New Approaches to Long-Read Assembly under High Error Rates

zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Philipp Bongartz

aus Bonn

Tag der mündlichen Prüfung:	15.01.2020
Erster Gutachter:	Prof. Dr. Alexandros Stamatakis, HITS Heidelberg, KIT Karlsruhe
Zweiter Gutachter:	Prof. Dr. Dirk Metzler, LMU München

Zusammenfassung

Das Gebiet der Genomassemblierung beschäftigt sich mit der Entwicklung von Algorithmen, die Genome am Computer anhand von Sequenzierungsdaten rekonstruieren. Es geriet erstmals in den Neunzigern mit dem Human Genome Project in den Fokus der Öffentlichkeit. Da nur kurze Abschnitte des menschlichen Genoms ausgelesen werden konnten, musste die Rekonstruktion längerer Genomsequenzen aus den ausgelesenen Abschnitten im Nachhinein am Computer erfolgen. Auch fast 20 Jahre nach der Veröffentlichung der menschlichen Genomsequenzen stellt die Genomeassemblierung nach wie vor noch einen essentiellen Verarbeitungsschritt für Sequenzierungsdaten dar. Nur Datendurchsatz, Länge und Fehlerprofil der ausgelesenen Genomabschnitte haben sich verändert und damit einhergehend auch die algorithmischen Anforderungen.

Damit komplementiert das Forschungsgebiet der Genomeassemblierung die Sequenzierungstechnologien, die sich mit enormer Geschwindigkeit weiter entwickelt haben. Zusammen erlauben sie die Entschlüsselung der Genome einer stark zunehmenden Anzahl von Lebewesen und bilden damit die Grundlage für einen Großteil der Forschung in verschiedensten Bereichen der Biologie und Medizin.

Trotz der beeindruckenden technologischen und algorithmischen Entwicklungen der vergangenen Jahrzehnte ist es bisher nur für bakterielle Genome gelungen, die komplette Genomsequenz zu rekontruieren. Bei der Assemblierung der wesentlich größeren eukaryotischen Genome bestehen mehrere ungelöste algorithmische Probleme. Diese Probleme hängen mit verschiedenen repetitiven Strukturen zusammen, die in fast allen Genomen höherer Lebewesen vorkommen. Deshalb werden eukaryotische Genome immer in wesentlich mehr unzusammenhängenden Sequenzen veröffentlicht als die jeweiligen Lebewesen Chromosomen haben.

Die repetitiven Strukturen, die für die Lücken in den Genomsequenzen verantwortlich sind, lassen sich grob in drei Klassen unterteilen. Mikrosatelliten und Minisatelliten sind sehr kurze Sequenzen, die sich tausende oder zehntausende Male direkt aufeinander folgend wiederholen können. Dieses Muster ist typisch für sogenannte Centromere und Telomere, die sich in der Mitte und an den Enden vieler Chromosome befinden. Sogenannte Interspersed Repeats, oft auch als Transposons bezeichnet, sind längere Sequenzen, die häufig in fast identischer Form an unterschiedlichen Stellen im Genome vorkommen. Sogenannte Tandem Repeats dagegen sind längere Sequenzen, die direkt aufeinanderfolgend mehrere Male in einem Genom auftreten können. Oft sind Tandem Repeats Genkomplexe, das heißt Ansammlungen fast iden-

tischer proteinkodierender Abschnitte, die es der Zelle erlauben, die kodierten Proteine besonders schnell zu produzieren.

Jede dieser repetitive Strukturen stellt spezifische Anforderung an Assemblierungsalgorithmen. In dieser Doktorarbeit leisten wir mehrere Beiträge zur Lösung der letzteren zwei vorgestellten Probleme, der Assemblierung von Interspersed Repeats und Tandem Repeats.

In Teil 1 der Arbeit stellen wir mehrere Datenverarbeitungsprozeduren vor, die Sequenzierungsdaten aufbereiten, um die seltenen Unterschiede zwischen mehrfach auftretenden Genomsequenzen zu identifizieren. Diese beinhalten Softwareprogramme zur Berechnung und Optimierung von Multiplen Sequenz Alignments (MSA) anhand dynamischer Programmierung und zur statistischen Modellierung und Analyse der Unterschiede, wie das MSA sie präsentiert.

In Teil 2 bauen wir auf dieser Analyse auf und präsentieren ein Softwareprogramm zur Assemblierung von Interspersed Repeats. Dieses Programm baut auf mehreren algorithmischen Neuerungen auf und ist in der Lage, Transposonfamilien mit sehr langen Sequenzen und sehr vielen verschiedenen Kopien effektiv zu assemblieren. Es ist das erste Programm dieser Art, welches in der Lage ist, Transposonfamilien mit dutzenden von Kopien zu assemblieren. Es gelingt uns zu zeigen, dass es auch für kleinere Transposonfamilien akkurater und schneller ist als das bisher einzige Konkurrenzprogramm, welches auf dieses Assemblierungsproblem spezialisiert ist.

In Teil 3 beschreiben wir eine Analysepipeline, die es uns ermöglicht, Genkomplexe aus dutzenden von Tandem Repeats zu assemblieren. Diese Pipeline enthält Clustering und Graph Drawing Algorithmen. Ihr Herzstück ist ein Fehlerkorrekturalgorithmus, der auf Neuronalen Netzwerken basiert. Wir demonstrieren den praktischen Nutzen dieser Pipeline durch die Assemblierung des Drosophila Histone Komplexes.

Im Abschluss diskutieren wir die Möglichkeit, Mikro- und Minisatelliten zu assemblieren und schlagen Forschungsansätze für weitere Verbesserungen im Bereich der Interspersed Repeat- und Genkomplexassemblierung vor.

Abstract

The research area of Genome Assembly engages in the development of algorithms for the computational reconstruction of genomes. It was subject of considerable media attention during the Human Genome Project in the 1990s and early 2000. Because only short sections of the genome could be deciphered in one go, longer sections of the genome had to be reconstructed on the computer in a post-processing step. Although it has almost been 20 years since the Human Genome Project published the sequence of the human genome, Genome Assembly still constitutes an essential step for analysing sequencing data. Only throughput, length, and error profile of the reads provided by sequencing machines have changed considerably, and with them the algorithmic requirements.

Genome Assembly is complementary to the rapidly developing sequencing technologies. Together, they allow for deciphering a steadily increasing number of genomes and provide the foundation for a substantial part of the research in different areas of biology and medicine.

Despite the extraordinary technological and algorithmic development over recent decades, so far only bacterial genomes have been fully assembled. There are several unresolved problems in the assembly of the substantially larger eukaryotic genomes. These problems are due to different repeat structures, which occur in almost all genomes of higher organisms. The draft genomes of eukaryotic genomes are currently published in many more disparate parts than the respective animal or plant has chromosomes.

The repeat structures that are responsible for the preponderance of these gaps in chromosomes can be roughly divided into three classes.

Micro- and *Minisatellites* are very short sequences that repeat directly adjacent to each other for thousands or tens of thousands of times. This pattern is typical for centromeric or telomeric regions in the middle and at the end of many chromosomes. *Interspersed repeats*, also denoted as *transposable elements*, are longer sequences that occur in almost identical form many times in different regions of the genome. *Tandem repeats* are longer sequences that occur numerous times directly adjacent to each other. Often, tandem repeats form part of so called gene complexes, that means, collections of repeated protein coding sequences that allow for the rapid production of the encoded protein in the cell.

Each of these repetitive structures poses unique assembly challenges. In this thesis, we contribute several ideas for improving the assembly of interspersed repeats as well as gene complexes.

In part 1 of this thesis, we present several data processing procedures that

allow for the detection and extraction of rare differences between different copies of a repeat sequence. These are implemented in tools for computing and optimising multiple sequence alignments (MSA) utilising dynamic programming, as well as a tool for the statistical modelling and analysis of repeat differences as they can be extracted from the MSA.

In part 2, we build on these pre-processing procedures and present a software program for the assembly of interspersed repeats. This program is based on several novel algorithmic ideas and is capable of assembling transposon families with long sequences and a high number of copies. It is the first such program that can handle transposon families with dozens of copies. We show that our method is superior in speed and accuracy to the only existing competing repeat resolution tool.

In part 3, we describe an analysis pipeline for the assembly of gene complexes, consisting of dozens of tandem repeats. This pipeline contains clustering and graph drawing algorithms. Its core is a de-noising algorithm that is based on neural networks. We demonstrate the practical utility of this pipeline by assembling the Drosophila Histone Complex, a long standing gap in the genome of this important model organism.

Acknowledgements

First and foremost, I want to thank my supervisor Prof. Dr. Alexandros Stamatakis, who took me on as a student halfway through my PhD, when neither official nor practical supervision had materialised as planned. This was connected with considerable additional work and no benefit for him. His competent and structured supervision in the last year easily doubled my productivity and is the main reason that this thesis is now complete.

I am grateful to my co-advisor Prof. Dr. Dirk Metzler, who gracefully accepted the hassle of jumping through the bureaucratic hoops and expending the necessary time to review my thesis.

The Heidelberg Institute for Theoretical Studies provided a near ideal study environment. Even though I would have to admit that I did not make the most of it, these five years marked profound changes in my knowledge, my abilities, my interests, and my identity, and I wouldn't have wanted to miss them.

I am grateful to Philipp, Martin, Siegfried, Sean, Beifei, Kashif, Michael, and many others for interesting discussions, hard fought table tennis matches, and simply companionship. Finally, I also want to thank the HITS management, who did not drop me when my research group was officially disbanded, and the Klaus Tschira Foundation for funding my position.

Contents

Part I

Preliminaries

Chapter 1

Introduction

1.1 Motivation

Knowing the precise sequence of bases that constitute a genome lies at the heart of a wide range of scientific inquiries.

The genome is the vehicle of information on which evolution acts, and deciphering it allows a uniquely accurate view into evolution's unfolding. Evolutionary processes can be reconstructed by the inferring of phylogenetic trees, thereby determining the relatedness of existing species. They can also be directly observed either in real time (for bacteria, viruses, and cancer cells) or in retrospect, via paleo genetics, that is, the sequencing and assembly of ancient (usually) human DNA. These scientific fields do not only shed light on the origin of life and human development, but can also, for instance, predict the spread of pathogens and the development of drug resistances.

In the last sixty years, we have gone from understanding the structure and purpose of DNA to actively changing it in a variety of ways. Knockout mice have been at the forefront of gene function investigation and gene interaction research for almost thirty years. In recent years, Crispr/Cas9 has made the manipulation of genes substantially more versatile and precise. These and numerous other techniques rely on both the knowledge of what is available for manipulation and the ability to check whether an intervention was successful.

A large number of traits and diseases, however, do not depend on a single gene or a small number of genes. Instead, they are polygenic, that is, influenced by numerous genes. Some traits even depend on single nucleotide polymorphisms of small effect at thousands of positions in the human genome, which makes it challenging to detect the variants that contribute causally to the trait or disease. The difficulty of discovering causal gene variants

for traits, that twin studies have conclusively established are significantly heritable, has been dubbed the *missing heritability problem*. Most of the scourges of modern societies like cancer, diabetes, obesity and heart disease are massively polygenic. Therefore, this missing heritability led to some disillusionment in the years after the successful completion of the human genome project when it became apparent, that there was no "gene for cancer". As almost all causal gene variants of polygenic traits have small effect, massive sample sizes are needed to detect them. We are now on the cusp of being able to achieve just that via genome wide association studies (GWAS) with hundreds of thousands of (partly) sequenced genomes as provided by comprehensive databases like the UK biobank.

Genome assembly also facilitates the classification of cancers into numerous different diseases, as a one-size-fits-all approach to drugs is known to be particularly ineffective in chemotherapy. Apart from adapting existing chemotherapeutic drugs accurately to the tumour and patient, genome assembly plays a major role in the development of new cancer therapies. The approach to sequence cancer genomes and tailor immune cells to attack surface proteins detected in the cancer genome, for example, is on the leading edge of cancer research.

Thus, further improving genome assembly methods will yield advances in almost all biological and medical research fields. In this thesis, the focus is on improving the sequence resolution, that means, getting closer to recreating fully contiguous sequences for each chromosome, instead of numerous disparate parts. Only highly resolved genome assemblies allow the study of structural variation. The term *structural variation* denotes large insertions, deletions, duplications and inversions, which are a key mutational process in cancer. Furthermore, highly resolved genome assemblies are a prerequisite for investigations into repetitive regions of the genome and into the spatial organisation of the chromosome molecule in the cell. These are essential for developmental genetics and research into regulatory functions.

1.2 Scientific contribution and overview

In this thesis, we present several novel contributions to the field of genome assembly. While these contributions primarily address the challenge of improving the sequence resolution of de novo assemblies, they are also potentially relevant to the problems of haplotype disambiguation and metagenomics.

In Chapter 4, we present a tool for the refinement of multiple sequence alignments. The typical multiple sequence alignment tool is specialised in the alignment of proteins and limited in the number and length of sequences

it can process. Our tool is specifically optimised for the alignments of long read repeat sequences and can accommodate ten thousands of sequences, each of them ten thousands of bases long. We show that our optimisation objective translates well to the task of determining copy differences for repeat resolution.

In Chapter 5, we introduce a highly parallelisable method to statistically model and identify repeat copy differences in noisy long reads. This method is implemented as a highly efficient bitvector algorithm.

In Chapter 6 and 7, several novel algorithms are introduced, that serve to hierarchically cluster repeat sequences by the identified copy differences and to use the resulting clusters to resolve repeat regions. We show that our clustering tool significantly outperforms state-of-the-art methods in repeat resolution.

In Chapter 9, we describe a de-noising algorithm utilising neural networks to reduce the error rate in extracted copy differences to such a degree, that automatic assembly of the highly conserved Drosophila Histone Complex becomes possible. We believe that this novel algorithm has wide applicability beyond the specific genome assembly problem.

Several of the presented tools and algorithms are integrated into a pipeline for the automatic assembly of highly repetitive gene complexes, together with further analysis steps including preprocessing, clustering and graph traversal algorithms. These additional analyses and their results are presented in Chapter 3 and 10.

This is the extent of the novel scientific contributions presented in this thesis. The rest of this thesis is structured as follows.

In Chapter 1, we detail motivation and scientific contribution. In Chapter 2 we give a historical and methodological introduction to the field of genome assembly. Similarly, we introduce the basics of neural networks in Chapter 8, while finishing the thesis with conclusion and outloook in Chapter 11.

Chapter 2

Introduction to Genome Assembly

2.1 Foundational research

The concept of a *genome* as the collection of the units of hereditary information is the culmination of a long history of scientific and philosophical inquiry into the nature of heredity [1].

While some of the cornerstones of modern genetics were already anticipated by de Maupertuis in the 18th century [2], the idea of discrete heritable units containing hereditary information was systematically examined by Gregor Mendel in his famous pea experiments [3]. The existence of *genes* is implied by one of his three laws, the law of "independent assortment", which states that separate traits are passed on independently.

The term *gene* itself was only introduced later-on. The heritable units were first called *pangenes* (or *gemmules*, see [4]) after Darwin's 1868 work *The variation of Animals and Plants under Domestication* [5], and finally *genes*. The term *genome* was coined by Hans Winkler in 1920 to describe the collection of all genes [6].

Only two decades later, the DNA macromolecule was identified as carrier of genetic information by Avery, MacLeod, and MacCarty [7], based on earlier work by Frederick Griffith [8], via experiments with streptococcus pneumoniae strains.

Another decade later, in 1953, Francis Crick and James D. Watson published the paper *Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid* [9], in which they describe the double helix structure of DNA. This finding was based on a wealth of data collected by other researcher, most notably maybe Rosalind Franklin and M.H.F. Wilkins.

Earlier investigations into heredity were based on pedigrees and the measurement of traits which yielded vague and often seemingly contradictory evidence. The new knowledge about the chemical composition of the genome introduced the possibility of deciphering the blueprint of life itself and to base future inquiries onto firmer grounds.

Following Crick and Watson's breakthrough, rapid progress in biochemical genetics allowed the elucidation of the *genetic code* and laid the ground work for building the first DNA sequencers [10]. By then it was well known, that the genomic information is laid out in chromosomes containing sequences of four nucleobases: adenine, cytosine, guanine, and thymine, (A,C,G,T). And it had also been discovered, that sub-sequences in each chromosome code for proteins by encoding amino acids with three subsequent DNA bases.

DNA sequencing took its first small step with the full sequence of the genome of bacteriophage MS2 determined in 1972 by Walter Fiers [11]. MS2 is a RNA virus that infect the bacterium Escherichia coli and has one of the smallest known genomes with just 3.5 kilo base pairs. The major step forward happened in 1977 with the invention of chain-termination DNA sequencing by Frederick Sanger [12]. Sanger sequencing is conducted by primer extension. Primer extension means starting with a given short sequence, the primer, which is extended base by base in a fashion that facilitates the classification of the extending bases. By creating the next primer from the newly sequenced part, one slowly walks along the genome, deciphering its sequence primer by primer and base by base.

Shortly after this, an alternative to primer walking was proposed by Staden [13] and applied by Gardner *et al.* [14] for sequencing a viral genome. This strategy was based on random shearing of the genome which allows sequencing of many parts in parallel. The parallel nature of this process is alluded to in the name *shotgun sequencing*. Shotgun sequencing requires the use of computers to piece together the full sequence from many separately sequenced parts, the so-called *reads*. From that moment on, DNA sequencing and genome assembly developed into two interlocking research fields, a symbiosis that will likely last until the sequencing technology matures to the point of reading whole chromosomes.

2.2 Algorithm development

The eighties witnessed a steady stream of algorithmic innovations that led up to the early culmination of this young research area, the Human Genome Project. Central to the field of genome assembly are pairwise sequence alignment algorithms that compute the similarity between two sequences. The

```
Read1: aaagtaggatcgaccggccca
Read2: agcgatggccgggcctatagttt

Alignment:
       aaagtag-gatcgaccggccc-a
            ||*|||*|*||||*||*|
            agcgat-ggccgggcctatagttt

Edit script:
       dddddmmimmmdmsmmmmsmmimiiiiii
```

(a) Two reads are aligned according to an edit script.

		a	a	a	g	t	a	g	g	a	⋯
	0	0	0	0	0	0	0	0	0	0	⋯
a	0	0	0	0	1	1	0	1	1	0	⋯
g	0	1	1	1	0	1	1	0	1	1	⋯
c	0	1	2	2	1	1	2	1	1	2	⋯
g	0	1	2	3	2	2	2	2	1	2	⋯
a	0	0	1	2	3	3	2	3	2	1	⋯
⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯

(b) The edit script was extracted by backtracking the optimal path through the alignment matrix.

Figure 2.1: Illustration of alignment, edit script and partial alignment matrix.

Needleman-Wunsch sequence alignment algorithm [15] has become the standard alignment algorithm and was refined and adapted several times, by Smith-Waterman, Gotoh, Altschul, Erickson, and Myers [16, 17, 18, 19, 20]. The Needleman-Wunsch algorithm, see Figure 1, utilizes dynamic programming to compute an explicit sequence comparison. This sequence comparison takes the form of an edit script E that encodes the most parsimonious way to edit a sequence $a_{i<m}$ into another sequence $b_{j<n}$. The edit script E lists four edit operations, the match m, where the original base is adopted into the edited sequence, the substitution s, where it is changed into a different base, as well as the deletion d and the insertion i. The last two denote a base being skipped or an additional base being added to the edited sequence.

Which edit script is most parsimonious depends on the scoring function ω and the gap penalty g. In some versions of the algorithm, gaps are scored by a function that takes the size of the gap into account. In this thesis we only use a fixed gap penalty of $g = 1$. ω scores the substitution of bases, with $\omega(b, b) = 0$ and $\omega(b_1, b_2) = 1$ for $b_1 \neq b_2$. In many use cases ω will be more complex and take for example the likelihood of certain mutations into account. For the alignment of sequencing reads these binary values for g and f are sufficient and computationally efficient.

The number of edit operations other than m, the so-called alignment score, provides a measure of sequence similarity. If the alignment score is sufficiently explained by the expected error rate, the two reads are likely to have been sequenced from the same position in the genome.

The alignment score is computed by filling a matrix M. Each entry M_{ij} contains the score of the alignment of the prefixes of a and b that end with the i-th and j-th base respectively. Naturally, M_{00} can be initialised with 0, as changing the empty sequence into the empty sequence requires no edit

operations, see Figure 2.1.

However, comparing pairs of sequences is only at the beginning of the assembly process. It is a priori not granted that a random sampling of reads will cover all regions of a genome. Oversampling is a necessity. That means, every position in the genome will on average be sampled several times. There is a trade-off between the sampling rate and the length of regions that will have enough coverage to be faithfully reconstructed.

Statistical concepts for the assembly of full genomes of higher organisms as well as a computational framework for genome assembly were developed by Lander and Waterman [21]. Originally, this work formulated the problem of genome assembly as a shortest common superstring problem. This problem is well known to be NP-complete [22] and therefore computationally intractable. To circumvent this diffculty, efficient greedy algorithms were proposed by Ukkonen and Tarhio [23].

2.3 The Human Genome Project

The Human Genome Project represents one of the scientific highlights of the past century. Shortly after the invention of Sanger sequencing, before even a less complicated bacterial genome had been fully sequenced, leading biologists began to discuss a complete mapping of all human genes. The project was officially launched in 1990 and represents the largest collaborative biological research effort ever conducted.

The sequencing strategy was to conduct *hierarchical shotgun*, which means shotgun sequencing of large sections of the genome. When the effort of sequencing the human genome was already under way, new theoretical results made a different approach seem like a promising alternative.

In 1995, Roach *et al.* [24] showed on simulated data that paired end sequencing with inserts of variable size was a feasible strategy, even for very large genomes. Simultaneously, Kececioglu and Myers [25] introduced the overlap layout consensus concept for whole genome shotgun sequencing, for a simplified overview see Figure 2.2.

These developments motivated Craig Venter to try to overtake the public effort with his company Celera, by using paired-end whole genome sequencing on top of the data provided by the international effort.

Despite starting three years later, his plan almost succeeded [26] and legal uncertainties forced the public project to publish the first human draft genome already in 2001 [27], almost five years earlier than planned.

Algorithm 1 Needlemann-Wunsch

$a_{i<m} \leftarrow$ sequencing read
$b_{j<n} \leftarrow$ sequencing read
$\omega \leftarrow$ scoring function
$g \leftarrow$ gap penalty
procedure DYNAMIC PROGRAMMING
    $M_{0,0} \leftarrow 0$
    $M_{i,0} \leftarrow 0, 1 \leqq i \leqq m$
    $M_{0,j} \leftarrow 0, 1 \leqq j \leqq n$
    for $1 \leqq i \leqq m$ **do**
        for $1 \leqq j \leqq n$ **do**

$$M_{i,j} \leftarrow \max \begin{cases} M_{i-1,j-1} + \omega(a_i, b_j) & Match/Substitution \\ M_{i-1,j} + g & Deletion \\ M_{i,j-1} + g & Insertion \end{cases}$$

procedure BACKTRACKING
    $i \leftarrow m$
    $j \leftarrow n$
    $E \leftarrow \emptyset$
    while $i > 0, j > 0$ **do**
        if $M_{i,j} = M_{i-1,j-1} + \omega(a_i, b_j) \bigwedge a_i = b_j$ **then**
            $E \leftarrow m + E$
            $i \leftarrow i - 1$
            $j \leftarrow j - 1$
        else if $M_{i,j} = M_{i-1,j-1} + \omega(a_i, b_j) \bigwedge a_i \neq b_j$ **then**
            $E \leftarrow s + E$
        else if $M_{i-1,j} + g$ **then**
            $E \leftarrow d + E$
            $i \leftarrow i - 1$
        else if $M_{i,j-1} + g$ **then**
            $E \leftarrow i + E$
            $j \leftarrow j - 1$

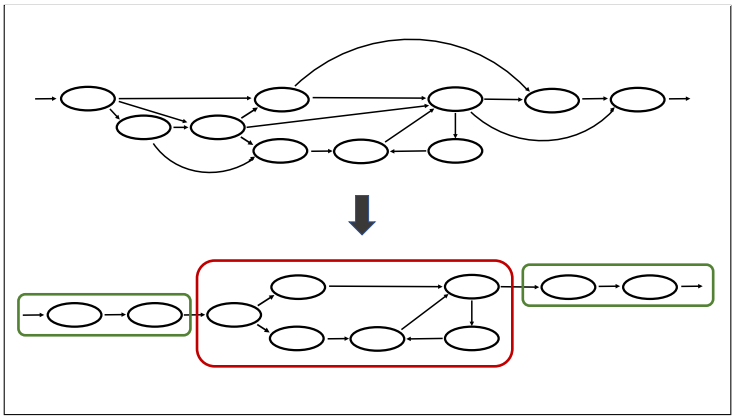

Figure 2.2: In the overlap layout consensus approach, overlaps, that means partial alignments, are calculated for all reads. The resulting graph can be quite complicated. But it can be radically simplified by eliminating all fully contained or transitively inferable overlaps. The resulting *reduced overlap graph* contains contiguous paths without junctions, so called *contigs*, indicated by the green boxes. These contigs are extracted from the graph and a consensus sequence is called on their reads.

2.4 Next generation sequencing

With the completion of the Human Genome Project, the era of *next generation sequencing* NGS started. NGS is an umbrella term for a number of sequencing technologies like 454 Life Sciences' 454 GS (Genome Sequencer), Illumina's MiSeq and HiSeq, ABI's SOLiD (Sequencing by Oligonucleotide Ligation and Detection), and Life Technologies' Ion Torrent and Proton Torrent platforms. The NGS platforms all share certain defining features: The reads are short, from dozens to up to approximately 400 bases, the error rate is low (for the most recent machines even below 1%), and the price of sequencing runs is low and continues dropping.

With Illumina being the dominant sequencing technology, the era of high throughput sequencing had started and the algorithmic requirements changed dramatically. To find candidates for overlapping reads, k-mer seeding was necessary for the relatively long Sanger reads. K-mer seeding is the practice of determining substrings of k bases, so-called k-mers, that are shared between two reads. K-mer seeding is done to assess the probability that reads overlap before conducting a costly sequence alignment step.

With reads only being dozens or a few hundred bases long, the number of reads increased. This constituted a problem, as the number of computationally expensive read comparisons increased quadratically in the number of

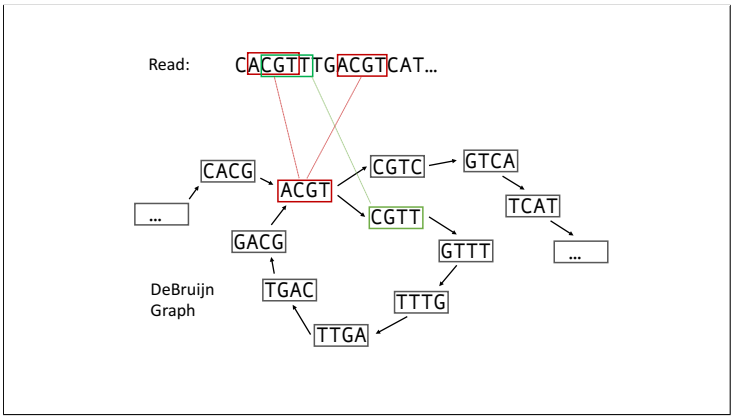

Figure 2.3: To build a DeBruijn graph substrings of fixed length (kmers) are extracted from each read. These constitute the nodes of the graph. Kmers that can be changed into each other by deleting a base on one end and appending a base on the other end, are connected by edges. Touring the graph allows to recover the genome sequence. But kmers that occur several times lead to cycles that are not always resolvable.

reads. However, with k-mers spanning a substantial part of the short reads, calculating actual pairwise alignments became less necessary. This led to the introduction of the DeBruijn graph approach [28]: To build such a graph only the k-mers were extracted from the reads and the genome was recreated by touring the graph implicit in single base extensions of the counted k-mers, see Figure 2.3.

This approach together with the low and decreasing cost, led to a deluge of new genome assemblies. But the inexpensive data came at a cost. With the short reads, the resolution of structural variations and repetitive sequence was mostly impossible, see Figure 2.4. Consequently, the new genome assemblies generally could not reach the quality of the earlier Sanger assemblies. The assembly resolution is quantified by statistics on the length of contiguous sequences, so-called *contigs*. Most commonly for example in the N50 measure that is defined as the length of the longest contig, such that it and all longer contigs constitute more than 50% of the assembly sequence.

Of course, maximizing the N50 or similar measures does not necessarily lead to an assembly of higher quality. However, if the probability of misassembly has been kept to a minimum and the accuracy of the final genome sequence is sufficiently high to support downstream analysis, a more contiguous assembly is a better assembly.

For instance, only with the currently feasible highly contiguous assemblies can we begin to routinely investigate the full three dimensional structure

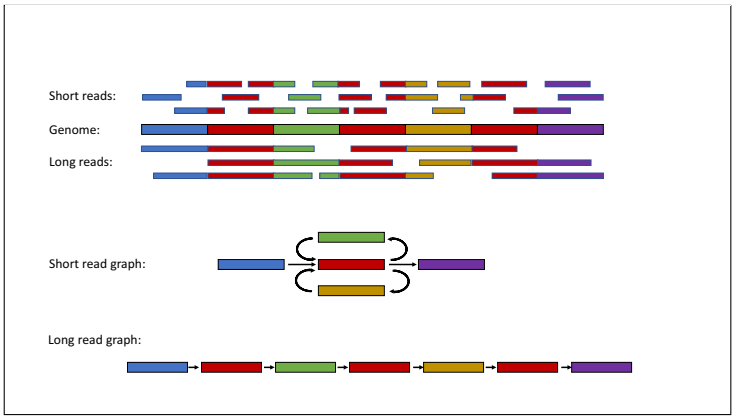

Figure 2.4: This figure illustrates the problem of interspersed repeats. Short reads that do not fully span interspersed repeats will lead to cycles in the assembly graph that are not resolvable. Long reads allow full resolution of interspersed repeats that can be fully spanned.

of the folded DNA sequence, using techniques like chromatin conformation capture [29]. This is a key requisite to create comprehensive genome-wide interaction maps that are necessary to study the interaction of genes from very different regions of a chromosome.

2.5 Single molecule real-time sequencing

After a decade of NGS, the field of genome sequencing underwent another revolution. While Sanger sequencing and all NGS technologies were based on the polymerase chain reaction (PCR) to amplify the genetic material and create a strong signal, the new technologies by Pacific Biosciences [30] and Oxford Nanopore [31, 32] were single molecule real time sequencing (SMRT). This means that a single strand of DNA is sequenced at a time.

On the one hand, this entails that the error rate increased substantially, as the signal was no longer amplified. On the other hand, these new reads are substantially longer, with several thousand base pairs average read length, and progressively becoming longer. Additionally, SMRT reads alleviated the systematic errors and coverage gaps that are a result of amplification biases typical for PCR. With a truly random error, even relatively high error rates can be averaged out.

The high error rate of long read technologies made the prevailing De-Bruijn graph paradigm infeasible. A return to the overlap-layout-consensus approach was mandated and realized for instance by Koren *et al.* with the

Canu assembler [33], Chin *et al.* with the Falcon assembler [34], and with MARVEL [35] developed at HITS.

These assemblers made de novo assemblies of unprecedented quality possible [36]. MARVEL specializes on the assembly of large and repetitive genomes. Using this assembler, it was possible to assemble the genome of the axolotl (Ambystoma mexicanum) [35], which, with a size of 30 gbp (i.e. 30 billion bases), is roughly ten times larger than the human genome. The improvement over earlier sequencing technologies is striking in the assembly of the genome of the flatworm schmidtea mediterranea [37]. This genome contains 61.7% repetitive sequence as well as a high AT-bias (i.e. a substantially higher percentage of the bases A and T as compared to the bases C and G), and could not be assembled with NGS short read data and respective assemblers. Its existing Sanger sequencing assembly [38] had a N50 of only 19 kbp, which the MARVEL assembly improves upon by almost two orders of magnitude.

Both, MARVEL and Canu, routinely assemble bacterial genomes perfectly. In less complicated eukaryotic genomes they occasionally manage to assemble a full chromosome arm from the telomeric region at one end to the centromeric region in the middle of the chromosome. In one case, MARVEL even assembled a full chromosome of the yeast cyberlindnera.

Despite these advances, there still remain some repetitive structures that can not be resolved by any existing assembler. Centromeric regions and the telomeres are composed of micro-satellites, sequences of just a handful of bases repeated hundreds of thousand times. Reads sampled from micro-satellite sequence cannot be reasonably overlapped, see Figure 2.5. It is likely that both, longer reads and lower error rates, are necessary to resolve centromeres and telomeres.

More progress has been made in the resolution of interspersed repeats. These repetitive sequences are typically transposable elements. Transposable elements are subsequences within a genome that have the ability to insert a copy of themselves into the genome at a different position [39]. They therefore appear to be randomly strewn into the genome. Interspersed repeats can be resolved by reads that span the entire repeat and can be anchored in unique flanking sequence on both sides. The improved resolution of genomes under the new long read paradigm is mainly due to the higher number of spanning reads. However, numerous transposon families are longer than even the longest current reads. This fact accounts for the number of contigs of very complicated genomes that is still high.

While interspersed repeats may eventually all be spanned by reads of continuously increasing length, tandem repeat clusters pose an additional difficulty. These repeats are placed side by side, sometimes comprising dozens

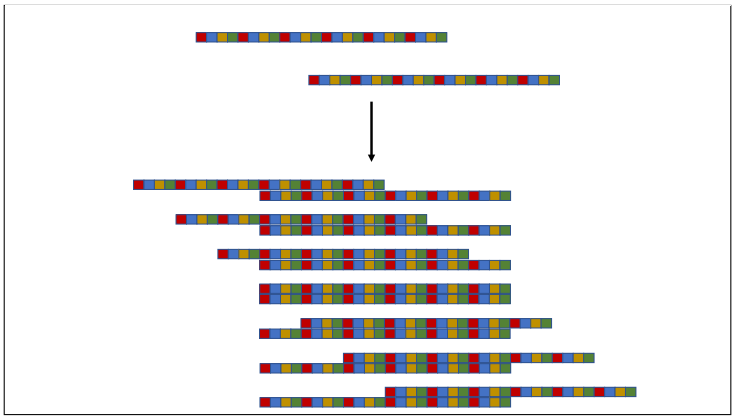

Figure 2.5: In micro-satellite regions, the same combination of bases is repeated thousands of times. Reads sequenced from such a region exhibit many well-scoring overlaps because shifting a well-scoring alignment by the length of the repeated pattern leads to another alignment with low error rate. The underlying relative position in the genome is not recoverable from alignments alone.

of copies. This entails that flanking sequences of almost all copies are further repeat copies and therefore not unique. Read lengths would have to increase by orders of magnitude before spanning reads for these regions can be sequenced.

Interspersed repeats and tandem repeat clusters are the current bottleneck in de novo genome assembly resolution and constitute the focus of this thesis.

Part II

Preprocessing

Chapter 3

Preprocessing Preamble

In this thesis, we present algorithms for two distinct problems in genome assembly. One problem is the disambiguation of repeat copies from families of interspersed repeats. The other problem is the resolution of tandem repeats, that is, repeats whose copies occur right next to each other in the genome. Especially challenging is the resolution of repeat complexes where dozens of repeat copies are tandemly repeated.

The solutions to both of these problems require the detection of disambiguating differences between repeat copies. However, for both repeat types knowing these differences is not sufficient due to the presence of noise. Therefore, the strategies for resolving these repeat types use the same preprocessing steps to extract disambiguating differences. Thereafter, they use different and unrelated analysis pipelines.

In the following, we first describe the shared preprocessing steps in part II and then present the algorithmic ideas for the resolution of interspersed repeats in part III. The pipeline for the assembly of tandem repeats will be presented in the next and final part IV of this thesis.

We will first describe our method for computing multiple sequence alignments (MSAs). In MSAs sequences are aligned by inserting gaps such that differences and similarities between the sequences line up and can be analysed and extracted for downstream processing.

Then, we will explain the statistical analysis for detecting systematic differences between repeat sequences and detail the choices we made for extracting these putative repeat copy differences.

After the introduction of these preprocessing steps, we will demonstrate how the extracted differences can be used to identify and classify sequences stemming from distinct repeat copies. We also show how these classes of possible repeat copy groups can be used to resolve interspersed repeat regions in genome assemblies.

Chapter 4

Multiple Sequence Alignment

4.1 Introduction

Multiple sequence alignments (MSA) are used in numerous fields of Bioinformatics, in evolutionary and comparative studies, as well as for different aspects of genome assembly and sequence analysis. In the former fields they strive to identify homologous bases and proteins, that means, bases and proteins with a common evolutionary history. This subsequently allows to investigate evolutionary processes, for example, by inferring phylogenetic trees. The computation of MSAs for this purpose typically requires incorporating evolutionary parameters into the alignment algorithm, for example, the probability of transitions from one nucleotide to another [40]. Additionally, there is an intricate interplay between inferred phylogenies and MSAs: While phylogenetic trees are inferred on MSAs, knowing the phylogeny can also help to avoid errors in the MSA computation [41]. Therefore, the reconstruction of phylogenies and the computation of MSAs are inextricably linked. The tree and the MSA can even be calculated simultaneously [42], though at a large computational cost.

In genome assembly, however, computing MSAs is fortunately less complex. The aim is usually to compute a consensus sequence on a set of given sequences covering a genome region by averaging out sequencing error, or, as is the case here, to detect underlying differences among highly similar sequences that might be obscured by sequencing error. In the following, we define the term MSA mathematically and introduce a task-appropriate MSA optimality criterion. Then we present the algorithms used to compute and optimise the MSA according to this criterion. Finally, we empirically justify the chosen optimality criterion and discuss some of the limitations of our approach.

4.2 Definition

Given a set of DNA sequences S_i with $S_i \in \{A, C, G, T\}^{l_i}$ for all $i < n$, an MSA of these sequences S_i is a set of derived sequences $S'_i \in \{A, C, G, T, \_, -\}^L$, where each S'_i is obtained by inserting the additional symbols $\{-, \_\}$ (for alignment gaps and coverage gaps) into S_i such that all sequences S'_i have the same length L. Coverage gaps are gaps between different sequences in the same row of the MSA, or between the end or the beginning of a sequence and the end or beginning of the MSA. Alignment gaps are opened within a sequence while aligning it to the other sequences to accommodate for bases in the other sequences that have no corresponding base in the sequence itself.

This mathematical definition of an MSA is a generalisation of the definition of a pairwise sequence alignment. But the purpose of computing an MSA is often the exact opposite of the purpose of computing a pairwise sequence alignment: Pairwise alignment intends to identify the similarities between sequences, while an MSA aims to find the differences between sequences. Usually the sequences for which an MSA is computed are already known to be similar. Thus, the goal of an MSA is to arrange them in such a way that the differences between (groups of) the sequences become visible.

This is especially true when the differences of interest are obscured by a comparatively high number of spurious differences introduced by random sequencing error. In that case a single pair-wise sequence comparison is not capable of determining which bases differ due to sequencing error and which bases differ for biological reasons (e.g. repeat copies). For instance, two PacBio reads sequenced from the exact same section of a genome will contain such a substantial amount of sequencing errors, that their similarity is only between 70-80%. However, by comparing each sequence to numerous others, the bases with majority support can be determined for each of them. Finding true repeat copy differences among the deviations from the respective majority base is then within the realm of statistical analysis (see Chapter 5.1).

When computing an MSA, we attempt to optimise a scoring function that is appropriate for the task at hand. This scoring function is usually given by a substitution matrix that scores the mismatch of bases, and a gap penalty function that punishes the introduction of gaps. The sum-of-pairs unit score $\sum_{k<L} \sum_{j<n} \sum_{i<n} U(S_j[k], S_i[k])$ where $U(b, b) = 0$ and $U(b_1, b_2) = 1$ for $b_1 \neq b_2$, L the width of the MSA and n the number of sequences, constitutes a simple example of an MSA scoring function. Here, the substitution matrix and the gap penalty function are both denoted by $U(b_1, b_2)$ as $b_1, b_2 \in \{A, C, G, T, \_, -\}$.

4.3 Algorithm

The definition of the MSA and its scoring function is a straightforward generalisation of the pairwise alignment case. The question arises if for any number of sequences an actual MSA can be computed by generalising the Needlemann-Wunsch algorithm 1. However, as the dynamic programming matrix that has to be computed is n-dimensional, the runtime complexity of this generalized algorithm is $\mathcal{O}(L^n)$.

Given this exponential time and space complexity, in practice initial MSAs are computed using various heuristics and the resulting suboptimal MSA is then further optimised in an additional processing step. There are two basic approaches to refining an initial MSA (and combinations of these approaches): Optimising the MSA row by row, or optimising it column by column. The row-by-row optimisation method that we present in this chapter was introduced by Anson [43]. The column by column optimisation is, for example, used by the PacBio consensus program quiver [44] that we use in Chapter 10.5 to compute the final consensus sequence of the finished assembly of the Drosophila Histone Complex. The fundamental difference between these two basic MSA refinement approaches is that only the row-by-row approach sidesteps the effect of the exponent of the runtime complexity. The column-by-column optimisation is therefore limited to narrow sections of an MSA, that means a small number of adjacent columns, and is consequently only used for local refinement.

We implement a version of the row-by-row paradigm that optimises the unit score of pairwise alignment as defined above [45]. This choice is motivated by the specific properties of the task at hand. We do not attempt to arrange sequences according to a biologically defined similarity score. This would require the use of dedicated scoring functions that encode, for instance, knowledge of bio-chemical similarities or of likely evolutionary changes among protein-coding sequences. Instead, we align sequences whose differences are to the largest extent a result of random sequencing error. This means, that the only task-specific knowledge that we might exploit with our scoring function are the relative frequencies of indels (i.e. insertions and deletions) and substitutions. Given the abundance of insertions in our data, it might seem logical to penalize alignment gaps less severely. However, this would also favour arranging substitutions in separate columns, which seems undesirable. Moreover, encouraging alignment gaps will increase the breadth of the MSA significantly, which is a major runtime impediment of the refinement algorithm. We therefore implemented the unit scoring as it accelerates computation and ease of implementation.

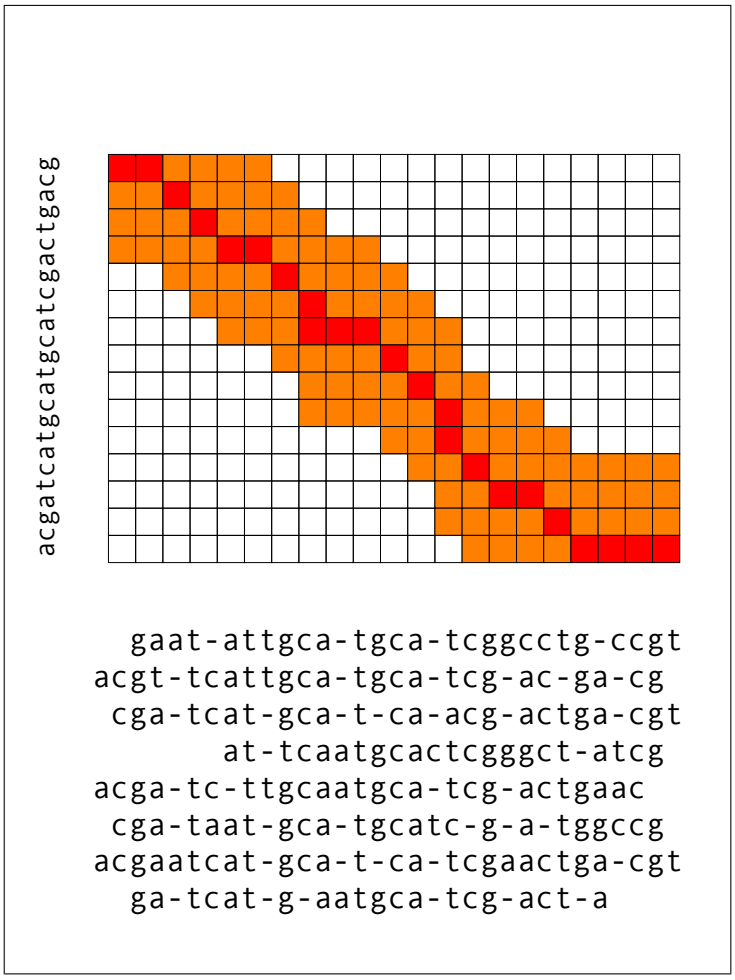

Figure 4.1: When re-aligning a sequence to the remaining MSA we only compute a part of the dynamic programming matrix. This part is a band (in orange) around the backtracking path (in red) of the existing alignment of the sequence to the MSA. The width of this band is a parameter that allows us to fine-tune the accuracy versus speed trade-off.

Even when using unit scoring, MSAs of PacBio reads can grow to a considerable breadth because an abundance of erroneous insertions that are different for each read has to be accommodated. This makes the optimal global re-alignment of a sequence to the MSA prohibitively costly.Therefore, we restrict our search for the alignment of a given sequence to a small part of the dynamic programming matrix, see Figure 4.1. This part of the matrix is defined by a band that is placed around the backtracking path of the alignment that resulted from the last round of realigning. The initial multiple sequence alignment is calculated by aligning the sequences to a template se-

quence. This template sequence can be one of the sequences, a consensus of several sequences, or a repeat sequence that has already been published in the literature. In this initial step, due to the limited length of the template and because each pair-wise alignment is calculated only once, we can afford computing the entries of the entire dynamic programming matrix, see Algorithm 1. Once the initial MSA is computed, it is refined by the row-by-row re-alignment algorithm, see Figure 4.2.

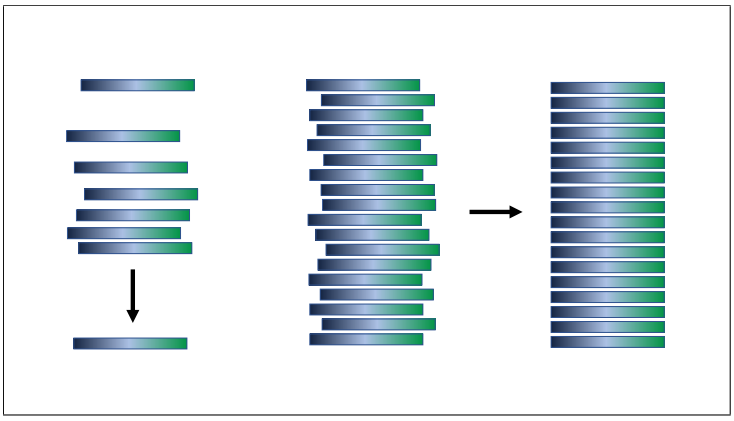

Figure 4.2: First we align all sequences to a template, building an initial alignment. This template can be one of the sequences, a consensus of several sequences or a known repeat sequence from the literature. In a second step, this initial alignment is refined via row-by-row optimisation of the sum-of-pairs unit score.

In our pairwise realignment algorithm, we optimise a given MSA M_{ij}, where $i < n$ sequences S_i are arranged into $j < m$ columns. For the i-th sequence we define a *path* $T_i = \{j < m | M_{ij} \in \{A, C, G, T\}\}$, which encodes the existing alignment of the sequence S_i to the MSA. We also define a scoring function $\omega_i(j, b) = \sum_{k<n|k\neq i \wedge M_{kj} \neq b \wedge M_{kj} \neq -} 1$. This scoring function penalises all entries in a column j (apart from coverage gaps) that are different from the base $b \in A, C, G, T, -$. Note, that we exclude the entries of the i-th row. This can be interpreted as removing the sequence S_i from the MSA to then align it back to the remaining MSA. Additionally, we define the gap penalty $g(j) = \sum_{k<n|k\neq i \wedge M_{kj} \in \{A,C,G,T\}} 1$ as the number of bases (without gaps) in the column j after removing S_i.

The pairwise realignment algorithm, see Algorithm 2, is a straightforward generalisation of the Needlemann-Wunsch algorithm. Instead of aligning a sequence to another sequence, we align a sequence to an MSA. The scoring function defined above allows us to score the mismatch between columns of the MSA and bases of the sequence that is being re-aligned. The initialisation avoids penalising alignment gaps before the beginning or after the end of

the sequence because not every sequence will stretch over the entire width of the MSA. The core of the algorithm consists of a dynamic programming procedure: The calculation of the alignment is broken down into alignments of sequence/MSA prefixes that can be dynamically extended by either matching the next base and column, or by inserting a gap in either sequence or MSA. The computed prefix alignment scores are stored in an dynamic programming (DP) matrix D_{ij}. After filling the DP matrix, the best scoring global alignment is extracted by backtracking through the DP matrix. The backtracking starts with the entry of the DP matrix containing the score of the full alignment of the sequence to the MSA. Then we successively go back to the entry that was used to calculate the current score until we arrive at the entry D_{00}.

Algorithm 2 Pairwise ReAligner

1: $S_i \leftarrow$ sequence in row i
2: $\omega_i \leftarrow$ scoring function for S_i
3: $T_i \leftarrow$ path for row i
4: $D_{xy} \leftarrow$ DP matrix
5: $n_i \leftarrow$ length of S_i
6: $m \leftarrow$ number of columns
7: $bh \leftarrow$ width-of-band/2
8: **procedure** DYNAMIC PROGRAMMING
9: $\quad D_{0,y} \leftarrow 0, 0 \leqq j \leqq m$
10: $\quad D_{x,0} \leftarrow$ INT_MAX, $1 \leqq x \leqq n_i$
11: $\quad$ **for** $1 \leqq x \leqq n_i$ **do**
12: $\quad\quad$ **for** $\max(1, T_i[x] - bh) \leqq y \leqq \min(m, T_i[x] + bh)$ **do**
13: $$D_{x,y} \leftarrow \min \begin{cases} D_{x-1,y-1} + \omega_i(y, S_i[x]) & \text{Base-column match} \\ D_{x-1,y} + g(y) & \text{New column} \\ D_{x,y-1} + \omega_i(y, -) & \text{Alignment gap} \end{cases}$$
14: **procedure** BACKTRACKING
15: $\quad x \leftarrow m$
16: $\quad y \leftarrow n$
17: $\quad E \leftarrow \emptyset$
18: $\quad$ **while** $x > 0, y > 0$ **do**
19: $\quad\quad$ **if** $D_{x,y} = D_{x-1,y-1} + \omega_i(y, S_i[x])$ **then**
20: $\quad\quad\quad E \leftarrow m + E$
21: $\quad\quad\quad x \leftarrow x - 1$
22: $\quad\quad\quad y \leftarrow y - 1$
23: $\quad\quad$ **else if** $D_{x,y} = D_{x-1,y} + g(y)$ **then**
24: $\quad\quad\quad E \leftarrow d + E$
25: $\quad\quad\quad x \leftarrow x - 1$
26: $\quad\quad$ **else if** $D_{x,y} = D_{x,y-1} + \omega_i(y, -)$ **then**
27: $\quad\quad\quad E \leftarrow i + E$
28: $\quad\quad\quad y \leftarrow y - 1$

4.4 Empirical Accuracy Assessment

In the following, we empirically assess the performance of our realignment tool. We start with several observations.

By design, we compute a part of the DP matrix that contains the backtracking path of the current alignment. Therefore, in the worst case, our re-alignment computation will either result in the current alignment or an

alignment with the same score. This means that the pairwise alignment score is guaranteed to decrease monotonously. Therefore, when the pairwise alignment score converges, we know that within the band of evaluated alignments each sequence is optimally aligned to the MSA. This, however, is not necessarily the globally optimal MSA. We illustrate this by constructing a local minimum:

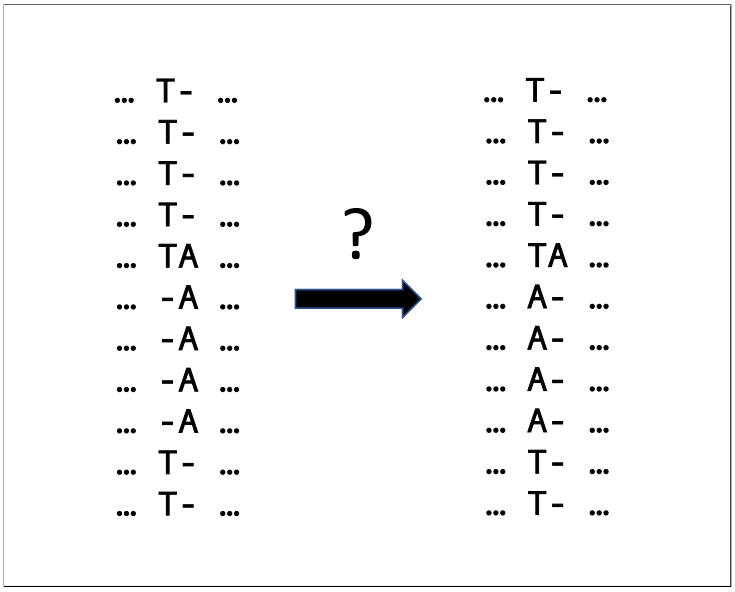

Figure 4.3: In this MSA we observe a substitution of the prevalent nucleotide T by a nucleotide A in several rows. In the left part of the Figure, this substitution is arranged in a separate column. On the right part of the Figure, we see the arrangement that reflects a substitution, with both variations being located in the same column. This is also the better scoring version as it arranges the $-$'s into a single column. However, under row-by-row realignment, the left version is unlikely to transform into the right version, as in every realignment the A will be placed with the other A's.

The arrangement in Figure 4.3 constitutes a local optimum and is not likely to be changed by additional realignments. We can escape this type of local optimum via a column optimisation. However, the practical impact of this approach is small, as there are other types of local optima that comprise several columns. Such wider local optima, spanning more than one column, are increasingly computationally expensive to optimise.

Additionally, it is not obvious a priori that a chosen optimisation criterion correlates closely with the performance in the task for which the MSA is computed. In our case, the downstream use involves the detection of underlying variations between repeat copies among a substantial number of

error-induced differences between reads. These repeat copy variations are detected via a statistical method that we present in Chapter 5.1. In this method we compute a negative log-probability to determine the statistical significance score for each variation in a column. The detection of the true variations that do distinguish repeat copies depends on their scores being distinguishable from error-induced noise.

Thus, this statistical significance metric for distinguishing variations directly reflects the performance of our realignment heuristic in the context of repeat resolution. Therefore, it allows to assess our choice of the specific alignment scoring criterion. Figure 4.4 shows that there exists a strong positive correlation between decreasing pairwise alignment score and increasing statistical significance score of distinguishing variations on simulated data. The data set is simulated using the equidistant paradigm (see Section 6.2) with a coverage of $20\,X$, 20 repeat copies, 5% repeat copy difference differences, 1000 bp repeat length, and the empirical PacBio error profile. We simulate the data to gain statistical significance scores in an intermediate range to illustrate the full correlation without range restriction.

The runtime of our PairwiseRealigner (available at https://github.com/PhilippBongartz/RepeatResolver under GNU GPL v3.0) is asymptotically linear in the number of sequences. Therefore, it is well-suited for analysing the extremely large repeat families that we attempt to resolve. The bandwidth parameter can be used to adjust the size of the band in the DP matrix that is being computed. This allows for tuning the runtime versus accuracy trade-off. Only computing a band of the DP matrix also decouples the runtime from the MSA width. This is important as the MSA width can change considerably during the refinement process.

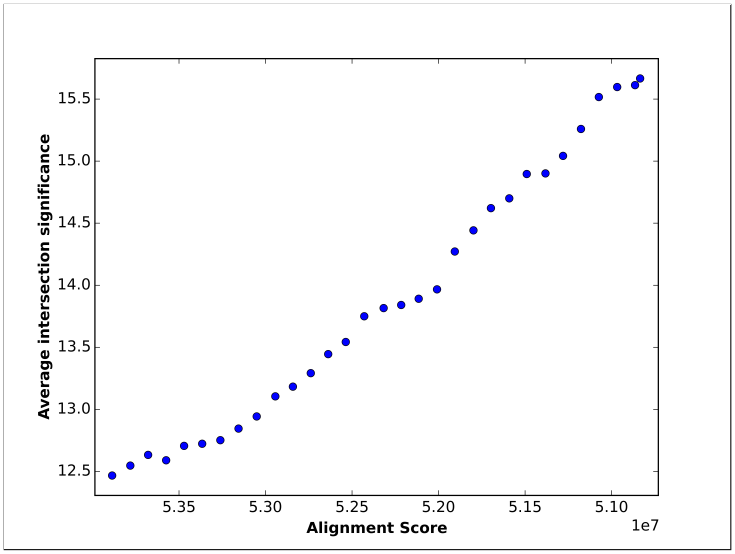

Figure 4.4: We depict the average statistical significance of the top 100 distinguishing variations in simulated data as the pairwise alignment score is decreased by our PairwiseReAligner tool.

Chapter 5

Variation extraction

5.1 Introduction

After computing and refining an MSA which comprises the repeat sequences, we can now extract variations in these sequences. These variations allow us to distinguish between the different repeat copies. Because of the substantial error rate and the large number of sequences, each MSA column is likely to contain all four bases as well as alignment gaps and coverage gaps. However, completely random error will lead to statistically independent variations in columns that are sufficiently distant to each other (i.e. have sufficiently many columns between them). In contrast, those variations that are typical for the same or strongly overlapping subsets of repeat copies will correlate.

Mathematically, we can express this using combinatorics. Every base $b \in \{A, C, G, T, -\}$ in each column i of the MSA defines a group of sequences G_i^b. This group G_i^b consists of exactly those sequences that exhibit a b at column i. The likelihood of the intersection $G_j^{b2} \cap G_i^{b1} := \{x | x \in G_j^{b2} \wedge x \in G_i^{b1}\}$ of two statistically independent groups exceeding a certain size, is described by the cumulative hypergeometric probability $CHG(G_j^{b2}, G_i^{b1}) := \min(P(X \geqq k), P(X < k))$, with $P(X = k) = \frac{\binom{K}{k}\binom{N-K}{n-k}}{\binom{N}{n}}$, where N is the shared coverage of both groups, K is the size of G_i^{b1} restricted to the shared coverage, n is the size of G_i^{b1} restricted to the shared coverage, and finally k is the size of the intersection $G_j^{b2} \cap G_i^{b1}$ [46, 47]. We use the shared coverage because not every sequence has data for every column, so care has to be taken to restrict this calculation to sequences that have data in both columns.

This is the well-known urn problem "drawing without replacement": From an urn with K white and $N - K$ black balls we draw n balls. What is the probability of drawing k white balls? In our use case the "drawing" is just

the examination of the second base group and "drawing a white ball" represents a sequence that contains both variations and therefore is contained in the intersection of the two base groups.

Consequently, a lower CHG-value yields a higher likelihood for the two base groups being present due to non-random variation. We call variations that define base groups that exhibit intersections with a CHG-value below a certain threshold "statistically significant variations". It might happen that some true copy differences do not produce statistically significant variations. However, this implies that there are no other variations delimiting a similar subset of sequences. As our downstream processing depends on redundant variations reinforcing each other, not identifying these copy differences will not impede accuracy.

Even without discriminating variations, the CHG-scores will vary from base group comparison to base group comparison. In that case the expected lowest CHG-value is the inverse of the number of comparisons. This inverse is therefore a sensible lowest threshold. All statistically significant variations above this threshold are more significant than the most significant error induced variation is expected to be. However, care has to be taken to avoid comparing columns in immediate vicinity. In an MSA close columns are not statistically independent. Therefore, we only compare base groups from columns that have at least 40 columns between each other. Moreover, PacBio reads do not always exhibit the same error rate. Some reads have a systematically higher error rate along the entire length of their sequence or at least in substantial parts of the sequence. If sequence S has such an elevated error rate, it is more likely to have a certain error induced variation in column x and it is also more likely to have a different error induced variation in column y. This induces a weak statistical dependence between errors in columns that are located far apart and can yield low CHG-values slightly more likely than predicted.

For our interspersed repeat resolution method this does not impact accuracy too much, so we can use the lowest threshold. As we will see, in this case downstream processing alleviates the pitfalls of low-quality data points. For the assembly of gene complexes in Part IV of this thesis, however, we will only use higher quality variations, as judged by the CHG-value.

We compute the CHG using the Gnu scientific library GSL [48]. The pairwise comparison of base groups requires quadratic time in the number of MSA columns. However, we can reduce compute time by almost two orders of magnitude by only comparing columns which contain bases in a majority of rows. Due to the high number of indels in PacBio data, each MSA contains numerous columns that do not contain substantial amounts of information, but rather accommodate inserted erroneous bases. This means that ignoring

these columns for allowing a substantial computational speed-up represents a reasonable trade-off. However, for the data sets used in this thesis, we do not yet have to apply this short-cut. Instead, we leverage parallelisation to gain comparable wall clock speed-ups.

Part III

Interspersed repeats

Chapter 6

Simple Repeat Resolution

This chapter is based on a peer-reviewed publication:

Philipp Bongartz.
"Resolving repeat families with long reads"
BMC Bioinformatics. 2019 May 11. 20:232.

Code and data available at
https://github.com/PhilippBongartz/RepeatResolver

Contributions: "Resolving repeat families with long reads" is a single author paper where problem selection, algorithm design, implementation, testing and writing was done by Philipp Bongartz.

6.1 Introduction

Long read sequencing technologies [30, 31, 32, 49] have brought us almost within reach of perfect genome assemblies. For circular bacterial genomes, full resolution is already considered as being the current standard for assemblers that are based on long-read sequencing technologies [33]. Perfect bacterial genome assemblies are achieved by spanning repeat elements with reads that are long enough to be anchored in unique sequences on both sides of the repeat [36]. However, eukaryotic organisms generally contain repeat families that are not spanned by the current read lengths [37]. In complex genomes, these repeat families are the most prevalent reason for assembly breaks. Frequently, most interspersed repeats originate from but a few repeat families [35]. As the number of indistinguishable repeat copies grows, it becomes increasingly unlikely to find a unique path through an assembly graph (see Figure 2.2). Thus, the only strategy to resolve a given repeat

family directly from the sequencing data is to detect distinguishing features between the copies of a repeat family. Several approaches to detect and utilize such repeat differences have been proposed [46] [47] [50]. However, these existing repeat resolution methods are geared toward 2-10 repeat copies. This limits their applicability to only a small subset of repeat structures as they occur in complex genomes [35, 37].

Here, we present a method that is similar to that of Tammi [47]. It uses clustering heuristics to overcome the limitation of Tammi's method to an error rate below 11%, and to repeat families with 10 or less copies. For simulated data sets with distinct repeat structures we are able to resolve repeat families with 100 copies under the typical PacBio error rate of 15%, while assuming an absolute number of repeat copy differences comparable to that of other methods. Our analysis of Drosophila melanogaster transposons proves that similar results can be achieved with empirical data, while our comparison to an existing repeat resolving tool for long read data demonstrates the improved accuracy (82.9% vs 50.6% resolved copies) and reduced runtime of our method.

6.2 Data sets

Simulating repeat data

To avoid overfitting our method to one specific repeat family structure, we use three different approaches to create simulated repeat families with $\geq x\%$ difference between copy pairs.

Equidistant Simulations: In *equidistant* simulated repeats, each copy has $x/2\%$ variants that distinguish it from the initial template. In pairwise comparisons these per-copy differences then yield a difference of $x\%$.

Distributed Variants Simulation: Additionally, we conduct a *distributed* variant repeat family simulation. Here, we distribute each variant over a subset of copies. Thus, each copy consists of an intersection of variants. In turn, these variants characterize a subset of copies rather than a single copy. Adding $3x\%$ variants again yields an expected difference between copy pairs of $x\%$, as the probability of two sequences not sharing a specific variation can be calculated as $\int_0^1 2(r \times (1-r))dr = \frac{1}{3}$.

Tree-like Simulations: Finally, we simulate *tree*-like variant repeats. Here we create a repeat family by building a binary tree of copies, each copy obtaining $x/2\%$ variants that distinguish it from the parent copy. The leaves of this tree create a repeat family where sister leaves show a difference of $x\%$. The binary tree simulates a simplified version of the phylogenesis of repeat

families via copying and mutation [51] [52].

Our three simulation scenarios pose distinct algorithmic challenges in variant detection and copy disambiguation. In general, a repeat resolving method should perform well under all three simulation scenarios. To benchmark our algorithms, we create synthetic data sets for each scenario described above. Each simulated data set contains 100 copies derived from a randomly created 30 kbp template. These 100 copies are diversified with equal numbers of substitutions, insertions and deletions of single bases. We create data sets with 0.1%, 0.5% and 1% minimal copy differences respectively. To each copy we add two unique 10 kbp flanking sequences on both sides of the 30 kbp repeat. From these copies 30-40 X coverage is randomly sampled, with the read length distribution modelled after the empirical PacBio data set described in the following paragraph [53]. Each read exhibits the typical PacBio error rate of 11.5% insertions, 3.4% deletions and 1.4% substitutions.

Transposon data sets

As simulated data is often less challenging to analyse than real data, we also test our algorithms on several empirical PacBio data sets obtained from a subline of the ISO1(y;cn,bw,sp) strain of Drosophila melanogaster [53]. Each data set is created by selecting reads that fully map to a transposon template. These templates are taken from the canonical transposon sequence set [54], with a length cutoff of >4 kbp, as resolving even shorter repeat sequences is not required due to current read lengths. There are seventeen transposon data sets numbered from 0 to 21, with the missing numbers indicating transposons below the length cutoff. The transposon template length varies between 4.4 kbp and 7.5 kbp with a mean of 5.8 kbp and a median of 5.3 kbp, the copy numbers lie between 7 and 157. Due to the selection of reads that fully fit the template, the initial sequencing coverage of 90 X is reduced to 35-54 X. The ground truth for the resolution of each repeat family is manually determined by clustering the flanking sequences of every transposon data set according to the Levenshtein distance [55].

Extracting distinguishing variants

In a pre-processing step, the simulated reads are arranged into a multiple sequence alignment (MSA) [45]. This initial MSA is computed by aligning all reads to a repeat family template. In our test data sets we use simulated repeat templates and templates extracted from existing genome assemblies, but in practice any consensus sequence of a repeat family of interest can be

used. As described in Chapter 4, the initial MSA is subsequently refined by realigning all sequences until the sum of pairwise alignment scores does not further improve.

Due to the high error rate, each column of this refined MSA contains all four bases as well as coverage and alignment gaps. To find the columns where this variation can be explained by significant differences between repeat copies that are beyond random error, we conduct a statistical analysis of the co-appearance of bases in different MSA columns. Every base $b \in \{A, C, G, T, \}$ in column j defines a group G_j^b containing the sequences that have a b in column j. We then proceed to extract all base groups that are statistically significant as detailed in Chapter 5 .

6.3 Refining base groups

A completely error free base group G extracted from the MSA could be modelled as a union of true copy groups T_i with $i \in I_G$. Here T_i contains exactly those reads sampled from repeat copy number i and I_G describes which copies comprise the base that defines the base group G. Due to the existing error rate, G will contain a fraction p of these true positives in the T_is with $i \in I_G$ and also, a fraction q of the sequences in the T_is with $i \notin I_G$ as false positives.

In the following, we describe a framework to refine such groups and to identify those, where the refinement induces a low proportion q of false positives and a high proportion p of true positives. In this analysis, we assume that all groups have been restricted to contain only sequences that show no coverage gaps in any of the MSA columns from which the groups are derived.

First, we calculate a clique C of n groups G_j, with $j \in J$ and $|J| = n$, that share the most significant positive intersection with G. A positive intersection is an intersection that is larger than expected by chance, see Section 5.1. The parameter n is chosen empirically. This is a clique in the graph that contains groups as nodes and statistically significant intersections between those groups as edges. Now we can define a consensus group $C_k := \{s | s \in G_j \text{ for } j \in J \text{ with } |J| > k\}$ for every cut-off $k \leq n$. The cut-off k determines in how many groups of the clique a given read has to occur in order to be included in the consensus group C_k. If the groups that constitute a clique all share the same I_G, that is, they all describe the same ground truth group, the following formula gives the probability p that a specific read is in the consensus group C_k:

$$p = \sum_{l>k}^{n} \sum_{i+j=l} \Pr(i, l, p) \times \Pr(j, n-l, q)$$

This formula is a sum over the probabilities that a given read occurs in exactly l out of n groups, with $l > k$. A given read occurs in exactly l groups if it occurs in i groups as true positive, that is, an element of the T_i with $i \in I_G$, and in j groups as false positive while $i+j = l$. These probabilities are given by Pr(.,.,.), the probability mass function of the binomial distribution, which takes as parameters the probabilities p,q of a group element being a true positive or a false positive, respectively.

The fraction of false positives in C_k coming from the T_i with $i \notin I_G$ is described by the cumulative probability function $\sum_{i=k+1}^{n} \binom{n}{i} q^i (1-q)^{n-i}$ of the binomial distribution. As shown in Figure 6.1, this fraction of false positives decreases quickly with increasing cut-off k, while the number of true positives remains constant for larger k. This is due to p being significantly larger than q. In reality, the subset of T_is described by the groups that form a clique can vary considerably. Also, not every T_i is described by either all or none of the groups. If we consider the T_is separately, we find that if a T_i is contained in m groups of the clique, we expect the fraction $\sum_{l>k}^{m} \sum_{i+j=l} \Pr(i, l, p) \times \Pr(j, n-l, q)$ of the elements of T_i to occur in the consensus group C_k. In this formula l is the number of groups, in which an element occurs. This number is split into i true positives in the m groups that describe T_i , and j false positives in the groups not describing T_i. For low cut-offs k, this fraction is close to 100% and we expect all elements of T_i to occur in C_k. As k increases, the expected number of true positives decreases to zero. So, for every T_i there are three separate value ranges for the cut-off k, the *perfect* range, in which all elements are contained, the *dropping* range, in which the number of true positives decreases, and the *zero* range, where no elements of T_i are part of C_k any more, see Figure 6.1 for an illustration.

For distinct T_is the k value ranges for perfect and dropping accuracy will be different due to different values of m, the number of groups describing T_i. If k is high enough for the number of false positives from the T_is *not* described by any clique members to decrease to zero, the number of elements of C_k is equal to the sum over the cardinalities of $T_i \cap C_k$ as given above. As we will see, minimizing the difference between C_k and C_{k+1} allows us to determine the optimal cut-off value k, which places most T_is into, or close to, either their *perfect* or *zero* range (see Figure 6.2).

We call the size difference between C_k and C_{k+1} the *drop-off* between C_k and C_{k+1}. The size of the *drop-off* is determined by the number of T_is for which the cut-off value k is in the *dropping* range. Therefore, a drop-off

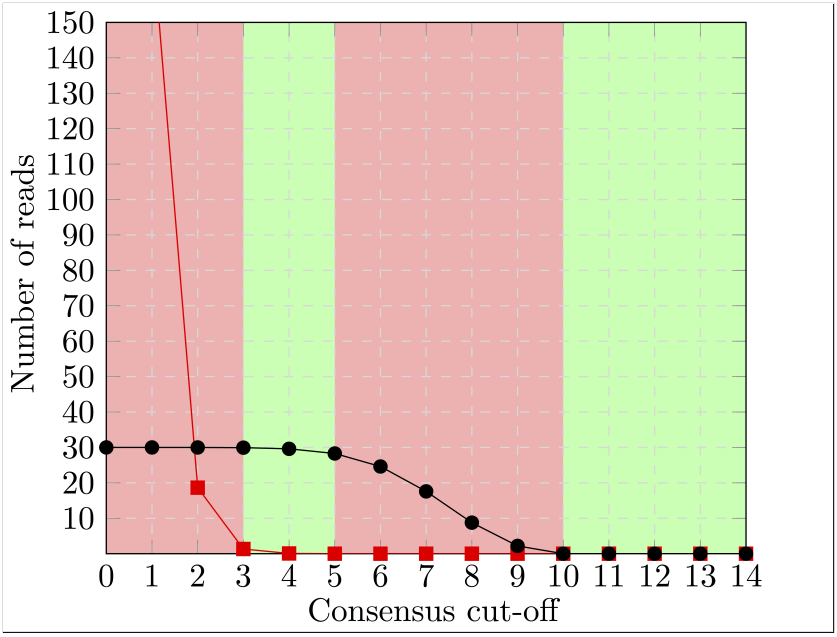

Figure 6.1: If the groups G_j of a clique all describe a single T_i, the number of false positives (red squares) coming from other groups T_j decreases quickly, while the number of true positives (black dots) remains constant until the cut-off value is relatively high. In the green areas, the cut-off guarantees to yield a consensus that either perfectly contains T_i or is completely empty.

close to zero indicates that all T_is are either completely contained in C_k or not contained therein at all. The drop-off allows to determine the optimal cut-off value k for every clique of groups. More importantly, it allows to rank the different clique consensuses by their likelihood of perfectly describing a subset of T_is.

6.4 Clustering

The refinement procedure described above aims to extract sufficiently strong signals to accurately classify the sequences into two subsets of copy versions. It can then be applied recursively to each of the respective subsets. For a recursive subdivision to work, it needs to be highly accurate. Otherwise, noise will accumulate in subsets, making subsequent analyses increasingly difficult. We achieve this increases accuracy in the recursion via the refinement and the drop-off precision estimate for each refinement. The recursive subdivision terminates, once no subset is left that can produce a consensus group which is sufficiently refined for further subdivision.

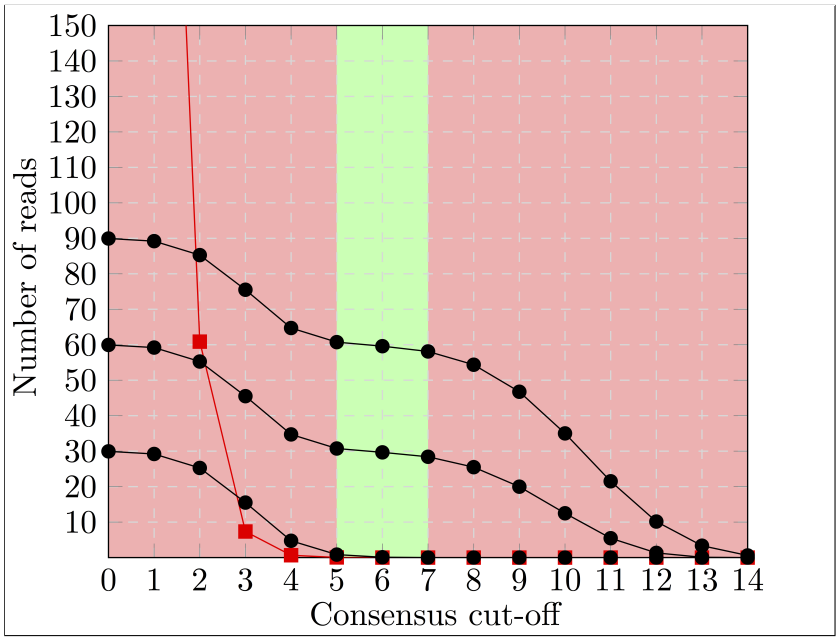

Figure 6.2: If the groups G_j of a clique describe several T_i, the size of the consensus is determined by the aggregate of the true positives (black dots) from each T_i, as well as the false positives (red squares) from the remaining T_js. The green area shows the cut-offs that create a consensus that accurately distinguishes one subset of T_is from the rest. It is exactly this range where the *perfect* or *zero* ranges of all T_i overlap. Furthermore, the aggregate number of true positives (denoted by the uppermost black dots) stays constant in this range.

When the recursive subdivision process has terminated, we apply a simple clustering algorithm to each of the remaining subsets. It assigns reads to centroids according to the differences that are significant for the subset. To that end, we initially recalculate the statistical significance of each variation restricted to that subset. Only those variations that still show statistically significant intersections of their base groups are then used for clustering. For each read we extract the instances of these still significant variations into a so-called read signature. Then, every signature is corrected with the four most similar other signatures for noise reduction and is subsequently used as a centroid. In the first round of clustering, signatures are assigned to centroids by the best fit according to the Hamming distance. This creates a large number of clusters of varying size. Some clusters will have fewer elements than half the expected sequencing coverage. We resolve these small clusters by merging their elements into other clusters, again according to the smallest Hamming distance between the signature and the centroid.

6.5 Resolution

The output of the recursive division step and the subsequent clustering consists of groups of reads that are required to resolve a repetitive region. To resolve a large repetitive region in a genome, we likely have to subdivide our MSA into several sections whose reads are clustered separately. This keeps the number of reads that completely cover each section high. Together, these sections and their clusterings cover the entire repeat. Initially, however, we examine a simplified one-clustering scenario with some of the reads of each repeat copy having a unique flanking sequence on the $5'$ end and the other half having unique flanking sequences on the $3'$ end. We now answer the question how many of these flanking sequences we can accurately connect using the clustering information.

We propose a model to calculate a confidence score for each possible connection. This can be used as a basis for a resolution that takes the probability of mis-assemblies into account as opposed to just providing a "best guess". In the one-clustering scenario it can also be used to assess how well a clustering corresponds to the ground truth. The calculated clusters can be seen as hubs which are entered by incoming reads and can be exited by outgoing reads. We can, for instance, sample a random path from one flanking sequence cluster to another flanking sequence cluster on the other side of the repetitive region. This is done by randomly choosing shared reads that connect the current hub to the next, see Figure 6.3. We use the probability of such a randomly sampled path to connect two flanking sequence clusters to define a unidirectional connection confidence. The full connection confidence is then calculated as the product of the unidirectional connection confidences in both directions. It is normalized such that the connection confidences of all possible connections for a flanking sequence cluster sum to 1.0. Naively, calculating the fraction of randomly sampled paths that start from a $5'$ flanking sequence and end in a $3'$ flanking sequence has a time complexity that is exponential in the number of clusterings. To address this, we break down the calculation into clustering-to-clustering path probability matrices that can be multiplied to give the probability of a complete path. This is possible because the probability of reaching a specific cluster from a given cluster is independent of the path taken to the given cluster.

Let $X_{i<n}$ be the $3'$-flanking sequence clusters and $Y_{i<m}$ be the $5'$-flanking sequence clusters, while $H^k_{i<n_k}$ denotes the $0 < k \leq l$ calculated clusterings of sections of the repetitive sequence that lie in between. The following matrices describe the probability of a randomly chosen read from one particular cluster to connect to a read from a cluster that is part of the next clustering.

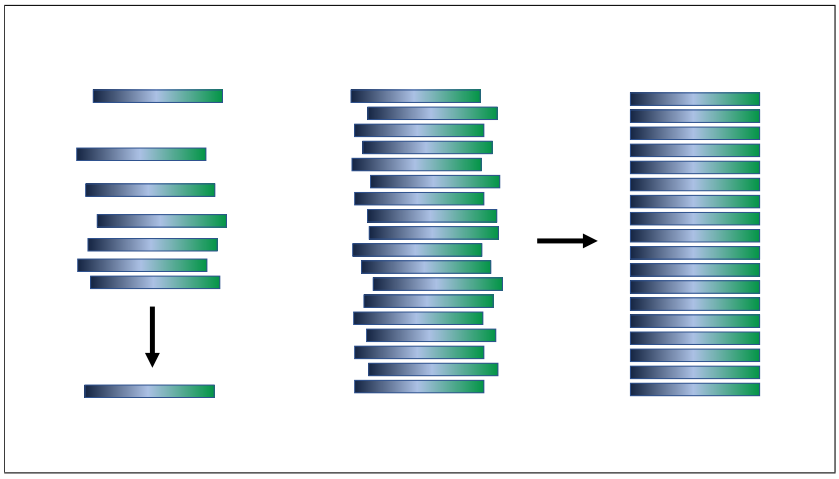

Figure 6.3: An MSA of a repetitive region is divided into 5 sections. The flanking section sequences are clustered according to Levenshtein distance, the repeat section are clustered according to the method presented in this thesis. By randomly sampling shared reads between clusters of successive sections, we can sample paths connecting flanking section with flanking section, providing the resolution of one repeat copy.

$$I_{ij} = \frac{|X_i \cap H_j^1|}{|X_i|} \in \mathbb{Q} \cap [0,1]$$

$$O_{ij} = \frac{|Y_i \cap H_j^l|}{|H_j^l|} \in \mathbb{Q} \cap [0,1]$$

$$C_{ij}^k = \frac{|H_i^k \cap H_j^{k+1}|}{|H_i^k|} \in \mathbb{Q} \cap [0,1]$$

The probability of a longer path can then be calculated by multiplying the connection probabilities between all clusters along the path. The overall connection probability is calculated by summing over the connection probabilities of all possible paths. In the one clustering scenario, the path possibilities are given by the different clusters that can be taken, so the probability of connecting X_i and Y_k is $P_{ik} = \sum_{j<n_1} I_{ij} \times O_{jk}$. In matrix notation this simplifies to $P = IO$. For longer paths we can inductively expand this probability matrix for any number of clusterings. Thus, the overall probability of connecting X_i to Y_k for l clusters H^k is $P = IC^1C^2 \dots C^lO$. This stepwise path-probability calculation reduces runtime complexity to l matrix multiplications. The matrix size does not exceed the estimated number of repeat copies. In the multi-step calculation of connection confidence, we analogously

multiply the path-probability in both directions and normalize, so that the confidences of all possible connections of a flanking sequence cluster sum to 1.0.

Chapter 7

Simple Repeat Resolution Results

This chapter is based on a peer-reviewed publication:

Philipp Bongartz.
"Resolving repeat families with long reads"
BMC Bioinformatics. 2019 May 11. 20:232.

Code and data available at
https://github.com/PhilippBongartz/RepeatResolver

Contributions: "Resolving repeat families with long reads" is a single author paper where problem selection, algorithm design, implementation, testing and writing was done by Philipp Bongartz.

7.1 Introduction

We integrated the algorithms for realigning, detecting significant variants, calculating drop-off consensus groups, subsequent hierarchical subdivision, final clustering, and connecting flanking sequences into our repeat resolution tool. We test it on nine simulated data sets, one for each combination of the minimal copy differences 0.1%, 0.5%, 1% and the repeat structures being equidistant, distributed, and tree-like. We also conduct experiments on 17 empirical transposon data sets (see Chapter 6 Data sets). To assess the resolution of isolated clusterings, we use the one-step resolution algorithm introduced above. Here the ground truth groups are used to replace both flanking clusterings. That means, we calculate the connection confidence from a ground truth cluster to itself via the calculated clustering. If the

calculated clustering does not differentiate between two ground truth groups of sequences, each of these groups will not correctly or unambiguously be connected to itself. We call two ground truth groups *connected* if for both groups the other group provides the connection with the highest connection confidence. If this connection is correct, that is, if the two connected groups are identical, we say that the ground truth group or copy group is *resolved* by the calculated clustering. If the connected ground truth groups are different groups, we call the connection a *false positive*. For the simulated data sets we additionally calculate the connection confidences from one flanking clustering to the other flanking clustering via the calculated clustering in between using the multi-step resolution algorithm. Here, we call two flanking clusters *connected* if, for both clusters, the other group provides the connection with the highest connection confidence. A copy group is likewise called *resolved* if both flanking clusters belonging to the copy group are *connected* to each other. The multi-step resolution is necessary for the practical feasibility of our clustering algorithm. We compare the single-step resolution of the simulated MSA sections with the single-step resolution of the transposon MSAs. We additionally examine the relationship between single-step resolution results and multi-step resolution results to assess the applicability of our algorithms to very long repeats.

7.2 Simulated data sets

To make the single-step resolution of our simulated data sets comparable to the transposon data sets, we divide each MSA into six non-overlapping sections that approximately cover 5 kbp of repeat sequence. We then compute clusterings for each of these sections separately. We additionally use these six clusterings to calculate a resolution for the entire repeat, that is, we determine which flanking sequence clusters are connected by applying the multi-step algorithm, see Chapter 6. Figure 7.1 shows that for both, single-step resolution and multi-step resolution, the number of resolved copies is high ($\geq$95%) for all data sets with 0.5% or 1% minimal copy differences. In fact, only the distributed data sets show an appreciable decline in resolved copies between the 1% and 0.5% copy difference data sets. The number of resolved copies for the single-step resolution of the 0.1% copy difference data sets remains high. However, with just 0.1% minimal copy differences the clusterings are not accurate enough to support robust multi-step resolution. Only the tree-like data set with 0.1% copy differences can still resolve more than 40% of its copies over the entire repeat length.

This failure of the multi-step resolution can be predicted from the single-

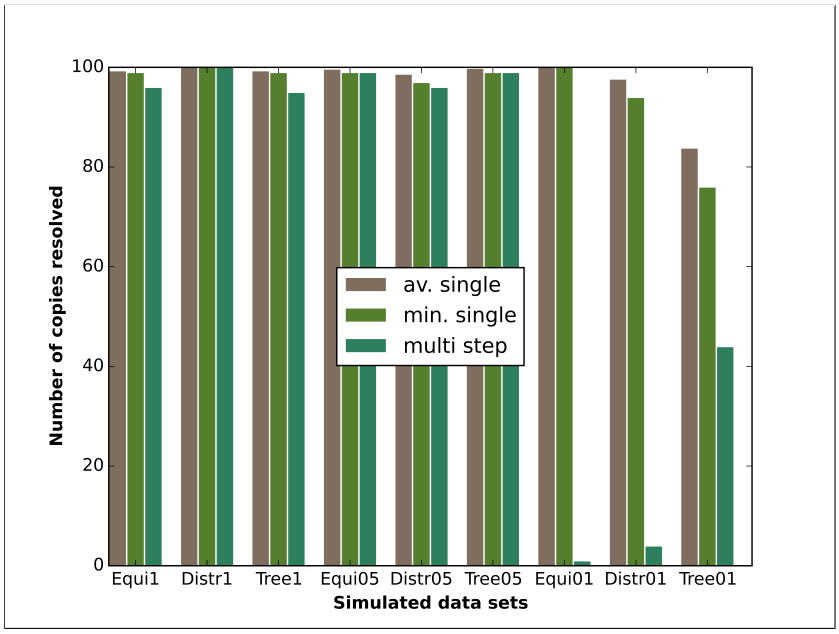

Figure 7.1: We compare the average number of resolved copies of the single-step resolutions, the minimal number of resolved copies of the single-step resolution and the number of resolved copies of the multi-step resolution for all simulated data sets.

step connection confidences. In Figure 7.2 we see, that for data sets with just 0.1% minimal copy differences the connection confidences are generally very low, with the exception of the tree-like data set. The tree-like data set however, has on average only 84% resolved copies in the single step resolution. The fraction of unresolved copies compounds over six resolution steps, which explains the 44% multi-step resolved copies for that particular data set.

The simulated data sets show that multi-step resolution results depend on two properties of the single step resolutions: The fraction of unresolved copies and the connection confidence. In particular, resolved copies with a connection confidence below 0.2 seem not to support the multi-step resolution. It is worth noting that there were no false positives in either single-step resolutions or multi-step resolutions.

7.3 Transposon data sets

The simulated data sets are designed in such a way that the information necessary to obtain a full resolution is available in each data set. Evidently, we cannot expect to replicate these results for empirical data. Some transposon

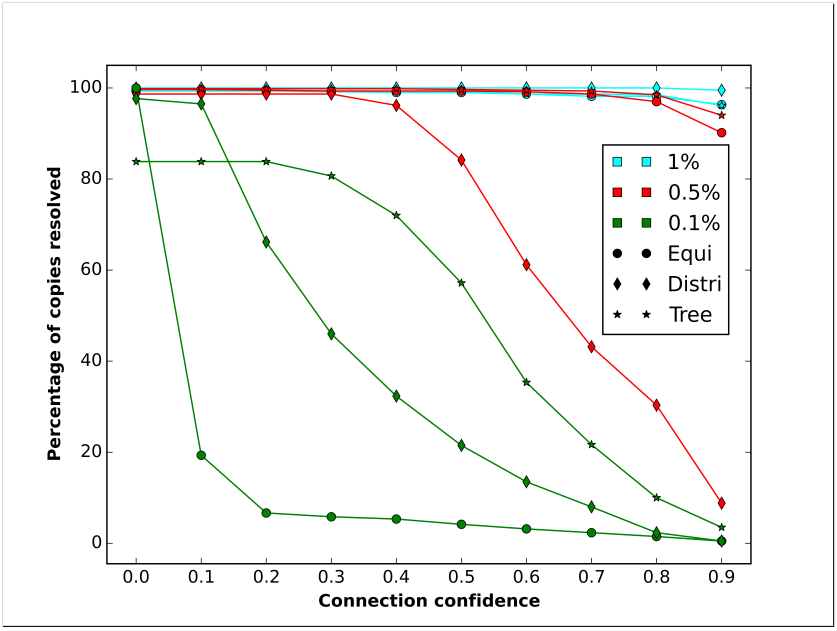

Figure 7.2: This figure shows the average number of resolved copies of the single-step resolution above a specific connection confidence for all simulated data sets.

families will be too highly conserved, while others will contain at least some members that have only recently diverged and did not yet accumulate sufficient differentiating mutations. Moreover, the ground truth copy groups have been obtained by clustering the unique flanking sequences. This process is unlikely to provide a completely accurate ground truth, as is available for the simulated data. Instead copy groups that have not been accurately resolved in the ground truth will add noise to the assessment process,worsening the apparent results. Despite these limitations, according to the metrics introduced above, in all transposon data sets at least 65% of the copy groups are resolved, while only a single data set shows false positives. Several of the smaller data sets are perfectly resolved, while some of the larger data sets are almost perfectly resolved with 35 out of 37, 33 out of 34, and 47 out of 49 copy groups being correctly resolved respectively. The three largest groups with 89, 135 and 157 copies respectively, are resolved by more than 84% on average. Moreover, the initial refinement step with recursive subdivision already resolves more than 50% of all copies. This shows that it constitutes an essential part of the clustering algorithm, see Figure 7.3.

These are surprisingly good results, that come, however, with a caveat. Many of these copy groups are resolved only due to very few or just a single and noisy distinguishing variant. The results on simulated data show that as

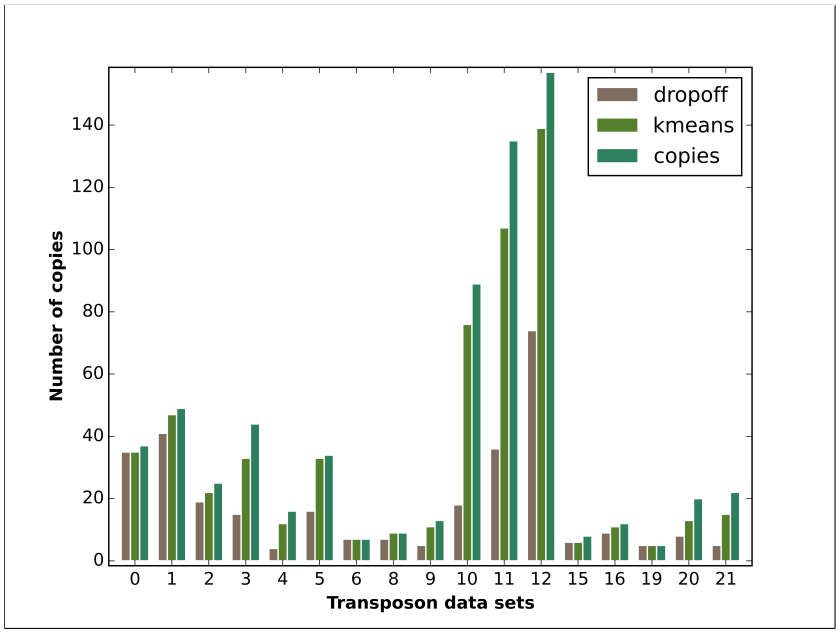

Figure 7.3: This figure compares the number of resolved copies for the dropoff subdivision algorithm and for the full algorithm with the additional kmeans-like clustering to the total number of copies in each transposon data set.

few as 5 differences (0.1% of 5kbp) between repeat copies still allow for single-step resolution. However, they also indicate that the resulting clusterings do not necessarily support the multi-step resolution that is necessary to resolve transposons of a realistic length. To put our results into perspective, we assess the number of statistically significant differences between transposon copies. To this end, we utilize the ground truth information to create consensus signatures for each copy group that consist of the most common base for that copy group in each statistically significant column in the respective MSA. We then compare these consensus signatures to each other to obtain an estimate of the number of differences between transposon copies.

This analysis shows that the percentage of copies that differ from all other copies by at least n bases drops quickly with increasing n, see Figure 7.4. This decrease is mirrored by the number of resolved copies that have a connection confidence above a specific threshold, see Figure 7.5. Around 45-55% of the copies exhibit sufficient differences. Consequently, this is the percentage of transposon copies that achieve a high enough connection confidence to support multi-step resolution.

This would be a relatively low number if the copies with high connection confidence were to be different for every resolution step. However, given that

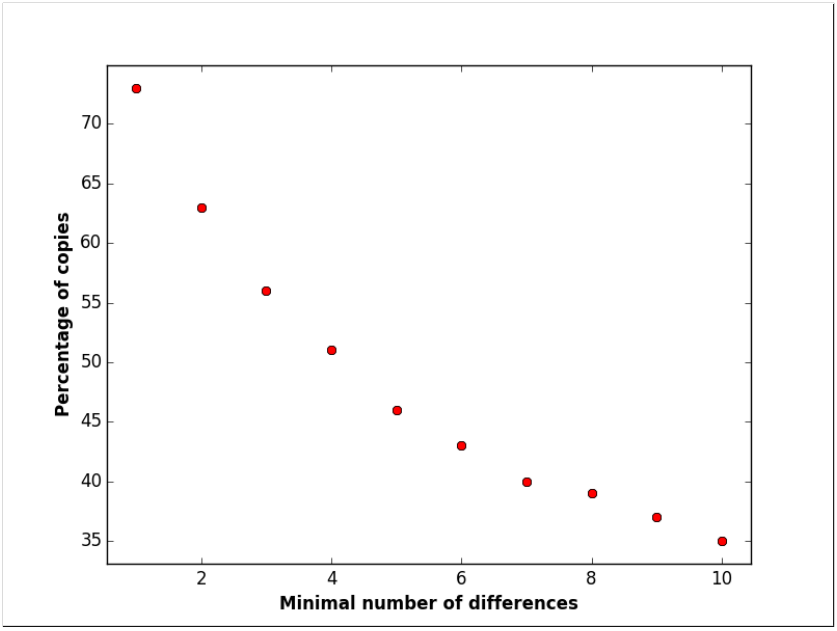

Figure 7.4: The overall percentage of copies in all transposon data sets which exhibit a minimal number of differences compared to the other copies.

the number of differences is a result of the evolutionary history shared by the entire sequence of a copy, it is likely that the resolvable copies will tend to remain unaltered for each section of the MSA. We observe this in the transposon data, where the minimal number of differences to other copies in the first half of a copy and the minimal number of differences in the second half tend to correlate significantly for most data sets (median Pearson correlation 0.483). This correlation, and the total number of differences, is likely to increase with larger MSA sections.

7.4 Comparison with competing method

In this section, we compare our methods against the long read assemblers Canu and MARVEL, and against the dedicated repeat resolving method split_dis that is part of the Daccord package [56].

Full-scale long read assemblers, like Canu and MARVEL, do not use repeat resolution methods that go down to specific differences in the repeat copies. This fundamentally limits their repeat resolution capabilities. In MARVEL, repeat resolution is restricted to the use of spanning reads and the detection of unique combinations of repeat modules [35], making the resolution of a 30 kbp repeat family challenging for MARVEL.

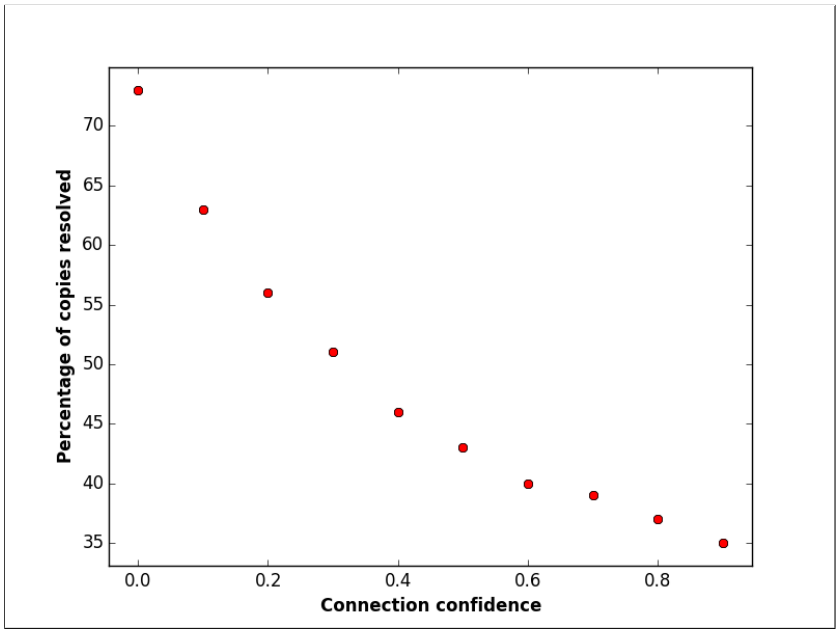

Figure 7.5: The overall percentage of resolved copies in all transposon data sets with a connection confidence above a specific threshold.

Canu has a dedicated repeat resolution step that is based on alignment score differences of corrected reads. In the original Canu paper [33] this resolution step of Canu and the resolution capability of FALCON are benchmarked in a very similar fashion as we do on our simulated datasets: A PacBio read dataset is simulated with a 30 kbp repeat and subsequently assembled. For Canu, the simulated dataset cannot be assembled contiguously for repeat differences below 3%, for FALCON [34] this threshold is 5%. The parameter differences between this benchmark and our simulated data, with 2 repeat copies versus our 100 repeat copies, 12% read error versus our 15% read error, and 3% copy differences versus the 1% or less we use, decrease the chances of Canu resolving our simulated data.

To adapt our simulated data sets for the requirements of genome assemblers, we modified our simulated datasets with 1% copy differences to model contiguous sequences (instead of just repeat sequence with flanking sequence). To this end, we double the flanking sequences to avoid possible confusion stemming from reads spanning unique sequence between reads, and concatenate all repeat copy sequences into a 7 mbp sequence. We then sample PacBio-typical reads from this sequence with an error rate of 15% and a coverage of 40 X.

We then proceeded to run Canu on the resulting dataset. As expected,

all three types of simulated datasets were assembled into roughly one hundred contigs, indicating an unsuccessful repeat resolution. Specifically, the distributed dataset results in 97 contigs, the equidistant dataset results in 106 contigs, and the tree-like dataset results in 122 contigs.

We also executed MARVEL on all three simulated datasets, similarly resulting in roughly one hundred disjointed contigs, see Figure 7.6. These results confirm Canu's earlier repeat resolution assessment and MARVEL's methodological limitations.

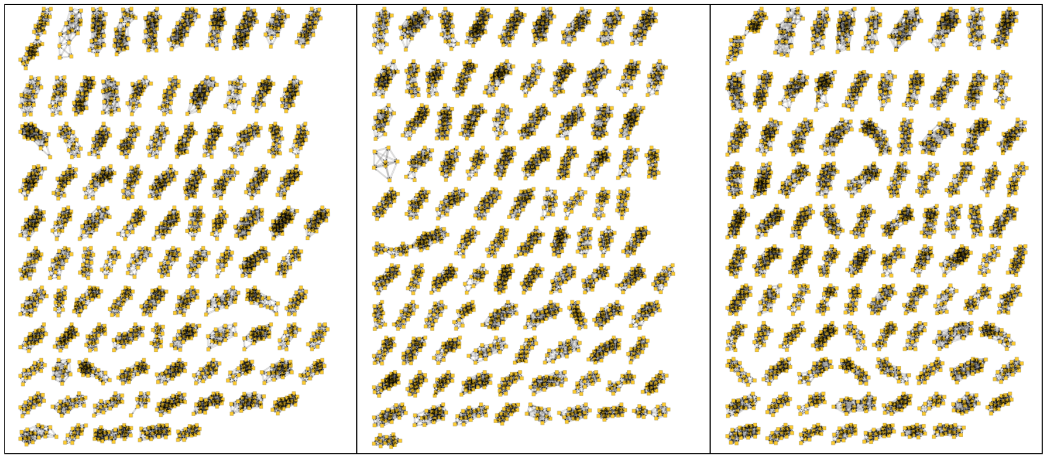

Figure 7.6: The MARVEL assembly graphs of our simulated data sets (distributed, equidistant, tree-like) show that MARVEL cannot resolve 30 kbp repeats with 1% copy differences. The distributed dataset results in 100 contigs, the equidistant dataset results in 101 contigs, and the tree-like dataset results in 101 contigs.

However, there exists one dedicated repeat resolution tool for PacBio data called split_dis. The split_dis tool [50] is part of the Daccord package [56]. The Daccord package contains a suite of tools for processing long reads, centered around the read correction program daccord. The processing necessary for executing split_dis involves computing local alignments with daligner [57], calling a corrected consensus version of each read using the daccord program, and computing quality values for the bases of each read. Split_dis then filters the local alignments for each read separately and retains only those that do not exhibit differences that are likely to be associated with copy differences. By finally selecting only those overlaps of the processed read, where the retained local alignments span almost all of the overlapping region, we generate a list of "true" overlaps for the read. In the following, we will refer to this entire repeat resolving pipeline as *Daccord*, while we refer to our pipeline as *RepeatResolver*.

According to its author, using Daccord is not feasible for repeat families

with substantially more than 10 copies (German Tischler, personal communication, 12.9.2018). Our results confirm this, as the runtime per read ranges from minutes for the transposon data sets with fewer than 10 copies, to hours for data sets with copy numbers between 10 and 20, and days for transposon data sets with more than 20 copies (see Table 7.1). For comparison, the entire RepeatResolver pipeline takes 2 hours for transposon data set 2, which has 25 repeat copies. This amounts to 0.1 minutes per read as opposed to 1 day and 19 hours required by Daccord. As the data sets contain between 257 and 7317 reads (see Table 7.1) this makes the application of Daccord infeasible for large repeat families. Accordingly, we assess performance only for transposon data sets with less than 30 copies.

Table 7.1: Properties of the transposon data sets

No	Average length	Copy number	Coverage	Ground truth coverage	Daccord runtime per read	Number of reads compared	Average comparison cluster size	Number of reads
0	7215 bp	37	37	23	None	None	None	1388
1	7380 bp	49	45	27	None	None	None	2240
2	4672 bp	25	45	27	1d19h	18	15.5	1297
3	6850 bp	44	51	35	None	None	None	2207
4	6093 bp	16	37	29	16h	29	20.4	605
5	7538 bp	34	46	26	None	None	None	1575
6	4479 bp	7	51	41	3m	289	16.1	359
8	5168 bp	9	54	41	1h12m	73	19.0	487
9	6371 bp	13	49	33	13h	18	14.2	646
10	5201 bp	89	48	30	None	None	None	4284
11	4762 bp	135	48	33	None	None	None	6480
12	4690 bp	157	46	33	None	None	None	7317
15	4361 bp	8	48	29	8m	239	20.5	391
16	7481 bp	12	35	18	3h49m	66	16.3	423
19	6128 bp	5	51	32	35m	164	20.4	257
20	5376 bp	20	41	30	14h	17	30.3	832
21	5048 bp	22	43	32	1d17h	12	62.8	961

Daccord computes a list of overlaps for each read, while RepeatResolver yields a clustering of reads. To compare the results, we transform the RepeatResolver output into Daccord-style output, by assigning to each read all the reads in the same cluster as overlaps.

For the comparison, we only consider reads for which ground truth information is available. We test three methods of overlap selection to achieve the highest possible resolution accuracy for Daccord. We 1.) select the 20

longest overlaps, 2.) select the 20 longest local alignments provided by Daccord, 3.) choose the 20 local alignments with the best alignment score among those local alignments that span more than 90% of the overlapping region. We compare the RepeatResolver result to the best result among these three methods for each data set.

Selecting substantially more than 20 overlaps reduces accuracy, as more than 30 reads with ground truth information are not always available for each repeat copy. Selecting less than 20 overlaps does not increase accuracy. The average number of reads per RepeatResolver cluster with ground truth information varies from 14 to 62 between data sets. They cluster close to the number of 20 chosen for the Daccord overlaps, see Table 7.1.

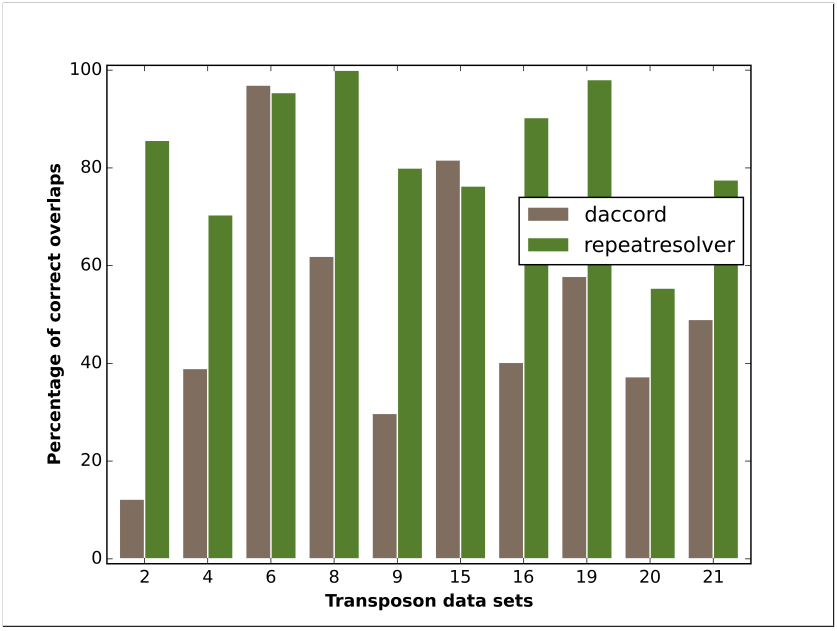

Figure 7.7: We compare the percentage of correct overlaps as provided by Daccord and RepeatResolver for all transposon data sets with fewer than 30 repeat copies.

In Figure 7.7 we show the percentage of overlapping reads that match the repeat copy of the underlying read for both pipelines and for all transposon data sets with fewer than 30 copies. Daccord shows excellent results, slightly outperforming RepeatResolver, in two out of ten data sets, proving the method sound. In the other 8 data sets, however, RepeatResolver outperforms Daccord. This leads to an average accuracy difference over all 10 data sets exceeding 30%, with an average accuracy of 82.9% for RepeatResolver and an average accuracy of 50.6% for Daccord.

7.5 Discussion

Overall our results indicate that, as long as sufficient signal is contained in the data, our novel algorithms are capable of resolving repeats with extremely high copy numbers. This even holds when the resolution has to proceed over many steps to span repeats that are several tens of thousands bases long. While the empirical transposon data shows that not all repeat sequences in genomes are likely to be resolvable, it also indicates that our method is capable of increasing the accuracy of genome assemblies.

Our comparison with the Daccord-pipeline shows that our method is superior in accuracy to a current state-of-the-art repeat resolving method for PacBio data, while at the same time remaining computationally feasible for repeat families with a significantly higher number of repeat copies.

Part IV

Tandem repeat complex

Chapter 8

Introduction to Neural Networks

8.1 History

Artificial neural networks (ANN) were originally conceived as computational models of the nervous system. They were based on the neuron doctrine and the all-or-none law of firing neurons. The neuron doctrine is a result of the neurological work of Santiago Ramón y Cajal [58], which states that brains are built of discrete individual cells, see Figure 8.1.

The all-or-none law of firing neurons [59] is the observation that neurons have no graded response. If the input to a neuron exceeds a certain threshold, the neuron emits an electro-chemical pulse, that can, in turn, excite other neurons. The strength of this pulse does not depend on the strength of the input. If the input does not reach the threshold the neuron remains inert.

McCulloch and Pitts [60] modelled these biological neurons as simple integrate-and-fire threshold step functions. Frank Rosenblatt then arranged the newly invented artificial neurons into the first influential ANN, the so-called perceptron [61] which, unlike biological brains, is arranged in layers without feedback information. The perceptron can be used to process an input vector in a feed-forward fashion. Mathematically, the perceptron is a nested function whose components switch between linear mappings and non-linear activation functions. The combination of a linear function and the accompanying activation function is called a layer.

8.2 Mathematical definition

The linear functions $L^l(x)$ are of the form $W^l x + b^l$ for each layer l, where $x, b^l \in \mathbb{R}^n$, are the input vector and the so-called bias vector respectively. The matrix $W^l \in \mathbb{R}^{n_l, n_{l-1}}$ is called the weight matrix. The non-linear functions φ_l can be chosen among a wide variety of possibilities which all at least approximate the step-function. A typical example is the sigmoid function $\varphi(x) := \frac{1}{1+e^{-x}}$. The original perceptron had three layers, an input layer, a hidden layer, and the output layer: $Px = \varphi_3 \circ L_3 \circ \varphi_2 \circ L_2 \circ \varphi_1 \circ L_1(x)$. This can be generalized to any number of layers, though. In this thesis, we will initially not require anything beyond a simple three-layer ANN. However, substantially more layers and increasingly complex constructions are currently in use. The perceptron can be visualised as a directed (and incidentally bipartite) graph, where the synapse weights denote the multiplicative strength of an edge and biases denote the additive strength of a node, see Figure 8.2.

8.3 Training

The purpose of an ANN is to process information. Each input vector is transformed into an output vector. The desired mapping has to be learned by adapting the weight matrices and the bias vectors. There is a variety of learning algorithms to choose from, starting from the neurologically plausible Hebbian learning [62], over simple yet computationally inefficient genetic algorithms [63, 64], to layer-wise learning of restricted Boltzmann machines [65]. Nevertheless, ANNs experienced their first bloom in the eighties, when an algorithm first introduced by Seppo Linnainmaa [66] seemed to promise computationally tractable learning of ANNs with several layers: The *backpropagation algorithm* [67]. The backpropagation algorithm is an efficient implementation of the stochastic gradient descent (SGD) optimisation method for ANNs. The fundamental idea of *gradient descent* is to follow the error gradient of a cost function, for example $C(x) = \frac{1}{2n}\|y(x) - P(x)\|^2$, towards a minimum, with x being the input vector and $y(x)$ the associated output target vector. The term *stochastic* means that the training data of correct mappings is not completely used for calculating each gradient. Instead, the full gradient is approximated by the gradient of a random sample, called a mini-batch. The training by gradient descent is the reason why today's neural networks approximate the step function with a differentiable function.

In the following, z^l stands for the input vector of layer l, while a^l is the output vector of layer l, the so-called activation vector. Here, $\odot$ denotes the elementwise multiplication or Hadamard product. The sigmoid φ function

was already defined above. We use δ as the partial derivative symbol, while ∇_a is the gradient in the direction a^L. The term $\delta^l = \frac{\partial C}{\partial z^l}$ can be interpreted as the error in layer l that is progressively backpropagated through the layers. Hence, the name backpropagation algorithm. The matrix multiplication with the transpose of W^{l+1} moves the error one layer back, while the element-wise multiplication with $\varphi'(z^l)$ can be seen as moving the error backward through the activation function φ. This allows to efficiently compute the error at each layer with a single pass through the network.

Algorithm 3 Backpropagation algorithm

procedure FORWARD PASS
    $a^1 \leftarrow \varphi(W^1 x + b^1)$
    for $l \in 2, 3...L$ **do**
        $z^l \leftarrow W^l a^{l-1} + b^l$
        $a^l \leftarrow \varphi(z^l)$
procedure BACKWARD PASS
    $\delta^L \leftarrow \nabla_a C \odot \varphi'(z^L)$
    for $l \in L, L-1, ...2$ **do**
        $\delta^l \leftarrow ((W^{l+1})^T \delta^{l+1}) \odot \varphi'(z^l)$
procedure WEIGHT UPDATE
    $W^l_{jk} -= a^{l-1}_k \delta^l_j = \frac{\partial C}{\partial W^l_{jk}}$
    $b^l_j -= \delta^l_j = \frac{\partial C}{\partial b^l_j}$

The initial excitement generated by this apparent algorithmic breakthrough quickly faded again when it turned out that, at the time, even the backpropagation algorithm could not train a neural network to solve tasks that went beyond small scale problems [68]. This disappointment in neural networks among other things, like the failure of Lisp machines and expert systems, led to the most severe of the so-called AI winters, enduring downturns in funding and public approval in the field of AI. It was only in the 2000s when due to a substantial increase in available computing power and an equally substantial increase in available training data, the neural network paradigm, rebranded as "Deep Learning", became popular again.

8.4 Current use

In the 2010s, systems based on deep artificial neural networks became the state of the art for several image recognition tasks. Most notably in the ImageNet Challenge[69], where in the following years neural network entries

improved the classification of pictures into one thousand categories surpassing the performance of humans specifically trained for the task. Speech recognition was another task that substantially profited from the introduction of neural networks [70] and finally reached a level that allowed practical application at an unprecedented scale. Text-to-speech synthesis [71], the complement to speech recognition, experienced analogous advancements in just the last two years. Numerous natural language tasks [72] are now based on the neural network paradigm and the game of go is now played on a superhuman level, a decade earlier than anybody expected [73].

A plethora of projects in science, technology, and especially the medical domain currently attempt to harness the modelling ability of deep neural networks. The architectures of the neural networks used are quickly diversifying. This process is sometimes driven by meta-learning that means by applying neural networks to the task of designing neural networks. In this thesis, we use relatively simple 3-layer neural networks that are separately trained and then combined into a more complex recursive de-noising function.

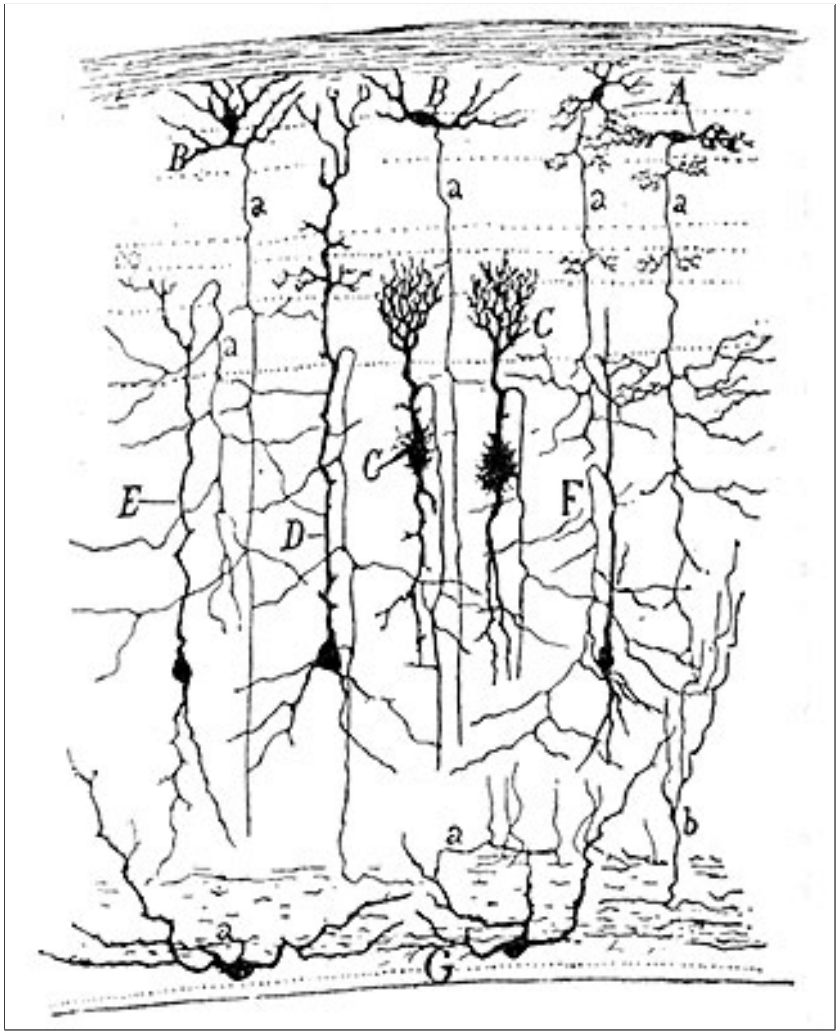

Figure 8.1: Drawing of a section through the optic tectum of a sparrow from "Structure of the nervous centres of the birds" by Santiago Ramon y Cajal, 1905.

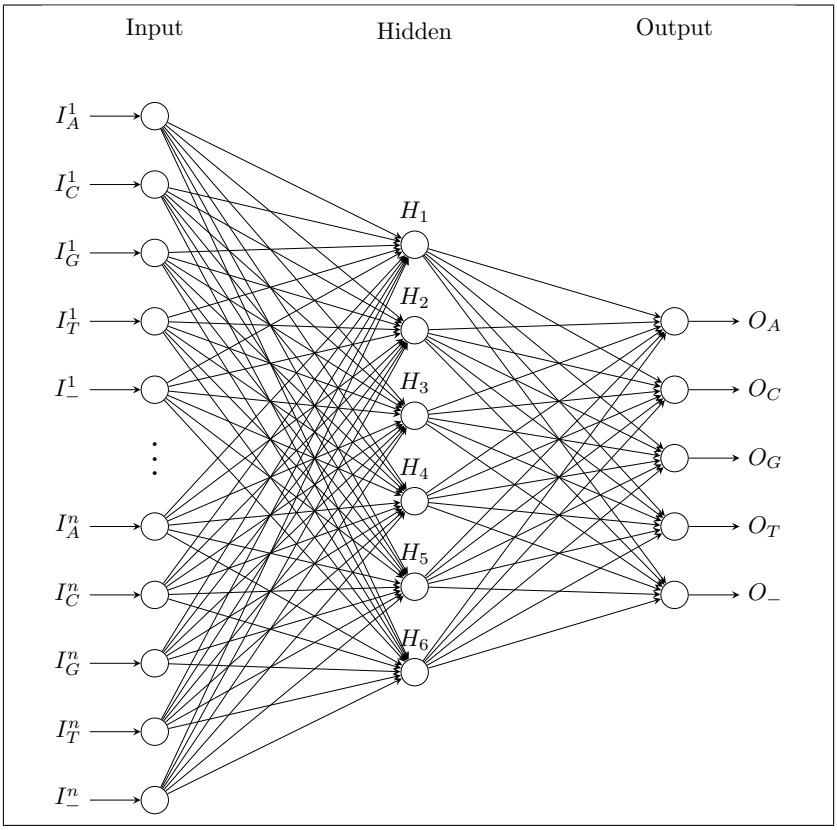

Figure 8.2: A neural network with an input layer, an output layer, and a single hidden layer.

Chapter 9

Correction of signatures

This chapter is based on a peer-reviewed publication:

Philipp Bongartz, Siegfried Schloissnig.
"Deep repeat resolution –
the assembly of the Drosophila Histone Complex"
Nucleic acids research. 2018 Nov 26.

Code and data available at
https://github.com/PhilippBongartz/DrosophilaHistoneComplex

Contributions: To the paper "Deep repeat resolution –the assembly of the Drosophila Histone Complex", Philipp Bongartz contributed algorithm design, implementation, testing and writing. Siegfried Schloissnig contributed problem selection and feedback.

9.1 Introduction

Drosophila melanogaster (see Figure 9.1) has been one of the most important model organisms for over a hundred years [74]. It has also played a crucial role as a proof of concept genome [75] for the technologies and algorithms that lead to the successful Human Genome Project [27], [26].

Despite the successful assembly of the Drosophila genome, the Drosophila Histone Complex has eluded assembly by conventional assembly methods [76]. It consists of over a hundred copies of a 5 kbp repeat sequence [77] that contains the coding information of the four core histones (H2A/B, H3,H4) as well as the linker histone H1. These proteins package the DNA into structural units by acting as spools around which the DNA double helix is wound, see Figure 9.2. The fast expression of histone proteins is essential

Figure 9.1: This image shows a small male *Drosophila melanogaster* fly. This is a photograph by Andre Karwath published under the CC ASA 2.5 Generic licence.

for DNA replication and cell division. Therefore, eukaryotes always contain multiple clustered copies of the histone coding sequence.

The Drosophila Histone Complex contains tandemly repeated [78], highly conserved coding sequences in an unusually high number of copies. It thus represents a particularly challenging instance of a structure that occurs in similar configurations in most eukaryotes, including humans. Therefore, it provides a reasonable basis for investigating the resolution of repeat clusters. The development of third generation sequencing technologies like the Pacific Bioscience single molecule real-time sequencing [49], [30], [32], and more recently, Oxford Nanopore Technologies nanopore sequencing [31], has revolutionized de novo genome assembly. With read lengths exceeding 10 kbp, the assembly of bacterial genomes can now be regarded as a solved problem [36]. In addition, for eukaryotic genomes the contiguity of de novo assemblies has increased by orders of magnitude, due to the possibility to span interspersed repeats with the support of unique flanking sequences. However, while genome assemblers like FALCON [34], MARVEL [35], and Canu [33] have solved the problem of arranging millions of long reads into long contiguous assemblies, tandemly arrayed repeats still generally remain unresolved. In particular, high quality Drosophila melanogaster long read assemblies have been created with each of these assemblers, but the Drosophila

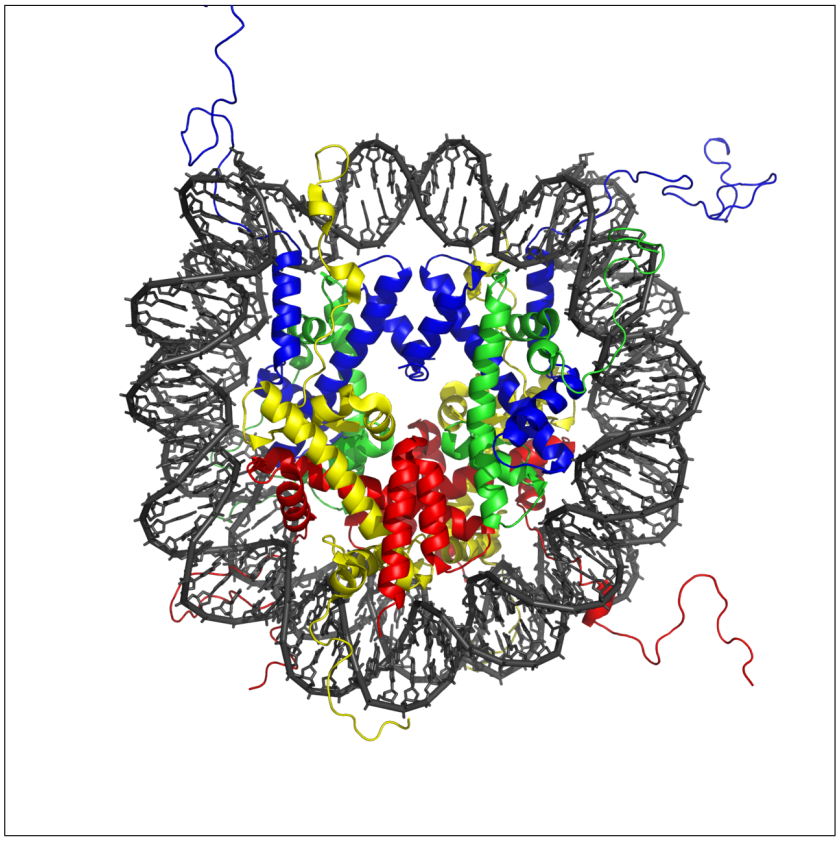

Figure 9.2: The crystal structure of the nucleosome core particle consisting of H2A, H2B, H3 and H4 core histones, and DNA. The view is from the top through the superhelical axis. This picture was published by Zephyris under the CC BY-SA 3.0 licence.

Histone Complex is not resolved in any of them. The resolution of tandemly arrayed repeats is hindered by the occurrence of further repeat copies as flanking sequences. This compels us to distinguish the slightly distinct copies of the tandem repeat and to order copy versions between the unique flanking sequences of the complex. The various copy versions of the repeated sequence have to be classified by identifying the large-scale variations (i.e., indels >100 bp) and single nucleotide variants (SNVs) that characterize each copy. Here, we introduce a novel correction heuristic based on artificial neural networks. Our heuristic decreases the error rate in extracted SNVs to such a degree that automatic assembly of the complex becomes feasible.

9.2 Data and preprocessing

Data

To test our correction heuristic, we use several repeat data sets: A data set of reads sampled from the histone complex, a simulated repeat data set and 17 transposon data sets (see Table 9.1 for an overview). Each data set is constructed around a repeat sequence template. The templates are consensus sequences of the repeat that is characteristic of the data set.

Data sets	Copies	Coverage	Length	Variations
Histone	107	30–90X	5kbp	185
Transposons	5–157	35–54X	4.3–7.5kbp	300
Simulated	100	50X	5kbp	300

Table 9.1: Properties of the data sets. The number of variations used for correction, is comparable in all data sets, as the histone correction uses additional neighbouring signatures.

For the histone data set, the template is the histone coding sequence [77]. It is used to extract around 5000 reads that contain instances of this sequence via mapping from a high quality PacBio data set sequenced from a subline of the ISO1 (y;cn,bw,sp) strain of Drosophila melanogaster [79]. We extract all reads that align to the template over 1 kbp with an alignment error below 30%. This data set has a coverage of $>90\,X$ and an average read length of >10 kbp.

To independently benchmark our correction heuristic, we extract reads containing transposable elements from the same sequencing run to build 17 different transposon data sets. The extraction is again done via mapping, with a minimal local alignment length of 1 kbp and a maximal alignment error of 30%. The transposon data sets (see Table 9.2) are based on templates taken from the canonical transposon sequence set [54] [https://github.com/cbergman/transposons/]. They contain Drosophila transposons of comparable length (>4 kbp) as the histone repeat. Additionally, we simulate a 5 kbp repeat family with 100 copies using a simulation tool we implemented. As template for the simulation, we use an empirical sequence that is randomly sampled from the E. coli reference. We then first create 100 identical copies of the template. These are subsequently perturbed via a pool of 300 random single nucleotide variants (SNV), each of which we assign to a random subset of the copies. Finally, the actual simulated reads are obtained from these perturbed copies superimposing the typical PacBio

No.	Av. length	Copies	Coverage	Ground truth coverage
0	7215 bp	37	37	23
1	7380 bp	49	45	27
2	4672 bp	25	45	27
3	6850 bp	44	51	35
4	6093 bp	16	37	29
5	7538 bp	34	46	26
6	4479 bp	7	51	41
8	5168 bp	9	54	41
9	6371 bp	13	49	33
10	5201 bp	89	48	30
11	4762 bp	135	48	33
12	4690 bp	157	46	33
15	4361 bp	8	48	29
16	7481 bp	12	35	18
19	6128 bp	5	51	32
20	5376 bp	20	41	30
21	5048 bp	22	43	32

Table 9.2: Properties of the transposon data sets. The numbering is according to the transposon sequence canonical set, missing numbers are due to transposons being below the length cut-off. Ground truth coverage is calculated on the basis of the reads that can be assigned to a cluster of flanking sequences.

error rate (11.5% insertions, 3.4% deletions, 1.4% mismatches). We use this simple version of a simulated repeat data set to verify our correction heuristic and the associated preprocessing steps. The test on simulated data further shows that the problems solved in the processing steps are not specific to the selected empirical test datasets of the chosen genome. In the following we provide an overview over the individual (pre-)processing steps.

Multiple sequence alignment

In the first preprocessing step, our objective is to subdivide each extracted histone read into instances of the repeat sequence. Some of these instances contain insertions, deletions, or duplications. If properly identified, insertions, deletions, and duplications allow for assigning these instances either to unique repeat copies or to small groups of repeat copies. All other instances of the repeat sequence that do not deviate in such a significant way from the histone template will have to be disambiguated by more sophisticated means.

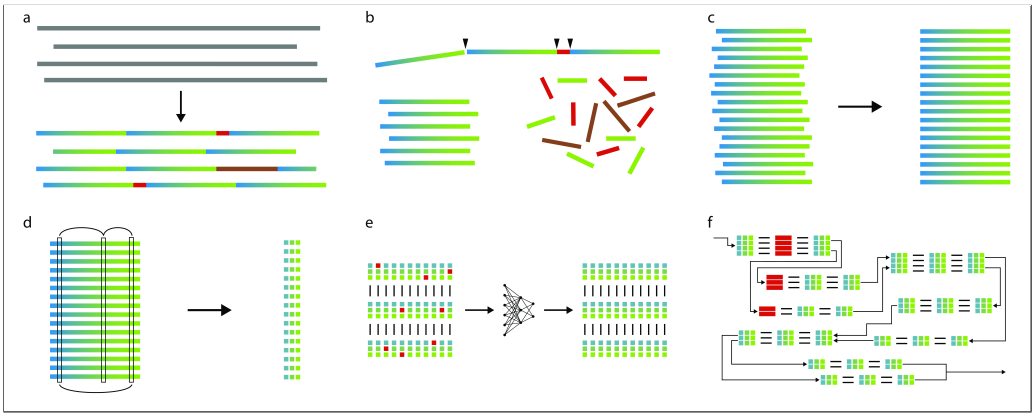

Figure 9.3: **a** shows how raw sequencing data is categorized as repeat or unique sequence using the mapping information of sub-sequences of the repeat template. In **b**, the reads are cut and the repeat sequences are arranged into a multiple sequence alignment. **c** shows the refinement of the multiple sequence alignment. In **d**, corrections between rows are detected and statistically significant bases are collected into signatures. **e** illustrates how the signatures are corrected via neural networks. In **f** finally, the signatures are clustered and the resulting assembly graph is traversed.

To that end, we map short (100-250 bp) substrings of the histone coding sequence to each read. The mapping information is then used to detect several insertions, deletions, or duplications between 100 bp and 8 kbp in length (see Figure 9.3a). These large-scale deviations from the template uniquely classify several distinct repeat copies. We then cut all histone reads into such instances of uniquely classified copies and into all other, non-deviating instances of the histone template [77] [80], see Figure 9.3b. Subsequently, the non-deviating instances of the histone template are arranged into a global multiple sequence alignment (MSA). The uniquely classified instances will be used later on for providing additional information to the correction algorithm and the assembly. The transposon reads are cut into unique flanking sequences and transposon sequences, analogously. The transposon sequences are also arranged into an MSA. The flanking sequences are clustered according to alignment scores (Levenshtein distance) to provide the ground truth for copy versions. For the simulated data, only repetitive sequences are generated. They can hence be immediately arranged into an MSA without any preprocessing. All initial MSAs are built incrementally by aligning the repetitive sequences to the respective repeat sequence template. These MSAs are then refined by realigning the sequences iteratively, minimizing the unweighted sum-of-pairs score [45], (see Figure 9.3c and Chapter 4.4). Both tools were

implemented from scratch in the C programming language and are available at https://github.com/PhilippBongartz/DrosophilaHistoneComplex.

Identifying discriminative columns

The refined MSAs contain sequences that are sampled from highly similar but not identical repeat copies. These sequences are now arranged in such a way that a difference between repeat copies becomes detectable as groups of different bases or alignment gaps within a column. As detailed in Chapter 5 we compute statistical significance scores for the base groups in the columns of our refined MSA. However, unlike in Chapter 6, we are not immediately interested in these base groups. Instead, we extract columns that allow us to discriminate between repeat copies. We call a column a *discriminative column* if the statistical significance of the intersection of at least one of its base groups with another base group is above the statistical significance cutoff.

The ubiquity of erroneous insertions (11% in PacBio data) means that discriminative insertions will share a column with a large number of false positives. This makes the detection of discriminative insertions difficult, while handling the noise in the detected discriminative insertions is even more challenging. For this reason, we restrict our detection to columns that contain a majority of bases as opposed to those containing mostly gaps. We choose a relatively high significance cutoff of negative log-probability $-log(CHG(G^i_{B_1}, G^j_{B_2})) > 15$, see Chapter 5.1, based on an empirical assessment via trial and error.

This gives us a set of 185 discriminative columns. In our correction heuristic, we are going to utilise the discriminative columns of neighbouring repeat sequences, whenever such a neighbouring repeat sequence exists. On average this amounts to approximately 300 discriminative columns. The transposon sequences are generally less highly conserved than the histone coding sequence. Thus, we restrict the number of selected discriminative columns in our transposon test sets to 300 to conduct an as fair as possible correction comparison to the histone data set. For the same reason, we also extract the top 300 columns from our simulated data set, which would, in the ideal case, be the entirety of the added SNVs.

The selected discriminative columns $D = \{d_1, d_2, d_3, ...d_n\}$ constitute the n distinguishing features, on the basis of which we disambiguate the instances of the repeat sequence. For each row i of the refined MSA M, we define a signature $S_{j<n}$ as a vector of entries $\{M_{id} | d \in D\}$, (see Figure 9.3d). For the histone data set, we additionally assign a unique signature to each large-scale variation which is likewise an element of $\{A, C, G, T, -\}^n$. This allows

us to model each read $R := [S^1, S^2, S^3, ...S^{N_R}]$ as a list of signatures, with N_R being the number of histone coding sequence copies in the read R.

9.3 Correction

Introduction

The native sequencing error rate is expected to carry over into our extracted signatures. Additionally, the high insertion/deletion rates in PacBio reads result in an unavoidable bias. This bias arises as those frequently inserted bases often provide better options for optimizing the alignment score than the correct base. Consequently, rare variations are more likely to be treated as an error by the MSA algorithm. Furthermore, the differences between copies are distributed in a hierarchical fashion due to an evolutionary process of copying and mutation [51]. This means that a large fraction of the copies is highly similar, if not identical. Complicating matters even more, not every read shows the average error rate. Instead, the error rate, especially the rate of insertions, varies strongly from read to read. For these reasons, successfully extracting distinguishing features from the reads does not automatically induce a correct disambiguation of repeat copies.

Note that the expectation that sequences with more similar features belong to the same repeat copy does not hold. The large number of copies, the high and varying error rate, the MSA bias, and the small number of differences between repeat copies lead to the situation that similarity measures between signatures are almost completely dominated by noise (see Figure 9.4a). Subreads sequenced from the same repeat copy are not characterized by having more similar features, although on average this is the case, but rather by sharing a characteristic subset of features which is a priori unknown. This misleads standard clustering algorithms or read overlapping approaches. Non-standard clustering approaches based on rare subsequences of signatures face substantial run time limitations, as the number of possible subsequences quickly becomes prohibitive with increasing subsequence length.

Neural networks

We develop a machine learning architecture that specifically exploits the structure described above to correct signatures to a point where standard approaches do become applicable. We use neural networks to utilise the underlying characteristic subsets of features. The task of these networks is to

predict the instance of a target feature within a signature, based on all other entries of the signature and of the two neighbouring signatures within the same read, if available. The two neighbouring signatures provide additional information that allows the neural networks to improve prediction. However, using even more bases from the same read would impede generalisation as signatures with such a high number of neighbours become rare.

The basic structure of the neural networks we use is a simple, fully connected network with one hidden layer, (for a comprehensive treatment of neural networks, see [81]). Every i-th base $B \in \{A, C, G, T, -\}$ in a signature $S \in \{A, C, G, T, -\}^n$ is encoded as a one-hot vector of length five $I^i = (I^i_A, I^i_C, I^i_G, I^i_T, I^i)$ with $I^i_B = \begin{cases} 1 & S_i = B \\ 0 & S_i \neq B \end{cases}$. The input vector I for a given signature consists of a concatenation of the vectors $I^{i<n}$ of all bases of the full signature or in the histone case of the signature and its two direct neighbours, but excluding the target base S_j. The excluded target base S_j, represented by the vector I^j, is then approximated by the output vector of the neural network, $O \in (0, 1)^5$ with $\sum_{B \in \{A,C,G,T,-\}} O_B = 1.0$. The output vector is calculated using a softmax function $\sigma(z_j) := \frac{e^{z_j}}{\sum_{k=1}^{5} e^{z_k}}$ and can be interpreted as a probability distribution over $\{A, C, G, T, -\}$.

For each feature, a distinct and randomly initialised network is trained using backpropagation with gradient descent [66]. For each given signature, it strives to predict the base at the chosen feature using the bases at all other features as input. Until a given accuracy is reached, the learning rate is adapted. L_2-regularisation [82] is used to aid generalisation. Dropout [83] can be used, but is discarded if learning stalls. If possible, we train until we reach a prediction accuracy of >97%. However, some networks stall substantially below this threshold. Each trained neural network can then be used to correct its target base in every signature by changing it to the most probable predicted base.

As illustrated in Figure 9.4b, the hidden layer enables the network to model the sub-signatures that characterize groups of signatures that share a certain base at the target feature. The idea is that this allows the network to create generalised predictions that are closer to the underlying truth than the actual data, without being misled by overall similarity or dissimilarity of the complete signatures. Consequently, simply using linear regression is bound to fail, while deploying more hidden layers would not improve the accuracy. This intuition behind the hidden layer can be verified by observing certain sub-signatures that fully excite neurons of the hidden layer in trained networks.

Converging corrector

While some discriminative SNVs occur in many copies and in combination with other SNVs, there also exist SNVs that only describe a single specific copy and are the only SNV that does so. Therefore, the signal that has to be recognized to accurately predict the different copy versions that lead to a certain base at a certain feature varies greatly. Picking up the weakest signal requires overfitting the network to the data. This means that the network learns to predict the bases of individual signatures by recognizing their specific pattern of erroneous bases. This initially impedes generalized prediction.

By training a separate network, $N_j : S_{i<n, i\neq j} \longrightarrow S_j$, for every feature $j < n$ that is part of the signatures, we create a signature to signature function by concatenation, $N := \odot_{j<n} N_j : S_{j<n} \longrightarrow S_{j<n}$. A fixpoint of this function is a signature in which all bases are consistent with each other as judged by the neural networks. Intuitively, every error free signature should be internally consistent and therefore constitute a fixpoint. This suggests that one should repeatedly apply the function to a signature until convergence. In the first pass over the data, the neural network might still recognize individual signatures. Subsequent corrections can only rely on the underlying pattern, since the characteristic signature errors have already been corrected in earlier passes. This yields a generalized, instead of an overfitted correction.

Regarded on a more abstract level, the function that outputs the number of bases in a signature that are equal to the predicted base can be seen as describing a consistency landscape of signatures. Fixpoints are maxima in this consistency landscape since every base is the predicted base. The repeated application of our concatenated networks to a signature until convergence can be seen as an ascent towards the nearest consistency maximum.

Results

To assess the results achieved by our correction heuristic, we use transposon reads from the same data set, for which the ground truth is provided by unique flanking sequences, as well as a simulated data set. For the histone data set the ground truth is given by the manual assembly described above.

To quantify the results of the presented correction heuristic, we use the following metrics that are calculated for those signatures that are assigned to a ground truth group. For both, the first pass of the correction, and the fully converged correction, each signature has a corresponding corrected signature, a ground truth consensus of uncorrected signatures (consensus),

and a ground truth consensus of corrected signatures (cor-consensus).

The "error" is the percentage of bases of the signatures that differ from the respective consensus. We report the percentage of bases of the cor-consensuses that differ from the consensuses as "collapsed variations". "Internal consistency" denotes the similarity of uncorrected signatures to the consensus and corrected signatures to the cor-consensus, respectively.

"Recall" is the percentage of true positives among the correct positives according to the consensus, or 100% - "error" as defined above, whereas "precision" is the percentage of true positives among the positives.

Data sets	Original error	Corrected error	Collapsed variations
Histone	8.22%	1.32%	0.42%
Transposons(median)	6.51%	1.78%	0.41%
Transposon(mean)	7.26%	2.01%	0.80%
Transposons(best)	5.26%	0.20%	0.00%
Simulated	8.89%	1.34%	1.29%

Table 9.3: This table presents the error rate reduction of the converging corrector. The error rate is the average percentage of the bases of a signature, that differ from the consensus of the uncorrected signatures in the ground truth group (i.e. manual assembly copy group) to which the signature belongs. "Original" denotes the uncorrected signatures, "corrected" the corrected signatures. "Collapsed variations" are variations were the majority of signatures of a ground truth group have been corrected towards the incorrect majority base.

Table 9.3 shows that we achieve significant error reduction in all data sets. The comparison with Table 9.4 shows that the converging corrector leads to a substantial accuracy improvement compared to the accuracy achieved in the first pass. In our target data set, the Drosophila histone complex, the error of 8.22% is reduced in the first pass to 4.64%. This is then further decreased to 1.32% in the converging corrector. Figure 9.5 illustrates how this error reduction translates into substantially improved overlap accuracy and "sharper" differences among signature groups. It is worth noting that reinserting the corrected SNVs into the reads would not substantially change the read error. Our correction heuristic solely intends to "sharpen" the extracted differences between repeat copies.

There is an intrinsic limit as to what de-noising algorithms can achieve. The presented heuristic is no exception. If the information necessary to correct a certain feature for a certain copy is not part of the data, the variation will be "collapsed", that is, transformed into the majority instance of that

Data sets	Original error	Corrected error	Collapsed variations
Histone	8.22%	4.64%	0.19%
Transposons(median)	6.51%	4.14%	0.05%
Transposon(mean)	7.26%	4.45%	0.12%
Transposons(best)	5.26%	2.04%	0.00%
Simulated	8.89%	2.49%	1.05%

Table 9.4: This table shows the error rate reduction of the first pass correction.This table shows the error rate reduction of the first pass correction. The error rate is the average percentage of the bases of a signature, that differ from the consensus of the uncorrected signatures in the ground truth group (i.e., manual assembly copy group) to which the signature belongs. "Original" denotes the uncorrected signatures, "corrected" the corrected signatures. "Collapsed variations" are variations were the majority of signatures of a ground truth group have been corrected towards the incorrect majority base.

particular feature. This collapse is unavoidable and more "sharply" distinguishes between signatures from different copies, in the sense that now all (or most) signatures from this particular repeat copy have the same base at the given position. This means that once the best possible distinction on the basis of the corrected signatures has been achieved, it might be useful or necessary to return to the original signatures for further refinement. A collapse also occurs if error-induced false variations are above the significant threshold. In this case the collapse is just a correction. Table 9.3 shows that the collapse of variations does not represent a substantial problem in our data sets.

The rarer bases at any given SNV position are subjected to a stronger bias by the MSA. The incentive to maximize the count of the most frequent base in every column is inseparable from the MSA algorithm itself. The high insertion rate of PacBio data leads to an abundance of alignment options and as a result the rarer base is often "shifted" out of the correct column. Therefore, rare bases initially show a significantly higher error rate.

Rare bases describe but a few copies of the repeat family. This means that they are more easily collapsed. Indeed, in 1 out of 17 transposon data sets the average recall of the rare bases is clearly worse after the first pass correction. This is due to a number of collapsed variations which decrease the average recall, with their recall of 0.00%. In the corrected version, this number grows to 5 out of 17. Table 9.5 shows that overall, even the rare bases considerably improve recall.

Rare bases also compete with a high number of false positives, which

Data sets	Original recall	First pass recall	Corrected recall
Histone	77.7%	83.1%	88.6%
Transposons(median)	85.4%	87.6%	90.7%
Transposon(mean)	85.7%	87.8%	86.0%
Transposons(best)	92.7%	95.5%	99.7%
Simulated	85.4%	88.1%	90.7%

Table 9.5: This table presents the recall of rare bases. The percentage of true positives among the correct positives. "Original" denotes the uncorrected signatures, "corrected" the corrected signatures, while "first pass" describes the signatures after a single application of the neural network corrector.

from an assembly point of view is more problematic than the collapse of variations. Our correction heuristic achieves improved precision for every transposon data set. Table 9.6 depicts the overall results for rare base precision improvements.

Data sets	Original precision	First pass precision	Corrected precision
Histone	69.0%	79.1%	93.1%
Transposons(median)	86.3%	87.0%	91.4%
Transposon(mean)	80.0%	81.7%	89.8%
Simulated	94.1%	99.0%	99.7%

Table 9.6: This table shows the precision of rare bases. The percentage of true positives among the positives. "Original" denotes the uncorrected signatures, "corrected" the corrected signatures, while "first pass" describes the signatures after a single application of the neural network corrector.

The collapse of variations may also lead to scenarios in which the average accuracy of corrected bases presents a misleading picture. If a minority base has been fully corrected for one group and collapsed for another group, the average accuracy would be the same as when both groups show an accuracy of 50%. In the former case however, the internal consistency of both groups is higher, although the overall accuracy is the same. Table 9.7 shows that while the correction improves the overall accuracy of rare bases, it improves the internal consistency to an even larger degree.

Data sets	Original precision	First pass precision	Corrected precision
Histone	91.7%	95.3%	98.8%
Histone rare	77.7%	83.4%	91.3%
Transposons(median)	93.4%	95.8%	99.2%
Transposon(mean)	92.7%	95.6%	98.8%
Transposon(best)	94.7%	98.1%	99.8%
Trans. rare(median)	85.4%	90.2%	96.8%
Trans. rare(mean)	85.7%	89.6%	95.4%
Trans. rare(best)	92.7%	95.5%	99.7%
Simulated rare	91.1%	98.5%	99.9%
Simulated	85.4%	95.8%	99.7%

Table 9.7: This table presents the internal consistency of groups over all data sets before and after correction. "Original" denotes the uncorrected signatures, "corrected" the corrected signatures, while "first pass" describes the signatures after a single application of the neural network corrector. In the "corrected" and "first pass" cases the internal consistency is the similarity of corrected signatures of a ground truth group (i.e., manual assembly copy group) to the consensus of the corrected group, instead of to the consensus of the uncorrected group. In the uncorrected case, internal consistency is just accuracy, that means 100% - error rate.

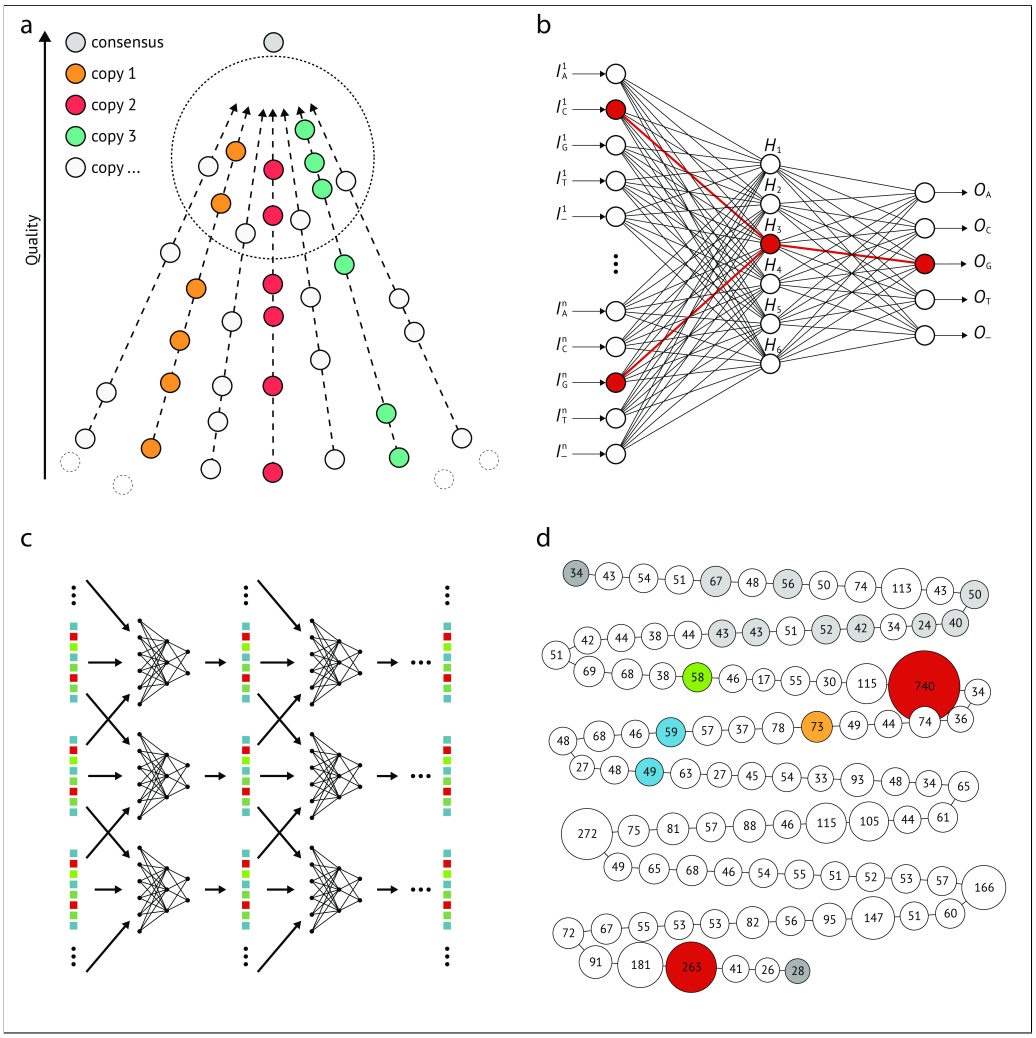

Figure 9.4: In **a**, we illustrate the fundamental problem: A hypothetical master consensus of all copy versions is more similar to the signatures than signatures that belong to the same copy are to each other. With that property, it acts like a vanishing point, signatures with low error rate all seem to be quite similar. The neural network depicted in **b** solves this problem because it is able to pick up on the sub-signatures shared by signatures from the same copy. In **c**, we show the unrolled converging corrector: The signatures are repeatedly corrected by the concatenated neural networks using them and both neighbouring signatures until the bases stop changing. **d** shows one assembly graph greedily traversed starting from one end of the complex. Node size and number give the size of each assembly group after mapping all clusters, even those that do not fit anywhere well. This overmapping allows us to double check on overrepresented groups and to catch the two collapsed parts of the complex marked in red. The other coloured nodes stand for large scale variations.

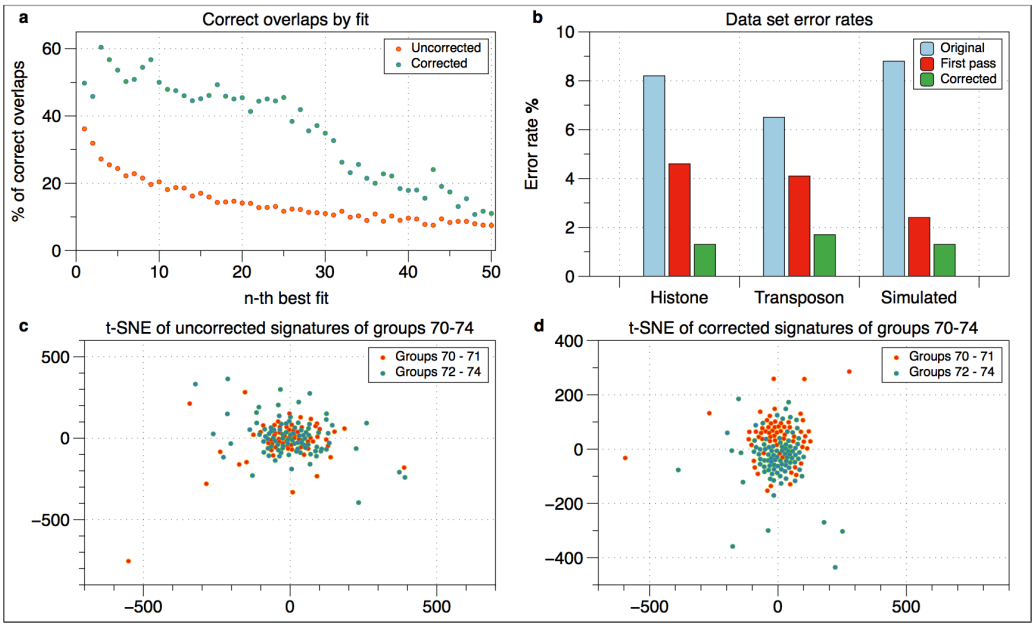

Figure 9.5: In **a**, we examine full signatures with ground truth information. For each signature, we calculate the likelihood that the n-th best overlapping signature belongs to the same ground truth copy group. This likelihood starts low and drops fast, whereas the corrected signatures have a significantly higher likelihood of correct overlaps, which stays stable for the 25 best overlaps. **b** shows the error reduction achieved by first pass correction and the converging corrector. In **c** and **d**, we use a t-SNE visualisation to show how the correction facilitates the separation of neighbouring groups of signatures.

Chapter 10

Assembly

This chapter is based on a peer-reviewed publication:

Philipp Bongartz, Siegfried Schloissnig.
"Deep repeat resolution –
the assembly of the Drosophila Histone Complex"
Nucleic acids research. 2018 Nov 26.

Code and data available at
https://github.com/PhilippBongartz/DrosophilaHistoneComplex

Contributions: To the paper "Deep repeat resolution –
the assembly of the Drosophila Histone Complex", Philipp Bongartz contributed algorithm design, implementation, testing and writing. Siegfried Schloissnig contributed problem selection and feedback.

10.1 Clustering

After reducing the error rate of the signatures, we can cluster the corrected signatures and traverse the assembly graph that is given by the resulting clusters. The clustering algorithm we use on the corrected signatures is a variation of the popular k-means clustering algorithm [84]. To reach the best possible assembly result, we use all available information. This includes the corrected signatures, the unique sections classified in our first preprocessing step and the results of the following two additional analyses.

There exists a shorter version (4.8 kbp) of the repeat sequence that has already been described in the literature [77]. It differs from the histone template by a large deletion, that we also detect and classify in our preprocessing analysis. We create a separate MSA for this 4.8 kbp repeat. Due to the sub-

stantially lower copy number compared to the whole complex, the detected SNVs allow us to divide these shorter sequences into 3 different copy versions.

Additionally, we extract indels of 3 to 30 bases by clustering sections of the rows of the original MSA. These indels are challenging to detect from the extracted features alone. Both, short versions and indels are added to the corrected signatures in the form of fake triple base features. Furthermore, we extend each signature to encompass the features of all neighbouring signatures within the same read.

As centroids, we choose all extended and corrected signatures whose coverage extends in both directions for at least c bases. The selected centroids are restricted to these c bases, to ensure that all centroids have the same coverage. The parameter c is empirically chosen to be 1.2 times the length of a signature. This results in an average coverage of each repeat copy by 7 centroids. Thereby it is highly unlikely that a repeat copy is missed. After this initialisation, all extended signatures S are distributed among the centroids C by the first best fit according to $D(C,S) = \sum_{i<n|S_i \neq C_i} 1$. In a second round, the consensuses of these initial clusters are used as centroids. Finally, all elements of clusters below a size cut-off are distributed among the remaining clusters.

10.2 Graph touring

Due to the high number of initial centroids, the clustering generally splits the signatures into more clusters than there could possibly be repeat copies. These clusters $c_i, c_j \in C$ naturally constitute the nodes of an assembly graph $G := (C, E_{0<n<3})$. Two nodes $c_i, c_j \in C$ are connected by a directed edge if there exists a read $R = [...S1k, ..., S^{k+d}]$ such that $S^k \in c_i$ and $S^{k+d} \in c_j$ with $d \in [1,2]$. For a given distance d, the set of these directed edges is denoted by $E_d(c_i, c_j)$.

Therefore, the problem we need to solve is to draw a layered graph in which clusters are linearly ordered into layers that can contain more than one cluster, such that the order of these layers respects the order of signatures in reads to the largest possible extent.

At each step, we choose the cluster whose best placement maximizes the scoring function $SF(c,l) := SF_1 + 2 \times SF_2 - SF_3 - 2 \times SF_4 - SF_5$, where

$$SF_1 := \sum_{E_1(v,c)|v \in L_{l-1}} 1 + \sum_{E_1(v,c)|v \in L_{l+1}} 1$$

$$SF_2 := \sum_{E_2(v,c)|v \in L_{l-2}} 1 + \sum_{E_2(v,c)|v \in L_{l+2}} 1$$

$$SF_3 := \sum_{E_1(v,c)|v \notin L_{l-1}} 1 + \sum_{E_1(v,c)|\notin L_{l+1}} 1$$

$$SF_4 := \sum_{E_2(vmc)|v \notin L_{l-2}} 1 + \sum_{E_2(v,c)|v \notin L_{l+2}} 1$$

$$SF_5 := \sum_{v \in L_l} D(v,c), D(c,v) := \sum_{i<n|C_c[i] \neq C_s[i]} 1$$

and C_c is the consensus of cluster c, L_l contains the cluster elements currently assigned to the l-th layer, and $E_d(v,c)$ is the set of directed edges defined above. This score rewards edges that are consistent with the already placed clusters and punishes edges that are inconsistent with the preceding cluster placements, as well as differences between the cluster consensuses within a layer.

On the resulting assembly, we then call a consensus sequence using the designated PacBio variant caller quiver [44]. This is expected to result in a Q40 sequence (99.99% correct) for the minimal coverage of our assembly, Q45 for 97% and Q55 for 82% of the complex (Qx is a quality score defined by $x = -\log(e)$ for the error rate e).

10.3 Validation

To validate our automated assembly and to assess the accuracy of the correction algorithm, we also created a hand-curated assembly of the uncorrected signatures. In the following, we describe several insights that make manual assembly possible. The expansion of the complex by unequal recombination [51] tends to create adjacent identical copies. This means that the instances of a given distinguishing feature are likely to be clustered within the complex. Such neighbour similarity yields overlapping approaches infeasible, mainly due to the difficult statistical assessment of the trade-off between long overlaps and "good" overlaps. However, with a simple statistical analysis we can utilise the neighbour similarity to considerably reduce the problem size.

For each instance $b \in \{A, C, G, T, -\}$ of a feature $v < n$, we define a clustering coefficient $C_c(v,b) := \frac{\sum_{S|S_v=b} 1}{\sum_{S|S_v \neq b} 1}$ for the $S \in R$ with $\sum_{S^{i<N_R}|S^i_v=b} 1 > 1$, where each read R is represented as a list of signatures $R := [S^1, S^2, S^3, ...S^{N_R}]$ and n is the number of features in a signature. The clustering coefficient is simply the number of occurrences of an instance divided by the number of non-occurrences of the instance over all reads in which it occurs at least twice.

Intuitively, this number captures in how many contiguous parts the sub-complex described by the instance b occurs within the complex, that is, how strongly the copies belonging to the sub-complex are clustered. The problem can now be broken down into smaller locally connected subclusters. This is done by selecting subsets of reads that are defined by the occurrence of an instance. We then manually analyse instances in order of decreasing clustering coefficients.

In each of these smaller problems we try to identify combinations of features that appear to describe a single repeat copy. Each of these combinations of features defines a fixed group of signatures in which it occurs. These fixed groups are connected if their signatures occur in the same reads. They are strongly and consistently connected if their signatures occur often and with a constant distance in the same reads. To each side a third of the signatures has a full signature neighbour, that is a signature with coverage for all features. Assuming a coverage of C and an independent assignment of signatures to two fixed groups of size C_1 and C_2, we can expect $\frac{C_1 \times C_2}{C \times 3^d}$ consistent connection between them if they accurately describe copies that have a distance d in the complex. We use this observation to validate independently fixed groups. We cannot always define unambiguous groups that are just one or two copies away from each other. These gaps have to be filled with long reads that can be anchored in validated fixed groups.

By this slow manual process, the whole complex can be assembled. This manual assembly constitutes the ground truth for the automated assembly described above.

10.4 Results

The greedy layered graph drawing algorithm can, in principle, be started with any cluster. Here, the flanking sequences and the unique large-scale variations constitute the obvious initial choices. Depending on the starting cluster, our automated assembly algorithm correctly orders up to 90 consecutive copies out of 113 (including flanking sequences and large-scale variations).

Starting from the left end of the complex and for most other unique large-scale variations as starting cluster, our automated assembly algorithm only fails at two histone complex locations. Around repeat copy 32, a combination of two copy versions occurs twice, one after another, and is not resolved by the clustering. Also at the right end of the complex, the copies are extremely similar and therefore remain unresolved. Between these locations and the flanking sequences, 35, 60, and 3 of the copies are correctly arranged. Figure

9.4d shows the assembly graph resulting from a graph traversal starting from the left end of the complex. The red nodes indicate misassemblies consisting of collapsed copies.

The combination of clustering and a greedy layered graph traversal places 90% of the signatures into a layer which contains a majority of signatures from the same copy according to the manual assembly, resulting in almost identical consensus signatures.

The final assembly of the histone complex contains 107 copies of the histone repeat sequence and stretches over 570 kbp. Two of these copies are shortened by the same long deletion, and two are extended by a dublication of the H2A gene. Finally, another two copies have insertions of different length. The known 4.8 kbp repeat version [77] occurs 9 times loosely clustered at the beginning of the complex.

The preprocessing showed 11 reads with a divergent arrangement of two large-scale variations demonstrating the presence of a second haplotype represented by 10-15% of the data, despite the highly inbred strain. These reads have been excluded from all our analyses. Beyond the $3'$ end of the complex, we found an assortment of dysfunctional histone copies. The mechanism driving this local accumulation of degenerate copies is yet unknown. No further copies outside the complex were found.

10.5 Discussion

We want to emphasize that assembly via clustering, including the very specific analyses it entails, does not represent a general repeat cluster assembly algorithm. Instead, it illustrates the type of analysis that is feasible via our correction heuristic.

We expect other repeat complexes to exhibit their own idiosyncrasies. It is thus unlikely that our assembly approach will be directly applicable. For instance, all histone reads were oriented according to the template upon extraction by mapping. This simplifies downstream processing, but is only possible because there is no strand reversal in the complex. However, strand reversal does occur, for example, in the Drosophila rRNA-complex. The rRNA-complex also contains several distinct repeat sequences which would require further non-trivial adaptations of our clustering algorithm.

While the clustering and graph traversal resolves large stretches of the complex correctly, an important question is how to detect possible misassemblies. Figure 1d shows the assembly graph with misassemblies consisting of collapsed copies indicated by the red nodes. We detect these misassemblies by mapping the clusters that could not be reliably placed onto the assembly

which results in coverage anomalies (significantly more than the average 125 signatures per graph layer) in collapsed assembly groups. In Figure 1d the coverage is indicated by both, the labels and the node diameters.

The first misassembly cuts out a large-scale variation by collapsing similar copies at either side of the variation. Starting from this large-scale variation we obtain an assembly which correctly orders the first 90 copies. For the second misassembly, this trick does not work because the collapse already occurs during the clustering phase. Here, we have to resolve the ambiguous part on the basis of a few very long reads.

To our knowledge, this is the first approach that is capable of dealing with the intrinsic complexity of tandem repeat resolution. We expect a wide applicability of the presented methods for the resolution of tandem repeats in genome assembly, but also for the resolution of non-tandem repeat clusters and long transposable elements.

Chapter 11

Conclusion and Outlook

In this thesis, we presented several novel ideas to improve long-read genome assemblies. Specifically, we introduced algorithmic solutions for two important problems: The resolution of interspersed repeats and the assembly of tandem repeat complexes.

We implemented several tools for this purpose. All of them are freely available at https://github.com/PhilippBongartz. The tool **LargeScaleVars** computes statistics about the occurrence of sections of a template in a file containing reads. **ReadCutter** divides reads according to the occurrence of a repeat template. **InitialAligner** computes an initial MSA and **PW_ReAligner**, see Chapter 4.4, optimises the unit score of pairwise alignments for a given initial MSA. **Correlation**, see Chapter 5.1, then computes the statistical significance score for each base group in each column of the optimised MSA.

These pre-processing tools allowed us to base our repeat resolution algorithms on sets of extracted differences between repeat copies. In the case of our interspersed repeat resolution tool, we refined these differences combinatorically until we could subdivide the reads of our data set hierarchically. This subdivision was then complemented with a clustering approach to create clusters of reads that correspond to the underlying repeat copy groups. These clusters were then used to resolve repeat families with dozens of copies spanning over tens of thousands of bases.

For assembling tandem repeat complexes, we introduced a more powerful refinement heuristic based on neural networks that additionally utilizes the information in neighbouring repeats. Using the de-noised repeat differences from the neural network then facilitated clustering and graph drawing algorithms that can actually assemble almost the entire Drosophila Histone Complex.

We benchmarked our interspersed repeat resolving tool against its only

current competitor and found that it performs substantially better in terms of runtime and accuracy. The assembly of the Drosophila Histone Complex had so far eluded a large community of drosophila researchers and was missing in the six genome releases published so far by the Berkeley Drosophila Genome Project. This complex could also not be disentangled by existing long-read assemblers, such as Canu, MARVEL, and FALCON. We managed to close this gap in the drosophila melanogaster draft genome. We are confident that the methods presented here are suitable to also close the other substantial gap in this important genome assembly, the rRNA complex.

We hope that the methods presented here will enable the assembly of full chromosome arms with long-reads. The most promising direction for future research that will also improve the applicability of our methods is to automate the classification of repetitive and unique sequence in a repeat data set. This classification constitutes a necessary first step for analysing repeat sequences and currently requires a substantial amount of manual work.

There are several potentially promising approaches for automating this classification step. It might be sufficient to conduct a careful analysis of local alignments between reads sampled from a repetitive region or of k-mer distributions to arrange all repeat sections containing similar sequences into MSAs. A possibly more powerful approach combines classical alignment algorithms with machine learning ideas. A pool of weighted consensus sequences compete to represent sections of each read. If a consensus sequence best represents a section of a read, this section becomes part of its consensus, which constitutes a learning mechanism. After the learning, for each consensus sequence one can create its corresponding MSA for the read sections it represents.

Another idea for future work is to combine all neural networks that now independently correct single variations into a single neural network that corrects all variations. This allows to reuse internal representations of typical sub-signatures of repeat copies and will be computationally more efficient. It requires masking parts of the error gradient during learning to conceal the respective input variation for each output variation. We call this setup a *self-blind autoencoder*.

It might also be possible to improve clustering results by inferring the centroids from the co-occurrence of variations. The idea is that there is a set of centroids that optimally explain the co-occurrence of variations in our signatures. This set of centroids can be computed by solving a set of linear equations. This is a mathematically elegant way to choose centroids for the k-means clustering step. However, it is possible that the error rate in our data induces too much noise for this method to work.

The most straight-forward direction of future research is to work on the

assembly of microsatellite regions. A solution to this problem is necessary to enable "perfect" genome assemblies. A promising approach consists of detecting pairs of k-mers of a certain frequency that co-vary in a fixed distance to each other. It is likely that this method will become applicable in the near future when even longer reads are available.

Bibliography

[1] M. Cobb, "Heredity before genetics: a history," *Nature Reviews Genetics*, vol. 7, no. 12, p. 953, 2006.

[2] P. L. M. de Maupertuis, *Vénus physique*. 1745.

[3] G. Mendel, "Versuche über pflanzenhybriden," *Verhandlungen des naturforschenden Vereines in Brunn 4: 3*, vol. 44, 1866.

[4] F. Galton, *Hereditary genius: An inquiry into its laws and consequences*, vol. 27. Macmillan, 1869.

[5] C. Darwin, *The variation of animals and plants under domestication*, vol. 2. O. Judd, 1868.

[6] H. Winkler *et al.*, "Verbreitung und ursache der parthenogenesis im pflanzen-und tierreiche," 1920.

[7] O. T. Avery, C. M. MacLeod, and M. McCarty, "Studies on the chemical nature of the substance inducing transformation of pneumococcal types: induction of transformation by a desoxyribonucleic acid fraction isolated from pneumococcus type iii," *Journal of experimental medicine*, vol. 79, no. 2, pp. 137–158, 1944.

[8] F. Griffith, "The significance of pneumococcal types," *Epidemiology & Infection*, vol. 27, no. 2, pp. 113–159, 1928.

[9] J. D. Watson, F. H. Crick, *et al.*, "Molecular structure of nucleic acids," *Nature*, vol. 171, no. 4356, pp. 737–738, 1953.

[10] H. G. Khorana, "Nucleic acid synthesis in the study of the genetic code," *Nobel Lectures: Physiology or Medicine (1963–1970)*, pp. 341–369, 1968.

[11] W. M. Jou, G. Haegeman, M. Ysebaert, and W. Fiers, "Nucleotide sequence of the gene coding for the bacteriophage ms2 coat protein," *Nature*, vol. 237, no. 5350, p. 82, 1972.

[12] F. Sanger and A. R. Coulson, "A rapid method for determining sequences in dna by primed synthesis with dna polymerase," *Journal of molecular biology*, vol. 94, no. 3, pp. 441–448, 1975.

[13] R. Staden, "A strategy of dna sequencing employing computer programs," *Nucleic acids research*, vol. 6, no. 7, pp. 2601–2610, 1979.

[14] R. C. Gardner, A. J. Howarth, P. Hahn, M. Brown-Luedi, R. J. Shepherd, and J. Messing, "The complete nucleotide sequence of an infectious clone of cauliflower mosaic virus by m13mp7 shotgun sequencing," *Nucleic acids research*, vol. 9, no. 12, pp. 2871–2888, 1981.

[15] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of molecular biology*, vol. 48, no. 3, pp. 443–453, 1970.

[16] T. F. Smith and M. S. Waterman, "Comparison of biosequences," *Advances in applied mathematics*, vol. 2, no. 4, pp. 482–489, 1981.

[17] O. Gotoh, "An improved algorithm for matching biological sequences," *Journal of molecular biology*, vol. 162, no. 3, pp. 705–708, 1982.

[18] S. F. Altschul and B. W. Erickson, "Optimal sequence alignment using affine gap costs," *Bulletin of mathematical biology*, vol. 48, no. 5-6, pp. 603–616, 1986.

[19] E. W. Myers and W. Miller, "Optimal alignments in linear space," *Bioinformatics*, vol. 4, no. 1, pp. 11–17, 1988.

[20] G. Myers, "A fast bit-vector algorithm for approximate string matching based on dynamic programming," *Journal of the ACM (JACM)*, vol. 46, no. 3, pp. 395–415, 1999.

[21] E. S. Lander and M. S. Waterman, "Genomic mapping by fingerprinting random clones: a mathematical analysis," *Genomics*, vol. 2, no. 3, pp. 231–239, 1988.

[22] K.-J. Räihä and E. Ukkonen, "The shortest common supersequence problem over binary alphabet is np-complete," *Theoretical Computer Science*, vol. 16, no. 2, pp. 187–198, 1981.

[23] J. Tarhio and E. Ukkonen, "A greedy approximation algorithm for constructing shortest common superstrings," *Theor. Comput. Sci.*, vol. 57, no. 1, pp. 131–145, 1988.

[24] J. C. Roach, C. Boysen, K. Wang, and L. Hood, "Pairwise end sequencing: a unified approach to genomic mapping and sequencing," *Genomics*, vol. 26, no. 2, pp. 345–353, 1995.

[25] J. D. Kececioglu and E. W. Myers, "Combinatorial algorithms for dna sequence assembly," *Algorithmica*, vol. 13, no. 1-2, p. 7, 1995.

[26] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt, *et al.*, "The sequence of the human genome," *science*, vol. 291, no. 5507, pp. 1304–1351, 2001.

[27] I. H. G. S. Consortium *et al.*, "Initial sequencing and analysis of the human genome," *Nature*, vol. 409, no. 6822, p. 860, 2001.

[28] P. E. Compeau, P. A. Pevzner, and G. Tesler, "Why are de bruijn graphs useful for genome assembly?," *Nature biotechnology*, vol. 29, no. 11, p. 987, 2011.

[29] J. Dekker, K. Rippe, M. Dekker, and N. Kleckner, "Capturing chromosome conformation," *science*, vol. 295, no. 5558, pp. 1306–1311, 2002.

[30] J. Eid, A. Fehr, J. Gray, K. Luong, J. Lyle, G. Otto, P. Peluso, D. Rank, P. Baybayan, B. Bettman, *et al.*, "Real-time dna sequencing from single polymerase molecules," *Science*, 2008.

[31] S. Howorka, S. Cheley, and H. Bayley, "Sequence-specific detection of individual dna strands using engineered nanopores," *Nature biotechnology*, vol. 19, no. 7, p. 636, 2001.

[32] M. Foquet, K. T. Samiee, X. Kong, B. P. Chauduri, P. M. Lundquist, S. W. Turner, J. Freudenthal, and D. B. Roitman, "Improved fabrication of zero-mode waveguides for single-molecule detection," *Journal of Applied Physics*, vol. 103, no. 3, p. 034301, 2008.

[33] S. Koren, B. P. Walenz, K. Berlin, J. R. Miller, N. H. Bergman, and A. M. Phillippy, "Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation," *Genome research*, pp. gr–215087, 2017.

[34] C.-S. Chin, P. Peluso, F. J. Sedlazeck, M. Nattestad, G. T. Concepcion, A. Clum, C. Dunn, R. O'Malley, R. Figueroa-Balderas, A. Morales-Cruz, *et al.*, "Phased diploid genome assembly with single-molecule real-time sequencing," *Nature methods*, vol. 13, no. 12, p. 1050, 2016.

[35] S. Nowoshilow, S. Schloissnig, J.-F. Fei, A. Dahl, A. W. Pang, M. Pippel, S. Winkler, A. R. Hastie, G. Young, J. G. Roscito, *et al.*, "The axolotl genome and the evolution of key tissue formation regulators," *Nature*, vol. 554, no. 7690, p. 50, 2018.

[36] S. Koren and A. M. Phillippy, "One chromosome, one contig: complete microbial genomes from long-read sequencing and assembly," *Current opinion in microbiology*, vol. 23, pp. 110–120, 2015.

[37] M. A. Grohme, S. Schloissnig, A. Rozanski, M. Pippel, G. R. Young, S. Winkler, H. Brandl, I. Henry, A. Dahl, S. Powell, *et al.*, "The genome of schmidtea mediterranea and the evolution of core cellular mechanisms," *Nature*, vol. 554, no. 7690, p. 56, 2018.

[38] B. L. Cantarel, I. Korf, S. M. Robb, G. Parra, E. Ross, B. Moore, C. Holt, A. S. Alvarado, and M. Yandell, "Maker: an easy-to-use annotation pipeline designed for emerging model organism genomes," *Genome research*, vol. 18, no. 1, pp. 188–196, 2008.

[39] B. McClintock, "The origin and behavior of mutable loci in maize," *Proceedings of the National Academy of Sciences*, vol. 36, no. 6, pp. 344–355, 1950.

[40] J. L. Thorne, H. Kishino, and J. Felsenstein, "An evolutionary model for maximum likelihood alignment of dna sequences," *Journal of Molecular Evolution*, vol. 33, no. 2, pp. 114–124, 1991.

[41] A. Löytynoja and N. Goldman, "Phylogeny-aware gap placement prevents errors in sequence alignment and evolutionary analysis," *Science*, vol. 320, no. 5883, pp. 1632–1635, 2008.

[42] R. Fleissner, D. Metzler, and A. Von Haeseler, "Simultaneous statistical multiple alignment and phylogeny reconstruction," *Systematic biology*, vol. 54, no. 4, pp. 548–561, 2005.

[43] E. L. Anson and E. W. Myers, "Realigner: a program for refining dna sequence multi-alignments," *Journal of Computational Biology*, vol. 4, no. 3, pp. 369–383, 1997.

[44] C.-S. Chin, D. H. Alexander, P. Marks, A. A. Klammer, J. Drake, C. Heiner, A. Clum, A. Copeland, J. Huddleston, E. E. Eichler, *et al.*, "Nonhybrid, finished microbial genome assemblies from long-read smrt sequencing data," *Nature methods*, vol. 10, no. 6, p. 563, 2013.

[45] D. Gusfield, *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge university press, 1997.

[46] J. Kececioglu and J. Ju, "Separating repeats in dna sequence assembly," in *Proceedings of the fifth annual international conference on Computational biology*, pp. 176–183, ACM, 2001.

[47] M. T. Tammi, E. Arner, T. Britton, and B. Andersson, "Separation of nearly identical repeats in shotgun assemblies using defined nucleotide positions, dnps," *Bioinformatics*, vol. 18, no. 3, pp. 379–388, 2002.

[48] M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, P. Alken, M. Booth, and F. Rossi, "Gnu scientific library," *Network Theory Ltd*, vol. 3, 2002.

[49] A. Rhoads and K. F. Au, "Pacbio sequencing and its applications," *Genomics, proteomics & bioinformatics*, vol. 13, no. 5, pp. 278–289, 2015.

[50] G. Tischler, "Haplotype and repeat separation in long reads," *bioRxiv*, p. 145474, 2017.

[51] A. B. Reams and J. R. Roth, "Mechanisms of gene duplication and amplification," *Cold Spring Harbor perspectives in biology*, vol. 7, no. 2, p. a016592, 2015.

[52] M. P. Calos and J. H. Miller, "Transposable elements," *Cell*, vol. 20, no. 3, pp. 579–595, 1980.

[53] R. H. Miller D.E., C.B. Smith and C. Bergman, "Pacbio whole genome shotgun sequences for the d. melanogaster reference strain.," 2013.

[54] H. Attrill, K. Falls, J. L. Goodman, G. H. Millburn, G. Antonazzo, A. J. Rey, S. J. Marygold, and F. Consortium, "Flybase: establishing a gene group resource for drosophila melanogaster," *Nucleic acids research*, vol. 44, no. D1, pp. D786–D792, 2015.

[55] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, pp. 707–710, 1966.

[56] G. Tischler and E. W. Myers, "Non hybrid long read consensus using local de bruijn graph assembly," *bioRxiv*, p. 106252, 2017.

[57] G. Myers, "Efficient local alignment discovery amongst noisy long reads," in *International Workshop on Algorithms in Bioinformatics*, pp. 52–67, Springer, 2014.

[58] S. R. y Cajal, *Estructura de los centros nerviosos de las aves*. 1888.

[59] E. D. Adrian, "On the conduction of subnormal disturbances in normal nerve," *The Journal of physiology*, vol. 45, no. 5, pp. 389–412, 1912.

[60] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[61] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[62] D. O. Hebb, "The organization of behavior: A neurophysiological approach," 1949.

[63] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary computation*, vol. 10, no. 2, pp. 99–127, 2002.

[64] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution strategies as a scalable alternative to reinforcement learning," *arXiv preprint arXiv:1703.03864*, 2017.

[65] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[66] S. Linnainmaa, "The representation of the cumulative rounding error of an algorithm as a taylor expansion of the local rounding errors," *Master's Thesis (in Finnish), Univ. Helsinki*, pp. 6–7, 1970.

[67] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, p. 533, 1986.

[68] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[69] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[70] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[71] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio.," in *SSW*, p. 125, 2016.

[72] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[73] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.

[74] T. H. Morgan, "An attempt to analyze the constitution of the chromosomes on the basis of sex-limited inheritance in drosophila," *Journal of Experimental Zoology*, vol. 11, no. 4, pp. 365–413, 1911.

[75] E. W. Myers, G. G. Sutton, A. L. Delcher, I. M. Dew, D. P. Fasulo, M. J. Flanigan, S. A. Kravitz, C. M. Mobarry, K. H. Reinert, K. A. Remington, *et al.*, "A whole-genome assembly of drosophila," *Science*, vol. 287, no. 5461, pp. 2196–2204, 2000.

[76] R. A. Hoskins, J. W. Carlson, K. H. Wan, S. Park, I. Mendez, S. E. Galle, B. W. Booth, B. D. Pfeiffer, R. A. George, R. Svirskas, *et al.*, "The release 6 reference sequence of the drosophila melanogaster genome," *Genome research*, pp. gr–185579, 2015.

[77] Y. Matsuo and T. Yamazaki, "Nucleotide variation and divergence in the histone multigene family in drosophila melanogaster.," *Genetics*, vol. 122, no. 1, pp. 87–97, 1989.

[78] R. Lifton, M. Goldberg, R. Karp, and D. Hogness, "The organization of the histone genes in drosophila melanogaster: functional and evolutionary implications," in *Cold Spring Harbor symposia on quantitative*

biology, vol. 42, pp. 1047–1051, Cold Spring Harbor Laboratory Press, 1978.

[79] K. E. Kim, P. Peluso, P. Babayan, P. J. Yeadon, C. Yu, W. W. Fisher, C.-S. Chin, N. A. Rapicavoli, D. R. Rank, J. Li, *et al.*, "Long-read, whole-genome shotgun sequence data for five model organisms," *Scientific data*, vol. 1, p. 140045, 2014.

[80] L. Y. Geer, A. Marchler-Bauer, R. C. Geer, L. Han, J. He, S. He, C. Liu, W. Shi, and S. H. Bryant, "The ncbi biosystems database," *Nucleic acids research*, vol. 38, no. suppl_1, pp. D492–D496, 2009.

[81] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015.

[82] A. Krogh and J. A. Hertz, "A simple weight decay can improve generalization," in *Advances in neural information processing systems*, pp. 950–957, 1992.

[83] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[84] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, Oakland, CA, USA, 1967.