dekan:	Prof. Dr. Wolfgang Rosenstiel
1. Berichterstatter:	Prof. Dr. Kay Nieselt
2. Berichterstatter:	Prof. Dr. Daniel Huson

Abstract

The introduction of next-generation sequencing technologies led to an increase in the amount of sequencing data that needs to be analyzed. However, not all sequencing data are equally analyzable owing, for example, to differences in read length. This thesis addresses some of the problems related to the short reads generated by next-generation sequencing technologies. This is of great relevance, as most recent and ongoing sequencing projects still use next-generation sequencing. Admittedly, third generation sequencing, which is still in its infancy to date, promises the generation of much longer reads. However, some research questions cannot make use of these long-read technologies. A prominent example are ancient DNA projects that sequence fragments of DNA from ancient samples, which have degraded over time and are typically very short. The present thesis introduces several methods and programs addressing different research applications of next-generation sequencing data, which can be used especially when, as in the case of research on ancient DNA, have to deal with short fragments. We present methods for the reconstruction of the sequence of repetitive regions using only short reads by reconstructing each repetitive region separately, thus eliminating the problem of reads that map to multiple locations. Additionally, we present a two-layer approach to address the *de novo* assembly of short fragments, which generally contains reads of different lengths. Furthermore, we present an automated method to compare mapping-based reconstructions that share the same reference by analyzing them simultaneously. Finally, we present `SNPViz`, which visualizes amino acid affected by a SNP within the protein sequence and its corresponding three-dimensional structure. Taken together, this thesis presents several methods to improve the analysis of next-generation sequencing technologies. These methods can support researchers in better understanding their data and can help them uncover new foci for future research.

Kurzfassung

Die Einführung von Sequenziermethoden der nächsten Generation führte zu einem Anstieg an sequenzierten Daten, die analysiert werden müssen. Unterschiede in den Sequenzierdaten, wie zum Beispiel Reads unterschiedlicher Länge führen dazu, dass nicht alle Daten gleich ausgewertet werden können. Diese Arbeit beschäftigt sich mit Problemen, welche auf die kurzen Readlängen von Sequenziermethoden der nächsten Generation zurückzuführen sind. Die hier vorgestellten Methoden sind wichtig für aktuelle Sequenzierprojekte, da ein Großteil von ihnen diese Sequenziermethoden nutzen. Zugegebenermaßen versprechen Sequenziermethoden der dritten Generation, welche sich aktuell noch in der Anfangsphase befinden, viel längere Reads. Dennoch gibt es Forschungsfragen, welche nicht von diesen langen Reads profitieren können. Ein bedeutendes Beispiel hierfür sind Projekte, welche DNS Fragmente von alten Proben sequenzieren, da diese über die Zeit degradieren und deshalb typischerweise nur sehr kurz sind. Die vorgelegte Arbeit beschreibt mehrere Methoden und Programme, die sich auf unterschiedliche Forschungsfragen bezüglich Sequenziermethoden der nächsten Generation beziehen. Diese Methoden wurden gezielt für die Analyse von kurzen DNS Fragmenten entwickelt, wie sie auch in alten DNS Proben vorkommt, sind aber auch in Bezug auf andere Fragestellungen einsetzbar. Wir beschreiben Methoden für die Rekonstruktion von repetitiven Sequenzen, welche auf der Idee jede Region separate mit den kurzen Reads zu rekonstruieren beruht. Dies beseitigt das Problem, dass Reads an unterschiedliche Positionen im Genom platziert werden können. Des Weiteren zeigen wir eine Herangehensweise, welche auf zwei Schichten basiert, um die Denovo Assemblierung von kurzen Fragmenten mit unterschiedlichen Readlängen zu verbessern. Außerdem beschreiben wir einen automatisierten Ansatz um Rekonstruktionen, die auf dem Vergleich und der Positionsbestimmung der Reads mit einer bekannten Referenz basieren, zu vergleichen. Diese Methode basiert auf der Idee, alle Proben simultan zu analysieren. Zuletzt präsentieren wir `SNPViz`, ein Programm zur Visualisierung von Aminosäuren in der zugehörigen dreidimensionalen Struktur, die von Einzelnukleotid-Polimorphismen betroffen sind. Zusammengefasst beschreibt diese Arbeit Methoden um die Analyse von Daten, die mittels Sequenziermethoden der nächsten Generation erstellt wurden, zu verbessern. Diese Methoden können Forscher dabei unterstützen ihre Daten besser zu verstehen und ihnen helfen, neue Schwerpunkte für künftige Forschungsfragen zu finden.

Acknowledgments

First and foremost, I would like to thank my supervisor Prof. Kay Nieselt for the opportunity to work with her on many different and exciting research and collaboration projects. Without her support, encouragement, discussions, and freedom to explore my ideas, this Ph.D. would not have been possible. I would also like to thank her for her comments regarding this thesis, which helped me to improve its quality greatly.

I would also like to thank Prof. Daniel Huson for agreeing to co-supervise and review my dissertation.

Furthermore, I thank my colleagues at the Integrative Transcriptomics Group, André Hennig, Alexander Peltzer, Julia Söllner, and Judith Neukamm for helping to generate a great group environment. We always had great discussions, both on- and off-topic.

Additionally, I want to thank Linus Backert, Michael Römer, and Lara Ditrich for their stimulating discussions, as well as their feedback on this dissertation.

A special thanks goes to Niklas Heinsohn for his insights, as well as his ability to help me look at my problems from new viewpoints.

I am grateful for my friends and family who supported and encouraged me throughout my studies. Without them, I would never have made it this far.

Finally, my deepest gratitude goes to Natascha Barz, who helped me to not get lost in my work and patiently supported me while I finished writing.

In accordance with standard scientific protocol, I will use the personal pronoun ***we*** to indicate the reader and the writer, or my scientific collaborators and myself.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Technical advancements in next-generation sequencing (NGS) technologies enable scientists to generate large amounts of sequencing data from almost any genomic material at comparatively low costs (Bentley *et al.*, 2008). Thereby, these technologies have revolutionized modern sequencing projects, allowed for large sequencing studies, and made it possible to study ancient DNA (aDNA) (Der Sarkissian *et al.*, 2015). One example where the inclusion of aDNA can drastically improve the results is the determination of phylogenetic ancestry, for example when studying the evolution of human pathogens, thus in the generation of the corresponding phylogenetic trees. For example, Schuenemann *et al.* (2013) published the first phylogeny of *Mycobacterium leprae*, the bacterium responsible for the leprosy epidemic in the middle ages, using DNA extracted from victims of the respective outbreaks. A recent follow-up study (Schuenemann *et al.*, 2018a) extended these results and confirmed the original finding that the genome of the ancient bacterium itself is highly similar to the modern strains that still cause infections today. However, whereas an infection with *M. leprae* was a death sentence in the middle ages, the mortality rate is very low today (Noordeen, 1988). One possible explanation for this would be a shift in the *Human Leukocyte Antigen* (HLA) alleles towards HLAs that provide an improved resistance to infections with *M. leprae*. The study of Krause-Kyora *et al.* (2018) showed that the HLA allele *DRB1*15:01*, which is known to provide less resistance against leprosy, was present in a significantly higher percentage in the ancient human population as compared to today.

While methods developed for the analysis of aDNA, like DNA capture and enrichment protocols, are mostly applied to ancient samples, aDNA methods can also be applied in other fields that share the same restrictions as aDNA analyses. For example, these capture and enrichment methods can be applied to distinguish modern bacterial strains that are otherwise hard to cultivate (Rajwani *et al.*, 2018). A prominent example is the bacterium *Treponema pallidum*. Infections with the three subspecies *Treponema pallidum* subsp. *pallidum*, *Treponema pallidum* subsp. *pertenue*, and *Treponema pallidum* subsp. *endemicum* are the cause for the diseases syphilis, yaws, and bejel, respectively. Currently, there are no serological or immunological tests capable of differentiating between them, and an identification of the exact subtype is only possible using sequencing information (Marra *et al.*, 2010). In addition, these strains are hard to cultivate and need

to be amplified through rabbit passage. Although the three diseases manifest similar symptoms, they need to be treated differently, which can lead to unnecessary and ineffective antibiotic treatments if the subtype causing the infection is not identified correctly (Mitjà *et al.*, 2013). Arora *et al.* (2016) were able to use methods developed for the study of ancient genomics to sequence clinical samples of *T. pallidum*, which were extracted directly from the patients. Additional work by Pinto *et al.* (2016) and Sun *et al.* (2016) made it possible to gain more insight into the genetic diversity of this bacterium. Recently, Schuenemann *et al.* (2018b) published the first ancient syphilis genome sequences, which may lead to a better understanding of the origin and evolution of *T. pallidum*.

These publications reconstruct the respective genomes based on the sequencing of the DNA from the isolated samples. In general, there are two possibilities to reconstruct a genome based on the sequenced reads (Pop, 2009). In a *de novo* assembly, overlapping reads are extended into longer, continuous sequences called contigs, whereas in a so-called mapping-based assembly, all reads are aligned against a closely related reference genome.

Currently, the majority of sequencing projects uses NGS technologies to generate the sequencing reads (Heather and Chain, 2016). However, one major drawback of NGS technologies is the shortness of the reads they produce (Van Dijk *et al.*, 2014) compared to the longer Sanger reads. This makes the reconstruction of genomes more difficult (Li *et al.*, 2010; Sawyer *et al.*, 2012). Large numbers of sequencing reads, combined with the usage of paired-end and mate-pair sequencing can mitigate this problem. The known distance between the forward and reverse read, called insert size, can be integrated into the reconstruction of the genome and can enhance the reconstruction of repetitive regions or the determination of the order of the contigs in a *de novo* assembly.

To be able to sequence enough material of ancient bacterial pathogens from human bones (Avila-Arcos *et al.*, 2011) or modern, clinical pathogens from human tissue (Mothershed and Whitney, 2006), the target DNA needs to be extracted. This extraction is designed to select mainly the DNA fragments from the target organism and uses the principle of hybridization (Maricic *et al.*, 2010). Regions complementary to the target sequence are fixed to probes in order to extract and consequently enrich only the target DNA. However, these methods were developed for short read sequencing technologies, which is why they mainly recover short DNA fragments (Eckert *et al.*, 2016). Furthermore, the DNA in the ancient organisms has degraded over time. Therefore, the lengths of the fragments extracted from aDNA samples are expected to lie between 44 and 172 bp (Sawyer *et al.*, 2012). Because there are fragments of longer sizes, they are still sequenced with the maximum possible length to gain the most information. On the shorter fragments, this leads to overlapping forward and reverse reads with a negative inner mate pair distance.

In addition to the reconstruction of the full genome, where the order of the bases, including possible genomic rearrangements, are of particular interest, sequencing projects often only focus on the identification of single-nucleotide variations (SNVs) in the se-

quenced genome (Nič *et al.*, 2009). These variations can then be used for phylogenetic analyses or be studied to determine their structural and functional impact, for example in hereditary or genetic diseases (Zhao *et al.*, 2014). If such a variation lies within the coding region of a gene, it can lead to amino acid exchanges in the protein, which in turn can change the protein's structure and, thus, its function (Krawczak *et al.*, 2000). Even if a SNV lies outside of the coding region, it can also influence a gene and thus the phenotype of the sequenced specimen. For example, if a SNV lies within the promoter region, it can affect the expression of the gene and thus the amount of corresponding protein in the cells.

Thus, these sequencing projects need computational methods that can analyze many samples reliably and efficiently. One pipeline developed for the mapping-based reconstruction of aDNA samples is `EAGER` (Peltzer *et al.*, 2016). However, its main focus lies on the reconstruction of the individual samples, without addressing the comparative analyses.

1.1 Contributions of this thesis

This thesis is based upon research at the Integrative Transcriptomics group of the University of Tübingen. It was driven by collaboration projects with the Paleogenetic department of the University of Tübingen, the Max Planck Institute for the Science of Human History in Jena, the Institute of Forensic Medicine at the University of Zürich, and other departments from different institutions. It focuses on the analysis of NGS data, with a particular interest in ancient DNA and hard to cultivate clinical bacteria, like *T. pallidum*. One primary goal was to overcome problems related to the short fragments that are obtained when sequencing these kinds of samples. Additional research questions evolved about the identification of variations in different genomes, as well as about inferring a functional impact on the proteins that show SNVs in comparison to the reference protein.

`MUSIAL`: Postprocessing of multiple samples

While it is possible to generate a mapping-based reconstruction of multiple samples with pipelines like `EAGER`, they do not address the comparative analyses needed, for example, to generate a phylogenetic tree. `MUSIAL`, which focuses on the comparison of several samples that were mapped against the same reference genome, addresses this issue. For each sample, the results of the genotyping are used to generate a base-call for each position in the reference genome. Based on these calls, the SNVs that are identified in each sample are summarized in a snvTable to compare and identify mutations that are shared between the samples. Additionally, it is possible to automatically annotate the functional impact of the SNVs using `SNPeff`. For further phylogenetic analysis, `MUSIAL` also creates several alignments, based on the positions of the reference genome. These alignments include a whole genome alignment, a SNV alignment, and, if desired, alignments on different genes. For a quick overview, `MUSIAL` generates a summary containing only statistics like the number of SNVs or the number of unresolved positions. Finally, if allowing for small insertions and deletions (indels), the reconstructed genomes are generally of different length. To allow for indels, `MUSIAL` generates an extended reference containing all insertions, so that it is possible to account for all variations in all samples.

DACCOR: Resolving repetitive regions

`DACCOR` is an implementation of an automated pipeline using mapping-based assembly to reconstruct full genomes with an increased base-pair resolution in their repetitive regions from short-read NGS data. To achieve this for all repetitive regions, `DACCOR` first uses a *k*-mer based approach to identify all repetitive regions in the given reference *de novo*. The increased resolution in the repetitive regions is achieved by reconstructing each repetitive region separately and then inserting them into the respective regions in the reconstructed full genome of each sample. This approach allows for the reconstruction of the repetitive regions, which is not possible when using only the complete genome as a reference for

the mapping. Furthermore, it becomes possible to infer variations that are present only in some, but not all, copies of the repetitive regions.

MADAM: *De novo* assembly

MADAM was developed to overcome problems in short fragment *de novo* assembly, which is the case in aDNA or hard to cultivate clinical bacteria. Here, the reconstruction has to be based solely on the sequence of the short reads without the possibility to use additional information contained in short reads generated from long fragments, like an approximate distance between the forward and reverse reads. The idea of MADAM is to use several different *k*-mer sizes for *de Bruijn* based assemblies, followed by another assembly based on string overlaps. The second, overlap based assembly uses the different assemblies generated in the initial *k*-mer based assemblies as input. This two-layer approach can make use of the information contained in both the shorter, as well as the longer fragments of the sequenced samples to increase the quality of the assembly.

SNPViz: Visualizing SNPs in Proteins

The idea of SNPViz is to use three-dimensional information of the proteins to infer possible structural instabilities and thus clinical relevance for SNVs. SNPViz automatically identifies the protein, and, if available, the three-dimensional structure contained in the Protein DataBase (PDB). Additionally, the amino acid affected by the SNV is identified and highlighted in an interactive visualization of the protein. This visualization may give insights into which SNVs may change the structure and therefore the function of the protein.

1.2 Thesis structure

This thesis is structured into seven chapters. The next chapter provides the biological, statistical, and bioinformatical background of the research this thesis is based upon. It encompasses a brief overview of the history of the sequencing technology, and a subsequent introduction into aDNA projects, including their challenges. This is followed by an overview of phylogenetic reconstruction, as well as some methods for the comparison of phylogenetic trees. Finally, current methods for the reconstruction of genomes from NGS sequencing data are introduced. This part is separated into mapping-based and *de novo* reconstruction methods and outlines their advantages, shortcomings, and statistical methods that can be used to compare them.

Chapter 3 on page 19 describes the structure and workings of MUSIAL, a tool for the comparison of multiple samples based on mapping-based reconstructions. This is followed by a detailed description of DACCOR in Chapter 4 on page 39, a program developed

to increase the base-pair resolution in repetitive regions for mapping-based reconstruction of NGS data. Afterward, the program `MADAM` is introduced in Chapter 5 on page 57. It is designed to improve the quality of *de novo* assemblies based on aDNA data. Chapter 6 on page 77 presents `SNPViz`, a program to identify the amino acids in the corresponding protein that are affected by SNVs, together with an interactive visualization of the protein, including the possibility to highlight these amino acids.

Finally, the programs presented in this thesis are discussed in Chapter 7 on page 85, including an outlook into possible future research questions.

Chapter 2

Background

This chapter introduces the biological and computational background of this thesis. First, an overview of DNA sequencing technologies, loosely based on the review of Heather and Chain (2016), is presented. This is followed by a discussion of specific characteristics of aDNA, a short introduction into phylogenetic trees, and into current methods for the reconstruction of a genome from sequencing data. This chapter ends with an introduction into the current methods for the detection of genetic variations and their biological significance.

2.1 DNA sequencing

The goal of DNA (deoxyribonucleic acid) sequencing is to determine how the four amino acids Adenine (A), Cytosine (C), Guanine (G), and Thymine (T) that comprise the DNA are ordered in the genome of the desired target organism (Nič *et al.*, 2009). After Watson and Crick solved the three-dimensional structure of the DNA in 1953 (Watson and Crick, 1953), it took another 12 years, before the group around Robert Holley was able to determine the first whole DNA sequence, the alanine tRNA sequence from *Saccharomyces cerevisiae* (Holley *et al.*, 1965). The breakthrough that changed sequencing technology forever came in 1977, when Sanger *et al.* (1977) developed the chain termination method, which nowadays is known as the "Sanger" method. This sequencing method produces so-called reads that are slightly shorter than one kilobase (kb) in length (Heather and Chain, 2016). The sequencing breakthroughs were further aided by the development of the polymerase chain reaction (PCR), a method to rapidly amplify targeted DNA fragments (Saiki *et al.*, 1988a,b).

The next big step in DNA sequencing, called pyrosequencing and licensed by 454 Life Sciences (Ronaghi *et al.*, 1998), was the first commercially available "next-generation" sequencing (NGS) technology. This sequencing technology, which was later purchased by Roche, changed the field of DNA research, as the sequencing reactions could be massively parallelized. Thus, the amount of DNA sequenced in one run of this pyrosequencing technique was drastically increased.

The most important sequencing technology that followed after the success of the 454 technique was the Solexa sequencing technology, which was later acquired by Illumina

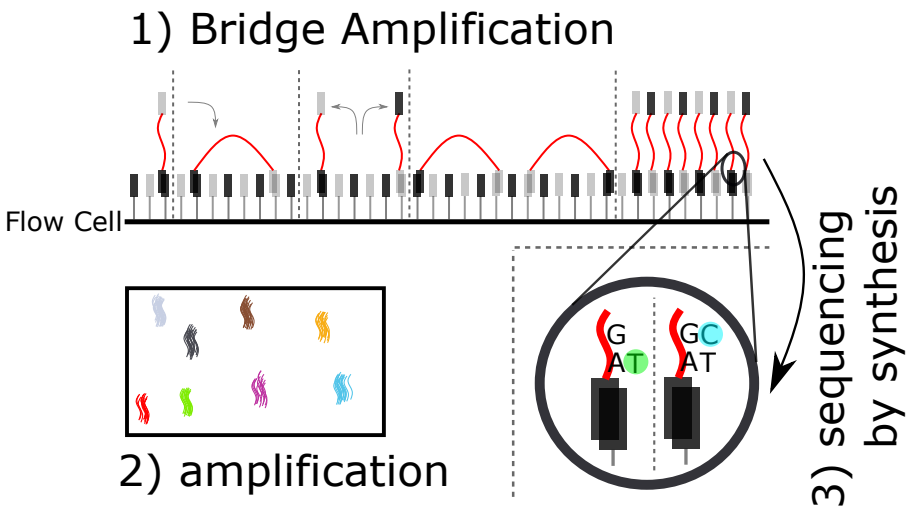


Figure 2.1: Methodology of the Illumina sequencing technology. After the DNA fragments are bound to the flow cell, they are bridge-amplified (top). The result are clusters containing many copies of the same fragments (bottom left). Finally, labeled nucleotides are added that bind to the fragments. They are excited through a laser and emit a specific color spectrum. This is done one base after another and is called "sequencing by synthesis" (bottom right). Figure adapted from Heather and Chain (2016) and Illumina (2009).

technologies (Voelkerding *et al.*, 2009). Its central idea is called "sequencing by synthesis" (Illumina, 2010). The general workflow of this technology, including both, the bridge amplification and the sequencing process is illustrated in Figure 2.1. For this, the DNA is first amplified using solid-phase amplification. Here, the DNA is randomly fragmented, and adapters are ligated to both ends of the fragments. The single-stranded fragments are bound randomly to the surface of the flow cell channels. After the addition of unlabeled nucleotides and enzymes to initiate the solid-phase bridging, both ends of the single-stranded DNA fragments are bound to the flow cell. The unlabeled nucleotides bind to the single-strand DNA and build double-stranded bridges. After denaturation, the number of single-stranded templates anchored to the flow cell is doubled. These steps are repeated until there are several million dense clusters of double-stranded DNA in each channel of the flow cell. The first base of the bound single-stranded DNA fragments is determined by adding four labeled reversible terminators, special primers, and DNA polymerase. They bind to the primers and the first base of the actual fragment. After laser excitation, the DNA fragment's first base can be identified based on the emitted fluorescence from each cluster. Afterward, again four labeled reversible terminators, primers, and DNA polymerase are added, and the second base is captured as before. This is repeated to determine the sequence of the nucleotides of the fragments, one base at a time for each cluster. After a predetermined number of bases are sequenced, it is possible to bind the other end of the DNA fragments to the flow cell, again using the bridging

already used for the bridge amplification. This allows for the sequencing of the other end of the DNA fragment, which is called paired-end sequencing. The resulting sequences of the sequencing experiment are called reads, where there is one or, in the case of paired-end sequencing, two reads for each cluster on the flow cell. Because the length of the sequenced fragments can be approximated, this paired-end sequencing creates additional information about the distance between two sequenced reads of the same fragment. This sequencing of both ends of the fragments could be further extended to the sequencing of very long fragments, called mate-pair sequencing (Illumina, 2009). In this case, long DNA fragments (2-5 kb) are extended with biotin-labeled Deoxynucleotides (dNTPs). These dNTPs can bind to each other so that the fragment is circularized. These DNA circles are then fragmented into fragments of 400-800 bps. The fragments containing biotin in the middle of the fragment are then extracted and sequenced using paired-end sequencing, thus actually sequencing the ends of the long DNA fragment.

The difference of the NGS technologies to the so-called third generation sequencing technologies is discussed heavily (Schadt *et al.*, 2010; Niedringhaus *et al.*, 2011; Pareek *et al.*, 2011; Gut, 2013; Heather and Chain, 2016). In our opinion, it is a combination of being able to sequence single DNA fragments, without the need for amplification and being able to generate very long reads. The first successful technology that fulfilled these requirements was probably the single-molecule real-time (SMRT) technology from Pacific Biosciences (PacBio). It is based on zero-mode waveguides, which are tiny holes in a metallic film covering a chip. Through them, properties of light passing through a diameter smaller than its wavelength can be exploited. A DNA polymerase molecule can be placed inside one of these waveguides. The deposition of a single DNA polymerase molecule inside one of the waveguides illuminates them. As the washed over fluorescent dNTPs are incorporated into the DNA, single nucleotides can be monitored in real time, as only the ones that are being incorporated provide a detectable signal. It can generate reads of over 10 kb in length.

Recent advancements in Nanopore sequencing promise to generate very long reads from non-amplified sequence data at a fraction of the current costs and in a fraction of the time currently required (Branton *et al.*, 2008). This technology electrophoretically drives a single-stranded DNA fragment through large α-hemolysin ion channels (Kasianowicz *et al.*, 1996). The passage of the DNA molecule through the channel alters the ion flow, which can be measured and is slightly different for each nucleotide. The use of non-biological, solid-state technology to generate other Nanopores may also allow the sequencing of double-stranded DNA molecules (Li *et al.*, 2001; Dekker, 2007). The main problem with this technology is its sequencing error rate, which lies between 10% and 20% (Koren *et al.*, 2017). However, Nanopore sequencing technology was already used for the generation of reference datasets (Quick *et al.*, 2014; Loman *et al.*, 2015), and may be combined with Illumina sequencing for the same sample. In this case, the Illumina reads are often mapped against the Nanopore reads to correct the errors in the Nanopore reads (Karlsson *et al.*, 2015; Ashton *et al.*, 2015; Madoui *et al.*, 2015).

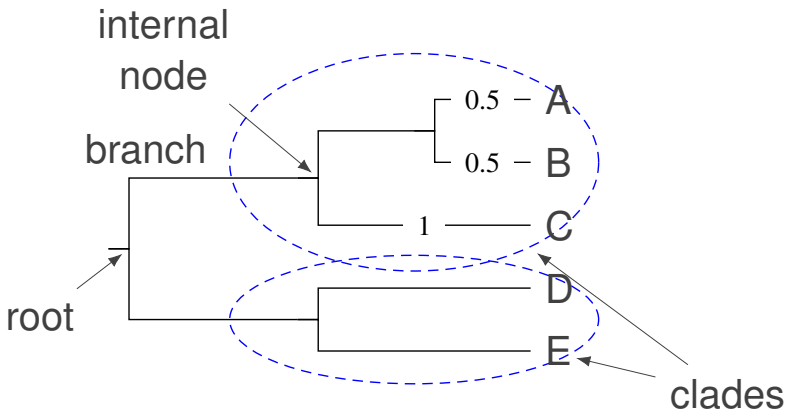
2.2 Ancient DNA

The discovery of the presence of DNA in ancient mummies (Pääbo, 1985) and the introduction of NGS technology, coupled with the development of DNA hybridization and enrichment (Mamanova *et al.*, 2010), have resulted in many ancient DNA (aDNA) projects. As of today, genomes from ancient humans (Haak *et al.*, 2015; Allentoft *et al.*, 2015), Neanderthals (Prüfer *et al.*, 2014), Denisovans (Sawyer *et al.*, 2015), different pathogens (Schuenemann *et al.*, 2018a), and many more species (Mitchell *et al.*, 2014; Evin *et al.*, 2015; Kehlmaier *et al.*, 2017) have been sequenced, reconstructed, and analyzed. However, the analysis of aDNA always needs to address several problems specific to aDNA sequencing. One of the most significant problems is the degradation of the DNA over time, which leads to shorter DNA fragments. These aDNA fragments typically have lengths of sizes between 44 and 172 bp (Sawyer *et al.*, 2012). The use of paired-end sequencing thus often leads to negative inner mate-pair distances (overlapping forward and reverse reads). Additionally, sequencing with long read technologies, like PacBio, will not lead to a gain in information as the DNA fragment is already sequenced entirely by the short-read technologies.

Another problem with aDNA is the deamination of cytosine to uracil over time (Rasmussen *et al.*, 2010). Because sequencing technologies identify uracil as a thymine rather than a cytosine base, these deaminations lead to sequencing errors. However, because contamination from both modern and ancient sources are another problem when analyzing aDNA, these deaminations can be used for the authentication of aDNA (Hofreiter *et al.*, 2001). When mapping aDNA reads against a suitable reference genome (see Section 2.4.2 below), these errors can be detected. In addition, treating the DNA fragments with *Uracil-DNA Glycosylase* (UDG) before sequencing, can resolve most of these errors (Briggs *et al.*, 2010).

The last major problem with the analysis of aDNA is its metagenomic nature and, in consequence, a low amount of endogenous DNA, because, over time, other bacteria and microbes populate the remains (Sawyer *et al.*, 2012). Consequently, the resulting DNA sequencing experiment will always be based on a metagenomic sample (Tringe and Rubin, 2005).

The aforementioned DNA hybridization enrichment, also called DNA capture methods (Avila-Arcos *et al.*, 2011), can be used to increase the amount of endogenous DNA. For this procedure, short DNA sequences complementary to the desired genome are fixed to probes on glass slides (array capture (Hodges *et al.*, 2007)) or magnetic beads (in-solution capture (Gnirke *et al.*, 2009)). The desired DNA fragments bind to the probes and can be amplified. As the amplification is based on prior knowledge of the desired genome, it can only amplify what was previously fixed to the probes. DNA sequences contained in ancient samples but not in modern ones are subsequently not amplified (Khan *et al.*, 2013). Nonetheless, the amount of endogenous DNA is increased. However, this amplification of the target DNA fragments is no purification step, and therefore the resulting sample is still a metagenomic one. Despite this issue, these capture methods are



Newick representation: (((A:0.5,B:0.5),C:1),(D,E))

Figure 2.2: Example of a rooted phylogenetic tree with five taxa (*A* to *E*) and the corresponding tree in the Newick representation. Two examples of clades are highlighted. The branches leading to the taxa *A* and *B*, as well as branch leading to their common predecessor are only half as long as the rest of the branches.

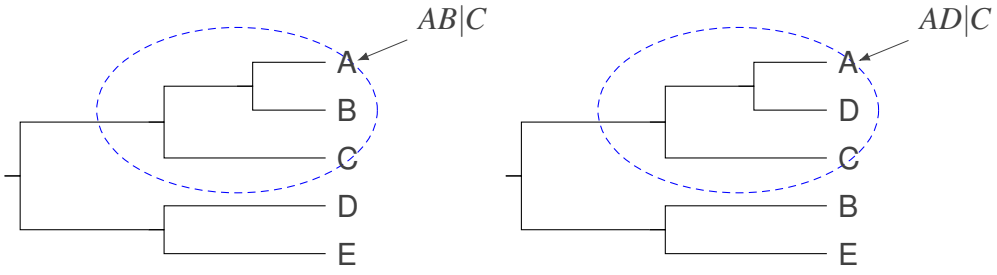
used in many aDNA projects to increase the success of sequencing the desired organism's genome from ancient samples (Shapiro and Hofreiter, 2014).

It is also important to note that these capture methods are not only used in the context of aDNA but also when sequencing clinical pathogens that are hard to cultivate (Mothershed and Whitney, 2006). One such example is the sequencing of *Treponema pallidum* samples by Arora *et al.* (2016).

2.3 Phylogeny

The phylogenetic trees that are known and used today are based on the evolution theory of Darwin (1859), who already used the taxa at the leaves of the tree and hypothetical ancestors as internal nodes (Morrison, 2012). An early evolutionary tree was published by Lamarck (1809), who used taxonomic groups for both the leaves and internal nodes. Thus, his trees represented transformations between taxonomic groups, which does not match the current representation of a phylogenetic tree. An example of a current rooted phylogenetic tree is shown in Fig. 2.2.

The different extant species for which the phylogenetic tree is reconstructed are assigned to the leaves of the tree (Hall, 2011). The relationships of these species are described by inferred, hypothetical ancestors in the inner nodes. Closely related species are clustered together into *clades*. All species in a clade share the same *least common ancestor* (also called *lowest common ancestor*), the internal node that separates all species



(a) Tree 1 (b) Tree 2

Figure 2.3: Example trees used for the input for explanation of the triplet and quartet distances. One triplet decomposition is shown for each tree.

in a subtree from the rest of the tree (Aho *et al.*, 1976). In Fig. 2.2 on the preceding page, two possible clades are highlighted. It is important to note that every node in the phylogenetic tree represents its own clade. The clade defined by the root node contains all species, whereas the leaf nodes define their own clades of only one species.

The length of the branches reflects the evolutionary distance between the species. Even though it is assumed that the evolutionary speciation was and is a binary process (Hall, 2011), it is possible that an internal node contains more than two children. This is called a multifurcation. In addition to rooted phylogenetic trees, which try to reflect the evolutionary path of the speciation, unrooted phylogenetic trees only show the relational connection between the species (Swofford *et al.*, 1996). The phylogenetic trees contained in this thesis focus on rooted phylogenetic trees.

The current standard representation of phylogenetic trees (Cardona *et al.*, 2008) is the *Newick* file format (Olsen, 1990). Its name is based on Newick's seafood restaurant in Dover, New Hampshire, USA, where six scientists created it during an informal meeting. It is a linear representation of the tree using commas and parentheses for the separation and clustering of the nodes. The Newick representation of the tree shown in Fig. 2.2 on the previous page is depicted below the tree.

To compare different trees, one possibility is the quartet distance, which was published by Estabrook *et al.* (1985), or the triplet (triples) distance, published by Critchlow and Pearl (1996). Both distances enumerate all possible subtrees of size four or three and count the number of subtrees both analyzed trees share (Sand *et al.*, 2014). The quartet distance does not account for the rooting of a tree, whereas the place of the root is considered in the triplet distance. The example trees used for the explanation of the triplet and quartet distances are shown in Fig. 2.3. Table 2.1 on the next page shows the triplet and quartet decompositions, as well as their corresponding distances based on the trees shown in Fig. 2.3. Even though only two leaves were switched between the two clades, the triplet and quartet distances differ significantly. However, in larger trees, the significance of these small changes decreases drastically (Sand *et al.*, 2014).

Table 2.1: The triplet and quartet decompositions and distances of the trees shown in Fig. 2.3 on the facing page

	triplet			quartet		
	subset	Tree 1	Tree 2	subset	Tree 1	Tree 2
decomposition	ABC	$AB\|C$	$AC\|B$	$ABCD$	$AB\|CD$	$AD\|CB$
	ABD	$AB\|D$	$AD\|B$	$ABCE$	$AB\|CE$	$AC\|BE$
	ABE	$AB\|E$	$A\|BE$	$ABDE$	$AB\|DE$	$AD\|BE$
	ACD	$AC\|D$	$AD\|C$	$ACDE$	$AC\|DE$	$AD\|CE$
	ACE	$AC\|E$	$AC\|E$	$BCDE$	$BC\|DE$	$BE\|CD$
	ADE	$A\|DE$	$AD\|E$			
	BCD	$BC\|D$	$B\|CD$			
	BCE	$BC\|D$	$BE\|C$			
	BDE	$B\|DE$	$BE\|D$			
	CDE	$C\|DE$	$CD\|E$			
subsets		10			5	
distance		9			5	
normalized distance		$\frac{9}{10} = 0.9$			$\frac{5}{5} = 1$	

2.4 Analysis of NGS data

Since the emergence of NGS sequencing data, various algorithms and analysis pipelines that can be used to analyze them have been published. Currently, the PubMed database of NCBI lists $44,701$ [1] publications containing the term "next-generation sequencing". To reconstruct a genomic sequence out of NGS sequencing data, the reads can be aligned to an already known reference genome (mapping assembly, see Section 2.4.2 on the next page), or they can be reconstructed *de novo*, where overlaps in the read data are used to combine them and create longer sequences (*de novo* assembly, see Section 2.4.3 on page 15). In either case, the quality of the underlying data needs to be assessed. If necessary, the low-quality reads or bases are removed before further analysis (see Section 2.4.1).

2.4.1 Preprocessing

There are several factors that must be considered when assessing the quality of sequencing data. The Illumina sequencing machines always sequence reads of a fixed maximum

[1] `https://www.ncbi.nlm.nih.gov/pubmed/?term=next-generation+sequencing` (Date accessed: May 29, 2019)

length[2]. When short fragments are sequenced, which is the case in most aDNA studies, the fragments can be shorter than the sequenced read length. Thus, the resulting read also contains parts of the ligated adapter. These adapter sequences need to be removed before the data can be used for further analyses. Another problem stems from the declining accuracy of the Illumina sequencing machines towards the 3' ends of the reads (Nakamura *et al.*, 2011). The Illumina sequencing machines report the quality score of each base as a Phred quality score (Ewing *et al.*, 1998):

$$Q = -10\log_{10}(P) \tag{2.1}$$

where P is the probability the base was called incorrectly. A program for the automatic quality assessment of the reads is `FastQC` (Andrews, 2010). It analyzes the reads and generates a report containing a visual representation of the Phred scores for each position of the reads. It also scans the reads for the presence of different adapter sequences and provides further metrics for assessing the quality of the sequenced reads.

The low-quality bases at the end of the reads are often trimmed before further processing. In the studies presented in this thesis, the threshold for the minimum quality has been set to 20 (99% base call accuracy), but other studies go up to 30 (99.9% base call accuracy) (Schmid *et al.*, 2003). Programs that can clip the adapters, trim the low-quality reads, and merge overlapping forward and reverse reads are, for example, `Clip&Merge` (Peltzer *et al.*, 2016) and `AdapterRemoval` (Schubert *et al.*, 2016). The `FastX` toolkit (Gordon and Hannon, 2010) also provides the functionality to clip the adapters and trim the low-quality reads, but both steps must be called manually, whereas the other two programs can run all steps automatically.

2.4.2 Mapping assembly

The advances in sequencing technologies (see Section 2.1) led to the publication of full genome sequences of many species. Currently there are 42,553 full genome sequences available in the NCBI genome database[3]. If a closely related genome has already been sequenced, it can be used as a reference genome for the alignment of the sequenced reads (Pop, 2009). Programs for the efficient alignment of NGS reads are, for example, `SSAHA2` (Ning *et al.*, 2001), `Bowtie2` (Langmead and Salzberg, 2012), and `BWA` (Li and Durbin, 2009; Li, 2013).

After the reads are aligned against the reference genome, programs like `GATK` (Genome Analysis Toolkit) (McKenna *et al.*, 2010) or `samtools mpileup` (Li, 2011) can calculate a consensus call for each reference base. The resulting calls are typically stored in the `VCF` file format (Danecek *et al.*, 2011). After this, the new reconstruction of the genome

[2] `https://www.illumina.com/systems/sequencing-platforms.html` (date accessed: May 29, 2019)

[3] `https://www.ncbi.nlm.nih.gov/genome/browse/#!/overview/` (date accessed: May 29, 2019)

can be generated based on the consensus of each respective position of the reference genome.

A computational pipeline that can reconstruct genomes from raw sequencing data using the mapping-based assembly approach is the EAGER pipeline (Efficient Ancient GEnome Reconstruction) (Peltzer *et al.*, 2016). It follows the GATK *Best Practice's guidelines* (Van der Auwera *et al.*, 2013) and is thus not only applicable to the reconstruction of ancient genomes, but can also be used for the reconstruction of modern samples. EAGER has, for example, been intensively used for the reconstruction of genomes of several modern clinical samples of *T. pallidum* (Arora *et al.*, 2016).

The general steps of EAGER are as follows: After the preprocessing, such as adapter trimming and low-quality base clipping, the resulting reads are mapped against a prechosen reference genome. A de-duplication step is performed to remove mapped reads that stem from the same PCR duplicated fragment. Then, the quality of the mapping is assessed, and a genotyping program calculates possible single nucleotide variations (SNVs). Finally, it is possible to generate a new genome based on the coordinates of the reference genome and the genotyping calls. Additionally, EAGER can estimate the complexity and possible contamination of the sequencing experiment, as well as search for aDNA specific damage patterns.

The big advantage of the reconstruction of a genome using mapping-assembly is the usage of a shared coordinate system (the reference genome) (Pop, 2009). The reconstructed genomes can be compared directly, as they are all of the same length. Additionally, the SNVs identified in the different samples can be compared directly.

2.4.3 *De novo* assembly

Using mapping-based assembly approaches, it is possible to identify SNVs and small indels, if a closely related reference genome is available. However, it is not possible to identify larger insertions, deletions, or genomic rearrangements. Using a *de novo* assembly of the reads, it is possible to identify these variations (Pop, 2009). Furthermore, if no closely-related reference genome is available, the respective genome can only be reconstructed using such a *de novo* assembly.

The general idea of a *de novo* assembly is to first identify overlaps in the underlying read data. Based on these overlaps, a graph is generated and longer contiguous sequences, called contigs, are generated. There are two main strategies for the construction of the graph and the generation of the contigs, the *de Bruijn* graph and the overlap-layout-consensus (OLC) approach (Li *et al.*, 2012).

For the *de Bruijn* graph approach, the sequenced reads are first split into all possible sequences of length k, called k-mers (Pevzner *et al.*, 2001). Typical values for k lie between 16 (Carvalho *et al.*, 2016) and 127 (Cha and Bird, 2016), depending on the input read lengths. The quality of the resulting assembly is highly dependent on the chosen value of k (Cha and Bird, 2016). Using short values for k makes it impossible to reconstruct repetitive regions longer than the chosen value for k. However, based on the

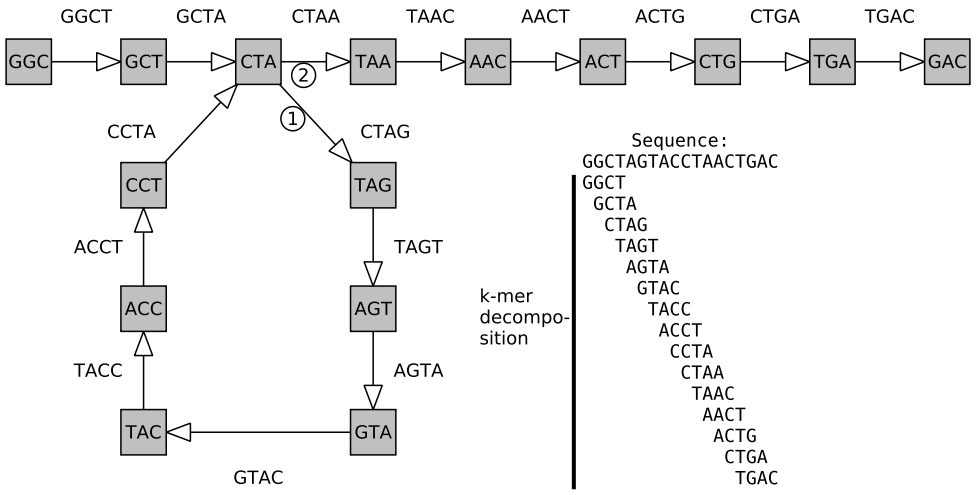


Figure 2.4: An example of a *de Bruijn* graph based on the sequence and its k-mer decomposition depicted on the bottom right (here $k = 4$ was used). It is possible to perfectly reconstruct the original sequence by traversing the Euler path of the graph.

preprocessing and the length of the reads, it is possible that large values for k miss entire reads as they are shorter than the chosen value for k. Furthermore, shorter k-mers help to minimize the influence of sequencing errors. The graph reconstructed based on the k-mers is called a *de Bruijn* graph (De Bruijn, 1946), named after its inventor, the Dutch mathematician *Nicolaas Govert de Bruijn*.

Each k-mer represents an edge in the *de Bruijn* graph, connecting the nodes, which represent the different $(k-1)$ mers. The general idea for the calculation of contigs is to then search for Eulerian paths (visiting each edge exactly once) in the generated *de Bruijn* graph. Figure 2.4 illustrates the k-mer decomposition of a sequence, the resulting *de Bruijn* graph, and an Euler path that can be used to reconstruct the original sequence. Many programs have been developed for the assembly of NGS reads use this *de Bruijn* graph approach. Examples are `VELVET` (Zerbino and Birney, 2008), `SOAPdenovo2` (Luo *et al.*, 2012), and `SPAdes` (Bankevich *et al.*, 2012).

The general idea of the second method for *de novo* assembly, the *OLC* approach, is to generate an overlap-graph of the input reads and then search for a path in this graph that visits each vertex exactly once, also called a Hamiltonian path (Libura, 1991). Furthermore, Myers (2005) proposed the fragment assembly string graph, a modified version of the overlap graph where the problem is not the identification of a Hamiltonian path, but an Eulerian path. Later, Simpson and Durbin (2010) showed that it is possible to identify all pairwise overlaps of the reads in linear time. They make use of suffix arrays (Manber and Myers, 1993) and the Ferragina and Manzini (FM) index (Ferragina and Manzini, 2000). This algorithm was implemented in the `String Graph Assembler` (`SGA`) (Simpson and Durbin, 2012). Other programs that use this approach

are for example `Edena` (Hernandez *et al.*, 2008), `LEAP` (Dinh and Rajasekaran, 2011), and `Readjoiner` (Gonnella and Kurtz, 2012). No matter which assembly approach is used, the *de novo* assembly of read data is NP-complete (Myers, 1995; Medvedev *et al.*, 2007).

It is rarely possible to generate one contig that represents one complete chromosome or genome. One reason for this is the presence of repetitive regions. If the repetitive region is longer than the length of the reads, it is not possible to know how long this region is supposed to be. Because of this, a *de novo* assembly generates multiple contigs.

Subsequently, these contigs need to be sorted based on their order in the genome, which is called *scaffolding* (Boetzer *et al.*, 2011). To this end, and sometimes also to estimate the distance between contigs, mainly paired-end and mate-pair sequencing is used.

As the correct sequence of a *de novo* assembly is generally unknown, the comparison between different assemblies relies on the idea of assembling a whole genome, or each chromosome, as one contig (Bradnam *et al.*, 2013). Thus, a general comparison metric is the mean or median contig length, the longest contig, and the number of contigs in the assemblies. However, these metrics should be compared separately, as a combination of them can provide more information about the quality of the assemblies. The idea is to generate an assembly with very few contigs, resulting in large mean or median contigs, as well as a large maximum contig size. Thus, depending on the length distribution, an assembly with one very long contig and many tiny ones, resulting in an average or small mean and median contig size, might be preferable to an assembly with fewer contigs of average length, even if the mean and median lengths of this second assembly are larger.

To get a better idea about the quality of the assembly, the *N50* value, which can be regarded as a weighted median of the contig lengths, is often used (Bradnam *et al.*, 2013). It is calculated based on the assembly length, meaning the sum of the length of all contigs in the generated assembly. The contigs are first sorted according to their lengths. Then their lengths are summed up starting with the longest contig until the length exceeds half of the assembly length. The length of the contig responsible for exceeding half the assembly length is the N50 value. There are variations of this metric, like the *N70* or *N90*, where the threshold for choosing result is not half the contig length (*N50*), but 70% and 90% respectively. If the approximate size of the assembled genome is known, another variation of the *N50*, the *NG50* can be used. It is not based on the assembly length, but the length of the desired genome.

Finally, if a closely related reference genome is known, it is possible to use it as a gold standard for the comparison of the different assemblies. The generated contigs are mapped against this genome, which allows for a direct comparison of the generated assemblies using metrics like the extent of the genome covered, what percentage of the contigs can be mapped, and is the genome present only once or multiple times. This also allows for the identification of larger genomic arrangements, deletions, or insertions, which are hard to identify using only mapping-based approaches.

Chapter 3

MUSIAL: Postprocessing of multiple samples

Parts of this chapter were submitted to:

***A. Seitz**, A. Herbig, K. Nieselt (2018)*
MUSIAL - Multi sample variant analysis
International Symposium on Biomolecular Archaeology (ISBA) 2018

as well as published in:

V. Schünemann, C. Avanzi*, B. Krause-Kyora*, **A. Seitz***, *et al. (2018)*
Ancient genomes reveal a high diversity of Mycobacterium leprae in medieval Europe

PLOS Pathogens 14(5), e1006997

* Joint First Authors.

3.1 Introduction

The reconstruction of genomes using a mapping-based assembly allows for fast and accurate reconstruction of closely related and thus highly similar genomes (Pop, 2009). However, even highly similar individuals can have small indels in their respective genomes. One example is the bacterium *Treponema pallidum*. Arora *et al.* (2016) published a study analyzing and reconstructing many different strains of *T. pallidum*. In one of the samples, called *IND1*, a deletion of the gene *TP1030* was detected. This deletion is also present in all other samples belonging to the *Yaws* family (Arora *et al.*, 2016).

Using mapping-based reconstruction methods, there are no reads that map to deletions of the newly sequenced sample. Therefore, it is difficult to differentiate between deletions and poorly sequenced regions automatically, leading to deletions often being represented as unresolved bases (N) in reconstructed genomes. This situation also arises when there is not enough coverage or conflicting information (Pop, 2009). For the direct comparison of the genomes reconstructed by mapping-based assemblies, this does not have a negative influence on the downstream analyses, as there is no wrong information added to the result (i.e., it is not known if any a base should be at these specific positions),

except for the length of the reconstructed genome. More significant problems arise from insertions present in the newly sequenced sample. At insertion sites, the sample contains bases that the reference does not have. To compare the different reconstructed genomes directly, they need to be of the same length. Because of this, the insertions in the newly reconstructed samples must be ignored. In contrast to deletions, in the case of insertions, actual information is missing from the mapping-based reconstructed genome that may be important.

While it is not impossible to reconstruct the genomes including all indels, it is very complicated. The genomes reconstructed with indels are all of different length, which is why they are no longer directly comparable. However, it would be possible to generate a whole genome alignment (WGA) with all the reconstructed genomes using for example `progressiveMAUVE` (Darling *et al.*, 2010). The resulting WGA could then constitute the basis of a data structure like the SuperGenome (Herbig *et al.*, 2012) to create a consensus sequence incorporating all input genomes. This consensus sequence then contains every sequence that is present in each sample. Based on this consensus sequence, it would then be possible to create genome reconstructions for each sample accounting for all indels that are present in all samples. However, it is still very hard to align multiple sequences (Liu *et al.*, 2010). While there are programs that can align short sequences like proteins, the alignment of whole genomes is another matter. In the *Alignathon* (Earl *et al.*, 2014), which focused on accuracy instead of runtime, for example, many programs could not complete their analysis on all of the provided datasets due to the required runtime.

To still be able to directly account for indels in genomes reconstructed by mapping-based approaches, we developed the Java program `MUSIAL` (MUlti Sample varIant AnaLysis). `MUSIAL` is based on the concept of the `MultiVCFAnalyzer` (Bos *et al.*, 2014), which was adapted, extended further, and submitted as `MUSIAL` to the International Symposium on Biomolecular Archaeology (ISBA) 2018.

The idea is to analyze all samples belonging to a common project simultaneously, which allows for the identification of all SNVs between all samples and the generation of comparative summaries of these SNVs. These summaries can be extended to contain annotations for all identified SNVs in a project using programs like `SNPeff` (Cingolani *et al.*, 2012). `MUSIAL` is a new implementation, based on the concepts of the `MultiVCFAnalyzer`. This allows the analysis of not only the `Unified Genotyper` (discontinued since 2017), but also the `Unified Haplotyper`, which are both implemented in `GATK` (McKenna *et al.*, 2010). Though the `Unified Genotyper` has been discontinued, `MUSIAL` can still analyze its output. The introduction of the `Unified Haplotyper` also allowed for the identification of indels. Because `MUSIAL` analyzes all samples simultaneously, it can reconstruct different genome alignments based on an extended reference genome, including all indels, similar to the consensus sequence of a WGA. These alignments can then be used for the calculation of phylogenetic trees.

For the correct placement of the root of a phylogenetic tree, an outgroup sample is needed (Graham *et al.*, 1998; Wheeler, 1990). While it is possible to treat this outgroup

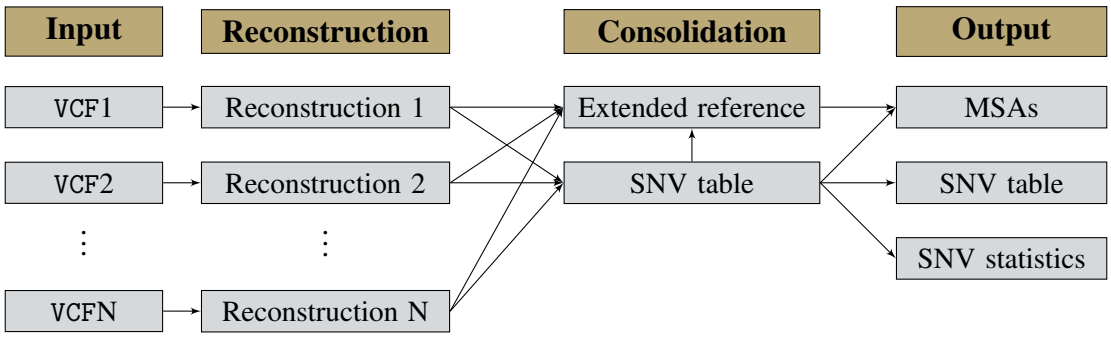


Figure 3.1: Workflow of MUSIAL. The input VCF files are first analyzed. Next, they are combined in a table containing all SNV positions of all samples, as well as an MSA based on an extended reference, containing all indels. These are then combined to generate different output files.

sample in the same way as the other samples, its evolutionary distance to the other samples can lead to problems like *long branch attraction* (LBA) (Felsenstein, 1978; Williams *et al.*, 2015). To reduce this risk, as well as the number of aligned positions, it is often preferred to use only the variant positions that are present in at least one of the samples for the calculation of the desired phylogeny.

3.2 Methodology

The general idea of MUSIAL, as illustrated in Fig. 3.1, is to analyze and reconstruct the genomes of all NGS-samples of a common project simultaneously. This reconstruction is the same as when reconstructing each sample on its own. However, before the final reconstructions are generated, all variations contained in all samples are extracted. These variations can then be summarized in a SNV table. This table contains each genomic position in the reference genome where at least one sample differs from the reference genome. As these variations can also include indels, it is possible to generate an extended reference that contains gaps corresponding to insertions contained in the different samples. This extended reference makes it possible to include these gaps not only in the reference genome but also in the newly reconstructed genomes. Thus, it is possible to generate a WGA including indels without the need for a separate calculation of the individually reconstructed genomes. This also holds true for multiple sequence alignments (MSA) based on genes or variations only.

3.3 Implementation

MUSIAL is implemented in Java and available on GitHub[1]. It uses the HTSJDK [2] Java library to parse the VCF files generated by GATK. For this, the output of GATK must contain either a call for each reference position or be in the GVCF format, where it contains the non-variant sites in condensed blocks. MUSIAL analyzes each sample and generates a base-call in the IUPAC code corresponding to each position in the reference genome. The default parameters to generate a confident base-call is a minimum coverage of five and a minimum frequency of the base of 90% at each position. The user can adjust these parameters. If either is not met, the corresponding position is called as an unresolved base (N). Furthermore, to allow for sequencing errors in low coverage regions, the minimum allele frequency is relaxed if the minimum coverage criterium is met and only one read differs from the rest of the reads for that position.

MUSIAL can also call heterozygous positions if the frequency of the corresponding base-call lies between two adjustable thresholds. The default thresholds for this is a base frequency between 45 and 55%. To call a heterozygous position, all relevant bases need to be above the coverage threshold. Thus, the minimum coverage for a heterozygous position is two times the minimum coverage for a homozygous position. Fig. 3.2 on the next page illustrates the general idea of this method by an example.

To resolve bases in low-coverage regions, it is possible to set an additional, lower coverage threshold for the generation of a genotype call. Regions that lie above the standard coverage threshold are still treated as before, and regions below this additional coverage threshold are also called as unresolved. However, regions that have a coverage between the two thresholds are treated differently. If all reads support a reference call for a position, it is called as the reference base and not as unresolved. If all reads support a SNV, it is called as that SNV if that specific SNV was already called in at least one other sample.

MUSIAL requires as input the reference genome file against which the mappings and genotyping were performed and an output directory, which will be created if it does not exist. There are three ways to pass the input files to MUSIAL. It is possible to add them as space-separated list directly in the command line or as a file where each line corresponds to a path of one of the input files. The name for each sample will be the name of the directory in which the respective file is stored. Another method is to use the output of a single- or multi-EAGER run directly as input. To be able to handle the samples in the same manner as the other two methods, MUSIAL creates a new directory named VCFs in the output directory. For each sample, a subdirectory with the same name as the corresponding sample name, containing a symbolic link to the corresponding VCF file will be generated. Finally, to run MUSIAL again with the same input files, a txt file containing the path to all used input files of the analysis will be created and stored in the

[1] https://github.com/Integrative-Transcriptomics/MUSIAL
[2] https://github.com/samtools/htsjdk

Ref	T	A	G	A	T	G	G	A	C	G	C	A	G	T	A	G	G	G	T
Mapped reads	T	A	G	A	G	G	G	A	C	T									
			G	A	G	C	G	A	C	G	C	A							
				A	G	C	G	A	G	T	C	A	G	T					
					G	C	G	A	G	G	C	A	G	T	A	G	G		
							G	A	G	T	C	A	G	T	A	G	G	G	
									C	T	C	A	G	T	A	G	G	G	T
Cov	1	1	2	3	4	4	5	5	6	6	6	5	5	4	3	3	3	2	1
Base call	N	N	N	A	G	C	G	A	S	N	C	A	G	T	A	G	G	N	N

Figure 3.2: Example of how `MUSIAL` calculates base-calls (in IUPAC code), based on a mapping. The first row represents the reference, lines 2 to 7 show the mapping of reads against the reference. "Cov" depicts the coverage of each position and the bottom row the sequence reconstructed by `MUSIAL` in the IUPAC code. Here, a minimum coverage of 3, a genotype frequency of 90%, and a frequency between 45 and 55% for a heterozygous call was assumed. The first and last three bases are called as "N" because they do not meet the coverage threshold of 3. Bases marked in red in the bottom row are positions, where a base is called that is different from the reference and above the coverage threshold. The first one (G) is consistent because it is backed up by all reads. The next one (C) falls below the 90% frequency threshold. This call is still made because there is only one deviating read. The heterozygous call (S) has a frequency of 50% and both the call for C and the one for G are above the coverage threshold. The following *N* call has a frequency of 66.6%. It falls neither in the homozygous call of over 90% nor in the heterozygous call of between 45 and 55%.

output directory. This file can be used directly as input for the next analyses.

Each input file is then analyzed separately, to reconstruct the corresponding genome. Since for this part the files do not depend on each other, they can be analyzed simultaneously on multiple threads. For each sample, a base call in the IUPAC format is generated for each position of the reference genome if it fulfills the quality criteria, which can be adjusted by the user. The call of heterozygous positions is disabled by default. If a position does not meet any of the quality measures, it is called as an unresolved base (N).

By default, the execution of `MUSIAL` saves several files. To differentiate between the different sequences of the reference (multiple chromosomes, plasmids, and so on), the name of the reference genome, as stored in the reference `FASTA` file, is used as a prefix for all output files corresponding to this sequence.

An overview of all samples is written to the file with the suffix `snvStatistics`. Table 3.1 on the following page shows an example of this file. It contains general statistics, like the number of input samples, the coverage threshold, and the used allele frequency.

Table 3.1: Example for the table containing the SNV statistics for each sample, as well as general parameters used for the calculation.

SNV statistics for 169 samples
Coverage Threshold: 5.0
Minimum SNV allele frequency: 0.9 (90%)

sample	SNV Calls (all)	SNV Calls (het)	Coverage (fold)	Coverage (%)	Reference Calls	total Calls	no Calls
ARLP 30	103	0	115.246	99.98	3,189,048	3,268,203	79,052
ARLP-12	102	0	93.904	99.97	3,188,041	3,268,203	80,060
ARLP-23	111	0	100.827	99.98	3,188,442	3,268,203	79,650
ARLP-29	135	0	93.795	99.97	3,187,960	3,268,203	80,108
ARLP-49	138	0	85.487	99.98	3,188,356	3,268,203	79,709

Furthermore, it contains sample-specific statistics like the overall number of SNVs, the number of heterozygous SNVs, together with the percentage of the genome that has at least a coverage of the minimum coverage parameter, the number of reference calls, the number of total calls, and the number of positions, where no call could be made.

A detailed description of each SNV position and the corresponding base-call for each sample is summarized in the file with the suffix `snvTable`. An excerpt of this table can be seen in Table 3.2 on the next page.

Two alignment files with the suffixes `snvAlignment` and `genomeAlignment` contain the alignment of all variant positions and every position of the reference genome, respectively. Additionally, it is possible to generate alignments of specific genes. The names of the genes must match the names in the `GFF` annotation file, which also needs to be provided for the calculation of gene alignments. `MUSIAL` then creates a file containing the alignment for each respective gene, using the name of the gene as a suffix for the corresponding output filename.

`MUSIAL` can also annotate all identified SNV positions using `SNPeff` (Cingolani *et al.*, 2012). For this, the `GFF` file corresponding to the reference is also required. `MUSIAL` tries to download the current version of `SNPeff`. If this is not possible, a packaged version of `SNPeff` is extracted from the `JAR` file of `MUSIAL`. `MUSIAL` always generates a new `SNPeff` database based on the provided `GFF` file and then annotates the identified SNV positions using this database. It generates an output file with the same prefix as the input file of `SNPeff` and the suffix `snpeff_output`. `MUSIAL` again parses this file and generates a new SNV table with the annotations from `SNPeff`. This file has the same prefix again as the output file of `SNPeff` and the suffix `snvTableWithSnpEffInfos`.

Furthermore, `MUSIAL` can compute a list of positions that have a low coverage in multiple samples. For this, all positions where at least one sample has coverage lower than the minimum additional call coverage parameter, or lower than the mean cover-

Table 3.2: Example of the table containing all SNVs. The first column is the position of the SNV based on the reference genome, the second column is the reference base and the following columns are the samples with their respective SNVs (in IUPAC code). A dot denotes the reference base.

Position	Ref	ARLP 30 Ethiopia 2015	ARLP-12 Ethiopia 2015	ARLP-23 Ethiopia 2015	ARLP-29 Ethiopia 2015	ARLP-49 Ethiopia 2015	⋯
1113	C	.	.	.	T	.	⋯
2251	C	.	.	.	.	T	⋯
4513	G	A	.	A	.	.	⋯
4972	G	.	A	.	.	.	⋯
6864	C	.	T	.	.	.	⋯
8453	T	C	C	C	C	C	⋯
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱

age minus the standard deviation of the coverage are written to a file with the suffix `lowCoveragePositions`, together with the corresponding sample names. Furthermore, because the coverage for each position is known from the reconstruction process, it is possible to identify all uncovered positions for each sample.

Variant positions that are known to be present in negative controls or may stem from contamination should be removed from the SNV alignment prior to downstream analyses, such as phylogenetic tree reconstructions. These contaminations could lead to a false clustering of the affected taxa in the tree (Laurin-Lemay *et al.*, 2012). Another problem in the generation of phylogenetic trees is the placement of the root. For this, `MUSIAL` can add an outgroup sample containing only the positions that contain a variation in at least one of the other samples.

If the multithreaded version of `RAxML-NG` (Stamatakis, 2014; Kozlov, 2017) is installed on the server running the analysis, `MUSIAL` can automatically calculate a maximum likelihood phylogeny using the SNV alignment. `RAxML-NG` generates several files. They all have the same prefix as the SNV alignment. The final tree has the suffix `bestTree` and is written in the `Newick` file format. To get a better idea which SNV positions are more conserved and are thus probably responsible for splits in the tree closer to the root, `MUSIAL` can calculate shared allele frequencies for each SNV position. This calculates the frequency of the major allele for each variant position in the reference genome, based on the base calls of the different samples. The result of this analysis is stored in the file with the suffix `alleleFrequencies`.

Finally, if desired, it is possible to compare the result of the current run to a previously generated SNV table and calculate the differences between the two. This comparison allows for a fast identification of new SNV positions when for example new samples have

Table 3.3: Example of the file containing the differences compared to another SNV table. The first column describes the type of difference. A "+" describes a SNV position that is present in this SNV table but was not present in the other run. A "-" describes the exact opposite. An "=" describes a SNV position that is present in both cases, but with a difference in one of the samples.

Difference	Position	Sample	Current	Other
=	1011	Br14-3 Brazil 2014	A	
=	6736	Br2016-15 Brazil 2016	A	N
+	7386	Br14-3 Brazil 2014	T	
-	9834	Jorgen 625 Denmark 1283-1329		G

been added to a project. The differences are written to a file with the suffix `differences`. This file contains one row for each difference. Each row contains five columns, separated by tabulators. Table 3.3 shows an example excerpt of this file. The first line of this excerpt shows that the corresponding SNV (position 1011) was already present in the other run, but no call was made for this sample. This observation is only possible if this sample was not present in the previous analysis. The next line shows a previously unresolved SNV that could now be resolved. The final two represent SNVs that are only present in one of the two runs.

All parameters and their respective default values of `MUSIAL` are described in supplementary Table B.1 on page 92.

3.4 Application

`MUSIAL` was used in our publication *Ancient genomes reveal a high diversity of Mycobacterium leprae in medieval Europe*, published in *PLOS Pathogens* (Schuenemann *et al.*, 2018a). The goal of this study was to identify how the bacterium *M. leprae* spread through Europe in the Middle Ages and where it originated. Additionally, the goal was to estimate when *M. leprae* differentiated from other *Mycobacteria* and evolved into the pathogen known today. In total, we analyzed 169 *M. leprae* samples, 17 of which were extracted from ancient human remains, five from modern squirrels, four from modern monkeys, one from a modern armadillo, and 142 from modern humans.

The analysis pipeline of this paper is illustrated in Fig. 3.3 on the facing page. The raw `FASTQ` files of all sequenced samples were first analyzed with the `EAGER` pipeline (Peltzer *et al.*, 2016). After the `FASTQ` quality assessment via `FastQC` (Andrews, 2010), the tool `Clip&Merge` was used to remove sequencing adapters, merge overlapping forward and reverse reads, and remove bases of a Phred score of < 20. The resulting reads were then mapped using `BWA` (Li and Durbin, 2009) against the reference strain *M. leprae TN*. Afterward, the quality of the mappings was assessed via `QualiMap2` (Okonechnikov

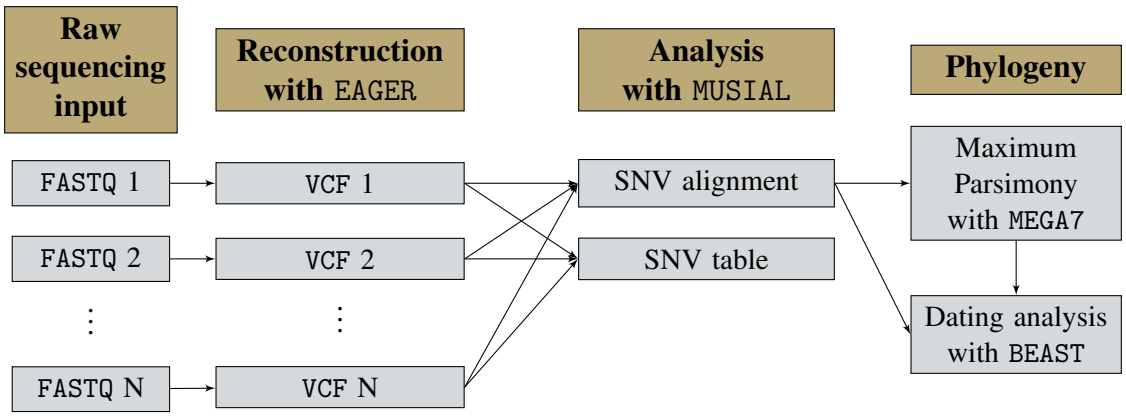


Figure 3.3: Analysis pipeline used in our paper (Schuenemann *et al.*, 2018a). The sequenced samples were first analyzed with EAGER. The output of EAGER was used by MUSIAL to calculate different alignments, which were then used for the calculation of different phylogenies, as well as the dating analysis of *M. leprae*.

et al., 2015). PCR duplicated reads were removed with MarkDuplicates, implemented in the Picard tools [3]. Afterward, the genotypes were called with GATK (McKenna *et al.*, 2010) on the resulting files.

The VCF files generated by GATK were then used to generate different alignments. We analyzed the VCF files with MUSIAL, using a minimum coverage of five and a minimum allele frequency of 90%. This analysis resulted in a total of 3124 different SNV positions, which were taken for the calculation of a maximum parsimony tree with MEGA7 (Kumar *et al.*, 2016). Some of the modern *M. leprae* samples have a mutation in the endonuclease III gene *nth* (Locus ID: ML2301), which leads to a rapid accumulation of SNVs (Benjak *et al.*, 2018). Because all of these samples are resistant to antibiotics, these mutations are probably due to their adaption to the antibiotic treatment. Thus, the high number of SNVs are not based on natural evolutionary pressure, but human-introduced additional evolutionary pressure. To not include this bias in the dating analysis of the evolutionary divergence estimation, these samples were removed from the analysis, as they would have skewed the overall mutation rate of *M. leprae*. The analysis without these samples resulted in 2371 unique SNV positions. Based on these positions, another maximum parsimony tree was calculated with MEGA7. This tree was then used as a guide-tree in the divergence estimation with BEAST (Drummond and Rambaut, 2007). With the compare option of MUSIAL, it was possible to identify the differences between the run containing all samples and the one without the hypermutating strains. This comparison allowed for a fast verification of the responsible mutations.

To see the difference in the genomic resolution when allowing for a call in lower

[3] http://broadinstitute.github.io/picard/

Table 3.4: The number of unresolved bases when setting the additional coverage parameter to three while leaving the default coverage parameter at five, compared to only setting the default coverage parameter to five without setting the additional coverage parameter.

Method	Unresolved in SNV alignment	Unresolved in genome Alignment
`MUSIAL` without additional coverage parameter	16,091	23,852,738
`MUSIAL` with additional coverage parameter	9,852	17,153,628
improvement	38.22%	28.09%
resolved SNV positions	31	31
resolved reference positions	6,208	6,699,079

coverage regions, the analyses with `MUSIAL` were rerun on the leprosy data with the parameter for the additional coverage set to three while leaving the default coverage parameter at five. Furthermore, the result was compared to the previous result without the additional coverage parameter. Table 3.4 shows the number of unresolved bases in the SNV, as well as in the whole genome alignment generated by `MUSIAL` together

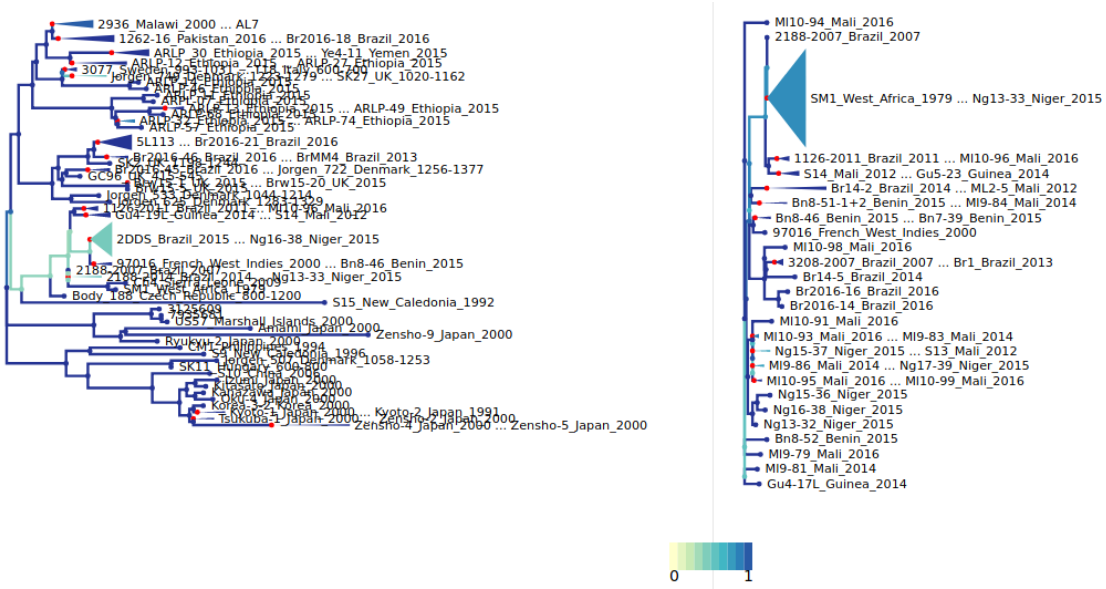


Figure 3.4: Visual comparison of a maximum likelihood tree, calculated with `RAxML-NG` directly through `MUSIAL` and a maximum parsimony tree calculated with `MEGA7`. The colors represent the similarity, the darker the branch, the more similar the subtrees. This comparison was calculated with `Phylo.io` (Robinson *et al.*, 2016)

Table 3.5: Statistical comparison of a maximum likelihood tree, calculated with RAxML-NG directly through MUSIAL and a maximum parsimony tree calculated with MEGA7. This comparison was calculated with tqDist (Sand *et al.*, 2014)

Metric	Value
Number of triplets	790,244
Triplet distance	3,312
Normalized triplet distance	0.00419
Number of quartets	32,795,126
Quartet distance	424962
Normalized quartet distance	0.01296

with the achieved improvement of the number of unresolved bases for both cases. The analysis with the additional coverage parameter resulted in a resolution of almost 40% of the previously unresolved bases in the SNV alignment. In the genome alignment, over 17 million additional bases were resolved with the additional coverage parameter, which is almost 30% of the previously unresolved positions. The vast majority of additionally resolved bases are reference calls (only 31 additionally resolved SNV positions).

Furthermore, the automatic calculation of the maximum likelihood phylogenetic tree in MUSIAL was compared to the calculation of a maximum parsimony tree, calculated with MEGA7. Both trees were calculated on the snvAlignment calculated by MUSIAL. A side-by-side comparison, calculated by Phylo.io (Robinson *et al.*, 2016), is shown in Fig. 3.4 on the facing page. It shows that, while there are differences between the two trees, the general structure of the tree is quite similar.

However, the two trees were not only compared visually but also with the tree comparison metrics described in Section 2.3 on page 11. These results, calculated with tqDist (Sand *et al.*, 2014), are illustrated in Table 3.5. It confirms the observation of the visual comparison, namely that the two trees are very similar.

Additionally, the phylogeny based on a gene alignment calculated by MUSIAL was evaluated. To identify candidate genes for this analysis, MUSIAL was again run with the option to annotate the SNVs with SNPeff. The resulting annotated SNVs were then grouped by their affected genes. Finally, the results were sorted by the number of times a gene was affected by a SNV. An excerpt of the five genes containing the greatest number of SNV positions is shown in Table 3.6 on the next page. The gene with the highest number of variant positions, *ML0411*, codes for an antigen that is recognized by the antibodies in the immune response (Parkash *et al.*, 2006). It is also possible to use it as a marker to test for the presence of the bacterium in suspected leprosy patients. Nothing is known about the gene *ML1750*, which contains the second most SNVs. It is only annotated as a hypothetical (NCBI, 2016) or an uncharacterized (Uniprot, 2018b) protein. The third one in the list, *gyrA*, is involved in the "ATP-dependent breakage, passage and rejoining of double-stranded DNA" (Uniprot, 2018a). Finally, MUSIAL was rerun with the

Table 3.6: Genes of *M. leprae* containing the most SNV positions

Gene name	Number of SNVs
fadD9	14
rpoB	14
gyrA	18
ML1750	19
ML0411	36

option to calculate a gene tree for the gene *ML0411*, which contains the greatest number of SNVs.

The maximum parsimony tree based on the gene *ML0411*, calculated with `MEGA7`, is illustrated in Fig. 3.5. It contains many unresolved subtrees and taxa. However, some differentiation into general subtrees is present. This differentiation is verified by comparing this tree to the maximum parsimony and maximum likelihood trees of the original SNV alignment with the metrics described in Section 2.3 on page 11. Their results, calculated with `tqDist`, are illustrated in Table 3.7 on the facing page. Both the triplet, as well as the quartet distance between the gene tree and the maximum parsimony tree on the SNV alignment share more than 50% of the corresponding triplets and quartets, while the triplet distance is smaller than the quartet distance. The comparison to the maximum likelihood tree shows a different picture. There, the normalized quartet distance is

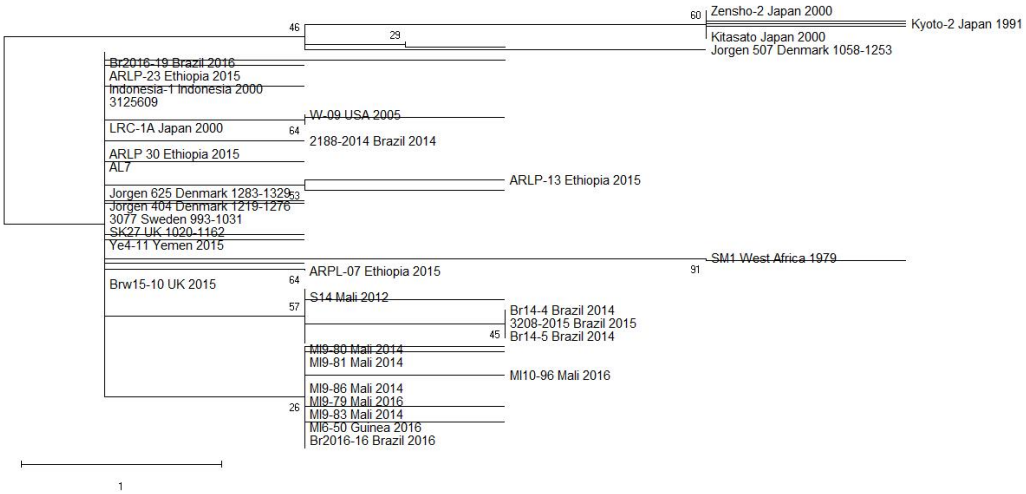


Figure 3.5: Maximum parsimony phylogenetic tree, calculated by `MEGA7`, of *M. leprae* based on the alignment of the gene *ML0411*. The gene alignment was generated with `MUSIAL`.

Table 3.7: Distances between the maximum parsimony tree based on the gene *ML0411* and the two trees (maximum parsimony and likelihood) trees based on the SNV alignment. These results were calculated with `tqDist`

	Metric	SNV alignment	
		Maximum parsimony	Maximum likelihood
ML0411	Number of Triplets	790,244	
	Triplet distance	258,567	426,234
	Normalized triplet distance	0.32720	0.53937
	Number of quartets	32,795,126	
	Quartet distance	14,119,179	14,165,363
	Normalized quartet distance	0.43053	0.43194

smaller than the normalized triplet distance. Both distances are around 50% and larger than the corresponding distance of the maximum parsimony tree.

Additionally, a maximum-likelihood tree using `RAxML-NG` was created based on this gene alignment. However, like the result of the maximum parsimony tree, the taxa could hardly be resolved, which is why only the results of the maximum parsimony tree are shown. Furthermore, as the resulting trees on the gene *ML0411* are already unresolved, the phylogenies on the other gene alignments were not calculated.

3.5 Further developments of `MUSIAL`

For the further identification of shared SNV positions, two bachelor theses built upon the output of `MUSIAL`. The first one used the SNV table, together with the generated phylogeny to identify clade-specific SNVs and SNV positions that do not concur with the calculated phylogeny. This result, together with the annotated SNV positions, was then taken in another bachelor thesis to implement an interactive visual analytics tool for the identification of clade-specific SNVs and other metadata shared by samples in the phylogenetic tree.

The prerequisite to identify clade-specific SNV positions is the phylogenetic tree, as well as the SNV table. To differentiate between the different internal, unlabeled nodes, each node is assigned a unique ID through post-ordering of the nodes. For each variant position in the SNV table, the leaves of the phylogenetic tree (taxa) are labeled with their respective base. These are then propagated to the root, same as the forward-pass of the Fitch parsimony algorithm, which was actually described and published by Hartigan (1973) (Semple and Steel, 2003). In contrast to the backward-pass of the Fitch parsimony algorithm, the tree is now not labeled with the base that minimizes the number of mutations. Instead, nodes, where all children have the same base are identified. Thus, these bases are responsible for the separation of the samples into the respective clades.

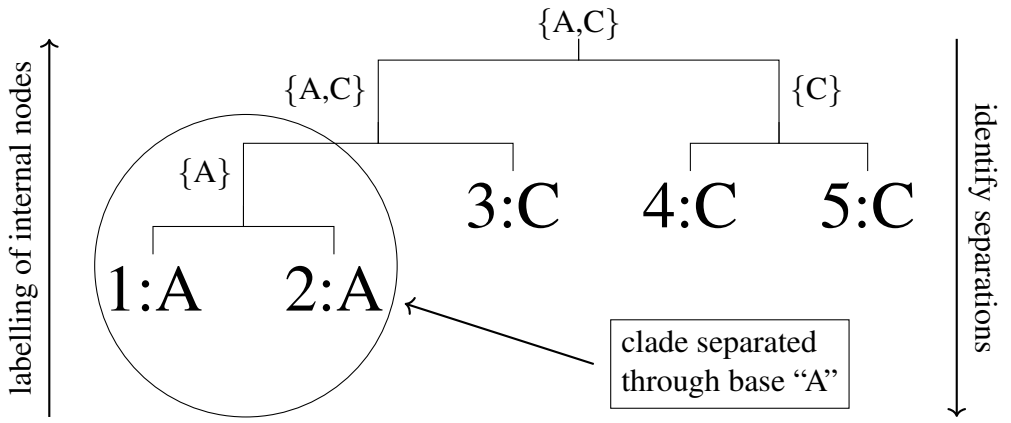


Figure 3.6: Example of a tree labeled with a SNV. The taxa, named 1 to 5, are labeled with a SNV. This SNV is propagated up to the root. Afterward, a subtree containing only a single, common base is identified. Here, the clade of the samples 1 and 2 is separated through the nucleotide “A” from the rest of the tree.

An example of this is illustrated in Fig. 3.6. After all SNVs are processed, each one is classified into supporting (i.e. SNV positions that separate the tree into clades), and non-supporting (i.e. SNV positions which occur in different parts of the tree so that no distinction can be made). An example of a non-supporting SNV would be, if either taxon number 4 or 5 in Fig. 3.6 would also be labeled with an “A”. Then the tree could not be split into two subtrees based on this SNV.

To get an overview of how many SNV positions are in support of a maximum parsimony tree, and how many disagree with the topology, this tool was applied to the maximum parsimony tree of the 169 leprosy samples published by Schuenemann *et al.* (2018a). Table 3.8 summarizes the result of this analysis. The majority (93.97%) of the SNVs support the maximum parsimony topology.

This result, together with the phylogenetic tree and the SNV table is then taken by `EVIDENTE` (Efficient VIsual analytics tool for Data ENrichment in phylogenetic TreEs), an interactive visual analytics tool for the identification of clade-specific similarities. `EVIDENTE` can also use optional metadata about the samples and incorporate them into its interactive visualization. An example of the visualization of the Leprosy samples

Table 3.8: The number of SNV positions that support the maximum parsimony based on the SNV alignment as well as the number of SNVs that do not support the resulting tree topology.

supporting SNVs	non-supporting SNVs
3270	210

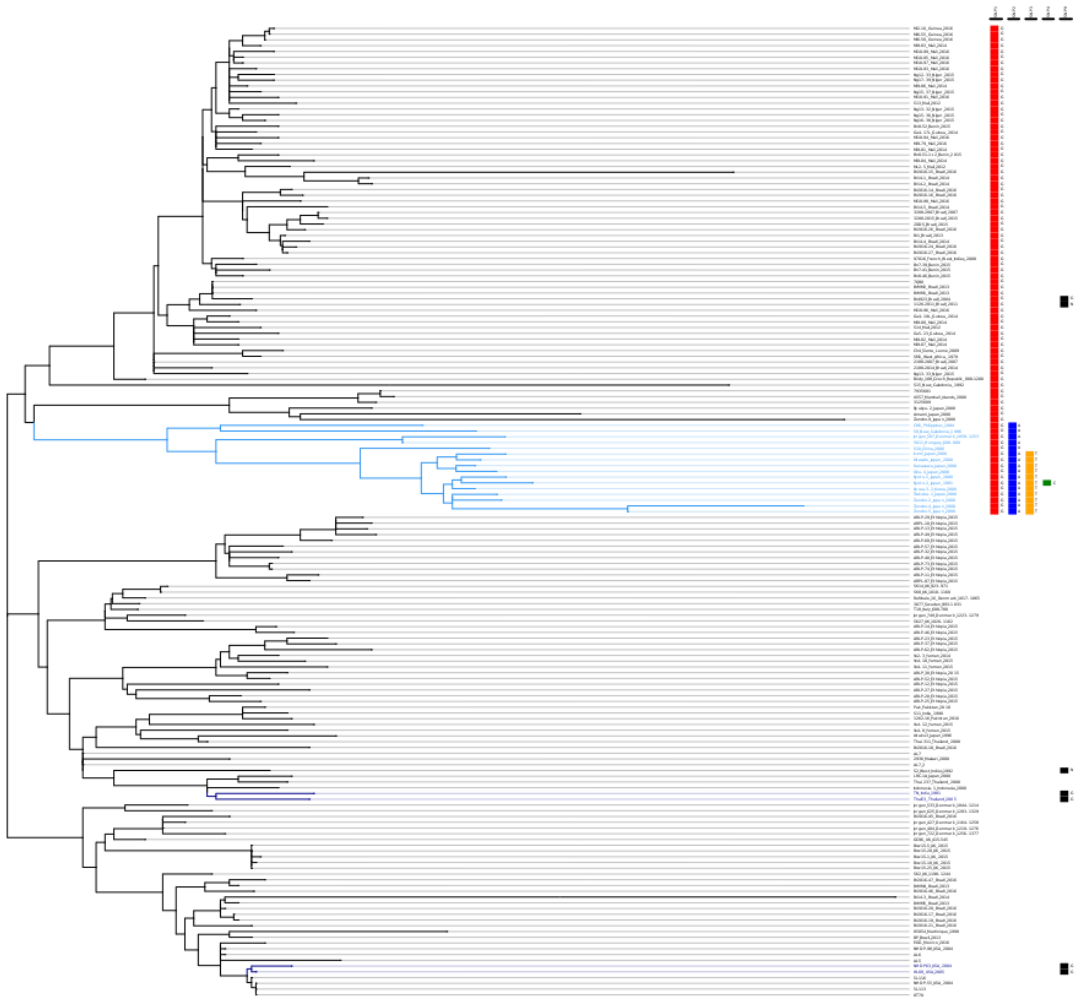


Figure 3.7: Example visualization with EVIDENTE. Four of the five SNV positions visualized on the right confirm the tree topology (colors red, blue, orange, and green, and one position is not conform with the tree topology (black). The highlighted subtree contains a significantly enriched number of samples originating from Japan (see Fig. 3.8 on the next page).

discussed in Section 3.4 on page 26 is shown in Fig. 3.7. It shows the whole tree, with an additional five visualized SNVs. Four of these SNVs are in concordance with the tree. These SNVs are colored according to their variant base: Blue for adenine, orange for thymine, red for guanine, and green for cytosine. SNVs that do not concur with the phylogenetic tree are colored in black. Furthermore, it is possible to test for an enrichment using Fischer's exact test of the provided metadata of the taxa and SNVs. The subtree highlighted in blue in Fig. 3.7 was chosen for the enrichment analysis of the country of origin. The goal was to see if the subtree contains a significantly enriched

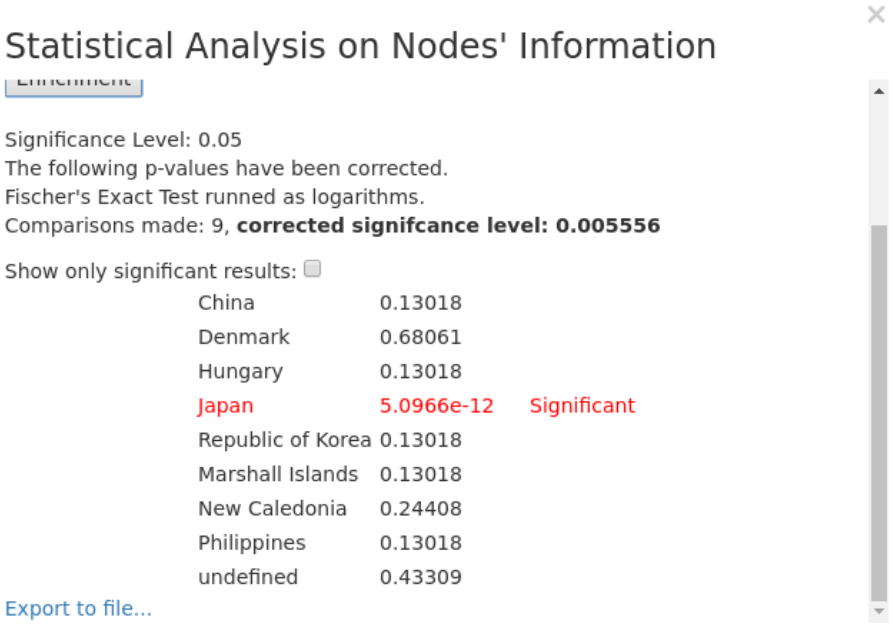


Figure 3.8: Enrichment analysis of the country of origin in the subtree highlighted in Fig. 3.7 on the previous page.

number of samples that were collected in certain countries. The result of this enrichment analysis is shown in Fig. 3.8. While there are samples stemming from six different countries, the subtree contains a significant number of Japanese samples.

In addition to the enrichment analysis on the country of origin, the same subtree was analyzed for an enrichment of mutated genes. Fig. 3.9 on the facing page shows an excerpt of the results of this analysis. There are 56 out of 289 different mutated genes that show significant enrichment for the selected subtree.

Based on these enrichment analyses, it is possible to highlight the taxa that share the desired characteristic. If all samples of a subtree have that characteristic in common, the last common ancestor of those samples is highlighted.

3.6 Discussion and conclusions

`MUSIAL` is a convenient command line tool for the generation of different alignments using mapping-based reconstruction methods. The simultaneous analysis of all samples in a common project allows for a fast and accurate generation of MSAs of whole genomes, genes, or only all SNVs, including indels without the need for a separate calculation of an MSA based on the individually reconstructed genomes. These alignments can be used as input for the calculation of phylogenies. It is possible to automatically construct a phylogenetic tree based on the SNV alignment using `RAxML-NG`.

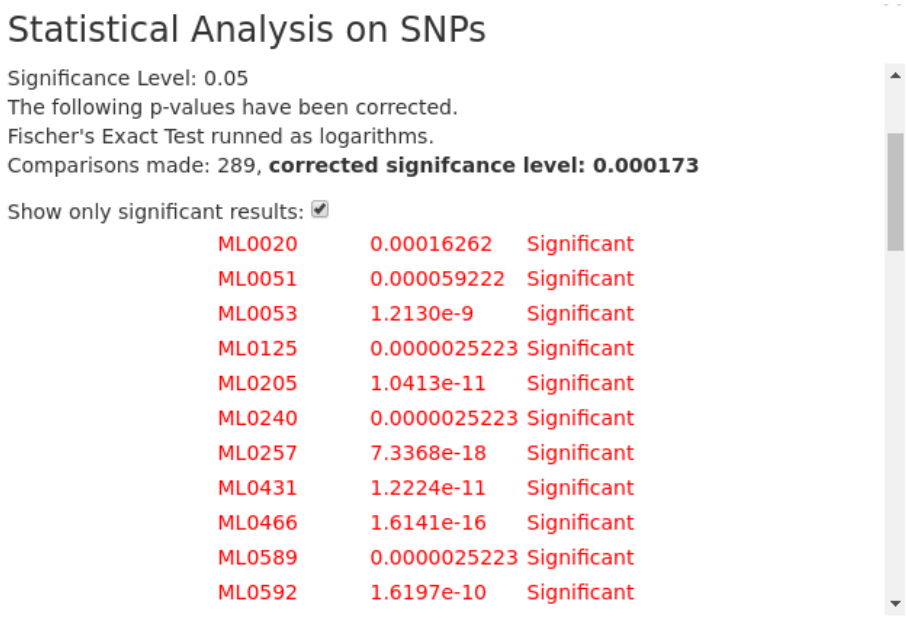


Figure 3.9: Enrichment analysis of the affected genes in the subtree highlighted in Fig. 3.7 on page 33. Only significant results are shown.

The comparison with other results helps to identify new SNV positions, as well as to give an idea of how complete the SNV table already is. If the addition of new samples leads to the identification of many new SNV positions, it is very likely that other new strains will also lead to new SNV positions. However, no or very few newly identified SNV positions may indicate a closed pan-genome (Tettelin *et al.*, 2008).

Low-coverage regions, together with uncovered regions, can help to identify systematic genomic changes or problems in the sequencing data. These changes might be due to samples that are missing specific regions entirely or indicate problems in the sequencing experiment, e.g., that whole regions were sequenced poorly.

The shared allele frequencies can give an indication about closely related samples without the need for phylogenetic analysis. If samples share many SNV positions, there is a high probability that they will also cluster together in a phylogenetic tree.

The additional coverage parameter led to a decrease of unresolved bases by about 40% and 30% for the SNV and genome alignment respectively. Out of those resolved bases, only 31 were SNVs. The rest were reference calls. For the call of these reference positions, the corresponding positions need to have a coverage between three and five, where all reads must support the reference base. Even if these few SNV positions do not increase the resolution of the phylogenetic tree, the additional reference positions can rule out the presence of these SNVs where previously nothing was known. This method is particularly interesting for ancient samples, where the minimum coverage is often set higher because of the metagenomic origin of the sample. This helps to reduce the

amount of wrongly called SNVs. Thus, using this method with the additional coverage parameter, it is now possible to call more reference positions without the fear of calling wrong SNVs.

Both, the maximum likelihood, as well as the maximum parsimony trees that were based on the SNV alignment are very similar. The main differences lie in the ordering of samples that already have a low bootstrap value in the original publication of Schuenemann *et al.* (2018a). Thus, these differences in the trees are due to a lack of resolution in these subtrees. There, the programs are not able to know the exact order and placement but must place the samples somewhere, which results in the differences as there is no guideline on how to place these taxa.

The differences between the tree based on the gene alignment of *ML0411* and the ones based on the SNV alignment is somewhat expected. Even though this gene has a length of 1227 base-pairs, it only contains 36 variant positions. When calculating a tree on 169 samples, it is clear that with only 36 differences between the samples, it is not possible to cluster them without ambiguities. However, the general clusters that arise based on this gene alignment are also present in the phylogenies based on the complete SNV alignment. Thus, it might be possible to identify SNVs within this gene that may allow a prediction of the placement of new taxa.

These similarities are also reflected in the triplet and quartet distances of the trees. The quartet distance is always larger than the triplet distance, which is probably due to the higher number of compared characteristics. While the two trees based on the SNV alignment are very similar with respect to the two-distance metrics, the distances between the tree based on the gene alignment and the ones based on the SNV alignment are larger. For both metrics, the gene tree is more similar to the maximum parsimony tree than the maximum likelihood tree. This similarity is probably because the gene tree is also a maximum parsimony tree. However, while there is a significant discrepancy of the triplet distance when comparing the gene tree to the two SNV trees, the quartet distance is almost the same. The difference between the two-distance metrics, as described in Section 2.3 on page 11, is that the triplet distance was designed to compare rooted trees, while the quartet distance ignores the rooting of the tree and compares them as unrooted trees. Thus, this similarity of the quartet distances compared to the triplet distances might hint at a variance in the placement of the root in one of the trees.

While `MUSIAL` is focused on the identification of all variations present in the sequenced samples, `EVIDENTE` focuses on the identification of shared characteristics. The labeling of the SNVs to the corresponding nodes of the phylogenetic tree allows for a direct comparison within and between the samples and clades. The integration of the `SNPeff` analysis can help to identify genes that are mutated in specific subtrees and thus hint at the evolutionary adaption process. Additionally, this comparison is not limited to the SNVs and their affected genes but can also include other metadata. Thus, it is possible to associate metadata information, like antibiotic resistance, with the SNVs. This association can help to find SNVs or even genes that are involved or responsible for them. Its interactivity can lead to the discovery of similarities and thus point to new aspects for

further investigation.

As MUSIAL can automatically calculate a phylogenetic tree with RAxML-NG, an integration of the calculation of clade-specific SNVs could simplify the analysis. However, researchers might not want to analyze a maximum likelihood tree. Thus, the possibility to calculate these clade-specific SNVs for a separately reconstructed tree needs to remain. One possibility would be to integrate this calculation not into MUSIAL, but into EVIDENTE. The drawback here is that EVIDENTE was designed as an interactive visualization tool and this calculation can take several minutes, especially on larger trees, which stands in conflict with the interactivity of EVIDENTE and is thus not a viable option.

Chapter 4

DACCOR: Resolving repeats

Parts of this chapter were published in:
***A. Seitz**, F. Hanssen, and K. Nieselt (2018)*
DACCOR-Detection, characterization, and reconstruction of repetitive regions in bacterial genomes
PeerJ 6:e4742 https://doi.org/10.7717/peerj.4742

The previous chapter focused on the comparison of multiple samples based on mapping-based genome reconstruction. However, the informative value of these comparisons depends on the quality of the underlying reconstructions. This chapter introduces a novel method to improve the base-pair resolution of repetitive regions in mapping-based genome reconstructions.

4.1 Introduction

The mapping-based reconstruction of genomes relies on the correct placement of the reads to a given reference genome (Pop, 2009). Alignment programs like `BWA` or `Bowtie2` provide a quality score for each mapped read. This Phred quality score is defined as[1]

$$Q = -10\log_{10}(p) \tag{4.1}$$

where Q is the mapping quality and p is the probability that the read was mapped to an incorrect position (Li *et al.*, 2009), similar to the base-pair quality of sequenced reads described in Section 2.4.1 on page 13. This way, alignments with low mapping qualities can be removed, as these reads most likely stem from contaminations or have a low sequencing quality (Smith *et al.*, 2008). However, if a read stems from a repetitive region, it can be placed at multiple positions in the genome with an equal score. The resulting probability that the read is placed incorrectly (p) at any one of these positions is 1, which results in a mapping quality score (Q) of 0. Thus, the removal of reads with

[1] `https://samtools.github.io/hts-specs/SAMv1.pdf` accessed: May 29, 2019

a low mapping quality, which is often done in aDNA studies (Bos *et al.*, 2016), would also remove all reads mapped to repetitive regions. The reconstructions based on these mappings are then unable to generate a call for bases within these repetitive regions, as they are not covered at all. Repetitive regions are not only problematic in mapping-based sequence reconstruction but also lead to problems using *de novo* reconstruction methods (Simpson and Durbin, 2012).

However, repetitive regions in the genome play an essential biological role in many different species (Shapiro and von Sternberg, 2005). Both prokaryotic and eukaryotic genomes are known to contain thousands of repetitive regions (Treangen *et al.*, 2009). For example, the human genome is made up out of approximately 50% repetitive regions (Lander *et al.*, 2001). Furthermore, it appears as if tandem repeat regions are associated with outer membrane proteins. This suggests that they are involved in the adaption of the pathogens to their hosts (Denoeud and Vergnaud, 2004). Another example can be found in the bacterium *T. pallidum*. Its subspecies *T. pallidum pallidum*, which causes venereal syphilis, *T. pallidum pertenue*, which causes nonvenereal yaws, and *T. pallidum endemicum*, which causes bejel, can be distinguished by repetitive subsequences in the *arp* gene (Harper *et al.*, 2008). This is currently the only way to differentiate the subspecies, as there are no serological tests capable of it. Nonetheless, it is vital to be able to tell them apart, to provide the correct treatment for each disease.

A technological way to reconstruct these repetitive regions is to use long read sequencing like PacBio or Oxford Nanopore (see Section 2.1 on page 7 for more information). However, as described in Section 2.2 on page 10, it is not always possible to use these technologies.

Another method for the handling of repetitive regions is to mask duplicated and low-complexity regions before the mapping (Frith *et al.*, 2010). A masked version of the human reference genome is already available for download (UCSC, 2014). For references where no masked genome is available, it is possible to generate masked versions of the genome with programs like `RepeatMasker` (Smitt *et al.*, 1996). `RepeatMasker` compares the reference genome of choice to a database of known repetitive regions and generates the masked version based on identified matches. This may allow for the masking of a genome but cannot identify repetitive regions *de novo* and thus may fail to mask some regions that are not yet known as being repetitive.

While there are programs for the *de novo* identification of repetitive regions available, like `RepeatExplorer` (Novák *et al.*, 2013) and `RepARK` (Koch *et al.*, 2014), the first one is a galaxy-based web tool and the second one analyzes the sequenced reads instead of the reference genome. This analysis and removal of repetitive reads makes it impossible to know which regions are uncovered because of repetitive regions and which ones were not sequenced and may even be due to deletions. Thus, it is not possible to use either of them for the *de novo* identification of repetitive regions in an automated pipeline. A tool that identifies repetitive regions *de novo* using the program `VMatch` (Kurtz, 2003). It is based on suffix arrays (Weiner, 1973) and has been applied to identify and annotate repetitive regions (Lindow and Krogh, 2005) and create adapted reference genomes where the

repetitive regions are masked (Assuncao *et al.*, 2010).

The idea for the reconstruction of repetitive regions is to first identify all repetitive regions in a given reference genome. Each identified repetitive region is then reconstructed separately, together with the whole genome of the sample. Finally, the unresolved repetitive regions in the whole genome reconstruction are replaced with the separately reconstructed regions. The program DACCOR was developed to perform these steps automatically and allow easy integration in sequencing pipelines.

Friederike Hanssen initially developed the repeat-identification program of DACCOR in her Bachelor thesis. This was then extended and presented at the German Conference on Bioinformatics 2017 in Tübingen. It was later published as *DACCOR – Detection, characterization, and reconstruction of repetitive regions in bacterial genomes* in *PeerJ* (Seitz *et al.*, 2018).

4.2 Methods and implementation

The reconstruction using DACCOR is separated into three main steps, which are illustrated in Fig. 4.1 on the next page. First DACCOR identifies the repetitive regions *de novo* (see left part of Fig. 4.1 on the following page). This step is explained in detail in Section 4.2.1. After the *de novo* identification of the repetitive regions, DACCOR uses each identified region as a separate reference sequence. Thus, each identified repetitive region is reconstructed for the sequencing data of each sample. The top right part of the figure illustrates this step and a detailed description can be found in Section 4.2.2 on page 44. Finally, DACCOR combines the reconstructed repetitive regions with a reconstruction of the full genome to generate an enhanced reconstructed genome. There, bases inside repetitive regions that are unresolved in the full genome reconstruction are replaced with the corresponding bases of the separately reconstructed regions. This step is explained in detail in Section 4.2.3 on page 44.

4.2.1 *De novo* identification of repetitive regions

The first step of DACCOR is to *de novo* identify all repetitive regions in the given reference genome, implemented in the `identify` subprogram of DACCOR. It is separated into six steps, which are listed in the left box of Fig. 4.1 on the following page. Step 1 is to decompose the reference genome into its k-mers and discard all unique k-mers. Step 2 is an iterative process in which the current non-unique k-mers are merged if they overlap in all but one position (the first or the last position). This step is repeated until there are no matching k-mers or extended sequences left. Step 2 increases the length of the repetitive regions by one in each iteration. Low-complexity regions, consisting of only one base type, result in two separate regions that lie genomically right next to each other. These identical regions that lie directly next to each other are identified and combined in Step 3, which results in maximal exact repetitive regions.

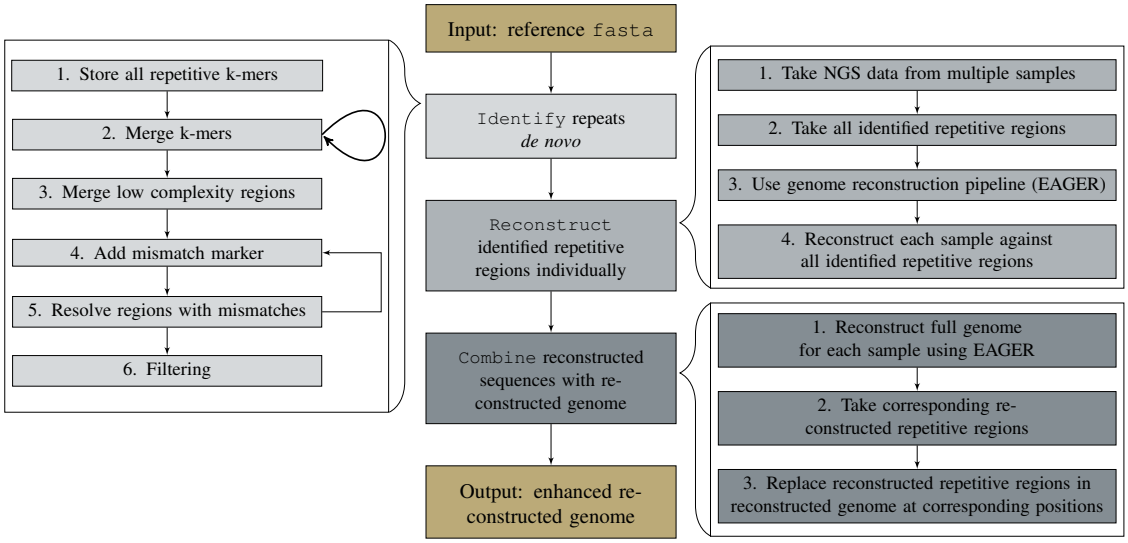


Figure 4.1: Workflow of DACCOR. The overview of the methodology is illustrated in the center part of the figure. First, the repetitive regions in the reference FASTA file are identified *de novo* (see left box for details). Each of these regions are then reconstructed separately (see top right box for details). Finally, the reconstructed repeat regions are combined with the reconstruction of the full genome to create an enhanced reconstructed genome (see bottom right box for details).

Steps 4 and 5 identify repetitive regions with mismatches. The general idea is that repetitive regions with one mismatch consist of two repetitive regions that are genomically next to each other, separated only by one base. To identify them, a mismatch marker is appended to all identified repetitive regions in step 4. This marker stands for an unknown base and represents one mismatch. If the two regions are separated by this one mismatch, the first of the two regions is now represented as the same region for both occurrences in the genome. This allows for the merging of the two repetitive regions into one containing a mismatch in Step 5. An illustration of the general idea of this method can be seen in Fig. 4.2 on the next page. The addition of a mismatch marker to *Repeat 1* extends it by one base and encompasses the mismatch. The two perfectly matching repeats, together with the separating mismatch are merged into one continuous *Repeat 3*, containing the mismatch. To account for more than one mismatch, Steps 4 and 5 are repeated until a predefined maximum number of mismatch markers have been added. The identification of the exact maximum repeats and the resolving of the regions with mismatches are parallelized.

In the final Step 6 of the repeat identification of DACCOR, all repetitive regions are filtered to remove regions shorter than a predefined length, chosen by the user. Additionally, trailing mismatch markers are removed from the sequences.

To identify repetitive regions with mismatches, the regions flanking the mismatch need to be identified first. The k-mer size used for the identification of maximum exact repeti-

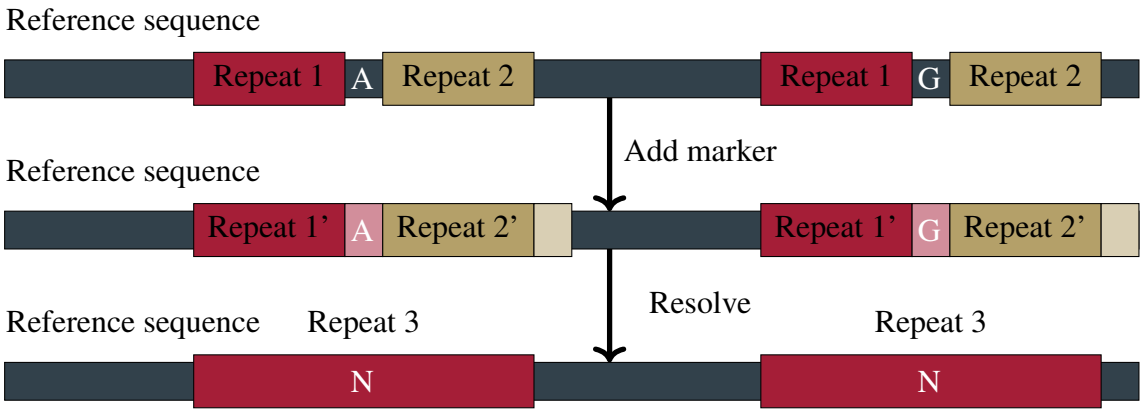


Figure 4.2: The general mismatch idea that is used for the identification of repetitive regions with mismatches in DACCOR. Repeat 1 and 2 are extended by one mismatch marker, which allows for the identification of the new Repeat 3 containing one mismatch. The extension of Repeat 2 is removed in the end, as the repetitive region can't be extended further to the right.

Table 4.1: Example summary of the repeat identification of DACCOR. The first line contains the reference name, followed by general statistics about the identified repetitive regions. These are followed by a description of each identified repetitive region.

>NC_021490.2	
category	value
Total number of repetitive regions	29
Total number of different repetitive regions	14
Minimum length	115
Maximum length	3283
Average length	823
Number of repetitive regions containing mutations	1
Number of repetitive regions also contained in other sequences	0

fasta Name	repeat Name	start index	end index	length	number mismatches	indices of mismatches
NC_021490.2_pos_[134901,152303]_length_1845	repeat_0	134901	136746	1845	0	
NC_021490.2_pos_[135160,152562,672904]_length_216	repeat_1	135160	135376	216	0	
NC_021490.2_pos_[333944,674935]_length_374	repeat_7	333944	334318	374	3	336, 337, 338

tive regions is essential for this step. Furthermore, DACCOR can only identify mismatches but no indels. However, if there are repetitive regions with indels, they are still identified as separate repetitive regions.

The output of the `identify` part of DACCOR consists of several different files. The output contains a multi-FASTA file containing all the identified repetitive regions. The sequence name of each region is comprised of the sequence name of the analyzed reference sequence, the starting positions of the regions in the reference genome, as well as the length of the region. Additionally, a summary of the identified repetitive regions is contained in the file with the suffix `Summary`. This summary contains general statistics about the analyzed sequences, as well as a description of each identified repetitive region. An example excerpt of this summary file can be seen in Table 4.1. If desired, the identified repetitive regions are extracted with an additional margin. This margin is

helpful for the reconstruction of the repetitive regions, as reads that are not fully contained in the repetitive region but overlap part of it can also be mapped. In addition to the multi-FASTA file, it is possible to write each identified repetitive region to its own FASTA file. This makes it possible to use them as separate references for the later reconstruction of all repetitive regions.

If the input was a multi-FASTA file, DACCOR analyzes the files together and can identify regions that are present in more than one input sequence, even if these sequences are not repetitive in the sequences themselves. However, it is also possible to analyze each sequence separately and only identify sequences that are repetitive within each of the different sequences.

Furthermore, it is possible to replace the repeat identification step implemented in DACCOR with any other repeat identification program. The usage of the output of VMatch for the identification of the repetitive regions is already implemented in DACCOR.

All parameters and their default values for the repeat identification of DACCOR are described in supplementary Table B.2 on page 93.

4.2.2 Automatic reconstruction of repetitive regions

After the identification of all repetitive regions, the subprogram `reconstruct` can generate a reconstructed sequence for each identified repetitive region and sequenced sample. For this, DACCOR automatically generates configuration files for the EAGER pipeline and uses EAGER to perform the reconstruction. DACCOR reconstructs each sequenced sample separately. For this, the location of the input FASTQ files need to be given in a tab-separated file, with the first column containing the sample name and the second and third containing the path to the paired-end FASTQ files. If a sample was sequenced using single-end and not paired-end sequencing, the third column can be omitted. First, each sample is reconstructed against the complete reference genome. If an EAGER reconstruction for the complete reference genome already exists, the path to this reconstruction can be given as a tab-separated file. This file must contain the sample name in the first column and the path to the EAGER result folder in the second column. Afterward, each sample is reconstructed against each of the previously identified repetitive regions. For this, a symbolic link is created for the preprocessed FASTQ files to reduce the overall runtime of the different EAGER runs.

All parameters and their default values for the separate reconstruction of the repetitive regions are described in supplementary Table B.3 on page 93.

4.2.3 Combining the reconstructed regions with the full genome reconstruction

Finally, the reconstructed genomes, as well as the reconstructed repetitive regions can be combined by the `combine` subprogram of DACCOR. For each sample, DACCOR uses

the complete genome reconstructed with EAGER. These reconstructed genomes are given through a tab-separated file with the first column being the sample name and the second column the path to the EAGER output folder. The reconstructed repetitive regions must be given in the same format.

DACCOR then replaces unresolved base-pairs in the reconstructed genome with the corresponding base-pairs of the reconstructed repetitive regions. As the name of each region contains its starting position in the reference sequence, DACCOR can directly compare the separately reconstructed region to the corresponding region in the full genome reconstruction of each sample. Each unresolved base in the full genome reconstruction is then replaced with the corresponding base of the separately reconstructed region.

To document the changes in the reconstruction, DACCOR creates a new FASTA file containing the enhanced reconstructed genome. In addition to the enhanced reconstructed genome, a file containing the positions that could be reconstructed in the enhanced reconstruction of DACCOR but were not resolved in the original reconstruction with EAGER is also created. The base-pairs that are still unresolved in the repetitive regions are written to another file.

All parameters and their default values for the combination of the reconstructed repetitive regions with the reconstructed genomic sequence are described in supplementary Table B.4 on page 93.

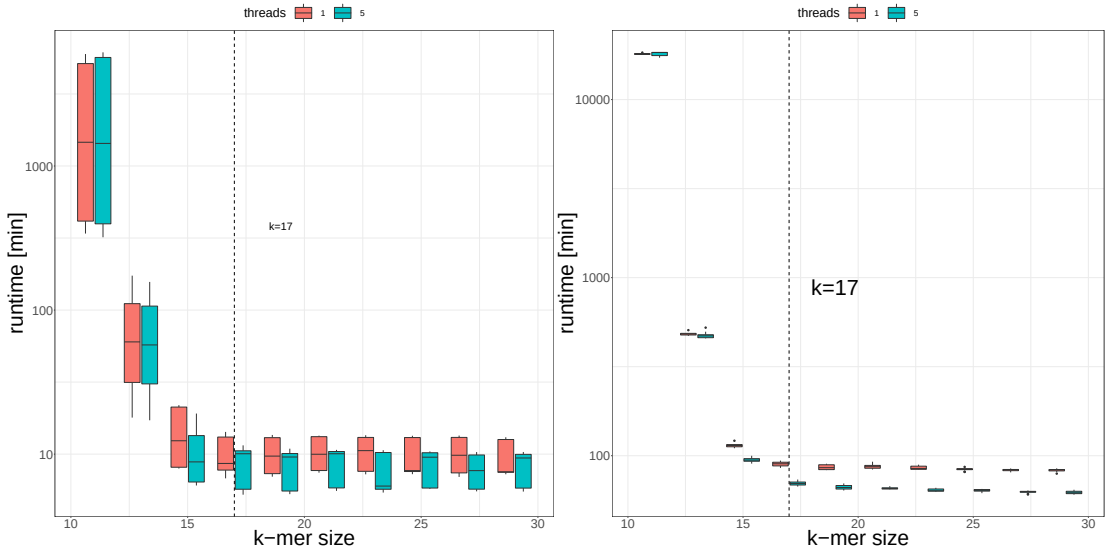
4.2.4 Automatically generating enhanced reconstructed genomes

For the automatic generation of enhanced reconstructed genomes with DACCOR, the subprogram `pipeline` combines the three previously described steps. It first identifies the repetitive regions in the reference genome using the `identify` subprogram of DACCOR. These regions, together with the full genome, are then automatically reconstructed for all input samples using the EAGER pipeline. The resulting reconstructions are then combined to generate an enhanced reconstructed genome for each sample. Here, each unresolved base in the full genome reconstruction is replaced with the corresponding base of the separately reconstructed region.

All parameters and their default values for the automatic pipeline resulting in enhanced full-genome reconstructions are described in supplementary Table B.5 on page 94.

4.3 Results

To establish the overall performance of DACCOR, it was evaluated on several different bacteria. The first part of this section addresses the runtime of the `identify` subprogram of DACCOR, while the second one compares the enhanced genome reconstruction pipeline of DACCOR with the EAGER pipeline and the *de novo* reconstruction using SPAdes.



(a) *T. pallidum* (b) *S. flexneri*

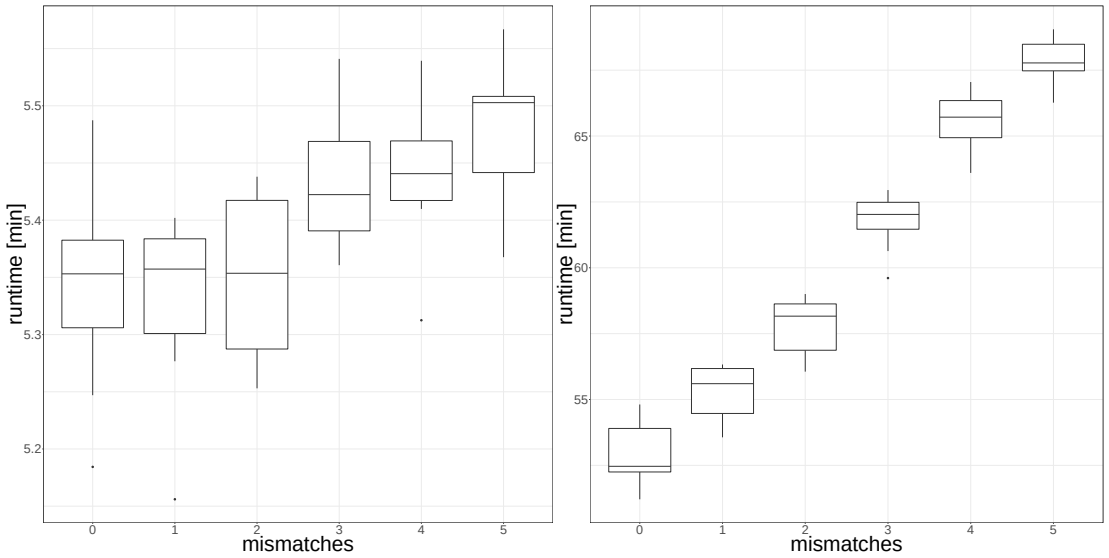
Figure 4.3: Runtime analysis of the `identify` subprogram of DACCOR on the genomes of *T. pallidum* (a) and on *S. flexneri* (b) using one (red) and five (cyan) threads. Each *k*-mer, runtime, and thread combination was repeated 10 times.

4.3.1 Runtime benchmark of the `identify` subprogram

To get an estimate of the runtime of the `identify` subprogram of DACCOR, the *Nichols* strain of *T. pallidum*, as well as the strain *301* of *S. flexneri* was analyzed with different parameters. It was evaluated with all odd *k*-mers between 11 and 29 to identify repetitive regions within the genome. Furthermore, the analysis was performed using one and five threads. The results of this analysis are illustrated in Fig. 4.3. For larger values of *k*, the choice of *k* generally has no impact on the runtime. However, we observed that the runtime increases drastically for small k. A slight increase in the runtime is observed for a *k*-mer size of 15, which increases for the *k*-mer sizes of 13 and below. Based on this observation, the *k*-mer size was set to 17 for all analyses in this chapter.

To evaluate the impact of the allowed number of mismatches on the runtime, the two genomes (*T. pallidum*, and *S. flexneri*) were analyzed with a *k*-mer size of 17, five threads, and a varying number of mismatches. These results are illustrated in Fig. 4.4 on the facing page. It shows that with an increasing number of mismatches, the runtime also increases slightly. In the case of *T. pallidum*, the runtime appears to stay relatively constant for one to three allowed mismatches, after which it starts to increase for four and five mismatches. This is different for the genome of *S. flexneri*, where there appears to be a linear increase in the runtime starting from the first allowed mismatch.

To get a general overview of the repetitiveness of different bacteria, the `identify` subprogram was used on four different published bacterial reference genomes: *Treponema*



(a) *T. pallidum* (b) *S. flexneri*

Figure 4.4: Runtime analysis of the `identify` subprogram of `DACCOR` on the genomes of *T. pallidum* (a) and on *S. flexneri* (b), allowing for a different number of mismatches. Each k-mer, runtime, and mismatch combination was repeated 10 times.

pallidum, *Shigella flexneri*, *Escherichia coli*, and *Mycobacterium leprae*. The results, depicted in Table 4.2 on the next page, show an overview of the repetitive regions that could be identified using a k-mer size of 17, allowing for up to five mismatches, and only reporting regions of length > 100 bp. The `identify` subprogram reports that over 8% of the genome of *S. flexneri* is repetitive, which increases to 9.4% when analyzing it together with its plasmid, whereas the genomes of the other analyzed bacteria are more unique with a maximum of 2.4% of the genome being repetitive in the genome of *E. coli*. This percentage is even smaller in the genome of *M. leprae*, with 2.3% and *T. pallidum*, with only 2%. The longest repetitive region, (5,383 bp), was identified in the genome of *S. flexneri*. This observation is interesting, as the average length of the repetitive regions is the shortest in this genome (570 bp without and 541 bp with the plasmid, compared to 643, 706, and 823 in the other bacteria). Even though the genomes of *E. coli* and *S. flexneri* are about the same length, the genome of *S. flexneri* contains over five times as many repetitive regions than the genome of *E. coli*. Considering only regions that also contain mismatches, *S. flexneri* still contains more than twice as many repetitive regions than *E. coli*. When analyzing *S. flexneri* with the plasmid, there are 316 additional repetitive regions. Out of those, about a third (103) are regions that are also contained in the genome. This result shows that when reconstructing multiple sequences simultaneously using a multi-`FASTA` file as references, these sequences may share repetitive regions.

Table 4.3 on page 49 shows the runtime for the identification of the repetitive regions

Table 4.2: Repetitiveness of different bacterial genomes based on repetitive regions that are longer than 100 base-pairs and contain up to 5 mismatches. The analysis was performed with a k-mer size of 17. The *S. flexneri* genome contains a plasmid. It was analyzed twice, once with the plasmid (G+P) and once without (G).

	E. coli	*M. leprae*	*S. flexneri* G	*S. flexneri* G+P	*T. pallidum*
genome size [bp]	4,646,332	3,268,203	4,607,202	4,828,820	1,139,633
# repetitive regions	242	190	1,249	1565	29
# different repetitive regions	104	76	482	701	14
max length of repetitive regions	3,141	2,578	5,383	5,383	3,283
average length of repetitive regions	706	643	570	541	823
repetitive regions with mismatches	47	9	127	238	1
# regions contained in multiple sequences	0	0	0	103	0
sum of repetitive bases	160,897	119,871	660,261	778,028	23,892
% of genome repetitive (non-overlapping)	2.4	2.3	8.3	9.4	2

on the different bacterial genomes with a k-mer size of 17, allowing for up to five mismatches, and only reporting repetitive regions longer than 100 bp. The identification of the repetitive regions on the smallest, least repetitive genome of *T. pallidum* (see Table 4.2) was the fastest. The identification of the repetitive regions on the *S. flexneri* genome with its plasmid required the most time. This genome is both the longest and the most repetitive one. This indicates that the runtime may be dependent on the size of the genome and its repetitiveness. After *T. pallidum*, the genome of *M. leprae* has the shortest genome and contains the fewest number of repetitive bases, followed by *E. coli* and *S. flexneri* without its plasmid. This observation is also reflected in the runtime of DACCOR. However, although the genome of *E. coli* and *S. flexneri* without the plasmid are about the same size, the repeat identification is faster on *E. coli*. The multithreaded calculation of the repeat identification resulted in a decreased runtime. However, there still appears to be a significant sequential overhead.

Table 4.3: Median runtime (in seconds) of the `identify` subprogram of DACCOR on the bacteria described in Table 4.2 on the preceding page with a *k*-mer size of 17, allowing for up to five mismatches, and differing number of threads on 10 repetitions.

sample	1 thread	5 threads	speedup [%]
E. coli	57.5	55.3	4.0
M. leprae	25.1	23.6	5.9
S. flexneri G	112.8	96.4	14.5
S. flexneri G+P	127.1	108.3	14.8
T. pallidum	8.3	6.6	20.5

4.3.2 Comparison of DACCOR to other genome reconstruction methods

To compare the repeat resolution of DACCOR with a *de novo* reconstruction of the genome, two sequenced samples of *T. pallidum*, published by Arora *et al.* (2016) were assembled using SPAdes. The sample *AR1* had a relatively low mean coverage of 7X, whereas the sample *AR2* had a very high one of 157X. Both were sequenced on an Illumina HiSeq 2500 using paired-end sequencing with a read length of 101 bp. General statistics about the assemblies are summarized in Table 4.4 on the following page. As expected, the assembly of *AR2* is better than the one of *AR1*. While there was only one contig of length $\geq 10,000$ base-pairs, the assembly of *AR2* contained 38 of them. The N50 of AR2 based on all contigs ≥ 1000 is four times larger than for AR1, the same holds for the mean contig length. The mapped contigs were then analyzed and compared to the identified repetitive regions to see which and how many of the repetitive regions could be reconstructed using the *de novo* assembly.

The results of the reconstructed repetitive regions with EAGER, the SPAdes assemblies, and the enhanced reconstruction of DACCOR are depicted in Table 4.5 on page 52. On the sample *AR1*, DACCOR could resolve 99.9% of the repetitive regions, compared to only 59% with the *de novo* approach using all mapped contigs of the assembly with SPAdes and 23% with the default mapping-based reconstruction of the EAGER pipeline. This result is like the one for sample *AR2*, which has a higher coverage. Using DACCOR, it was possible to resolve all repetitive regions, whereas 69.5% and 51% of them could be resolved using the SPAdes assemblies and only 42% with EAGER.

In addition to the two samples *AR1* and *AR2*, DACCOR was used to create enhanced genome reconstructions for all 106 samples of *T. pallidum* from the publications of Arora *et al.* (2016), Pinto *et al.* (2016), and Sun *et al.* (2016). The median of the percentage of additionally reconstructed bases in the repetitive regions over all samples is 81.6% (see bottom of Fig. 4.5 on page 51). It can be deduced that for the majority of all samples almost all unresolved positions could be resolved. As bases in the original reconstruction of the genome are only replaced if they were previously unresolved, all reconstructions

Table 4.4: Assembly statistics of the two clinical *T. pallidum* samples (*AR1* and *AR2*) published by Arora *et al.* (2016). The first `sample` column specifies the preprocessing steps in addition to the sample name. The *filtered* tag refers to contigs that are at least 1,000 bp in length and the *mapped* tag to contigs that were successfully mapped against the reference genome of *T. pallidum*.

sample	# contigs	N50	mean contig length	# contigs ≥1,000	# contigs ≥10,000	longest contig
AR1	43,162	247	245.9	396	1	10,286
AR1 mapped	1,137	978	696.6	174	1	10,286
AR1 filtered	396	1,941	1,895.6	396	1	10,286
AR1 filtered mapped	174	2,433	2,235.4	174	1	10,286
AR2	278,193	278	254.0	1,589	38	75,865
AR2 mapped	852	17,810	1,528.0	105	38	75,865
AR2 filtered	1,589	1,670	1,981.7	1,589	38	75,865
AR2 filtered mapped	105	20,441	10,694.0	105	38	75,865

using `DACCOR` contain at most the same number of unresolved bases as before. While there are 17 samples where no improvement was observed, it is important to note that in all of these samples, all repetitive regions were already perfectly resolved without the enhanced repeat resolution of `DACCOR`. Additionally, there are 45 samples with an improvement of $\geq 95\%$. A detailed illustration of the improvement of each sample is shown in the top part of Fig. 4.5 on the next page. In 30 samples, using `DACCOR`, it was possible to resolve all bases contained in the repetitive regions that were previously unresolved in the reconstruction of the `EAGER` pipeline.

The bases that remain unresolved in the enhanced reconstruction with `DACCOR` do not need to stem from low-coverage regions. If a sample gained a variation in only one copy of a repetitive region, `EAGER`, which is also used for the reconstruction of each repetitive region, will also call this position as unresolved. This unresolved base call is due to the base frequency of the mapped reads. If both copies were sequenced with similar coverage, the mentioned position would have a frequency of about 50% for each base. The two longest repetitive regions identified in the genome of *T. pallidum* were used to see if there are positions matching these criteria. The shorter of the two regions corresponds to the 16S rRNA, and the longer one corresponds to the 23S rRNA. As the goal was to identify positions that are mutated in only one copy of the gene, the expected frequency for the different base-calls is around 50%. This means that they would resemble a heterozygous position. Thus, the program `MUSIAL`, described in Chapter 3 on page 19, could be used to generate reconstructions of these regions while allowing for the identification of these positions. For this, `MUSIAL` was executed with a minimum coverage for each base of 3,

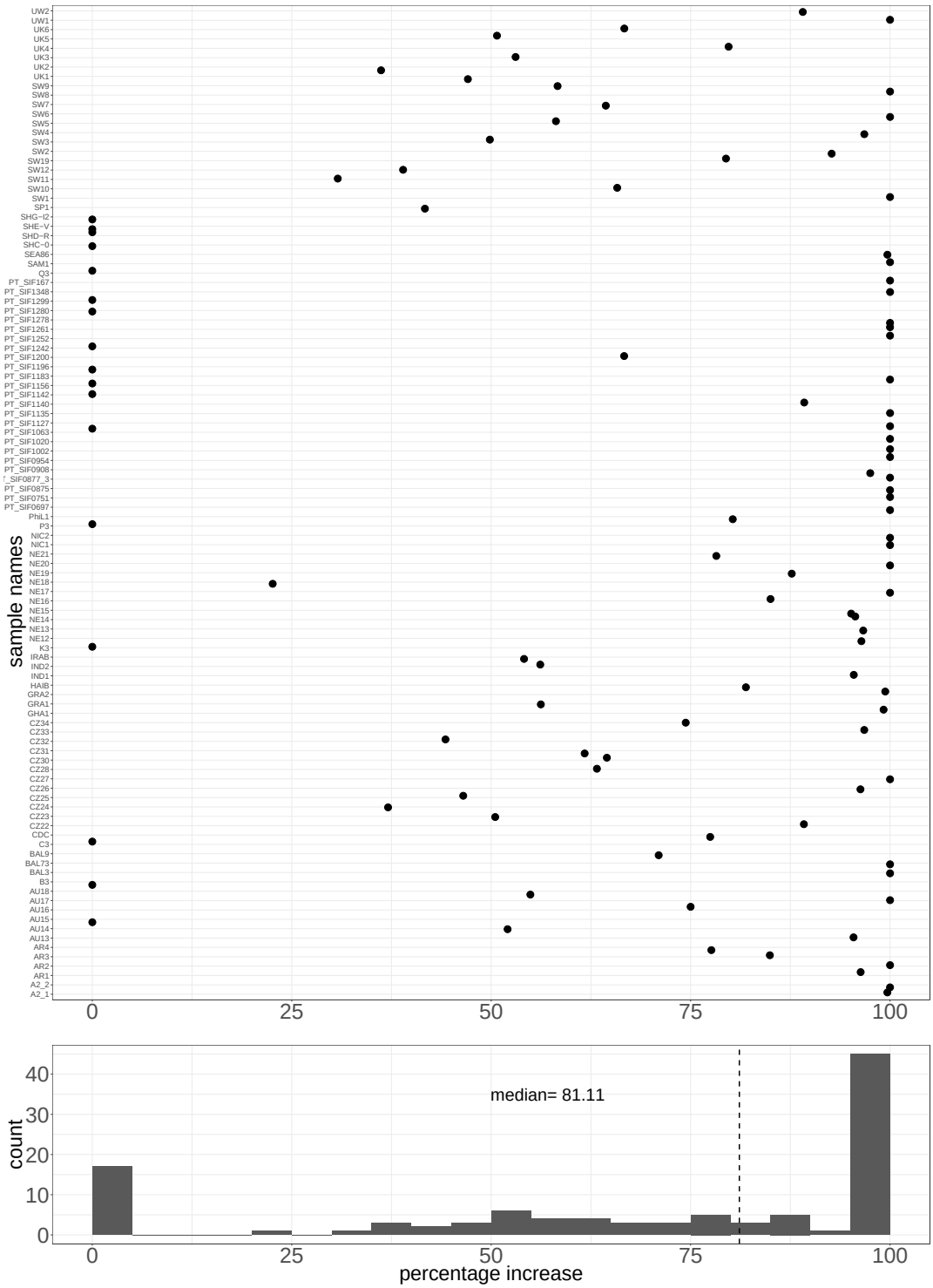


Figure 4.5: Improvements on the base-pair resolution of all bases contained in repetitive regions achieved with DACCOR in comparison to genome reconstructions generated with EAGER. The top plot shows the improvement for each sample, whereas the bottom plot shows a histogram summarizing these results.

Table 4.5: Comparison of standard genome reconstruction with EAGER, *de novo*-based with SPAdes, and DACCOR based on two clinical *T. pallidum* samples published by Arora *et al.* (2016) (*AR1* and *AR2*). The "SPAdes filtered" method only used contigs of length $\geq$ 1,000. The reported numbers only refer to the repetitive regions with the corresponding margin regions, which encompass 30,330 bp.

Sample	mean coverage	Method	#N	%N
AR1	7X	EAGER	23,348	76.98
		SPAdes	12,459	41.08
		SPAdes filtered	17,336	57.16
		DACCOR	29	0.1
AR2	157X	EAGER	17,549	57.86
		SPAdes	9,247	30.49
		SPAdes filtered	14,858	48.99
		DACCOR	0	0

Table 4.6: Variant positions in the 16S rRNA and 23S rRNA genes of *T. pallidum*. The number of variant positions refer to positions where at least one of the 106 reconstructed samples of *T. pallidum* show a variation. Similar, the number of positions that show variance between the two respective copies of the genes are counted if at least one sample shows such a behavior.

	16S rRNA	23S rRNA
gene length [bp]	1495	2900
# variant positions	36	46
# positions variant in only one copy	30	43

as well as a frequency between 25 and 75% for heterozygous calls. The results are shown in Table 4.6. Interestingly, 83% in the 16S rRNA and 93% in the 23 rRNA of the variant positions indicate the presence of variations in only one copy of the respective gene.

4.4 Discussion and conclusions

DACCOR is an automated pipeline that can increase the base-pair resolution of repetitive regions in mapping-based genome reconstructions. DACCOR first performs a *de novo* identification of all repetitive regions in the given reference genome. DACCOR has an integrated method for this identification, but it is possible to substitute the identification step with another program of choice. The output of VMatch is already supported, and other repeat-identification programs could also be integrated.

The identified repetitive regions are then reconstructed separately for each sample. DACCOR automatically creates and runs configurations for the EAGER pipeline. Additionally, DACCOR creates a configuration file for the reconstruction of the complete genome, if the reconstructed genome does not already exist.

The reconstructions of the different repetitive regions are then combined with the reconstruction of the full genome to create an enhanced genome reconstruction for each sample.

As the repeat-identification of DACCOR is based on k-mers, the identification of the repetitive regions is dependent on the choice of k. Another well-known bioinformatic method based on k-mers is the *de novo* assembly using *de Bruijn* graphs. There, the optimal choice for k depends on the length of the genome, as well as the reads, the quality of the reads, and the coverage of the genome (Zerbino and Birney, 2008). The main difference between the k-mer based *de novo* assembly and the repeat-identification of DACCOR is that in the assembly approach, the choice of k is set to maximize the uniqueness of the resulting k-mer sequences (Gardner and Hall, 2013), whereas in DACCOR, the number of non-unique k-mer sequences should be maximized. As DACCOR first splits the reference genome into its k-mers, it is not possible to identify repetitive regions that are shorter than k. Thus, we suggest a small value for k. However, the smaller the value of k, the more sequences need to be compared, which results in longer runtimes. If the value for k is set too low, randomly matching sequences can increase the runtime of DACCOR significantly. Furthermore, to identify repetitive regions with mismatches, DACCOR needs to identify the two repetitive regions flanking the mismatch(es) beforehand. If the distance between the mismatches is smaller than the value of k, it is not possible to identify the corresponding regions. However, in the case of the allowed mismatches, the runtime increased starting from the first allowed mismatch on the genome of *S. flexneri*, in contrast to *T. pallidum*, where a difference of the runtime is only observed after two, and a definite increase only after three allowed mismatches. This increase in the runtime can be explained, as the genome of *S. flexneri* is more repetitive than the genome of *T. pallidum*. However, the runtime also only increased for k-mers smaller or equal to 15. Thus, the increase of the k-mer size from 15 to 17, which increases the possible number of k-mers by a factor of 16 (4^2) from $\approx 1.07 \cdot 10^9$ to $\approx 1.7 \cdot 10^{10}$ appears to be enough to reduce the number of detected repetitive sequences that occur randomly at multiple locations in the genome. Thus, DACCOR does not need to analyze these sequences and only has to analyze biologically relevant repetitive sequences. Thus, we recommend setting the k-mer size for the repeat-identification of DACCOR to 17.

Projects using mapping-based approaches for the reconstruction of closely related genomes from short-read NGS data often use modified reference genomes where the repetitive regions are masked (Tarailo-Graovac and Chen, 2009). These modified reference genomes contain only one copy of each repetitive region. The other copies of the repetitive regions are masked. However, because of the high genomic complexity of repetitive regions, as well as the fact that repetitive regions can overlap, this masking approach is also problematic and can lead to incorrect reconstructions. An example of

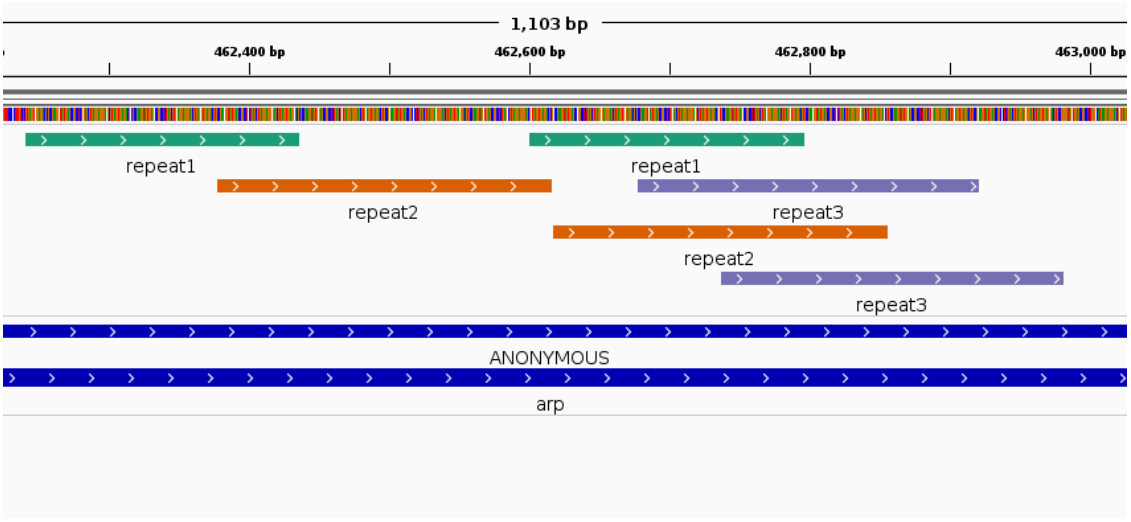


Figure 4.6: Repetitive regions in the *arp* gene of *T. pallidum*. They were identified using the repeat identification program of DACCOR with a *k*-mer size of 17 and allowing for zero mismatches. The image is based on a screenshot of the Integrative Genomics Viewer (Robinson *et al.*, 2011)

such high complex repetitive regions are the repetitive regions contained in the *arp* gene in *T. pallidum*. These repetitive regions are important for the differentiation between the different disease-causing bacteria of *T. pallidum pallidum*, *T. pallidum pertenue*, and *T. pallidum endmicum* (Harper *et al.*, 2008). There, it is not possible to mask one copy of a repeat without also masking parts of other repetitive regions, thus losing genomic information, or leaving parts of the repetitive regions unmasked. Fig. 4.6 shows these repetitive regions contained in the *arp* gene. They were identified using DACCOR with a *k*-mer size of 17 and zero mismatches. Fig. 4.6 shows that the purple regions (*repeat3*) are overlapping with itself. Thus, the masking of one copy of it would also mask most of the other copy. Furthermore, if the first copy of the green repetitive region (*repeat1*) is kept in the genome, the masking of the second occurrence would also mask parts of both occurrences of *repeat3*. Additionally, both occurrences of the orange repetitive region (*repeat2*) overlap with parts of both copies of either *repeat1* or *repeat3*. This example shows that it is not possible to reliably mask repetitive regions in reference genomes.

The runtime analysis of the repeat identification of DACCOR showed that its runtime depends mainly on the length of the input genome. However, although the genomes of *E. coli* and *S. flexneri* are very similar concerning the length of the genome, there was a difference in the runtimes on these two genomes. It took longer to identify all repetitive regions in the genome of *S. flexneri*. The main difference between the two genomes is that the one of *S. flexneri* is more repetitive. This observation shows that the runtime is not only dependent on the length of the genome, but also on the repetitiveness of the genome that is analyzed. Additionally, the usage of multiple threads decreased the

runtime in all cases, but again, the rate of improvement also appears to be dependent on the repetitiveness of the genome. While the improvement in the genomes of *T. pallidum*, *M. leprae*, and *E. coli* are very small, the largest improvement was observed on the genomes of *S. flexneri*. The reason for this probably lies in the implementation of the repeat identification. As all identified repetitive regions are extended by one base in each step, it is only possible to parallelize this step and extend all regions in parallel before continuing with the next iteration. Thus, when there are more repetitive regions to extend, the parallelization results in a more significant reduction of the runtime.

The example of *S. flexneri* further showed that its plasmid contains regions that are also present in its genome. Thus, even if a reconstruction method would account for the repetitive regions within each sequence, these regions need not be repetitive within the respective sequence to cause problems in the reconstruction. Because of this, it is essential to analyze all sequences that are contained in a multi-FASTA reference file to be able to account for these shared regions.

The comparison between the genome reconstruction approaches using DACCOR, the EAGER pipeline, and the *de novo* assembly of SPAdes showed the advantages of the separate reconstruction of each repetitive region with DACCOR. While the *de novo* assembly increased the resolution of the repetitive regions in comparison to the mapping of the short reads against the reference genome by up to 50% for the two samples, DACCOR was able to resolve 99.9% and more of all repetitive regions. The contigs created by the *de novo* assembly resulted in more genomic positions that were covered by unambiguously mapped sequences. However, the *de novo* assembly has similar problems as the mapping when reconstructing repetitive regions. It is not possible to create contigs spanning the complete repetitive regions, as there are ambiguities in the overlapping reads that cannot be resolved. Thus, the contigs span more of the repetitive regions than the raw reads, which is why the *de novo* assembly with SPAdes could reconstruct more repetitive bases than the EAGER pipeline, but not all of them. DACCOR, on the other hand, uses each identified repetitive region separately, which is why there are no ambiguities when mapping the raw reads against each repetitive region. The resulting high-quality mapping results are then used for the reconstruction of the regions leading to almost perfect reconstructions.

While the identification step of DACCOR can account for mismatches, it is currently not possible to identify repetitive regions containing indels. However, similar to the identification of mismatches, if the two repetitive regions flanking such an indel are long enough, DACCOR can identify them and use them as separate references for the reconstruction step. Thus, it is still possible to create an enhanced reconstruction of these regions, as the position containing the indel will be present in the reconstruction due to the additional margin that is used for the extraction of the repetitive regions.

While the comparison between the different reconstruction methods showed that it is possible to gain additional sequence information using DACCOR, the comparison of DACCOR and the default EAGER pipeline of 106 *T. pallidum* samples illustrated how much information can be gained through DACCOR in larger sequencing projects. In almost

half of the samples (45), it was possible to reconstruct all repetitive regions completely, whereas this was not possible using the standard EAGER pipeline. In about one-sixth of the samples (17), all repetitive regions were already completely reconstructed using the default mapping-based reconstruction of EAGER. On the remaining samples (44), the number of unresolved repetitive bases was decreased by at least 23% using DACCOR. The seventeen samples that were already perfectly reconstructed before the enhanced reconstruction with DACCOR are all samples with a mean coverage of more than 200 reads (Arora *et al.*, 2016; Pinto *et al.*, 2016; Sun *et al.*, 2016).

The individual reconstruction of the repetitive regions allowed the identification of positions that are only mutated in one copy of duplicated genes. An analysis of the repetitive regions corresponding to the 16S and 23S rRNA of *T. pallidum* show that most of the variant positions that lie within these two genes show samples where only one copy of the respective gene appears to be mutated. It might be interesting to study these samples in more detail because the 23s rRNA appears to be involved in the antibiotic resistance of *T. pallidum* (Arora *et al.*, 2016). As these positions would result in an unresolved base in both the default EAGER reconstruction, as well as in the one with DACCOR, these positions are responsible for some bases that remain unresolved in the reconstruction of DACCOR.

The next steps to improve DACCOR would be to decrease the runtime of the `identify` subprogram, as well as to integrate its enhanced genome reconstruction with MUSIAL. To decrease the runtime of the repeat-identification, it could be possible to reduce the number of comparisons DACCOR has to calculate in the second step of the repeat identification. There, the length of the sequences is extended by one in each iteration. Because it is not possible to extend a maximal exact repetitive region, these regions could be ignored in the next iterations. Thus, the number of comparisons would decrease in the remaining iterations, which should result in a decreased runtime.

Furthermore, the combination of DACCOR and MUSIAL could decrease the number of unresolved bases in the output of MUSIAL. There are two ways this could be achieved. One is to create an additional FASTA parser in MUSIAL so that it can directly parse the reconstructed genome file. The other one would be for DACCOR to create an additional VCF file. MUSIAL can then directly parse this file and use it for its comparative analysis.

Chapter 5

MADAM: *De novo* assembly

Parts of this chapter were published in:
***A. Seitz**, K. Nieselt (2017)*
Improving ancient DNA genome assembly
PeerJ 5:e3126 https://doi.org/10.7717/peerj.3126

5.1 Introduction

The analyses presented in the previous chapters focused mainly on mapping-based genome reconstruction, where the reads are aligned against the reference genome of a closely related species (see Section 2.4.2 on page 14). However, this reference-based approach cannot detect large insertions, deletions, or other genomic rearrangements. Furthermore, it is possible that bacteria could have lost genomic regions over time. The modern reference genome would, consequently, not contain these regions. Thus, sequencing reads stemming from such lost regions cannot be mapped against the modern reference genome and would probably be disregarded as contamination. To still be able to identify these regions, it is possible to reconstruct the target genome using a *de novo* assembly. This procedure is based solely on the sequencing reads and can thus be used to reconstruct lost regions when there is no closely related reference genome available.

De novo assembly is particularly interesting in the context of ancient DNA (aDNA) because there is no reference genome for extinct species. Even the analysis of ancient bacteria by comparing them to their modern reference genome can be problematic due to their fast mutation rate (Papadopoulos *et al.*, 1999) and evolutionary diversification. Thus, a *de novo* assembly can help to assess genomic similarities between the ancient sample and the modern genomes in order to compare these samples.

Since the introduction of NGS data, a number of *de novo* assembly programs for short reads, like `SOAPdenovo2` (Luo *et al.*, 2012) and `SPAdes` (Bankevich *et al.*, 2012) have been developed that can handle many short NGS reads. Compared to the longer Sanger reads, these shorter reads pose several challenges (Li *et al.*, 2010; Sawyer *et al.*, 2012). The large number of sequenced reads, as well as the additional information gained from the paired-end and mate-pair sequencing, can mitigate these problems in modern DNA sequencing projects. However, due to the degradation of ancient DNA (see Section 2.2

on page 10), reads sequenced from those samples only contain very short fragments with an average length between 44 and 172 bp (Sawyer *et al.*, 2012). As described in Section 2.4.1 on page 13, paired-end reads generated from such samples often overlap and are subsequently merged. Thus, the scaffolding process (see Section 2.4.3 on page 15) cannot use them to assess contig distances and rearrangements. Additionally, as described in Section 2.2 on page 10, long reads will not generate additional information, compared to the sequencing of modern samples.

Generating *de novo* assemblies is still challenging on NGS data generated from modern samples (Chao *et al.*, 2015), which is why the development and improvement of assembly programs is an active field of research (Phillippy, 2017). One *de novo* assembly program is `ALLPATHS-LG` (Gnerre *et al.*, 2011), which won the so-called Assemblathon (Earl *et al.*, 2011). It uses the information from long fragments by means of paired-end and mate-pair sequencing and can be seen as one of the best programs for *de novo* assembly currently available (Utturkar *et al.*, 2014). However, it is incompatible with aDNA because of the short fragments typically sequenced from aDNA.

After the aDNA read preprocessing (see Section 2.4.1 on page 13: trimming, clipping, and merging of overlapping paired-end reads), the read length distribution is often very skewed. *De Bruijn* graph assemblers are highly reliant on the length of the k-mer chosen for the generation of the graph (Li *et al.*, 2012). Even for sequencing projects using modern DNA, choosing an optimal value for k is a difficult problem (Durai and Schulz, 2016). On aDNA samples, this skewed read length distribution complicates the choice of k even further. Reads that are shorter than the chosen value of k cannot contribute any information to the *de Bruijn* graph, while assemblies based on small values for k cannot resolve repetitive regions.

The program `MADAM` was developed to overcome these problems. It was presented at the German Conference on Bioinformatics (GCB) 2016 in Berlin and was later published in *PeerJ* under the title "Improving ancient DNA genome assembly" (Seitz and Nieselt, 2017).

5.2 Two-layer assembly

The idea of `MADAM` is to combine two different assembly approaches, the *de Bruijn* graph and the overlap layout consensus (OLC) approach. The generation of the final assembly is divided into two layers. A program based on *de Bruijn* graphs is used in the first layer to generate multiple assemblies using different k-mers. These assemblies are then combined in the second layer using a program based on the OLC approach. Fig. 5.1 on the facing page shows the different steps used in this idea, which are implemented in `MADAM`.

`MADAM` uses the tool `Clip&Merge` (Peltzer *et al.*, 2016) for the preprocessing of the raw reads in the first layer. By default, it removes the adapters and low-quality bases with a Phred score of < 20. Afterward, overlapping forward and reverse reads are merged and

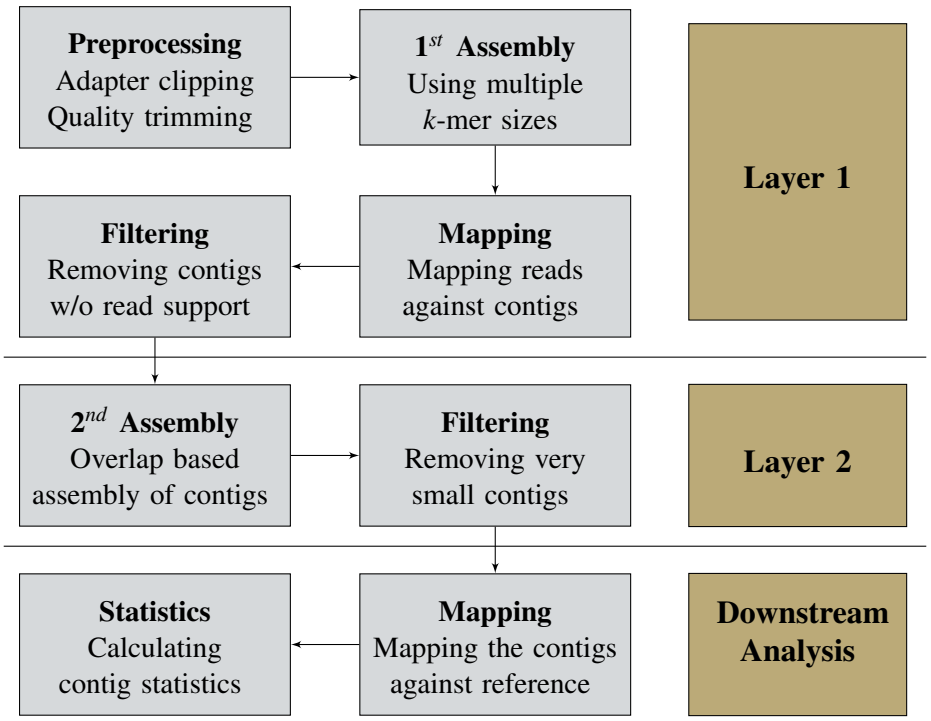


Figure 5.1: Workflow of MADAM. In a first layer, the raw input reads are preprocessed, and an assembly program based on *de Bruijn* graphs is used to generate multiple *de novo* assemblies using different *k*-mer sizes. The reads are then mapped against the assembled contigs to identify and remove contigs stemming from random overlaps of the *k*-mers. These filtered contigs are combined and then used in a new assembly based on the OLC approach. Short reads are removed before the resulting contigs are mapped against the corresponding reference genome to identify the relevant contigs. Finally, different assembly statistics are calculated.

combined with all reads that could not be merged. The resulting reads are then used in a single-end assembly. It is also possible to skip the merging and use the adapter-clipped and quality-trimmed reads in a paired-end assembly. Because of the quality-trimming, it is possible that some paired-end reads lose their corresponding mate. As assembly programs have problems with reads that do not have a mate when performing a paired-end assembly, reads without a partner are removed. To still use these reads, it is possible to combine all adapter-clipped and quality-trimmed reads in a single file that is then used to create a single-end assembly. Finally, all reads that are shorter than 25 bp are removed. These preprocessing methods will be called *merged*, *paired*, and *single*, for the rest of the chapter.

Depending on the preprocessing method, the resulting reads differ in length. These different read lengths stem mainly from the different fragment lengths contained in the sample. The two-layer assembly approach of MADAM addresses problems stemming from

these different read lengths. In the first layer, the reads are assembled using a *k*-mer based assembly program. MADAM incorporates the assembly programs SOAPdenovo2 (Luo *et al.*, 2012) and MEGAHIT (Li *et al.*, 2014), but any other program with variable *k*-mer sizes can be used. To be able to cover a broad range of *k*-mers that represent short, as well as long reads, the default setting for MADAM is to use ten different values for *k*: 37,47,57,...,127.

Due to the nature of *de Bruijn* graph assemblies, it is possible that contigs are created that stem from random overlaps of the *k*-mers and do not represent actual biological sequences (Myers, 2005). To remove these contigs, the preprocessed reads are mapped back against all contigs generated by the previous assemblies. Contigs without read support, i.e., contigs where no read mapped against, are removed.

The second layer assembly uses the assembled and filtered contigs generated in the first layer as input. All contigs are assigned a unique identifier before they are combined into one file. An assembly program based on string overlaps instead of *k*-mers, a concept initially introduced by Myers (2005), then uses the combined file as input. The String Graph Assembler (SGA) developed by Simpson and Durbin (2012) is based on this concept and is used in the second layer of MADAM. The graph generated by this assembler is not based on overlapping *k*-mers but on the overlaps of the input sequences, which are computed using suffix arrays (Manber and Myers, 1993).

To correct for sequencing errors, SGA provides different correction steps. However, here the input of SGA are already assembled contigs.

Thus, the sequencing errors should already have been averaged out by the previous assemblies (Schatz *et al.*, 2010), making further correction steps implemented in SGA largely unnecessary. MADAM only uses two corrections provided by SGA to remove duplicate information. First, it uses the duplicate removal step. Second, it uses the Ferragina Manzini (FM) index (Ferragina and Manzini, 2000) of SGA for the merging of unambiguously overlapping input sequences. Afterward, SGA assembles the resulting input using the default parameters of SGA on the remaining input sequences. Finally, short contigs are removed from the output. By default, MADAM removes all contigs shorter than 1000 bp.

All parameters and their default values for the DACCOR pipeline are described in supplementary Table B.6 on page 95.

5.3 Downstream Analysis

To assess the quality of the two-layer assembly, different assembly statistics, like the N50 (see Section 2.4.3 on page 15), are calculated. Additionally, the contigs stemming from organisms other than the desired one need to be removed. Afterward, the resulting contigs can be used to identify genomic rearrangements.

To analyze one organism out of a metagenomic sample, the contigs that stem from this organism need to be separated from the ones stemming from other organisms. For this, BWA MEM is used to map all contigs generated in the second layer of MADAM against

Table 5.1: Detailed information about the samples used in the experiments of MADAM.

Sample Name	Bacteria	No. reads	Mean coverage	mean fragment length	Archive ID
Jorgen507	*M. leprae*	2x 18,366,205	10.0	89.2	ERR1883823
Jorgen625L	*M. leprae*	2x 15,101,591	102.6	173.5	SRR847052
Jorgen625S	*M. leprae*	2x 6,751,711	49.3	88.1	SRR847050
Airaku3	*M. leprae*	2x 44,187,648	44.7	108.7	SRR1132821
OBS137	*Y. pestis*	2x120,018,142	279.5	79.4	ERR1193536
IND1	*T. pallidum*	2x 6,307,464	578.9	100.9	SRR3268698

the respective reference genome. The contigs that mapped against this genome are then extracted and used for further analysis.

This mapping can also be used to identify genomic variations, deletions, and rearrangements. A contig that is split into multiple parts, which are then all mapped to different locations on the reference genome, indicates a genomic rearrangement, an insertion in the reference genome, or a deletion in the newly sequenced sample. If not all parts of a contig can be mapped, then this is also an indication that there was an insertion in the newly sequenced genome or a deletion in the reference genome.

5.4 Results

For the evaluation of MADAM, five different NGS sequencing experiments of three different bacteria were analyzed with MADAM, SPAdes (version 3.11.1) (Bankevich *et al.*, 2012), SOAPdenovo2 (version 1.05) (Luo *et al.*, 2012), MEGAHIT (version 1.04) (Li *et al.*, 2014), and SGA (version 0.10.13) (Simpson and Durbin, 2012). The five sequencing experiments consist of one ancient *Yersinia pestis* (*OBS137*), one modern *Treponema pallidum* (*IND1*), and two ancient (*Jorgen507*, *Jorgen625*) and one modern (*Airaku3*) *Mycobacterium leprae* samples. There are two sequencing experiments available for the sample *Jorgen625*, one containing relatively long DNA fragments (*Jorgen625L*), and one with shorter DNA fragments (*Jorgen625S*), compared to general aDNA sequencing experiments (see Section 2.2 on page 10).

5.4.1 Analysis of the sequencing data

To get an overview of the sequencing experiment for the different samples, they were analyzed with the EAGER pipeline. Detailed information about the samples based on this analysis is shown in Table 5.1. All samples were sequenced with Illumina paired-end sequencing containing between six and 20 million reads. The analysis with the EAGER

pipeline showed that the samples also varied concerning the mean coverage (between 10X and 578X) and their mean fragment length (between 88.1 bp and 173.5 bp). The reference genomes that were used for the mapping of the different bacteria are described in Table 5.2

The read length of the raw FASTQ sequencing files, as well as the impact of the different preprocessing methods described in Section 5.2 on page 58, are illustrated in Fig. 5.2 on the next page. It shows that all samples, except for the two *Jorgen625* ones, were sequenced with a read length of 101 bp. The two *Jorgen625* samples were sequenced with 151 and 200 bp for the short and long fragment library, respectively. Additionally, except for of the modern *M. leprae* sample *Airaku3*, there is no difference in the read length distribution of the *single* and *paired* preprocessing method. As expected, the *merged* method resulted in longer reads, as the overlapping paired-end reads were merged to create longer ones. Interestingly, this merging was possible in all cases, ancient and modern.

To get an overview about the metagenomic composition of the samples, especially the ancient ones, they were analyzed with MALT (version 0.4.0) (Herbig *et al.*, 2016) against the complete bacterial database of all complete genomes contained in NCBI (NCBI, 2018). The resulting compositions were then visualized with MEGAN6 (Huson *et al.*, 2016). The composition on the species level are illustrated in Fig. 5.3 on page 64. As all samples were enriched for the target bacterium, the majority of the analyzed reads stem from the respective bacterium. Nonetheless, there are still reads that stem from other bacteria. This contamination is especially pronounced in the ancient samples. While the ancient *M. leprae* samples show a large amount of *M. leprae* DNA, more than half of the reads of the *Yersinia pestis* sample do not belong to the *Yersinia pestis* bacterium. The modern clinical samples *Airaku3* and *IND1* show only a small number of reads that do not seem to originate from the corresponding reference bacterium. In the case of *IND1* most of them stem from another bacterium from the genus *Treponema* (see Section 5.4.1 on page 64). However, these results only reflect the reads that could be mapped against the bacterial database. Because the samples were all extracted from humans, they were also mapped against the human reference genome *hg19* with the EAGER pipeline. Table 5.3 on page 65 shows the general mapping statistics of the MALT and the two EAGER analyses (against human and the respective bacterium). It shows that not all reads are accounted

Table 5.2: The reference genomes of the respective bacteria that were used for the mapping procedures.

Bacterium	Reference strain	RefSeq ID
M. leprae	*Mycobacterium leprae TN*	NC_002677.1
T. pallidum	*Treponema pallidum subsp. pallidum str. Nichols*	NC_021490.2
Y. pestis	*Yersinia pestis CO92*	NC_003143.1

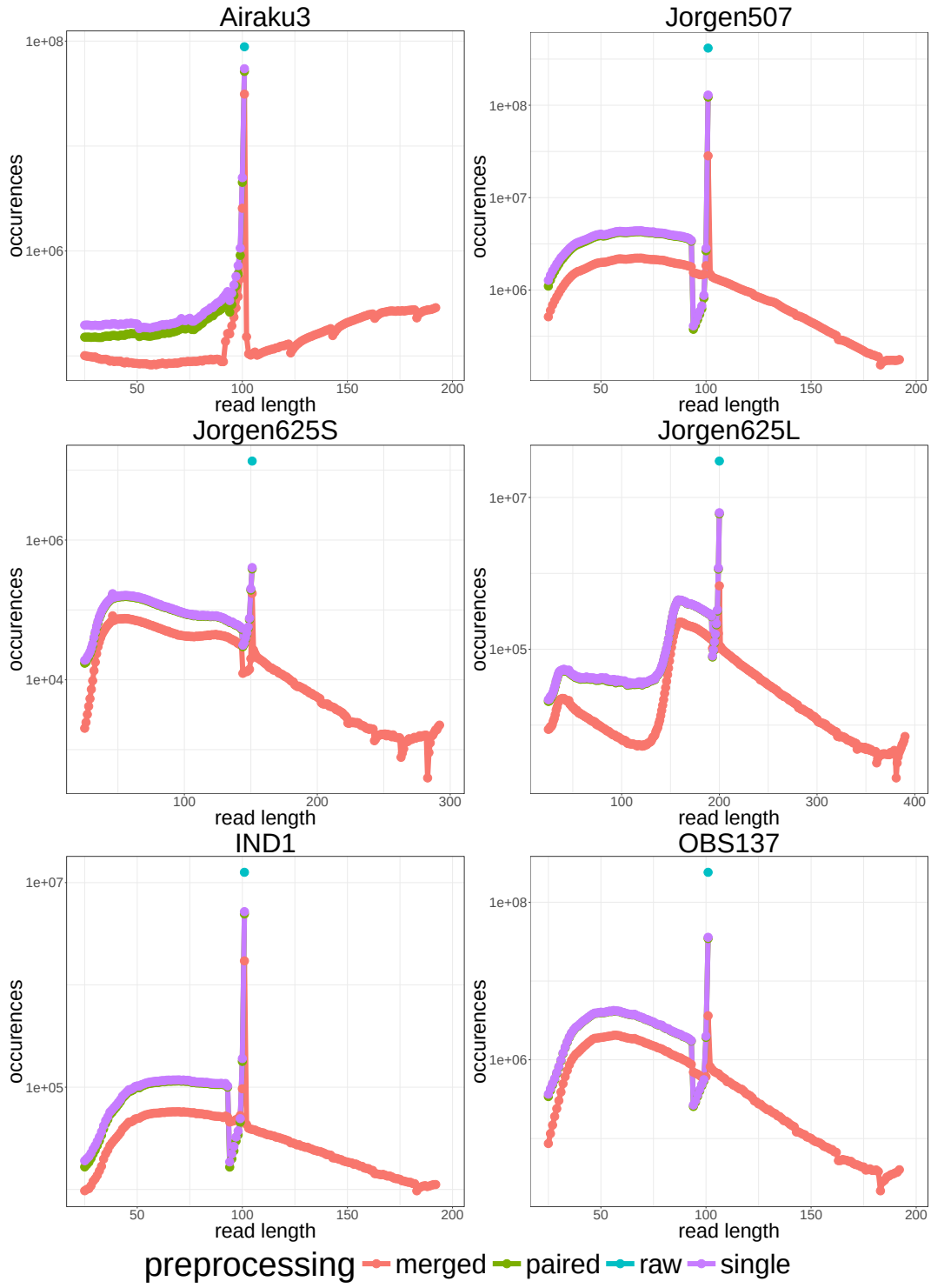
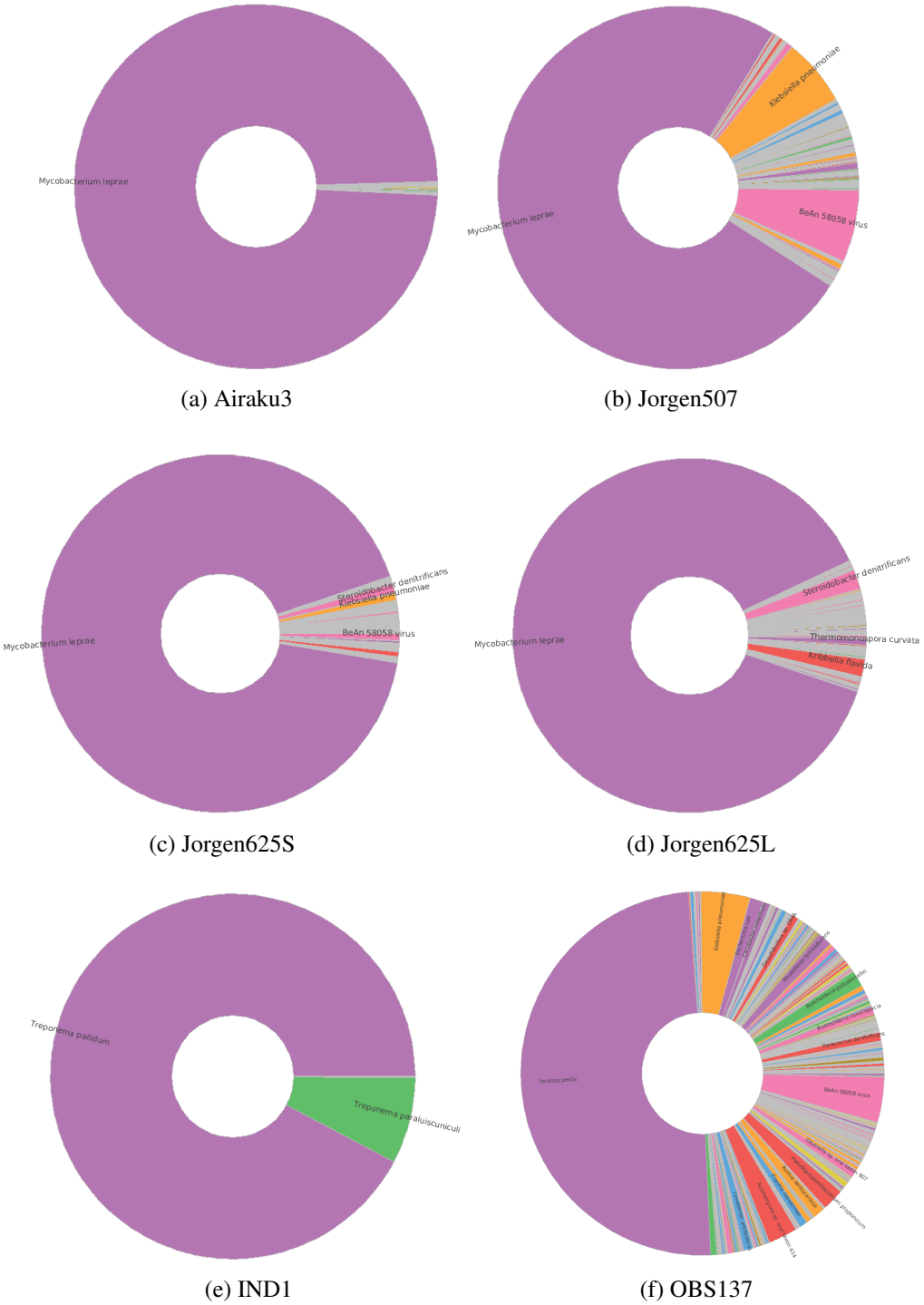


Figure 5.2: The read length distributions of the different preprocessing methods, as well as the length of the raw sequencing reads for the six samples described in Table 5.1 on page 61.



(a) Airaku3 (b) Jorgen507

(c) Jorgen625S (d) Jorgen625L

(e) IND1 (f) OBS137

Figure 5.3: Bacterial composition of the samples calculated with `MALT` and visualized with `MEGAN`.

Table 5.3: Percentage of mapped reads for the analysis with MALT, with EAGER against the respective bacterial reference, and with EAGER against the human reference genome *hg19*.

sample	MALT vs bacterial db % mapped	EAGER vs bacterium % mapped	EAGER vs human % mapped
Airaku3	34.4	24.9	5.6
Jorgen507	18.4	13.1	44.9
Jorgen625S	40.2	34.6	8.6
Jorgen625L	20.7	16.9	1.8
IND1	68.3	69.5	3.7
OBS137	35.0	25.0	39.5

for, even when combining the reads mapped against the bacterial database with the ones mapped against the human genome. Interestingly, for the sample *IND1*, MALT mapped fewer reads against the whole bacterial database than BWA MEM against the respective bacterial reference genome, even though this reference genome is also contained in the database. To identify possible reasons that might be responsible for the low mapping rates, all unmapped reads of the six samples were extracted. Out of them, the first ten reads were analyzed with BLASTN against the whole *nr* database using the NCBI BLAST webservice [1] . While it was not possible to identify a possible source for most of the analyzed reads, for the sample *IND1*, out of the ten analyzed reads, seven resulted in no match, two matched against rabbits and one actually belonged to *T. pallidum*. Fig. 5.4 on page 67 shows the alignment of this read against the *Nichols* reference genome, which contains a deletion of 197 bp.

5.4.2 Assembly benchmark

The general assembly statistics of MADAM with SOAPdenovo2 in the first layer and MADAM with MEGAHIT in the first layer, as well as separate assemblies using SOAPdenovo2, SGA, and MEGAHIT are shown in Table 5.4 on the following page. The samples were also assembled with SPAdes. However, even though SPAdes produced very long contigs, these did not reflect the corresponding bacteria but rather were misassemblies containing parts from multiple species contained in the metagenomic data. On the clinical samples with very little contamination, the results were not misassembled but resulted in very short contigs. Unfortunately, the same was true in some of the assemblies computed using MEGAHIT.

MEGAHIT can also use multiple different *k*-mer sizes to generate the assembly graph.

[1] https://blast.ncbi.nlm.nih.gov/Blast.cgi

Table 5.4: Assembly statistics of contigs ≥1,000 bp that mapped against the respective reference for different assembly programs on the samples described in Table 5.1 on page 61. Only the result of the preprocessing method resulting in the highest N50 score is shown for each sample. Additionally, only the best *k*-mer size/combination for SOAPdenovo2 and MEGAHIT were chosen. If no *k*-mer size is given, the combined version achieved the best result. The number of gaps reflect the number of genomic regions where no contig could be mapped continuously. The values in bold show the best scores. Note that the small number of gaps in SGA are not marked, as there the complete assembly is too small.

sample	assembler	preprocessing	# contigs ≥1000 bp	N50	longest contig	# gaps
Airaku3	SOAP K77	*paired*	641	6,344	29,995	395
	SGA	*merged*	451	1,786	5,929	446
	MEGAHIT	*paired*	229	**25,103**	**105,630**	126
	MADAM MEGAHIT+SGA	*paired*	**1,720**	5,790	26,940	**106**
	MADAM SOAP+SGA	*paired*	783	9,067	51,386	145
Jorgen507	SOAP K77	*paired*	243	25,945	88,625	104
	SGA	*merged*	1,094	2,333	10,869	881
	MEGAHIT	*merged*	1,203	3,732	18,738	531
	MADAM MEGAHIT+SGA	*paired*	**1,886**	14,003	103,702	98
	MADAM SOAP+SGA	*paired*	709	**19,632**	**111,018**	**92**
Jorgen625S	SOAP K77	*single*	299	15,922	41,086	116
	SGA	*merged*	3	1,126	1,633	4
	MEGAHIT	*single*	1,138	3,425	16,693	521
	MADAM MEGAHIT+SGA	*single*	**2,422**	8,125	52,690	91
	MADAM SOAP+SGA	*merged*	363	**23,945**	**134,545**	**83**
Jorgen625L	SOAP K127	*paired*	252	20,302	87,980	131
	SGA		misassembled			
	MEGAHIT	*merged*	785	9,950	58,157	252
	MADAM MEGAHIT+SGA	*merged*	**1,385**	2,458	20,644	982
	MADAM SOAP+SGA	*single*	1,118	**14,441**	**151,696**	**100**
IND1	SOAP K67	*paired*	276	3,691	16,250	186
	SGA	*merged*	193	2,108	7,470	152
	MEGAHIT	*single*	418	2,404	6,919	309
	MADAM MEGAHIT+SGA	*paired*	**508**	6,360	25,317	**35**
	MADAM SOAP+SGA	*single*	278	**8,301**	**56,324**	53
OBS137	SOAP K47	*paired*	1,410	1,734	5,450	1,193
	SGA	*merged*	113	1,136	1,963	114
	MEGAHIT	*merged*	1,660	2,618	11,246	1,062
	MADAM MEGAHIT+SGA	*single*	**3,104**	1,884	11,478	966
	MADAM SOAP+SGA	*merged*	1,952	**2,621**	**14,621**	**811**

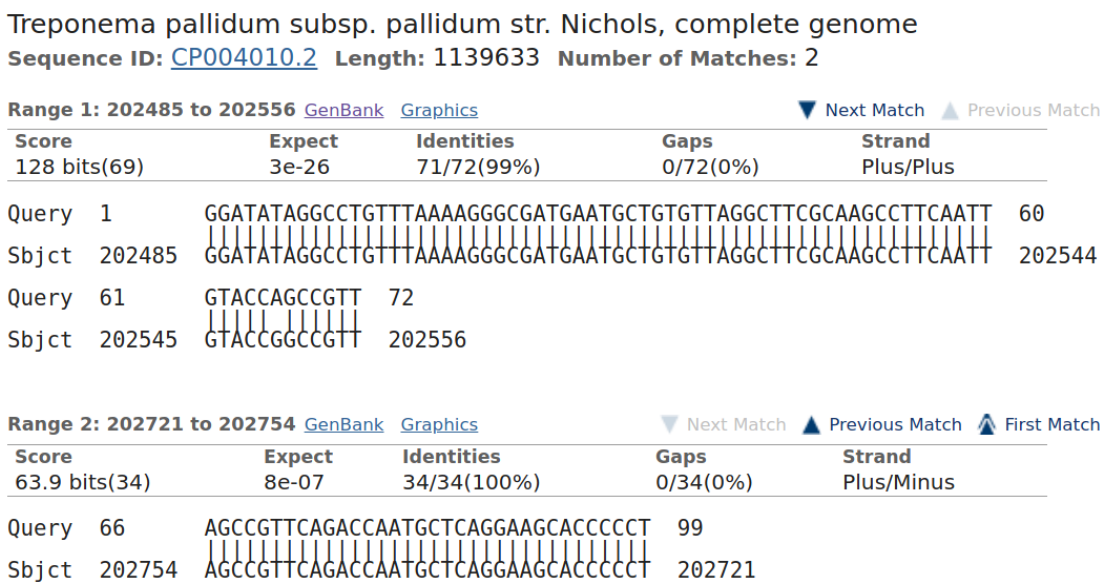


Figure 5.4: BLAST alignment of a read from the *T. pallidum* sample *IND1* that MALT could not align to any genome contained in the bacterial database, but BLAST aligned to the *Nichols* strain of *T. pallidum*. It is aligned to two different genomic locations spanning a deletion of 197 bp.

This combined result of MEGAHIT always performed better than the results using only one individual *k*-mer size. Because of this, only the results of MEGAHIT using multiple *k*-mers are reported. Thus, in the case of the sample *Airaku3*, the result of the two-layer approach with MEGAHIT and SGA is not getting worse, because the single *k*-mer results of MEGAHIT are not shown.

In all cases, the two-layer approach of MADAM resulted in an improvement of the assemblies, compared to using only the assembly program that was used in the first layer alone. Furthermore, MADAM achieved the best assembly results on all bacteria, except for the sample *Airaku3* when comparing the N50 value, as well as the longest generated contig. On the sample *Airaku3*, the best assembly result concerning these two metrics was achieved using MEGAHIT with its combined *k*-mer approach using the *paired* preprocessing method. Interestingly, when using SGA alone, it resulted in the worst assemblies in all samples and comparisons.

When comparing the different preprocessing methods, it is apparent that except for SGA, where the best results were always achieved using the *merged* data, all assembly approaches achieved the best results using the *paired* preprocessing method on the sample *Airaku3*. On all other samples, no single preprocessing method consistently outperformed the others for all assembly methods. Except for the samples *Jorgen507* and *Jorgen625S*, where the *single* and *paired* preprocessing methods never resulted in the best result for any assembly method, every preprocessing method was able to outperform

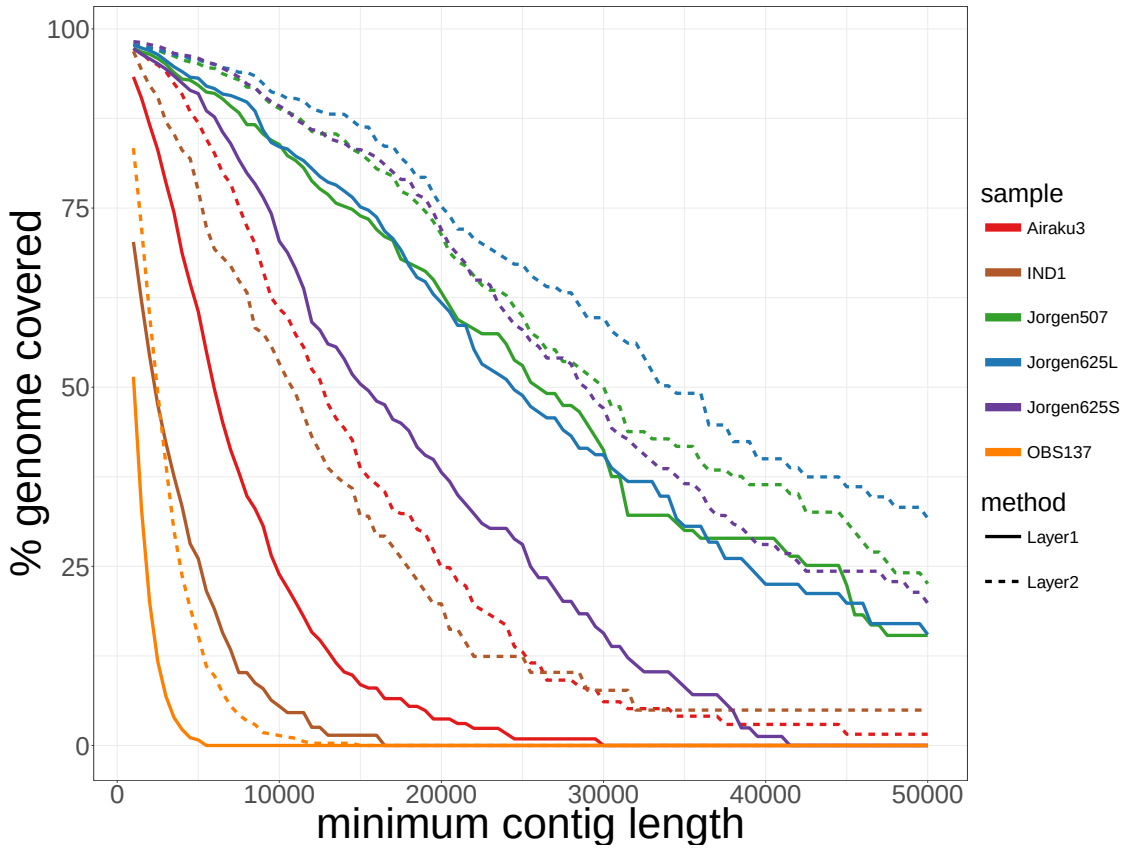


Figure 5.5: Coverage of the respective reference genome with respect to the minimum length of the assemblies generated in the first- and second-layer of `MADAM`. The results of the first layer reflect the same ones as presented in Table 5.4 on page 66. The figure displays minimum contig lengths between 1,000 bp and 50,000 bp, which were incremented by 500 bp in each step.

the others for at least one assembly method. Additionally, the best overall performance for the different samples was also achieved using different preprocessing methods. The *merged* method achieved the best results on the samples *Jorgen625S* and *OBS137*, the *paired* preprocessing on the samples *Airaku3* and *Jorgen507*, and the *single* preprocessing on the samples *IND1* and *Jorgen625L*.

Next, we computed the percentage of the respective reference genome that could be covered when mapping only the contigs of a certain minimum length against it using `BWA MEM` (see Fig. 5.5). For all six analyzed samples, the results of the second layer of `MADAM` always cover more of the respective reference genome compared to the best assembly generated in the first layer. This result gets more pronounced with increasing minimum contig length. The three ancient *M. leprae* samples appear to have resulted in the best assemblies. This is true for the assemblies generated in the first layer, as well as the ones generated in the second layer. Using all contigs of length $\geq 1{,}000$ bp, all

Table 5.5: Statistics about the gaps in the mapping of the assembled contigs against the reference genome of *M. leprae*. They represent the same gaps that are visualized in Fig. 5.6 on the next page. The reference genome contains a total of 122,916 annotated repetitive bases.

sample	Layer	total uncovered	uncovered in repetitive regions	% in repeats	improvement [%] total	improvement [%] repeats
Airaku3	1	218,750	88,342	40.4	53.3	10.7
	2	89,126	78,846	88.5		
Jorgen507	1	88,744	80,972	91.2	20.7	16.4
	2	70,353	67,660	96.2		
Jorgen625S	1	89,219	76,934	86.2	37.7	29.6
	2	55,612	54,198	97.5		
Jorgen625L	1	72,394	68,860	95.1	7.3	4.4
	2	67,130	65,845	98.1		

three of them were able to cover almost 100% of the respective reference genome for both the first- and second-layer assemblies. However, with increasing minimum contig length, the percentage of the genome covered by the assemblies generated in the first layer decreases faster than the ones generated after the second layer. Additionally, the first layer assembly of the sample *Jorgen625* with the short fragment library decreases faster than the long fragment library, whereas the results of the second layer assembly remain competitive. The results of the modern *M. leprae* sample *Airaku3* decreases faster than any of the ancient ones. While the assembly generated in the second layer could also cover almost the complete reference genome when mapping all contigs $\geq 1,000$ bp, the one generated in the first layer could cover significantly less. In both cases, the increasing minimum contig length resulted in a fast decrease of the coverage.

5.4.3 Detailed assembly analysis of the leprosy samples

To identify the regions that remain uncovered in the *M. leprae* samples, the gaps that remain in the mapping of the assembled contigs against the reference genome are illustrated in Fig. 5.6 on the following page. Additionally, repetitive regions that are annotated in the reference genome are shown together with the repetitive regions identified with DACCOR in Section 4.3 on page 45 and summarized in Table 4.2 on page 48. The repetitive regions identified by DACCOR all correspond to regions that are annotated in the reference genome. The remaining gaps mainly coincide with the known repetitive regions, except for the first layer assembly of *Airaku3*, which contains a lot more gaps, as already illustrated in Table 5.4 on page 66. The number of unresolved bases, as well as the ones coinciding with the annotated repetitive regions, are summarized in Table 5.5. This table shows that after the second layer, almost all uncovered bases lie within the an-

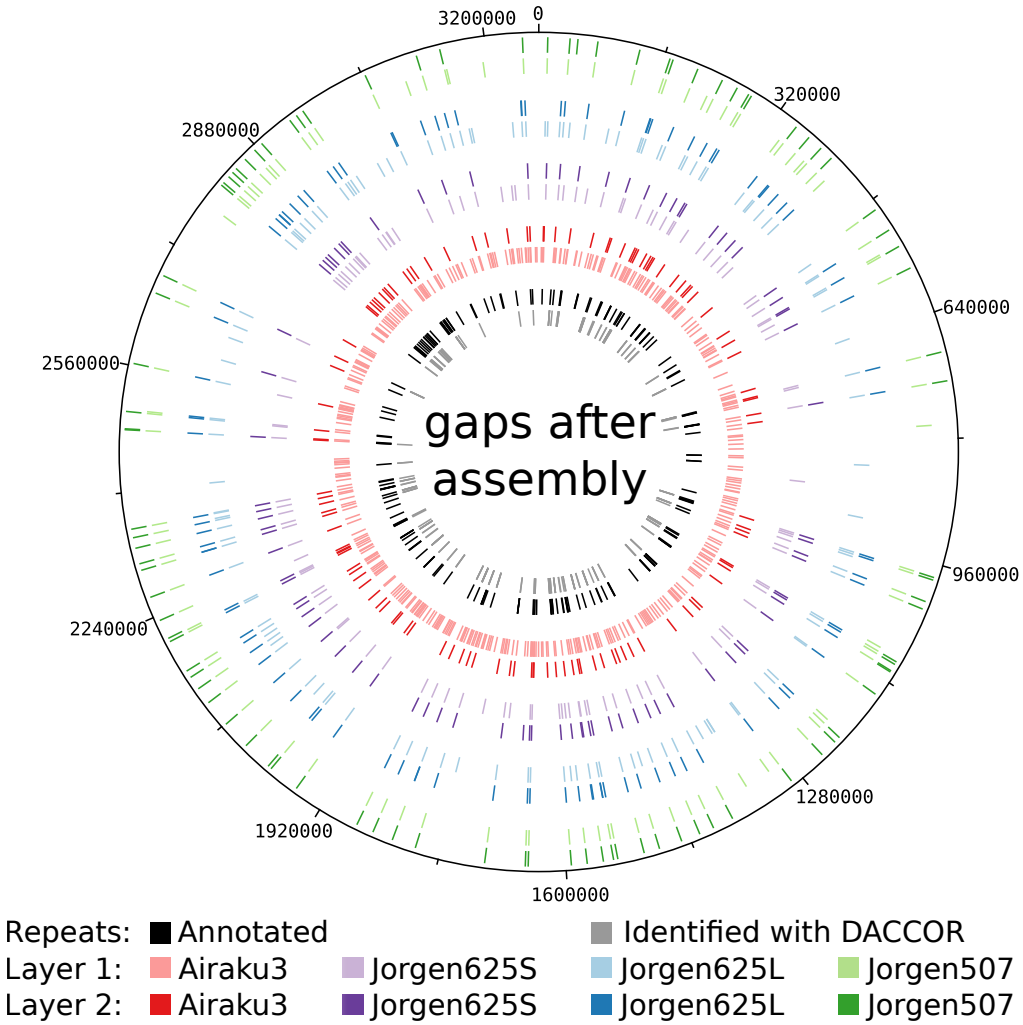


Figure 5.6: Gaps in the mapping of the assembled contigs of the four *M. leprae* samples for both the first layer assemblies (lighter color) and the second layer assemblies (darker color) together with repetitive regions annotated in the reference, as well as identified with DACCOR. There are a total of 122,916 bases annotated as belonging to repetitive regions, while DACCOR identified 76,563.

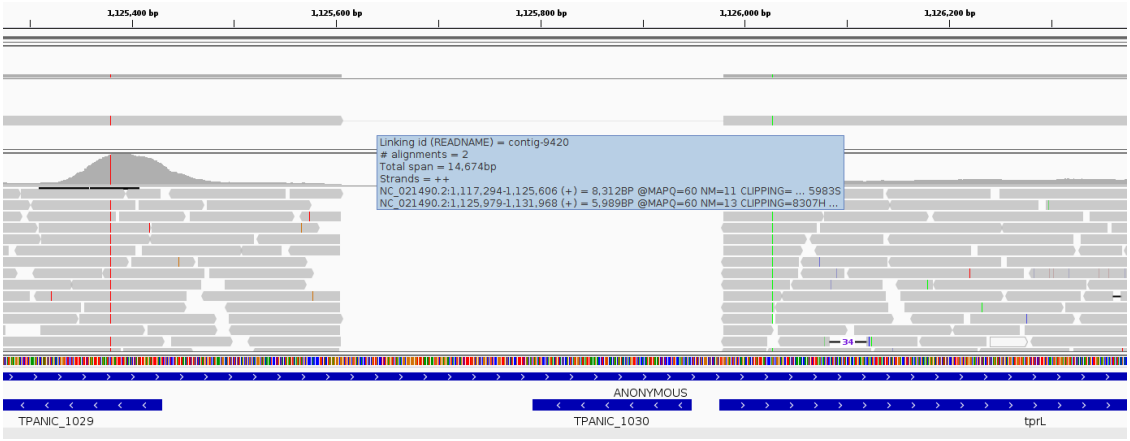


Figure 5.7: Deletion of the gene *TPANIC_1030* in the sample *IND1*, visualized with `IGV` (Robinson *et al.*, 2011). The assembled contig was split into two parts. The linking of the two contigs is also shown.

notated repetitive regions. Additionally, even on the assembly of *Jorgen625L*, where the assembly in the first layer was already excellent, the second layer could further improve it in both the total number of uncovered positions, as well as in the repetitive ones.

Finally, we tried using the assembled contigs to identify possible genomic rearrangements in the samples. To this end, we analyzed the mapping of the assembled contigs against the respective reference genome. Indicative of such rearrangements are contigs where parts belong to two different locations on the reference genome. `BWA MEM` splits these reads to map each part separately to the different locations on the reference genome. Because of the separate mapping, their identifier occurs multiple times in the resulting `BAM` file. Only one of the assembled samples showed any significant indications for genomic rearrangements, namely the *T. pallidum* sample *IND1*. There, the analysis of contigs that mapped to multiple locations hinted at a possible deletion. After a visual inspection with `IGV` (Robinson *et al.*, 2011) of the mapping of the contigs, together with the mapping of the preprocessed reads and the gene annotations, we identified a possible deletion of the gene *TPANCI_1030*. This visualization is depicted in Fig. 5.7. There, the split contig is visualized together with the linking of the two parts. Additionally, it is apparent that the respective deleted part is also not covered by any sequencing read. Furthermore, it is known that this deletion is also present in other *T. pallidum* samples that belong to the same phylogenetic clade (Arora *et al.*, 2016).

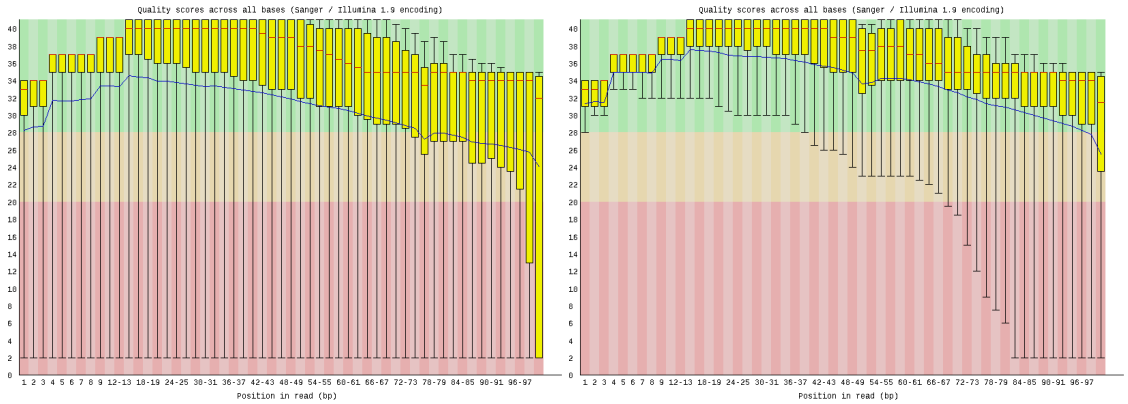
5.5 Discussion and conclusions

`MADAM` is an assembly pipeline designed to improve on assemblies generated from short fragment NGS sequencing experiments and is thus especially suited for, but not limited

to, the assembly of aDNA samples. Its two-layer approach makes use of both *de Bruijn*, as well as *OLC* assembly approaches. As an example, four ancient and two modern sequencing samples from three different bacteria were assembled with MADAM. The two-layer approach of MADAM increased the quality of the assemblies for all samples compared to the standard assembly approach using either *de Bruijn* or *OLC* assembly approaches. MADAM uses different *k*-mer sizes in the first layer using *de Bruijn* graph assemblies, which are then combined in the second layer with an *OLC*-based approach. Due to the preprocessing, reads stemming from short fragment sequencing experiments (see Section 2.4.1 on page 13) are of different lengths (illustrated in Fig. 5.2 on page 63), the use of a single *k*-mer size for the assembly cannot capture the characteristics of these different lengths. Ideally, the size for *k* is set as large as possible to resolve as many repetitive regions as possible, as larger *k*-mers generally result in better assemblies (Utturkar *et al.*, 2014). However, because of the different read lengths, using large *k*-mers would mean that all reads shorter than the chosen value for *k* could not contribute any information to the assembly. On the other hand, using small *k*-mer sizes makes it difficult to assemble some repetitive regions. Thus, using multiple different values for *k* that are then combined using overlaps of the different generated assemblies can make use of both short and long *k*-mers, resulting in a better assembly result.

The three different preprocessing methods introduced in Section 2.4.1 on page 13 result in different read length distributions. The merging of the overlapping forward and reverse reads results in longer reads, which can be beneficial for the resulting assemblies. Apart from the sample *Airaku3*, there is no difference between the read length distribution of the *single* and *paired* preprocessing methods. In the sample *Airaku3*, this difference stems from the sequencing quality of the FASTQ files, illustrated in Fig. 5.8 on the next page. The reverse reads show a typical per-base quality, with lower qualities at the 3' end of the read (Bolger *et al.*, 2014), while the forward reads contain low-quality bases at all positions of the reads. Because the preprocessing trims all bases with a lower quality of a Phred score of 20 and then discards reads that are shorter than 25 base-pairs, more forward than reverse reads are discarded. The *single* preprocessing method then combines all remaining reads to be used in the assembly. However, the *paired* preprocessing method only contains reads that have matching pairs in the forward and reverse reads. Thus, if a forward read is discarded, the corresponding reverse read is also discarded even if the reverse read would pass the quality criteria. This removal is needed, because many assembly programs cannot handle paired-end reads without partners. This discrepancy is the reason for the difference in the read length distribution of the *single* and *paired* preprocessing method for the sample *Airaku3*.

Even though the analyzed samples were of both ancient and modern origin, it was possible to merge the reads in every sample. As already described in Section 2.2 on page 10, DNA degrades over time, which is why ancient samples contain only short DNA fragments. This is not the case for the modern samples. However, they also contain mostly short DNA fragments (see Table 5.1 on page 61). The reason for these short DNA fragments in the modern samples is the use of the DNA capture methods (described in



(a) Forward reads

(b) Reverse reads

Figure 5.8: The Phred Base quality distribution for the forward (a) and reverse (b) reads for the sample *Airaku3*, calculated with `FastQC` (Andrews, 2010).

Section 2.2 on page 10). Because longer fragments are not amplified (García-García *et al.*, 2016), only the shorter fragments are sequenced.

The metagenomic analysis of the samples showed a diverse bacterial composition for the ancient samples and only a few contaminants in the modern clinical samples. The contaminant reads in the *T. pallidum* sample *IND1* were assigned to another bacterium from the genus *Treponema*, namely *Treponema paraluiscuniculi*. In addition to contamination with another *Treponema* species, these reads might hint at an inserted region. Even so, the assembly could not confirm this and further research is needed to confirm or deny this hypothesis. This bacterium causes veneral spirochetosis in rabbits and is closely related to *T. pallidum* (Šmajs *et al.*, 2011). Thus, this contamination might stem from the propagation of the bacterium in rabbits. It might even not be a contamination at all, but may instead represent sequences that actually belong to the genome of this sample but are not contained in the reference sequence. Interestingly it was possible to assign more reads to the single *T. pallidum* reference genome than to the complete bacterial database, which also included this reference. The reason for this seems to be that `MALT`, which was used to align the reads against the bacterial database, cannot handle deletions in the genome, which would result in reads that need to be split to align to the genome (see Fig. 5.4 on page 67). `BWA MEM`, on the other hand, can handle such reads, which is one reason for the increased coverage. However, there remain a large number of reads in all samples for which identifying a closely related reference genome was not possible. Some of these reads, but not all of them, stem from humans, which are the host for the analyzed diseases. The remaining unclassified reads show that there are still aspects in enriched samples, especially in the ancient ones, we do not fully understand. A thorough investigation of the origin of these reads would go beyond the scope of this thesis, but could give insights into the metagenomic composition, as well as hint to possible

technical problems in the sample preparation.

The assembly results in Table 5.4 on page 66 showed that while SGA had problems assembling the raw reads of the metagenomic samples, its best results were always based on the *merged* reads. The reason for this probably lies in its underlying methodology. While the other assembly programs are based on *k*-mers and the resulting *de Bruijn* graph, SGA uses the *OLC* approach, based on the calculated overlaps of the reads. Thus, the overlaps of the longer, merged reads appear to result in less ambiguity in the resulting graph, which may be the reason why SGA performed better on this data.

The two-layer approach of MADAM could improve the assembly on all samples, compared to the default assemblies with a single *k*-mer. This improvement shows that while the different assemblies in the first layer contained all the relevant information, using a single *k*-mer prevented consideration of this information. The different *k*-mer sizes resulted in slightly different assemblies, each containing relevant information. The assembly in the second layer, using SGA with its overlap-based approach, could then combine the different assembly results. Even so, the results of SGA on the raw reads were the worst of all compared assemblies. However, it worked well on the previously assembled contigs. This is most likely because of the increased length of the input. Here, it uses already assembled contigs, which are used to build the overlap graph. These results, together with the fact that SGA performed best on the *merged* reads, furthers the theory that SGA works better on longer input data. The resulting longer contigs achieved a more continuous coverage of the reference genome, showing that this second assembly can make use of the advantages of the different *k*-mer sizes.

The remaining gaps in the mapping of the contigs against the reference genome coincide with annotated repetitive regions. Interestingly, the reference annotation contains more annotated repetitive regions than the *de novo* repeat identification with DACCOR. The reason for this is that they were identified with RepeatMasker, a tool that compares the sequence with known repetitive regions of other genomes. Thus, if a gene is known to be repetitive in another genome, it will be annotated as repetitive even though it is not repetitive in this genome.

The identified deletion of the gene *TPANIC_1030* in the *T. pallidum* sample *IND1* demonstrates the usefulness of the *de novo* assembly. While it would be possible to identify this deletion solely based on the mapping of the short reads, this identification would be more complicated. Due to the large number of reads, as well as their short length, there are 85,916 reads that BWA MEM split in order to map them to two different locations. In comparison, nine assembled contigs had to be analyzed to identify the deletion. The other eight are all split into one very long part and another one containing only 15 to 53 bases. In six of the remaining eight contigs, the short part overlapped with annotated repetitive regions. The splitting of the last two contigs is probably due to sequencing errors, as in both cases other, continuously mapping contigs do not support deletions or rearrangements.

However, the mapping of the contigs against the corresponding reference genome can only identify contigs that overlap at least in some part with this genome. If the contig

belongs to a region that was completely lost, identifying it is not possible. One possibility to recover such a contig could be to use *k*-mer profiles and compare them to the reference genome. One possible machine learning approach is the `gkmSVM` (Ghandi *et al.*, 2016), which uses *k*-mer profiles to classify DNA and protein sequences.

While it was possible to identify the deletion in the *T. pallidum* sample *IND1*, this discovery was still based on a manual, visual screening of interesting regions in the genome. The assembly could minimize the number of these interesting regions. This reduction made it possible to look at all of them, but an automated filtering step that can, for example, remove the split contigs where one part is very short or is contained in another continuously mapping contig could still reduce the number of genomic positions that need to be analyzed.

Furthermore, the results of the second layer depend on the quality of the assemblies generated in the first layer. Even though it was possible to improve the results in all cases, good results in the first layer lead to better results in the second layer. Thus, an automated *k*-mer optimization could improve the results even further. Currently, the default values for *k* span a wide range of values to cover most read lengths. However, this does not mean that they are optimal for all applications. Slight changes could lead to further improvements, depending on the dataset. Additionally, it is possible that on sequencing experiments with shorter read lengths, especially on single-ended sequencing experiments, the longest reads are shorter than the largest value for *k*. In these cases, `MADAM` ignores these values for *k*, as they will result in empty assembly graphs and thus in no assembly result. An optimization could be to not only ignore them but try to use other values for *k* instead to add more information for the assembly in the second layer. One possibility for an automated search for an optimal *k* could be the use of `KmerGenie` (Chikhi and Medvedev, 2014), a tool to calculate the optimal value for *k*. It also reports an abundance histogram of the underlying dataset, which could be used to not only use the optimal value for *k*, but also the other, additional values.

Altogether, `MADAM` can lessen the problem of the different fragment lengths and can create better results than the assemblies based on only a single *k*-mer size.

Chapter 6

SNPViz: Visualizing SNPs

Parts of this chapter were published in:
***A. Seitz**, T. Koch, and K. Nieselt (2017)*
SNPViz- Visualization of SNPs in proteins
Genomics and Computational Biology, p. e100048. ISSN 2365-7154.
https://doi.org/10.18547/gcb.2018.vol4.iss1.e100048

6.1 Introduction

The previous chapters addressed the reconstruction of genomes based on NGS data, as well as the identification of SNVs and genomic rearrangements in these samples. The SNVs are of particular interest, as it is believed that they make up the majority of the genetic and phenotypic differences between humans (Vonholdt *et al.*, 2010). The substitution of a single nucleotide can lead to an incorporation of a different amino acid in the corresponding protein. These SNVs are called non-synonymous SNVs (nsSNVs). A prominent example of a SNV leading to the incorporation of a different amino acid with consequences for the resulting phenotype is the sickle-cell disease (Rees *et al.*, 2010). In this case, the 17^{th} nucleotide in the β-globin gene is mutated from thymine to adenine, which leads to a substitution of the glutamic acid to valine in the sixth amino acid of the β globin chain. The resulting structural change leads to a binding between the $\beta 1$ and $\beta 2$ chains of two hemoglobin molecules. This binding disrupts the architecture and flexibility of the cell and promotes cellular dehydration.

While sequencing projects like the 1000 Genomes project (Auton *et al.*, 2015) identified many nsSNVs, in most cases the impact on the structural changes in the affected proteins is still unknown (Zhao *et al.*, 2014). In Chapter 3 on page 19, the identification of the SNVs was complemented with their annotation using `SNPeff`. While the annotations of `SNPeff` include the affected gene and protein, no information about the phenotypic and pathogenic impact is given. This additional information can be found in specialized databases like ClinVar (Landrum *et al.*, 2018). Ideally, personalized medicine could use the SNVs identified through whole exome sequencing (WES) to identify the origin of diseases. However, although WES can identify several thousand SNVs (Menon *et al.*,

2013), if the causative SNV is not already known, it is not possible to find the SVN that is responsible for the disease.

6.2 Using protein structures to infer the effect of SNPs

While programs like SNPeff (Cingolani *et al.*, 2012) or SIFT (Ng and Henikoff, 2003; Kumar *et al.*, 2009) can predict whether a SNV has an impact on the function of the affected protein, the result only states whether the SNV causes damages or not. There is no confidence score or explanation, which might give the researcher more information about this statement.

To enable researchers to inspect how amino acid exchanges affect the corresponding proteins more effectively, our idea is to visualize putative changes in the three-dimensional protein structure. The central dogma of molecular biology (Crick, 1970) states that a coding SNV can directly affect an amino acid in the corresponding protein. The identification and visualization of the affected amino acid can help to gain insight into possible diseases related to the underlying SNVs. For this, we developed SNPViz, an automated pipeline to identify and highlight amino acids within a protein's three-dimensional structure that are affected by SNVs. This visualization can help researchers to gain insights into the effect of SNVs that have not yet been linked to pathogenic effects.

SNPViz consists of two parts: First, a Java program identifies the affected protein and amino acid for each SNV, after which a JavaScript program visualizes the three-dimensional structure of the proteins with the affected amino acids.

6.2.1 From SNV to structure

The Java program used to identify the amino acids in the affected proteins takes as input the VCF file with the identified SNVs, the reference FASTA and GTF files, as well as the ID-mapping file, generated by Uniprot [1]. Additionally, the GTF file needs to contain a transcript ID for each exon. We recommend using the annotation with the transcript IDs from ENSEMBL (Zerbino *et al.*, 2018). The general idea of the identification of the affected amino acid is depicted in Fig. 6.1 on the facing page. For each identified SNV position in the input VCF file, SNPViz uses the GTF file to identify all exons that are affected by the respective SNV. Based on the identified exons, the corresponding protein transcript IDs, contained in the GTF file, are extracted. The corresponding IDs from the Protein Data Bank (PDB) [2] (Berman, 2000) are then identified using the ID-mapping file if they exist. Afterward, the information about the proteins is downloaded from PDB.

[1] ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/idmapping/idmapping_selected.tab.gz

[2] http://www.rcsb.org/

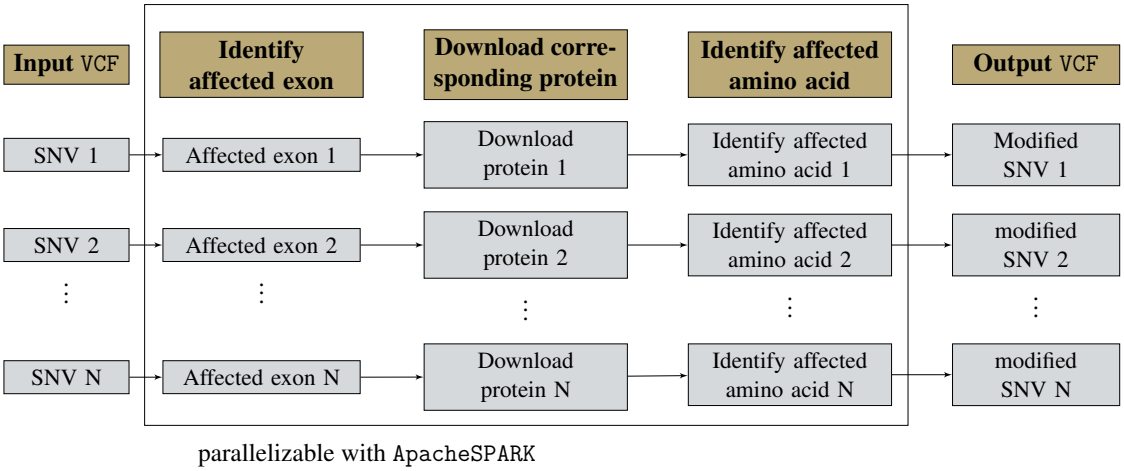


Figure 6.1: Methodology of the identification of the amino acids affected by the SNVs. Each SNV is analyzed separately. The SNVs where a protein structure could be identified are written to one output VCF file. The identification of the SNVs can be parallelized using the ApacheSpark framework (Zaharia *et al.*, 2016).

The PDB contains the three-dimensional structure, as well as the amino acid sequence of the protein.

To identify the location of the SNV in the protein, the affected exon is translated into the corresponding amino acid sequence. For the identification of the correct location, all six possible amino acid sequences (forward and reverse, as well as the three possible reading frames) are generated. These amino acid sequences are then compared to the full amino acid sequence obtained from the PDB to identify the location of the exon within the protein. After the location of the exon has been identified, it is possible to locate the affected amino acid, as well as the effect of the SNV (i.e., the substituted amino acid).

The result of this analysis is a new VCF file that contains the PDB-ID, the position of the affected amino acid, as well as the substituted amino acid in the comment section of each SNV in the VCF file. If it was not possible to identify a three-dimensional structure for the affected gene, or if the SNV does not affect the coding sequence of a protein, the resulting VCF file will not contain this SNV. If there is more than one identified PDB-ID, the output contains all of them, separated with commas.

Because the processing of each SNV is independent of all other SNVs, we parallelized this processing with ApacheSpark (Zaharia *et al.*, 2016).

All parameters and their default values for the identification of the affected amino acid in the three-dimensional structure of SNPViz are described in supplementary Table B.7 on page 96.

Figure 6.2: Initial visualization of SNPViz, with no modified VCF file loaded, showing the protein *Crambin* with the PDB-ID *1CRN*. A VCF file can be loaded using the button "Choose Files", located at the top of the webpage. An example visualization with a loaded VCF file is shown in Fig. 6.3 on the next page

6.2.2 Visualization

To provide an interactive visualization of the SNV in the protein, the affected proteins are downloaded from the PDB and visualized using the JavaScript library *NGL* (Rose *et al.*, 2018). Fig. 6.2 shows the protein *Crambin* (PDB ID 1rcn), loaded without a modified VCF file.

After the visualization page is opened in any web-browser, it is possible to load a VCF file using the file browser on the top of the page (see Fig. 6.2). If this VCF file contains the annotations generated by SNPViz described in Section 6.2.1 on page 78, it is possible to select the desired protein from the drop-down menu of the file browser. Using the JavaScript library *NGL*, the structure is downloaded from the PDB, and subsequently visualized. As this download can take some time and is performed in the background, it is not possible to know when the complete protein is downloaded. Because of this, only the structure of the desired protein is shown, without the amino acid that is affected by the SNV. Fig. 6.3 on the next page shows the visualization of the hemoglobin protein with a loaded VCF file before the amino acid changes are highlighted.

To highlight the affected amino acid, the button "*color Changes*" can be used, which changes the visualization of the protein. The affected amino acid is colored in red, and all other amino acids are colored in white. This visualization is depicted in Fig. 6.4 on

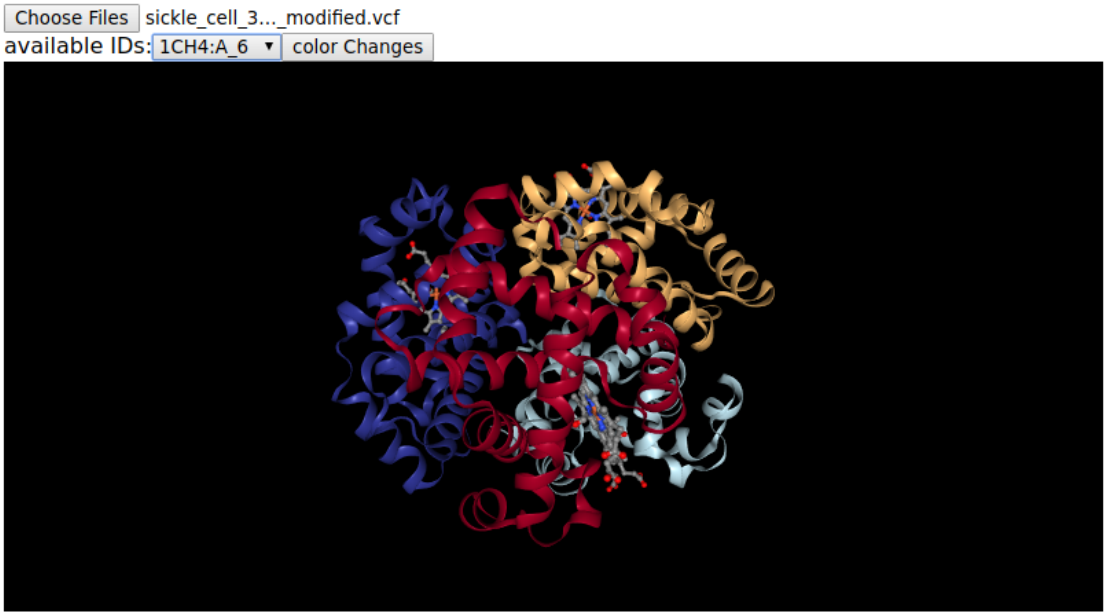


Figure 6.3: Visualization of SNPViz after selecting an entry of the modified VCF file. In this case, the hemoglobin alpha protein with the PDB-ID *1CH4* was selected. SNPViz first shows the structure of the protein without highlighting the affected amino acid. The highlighting is shown in Fig. 6.4 on the next page

the following page.

6.3 Results

For the evaluation of SNPViz, we used the mutation responsible for the sickle-cell disease, described in Section 6.1 on page 77. It is described as the mutation of the seventeenth nucleotide in the β globin gene from thymine (T) to an adenine (A). This mutation results in a substitution of the sixth amino acid of the protein from glutamic acid (GLU) to valine (VAL). Because we use a known mutation for the input of SNPViz and not the result of a human sequencing run that has been mapped against the human reference genome, the input VCF has to be generated manually. Based on the gene annotation file (GFF) [3], published by ENSEMBL (Zerbino *et al.*, 2018), we know that the mutated nucleotide responsible for the sickle-cell disease is located on chromosome 11 at position 5,248,232. The input VCF file for SNPViz is shown in Fig. 6.5 on the following page.

The modified VCF file generated by SNPViz contains the identified three-dimensional structures in the INFO field. It contains the PDB-ID of the three-dimensional structure,

[3] ftp://ftp.ensembl.org/pub/grch37/release-94/gff3/homo_sapiens/Homo_sapiens.GRCh37.87.chr.gff3.gz

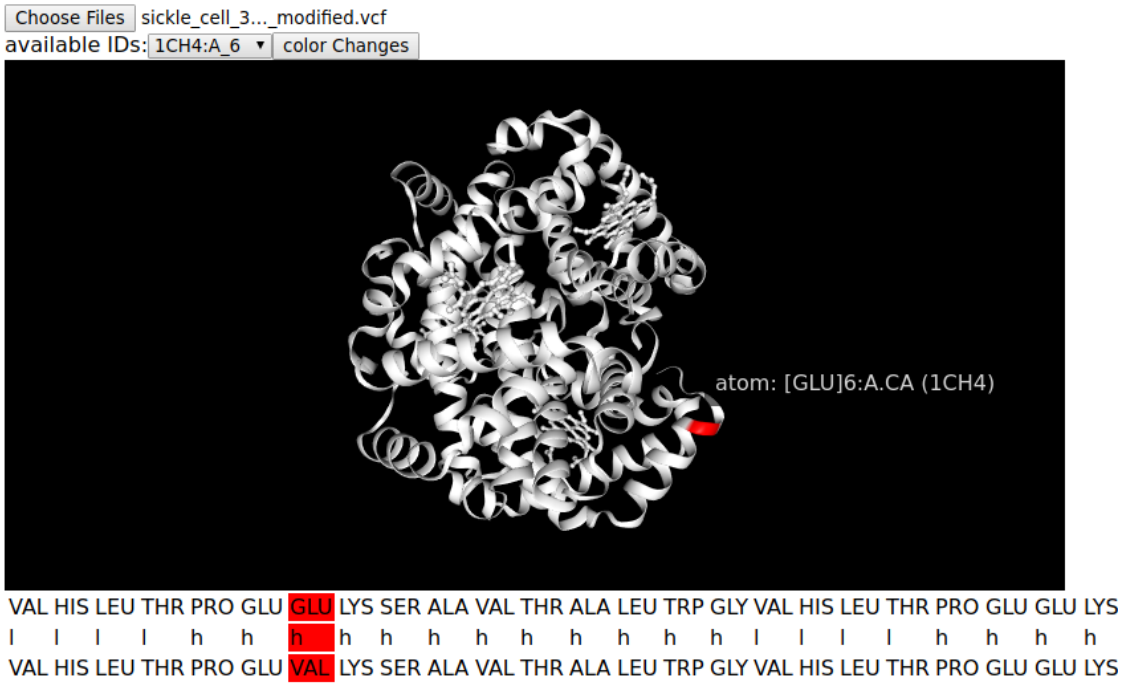


Figure 6.4: Visualization of SNPViz after the *color Changes* button was clicked to highlight the modified amino acid, as well as show information regarding the amino acids and the secondary structure around this modified amino acid.

the position of the substituted amino acid, as well as the substituted amino acid itself. The position of the modified amino acid is separated from the PDB-ID by an underscore (_), and the modified amino acid from the position by an exclamation mark (!). The resulting modified VCF file is shown in Fig. 6.6 on the next page. All in all, it contained 291 different PDB-IDs. However, all of them belong to the hemoglobin protein. Some of them, like the ID *3IC2*, show the structure with specific ligands. In the case of the displayed example with the ID *3IC2*, the ligand is the "potent antisickling agent, INN-266". Furthermore, in all 291 identified structures, the substituted amino acid is a valine at position 6 of the protein.

```
1 ##fileformat=VCFv4.1
2 #CHROM   POS      ID   REF   ALT   QUAL   FILTER   INFO
3 chr11    5248232       T     A     .      .
```

Figure 6.5: Input VCF file for SNPViz showing the mutation responsible for the sickle-cell disease, located on chromosome eleven at position 5,248,232 from thymine (T) to adenine (A).

```
1 ##fileformat=VCFv4.1
2 #CHROM   POS      ID   REF   ALT   QUAL   FILTER  INFO
3 chr11    5248232  .    T     A     .      .       PDB=1CH4:A_6!VAL, ...
```

Figure 6.6: Output VCF file for SNPViz showing the mutation responsible for the sickle-cell disease, located on chromosome eleven at position 5,248,232 from thymine (T) to adenine (A), together with the annotated substituted amino acids. By selecting the highlighted amino acid, additional information about the original amino acid at this position can be obtained.

Fig. 6.4 on the facing page shows the resulting visualization for the protein with the PDB-ID *1CH4* with the affected sixth amino acid highlighted. The table below the visualization shows more details about the substitution. There, it can be seen that the glutamic acid (GLU, top row) was substituted to a valine (VAL, bottom row). The substituted amino acid lies within an α helix (indicated by the "h" in the middle row) of the protein. Additionally, from the visualization of the structure, it is apparent that the substituted amino acid lies on the surface of the protein and could thus interact with other molecules and proteins. Users can also select other amino acids in the three-dimensional structure to obtain more information about them. In this case, the highlighted amino acid was selected. It shows that it is a glutamic acid at the sixth position of the protein. The visualized protein is always the original protein, without the substitution, with the modified amino acid highlighted. Thus, the information about the selected amino acid belong to the original amino acid and not the substituted one.

6.4 Discussion and conclusions

SNPViz is a program to visualize the amino acids affected by a SNV in the corresponding three-dimensional structure of the corresponding protein. For SNVs for which currently no information on pathogenic effects or disease-relations is available in established databases, the visualization with SNPViz can provide additional information to researchers, e.g., if the mutation is on the protein surface or in specific secondary structure elements.

In our validation example, we showed that SNPViz could identify known mutations and their resulting amino acid substitution correctly. Even so, it is difficult to surmise the effect of a substituted amino acid. The substitution of a proline, a known α-helix and β-sheet breaker (Li *et al.*, 1996), inside of an α-helix or β-sheet will probably destroy this secondary structure and thus may lead to a loss of function of this protein. The substitution of a hydrophilic amino acid in the hydrophobic area of transmembrane pump proteins will probably also lead to a loss of function. Across different substitutions, however, the assumptions about their possible effects are not as straightforward.

In the example of the sickle-cell disease, which was shown in Section 6.3 on page 81, the substituted amino acid creates an additional active site at the surface of the protein so that two hemoglobin proteins bind together. Even with expert knowledge about protein structures and binding behavior, it would be difficult to predict such an outcome. Furthermore, examples like the *dmd* gene, which codes for dystrophin has a huge number of identified SNVs that are known to have no impact on the protein, even if they result in the substitution of an amino acid (Aartsma-Rus *et al.*, 2006). Nevertheless, when studying the cause of an unknown disease, `SNPViz` can help researchers to select mutations which are likely to affect the function of proteins to reduce the number or prioritize SNVs for further research.

The presented results focused solely on human data. We also tried to visualize SNVs in bacterial proteins. However, this was not successful. The problem lies in the structures contained in the PDB. For many bacterial proteins, only the structure of one bacterium is contained in the database. Unfortunately, the sequences of homologous proteins between the bacteria differ, which currently makes it impossible to identify the position of the amino acid affected by a SNV.

Future developments of `SNPViz` could include highlighting of the active sites of the proteins in the visualization or the table below, as well as outputting the complete modified amino acid sequence. This addition could help researchers to identify SNVs that may change the active function of a protein or to use the modified sequence when trying to predict the protein folding of the modified protein. However, while providing such output can help identify SNVs that are more likely to result in a change or loss of function of the protein, a single SNV can still have numerous effects on a protein. Thus, while `SNPViz` is a tool that can help researchers prioritize which SNVs and proteins to study further, the space of three-dimensional changes caused by SNVs is currently too large and complicated to predict to be able to use `SNPViz` or other prediction algorithms for semi-automated medical diagnoses of genetic diseases.

Chapter 7

Conclusion and Outlook

NGS data analysis is a fast-growing and diverse field of research and can help answer research questions in areas ranging from the identification and evolution of pathogenic bacteria (Arora *et al.*, 2016), over clinical analysis of human genomes for the identification of genetic defects and their treatment (Collins *et al.*, 2017), to the reconstruction of complete genomes from extinct species (Green *et al.*, 2010). Furthermore, extended sequencing protocols like Chip-Seq (Park, 2009) allow the analysis of additional research questions that are not only sequence-based, such as questions dealing with DNA-protein interactions. These examples show but a fraction of the possibilities made available through NGS sequencing technologies.

The first part of this thesis (Chapter 3 on page 19) presented `MUSIAL`, a program for the generation and comparison of genomes using mapping-based reconstruction methods. This program allows for the generation of MSAs based on whole genomes, on genes, and the generation of MSAs based solely on the variations of the samples without the need for calculating whole-genome alignments. `MUSIAL` creates an extended reference genome containing all insertions of all samples so that the generated MSA contains the indels of each individual sample. Additionally, statistical values like the number of SNVs are summarized and a table containing all SNVs for all samples is generated. It is also possible to annotate the functional impact of the SNVs in the SNV table using `SNPeff`. Furthermore, `MUSIAL` was designed to recover genomic regions with low coverages without calling too many SNVs based on sequencing artifacts. One current drawback of `MUSIAL` is its relatively high memory consumption. To generate the SNV table and the alignments, `MUSIAL` needs to hold all reconstructed genomes in memory, together with additional quality data for each base of each genome. In the example of the 169 leprosy samples, `MUSIAL` needed approximately 80 gigabytes of memory. One possibility to reduce this memory consumption would be to generate the whole genome for each sample and only hold the variant positions in memory. When generating the SNV table, the non-variant positions could be read from the already reconstructed genomes. While this method would not need to hold all genomes in memory, the total runtime of the program would increase, as the desired bases would have to be read from different files.

While the focus of `MUSIAL` lies in the comparison of several reconstructed genomes,

the next part of this thesis, (Chapter 4 on page 39) focused on reconstructing repetitive regions using mapping-based methods, and thus, on improving the quality and resolution of the actual reconstructed genomes using the program DACCOR. Therefore, it actually addresses the step prior to MUSIAL. Repetitive regions longer than the sequenced read length often remain unresolved, because the reads could be mapped to multiple locations on the genome. While DACCOR cannot fully reconstruct all repetitive regions, we showed that it can significantly reduce the number of unresolved bases. This improvement makes it possible to include the sequences of these regions in phylogenetic analyses or even identify SNVs that are only present in one copy of the repetitive region. However, these reconstructions are still solely based on the sequence of the reference genome. Thus, they do not reflect the copy-number variation of these regions. While it might be possible to calculate the copy-number variation using other tools like CNVHumter (Sturm, 2018), DACCOR can only generate the corresponding sequences. However, as each repetitive region is reconstructed separately and the files resulting from the different mapping steps are available, it might be possible to estimate the copy-number variations by comparing the sequencing depth of the different samples. The next step in the development of DACCOR would be its integration into MUSIAL. It is currently not possible, as MUSIAL uses VCF files as input, and DACCOR produces FASTA files as output. While it would be possible to allow for FASTA files to be used as input for MUSIAL, these reconstructed regions would have to be reconstructed without indels, as the positions of the already reconstructed bases could otherwise not be directly associated with the common reference genome. This extension would allow for an increased resolution in the sequences generated by MUSIAL and, thus, would hopefully lead to better phylogenies.

To identify larger genomic rearrangements, insertions, or deletions, the mapping-based reconstruction methods MUSIAL and DACCOR employ (which are described in Chapter 3 on page 19 and Chapter 4 on page 39) are insufficient. Therefore, MADAM was developed and described in Chapter 5 on page 57. We showed that MADAM is able to overcome one of the main challenges of assembling ancient DNA reads, the different read lengths stemming from the fragmented DNA. The next big problem is that even after using DNA capture methods, we are still assembling a metagenomic sample. This metagenomic sample can lead to misassemblies originating from similar sequences of two different species contained in the sample (Olson *et al.*, 2017). One possible solution could be to first run a metagenomic analysis of the sample using, for example, MALT (Herbig *et al.*, 2016) and MEGAN (Huson *et al.*, 2016). However, it is not possible to remove all reads that do not stem from the target bacterium, as the ancient sample could have contained genes that are no longer present in the modern one, but in other species contained in the sample. Thus, based on the modern bacterium and other knowledge on the evolution of the species, the filtering could be performed based on the species, genus, or maybe even family level. A concrete filtering procedure would have to depend on the species, as well as the age of the sample. However, the removal of reads stemming from distantly related species will probably result in a better assembly, because reads that do not belong to the target organism do not have to be analyzed.

Finally, Chapter 6 on page 77 dealt with the analysis of SNVs. Currently, the function of many SNVs is unknown, and SNPViz can help to prioritize SNVs that might be involved in certain diseases. However, it is based on the three-dimensional structures contained in the PDB. While the number of available protein structures contained in the PDB grows rapidly[1], the main focus appears to lie on the analysis of human proteins. A possibility to overcome the problem of visualizing homologous, bacterial proteins could be to use different identification approaches, for example, BLAST for the identification of the location of the SNV within the protein. Even so, the diverging DNA and amino acid sequences could result in ambiguities regarding the affected amino acid. Furthermore, this calculation would increase the runtime of SNPViz drastically, as this program would have to be called for each SNV and identified protein. With increased research into the three-dimensional structure of different proteins, SNPViz, or at least its underlying idea may become more and more relevant. Nonetheless, it needs to be noted that SNPViz cannot directly identify the (genetic) origin of a particular disease. Instead, it can help researchers in identifying possible targets for further research.

Altogether, this thesis described several novel methods and programs for the analysis of NGS data. While the Chapters 3, 4, and 5 focused mainly on the analysis of (ancient) bacterial genomics, the focus of Chapter 6 lay on the analysis of human data. The introduction of new sequencing technologies, like the long reads of Nanopore sequencing, might seem to make the reconstruction of repetitive regions or genomic rearrangements using NGS data unnecessary. However, these technologies cannot be applied in the case of aDNA and even some clinical bacterial samples. Thus, the analysis using NGS data will stay relevant at least until the introduction of fourth-generation sequencing technologies or other technological advancements, which may or may not resolve some of the current sequencing problems.

[1] https://www.rcsb.org/stats/growth/overall accessed: May 29, 2019

Appendix A

Abbreviations

	Genetic code (after IUPAC)
A	Adenine
C	Cytosine
G	Guanine
T	Thymine
U	Uracil)
R	A or G
Y	C or T
S	G or C
W	A or T
K	G or T
M	A or C
B	C or G or T
D	A or G or T
H	A or C or T
V	A or C or G
N	any base (or unresolved)
-	gap
	General abbreviations
aDNA	ancient DNA
bp	base pair(s)
DNA	deoxyribonucleic acid
dNTP	Deoxynucleotide
GATK	Genome Analysis Toolkit
HLA	Human Leukocyte Antigen
indel	Insertion and/or Deletion
IGV	Integrative Genomics Viewer
LBA	Long Branch Attraction
MSA	Multiple Sequence Alignment
NGS	Next-Generation Sequencing
nsSNV	non-synonymous Single Nucleotide Variation

kb	kilobase
OLC	Overlap Layout Consensus
PCR	polymerase chain reaction
PacBio	Pacific Biosciences
PDB	Protein Data Bank
RNA	Ribonucleic acid
SGA	String Graph Assembler
SMRT	Single Molecule Real Time
SNP	Single Nucleotide Polymorphism
SNV	Single Nucleotide Variation
tRNA	transfer RNA
UDG	Uracil-DNA Glycosylase
WES	Whole Exome Sequencing
WGA	Whole Genome Alignment

Appendix B

Supplementary Material

Table B.1: The different input parameters available for MUSIAL.

Parameter	Description	Default value
-ar,–addReference < *arg* >	Add reference under given name to alignment	
-d,–compare < *arg* >	calculate differences compared to other SNVTable	
-c,–coverage < *arg* >	Minimum coverage to make call	5.0
-ca,–covAdd < *arg* >	Minimum coverage to make additional call	5.0
-e,–exclude < *arg* >	Exclude given positions from SNV-Alignment	
-f,–frequency < *arg* >	Minimum frequency to call a SNV	0.9
-g,–gff < *arg* >	The GFF file	
-gn,–geneNames < *arg* >	The gene names for gene alignments	
-h,–help	Show this help page	
-i,–input < *arg* >	The EAGER output folder used as input	
-if,–inputFile < *arg* >	The input VCF files from file	
-il,–inputList < *arg* >	The input VCF files as list	
-o,–output < *arg* >	The output folder	
-ogf,–outgroupFile < *arg* >	VCF File containing outgroup	
-ogn,–outgroupName < *arg* >	Name for outgroup (contained in input)	
-p,–phylogenetics	Calculate a phylogenetic tree	
-r,–reference < *arg* >	The reference FASTA file	
-s,–sharedAlleleFreq	Write shared allele frequencies for each position	
-sf,–snpEff	Run SNPEff	
-t,–threads < *arg* >	Number of threads to use [1]	
-u,–uncovered	Write uncovered positions for each sample	
-y,–heterozygous < *arg* >	Call heterozygous positions	0.45-0.55

Table B.2: The different input parameters available for the `identify` subprogram of DACCOR.

Parameter	Description	Default value
-g,–gff < *arg* >	Path of GFF file	
-h,–help	Prints options	
-i,–input < *arg* >	input filename	
-k,–kmersize < *arg* >	size of initial *k*-mer	read length/2 OR 17
-m,–mismatches < *arg* >	number of mismatches allowed	0
-M,–margin < *arg* >	Margin around repeat to extract	0
-o,–output < *arg* >	output filename	
-p,–processes < *arg* >	Number of threads per genome	1
-rl,–readlength < *arg* >	read length [17]	
-S,–separately	Analyze each sequence separately	
-t,–threshold < *arg* >	Min length of displayed results	read length OR 51
-v,–vmatch < *arg* >	don't analyze but parse vmatch output	
-ws,–writeSeparately	write entries into separate files	

Table B.3: The different input parameters available for the `reconstruct` subprogram of DACCOR.

Parameter	Description
-e,–eager < *arg* >	config file for reconstructed samples already reconstructed with EAGER
-f,–fastqs < *arg* >	config file for fastq files
-h,–help	Prints options
-i,–input < *arg* >	input filename (reference)
-o,–output < *arg* >	output folder
-r,–repeats < *arg* >	config file for repeat input files

Table B.4: The different input parameters available for the `combine` subprogram of DACCOR.

Parameter	Description
-g,–genomes < *arg* >	config file for reconstructed genomes
-h,–help	Prints options
-o,–output < *arg* >	output folder
-r,–repeats < *arg* >	config file for reconstructed repeats

Table B.5: The different input parameters available for the `pipeline` subprogram of `DACCOR`.

Parameter	Description	Default value
-e,–eager $< arg >$	config file for reconstructed samples already reconstructed with EAGER	
-f,–fastqs $< arg >$	config file for fastq files	
-g,–gff $< arg >$	Path of GFF file	
-h,–help	Prints options	
-i,–input $< arg >$	input filename	
-k,–kmersize $< arg >$	size of initial k-mer	read length/2 OR 17]
-m,–mismatches $< arg >$	number of mismatches allowed	0
-M,–margin $< arg >$	Margin around repeat to extract	0
-o,–output $< arg >$	output filename	
-p,–processes $< arg >$	Number of threads per genome	1
-rl,–readlength $< arg >$	read length	17
-S,–separately	Analyze each sequence separately	
-t,–threshold $< arg >$	Min length of displayed results	read length OR 51
-v,–vmatch $< arg >$	don't analyze but parse `VMatch` output	

Table B.6: The different input parameters available for the MADAM pipeline.

-a,–assembly < *arg* >	The assembly algorithm of the first Layer: SOAP, MEGAHIT	[SOAP]
-f,–forward < *arg* >	the forward adapter	[AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC]
-h,–help	show this help page	
-in1,–input1 < *arg* >	the forward and reverse fastq files	
-in2,–input2 < *arg* >	the forward and reverse fastq files	
-k,–kmer < *arg* >	the Kmers to use	[37,47,...,127]
-l,–filterlength < *arg* >	the minimum length of a contig to keep [1000]	
-m,–minFilter < *arg* >	the minimum number of reads that have to map against a contig to keep	[1]
-n,–no_merge	Do not merge the reads	
-o,–output < *arg* >	the output Directory	
-q,–quality < *arg* >	Minimum base quality for quality trimming	[20]
-r,–reverse < *arg* >	the reverse adapter	[AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA]
-R,–reference < *arg* >	the reference genome file	
-s,–insertSize < *arg* >	the insert size of the input	[20]
-S,–single	don't merge and combine for single end assembly (implies -n)	
-t,–threads < *arg* >	number of threads to use	[1]

Table B.7: The different input parameters available for SNPViz.

Parameter	Description	Default value
-d,–id $< arg >$	ID type for the genes	[EnsemblTRS]
-f,–fieldname $< arg >$	The name of the gtf info field containing the ID	[transcript id]
-g,–gtf $< arg >$	the gtf file	
-h,–help	show this help page	
-k,–keep	keep reference in Memory	
-m,–idmap $< arg >$	the id mapping file	
-o,–output $< arg >$	output vcf file	$[< INPUT >$_modified.vcf]
-r,–ref $< arg >$	the fasta reference file	
-s,–spark	run with apacheSpark parallelization	
-v,–vcf $< arg >$	the input vcf file to analyze	

Bibliography

Aartsma-Rus, A., Van Deutekom, J. C., Fokkema, I. F., Van Ommen, G.-J. B., and Den Dunnen, J. T. (2006). Entries in the Leiden Duchenne muscular dystrophy mutation database: an overview of mutation types and paradoxical cases that confirm the reading-frame rule. *Muscle & Nerve: Official Journal of the American Association of Electrodiagnostic Medicine*, **34**(2), 135–144.

Aho, A. V., Hopcroft, J. E., and Ullman, J. D. (1976). On finding lowest common ancestors in trees. *SIAM Journal on computing*, **5**(1), 115–132.

Allentoft, M. E., Sikora, M., Sjögren, K. G., Rasmussen, S., Rasmussen, M., Stenderup, J., Damgaard, P. B., Schroeder, H., *et al.* (2015). Population genomics of Bronze Age Eurasia. *Nature*, **522**(7555), 167–172.

Andrews, S. (2010). FastQC: a quality control tool for high throughput sequence data.

Arora, N., Schuenemann, V. J., Jäger, G., Peltzer, A., Seitz, A., Herbig, A., Strouhal, M., Grillová, L., *et al.* (2016). Origin of modern syphilis and emergence of a pandemic Treponema pallidum cluster. *Nature Microbiology*, **2**(December), 16245.

Ashton, P. M., Nair, S., Dallman, T., Rubino, S., Rabsch, W., Mwaigwisya, S., Wain, J., and O'Grady, J. (2015). MinION nanopore sequencing identifies the position and structure of a bacterial antibiotic resistance island. *Nature Biotechnology*, **33**(3), 296–302.

Assuncao, A. G. L., Herrero, E., Lin, Y.-F., Huettel, B., Talukdar, S., Smaczniak, C., Immink, R. G. H., van Eldik, M., *et al.* (2010). Arabidopsis thaliana transcription factors bZIP19 and bZIP23 regulate the adaptation to zinc deficiency. *Proceedings of the National Academy of Sciences*, **107**(22), 10296–10301.

Auton, A., Abecasis, G. R., Altshuler, D. M., Durbin, R. M., Bentley, D. R., Chakravarti, A., Clark, A. G., Donnelly, P., *et al.* (2015). A global reference for human genetic variation. *Nature*, **526**(7571), 68–74.

Avila-Arcos, M. C., Cappellini, E., Romero-Navarro, J. A., Wales, N., Moreno-Mayar, J. V. V., Rasmussen, M., Fordyce, S. L., Montiel, R., *et al.* (2011). Application and comparison of large-scale solution-based DNA capture-enrichment methods on ancient DNA. *Sci. Rep.*, **1**, 74.

Bankevich, A., Nurk, S., Antipov, D., Gurevich, A. a., Dvorkin, M., Kulikov, A. S., Lesin, V. M., Nikolenko, S. I., *et al.* (2012). SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing. *Journal of Computational Biology*, **19**(5), 455–477.

Benjak, A., Avanzi, C., Singh, P., Loiseau, C., Girma, S., Busso, P., Fontes, A. N., Miyamoto, Y., *et al.* (2018). Phylogenomics and antimicrobial resistance of the leprosy bacillus Mycobacterium leprae. *Nature Communications*, **9**(1).

Bentley, D. R., Balasubramanian, S., Swerdlow, H. P., Smith, G. P., Milton, J., Brown, C. G., Hall, K. P., Evers, D. J., *et al.* (2008). Accurate whole human genome sequencing using reversible terminator chemistry. - Supplement. *Nature*, **456**(7218), 53–9.

Berman, H. M. (2000). The Protein Data Bank. *Nucleic Acids Research*, **28**(1), 235–242.

Boetzer, M., Henkel, C. V., Jansen, H. J., Butler, D., and Pirovano, W. (2011). Scaffolding pre-assembled contigs using SSPACE. *Bioinformatics*, **27**(4), 578–579.

Bolger, A. M., Lohse, M., and Usadel, B. (2014). Trimmomatic: A flexible trimmer for Illumina sequence data. *Bioinformatics*, **30**(15), 2114–2120.

Bos, K. I., Harkins, K. M., Herbig, A., Coscolla, M., Weber, N., Comas, I., Forrest, S. A., Bryant, J. M., *et al.* (2014). Pre-Columbian mycobacterial genomes reveal seals as a source of New World human tuberculosis. *Nature*, **514**(7253), 494–497.

Bos, K. I., Herbig, A., Sahl, J., Waglechner, N., Fourment, M., Forrest, S. A., Klunk, J., Schuenemann, V. J., *et al.* (2016). Eighteenth century Yersinia pestis genomes reveal the long-term persistence of an historical plague focus. *eLife*, **5**, e12994.

Bradnam, K. R., Fass, J. N., Alexandrov, A., Baranay, P., Bechner, M., Birol, I., Boisvert, S., Chapman, J. A., *et al.* (2013). Assemblathon 2: Evaluating de novo methods of genome assembly in three vertebrate species. *GigaScience*, **2**(1), 1–31.

Branton, D., Deamer, D. W., Marziali, A., Bayley, H., Benner, S. A., Butler, T., Di Ventra, M., Garaj, S., *et al.* (2008). The potential and challenges of nanopore sequencing. *Nature Biotechnology*, **26**(10), 1146–1153.

Briggs, A. W., Stenzel, U., Meyer, M., Krause, J., Kircher, M., and Pääbo, S. (2010). Removal of deaminated cytosines and detection of in vivo methylation in ancient DNA. *Nucleic acids research*, **38**(6), 1–12.

Cardona, G., Rosselló, F., and Valiente, G. (2008). Extended Newick: It is time for a standard representation of phylogenetic networks. *BMC Bioinformatics*, **9**, 1–8.

Carvalho, A. B., Dupim, E. G., and Goldstein, G. (2016). Improved assembly of noisy long reads by k-mer validation. *Genome Research*, **26**(12), 1710–1720.

Cha, S. and Bird, D. M. (2016). Optimizing k-mer size using a variant grid search to enhance de novo genome assembly. *Bioinformation*, **12**(2), 36–40.

Chao, R., Yuan, Y., and Zhao, H. (2015). Recent advances in DNA assembly technologies. *FEMS Yeast Research*, **15**(1), 1–9.

Chikhi, R. and Medvedev, P. (2014). Informed and automated k-mer size selection for genome assembly. *Bioinformatics*, **30**(1), 31–37.

Cingolani, P., Platts, A., Wang, L. L., Coon, M., Nguyen, T., Wang, L., Land, S. J., Lu, X., and Ruden, D. M. (2012). A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3. *Fly*, **6**(2), 80–92.

Collins, D. C., Sundar, R., Lim, J. S., and Yap, T. A. (2017). Towards Precision Medicine in the Clinic: From Biomarker Discovery to Novel Therapeutics. *Trends in Pharmacological Sciences*, **38**(1), 25–40.

Crick, F. W. (1970). Central Dogma of Molecular Biology. *Nature*, **227**, 561.

Critchlow, D. E. and Pearl, D. K. (1996). the Triples Distance for Rooted Bifurcating Phylogenetic Trees. *Syst. Biol*, **45**(3), 323–334.

Danecek, P., Auton, A., Abecasis, G., Albers, C. A., Banks, E., DePristo, M. A., Handsaker, R. E., Lunter, G., *et al.* (2011). The variant call format and VCFtools. *Bioinformatics*, **27**(15), 2156–2158.

Darling, A. E., Mau, B., and Perna, N. T. (2010). Progressivemauve: Multiple genome alignment with gene gain, loss and rearrangement. *PLoS ONE*, **5**(6).

Darwin, C. (1859). On the origin of species by means of natural selection. *Murray, London*.

De Bruijn, N. G. (1946). A combinatorial problem. *Koninklijke Nederlandse Akademie v. Wetenschappen*, **49**(49), 758–764.

Dekker, C. (2007). Solid-state nanopores. *Nature Nanotechnology*, **2**(4), 209–215.

Denoeud, F. and Vergnaud, G. (2004). Identification of polymorphic tandem repeats by direct comparison of genome sequence from different bacterial strains: a web-based resource. *BMC bioinformatics*, **5**, 4.

Der Sarkissian, C., Allentoft, M. E., Ávila-Arcos, M. C., Barnett, R., Campos, P. F., Cappellini, E., Ermini, L., Fernández, R., *et al.* (2015). Ancient genomics. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, **370**(1660), 20130387.

Dinh, H. and Rajasekaran, S. (2011). A memory-efficient data structure representing exact-match overlap graphs with application for next-generation DNA assembly. *Bioinformatics*, **27**(14), 1901–1907.

Drummond, A. J. and Rambaut, A. (2007). BEAST: Bayesian evolutionary analysis by sampling trees. *BMC Evolutionary Biology*, **7**(1), 1–8.

Durai, D. A. and Schulz, M. H. (2016). Informed kmer selection for de novo transcriptome assembly. *Bioinformatics*, **32**(11), 1670–1677.

Earl, D., Bradnam, K., St. John, J., Darling, A., Lin, D., Fass, J., Yu, H. O. K., Buffalo, V., *et al.* (2011). Assemblathon 1: A competitive assessment of de novo short read assembly methods. *Genome Research*, **21**(12), 2224–2241.

Earl, D., Nguyen, N., Hickey, G., and Harris, R. (2014). Alignathon: A competitive assessment of whole genome alignment methods. *Genome research*, **24**, 2077–2089.

Eckert, S. E., Chan, J. Z.-M., Houniet, D., Consortium2, t. P., Breuer, J., and Speight, G. (2016). Enrichment of long DNA fragments from mixed samples for Nanopore sequencing. *bioRxiv*, (https://doi.org/10.1101/048850).

Estabrook, G. F., McMorris, F. R., and Meacham, C. A. (1985). Comparison of Undirected Phylogenetic Trees Based on Subtrees of Four Evolutionary Units. *Systematic Biology*, **34**(2), 193–200.

Evin, A., Flink, L. G., B l escu, A., Popovici, D., Andreescu, R., Bailey, D., Mirea, P., Laz r, C., *et al.* (2015). Unravelling the complexity of domestication: a case study using morphometrics and ancient DNA analyses of archaeological pigs from Romania. *Philosophical Transactions of the Royal Society B: Biological Sciences*, **370**(1660), 20130616.

Ewing, B., Ewing, B., Hillier, L., Hillier, L., Wendl, M. C., Wendl, M. C., Green, P., and Green, P. (1998). Base-Calling of Automated Sequencer Traces Using. *Genome Research*, **8**(3), 175–185.

Felsenstein, J. (1978). Cases in which Parsimony or Compatibility Methods will be Positively Misleading. *Systematic Biology*, **27**(4), 401–410.

Ferragina, P. and Manzini, G. (2000). Indexing Compressed Text. *Journal of the ACM*, **52**(4), 552–581.

Frith, M. C., Hamada, M., and Horton, P. (2010). Parameters for accurate genome alignment. *BMC bioinformatics*, **11**(1), 80.

García-García, G., Baux, D., Faugère, V., Moclyn, M., Koenig, M., Claustres, M., and Roux, A. F. (2016). Assessment of the latest NGS enrichment capture methods in clinical context. *Scientific Reports*, **6**(January), 1–8.

Gardner, S. N. and Hall, B. G. (2013). When whole-genome alignments just won't work: KSNP v2 software for alignment-free SNP discovery and phylogenetics of hundreds of microbial genomes. *PLoS ONE*, **8**(12).

Ghandi, M., Mohammad-Noori, M., Ghareghani, N., Lee, D., Garraway, L., and Beer, M. A. (2016). GkmSVM: An R package for gapped-kmer SVM. *Bioinformatics*, **32**(14), 2205–2207.

Gnerre, S., Maccallum, I., Przybylski, D., Ribeiro, F. J., Burton, J. N., Walker, B. J., Sharpe, T., Hall, G., *et al.* (2011). High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proceedings of the National Academy of Sciences of the United States of America*, **108**(4), 1513–8.

Gnirke, A., Melnikov, A., Maguire, J., Rogov, P., Leproust, E. M., Brockman, W., Fennell, T., Giannoukos, G., *et al.* (2009). Solution Hybrid Selection with Ultra-long Oligonucleotides for Massively Parallel Targeted Sequencing. *Nature biotechnology*, **27**(2), 182–189.

Gonnella, G. and Kurtz, S. (2012). Readjoiner: a fast and memory efficient string graph-based sequence assembler. *BMC Bioinformatics*, **13**(1).

Gordon, A. and Hannon, G. J. (2010). Fastx-toolkit. FASTQ/A short-reads pre-processing tools. *Unpublished http://hannonlab.cshl.edu/fastx_toolkit*.

Graham, S. W., Olmstead, R. G., and Barrett, S. C. H. (1998). Rooting Phylogenetic Trees with Distant Outgroups : A Case Study from the Commelinoid Monocots. **19**(10), 1769–1781.

Green, R. E., Krause, J., Briggs, A. W., Maricic, T., Stenzel, U., Kircher, M., Patterson, N. J., Li, H., *et al.* (2010). A Draft Sequence of the Neandertal Genome - Supporting Information. *Science*, **328**(5979), 710–722.

Gut, I. G. (2013). New sequencing technologies. *Clinical and Translational Oncology*, **15**(11), 879–881.

Haak, W., Lazaridis, I., Patterson, N., Rohland, N., Mallick, S., Llamas, B., Brandt, G., Nordenfelt, S., *et al.* (2015). Massive migration from the steppe was a source for Indo-European languages in Europe. *Nature*, **522**(7555), 207–211.

Hall, B. (2011). Phylogenetic trees made easy: How to manual. pages 68–89.

Harper, K. N., Liu, H., Ocampo, P. S., Steiner, B. M., Martin, A., Levert, K., Wang, D., Sutton, M., and Armelagos, G. J. (2008). The sequence of the acidic repeat protein (arp) gene differentiates venereal from nonvenereal Treponema pallidum subspecies, and the gene has evolved under strong positive selection in the subspecies that causes syphilis. *FEMS Immunology and Medical Microbiology*, **53**(3), 322–332.

Hartigan, J. A. (1973). Minimum Mutation Fits to a Given Tree. *Biometrics*, **29**, 53–65.

Heather, J. M. and Chain, B. (2016). The sequence of sequencers: The history of sequencing DNA. *Genomics*, **107**(1), 1–8.

Herbig, A., Jäger, G., Battke, F., and Nieselt, K. (2012). GenomeRing: Alignment visualization based on SuperGenome coordinates. *Bioinformatics*, **28**(12), 7–15.

Herbig, A., Maixner, F., Bos, K. I., Zink, A., Krause, J., and Huson, D. H. (2016). MALT: Fast alignment and analysis of metagenomic DNA sequence data applied to the Tyrolean Iceman. *bioRxiv*, page 050559.

Hernandez, D., François, P., Farinelli, L., Dohm, J. C., Lottaz, C., Borodina, T., Butler, J., Maccallum, I., *et al.* (2008). De novo bacterial genome sequencing : Millions of very short reads assembled on a desktop computer De novo bacterial genome sequencing : Millions of very short reads assembled on a desktop computer. *Genome research*.

Hodges, E., Xuan, Z., Balija, V., Kramer, M., Molla, M. N., Smith, S. W., Middle, C. M., Rodesch, M. J., *et al.* (2007). Genome-wide in situ exon capture for selective resequencing. *Nature genetics*, **39**(12), 1522–7.

Hofreiter, M., Jaenicke-Despres, V., Serre, D., von Haeseler, A., and Pääbo, S. (2001). DNA sequences from multiple amplifications reveal artifacts induced by cytosine deamination in ancient DNA. *Nucleic Acids Research*, **29**(23), 4793–4799.

Holley, R. W., Apgar, J., Everett, G. A., Madison, J. T., Marquisee, M., Merrill, S. H., Penswick, J. R., and Zamir, A. (1965). Structure of a ribonucleic acid. *Science*, **2**, 1462–1465.

Huson, D. H., Beier, S., Flade, I., Górska, A., El-Hadidi, M., Mitra, S., Ruscheweyh, H. J., and Tappu, R. (2016). MEGAN Community Edition - Interactive Exploration and Analysis of Large-Scale Microbiome Sequencing Data. *PLoS Computational Biology*, **12**(6), 1–12.

Illumina (2009). Preparing 2 — 5kb Samples for Mate Pair Library Sequencing. Technical report.

Illumina (2010). Illumina sequencing technology. Technical report.

Karlsson, E., Lärkeryd, A., Sjödin, A., Forsman, M., and Stenberg, P. (2015). Scaffolding of a bacterial genome using MinION nanopore sequencing. *Scientific Reports*, **5**(CLiC), 1–8.

Kasianowicz, J. J., Brandin, E., Branton, D., and Deamer, D. W. (1996). Characterization of individual polynucleotide molecules using a membrane channel. *Proceedings of the National Academy of Sciences*, **93**(24), 13770–13773.

Kehlmaier, C., Barlow, A., Hastings, A. K., Vamberger, M., Paijmans, J. L. A., Steadman, D. W., Albury, N. A., Franz, R., Hofreiter, M., and Fritz, U. (2017). Tropical ancient DNA reveals relationships of the extinct Bahamian giant tortoise *Chelonoidis alburyorum*. *Proceedings of the Royal Society B: Biological Sciences*, **284**(1846), 20162235.

Khan, A. A., un Nabi, S. R., and Iqbal, J. (2013). Surface estimation of a pedestrian walk for outdoor use of power wheelchair based robot. *Life Science Journal*, **10**(3), 1697–1704.

Koch, P., Platzer, M., and Downie, B. R. (2014). RepARK - de novo creation of repeat libraries from whole-genome NGS reads. *Nucleic Acids Research*, **42**(9), e80.

Koren, S., Walenz, B. P., Berlin, K., Miller, J. R., Bergman, N. H., and Phillippy, A. M. (2017). Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *Genome research*, **27**(5), 72–736.

Kozlov, A. (2017). RAxML-NG v0.5.0 BETA (Version 0.5.0). *zenodo*.

Krause-Kyora, B., Nutsua, M., Boehme, L., Pierini, F., Pedersen, D. D., Kornell, S. C., Drichel, D., Bonazzi, M., *et al.* (2018). Ancient DNA study reveals HLA susceptibility locus for leprosy in medieval Europeans. *Nature Communications*, **9**(1).

Krawczak, M., Ball, E. V., Fenton, I., Stenson, P. D., Abeysinghe, S., Thomas, N., and Cooper, D. N. (2000). Human Gene Mutation Database - A biomedical information and research resource. *Human Mutation*, **15**(1), 45–51.

Kumar, P., Henikoff, S., and Ng, P. C. (2009). Predicting the effects of coding non-synonymous variants on protein function using the SIFT algorithm. *Nature Protocols*, **4**(7), 1073–1082.

Kumar, S., Stecher, G., and Tamura, K. (2016). MEGA7: Molecular Evolutionary Genetics Analysis Version 7.0 for Bigger Datasets. *Molecular biology and evolution*, **33**(7), 1870–1874.

Kurtz, S. (2003). The Vmatch large scale sequence analysis software. *Ref Type: Computer Program*, **412**.

Lamarck, J.-B.-P. A. (1809). Philosophie zoologique.

Lander, E. S., Linton, L. M., Birren, B., Nusbaum, C., Zody, M. C., Baldwin, J., Devon, K., Dewar, K., *et al.* (2001). Initial sequencing and analysis of the human genome. *Nature*, **409**(6822), 860–921.

Landrum, M. J., Lee, J. M., Benson, M., Brown, G. R., Chao, C., Chitipiralla, S., Gu, B., Hart, J., *et al.* (2018). ClinVar: Improving access to variant interpretations and supporting evidence. *Nucleic Acids Research*, **46**(D1), D1062–D1067.

Langmead, B. and Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. *Nat Methods*, **9**(4), 357–359.

Laurin-Lemay, S., Brinkmann, H., and Philippe, H. (2012). Origin of land plants revisited in the light of sequence contamination and missing data. *Current Biology*, **22**(15), R593–R594.

Li, D., Liu, C. M., Luo, R., Sadakane, K., and Lam, T. W. (2014). MEGAHIT: An ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. *Bioinformatics*, **31**(10), 1674–1676.

Li, H. (2011). A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics*, **27**(21), 2987–2993.

Li, H. (2013). Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. *arXiv preprint arXiv:1303.3997*.

Li, H. and Durbin, R. (2009). Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, **25**(14), 1754–1760.

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., and Durbin, R. (2009). The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, **25**(16), 2078–2079.

Li, J., Stein, D., McMullan, C., Branton, D., Aziz, M. J., and Golovchenko, J. A. (2001). Ion-beam sculpting at nanometre length scales. *Nature*, **412**(6843), 166–169.

Li, R., Zhu, H., Ruan, J., Qian, W., Fang, X., Shi, Z., Li, Y., Li, S., *et al.* (2010). De novo assembly of human genomes with massively parallel short read sequencing. *Genome Research*, **20**(2), 265–272.

Li, S.-C., Goto, N. K., Williams, K. A., and Deber, C. M. (1996). Alpha-helical, but not beta-sheet, propensity of proline is determined by peptide environment. *Proceedings of the National Academy of Sciences*, **93**(13), 6676–6681.

Li, Z., Chen, Y., Mu, D., Yuan, J., Shi, Y., Zhang, H., Gan, J., Li, N., *et al.* (2012). Comparison of the two major classes of assembly algorithms: Overlap-layout-consensus and de-bruijn-graph. *Briefings in Functional Genomics*, **11**(1), 25–37.

Libura, M. (1991). Sensitivity analysis for minimum Hamiltonian path and traveling salesman problems. *Discrete Applied Mathematics*, **30**(2-3), 197–211.

Lindow, M. and Krogh, A. (2005). Computational evidence for hundreds of non-conserved plant microRNAs. *BMC genomics*, **6**, 119.

Liu, K., Linder, C. R., and Warnow, T. (2010). Multiple sequence alignment: a major challenge to large-scale phylogenetics. *PLoS Currents*, **2**.

Loman, N. J., Quick, J., and Simpson, J. T. (2015). A complete bacterial genome assembled de novo using only nanopore sequencing data. *Nature Methods*, **12**(8), 733–735.

Luo, R., Liu, B., Xie, Y., Li, Z., Huang, W., Yuan, J., He, G., Chen, Y., *et al.* (2012). SOAPdenovo2: an empirically improved memory-efficient short-read de novo assembler. *GigaScience*, **1**(1), 18.

Madoui, M. A., Engelen, S., Cruaud, C., Belser, C., Bertrand, L., Alberti, A., Lemainque, A., Wincker, P., and Aury, J. M. (2015). Genome assembly using Nanopore-guided long and error-free DNA reads. *BMC Genomics*, **16**(1), 1–11.

Mamanova, L., Coffey, A. J., Scott, C. E., Kozarewa, I., Turner, E. H., Kumar, A., Howard, E., Shendure, J., and Turner, D. J. (2010). Target-enrichment strategies for next-generation sequencing. *Nature Methods*, **7**(2), 111–118.

Manber, U. and Myers, G. (1993). Suffix Arrays: A New Method for On-Line String Searches. *SIAM Journal on Computing*, **22**(5), 935–948.

Maricic, T., Whitten, M., and Pääbo, S. (2010). Multiplexed DNA sequence capture of mitochondrial genomes using PCR products. *PLoS ONE*, **5**(11), 9–13.

Marra, C. M., Sahi, S. K., Tantalo, L. C., Godornes, C., Reid, T., Behets, F., Rompalo, A., Klausner, J. D., *et al.* (2010). Enhanced Molecular Typing of *Treponema pallidum* : Geographical Distribution of Strain Types and Association with Neurosyphilis. *The Journal of Infectious Diseases*, **202**(9), 1380–1388.

McKenna, A., Hanna, M., Banks, E., Sivachenko, A., Cibulskis, K., Kernytsky, A., Garimella, K., Altshuler, D., *et al.* (2010). The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Research*, **20**(9), 1297–1303.

Medvedev, P., Georgiou, K., Myers, G., and Brudno, M. (2007). Computability of Models for Sequence Assembly. In *Algorithms in Bioinformatics*, pages 289–301, International Workshop on Algorithms in Bioinformatics. Springer.

Menon, R., Deng, M., Rüenauver, K., Queisser, A., Pfeifer, M., Offermann, A., Boehm, D., Vogel, W., *et al.* (2013). Somatic copy number alterations by whole-exome sequencing implicates YWHAZ and PTK2 in castration-resistant prostate cancer. *Journal of Pathology*, **231**(4), 505–516.

Mitchell, K. J., Llamas, B., Soubrier, J., Rawlence, N. J., Worthy, T. H., Wood, J., Lee, M. S., and Cooper, A. (2014). Ancient DNA reveals elephant birds and kiwi are sister taxa and clarifies ratite bird evolution. *Science*, **344**(6186), 898–900.

Mitjà, O., Šmajs, D., and Bassat, Q. (2013). Advances in the Diagnosis of Endemic Treponematoses: Yaws, Bejel, and Pinta. *PLoS Neglected Tropical Diseases*, **7**(10).

Morrison, D. (2012). Who published the first phylogenetic tree? *http://phylonetworks.blogspot.com/2012/08/who-published-first-phylogenetic-tree.html*, (accessed: May 29, 2019).

Mothershed, E. A. and Whitney, A. M. (2006). Nucleic acid-based methods for the detection of bacterial pathogens: Present and future considerations for the clinical laboratory. *Clinica Chimica Acta*, **363**(1-2), 206–220.

Myers, E. W. (1995). Toward Simplifying and Accurately Formulating Fragment Assembly. *Journal of Computational Biology*, **2**(2), 275–290.

Myers, E. W. (2005). The fragment assembly string graph. *Bioinformatics*, **21**(SUPPL. 2), 79–85.

Nakamura, K., Oshima, T., Morimoto, T., Ikeda, S., Yoshikawa, H., Shiwa, Y., Ishikawa, S., Linak, M. C., *et al.* (2011). Sequence-specific error profile of Illumina sequencers. *Nucleic Acids Research*, **39**(13).

NCBI (2016). ML1750 hypothetical protein. *https://www.ncbi.nlm.nih.gov/gene/?term=ML1750*, (accessed: May 29, 2019).

NCBI (2018). NCBI Bacterial Genomes. *ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/ bacteria/assembly_summary.txt*, (accessed: May 29, 2019, downloaded: May 2018).

Ng, P. C. and Henikoff, S. (2003). SIFT: Predicting amino acid changes that affect protein function. *Nucleic Acids Research*, **31**(13), 3812–3814.

Nič, M., Jirát, J., Košata, B., Jenkins, A., and McNaught, A., editors (2009). *IUPAC Compendium of Chemical Terminology*. IUPAC, Research Triagle Park, NC.

Niedringhaus, T. P., Milanova, D., Kerby, M. B., Snyder, M. P., and Barron, A. E. (2011). Landscape of next-generation sequencing technologies. *Analytical Chemistry*, **83**(12), 4327–4341.

Ning, Z., Ning, Z., Cox, A. J., Cox, A. J., Mullikin, J. C., and Mullikin, J. C. (2001). SSAHA: A Fast Search Method for Large DNA Databases. *Genome Research*, **2**, 1725–1729.

Noordeen, S. (1988). Epidemiology of leprosy. In *Mycobacteria*, pages 379–397. Springer.

Novák, P., Neumann, P., Pech, J., Steinhaisl, J., and MacAs, J. (2013). RepeatExplorer: A Galaxy-based web server for genome-wide characterization of eukaryotic repetitive elements from next-generation sequence reads. *Bioinformatics*, **29**(6), 792–793.

Okonechnikov, K., Conesa, A., and García-Alcalde, F. (2015). Qualimap 2: Advanced multi-sample quality control for high-throughput sequencing data. *Bioinformatics*, **32**(2), 292–294.

Olsen, G. (1990). "Newick's 8:45" Tree Format Standard. *http://evolution.genetics.washington.edu/phylip/newick_doc.html*, (accessed: May 29, 2019).

Olson, N. D., Treangen, T. J., Hill, C. M., Cepeda-Espinoza, V., Ghurye, J., Koren, S., and Pop, M. (2017). Metagenomic assembly through the lens of validation: recent advances in assessing and improving the quality of genomes assembled from metagenomes. *Briefings in Bioinformatics*.

Pääbo, S. (1985). Preservation of DNA in ancient Egyptian mummies. *Journal of Archaeological Science*, **12**(6), 411–417.

Papadopoulos, D., Schneider, D., Meier-Eiss, J., Arber, W., Lenski, R. E., and Blot, M. (1999). Genomic evolution during a 10,000-generation experiment with bacteria. *Proceedings of the National Academy of Sciences*, **96**(7), 3807–3812.

Pareek, C. S., Smoczynski, R., and Tretyn, A. (2011). Sequencing technologies and genome sequencing. *Journal of Applied Genetics*, **52**(4), 413–435.

Park, P. J. (2009). ChIP-seq: Advantages and challenges of a maturing technology. *Nature Reviews Genetics*, **10**(10), 669–680.

Parkash, O., Kumar, A., Nigam, A., Franken, K. L., and Ottenhoff, T. H. (2006). Evaluation of recombinant serine-rich 45-kDa antigen (ML0411) for detection of antibodies in leprosy patients. *Scandinavian Journal of Immunology*, **64**(4), 450–455.

Peltzer, A., Jäger, G., Herbig, A., Seitz, A., Kniep, C., Krause, J., and Nieselt, K. (2016). EAGER: Efficient Ancient Genome Reconstruction. *Genome Biology*, **17**(1), 60.

Pevzner, P. A., Tang, H., and Waterman, M. S. (2001). An Eulerian path approach to DNA fragment assembly. *Proceedings of the National Academy of Sciences*, **98**(17), 9748–9753.

Phillippy, A. M. (2017). New advances in sequence assembly. *Genome Research*, **27**(5), xi–xiii.

Pinto, M., Borges, V., Antelo, M., Pinheiro, M., Nunes, A., Azevedo, J., Borrego, M. J., Mendonça, J., *et al.* (2016). Genome-scale analysis of the non-cultivable Treponema pallidum reveals extensive within-patient genetic variation. *Nature microbiology*, **2**(1), 16190.

Pop, M. (2009). Genome assembly reborn: Recent computational challenges. *Briefings in Bioinformatics*, **10**(4), 354–366.

Prüfer, K., Racimo, F., Patterson, N., Jay, F., Sankararaman, S., Sawyer, S., Heinze, A., Renaud, G., *et al.* (2014). The complete genome sequence of a Neanderthal from the Altai Mountains. *Nature*, **505**(7481), 43–49.

Quick, J., Quinlan, A. R., and Loman, N. J. (2014). A reference bacterial genome dataset generated on the MinION™ portable single-molecule nanopore sequencer. *GigaScience*, **3**(1), 1–6.

Rajwani, R., Yam, W. C., Zhang, Y., Kang, Y., Wong, B. K. C., Leung, K. S. S., Tam, K. K. G., Tulu, K. T., Zhu, L., and Siu, G. K. H. (2018). Comparative Whole-Genomic Analysis of an Ancient L2 Lineage Mycobacterium tuberculosis Reveals a Novel Phylogenetic Clade and Common Genetic Determinants of Hypervirulent Strains. *Frontiers in Cellular and Infection Microbiology*, **7**(January), 1–11.

Rasmussen, M., Li, Y., Lindgreen, S., Pedersen, J. S., Albrechtsen, A., and Others (2010). Ancient human genome sequence of an extinct Palaeo-Eskimo. *Nature*, **463**(7282), 757–762.

Rees, D. C., Williams, T. N., and Gladwin, M. T. (2010). Sickle-cell disease. *The Lancet*, **376**(9757), 2018–2031.

Robinson, J. T., Thorvaldsdóttir, H., Winckler, W., Guttman, M., Lander, E. S., Getz, G., and Mesirov, J. P. (2011). Integrative Genomics Viewer. *Nature biotechnology*, **29**(1), 24–26.

Robinson, O., Dylus, D., and Dessimoz, C. (2016). Phylo.io: Interactive Viewing and Comparison of Large Phylogenetic Trees on the Web. *Molecular Biology and Evolution*, **33**(8), 2163–2166.

Ronaghi, M., Uhlén, M., and Nyrén, P. (1998). A sequencing method based on real-time pyrophosphate. *Science*, **281**(5375), 363–365.

Rose, A. S., Bradley, A. R., Valasatava, Y., Duarte, J. M., Prlić, A., and Rose, P. W. (2018). NGL Viewer: Web-based molecular graphics for large complexes. *Bioinformatics*, **1**, 4.

Saiki, R. K., Chang, C.-A., Levenson, C. H., Warren, T. C., Boehm, C. D., Kazazian Jr, H. H., and Erlich, H. A. (1988a). Diagnosis of sickle cell anemia and β-thalassemia with enzymatically amplified DNA and nonradioactive allele-specific oligonucleotide probes. *New England Journal of Medicine*, **319**(9), 537–541.

Saiki, R. K., Gelfand, D. H., Stoffel, S., Scharf, S. J., Higuchi, R., Horn, G. T., Mullis, K. B., and Erlich, H. A. (1988b). Primer-directed enzymatic amplification of DNA with a thermostable DNA polymerase. *Science*, **239**(4839), 487–491.

Sand, A., Holt, M. K., Johansen, J., Brodal, G. S., Mailund, T., and Pedersen, C. N. (2014). TqDist: A library for computing the quartet and triplet distances between binary or general trees. *Bioinformatics*, **30**(14), 2079–2080.

Sanger, F., Nicklen, S., and Coulson, A. R. (1977). DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences*, **74**(12), 5463–5467.

Sawyer, S., Krause, J., Guschanski, K., Savolainen, V., and Pääbo, S. (2012). Temporal patterns of nucleotide misincorporations and DNA fragmentation in ancient DNA. *PLoS ONE*, **7**(3), e34131.

Sawyer, S., Renaud, G., Viola, B., Hublin, J.-J., Gansauge, M.-T., Shunkov, M. V., Derevianko, A. P., Prüfer, K., Kelso, J., and Pääbo, S. (2015). Nuclear and mitochondrial DNA sequences from two Denisovan individuals. *Proceedings of the National Academy of Sciences*, **112**(51), 201519905.

Schadt, E. E., Turner, S., and Kasarskis, A. (2010). A window into third-generation sequencing. *Human Molecular Genetics*, **19**(R2), 227–240.

Schatz, M. C., Delcher, A. L., and Salzberg, S. L. (2010). Assembly of Large Genomes using Cloud Computing. *Genome research*, **20**(9), 1165–1173.

Schmid, K. J., Schmid, K. J., Sorensen, T. R., Sorensen, T. R., Stracke, R., Stracke, R., Torjek, O., Torjek, O., *et al.* (2003). Large-scale identification and analysis of genome-wide single- nucleotide polymorphisms for mapping in Arabidopsis thaliana. *Genome Research*, **13**(6), 1250–1257.

Schubert, M., Lindgreen, S., and Orlando, L. (2016). AdapterRemoval v2: Rapid adapter trimming, identification, and read merging. *BMC Research Notes*, **9**(1), 1–7.

Schuenemann, V. J., Singh, P., Mendum, T. A., Krause-Kyora, B., Jäger, G., Bos, K. I., Herbig, A., Economou, C., *et al.* (2013). Genome-wide comparison of medieval and modern Mycobacterium leprae. *Science*, **341**(6142), 179–83.

Schuenemann, V. J., Avanzi, C., Krause-Kyora, B., Seitz, A., Herbig, A., Inskip, S., Bonazzi, M., Reiter, E., *et al.* (2018a). Ancient genomes reveal a high diversity of Mycobacterium leprae in medieval Europe. *PLOS Pathogens*, **14**(5), e1006997.

Schuenemann, V. J., Kumar Lankapalli, A., Barquera, R., Nelson, E. A., Iraíz Hernández, D., Acuña Alonzo, V., Bos, K. I., Márquez Morfín, L., Herbig, A., and Krause, J. (2018b). Historic Treponema pallidum genomes from Colonial Mexico retrieved from archaeological remains. *PLoS Neglected Tropical Diseases*, **12**(6), 1–20.

Seitz, A. and Nieselt, K. (2017). Improving ancient DNA genome assembly. *PeerJ*, **5**, e3126.

Seitz, A., Hanssen, F., and Nieselt, K. (2018). DACCOR–Detection, characterization, and reconstruction of repetitive regions in bacterial genomes. *PeerJ*, **6**, e4742.

Semple, C. and Steel, M. (2003). *Phylogenetics*. Oxford Univ Press, 24 edition.

Shapiro, B. and Hofreiter, M. (2014). A paleogenomic perspective on evolution and gene function: new insights from ancient DNA. *Science*, **343**, 1236573.

Shapiro, J. A. and von Sternberg, R. (2005). Why repetitive DNA is essential to genome function. *Biological reviews of the Cambridge Philosophical Society*, **80**(2), 227–50.

Simpson, J. T. and Durbin, R. (2010). Efficient construction of an assembly string graph using the FM-index. *Bioinformatics*, **26**(12), 367–373.

Simpson, J. T. and Durbin, R. (2012). Efficient de novo assembly of large genomes using compressed data structures. *Genome Research*, **22**(3), 549–556.

Šmajs, D., Zobaníková, M., Strouhal, M., Čejková, D., Dugan-Rocha, S., Pospíšilová, P., Norris, S. J., Albert, T., *et al.* (2011). Complete genome sequence of Treponema paraluiscuniculi, strain Cuniculi A: The loss of infectivity to humans is associated with genome decay. *PLoS ONE*, **6**(5).

Smith, A. D., Xuan, Z., and Zhang, M. Q. (2008). Using quality scores and longer reads improves accuracy of Solexa read mapping. *BMC Bioinformatics*, **9**(1), 128.

Smitt, A., Hubley, R., and Green, P. (1996). RepeatMasker Open-3.0. *http://www.repeatmasker.org*, (accessed: May 29, 2019).

Stamatakis, A. (2014). RAxML version 8: A tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, **30**(9), 1312–1313.

Sturm, M. (2018). NGS-BITS. *https://github.com/imgag/ngs-bits*, (accessed: May 29, 2019).

Sun, J., Meng, Z., Wu, K., Liu, B., Zhang, S., Liu, Y., Wang, Y., Zheng, H., Huang, J., and Zhou, P. (2016). Tracing the origin of Treponema pallidum in China using next-generation sequencing. *Oncotarget*, **7**(28), 42904.

Swofford, D. L., Olsen, G. J., Waddell, P. J., and Hillis, D. M. (1996). Phylogenetic Inference. *Molecular systematics*, **2**.

Tarailo-Graovac, M. and Chen, N. (2009). Using RepeatMasker to identify repetitive elements in genomic sequences. *Current Protocols in Bioinformatics*, **5**(1), 4–10.

Tettelin, H., Riley, D., Cattuto, C., and Medini, D. (2008). Comparative genomics: the bacterial pan-genome. *Current Opinion in Microbiology*, **11**(5), 472–477.

Treangen, T. J., Abraham, A. L., Touchon, M., and Rocha, E. P. (2009). Genesis, effects and fates of repeats in prokaryotic genomes. *FEMS Microbiology Reviews*, **33**(3), 539–571.

Tringe, S. G. and Rubin, E. M. (2005). Metagenomics: DNA sequencing of environmental samples. *Nature Reviews Genetics*, **6**(11), 805–814.

UCSC (2014). HGDownload. *http://hgdownload.cse.ucsc.edu/goldenPath/hg38/bigZips/*, (accessed: May 29, 2019).

Uniprot (2018a). UniProtKB - Q57532. *https://www.uniprot.org/uniprot/Q57532*, (accessed: May 29, 2019).

Uniprot (2018b). UniProtKB - Q9CBP6. *https://www.uniprot.org/uniprot/Q9CBP6*, (accessed: May 29, 2019).

Utturkar, S. M., Klingeman, D. M., Land, M. L., Schadt, C. W., Doktycz, M. J., Pelletier, D. A., and Brown, S. D. (2014). Evaluation and validation of de novo and hybrid assembly techniques to derive high-quality genome sequences. *Bioinformatics*, **30**(19), 2709–2716.

Van der Auwera, G. A., Carneiro, M. O., Hartl, C., Poplin, R., del Angel, G., Levy-Moonshine, A., Jordan, T., Shakir, K., *et al.* (2013). From fastQ data to high-confidence variant calls: The genome analysis toolkit best practices pipeline. *Current Protocols in Bioinformatics*, **43**(1), 1–33.

Van Dijk, E. L., Auger, H., Jaszczyszyn, Y., and Thermes, C. (2014). Ten years of next-generation sequencing technology. *Trends in genetics*, **30**(9), 418–426.

Voelkerding, K. V., Dames, S. A., and Durtschi, J. D. (2009). Next-generation sequencing:from basic research to diagnostics. *Clinical Chemistry*, **55**(4), 641–658.

Vonholdt, B. M., Pollinger, J. P., Lohmueller, K. E., Han, E., Parker, H. G., Quignon, P., Degenhardt, J. D., Boyko, A. R., *et al.* (2010). Genome-wide SNP and haplotype analyses reveal a rich history underlying dog domestication. *Nature*, **464**(7290), 898–902.

Watson, J. D. and Crick, F. C. H. (1953). Molecular structure of nucleic acid. A structure for deoxyribose nucleic acid. *Nature*, **171**(8), 737–738.

Weiner, P. (1973). Linear pattern matching algorithms. *Switching and Automata Theory, 1973. SWAT '08. IEEE Conference Record of 14th Annual Symposium on*, pages 1–11.

Wheeler, W. C. (1990). Nucleic acid sequence phylogeny and random outgroups. *Cladistics*, **6**(4), 363–367.

Williams, T. A., Heaps, S. E., Cherlin, S., Nye, T. M., Boys, R. J., and Embley, T. M. (2015). New substitution models for rooting phylogenetic trees. *Philosophical Transactions of the Royal Society B: Biological Sciences*, **370**.

Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., *et al.* (2016). Apache Spark: A Unified Engine for Big Data Processing. *Commun. ACM*, **59**(11), 56–65.

Zerbino, D. R. and Birney, E. (2008). Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Research*, **18**(5), 821–829.

Zerbino, D. R., Achuthan, P., Akanni, W., Amode, M. R., Barrell, D., Bhai, J., Billis, K., Cummins, C., *et al.* (2018). Ensembl 2018. *Nucleic Acids Research*, **46**(D1), D754–D761.

Zhao, N., Han, J. G., Shyu, C.-R., and Korkin, D. (2014). Determining Effects of Non-synonymous SNPs on Protein-Protein Interactions using Supervised and Semi-supervised Learning. *PLoS Computational Biology*, **10**(5), e1003592.