TOWARDS HANDLING REPEATS IN GENOME ASSEMBLY

NARMADA SAMBATURU

(B.E., VTU, India, 2009)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2015

Towards Handling Repeats in Genome Assembly

Narmada Sambaturu

January 2, 2015

Declaration

I hereby declare that this is my original work and has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in this thesis. This thesis has also not been submitted for any degree in any university previously.

Name: Narmada Sambaturu

Signed: S. Narmada

Date: 02/JAN/2015

To my parents.

Acknowledgement

First and foremost, I would like to thank my supervisor Prof. Wing-Kin Sung for all his help and support. His keen insight and sound knowledge of fundamentals are a constant source of inspiration to me. I appreciate his patience in explaining the biological reasons behind many of the biases that occur in the data we process on a daily basis in computational biology. His emphasis on understanding this has helped me with the current project, and I am sure it will hold me in good stead in the future as well. I am deeply thankful for his being available for questions and feedback at all hours, even during his absence from the country. Prof. Ken, I thank you with all my heart for everything you have done for me.

I would also like to thank the team from the Genome Institute of Singapore and the Institute for Infocomm Research - Eleanor Wong Huijun, Nguyen Hai An and Dr. Chua, Hon Nian Kenny, for handling the wet-lab aspects of this work. This project would not have been possible without their help in preparing the libraries, validating the results and in helping me understand the mechanism of Nextera cutting. I especially thank Eleanor and An for their patient explanations and co-operation, and Kenny for his support, encouragement and insightful discussions. I also thank Yang Peng from the Institute for Infocomm Research for his help with the local assembly aspect of this work.

My gratitude goes also to my family, for helping me through all the turmoil of being uprooted from my home for the first time and visiting a new country with new people and a new subject. I thank my father for the exciting discussions I could have with him regarding my work. I thank my mother for being my bedrock,

and for all the love and care I can feel from her even when we are far apart. I thank my sister for always being there for me, with help and advice from her own experiences, letting me know I am not alone.

A peer one can bounce ideas off of is invaluable in the life of any student. For me, this person has been my best friend Sujoy. His clear logic and sound reasoning have often helped me clarify my thoughts, allowing the solution to surface through the chaos. He has been the one cheering me on every day, making me smile through the toughest of times. He was also the one who helped me patiently through all the formatting issues with this thesis that I could not handle. Thank you.

I also thank my lab-mates for the active discussions we have had about various topics. Their stimulating conversation helped brighten the day. I thank my roommates and friends, who have made it possible for me to feel at home in a new place. Last but not the least, I thank the School of Computing and the National University of Singapore for the opportunities provided.

Contents

Summary

Repeat regions, i.e patterns of nucleotides that occur in multiple locations on the genome, have been shown to play a role in human-pathogen interactions [6]. Studying repeats could help open up new avenues for treatment. However, the amount of pathogenic material that can be extracted from a patient is limited. Given the need for a fast diagnosis, waiting for the bacteria to grow and multiply in the lab is not a viable option. Thus there is a need for a genomics pipeline which can work with small quantities of cells, work fast, and handle repeat regions.

In this project, we develop an algorithm to link the regions flanking a repeat given a library prepared with only picogram quantities of DNA. The algorithm exploits a 9bp overlap between adjacent fragments caused by the library preparation technique (Nextera). The algorithm was tested with an E.coli K-12 library prepared with 0.25pg of input DNA, and was able to assemble the sequences bridging 26 repeats.

Conventional assemblers struggle with repeat regions. This is because assembly relies on arbitrary length overlaps between sequenced fragments to help piece together the whole genome. If the repeat is long, fragments lying at the junction between the repeat and the rest of the genome will overlap up to the part that is in the repeat region. However at the junction, groups of reads will suggest different bases for extension, depending on which part of the genome they are originally from. Thus assemblers typically assemble up to the boundary of a repeat and proceed after the repeat.

The algorithm developed in this work accepts the sequences generated by a

conventional assembler, and links them using the 9bp overlap information in the sequences. The assembled sequences bridge repeat regions and join the non-repeat regions flanking it. In-silico analysis showed that the cell sequenced for this project was only 96% similar to its closest known reference genome. Using this reference, 57% of the reported links were validated. 4 more sequences were validated using biological techniques. This suggests that further biological experiments might reveal that a greater percentage of the assembled sequences are real. However, the reported sequences associated with very high confidence levels were found to have an accuracy of 85.7% with respect to the reference genome.

Also, a stretch of nucleotides from the strain DH10B (NC_010473.1) was discovered in the cells which could not be found in MG1655 (NC_000913.2) [17], the closest known reference genome. While other assemblers failed to link these two stretches, the algorithm developed in this project was able to assemble the bridge between these two sequences. This bridge was subsequently validated with biological experiments.

List of Tables

List of Figures

Chapter 1

Introduction

Organisms adapt to changing environments by evolving through the process of mutation. Mutations are changes in the genetic material or genome of an organism. Mutations can be broadly classified into (1) point mutation — a change in a single nucleotide (2) deletion — a stretch of nucleotides is deleted from the genome and (3) insertion — a new stretch of nucleotides is inserted into the genome.

Mutations occur at a particularly fast pace in bacteria, and play a vital role in their genetic diversity and ability to survive. Point mutations have been shown to help rifampicin-resistant M. tuberculosis survive better compared to populations of rifampicin-resistant cells without this compensatory mutation [4]. Deletions resulting in gene loss have been shown to help Salmonella enterica survive better in several growth conditions [11]. Insertions have been shown to impact gene expression, regulating the expression of their neighboring genes [19].

One type of insertion is when a segment of DNA is copied and pasted in multiple locations on the genome. Such a mutation creates repeats. In bacteria, repeats play a role in eliciting mammalian immune response, and also in the immune system of the bacteria itself [6]. Studying repeats can help gain insights into the mechanism of interaction between the bacteria and the human, potentially opening up new avenues for treatment. In this work, we focus on repeat regions.

1.1 Motivation

In clinical applications, a genomics analysis is done on both the patient and the invading pathogen. In such a case, identifying and analyzing the repeat regions in the bacteria can be instrumental in understanding the mechanism of attack. However, only a limited quantity of pathogenic cells can be extracted from a patient. Early diagnosis is helpful, and in fact essential in many diseases. Thus waiting for the cells to multiply in the lab constitutes an unacceptable delay. There is a need for a genomics pipeline which can work with small quantities of input DNA and yet identify and handle repeats.

Genomics pipelines involve three major steps — sequencing, assembly, and annotation. Sequencing attempts to read the sequence of nucleotides comprising the organism's DNA. However current technology can only read relatively short stretches of nucleotides in one go. Therefore multiple copies of the same DNA molecule are broken up into shorter fragments at random breaking points. These fragments are then read to give short stretches of nucleotides. Once this is done, assembly uses overlaps among these fragments to piece together the original genome. The assembled genome is then annotated and studied.

When a repeat region is longer than the fragments generated during sequencing, most of the fragments will lie fully within the repeat. This can prevent an assembler from realizing that these fragments come from different parts of the genome. In such cases, assemblers will typically collapse all occurrences of the repeat into one occurrence. Fragments lying at the boundary between the repeat and non-repeat regions also pose a problem. Such fragments will have the same sequence in the part that lies in the repeat. However the part that lies outside the repeat will be different. This confuses assemblers, causing them to stop assembly at such boundaries. The sequences before and after every occurrence of the repeat are output as separate stretches of assembled nucleotides.

This is a serious issue because current technology can only read fragments of length up to ˜800bp when the input quantity of DNA is low. However many repeats in bacteria are in the order of 1000s of base pairs.

1.2 Contribution

In this project, we identify a property which promises to help resolve repeats without requiring longer fragments or larger quantities of input DNA. This property is a result of one of the techniques available for breaking DNA into fragments, called Nextera.

We develop an algorithm to exploit this property and find the correct ordering of sequences generated by existing assemblers (contigs). Once the correct ordering of contigs is found, our algorithm assembles the sequence corresponding to the gap between the contigs, therefore linking the contigs. The assembled links cross repeat regions and place the contigs on either side of the repeat in the correct ordering and orientation.

We apply this algorithm to an E.coli K-12 cell known to have many repeat regions. Using only 0.25pg of DNA (˜50 molecules), we were able to correctly order and orient 26 contig pairs which were flanking repeat regions. We were also able to assemble the sequences linking them, thus generating longer contigs. The accuracy of these assembled links was 57% when compared to the closest known reference genome. However, the E.coli cells that were used to prepare the library were found to be only 96% similar to their closest known reference genome. 12 of the 26 sequences assembled by our algorithm were labeled as incorrect based on the reference genome. Biological validation was carried out for 7 of these, which showed that 4 of these 7 sequences were valid. This suggests that a greater percentage of the predictions might be true. Also, the 26 predicted sequences were associated with a confidence level, indicating how confident the algorithm was about the prediction. When considering only the very high confidence predictions (7 in number), the accuracy with respect to the reference genome was 85.7%.

We also found that the bacteria being studied had one sequence that could only be found in the strain DH10B (NC_010473.1), whereas the closest known reference was MG1655 (NC_000913.2) [17]. Using our algorithm, we were able to link this unique stretch to sequences from the reference strain. This link was experimentally validated to be correct.

1.3 Organization

The rest of this thesis is organized as follows. We first provide some Background relevant to this work, followed by a Literature Survey. We then detail the Problem Definition and Proposed Approach. The Results are then presented, followed by a Discussion. At this point, a tabular description of the results at every intermediate step in the algorithm is provided, which gives an overall feel for the flow of data through the algorithm. The Experimental Design is then described, including the properties of the library being studied. The algorithm is then explained in detail, followed by an exploration into possible Future Work. The references used in this work are listed at the end, followed by an Appendix.

Chapter 2

Background and Literature Review

Before we are able to analyze the genome of an organism, we need to determine the sequence of nucleotides that make up its genome. This is done by carrying out sequencing. Current sequencing technology has a limit on the number of nucleotides that can be read in one stretch. Therefore we first break the DNA molecules into fragments of length suitable for the sequencing technology. This is called library preparation.

2.1 Library Preparation

The methods available for library preparation or DNA fragmentation can be broadly classified into two groups — physical and enzymatic [8, 20].

2.1.1 Physical Methods

Physical methods are the most commonly used techniques to prepare next-generation sequencing libraries. Sonication applies ultrasonic waves to a sample of DNA. This produces gaseous cavities in the liquid, resulting in resonance vibration in the DNA and subsequent breakage. Nebulization forces DNA through a small hole using compressed air. This shears DNA into a fine mist which can be collected for sequencing. Physical methods typically require large quantities of input DNA (˜nanogram).

2.1.2 Enzymatic

There are two popular methods for enzymatic fragmentation available today. One method, proposed by New England BioLabs, uses a cocktail of two enzymes. One of the enzymes generates random nicks in one strand of DNA. The other enzyme recognizes the nicks produced by the first one and cuts the opposite strand across from the nick. This produces breaks in double stranded DNA. Any fragments that have been nicked but not cut on the other strand are repaired by DNA ligase.

Another enzymatic method for breaking DNA was proposed by Illumina, and is called Nextera. Here a transposase enzyme simultaneously fragments and inserts adapter sequences into the DNA molecule. This method, termed tagmentation, requires very small amounts of input DNA (picogram). Also, the sample preparation time is very low. This makes Nextera the library preparation method preferable in many cases, and is the method utilized in this paper.

In all fragmentation methods other than Nextera, adapters have to be ligated to the ends of the fragments to facilitate the sequencing process. In Nextera, the adapters are ligated in the same step as DNA cleavage.

2.1.3 Paired-end Sequencing

In paired-end sequencing, the library preparation step ligates sequencing adapters to both ends of each fragment. Thus long fragments of DNA are given as input to next-generation sequencers, and short stretches of the DNA are read from each end. The two ends are sequenced on complementary strands.

If the length of the fragment is known, this gives us extra information. For example, if the fragment is known to be 500bp long and we read 100bp from each end, we will know that the two 100bp sequences are 300bp apart. The fragment

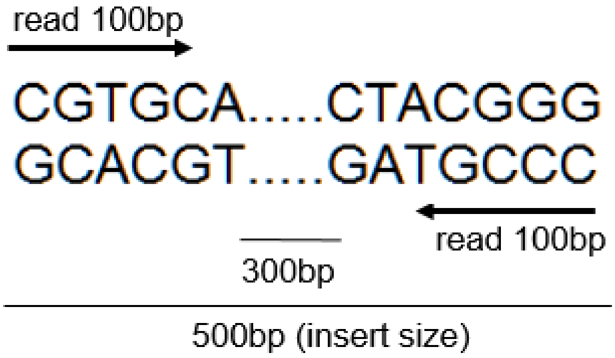


Figure 2.1: Paired-end sequencing. The DNA fragment is read for short distances from both ends. If the length of the fragment is 500bp, and we read 100bp from each end, we know that the two 100bp sequences are 300bp apart.

length is also termed insert size. This is illustrated in Figure 2.1.

The fragment length can be controlled during the library preparation step. If most of the fragments are of the same length, paired-end sequencing gives us an estimate of the distance between the two ends of every read. Thus downstream processing can use this extra information.

2.1.4 Mate Pair Sequencing

Mate pair sequencing [22] libraries are constructed by first breaking the DNA into very long fragments, between 10 and 15 kbp. These long fragments are circularized in a wash step, which simultaneously washes away fragments which were not circularized. The circular DNA is now fragmented and ligated with sequencing adapters. This method of library preparation generates reads with very long insert sizes (10 to 15kbp)

2.2 Sequencing Technologies

Sequencing is the process of determining the order of nucleotides present in a sample of DNA. Some basics about DNA are important to understand the sequencing process. DNA, or deoxyribonucleic acid, is a double stranded molecule. Each strand is made up of chemical elements called nucleotides, or bases. There are 4

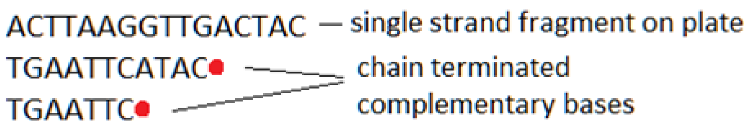


Figure 2.2: Chain termination in Sanger sequencing. Two chains are demonstrated here. First chain is terminated after 11 bases. Second chain is terminated after 7 bases.

types of bases in DNA — Adenine (A), Cytosine (C), Guanine (G) and Thymine (T). If the bases on one strand are known, the bases on the opposite strand can be derived from the fact that only complementary bases bind to each other. That is, A always binds to T on the opposite strand. Similarly, C always binds to G.

2.2.1 Sanger Sequencing (First Generation Sequencing)

This was the first large-scale method to sequence DNA, and was the method used in the first Human Genome Project. Sanger sequencing uses two basic principles — chain termination, and gel electrophoresis.

Chain Termination

Consider a single strand of DNA bound to a plate or other medium. If a pool of bases is allowed to flow across this strand, the bases complementary to the strand on the plate will bind to it. Now if some of the bases — say some of the Cs - flowing across this strand are modified such that it is impossible for another base to bind after it, the chain gets terminated. This is illustrated in Figure 2.2, where the first line is the single strand of target DNA bound to the plate. The following two lines show two chains of complementary bases. The first chain is terminated after 11 bases. The second chain is terminated after 7 bases.

Thus after flowing a mix of normal bases and some chain-terminating Cs, we will have many complementary strands starting at the same position and ending at every G in the target DNA. This is done for other bases as well.

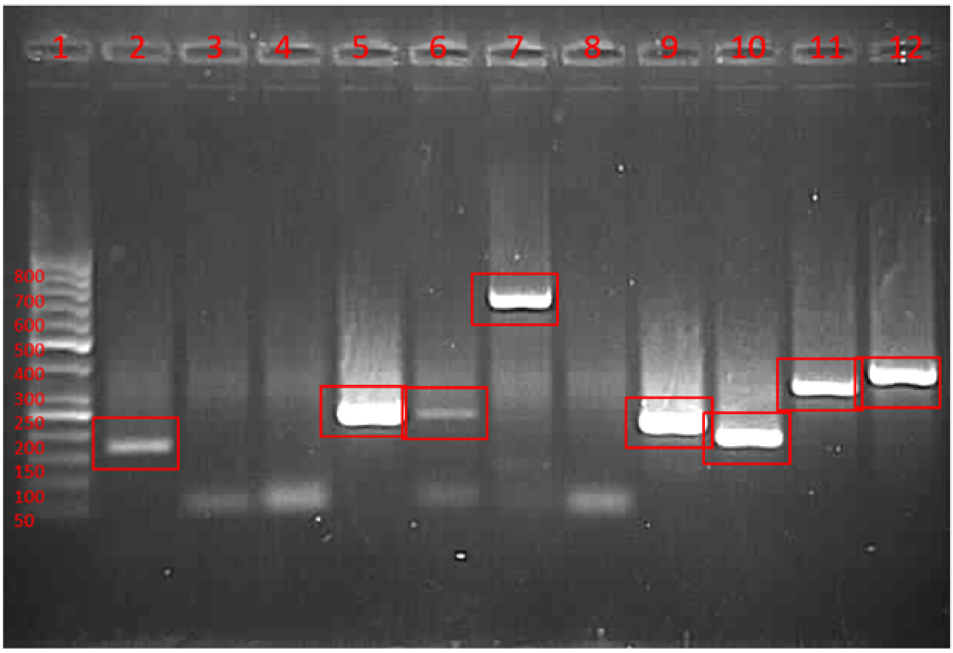


Figure 2.3: Gel electrophoresis. The smaller fragments travel faster across the gel. Thus the distance to which a fragment has traveled can tell us the length of the fragment.

Gel Electrophoresis

DNA is a negatively charged molecule. Gel electrophoresis exploits this fact by loading DNA into wells on one end of a gel, and applying a positive charge on the other end. Thus DNA strands migrate across the gel towards the other end. Due to friction, small fragments move faster across the gel (Figure 2.3).

After the chain termination step, we have all possible prefixes of the target DNA fragment. We also know which base is at the end of each prefix. Thus the length of the prefix will tell us at which position that base occurs. For example, in Figure 2.2, we know the base at the end of each prefix is C. If we carry out gel electrophoresis, we can determine that the first chain is of length 11 bases, and the second chain is of length 7 bases. Thus we know that the target DNA strand had a G at positions 11 and 7.

Using Sanger sequencing, we can read continuous stretches of ˜800 nucleotides.

2.2.2 Next-generation Sequencing

Cyclic Reversible Termination

The most popular next-generation sequencing technique follows a chain termination principle similar to that used in Sanger sequencing. An improvement made here is that the chain termination is reversible. Thus instead of finding all prefixes ending in one base, we can terminate after every base, read it, reverse the termination, and carry out the process again. This makes it possible to read massive number of fragments in parallel, speeding up the process considerably. This process is called Cyclic Reversible Termination.

After DNA is fragmented using one of the library preparation techniques discussed above, colonies of DNA are created using a process called PCR (Polymerase Chain Reaction). The exact protocol followed for the PCR differs from one company to the other. However they all serve the purpose of creating duplicates of the existing fragments of DNA. Thus after the PCR step, each colony is a cluster of duplicates, and every fragment is single stranded.

To read the actual nucleotide sequence, the 4 bases A, C, G and T are tagged with a fluorescent dye — a different color for each base. The tagged nucleotides are then allowed to pass over the fragments. The base that is complementary to the one on the fragment binds to the DNA on each colony. Since the bases being passed through are tagged with a fluorescent dye, they emit a color. Also, since the fragments within a colony are duplicates of each other, all the fragments will emit the same color. This makes the intensity of light high enough for current optics technology. An image of the emitted colors is captured with a camera, telling us exactly which base was next.

Illumina flows all 4 bases across the fragments in a single step. Helicos BioSciences flows one base at a time, making this technique slower.

Substitutions are a common error in this type of sequencing, and the error rate increases with the length of DNA read in one go. Therefore these techniques generate reads of length between 25bp and 200bp.

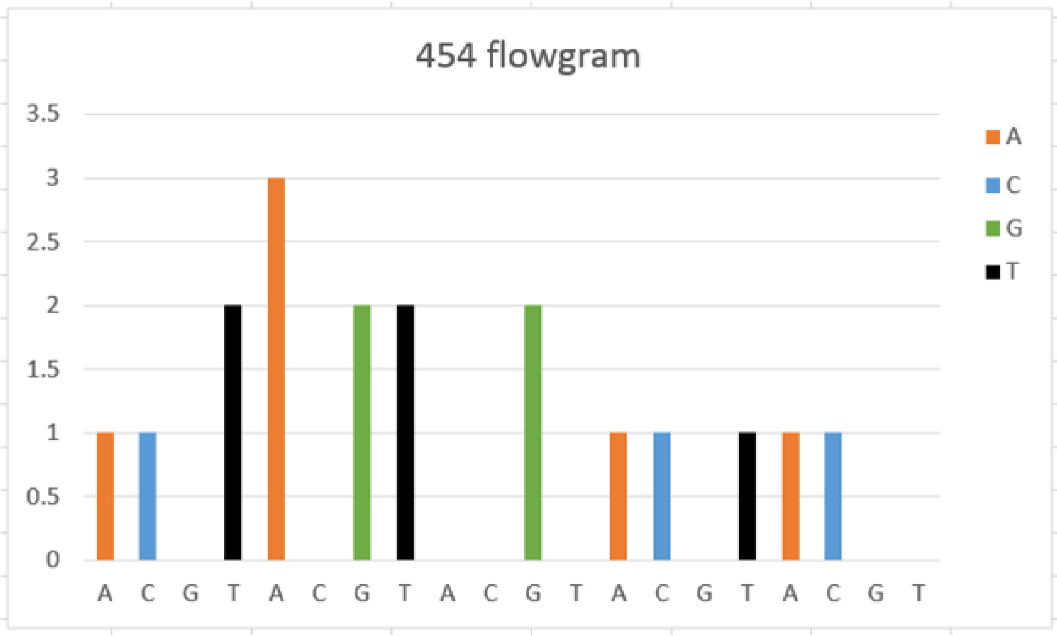


Figure 2.4: Flowgram generated by 454 sequencing. This flowgram represents the sequence ACTTAAAGGTTGGACTAC

454 Sequencing

454 sequencing uses sequencing by synthesis. In this method, the single stranded DNA molecules to be read are loaded into wells. In each iteration, one type of bases is flowed across the wells. If the base is complementary to the one on the target strand, polymerase (an enzyme which carries out DNA synthesis) extends the DNA by one base and releases a chemical called pyrophosphate. 454 technology uses enzymes sulphurylase and luciferase to convert the emitted pyrophosphate into visual light. This light tells us which wells had that base.

The output of this sequencing process is a flowgram, as illustrated in Figure 2.4. The sequence represented by this flowgram is ACTTAAAGGTTGGACTAC. Each base is represented with a different color. If the DNA strand was extended by exactly one base, light of unit intensity is generated. If a series of consecutive positions have the same base (homopolymer), the DNA is extended by that many bases in one go. Thus the intensity of light will be higher. In this example, positions 5, 6 and 7 have the base A. Thus the flowgram has intensity 3 at this point.

The problem with this technique is that it is difficult to differentiate between light intensity of n and n+1 or n-1 units, especially when n is long. Thus long homopolymers pose a serious problem to this type of sequencing.

Ion Torrent

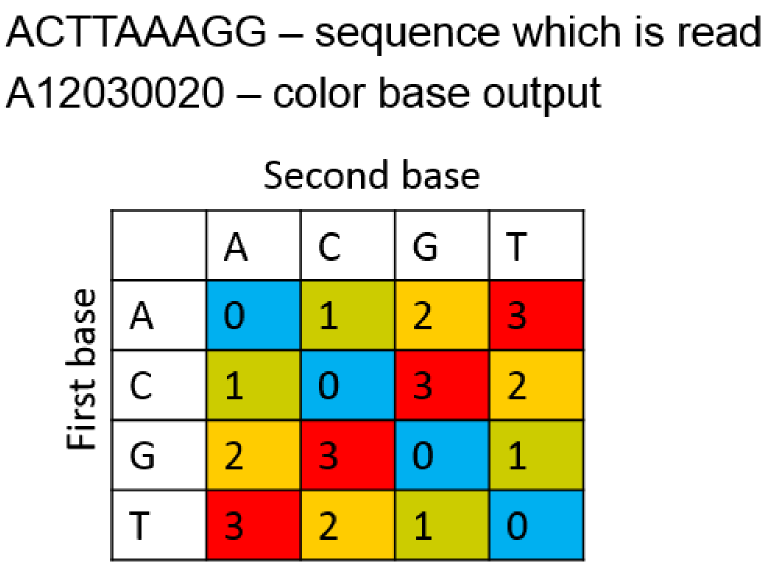
Ion torrent also uses sequencing by synthesis. When synthesis is carried out, an H^+ ion is emitted along with the pyrophosphate. Instead of converting pyrophosphate to visual light, Ion torrent uses a sensor to detect emission of H^+ as electric signals. This avoids the complicated camera and laser setup needed when visual light is used. However, this technique also struggles to handle long homopolymers.

SOLiD Sequencing

This method uses probes encoded with two-bases. The sequence that is read is output as a single base, followed by a series of numbers. In Figure 2.5, the sequence ACTTAAAGG is read as A12030020. A matrix such as the one shown in Fig 4 is then used to decode the actual sequence. For example, since the first base in the output (A12030020) is A, the first row of the matrix is the relevant row. The first number is 1. Therefore we look at the column in the first row which has the entry 1, which happens to be C. This gives us the first 2 bases as AC. Decoding proceeds in this manner to reconstruct the actual sequence.

Since encoding is done two bases at a time and each base is read twice, the number of single-nucleotide-variations (SNVs) is less. However, this method forces the additional overhead of converting from a color base to the nucleotides.

In summary, next-generation sequencing methods can handle millions of DNA fragments in each run. However a lot of preparation is required between runs. Thus the overall process is time consuming for long genomes. Also, the addition of bases at each step is error prone. Thus only short reads can be generated (25bp to 200bp).



ACTTAAAGG – sequence which is read
A12030020 – color base output

Second base

First base	A	C	G	T
A	0	1	2	3
C	1	0	3	2
G	2	3	0	1
T	3	2	1	0

Figure 2.5: SOLiD base calling gives the output in the color base. A decoding matrix is used to determine the actual sequence. In this example, the sequence ACTTAAAGG is encoded as A12030020.

2.3 Pacific BioSciences

Single Molecule Real Time sequencing (SMRT) is a sequencing technique introduced by Pacific Biosciences. First published in [10], this method also uses sequencing by synthesis, but immobilizes the DNA polymerase instead of immobilizing the strand. The technology is able to focus on a single nucleotide as and when it is incorporated by the polymerase. Up to 8,500bp can be read in one stretch using this technique [18]. However, the method is highly error prone, and repeated cycles of sequencing need to be performed to compensate for the high error rate. Thus a large quantity of input DNA (50ng) is required, and the process takes weeks to complete.

2.4 Mapping

Over the years, many genomes have been sequenced and published in public databases. When a new cell is sequenced, it is helpful to compare it to existing known genomes. Given a query sequence and a reference sequence, mapping determines the position on the reference sequence where the query has the best alignment.

In general, mapping proceeds in two steps:

1. Filtration step — Find seed(s) on the query sequence and a list of candidate hits (mapping positions)
2. Verification step — verify each hit, extending around the seed if necessary

When the query sequence is a paired-end read, the mapping algorithm has two pieces of extra information — (1) the estimated insert size (2) the fact that the two ends are on complementary strands. Thus reads which map with the correct insert size and orientation are called *concordant reads*. Other reads are termed *discordant*.

2.5 Genome Assembly

Even after the DNA sequence is read, we do not have the whole genome. Assembly is the process of utilizing overlaps among reads to piece together the original genome. Usually, assemblers cannot reconstruct the entire genome. Instead, they output a set of long assembled sequences, called contigs. In a post-processing step called scaffolding, assemblers attempt to estimate the distance between contigs.

There are two major approaches to assembly — de-Bruijn graph approach and overlap layout consensus approach. Further divisions can be made based on whether the approach uses single end reads or handles paired-end information.

2.5.1 de Bruijn Graph

A de Bruijn graph has all substrings of a certain length as its vertices. Let the length of these substrings be (k-1). Then, edges are added between two (k-1)-mers a and b if there exists a k-mer whose prefix is a and suffix is b. Thus, two vertices of the graph have an overlap of (k-2) characters. Walking along an edge and merging the overlap gives us the original k-mer.

Thus, genome assemblers using de Bruijn graphs usually expect k as a user-

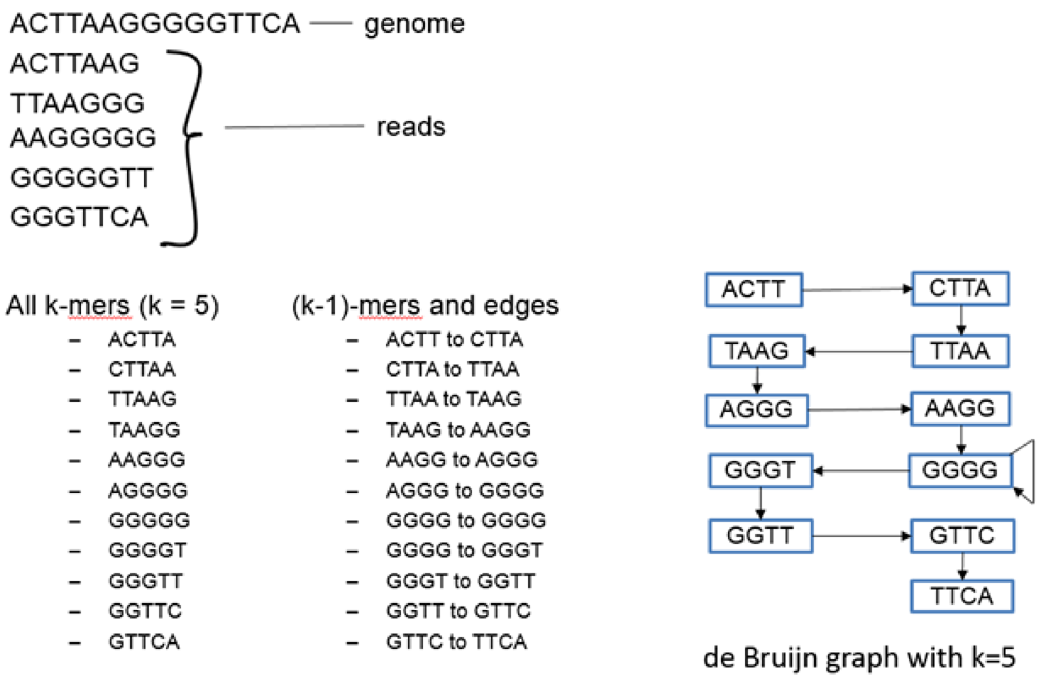


Figure 2.6: A de Bruijn graph constructed for the sequence ACTTAAGGGGGTTCA with k=5.

defined parameter. Once a value for k is chosen, the assembler adds all (k-1)-mers that occur in the reads to the set of vertices. An edge is added between two (k-1)-mers a and b if there exists a k-mer whose prefix is a and suffix is b. In Figure 2.6, we see a toy example to illustrate this construction. Here the genome is ACTTAAGGGGGTTCA, and sequencing generates 5 reads, each of which is 7bp long.

If k is chosen to be 5, the list of all 5-mers is as shown in the Fig 6. The edges in the de Bruijn graph are between the first 4 bases and the last 4 bases in every 5-mer. The graph itself is shown on the right.

If sequencing is perfect (no errors), finding the Euler cycle in the de Bruijn graph will give us the original genome. An Euler cycle is a cycle which visits every edge exactly once. In the de Bruijn graph, every edge represents a k-mer that occurs once in the genome. Also, the edges represent the correct order in which the bases follow each other on the genome. Thus an Euler cycle gives the correct assembly. It can be proved that every de Bruijn graph constructed from a set of reads with no sequencing errors has an Euler cycle [5].

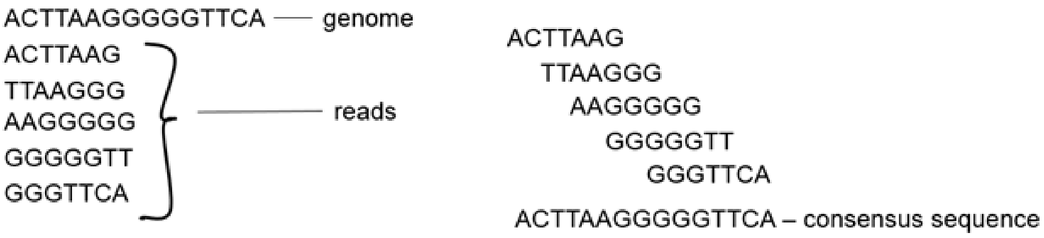


Figure 2.7: Overlap layout consensus approach. End-to-end overlaps among reads are used to layout the reads with respect to each other. Overlapping sequences are merged to get the consensus sequence.

This method is highly susceptible to errors in sequencing as a single base error can change the set of k-mers, and thus the structure of the whole graph. Thus de Bruijn graph assemblers carry out extensive error correction before constructing the graph.

If the input reads are single-end reads, the construction proceeds as discussed above. If the reads are paired-end reads, one approach is to proceed with assembly by regarding the reads as single end reads. Then the paired-end information is used in the scaffolding step to help link contigs [16]. Another approach is to correct the variance in insert sizes among reads, and then use the paired-end reads directly in the de Bruijn graph [3].

2.5.2 Overlap Layout Consensus

In the overlap layout consensus approach, end-to-end overlaps among reads are exploited to reconstruct the genome. This is illustrated in Figure 2.7. The same genome and reads used in the de Bruijn graph example are used here. Overlaps among reads are used to get the layout of reads with respect to each other. The constructed graph is shown on the right. Once the reads have been laid out, the consensus sequence is calculated, and output as the assembled sequence. The consensus sequence is usually constructed by reporting the most frequent base in each column.

This method is less prone to single nucleotide sequencing errors as the consensus sequence construction ensures that errors in a few reads is masked. If the input reads are single end, construction proceeds as described above. If the reads are paired-end reads, extension is done one base at a time and the extra information is used to determine the correct base at every step [2].

2.6 Literature Review

2.6.1 Handling Repeats

Williams et al. [21] have developed an algorithm to estimate the actual size of a genome including repeat regions by measuring the abundance of k-mers in the sequences. This technique models the frequency of 21-mers as over-dispersed Poisson distributions, and uses this to estimate the number of unique k-mers in the genome, and their relative abundance. The actual size of the genome is inferred from these values.

[21] is aimed only to determine the relative abundance of k-mers in the genome, and the actual size of the genome. No assembly is carried out.

2.6.2 de novo Assemblers

de Bruijn Graphs Approach

IDBA [16] uses the classic de Bruijn graphs approach to de novo assembly. The improvement implemented by IDBA was to automate the search for the optimal value for the parameter k. A special version of IDBA, called IDBA_ud (uneven depth) [16], was implemented to handle cases where the coverage depth on the genome was uneven. The algorithm works by trying different values for k, considering the reads as well as the contigs generated in previous iterations as input. Also, progressively deeper depth cutoffs are used to remove contigs with low depth of coverage and build longer contigs. Paired-end information is used primarily in the scaffolding step.

SOAP de novo [13] is another classic de Bruijn graphs assembler, requiring the parameter k to be tuned by the user. New data indexing and memory-efficient graph construction have been incorporated into the algorithm to make it faster and to optimize it for large genomes.

de-Bruijn graphs With Paired-end Information Approach

SPA-des [3] incorporates paired-end information directly into the de Bruijn graph rather than in a post-assembly scaffolding step. Extensive error correction is done to handle sequencing errors and chimeric reads. A process called k-bimer adjustment is used to reduce variance in insert sizes among paired-end reads. After this process, the estimated genomic distance is known for *bireads*. This information is used to guide traversal of the constructed de Bruijn graph, resulting in the final assembly.

Overlap Layout Consensus Approach

PE-Assembler [2] adapts the classic overlay layout consensus approach to incorporate paired-end information. This is done in the contig building stage by finding overlaps and laying out the overlapping reads as per the classic method. However, PE-Assembler maintains a pool of reads corresponding to the opposite end of every paired-end read used in the overlap+layout step. This pool is used to filter out unlikely extensions and extend the overlapping region one base at a time. Once a target length is reached, extension switches from single-end overlap + paired-end support through a pool, to direct paired-end overlaps. This is followed by the traditional scaffolding and gap filling steps.

Chapter 3

Problem Definition And Proposed Approach

Mutations are the driving force behind evolution in all organisms. Prokaryotes in particular mutate at a fast rate. Repeats, i.e nucleotide patterns that occur in multiple locations on the genome, are an important type of mutation. In bacteria, repeats are part of the organism's immune system, and also play a role in eliciting mammalian immune response [6]. With the advent of personalized medicine, a genomic analysis of a patient's invading pathogen is frequently carried out to tailor the treatment to the case at hand. In these cases, correctly identifying and studying the repeat regions in the bacteria can be instrumental in understanding the mechanism of interaction between the bacteria and the human. However, only small quantities of the pathogenic bacteria can be extracted from the patient. The need for quick diagnoses demand quick output from genomics pipelines, precluding the possibility of waiting for the bacteria to multiply in the lab.

Genomic analyses comprise of three main steps — sequencing, assembly and annotation. Although genome assembly has been studied for several years now, repeat regions have remained a major stumbling block for all assemblers.

The length of repeat that can be resolved by an assembler is tightly coupled with the characteristics of the sequenced library. The length of the fragments generated after breaking the DNA for sequencing is called the insert size. When the repeat region is longer than the insert size of the reads, assemblers typically collapse all occurrences of the repeat into one occurrence and output this as one segment, called contig. Also, the sequence up to the repeat and the sequence after the repeat are output as separate contigs. This is because the reads at the junction between the repeating and non-repeating regions suggest many valid branches that the assembler can take. However, the assembler does not have evidence to make the correct choice.

After assembly is complete, the assembler attempts to order the contigs that it has been able to find, in a process called scaffolding. This can be done for contigs which are less than 1 insert size apart by looking for paired-end reads with one end on each contig. However if the contig borders a repeat region, scaffolding will find paired-end reads which support all orderings.

This is illustrated in Figure 3.1, Here a repeat region (red) occurs in two places on the genome. The first occurrence has a blue sequence on the left and a green sequence on the right. The second occurrence has a green sequence on its left and a black sequence on its right. The yellow linked blocks in Figure 3.1A represent the paired-end reads that correspond to this part of the genome. As can be seen, the repeat region is longer than the insert size.

Figure 3.1B illustrates the problem that assemblers face. Typically only one occurrence of the repeat region is assembled, and output as one contig (red). The blue, green and black sequences are output as 3 separate contigs. In the scaffolding step, the assembler attempts to link these four contigs in the correct order by looking for a paired-end read which has one end on one contig and the other end on another contig. In this case, paired-end reads can be found which link all possible orderings — (blue, red, green), (green, red, black), (green, red, blue), (blue, red, black), (green, red, black), (black, red, blue).

Also, conventional sequencing methods suffer from the disadvantages of requiring very large quantities of input DNA (˜10 ng) and being very time consuming

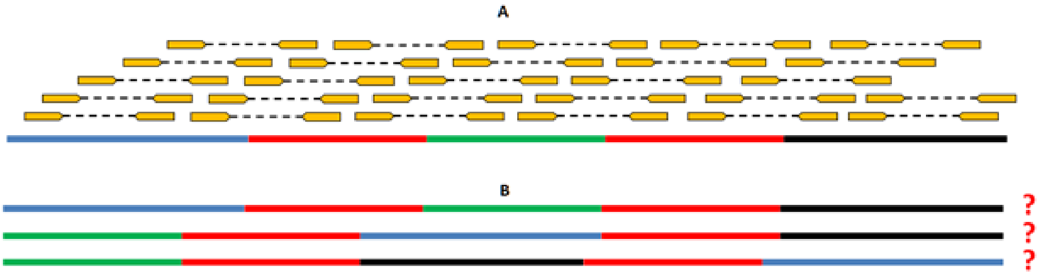


Figure 3.1: A: A repeat region (red) occurs in two places on the genome. First occurrence is flanked by a blue sequence and a green sequence. Second occurrence is flanked by a green sequence and a black sequence. The repeat region is longer than the insert size of the paired-end library (linked yellow blocks). B: In such situations, assemblers typically output 4 contigs, one corresponding to each of the blue, red, green and black sequences. However it cannot correctly resolve the order in which the contigs should be placed. Paired-end reads can be found which support every order - (blue, red, green), (green, red, black), (green, red, blue), (blue, red, black), (green, red, black), (black, red, blue).

(1 - 10 days).

Thus there is a need for a sequencing-assembly combination which can handle long repeats without requiring large quantities of input DNA.

3.1 Problem Definition

Given a library sequenced with very small quantities of input DNA and a set of contigs output by any assembler, identify adjacent contigs. Link adjacent contigs by assembling the gap between them. Output the set of long contigs constructed by linking input contigs. Also output any contigs for which no links could be found.

3.2 Possible Solution — Mate Pair Sequencing

One possible method to address this issue is to use mate pair sequencing [22]. This technique is capable of generating sequences with insert size up to 10kbp. Thus repeats of up to 10kbp can be resolved. However, this sequencing technique requires very large quantities of input DNA (˜50ng), and takes a long time (weeks) to prepare the library.

3.3 Proposed Approach

One method to sequence very small quantities of input DNA is by using Nextera technology. Using this method, picogram quantities of DNA can be sequenced [15]. When the library is prepared using Nextera, the sequences have a characteristic property. In order to understand this, we need to understand the cutting mechanism in greater detail.

3.3.1 Nextera Cutting and Its Consequent Property

The enzyme used for cutting in Nextera is a transposon called Tn5. During cutting, the transposon is inserted at a random position in the target DNA, leaving an overhang of 9bp on either side. The 9bp overhang is filled up on the complementary strand, after which the DNA is sheared into two pieces. At the end of this process, the fragment on either side of the cut site has a 9bp repeat (Figure 3.2).

When paired-end sequencing is carried out, the repeated 9bp is read once on the fragment to the right of the cut site and once on the fragment to the left of the cut site (Figure 3.3). This results in a 9bp overlap between adjacent fragments.

Let us make the simplifying assumption that Tn5 cuts truly randomly. That is, every cut-site is unique, and no two molecules are cut at the same place. Also, let us assume that 9bp is long enough to ensure that the overlapping sequence at every cut-site occurs only once in the genome. Under these assumptions (and no errors in sequencing), two paired-end reads involved in an end-to-end 9bp overlap are next to each other on the genome, and come from the same molecule in the sample.

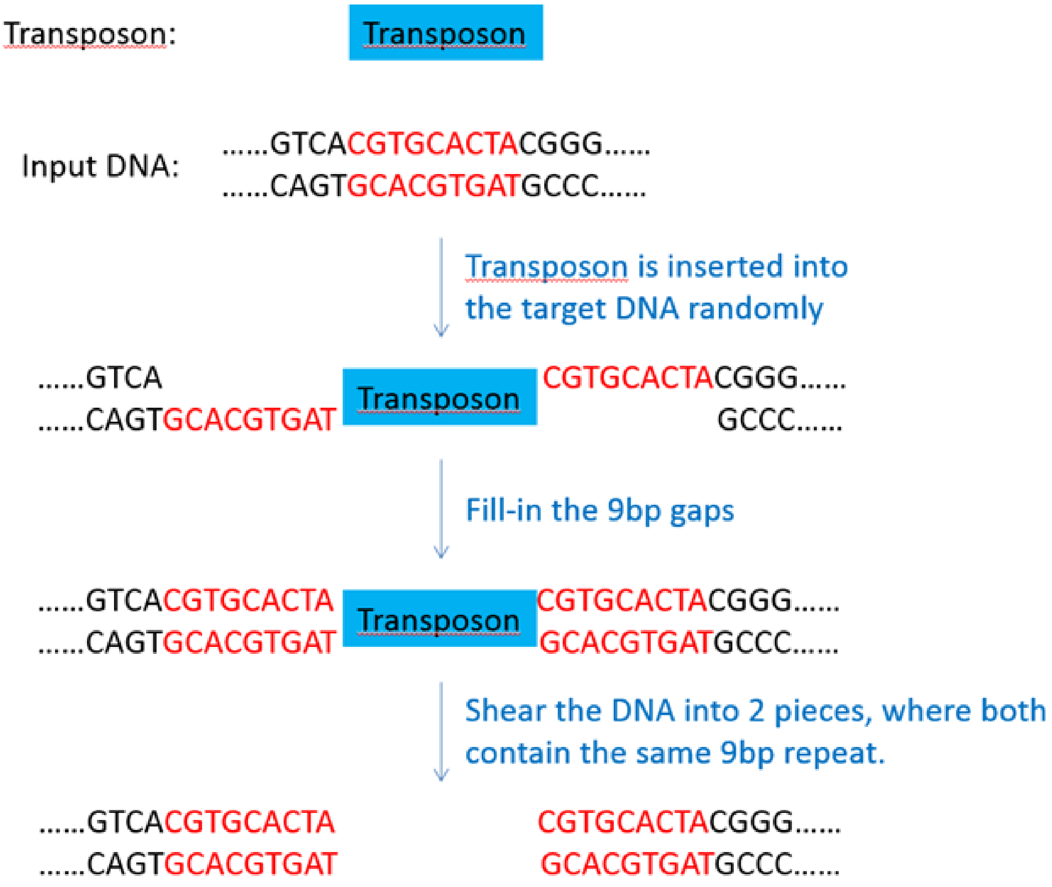


Figure 3.2: Mechanism of cutting using transposon in the Nextera XT kit leaves a 9bp repeat on the fragments on either side of the cut-site.

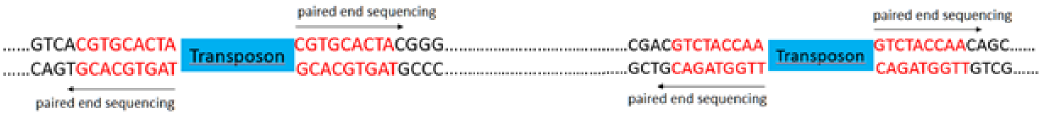


Figure 3.3: Paired end sequencing after transposon cutting results in 9bp overlap between adjacent fragments

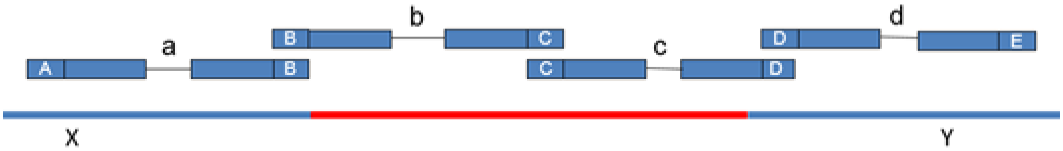


Figure 3.4: A unique chain of reads crossing a repeat region can identify the correct ordering of contigs

3.3.2 9bp Overlaps And Repeat Regions

As a result of the mechanism of cutting, a 9bp overlap between two paired end reads indicates that these two paired end reads might be next to each other on the genome. Therefore we construct chains of reads by utilizing the 9bp overlap. If a chain links contig X and contig Y, and the reads in this chain do not link any other pair of contigs, it indicates that contig X and contig Y must be adjacent to each other. It also gives us parts of the sequence corresponding to the gap between the contigs. This is illustrated in Figure 3.4, where the gap between contig X and contig Y is in fact a repeat region (red). However, the fact that reads a, b, c and d uniquely link these two contigs suggests that contig X is next to contig Y in the original genome. The reads lying in between the contigs also give us stepping stones to retrieve the (repeat) sequence that occurs between the contigs.

Chapter 4

Results

In order to test the proposed approach, an E.coli K-12 library was prepared using the Nextera XT kit. E.coli was isolated from Top10 competent cells (Life Technologies), and 0.25pg of E.coli corresponding to ˜50 molecules were used. The library comprised of 819,798 paired-end reads where each end was 100bp. Thus the average coverage was ˜35x. Also, the average insert size was 250bp.

Although the Top10 cells were reported to be genetically similar to the DH10B strain, mapping showed that only 9.62% of paired-end reads could map concordantly to this strain. On the other hand, 96.55% of the reads could be mapped concordantly to the MG1655 strain (see Chapter 5, *Experimental Design*). Thus the reference genome used in this thesis is E.coli K-12 MG1655 [17]. This library is referred to as N504 in the rest of the thesis. Further details about the library can be found in the *Experimental Design* chapter, Chapter 5.

4.1 Assemblers Fail At Repeat Regions

We hypothesize that assemblers fail at repeat regions. To test this, we assemble the N504 library using 4 different assemblers. The main methods used for assembly are (1) de-Bruijn graphs approach, (2) de-Bruijn graphs with paired-end information approach, and (3) overlap layout consensus approach. Assemblers from each of these methods were chosen – IDBA and SOAP de novo (de-Bruijn graphs),

Assembler	N50	Total length	No. of contigs
IDBA	76,486	4,489,247	155
SOAP de novo	30,180	4,517,303	961
SPA-des	76,398	4,488,147	179
PE-Assembler	18,013	7,846,089	920

Table 4.1: Summary of assembly results

SPA-des (de-Bruijn graphs with paired-end information) and PE-Assembler (overlap layout consensus graphs with paired-end information). Assembly was carried out after PCR duplicate removal. The results from each of these assemblers are summarized in Table 4.1. IDBA was chosen as the assembler for this work.

On mapping the generated contigs to the reference genome, it was discovered that 32 of the gaps between contigs were common to all the assemblers. 26 of these gaps could be explained by repeats (Figure 4.1). The shortest repeat was ˜800bp long. The large number of repeats in the sequenced cells can be attributed to the fact that the DH10B strain has been proved to have a 13.5-fold higher mutation rate than wild-type E.coli [7], caused by a drastic increase in insertion sequence (IS) transposition. A full table describing the contig gaps and the repeat regions causing them can be found in Appendix A.

Thus it can be demonstrated that repeat regions longer than the insert size of the library cannot be handled by current-day assemblers.

It is interesting to note that for every assembler, some contigs could not be mapped to the MG1655 reference genome [17]. In the case of IDBA, one contig could not be mapped to the reference genome. This contig was successfully mapped to DH10B (NC 010473.1).

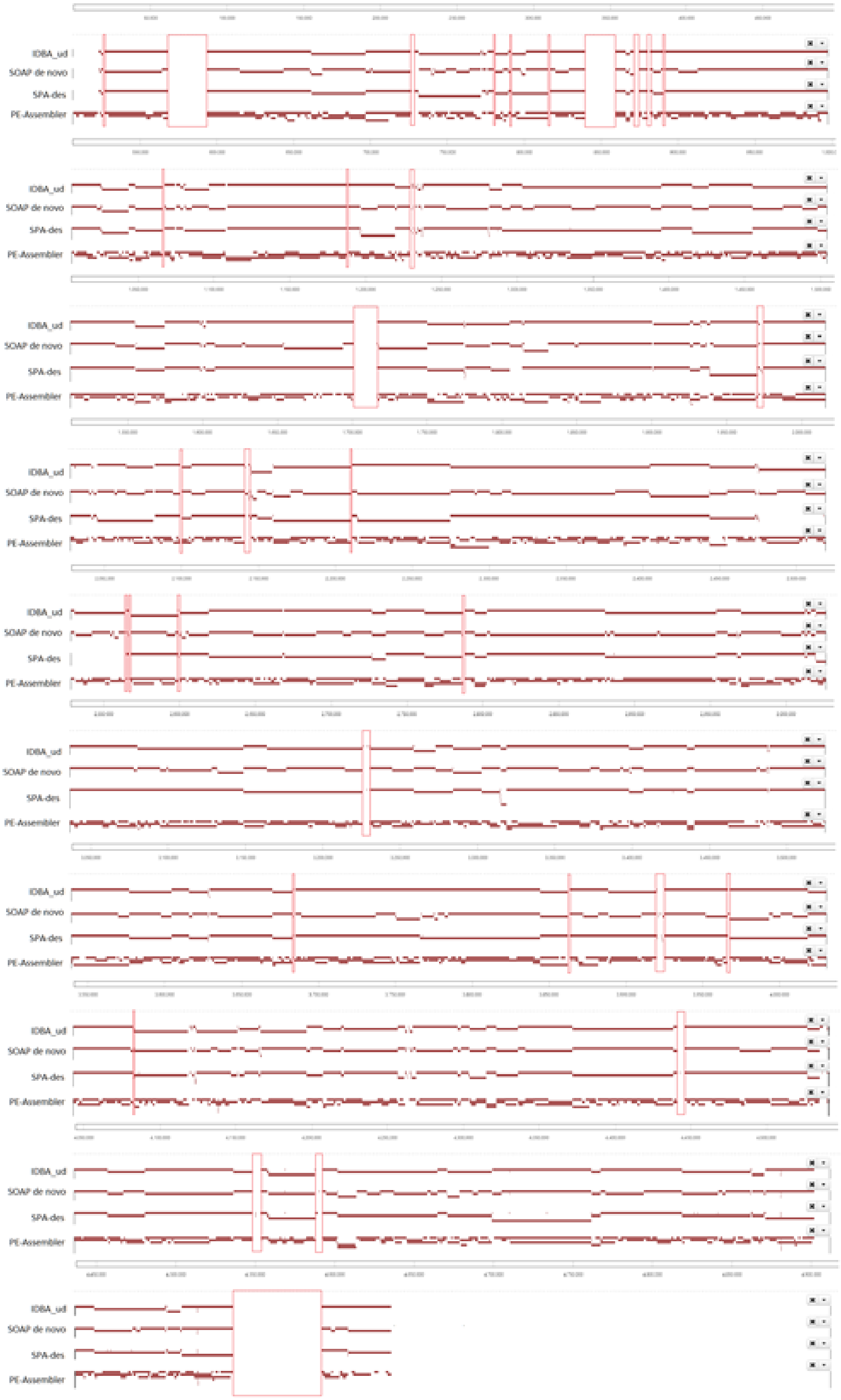


Figure 4.1: Assemblers fail at long repeats. IDBA_ud (uneven depth), SOAP de novo, SPA-des and PE-Assembler fail at common locations highlighted by red boxes.

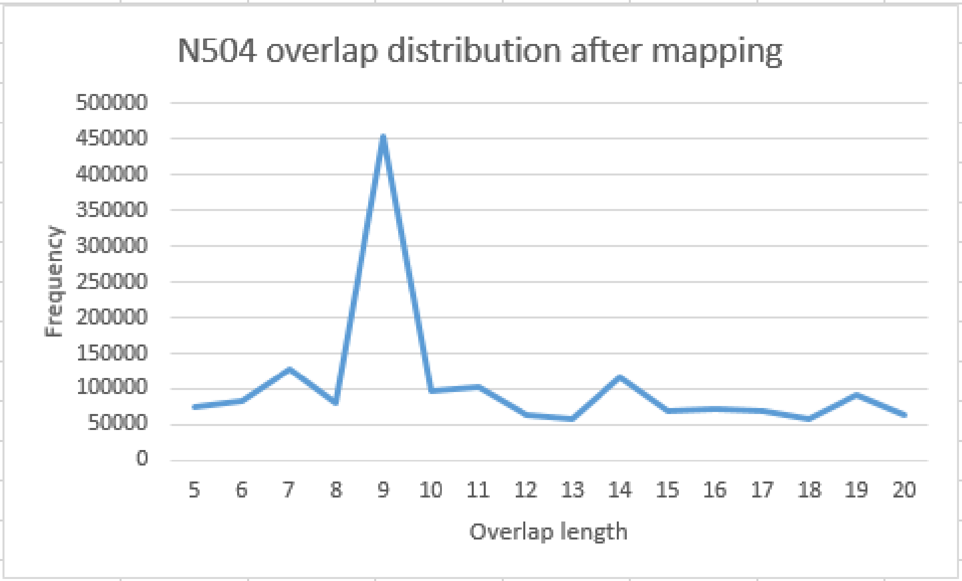


Figure 4.2: Reads were mapped to the closest known reference, and overlap length between adjacent paired-end reads was calculated. 9bp was found to be the most common overlap length

4.2 Nextera Leaves A 9bp Overlap

In order to verify that Nextera indeed leaves a 9bp overlap between adjacent reads, we mapped the sequenced library to the closest known reference genome. In our case, the closest known reference genome is E.coli K-12 MG1655 [17]. Mapping was carried out using BWA [12]. The length of overlap between adjacent paired-end reads was calculated, and the frequency of occurrence of each overlap length was measured (Figure 4.2). It was found that 9bp was indeed the most frequent overlap length.

In the N504 library, there are 15,573,574 pairs of paired-end reads involved in 9bp overlaps. Among them, only 452,987 are considered correct, i.e., they are mapped adjacently with 9bp overlap on the reference genome. This means that most of the 9bp overlaps are incorrect. We developed a de novo method to reduce the number of false adjacent paired-end reads (see Chapter 6, *Algorithm, Constructing overlaps graph*). After filtering, 75% of the overlaps retained by our method were found to be true with respect to the reference genome. 3% true

Start pos.	End pos.	Match	Mismatch
273038	274233	1195	0
573533	574728	1195	0
686793	687988	1195	0
1393507	1394702	1195	0
2098932	2100127	1195	0
2286030	2287225	1195	0
3126907	3128102	1195	0
3362177	3363372	1195	0
3648588	3649783	1195	0
2063342	2064537	1190	5

Table 4.2: Transposon repeat being studied. The repeat occurs in 10 locations on the genome. 9 of the occurrences are identical

positives were missed.

4.3 9bp Overlap Chains Link The Regions Flanking A Repeat

Since the Tn5 transposon used for cutting in the Nextera kit cuts the genome nearly randomly, we expect that each occurrence of a repeat is cut at different positions. To study this, we identify a repeat of length ˜1,100bp. This repeat occurs 10 times in the E.coli genome. 9 of the occurrences are exactly identical, while the 10th occurrence has a 5bp mismatch (Table 4.2).

Upon carrying out a BLAST [14] analysis, it was found that this sequence corresponds to *"Escherichia coli str. K-12 substr MG1655 beta-galactosidase (lacZ) gene, complete cds; insertion sequence IS5 transposase (insH) gene, complete cds;*

and lactose permease (lacY) gene, partial sequence". This indicates that this is a Transposable Element repeat, and is one of the repeat types that occurs frequently in humans as well. This repeat is referred to as the transposon repeat in the rest of this thesis.

We look for chains crossing the transposon repeat by discovering supported overlaps and chaining the overlapping reads. It was found that 7 of the 10 occurrences had chains crossing it. We were able to identify 2 chains per repeat where the reads used in the chains were not used to link any other contig pairs. The mapping locations of the reads on the reference genome revealed that the chains indeed had distinct cut sites (Figure 4.3).

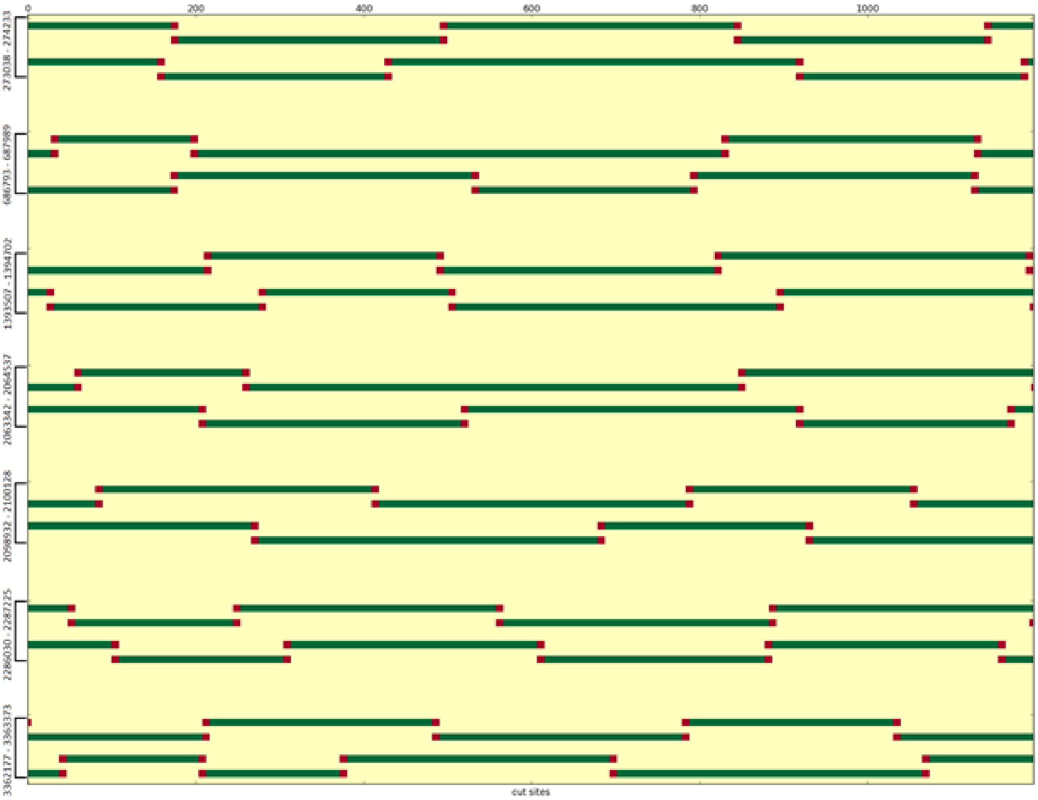


Figure 4.3: Read chains with 9bp overlap cross 7 out of 10 occurrences of a transposon repeat and link the flanking contigs. Mapping locations showed that the 7 occurrences were cut at different cut-sites, enabling us to link the correct contig pairs. X axis indicates the position of each read, offset from the repeat's start position. Y axis indicates the position of each repeat on the E.coli K-12 MG1655 genome.

4.4 Repeats Resolved

An algorithm was developed to exploit the 9bp overlap property of Nextera libraries. The algorithm accepts a library prepared using Nextera technology with very small quantities of input DNA (˜picogram). It also requires the contigs generated by any assembler as input. The algorithm first discovers 9bp overlaps among reads, constructing an overlaps graph in the process. It then finds reads which can serve as anchors on the contigs, and looks for a path in the overlaps graph which can lead from one contig to another. These contig pairs are declared potentially adjacent. All the contigs potentially adjacent to a given contig, along with the overlap chain linking them, forms the contig adjacency graph for that contig. A contig adjacency graph is constructed in this manner for every contig.

Local assembly is carried out to fill the gaps in the overlap chains, giving us longer contigs constructed by linking input contigs. These longer contigs are then filtered using split reads — reads where the two ends map across the boundary between the contig and the newly assembled sequence. If any adjacency is a subsequence of another, the two are merged.

Now, each contig can only be next to one contig on the left and one contig on the right. Thus, each contig will have degree ≤1 on each side. Thus the reported adjacencies resulting in contigs with degree >1 are identified. The candidate adjacencies are ranked, and the highest ranked one is retained. After all contigs have ≤1 contig on each side, the final adjacencies are ranked again to determine the level of confidence the algorithm has in that result. This final list of long contigs, along with the confidence scores, is given as output (see Chapter 6, *Algorithm*). Any input contigs not participating in the assembled longer contigs are also added to the output.

The flow of results through the algorithm is shown in the Intermediate Results section (Section 4.7) in Tables tables 4.4 to 4.13.

Mapping revealed that some short contigs could be mapped to multiple locations on the genome, because of which these short contig could be adjacent to

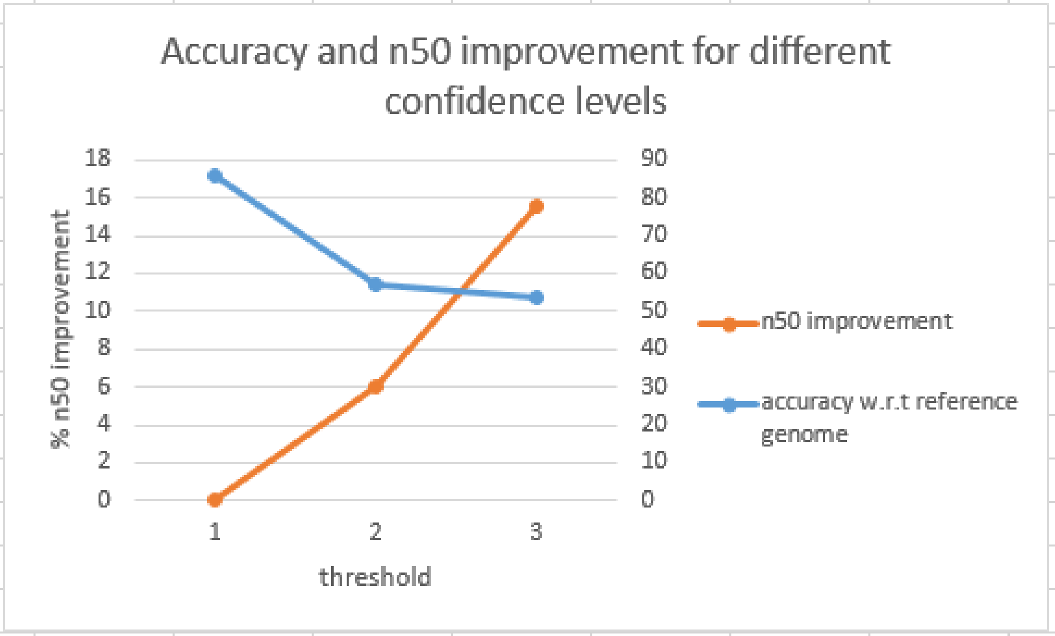


Figure 4.4: Ranking of assembled adjacencies can be cut off at various thresholds. Different thresholds result in different trade-offs between accuracy and improvement in n50.

more than 2 contigs. Thus the degree restriction was imposed only on long contigs ($\geq$ 3,000bp) during ranking.

In the N504 library, 26 longer contigs were presented as output at the end of the ranking process. These 26 adjacencies were created by linking 39 input contigs. The final adjacencies were further divided into 3 categories based on the confidence level — very high confidence, high confidence, and low confidence.

In-silico validation was carried out by comparing the generated longer contigs against the reference genome. Also, the improvement in n50 because of the longer contigs was calculated. The results are summarized in Figure 4.4. A tradeoff between the accuracy and % improvement in n50 is immediately apparent.

4.5 Experimental Validation

For 5 longer contigs (adjacencies), the input contigs were mapped next to each other on the reference genome, but an insertion was predicted between the two contigs. Out of these, 4 longer contigs were of relatively higher confidence. Also, one input contig could not be mapped to the MG1655 reference genome. Our algorithm was able to link this contig to 2 other contigs which were mapped successfully to MG1655, providing the missing bridge between these regions. These cases were given for biological validation to a group in the Genome Institute of Singapore.

To verify whether the predicted inserts were really present in the sequenced cells, primers were designed on either side of the predicted insert. The primers were chosen such that they belonged to the regions which map to the MG1655 reference genome. A standard Taq Polymerase (NEB) kit was then used to amplify the region in the between these primers. The length of the sequence between the primers in the presence/absence of the predicted inserts is known. Thus the length of the sequence amplified during PCR gives us an indication as to whether or not the predicted insert is present in the cells.

The results of the validation are as show in Table 4.3. The first 5 cases in the table are the cases where both contigs map to the MG1655 reference. The contigs map next to each other, but our algorithm predicted an insert. The validation was inconclusive in one case (case 4) while another case (case 1) showed that the sample had multiple types of cells. Some cells had the predicted insert while others did not. 1 insert (case 3) was conclusively proved to be true, while 2 were proved wrong. As can be seen from the table, the cases which were proved real were the cases with relatively higher confidence levels.

For the 2 cases where our algorithm assembled the bridge between a sequence mapping to MG1655 and a sequence mapping to DH10B (case 6, 7), both cases were validated to be true. That is, the cells in the sample had the predicted bridge. Ranking found that one of the two was a sub-sequence of the other. Thus the two sequences were merged, and the longer sequence was presented in the final output.

The adjacencies validated here were chosen because the contigs involved were mapped next to each other on the reference genome, but without the predicted insert. However, all the adjacencies were from the low and very low confidence groups. Testing the adjacencies in the higher confidence groups might reveal that a greater fraction of the predictions are in fact present in the sequenced cells.

Sl. no.	Left contig length	Right contig length	Linked contig length	Predicted insert	Valid insert?	Confidence level
1	162502	45477	209426	1447	Some cells had inserts while others did not	Low
2	8582	1744	11772	1446	N	Very low
3	81032	506	82265	727	Y	Very low
4	81032	27853	110085	1200	Inconclusive	Very low
5	31469	1638	34254	1147	N	Low
6	11125	845	12152	182	Y	Low
7	12729	845	13675	101	Y	Low

Table 4.3: Longer contigs 1 through 5 are cases where both contigs map next to each other on the MG1655 reference, but our algorithm predicted an insert. The validation was inconclusive in 1 case (case 4), while another case showed that the sample had multiple types of cells (case 1). One case was conclusively validated, while 2 cases were proved incorrect. Both predictions involving the contig not mapping to MG1655 (case 6, 7) proved to be correct.

4.6 Discussion

Our approach has successfully demonstrated that the 9bp overlap property of Nextera can be used to handle repeat regions. This addresses a gap in existing technology since conventional assemblers struggle with repeats. We were also able to find the bridge between a sequence from the MG1655 strain and the DH10B strain that were adjacent in the sequenced cells. This was validated experimentally, proving the efficacy of the algorithm.

For the N504 library used, 9bp overlap chains could be found linking 100 real adjacencies (real with respect to the MG1655 reference genome). Using the filtering criteria detailed in this thesis, only 14 of these could be unambiguously retained in the final list. Relaxing these filtering criteria might retain more of the real adjacencies, with an associated loss in accuracy. Also, IDBA generated 132 contigs, indicating that 131 adjacencies should have been discovered. However only 100 contig pairs had 9bp overlap chains linking them. This suggests that for the remaining 31 contig pairs, some part of the segment connecting them does not appear in the reads. This problem may be addressed by using more than 1 input library.

Although this project works exclusively with bacterial genomes, it is conceivable that the approach will work with human genomes as well. A quick back-of-the-envelope analysis will illustrate this point. The E.coli genome is ˜4,500,000bp long. Since the insert size is 250bp on average, this would imply ˜18,000 fragments per genome. ˜50 genomes were sequenced, giving us an ideal 900,000 fragments and cut-sites. If no two cut-sites are the same, this gives us one cut-site per 5 bases. This was sufficient information to provide overlap chains for 100 out of 132 gaps. For a human genome, which is ˜3 billion bp long, an insert size of 250bp would imply 12,000,000 fragments per genome. If 10 genomes are sequenced, this gives us 120,000,000 fragments and cut-sites. If no two cut-sites are the same, this gives us one cut-site per 25 bases. With a read length of 100bp, this should give us enough information to find overlap chains covering most of the genome.

4.7 Intermediate Results

Input	No. of reads(2x100)	Coverage
Reads	819,798	35.35x
PCR duplicate removal	809,714	34.91x

Table 4.4: Input reads

	n50	Total length	No. of contigs	Longest contig	Shortest contig
IDBA contigs	76,486	4,489,247	155	254,174	222

Table 4.5: Input IDBA contigs

No. of supported 9bp overlaps	581,632
No. of vertices in supported overlaps	520,017
Total no. of reads	809,714
Max. degree	89

Table 4.6: Constructing overlaps graph

	n50	Total length	No. of contigs	Longest contig	Shortest contig
Trim contigs	76,286	4,454,629	133	253,974	266

Table 4.7: Finding Anchors

no. of potential adjacencies	1292
mp. of contigs in potential adjacencies	123
Validation	
no. of contigs mapping to reference genome	132
no. real w.r.t reference genome	100
no. of contigs in real adj	112
no. of potential adj using the unmapped contig	2

Table 4.8: Constructing contig adjacency graph

no. of adj assembled	235
no. of contigs in assembled adj	90
Validation	
no. real w.r.t reference genome	69
no. of contigs in real adj	65
no. of assembled adj using the unmapped contig	2

Table 4.9: Local assembly

no. of adj retained after finding split reads	73
no. of real adj	32
no. of adj with split reads using the unmapped contig	1

Table 4.10: Split reads

no. of adj discarded after mergin	6
no. of adj after merging using the unmapped contig	1

Table 4.11: Mergin sub-sequences

no. of adj after ranking with deegree	26
no. of real adj	14
no. of adj using the unmapped contig	1

Table 4.12: Ranking

	no. of adj	accuracy (%)	n50 improvement (%)
very high confidence	7	85.7	0
very high + high confidence	14	57	6
very high + high + low confidence	26	53.8	15.5
adj using unmapped contig confidence level	low		

Table 4.13: Ranking results

Chapter 5

Experimental Design

5.1 Genomic DNA

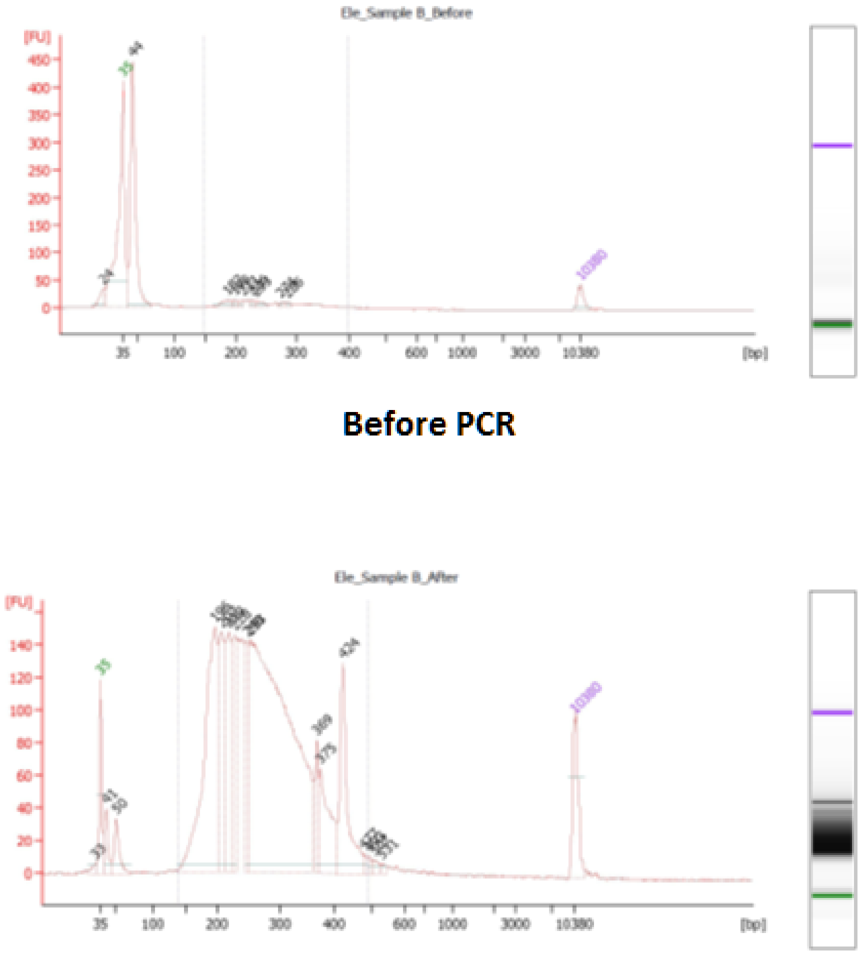
Genomic DNA for E.coli was prepared by the Genome Institute of Singapore (GIS). E.coli DNA was isolated from TOP10 competent cells (Life Technologies). The cells are genetically similar to the DH10B strain. The genotype is F^{-} $mcrA$ $\Delta(mrr\text{-}hsdRMS\text{-}mcrBC)$ $\phi 80 lacZ\Delta M15$ $\Delta lacX74$ $recA1$ $araD139$ $\Delta(ara\text{-}leu)$ 7697 $galU$ $galK$ $rpsL$ (Str^{R}) $endA1$ $nupG$ $\lambda -$.

5.2 Library Preparation

Isolated DNA was quantified using Qubit ds HS assay (Cat no.Q32854, Life Technologies) and diluted to 0.25pg. Tagmentation was performed by Nextera XT kit (Cat no. FC-131-1024, Illumina).

5.2.1 Tagmentation Protocol

For 0.25pg E.coli library, 0.5µl of DNA (0.488pg/µl) was incubated with 3µl of tagmentation buffer (Nextera XT kit), and 1µl of tagmentation mix and 1.5µl of nuclease free water (Promega). The reaction was incubated at 55°C for 8 minutes. The reaction was neutralized by adding 1.5µl of neutralization buffer with 5min incubation at room temperature. PCR amplification was performed



Before PCR

After PCR

Figure 5.1: Profile before and after PCR

by adding 7.5µl of Nextera PCR buffer and 2.5µl of PCR index primer N504 and N706. This was then cycled under standard Nextera XT conditions for 15 cycles. The amplified DNA was cleaned up using Ampure beads (Ampure XP, A63880, Beckman coulter) at 0.6x beads to volume ratio and eluted in 12µl of nuclease free water (Promega) to select a size range from 150 to 500bp. Libraries were run on High Sensitivity Bioanalyzer (Agilent) for size verification and sequenced by lllumina Hiseq as a paired end 101bp. The Agilent profiles before and after PCR are shown in Figure 5.1.

After sequencing, 819,798 paired-end reads were generated, where each end was 100bp. PCR duplicate removal caused the number of paired-end reads to

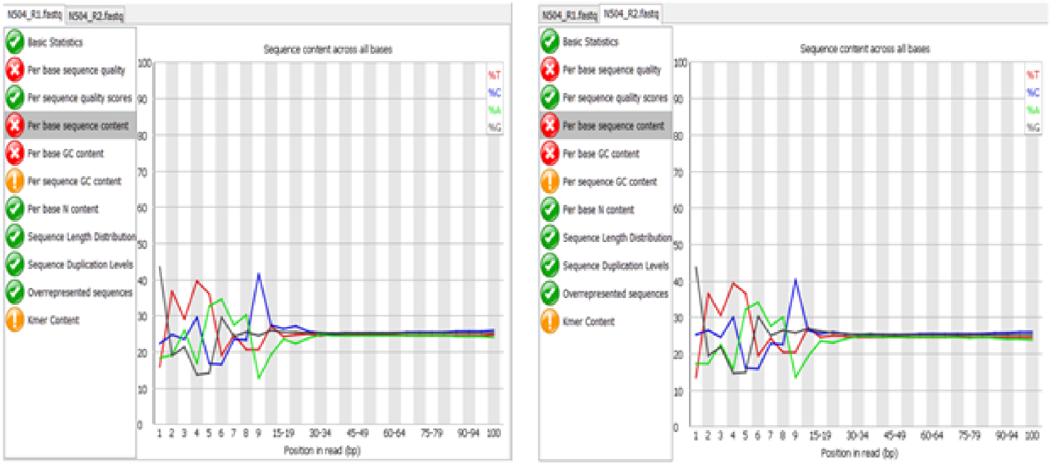


Figure 5.2: Analysis with fastqc shows a bias in the first 30bp

reduce to 809,714. The average coverage was 35x.

5.3 fastqc

An analysis with fastqc [1] showed that there is a bias in the first ˜30bp (Figure 5.2). The consensus sequence in the first 9bp was found to be GTTTTAAAC. The consensus was the same on both ends of the paired-end reads.

5.4 Reference Genome

The isolated Top10 competent cells (Life Technologies) are reported to be similar to the E.coli K-12 DH10B strain. Sequenced reads were mapped using BWA [12] to the DH10B reference genome (NC_010473.1). It was found that only 9.62% of reads mapped concordantly to this genome. Since the DH10B strain is a result of serial genetic recombination and is derived from the wild type strain MG1655 [7], we tried mapping the reads to the E.coli K-12 MG1655 strain (NC_000913.2) [17]. This time 96.55% of reads could be mapped concordantly. Thus we use the E.coli K-12 MG1655 as the reference genome in the rest of the thesis.

5.5 Insert Size

Insert sizes were calculated by mapping the reads to the reference genome using BWA [12]. The insert size distribution is as shown in Figure 5.3.

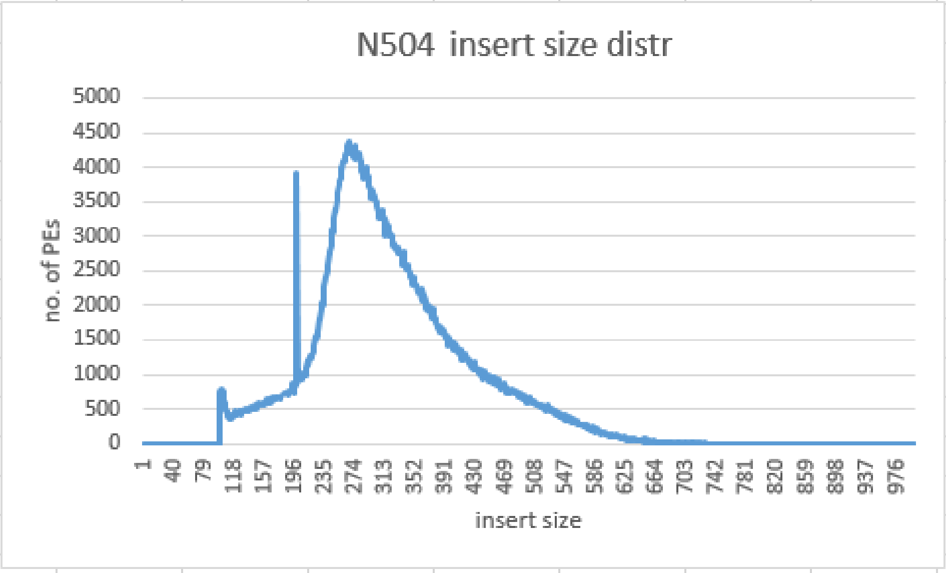


Figure 5.3: Insert size distribution of N504 library. Most frequent insert size = 267. Peaks at insert size 103 and 200 (Note: reads are 2 x 100)

5.6 Coverage

If a read cannot be mapped in one stretch or if only one end of a paired-end read can be mapped, the read is marked by BWA [12] with a mapping quality of zero. When considering only reads with non-zero mapping quality, ˜3% of the genome had no coverage. This indicates that there are several repeat regions which are sufficiently similar to each other to confuse the mapping software. Table 5.1 contains details of the coverage gaps. The frequency of occurrence of various gap lengths can be found in Figure 5.4.

Largest coverage gap	56595
Smallest coverage gap	1
Percent uncovered	3.2319
Number of gaps	88

Table 5.1: Details of coverage gaps left when the N504 library was mapped to E.coli K-12 MG1655. Only reads with non-zero mapping quality are considered

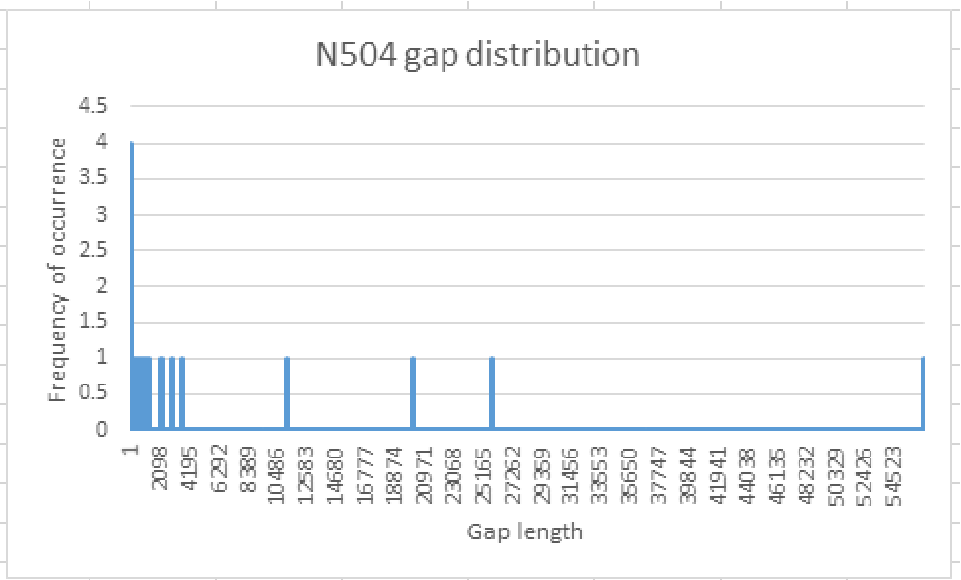


Figure 5.4: Frequency of occurrence of different gap lengths when the N504 library was mapped to E.coli K-12 MG1655. Only reads with non-zero mapping quality are considered.

Chapter 6

Algorithm

6.1 Overview

The aim is to find 9bp overlap chains linking adjacent contigs, and assemble the gaps in the chains. Here the contigs are part of the input provided by the user. If paired-end read b has a 9bp overlap with read a on the left and with read c on the right, reads a, b and c form a 9bp overlap chain.

However, PCR duplicates can confuse assemblers and make them believe a branch has more support than it actually does. Therefore we require that PCR duplicate removal be carried out before assembly. To this end, we have implemented a PCR duplicate removal tool, described briefly in the section PCR *duplicate removal*. Duplicates can also be removed using any tool of the user's choice.

Given the reads after PCR duplicate removal, we construct an *overlap graph* by discovering end-to-end 9bp overlaps among paired-end reads and chaining them together. Paired-end reads which map uniquely to one contig are identified as *anchors*. We use these anchors as starting points, and traverse the overlaps graph. If an overlap chain starts from an anchor on one contig and leads to an anchor on a different contig, the two contigs are declared potentially adjacent. Thus we discover all chains linking every contig to the other contigs. The collection of chains linking a given contig to other contigs forms a graph, termed the *contig* *adjacency graph*.

Every chain consists of paired-end reads joined end-to-end by the 9bp overlap. The sequence between the two ends of every paired-end read is as yet unknown. Thus we traverse the contig adjacency graph, carrying out local assembly to fill the gap between the two ends of every paired-end read in the chain. This allows us to recover the exact sequence appearing between adjacent contigs.

At this point in the process, we have a list of contig pairs identified to be adjacent, and the sequence between them assembled using local assembly. Ideally, a contig should only have two other contigs adjacent to it on the genome — one to the left, and one to the right. However if a repeat region is repeated identically, 9bp overlap chains and assembly can still report more than 2 contigs as adjacent to a given contig. In these situations, we use heuristics to rank the *adjacencies* (adjacent contig pairs) and report the highest ranked ones.

Thus we output a list of longer contigs generated by linking adjacent input contigs. Adding the input contigs which did not participate in any adjacencies gives us the final output.

Each step is described in detail in the following sections.

6.2 PCR Duplicate Removal

Two reads are considered PCR duplicates if the corresponding ends have the same sequence. The read lengths need not be the same. In Figure 6.1, read a and read b are PCR duplicates. The left end of read a is (X+P)bp long, and the left end of read b is Xbp long. The first Xbp in these two sequences is the same. Similarly, the right end of read a is Ybp long and the right end of read b is (Q+Y)bp long. The last Ybp in these two sequences are the same.

Once a set of reads is found to be PCR duplicates of each other, we merge them and output the consensus sequence. Thus the PCR duplicate removal algorithm proceeds as follows.

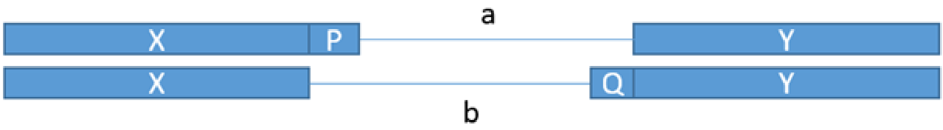


Figure 6.1: Read a and read b are PCR duplicates. Although the corresponding ends are not of the same length, the sequences match up to the available length.

Algorithm 1: Function *form_clusters*

```
input  : list of sequences
output: list of clusters such that sequences in the same cluster are similar
        to each other
begin
    Initialize empty list of clusters for every sequence in input list do
        Compare the sequence with first sequence in every cluster;
        if this is similar to any cluster then
            Add to that cluster;
        else
            Create new cluster with this sequence;
        end
    end
end
```

Algorithm 2: Function *merge_duplicates_pass*1

```
input  : list of 5‘ end reads; list of 3‘ end reads
output: list of 5‘ end reads and corresponding 3‘ end reads after merging
        PCR duplicates
begin
    Hash first 9bp of all reads. Group together all reads with same sequence
    in first 9bp for each group do
        Call form_cluster for reads with same sequence in first 9bp // this
            will result in clusters where one end of the
            paired-end read are similar to each other
        for each one-end cluster do
            Call form_cluster on other end // if the read in the
                original cluster is the 5‘ end, use the 3‘ end here
                and vice-versa.  Now the reads in one cluster are
                PCR duplicates
            Output consensus sequence for both ends as the merged read
        end
    end
end
```

N504	No. of reads (2 x 100)	Coverage
After sequencing	819,798	35.35x
After PCR duplicate removal	809,714	34.91x

Table 6.1: Statistics for PCR duplicate removal step for the N504 library.

Function $merge\_duplicates\_pass1$ is called with all the reads generated by sequencing as the input. This function groups together all reads which have exactly the same sequence (no mismatch) in the first 9bp. Thus many clusters are formed, with each cluster consisting of reads with exactly the same sequence in the first 9bp. The algorithm then checks the sequence in the rest of the read. If the sequences are the same (ungapped alignment, up to 4% mismatch), the reads in the cluster are considered duplicates of each other, and a single merged read is generated to represent the entire cluster. In this manner, one pass of duplicate removal is carried out. In the second pass, the reads generated by $merge\_duplicates\_pass1$ are considered. In function $merge\_duplicates\_pass2$, we form the initial clusters by considering the second 9bp (bases 10 to 18) of the same end of the read that was used to form the initial clusters in $merge\_duplicates\_pass1$. That is, reads with the same sequence (no mismatch) in bases 10 to 18 are grouped together. Further filtering and merging proceeds in the same manner as in $merge\_duplicates\_pass1$. This handles the case where two reads are really PCR duplicates, but a sequencing error in the first 9bp causes them to be placed in different clusters by $merge\_duplicates\_pass1$. In this work, the similarity measure used was hamming distance.

Table 6.1 shows the statistics after this step for the N504 library. It can be seen that there were only 1.2% PCR duplicates.

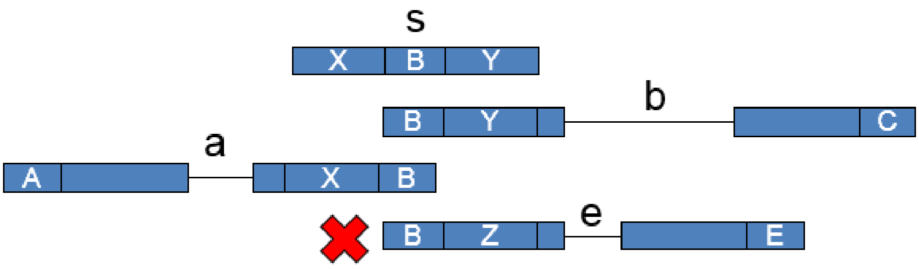


Figure 6.2: A paired-end overlap between a and b is considered to be true only if at least 1 supporting read s covers the overlap and has at least 10bp on either side of the overlap. Thus X and Y are ≥ 10bp long. A mismatch of up to 3bp is allowed.

6.3 Constructing Overlaps Graph

To construct the overlaps graph, we need to find end-to-end 9bp overlaps among paired-end reads. When working de novo, 15,573,574 end-to-end 9bp overlaps were detected. However only 452,987 read pairs were mapped next to each other on the reference genome with a 9bp overlap. To filter the overlaps detected de novo, we consider the overlap to be true only if there is another read supporting the overlap. That is, given two paired-end reads with an end-to-end overlap, we look for a single-end read that maps onto the overlapping ends in such a way that it covers the overlap. Such a supporting read indicates that the sequence resulting from merging the overlapping ends of the paired-end reads actually exists on the genome.

This is illustrated in Figure 6.2, where an end-to-end 9bp overlap exists between reads a and b, and also between reads a and e. A supporting read s exists for the overlap between a and b. That is, s covers the overlap B, the sequence X (from read a) to the left of B, and the sequence Y (from read b) to the right of B. X and Y must be at least 10bp long, and up to 3 mismatches are allowed. However, no read supports the overlap between a and e. Thus (a, b) is a supported overlap, while (a, e) is not. Note that X+B+Y should be the entire read s. That is, the entire sequence of read s must support the overlap.

As can be seen from the Figures 6.3, 6.4, 9bp is the most frequent overlap length with support. After filtering, 75% of the overlaps retained by our method

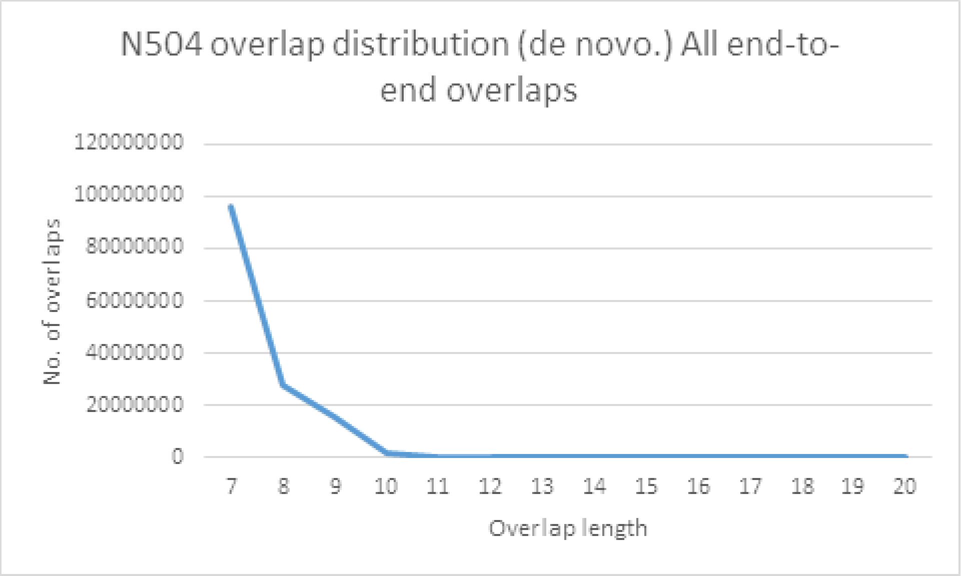


Figure 6.3: Potential overlaps distribution. All end-to-end overlaps detected de novo do not follow the expected pattern

No. of supported 9bp overlaps	581,632
No. of vertices in supported overlaps	520,017
Total number of reads	809,714
Max degree	89

Table 6.2: Statistics on overlaps graph for N504 library.

were found to be true with respect to the reference genome. 3% true positives were missed.

In the overlaps graph, every paired-end read is considered a vertex, and an overlap is considered an edge. Thus in Figure 6.5, a, b and c are vertices, and the edges are (a, b), and (a, c). All edges in the overlaps graph are undirected. Read s is the read which supports both the overlaps.

Table 6.2 shows the details of the overlaps graph in the N504 library. As can be seen from the table, only 62% of the reads participate in supported overlaps. This could be caused by the loss of some information during the sequencing process. It could also be caused by sequencing errors in the overlapping 9bp.

Under ideal conditions, two paired-end reads involved in an end-to-end 9bp

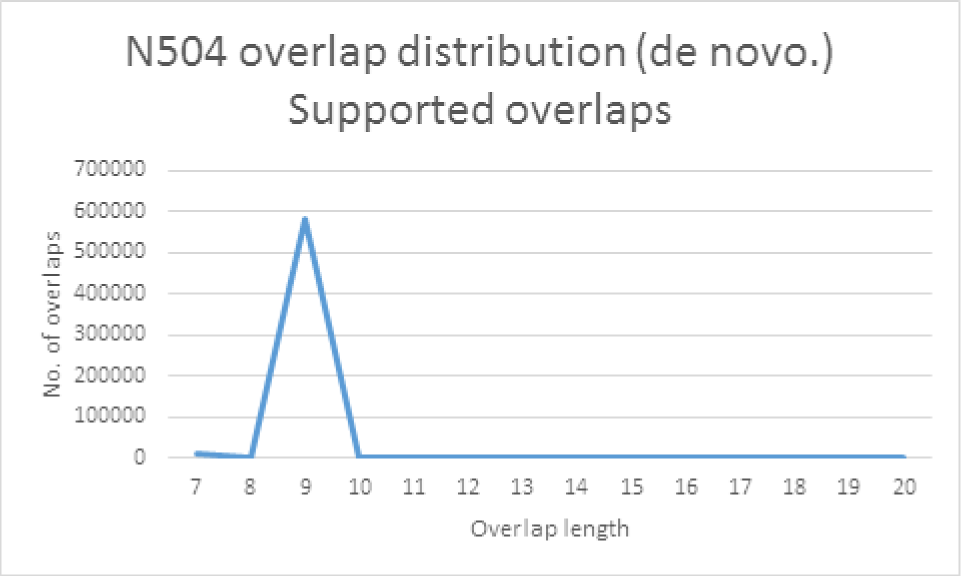


Figure 6.4: Supported overlaps. After filtering potential overlaps based on supporting reads, 9bp is the most frequent overlap length.

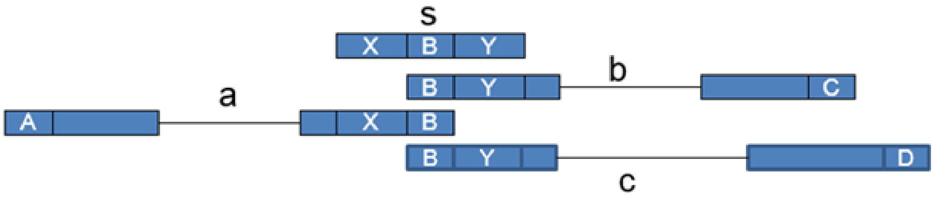


Figure 6.5: Paired-end read a has end-to-end overlaps with both read b and read c. Read s supports both overlaps. Thus the overlaps graph has vertices a, b and c. Edges are (a, b) and (a, c). All edges in the overlaps graph are undirected.

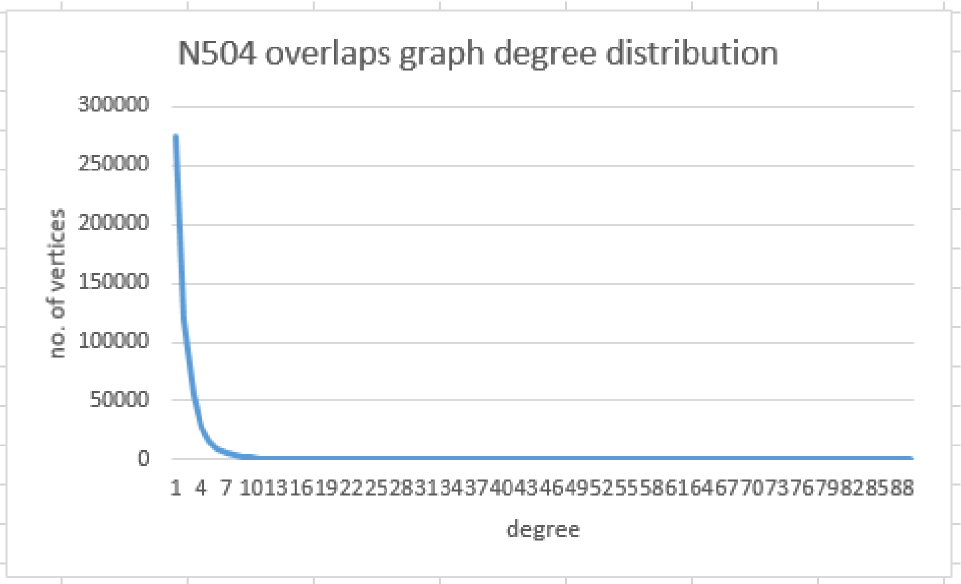


Figure 6.6: Degree distribution in overlaps graph for the N504 library. ˜92% of the vertices have degree ≤ 4.

overlap are actually next to each other on the genome, and come from the same molecule in the sample. This is only true if we assume the following:

- every cut-site is unique, and no two molecules are cut at the same place
- 9bp is long enough to ensure that the overlapping sequence at every cut-site occurs only once in the genome
- there are no errors in sequencing

If these assumptions were true, every paired-end read would have exactly 1 overlap on its left and 1 overlap on its right. As can be seen in Figure 6.6, most vertices (paired-end reads) have degree 1 or 2 in the overlaps graph. However the maximum degree is 89. Also, ˜92% vertices have degree ≤ 4. Thus we can infer that although the above assumptions are not strictly true, the 9bp overlap is able to constrain the number of choices for the next paired-end read on the genome to ≤ 4 in ˜92% of the cases.

6.4 Finding Anchors

We need to find overlap chains which link one contig to another. To do this, we need paired-end reads which we can be unambiguously associated with only one

	n50	total length	no. of contigs	longest contig	shortest contig
IDBA	76,486	4,489,247	155	254,174	222
After trimming	76,286	4,454,629	133	253,974	266

Table 6.3: Assembly statistics before and after trimming the contigs. After trimming, contigs shorter than the average insert size (250bp) were discarded.

contig. Thus, paired-end reads where both ends map uniquely to one contig are identified as *anchors*. We look for anchors near the ends of every contig so that we can find chains connecting the contigs.

Assemblers sometimes assemble contigs which encroach into the repeat region. However, a paired-end read which maps to the encroaching region is not a reliable anchor. The reason is as follows. If we manage to extend the other contigs flanking the same repeat, the paired-end read we now think maps uniquely will actually be mappable to multiple locations. To handle this, we trim 100bp from the ends of every contig before looking for anchors.

For the N504 dataset, IDBA was chosen as the assembler. After trimming, contigs shorter than the average insert size (250bp) were discarded. The statistics before and after trimming are shown in Table 6.3.

The mapping of paired-end reads onto contigs was carried out using BLAT [9]. Concordant reads where both ends map uniquely to one contig were then selected as anchors. Anchors could be found for 127 contigs.

6.5 Constructing contig adjacency graph

Two contigs are declared potentially adjacent if we can find a 9bp overlap chain linking an anchor on the first contig to an anchor on the second contig. Thus in Figure 6.7 (repeated from Proposed Approach section), paired-end read a is an anchor on contig X, and paired-end read d is an anchor on contig Y. The 9bp overlap chain through reads b and c passes through the repeat region (red)

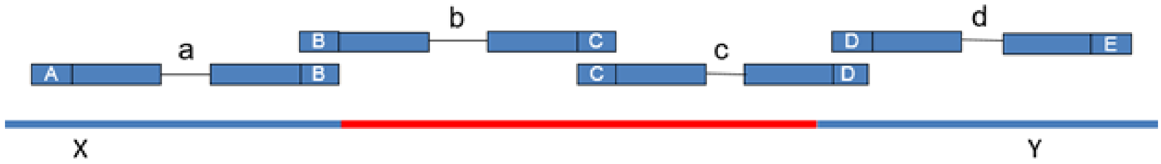


Figure 6.7: Paired-end read a is an anchor on contig X. Paired-end read c is an anchor on contig Y. A 9bp overlap chain leads from a to c through read b. Thus contigs X and Y are declared potentially adjacent.

in between the contigs and links the two anchors. Thus X and Y are declared potentially adjacent.

Any given contig might be potentially adjacent to many other contigs. The collection of chains linking one contig to other contigs forms the contig adjacency graph associated with that contig. We construct one contig adjacency graph per contig.

For the N504 library, 1,292 potential adjacencies were found. These potential adjacencies covered 123 contigs. Thus 123 contig adjacency graphs were generated. When compared against the reference genome, 91 direct adjacencies and 9 transitive adjacencies were found to be real. A transitive adjacency is when two contigs are linked by the overlap chain, but there is another contig in between them on the genome.

One contig out of the 133 contigs generated by IDBA could not be mapped to the MG1655 reference genome [17]. We refer to this as the *non-mapped* contig. Two adjacencies could not be validated against the reference genome as they linked the non-mapped contig to mapped contigs.

6.6 Local Assembly

In the contig adjacency graph, the sequence between the two ends of a paired-end read is unknown. Thus we carry out local assembly to fill the gap. The strategy used for local assembly is adopted from PE-Assembler [2], and uses the overlap-layout-consensus approach with paired-end information.

As we traverse the contig adjacency graph, we carry out local assembly at each gap. The algorithm used is described in Algorithm 3. Each step in the algorithm helps reduce false positives. In the sections below, we describe the scenarios in which each of these steps is helpful

Algorithm 3: Local assembly

```
input  : contig adjacency graphs
output: assembled adjacencies
begin
    Traverse each contig adjacency graph in topological sort order,
    assembling each gap;
    if a branch is encountered then
        assemble all branches;
        if only 1 branch is assembled then
            Proceed with traversal and assembly;
        else
            // > 1 branch is assembled
            if sequences are consistent then
                if sequences are of same length then
                    Merge and proceed;
                else
                    // consistent sequences but different lengths
                    Traverse both branches;
                end
            else
                // inconsistent sequences
                Stop traversal along this branch;
            end
        end
    end
end
```

6.6.1 No Branches

In the ideal case, Tn5 will cut different occurrences of a repeat region at different places. This would result in the situation illustrated in Figure 6.8. Here the repeat region (red) occurs twice on the genome. The first occurrence is flanked by contig X and contig Y. The second occurrence is flanked by contig U and contig V. Thus the correct adjacencies are X to Y, and U to V. Tn5 cuts the first occurrence of the repeat at cut site C. The second occurrence of the repeat is cut at sites G

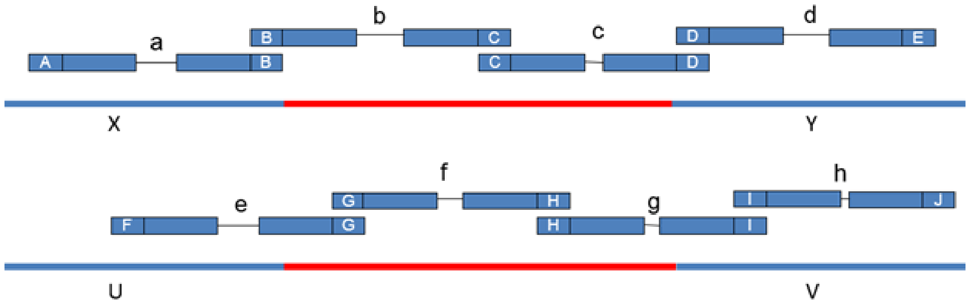


Figure 6.8: Ideal case. Different occurrences of a repeat are cut at different locations. 9bp overlap chains only link the correct contig pairs.

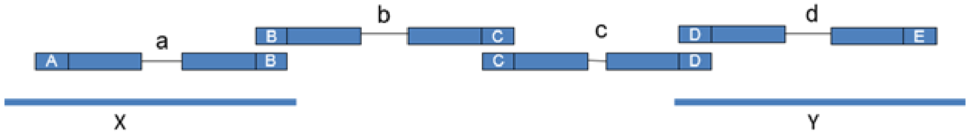


Figure 6.9: Contig adjacency graph associated with contig X when overlaps are is as in Figure 6.8.

and H, both of which are different from C. Thus we are able to find 9bp overlap chains linking only the correct contig pairs, and the contig adjacency graphs have no branches.

The contig adjacency graphs associated with contigs X and U are as illustrated in Figure 6.9 and Figure 6.10. The contig adjacency graphs have no branches, and local assembly is likely to be able to assemble all gaps.

6.6.2 Only One Branch Can be Assembled

The overlapping sequence in Tn5's cuts is 9bp long. However 9bp is not long enough to be unique on the E.coli genome. Thus a scenario as described in Figure 6.11 can occur. Here the same 9bp sequence occurs at two locations on the

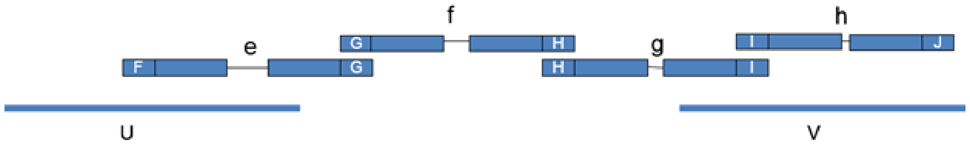


Figure 6.10: Contig adjacency graph associated with contig U when overlaps are is as in Figure 6.8.

genome and Tn5 cuts in both places (cut site C, yellow). Thus read b has degree 2, with edges (b, c) and (b, e). The region on either side of the 9bp is not repeated (first occurrence has a green sequence on the left while second occurrence has a pink sequence. Also, first occurrence has an orange sequence on the right while second occurrence has a grey sequence). Read c overlaps with read d, which is an anchor on contig Y. Also, read e overlaps with read f, which is an anchor on contig Z. Thus we discover two potentially adjacent contig pairs — X to Y and X to Z.

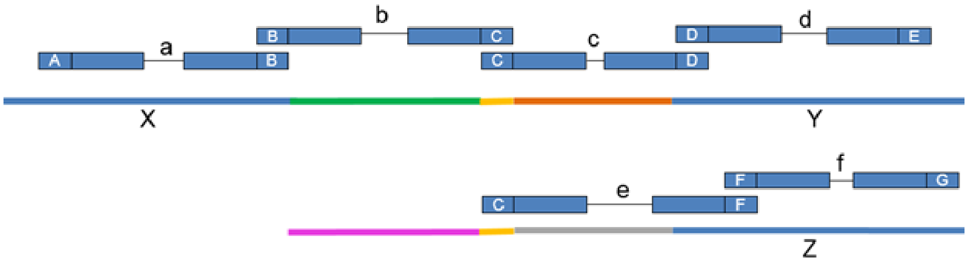


Figure 6.11: Two occurrences of the same 9bp is cut in both places (cut site C, yellow). However the region surrounding the repeated 9bp is different in the two cases. The first occurrence has a green sequence on the left and an orange sequence on the right. The second occurrence has a pink sequence on the left and a grey sequence on the right. Since read c at the first cut site leads to contig Y and read e at the second cut site leads to contig Z, we discover two potentially adjacent contig pairs X to Y and X to Z.

The contig adjacency graph associated with contig X resulting from this situation is shown in Figure 6.12. Here we see the branch at read b. One branch leads to contig Y, while the other branch leads to contig X.

Since green, yellow, orange is the correct sequence, there will be other paired-end reads which will help us assemble the gap in c (Figure 6.13). These reads will not support the assembly of the gap in read e. Thus local assembly will

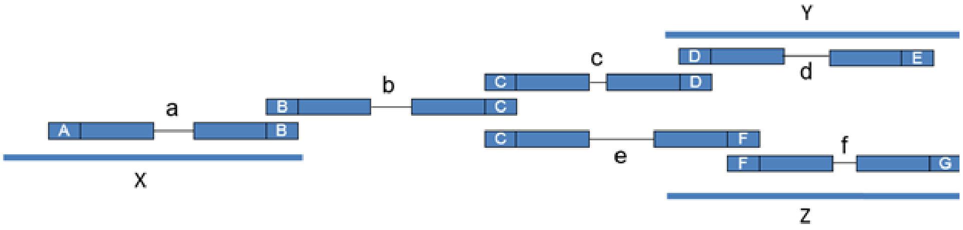


Figure 6.12: Contig adjacency graph associated with contig X when overlaps are is as in Figure 6.11.

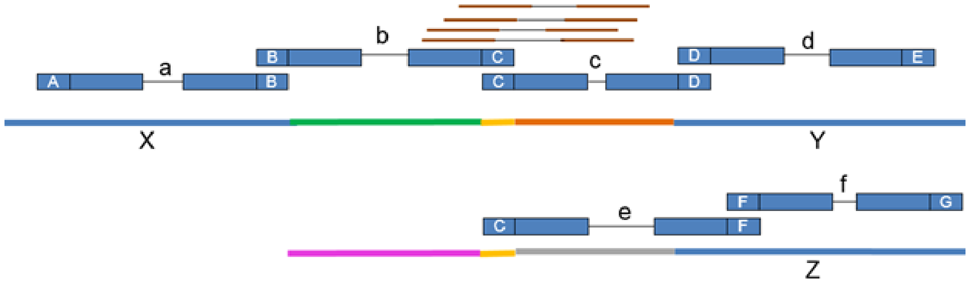


Figure 6.13: Support exists for assembly only for the correct sequence. Here the brown linked lines represent other paired-end reads.

fill the gaps only for the chain leading to the correct adjacency, X to Y. As we can see, assembling the gaps in the overlaps chain not only helps us retrieve the actual sequence occurring between the contigs, but also helps eliminate some false positives.

6.6.3 Both Edges Are Assembled, Sequences Are Inconsistent

Sometimes a sequence longer than one end of a paired-end read but shorter than the insert size is repeated on the genome. In the case of the N504 library, this would mean a repeat of length >100bp and <250bp. This is illustrated in Figure 6.14. The repeated sequence (yellow) has a cut site C in both occurrences. This leads to a branch at read b, with edges (b, c) and (b, e). Read c overlaps with read d, an anchor on contig Y. This leads to the adjacency X to Y. Read e overlaps with read f, an anchor on contig Z. This leads to the adjacency X to Z. However another molecule has cut sites outside the repeated sequence (H).

The contig adjacency graph associated with contig X is iilustrated in Figure 6.15.

This time the assembler might be able to assemble both branches (b, c) and (b, e). That is, the edge (b, c) will be assembled to get the sequence green, yellow, orange. The edge (b, e) will be assembled to get the sequence green, yellow, grey. This is because the repeated region (yellow) is long enough that the paired-end

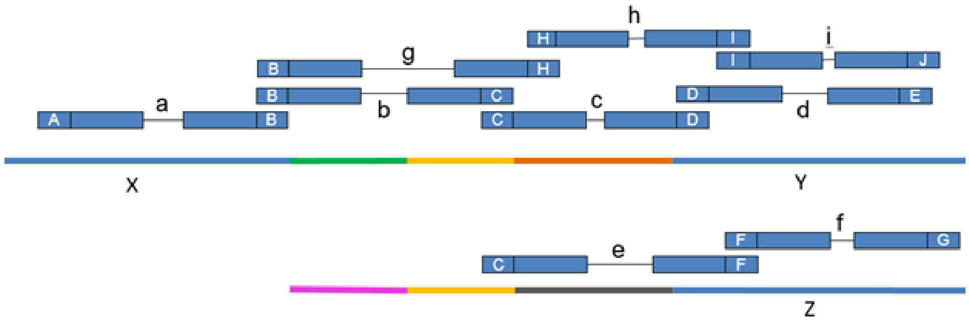


Figure 6.14: The yellow sequence is longer than one end of a paired-end read but shorter than the insert size, and is repeated on the genome. Both occurrences are cut at site C. This leads to a branch at read b, with edges (b, c) and (b, e). The region on either side of the yellow sequence is not repeated. Here the first occurrence has a green sequence on the left while the second occurrence has a pink sequence. Also the first occurrence has an orange sequence on the right while the second sequence has a grey sequence.

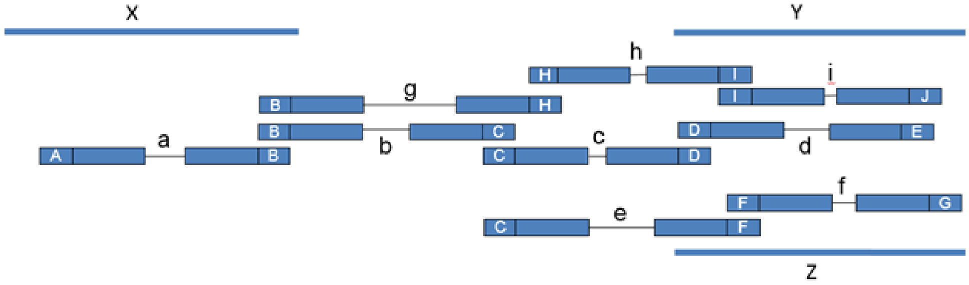


Figure 6.15: Contig adjacency graph associated with contig X when overlaps are as in Figure 6.14.

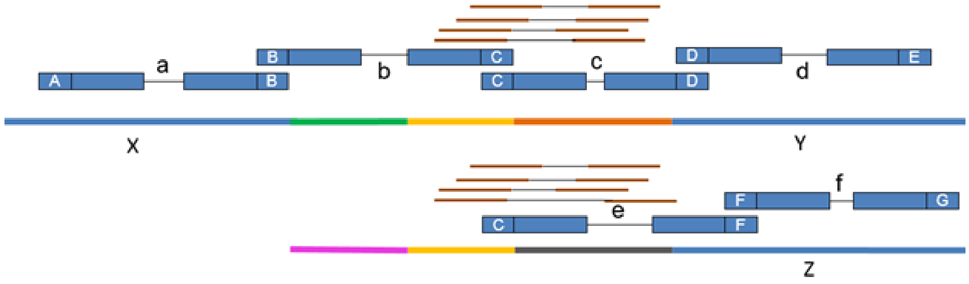


Figure 6.16: Other paired-end reads (brown) can support both branches (b, c) and (b, e). Thus both branches can be assembled.

reads which have one end covering the gap to be assembled have the other end in the repeated sequence (Figure 6.16).

In this case, we can only tell the correct adjacency using the path a->g->h->i, which unambiguously links contig X to contig Y. The chains a->b->c->d and a->b->e->f serve only to confuse the results. Now, we can see that although both branches (b, c) and (b, e) are assembled, the sequences assembled in the two branches are not the same. This is termed an inconsistent branch. Thus when assembly results in an inconsistent branch, we stop traversal along that branch and pursue other paths.

6.6.4 Both edges are assembled, sequence is consistent

In Figure 6.14, Figure 6.15 we see a branch at read a, with edges (a, b) and (a, g). This is caused by Tn5 cutting at the same location on one end (cut-site B), and different locations on the other end. Read b came from the other cut being at site C. Read g came from the other cut being at site H. In this case, the same reads support assembly of both branches, and both branches can be assembled successfully. After assembly the sequences in both branches will have the same prefix. This is termed a consistent branch. Since the lengths are different, both branches are pursued. If the lengths are the same, the two branches are merged. This can happen if the two reads are duplicates but the number of mismatches are more than that allowed by the duplicate removal procedure.

Whenever we succeed in assembling a path from start anchor to end anchor, we store the assembled sequence. After all paths in a contig adjacency graph have been traversed and assembled, there might be multiple sequences assembled between a given pair of contigs. For each such sequence, a portion of the sequence overlaps with the left contig, corresponding to the start anchor. Also, a portion of the sequence overlaps with the right contig, corresponding to the end anchor. We check whether the parts of the sequence overlapping with the contigs are similar to the contigs. If so, the sequence is declared to be consistent with the contigs.

Thus, for a given pair of contigs, we first find the sequences consistent with the contigs. We then group them into clusters such that the sequences in the same cluster are similar to each other. We consider the biggest cluster to represent the correct sequence between the two contigs, and output the consensus sequence.

For the N504 library, 123 contig adjacency graphs were traversed and local assembly was carried out. Out of the 1,292 adjacencies represented in these graphs, 235 adjacencies could be assembled. The 235 assembled adjacencies involved 90 contigs. When compared against the reference genome, 69 of the assembled adjacencies were found to be real. 65 contigs were involved in the real adjacencies. Two adjacencies could not be validated against the reference genome as they linked the non-mapped contig to mapped contigs.

6.7 Filtering Using Split Reads

Several false positive adjacencies are retained even after local assembly is carried out. To filter these, we first identify the positions where the contig ends and the assembled sequence begins. At each of these positions, we look for split reads. Split reads are of two types. Type1 split reads are paired-end reads with one end mapping across the position of interest, and the other end mapping either to the left or to the right. In Figure 6.17, read a is a Type1 split read with one end on the junction between the contig and the assembled adj, and the other end on the left. Similarly, read b is a Type1 split read with the other end on the right

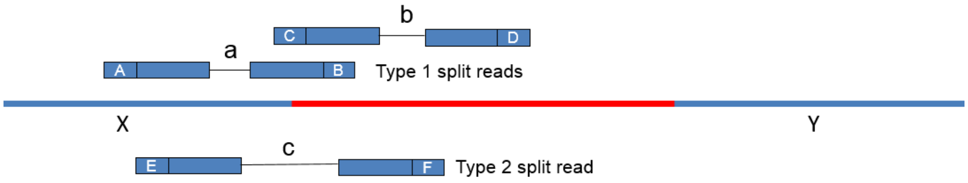


Figure 6.17: Split reads supporting one position. Here the adjacency X to Y has been assembled with the red sequence in between. We look for split reads for the junctions between the contig and the assembled sequence. Here reads a and b are Type1 split reads. Read c is a Type2 split read.

of the position of interest. Type2 split reads are paired-end reads where one end maps to the left of the position of interest, and the other end maps to the right. In Figure 6.17, read c is a Type2 split read.

Split reads cannot exist if the assembled sequence is wrong. Also, if the contig is really from the non-repeat part of the genome, there should be exactly one split read of either type from every molecule that was sequenced. On the other hand if the contig encroaches into the repeat region, the same read can be mapped to multiple assembled adjacencies. Thus we set the number of molecules in the input (50) as the threshold for the number of split reads expected. If an adjacency does not have $\geq$ 50 split reads for both junctions, the adjacency is filtered out.

For the N504 library, 73 adjacencies were retained after filtering using split reads. Of these, 32 could be validated with respect to the reference genome.

6.8 Merging Sub-sequences

After assembly, it is possible that some of the assembled adjacencies are sub-sequences of others. This is illustrated in Figure 6.18. Here contig U lies between contigs X and Y on the genome. We have been able to assemble all 3 adjacencies — X to Y, X to U and X to Y.

After all sub-sequences starting from contig X have been detected and merged, the shorter adjacencies are discarded from the list. In this case, adjacencies X to U and U to Y are discarded.

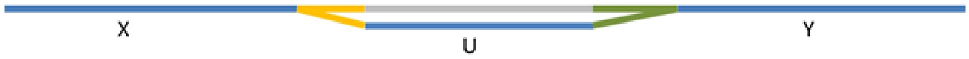


Figure 6.18: Merging sub-sequences. We discover that the sequence after assembly of X to U is a sub-sequence of X to Y. We look for and discover an assembled sequence from U to Y. If the sequence X to U and U to Y is similar to the sequence X to Y, we merge the adjacencies.

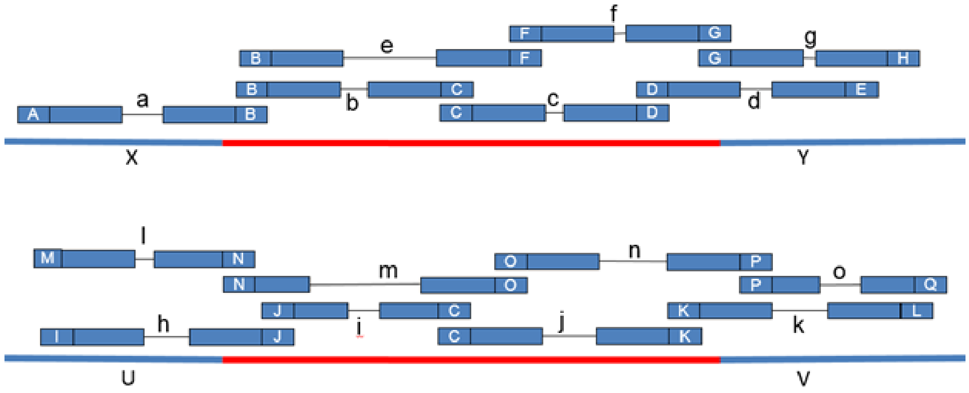


Figure 6.19: A repeat region (red) occurs in two places on the genome. The first occurrence is cut at sites {B, C, F, D, G}. The second occurrence is cut at sites {N, J, C, O, K}. C is a common cut site.

For the N504 library, 6 adjacencies were discarded after merging subsequences.

6.9 Ranking

Even after local assembly, there can be cases where one contig is declared potentially adjacent to more than 1 contig in the same direction. This can be explained by Figure 6.19. Here a repeat region (red) occurs in two places on the reference genome. The first occurrence is flanked by contig X and contig Y, and the other occurrence is flanked by contig U and contig V. Both occurrences of the repeat have a cut at site C (i.e the 9bp overlap resulting from the cut is C). Thus the branch (b, c) leads to the correct adjacency X to Y, and the branch (b, j) leads to the wrong adjacency X to V. As a result, contig X is declared adjacent to 2 contigs on its right. Similarly, the branch (i, j) leads to the correct adjacency U to V, while the branch (i, c) leads to the wrong adjacency U to Y. Contig U is declared adjacent to 2 contigs on its right.

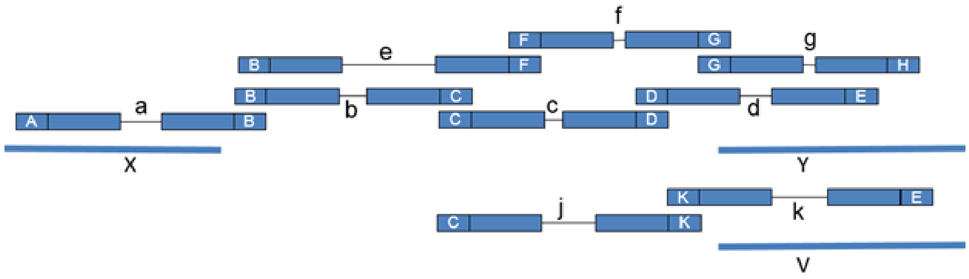


Figure 6.20: Contig adjacency graph associated with contig X when overlaps are as in Figure 6.19.

Contig pair (adjacency)	Start anchor	End anchor	Path	Assembly likely to succeed?	Real adjacency
X to Y	a	d	a, b, c, d	Y	Y
X to Y	a	g	a, e, f, g	Y	Y
X to V	a	k	a, b, j, k	Y	N

Table 6.4: Paths traversed and assembled starting from contig X. Contig adjacency graph is as in Figure 6.20. All paths are likely to be assembled correctly

The contig adjacency graph associated with contig X is shown in Figure 6.20. As we can see, the common cut site C causes us to discover the wrong adjacency X to V. Since the repeat is long and repeated identically, both branches (b, c) and (b, j) can be assembled.

In Table 6.4, we see the paths traversed and assembled starting from contig X.

When we consider the reads used in the paths, we can count the number of reads unique to each adjacency. This is demonstrated in Table 6.5. Here the adjacency X to Y has 5 unique reads, approximately representative of the cut sites unique to occurrence 1 of the repeat.

Thus the adjacency X to Y can be said to have more support than X to V, and we can rank X to Y higher. This helps us identify the correct adjacency. The corresponding details for contig U are shown below. Figure 6.21 shows the contig adjacency graph associated with contig U.

Tables 6.6 and 6.7 show that the same criterion can be used to rank the

Contig pair (adjacency)	Reads used	Unique reads	No. of unique reads	Real adjacency
X to Y	a, b, c, d, e, f, g	c, d, e, f, g	5	Y
X to V	a, b, j, k	j, k	2	N

Table 6.5: Reads used in the paths can be used to get the reads unique to each adjacency. This gives us information about cut sites which are unique to each occurrence of the repeat, allowing us to identify the correct adjacency.

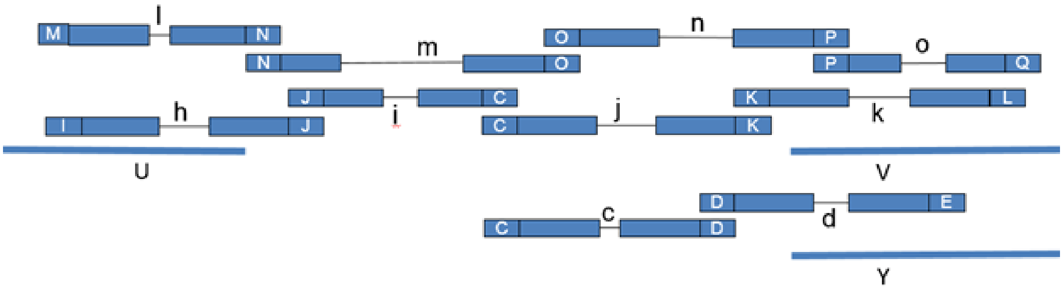


Figure 6.21: Contig adjacency graph associated with contig U when overlaps are as in Figure 6.19.

adjacency U to V higher than U to Y, allowing us to identify the correct adjacency.

Thus we can see that as long as there is at least 1 molecule in the sample where the different occurrences of a repeat are cut at distinct cut sites, we are able to rank the correct adjacency higher than the wrong one.

Another feature that is useful for ranking is the number of path clusters between a given pair of contigs. When we discover multiple paths between two contigs, it is possible for the paths to share some common reads. In Fig 16, 17, we can see paths a->b->c->d and a->e->f->g between contigs X and Y share one common read, a. Thus there is 1 path cluster between contigs X and Y. In Fig 16, 18, we can see paths h->i->j->k and l->m->n->o between contigs U and V have no reads in common. Thus the number of path clusters is 2. Since a 9bp overlap suggests that the overlapping reads come from the same molecule, having multiple path clusters indicates that we have been able to recover multiple molecules after assembly. Thus we are more confident that an adjacency with a larger number of

Contig pair (adjacency)	Start anchor	End anchor	Path	Assembly likely to succeed?	Real adjacency
U to V	h	k	h, i, j, k	Y	Y
U to V	l	o	l, m, n, o	Y	Y
U to V	h	d	h, i, c, d	Y	N

Table 6.6: Paths traversed and assembled starting from contig U. Contig adjacency graph is as in Figure 6.21. All paths are likely to be assembled correctly

Contig pair (adjacency)	Reads used	Unique reads	No. of unique reads	Real adjacency
U to V	h, i, j, k, l, m, n, o	j, k, l, m, n, o	5	Y
U to V	h, i, c, d	c, d	2	N

Table 6.7: Reads used in the paths can be used to get the reads unique to each adjacency. This gives us information about cut sites which are unique to each occurrence of the repeat, allowing us to identify the correct adjacency.

assembled path clusters is correct.

Thus if a contig is declared adjacent to more than 1 contig in the same direction (degree >1), we use the number of unique reads to select the top 2 ranking candidates. Among these two, we choose the candidate with more path clusters as the real adjacency.

After ranking is carried out to ensure every contig has at most 1 contig on each side, we have the final list of adjacencies that will be reported by the program. Before printing the output, we carry out a global ranking among all the adjacencies using the number of path clusters (more path clusters imply higher rank). This is done to determine the confidence level in the reported adjacency. We also add the input contigs which did not participate in any adjacency to the output list.

For the N504 library, 26 adjacencies were generated after ranking. Of these, 57% were validated by comparing to the reference genome. Another 7 adjacencies were tested, out of which 4 were validated using biological validation. This lends support to the in-silico discovery that the sequenced cells were only 96% similar to the reference genome. Further biological validation needs to be carried out to verify whether the rest of the predictions are real. The reported 26 adjacencies improved the n50 of the assembly by 15.5%.

Among the 26 reported adjacencies, 7 were reported as being very high confidence predictions. These predictions had an 85.7% accuracy when compared to the reference genome. However these adjacencies alone were not sufficient to impact the n50.

Chapter 7

Future Work

As part of future efforts, it would be interesting to consider other features which can help rank adjacencies after assembly. When a region is repeated n times on the genome, there are 2n contigs flanking the n occurrences. It has been observed that the adjacencies reported after assembly tend to be from within these 2n contigs. Identifying the cluster of adjacencies that form this bipartite graph (Figure 7.1) can provide useful information.

Another possibility is to cascade decisions taken at one step to other steps. For example, in Figure 7.1, if we rank the adjacency X to Y high with very high confidence, X to Q (and therefore Q to X) can be discarded. Thus the adjacency associated with contig Q will be Q to P by default. This implies that P to Q is

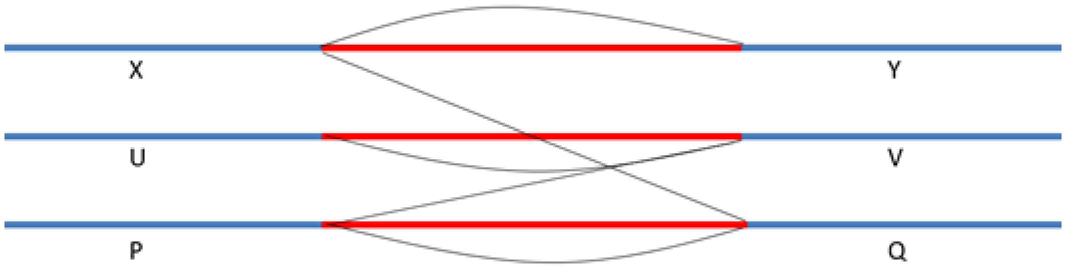


Figure 7.1: A repeat region (red) occurs 3 times on the genome. The correct adjacencies are X to Y, U to V and P to Q. If the occurrences are identical and/or some cut-sites are shared, some false adjacencies may be reported. However building a graph of the reported adjacencies can help get a picture of the various occurrences of the repeat and the contigs flanking it.

also chosen by default, allowing us to discard P to V. It is clear that a correct starting choice can help with future choices enormously. However cascading an incorrect choice can cause us to miss all the correct adjacencies.

In this work, the contigs generated by an existing assembler was accepted as part of the input. Instead, one could map the reads to the closest known reference genome, and use the mapping information to construct contigs. If we only allow unique mapping, repeat regions and structural variations will have no coverage. These gaps can then be linked using the algorithm developed here.

Another approach could be to develop a de novo assembler which exploits the 9bp overlap information at the contig-building step itself. Overlap chains with no branches offer a promising starting template. Local assembly on such chains would give us an initial set of contigs. Once an initial set of contigs has been assembled, direct application of this algorithm can help link and extend to get the final assembly.

Bibliography

[1] S. Andrews. FastQC A Quality Control tool for High Throughput Sequence Data.

[2] Pramila Nuwantha Ariyaratne and Wing-Kin Sung. Pe-assembler: de novo assembler using short paired-end reads. *Bioinformatics*, 27(2):167–174, 2011.

[3] Anton Bankevich, Sergey Nurk, Dmitry Antipov, Alexey A Gurevich, Mikhail Dvorkin, Alexander S Kulikov, Valery M Lesin, Sergey I Nikolenko, Son Pham, Andrey D Prjibelski, et al. Spades: a new genome assembly algorithm and its applications to single-cell sequencing. *Journal of Computational Biology*, 19(5):455–477, 2012.

[4] Iñaki Comas, Sonia Borrell, Andreas Roetzer, Graham Rose, Bijaya Malla, Midori Kato-Maeda, James Galagan, Stefan Niemann, and Sebastien Gagneux. Whole-genome sequencing of rifampicin-resistant mycobacterium tuberculosis strains identifies compensatory mutations in rna polymerase genes. *Nature genetics*, 44(1):106–110, 2012.

[5] Phillip EC Compeau, Pavel A Pevzner, and Glenn Tesler. How to apply de bruijn graphs to genome assembly. *Nature biotechnology*, 29(11):987–991, 2011.

[6] Nicholas Delihas. Impact of small repeat sequences on bacterial genome evolution. *Genome biology and evolution*, 3:959–973, 2011.

[7] Tim Durfee, Richard Nelson, Schuyler Baldwin, Guy Plunkett, Valerie Burland, Bob Mau, Joseph F Petrosino, Xiang Qin, Donna M Muzny, Mulu Ayele, et al. The complete genome sequence of escherichia coli dh10b: insights into the biology of a laboratory workhorse. *Journal of bacteriology*, 190(7):2597–2606, 2008.

[8] Steven R Head, H Kiyomi Komori, Sarah A LaMere, Thomas Whisenant, Filip Van Nieuwerburgh, Daniel R Salomon, and Phillip Ordoukhanian. Library construction for next-generation sequencing: overviews and challenges. *BioTechniques*, 56(2):61–4, 2013.

[9] W James Kent. Blatthe blast-like alignment tool. *Genome research*, 12(4):656–664, 2002.

[10] Jonas Korlach, Keith P Bjornson, Bidhan P Chaudhuri, Ronald L Cicero, Benjamin A Flusberg, Jeremy J Gray, David Holden, Ravi Saxena, Jeffrey Wegener, and Stephen W Turner. Real-time dna sequencing from single polymerase molecules. *Methods in enzymology*, 472:431–455, 2010.

[11] Sanna Koskiniemi, Song Sun, Otto G Berg, and Dan I Andersson. Selection-driven gene loss in bacteria. *PLoS genetics*, 8(6):e1002787, 2012.

[12] Heng Li and Richard Durbin. Fast and accurate short read alignment with burrows–wheeler transform. *Bioinformatics*, 25(14):1754–1760, 2009.

[13] Ruibang Luo, Binghang Liu, Yinlong Xie, Zhenyu Li, Weihua Huang, Jianying Yuan, Guangzhu He, Yanxiang Chen, Qi Pan, Yunjie Liu, et al. Soapdenovo2: an empirically improved memory-efficient short-read de novo assembler. *Gigascience*, 1(1):18, 2012.

[14] Thomas Madden. The blast sequence analysis tool. 2013.

[15] Nicholas J Parkinson, Siarhei Maslau, Ben Ferneyhough, Gang Zhang, Lorna Gregory, David Buck, Jiannis Ragoussis, Chris P Ponting, and Michael D

Fischer. Preparation of high-quality next-generation sequencing libraries from picogram quantities of target dna. *Genome research*, 22(1):125–133, 2012.

[16] Yu Peng, Henry CM Leung, Siu-Ming Yiu, and Francis YL Chin. Idba-ud: a de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth. *Bioinformatics*, 28(11):1420–1428, 2012.

[17] Monica Riley, Takashi Abe, Martha B Arnaud, Mary KB Berlyn, Frederick R Blattner, Roy R Chaudhuri, Jeremy D Glasner, Takashi Horiuchi, Ingrid M Keseler, Takehide Kosuge, et al. Escherichia coli k-12: a cooperatively developed annotation snapshot2005. *Nucleic acids research*, 34(1):1–9, 2006.

[18] Richard J Roberts, Mauricio O Carneiro, and Michael C Schatz. The advantages of smrt sequencing. *Genome Biol*, 14(6):405, 2013.

[19] Patricia Siguier, Edith Gourbeyre, and Mick Chandler. Bacterial insertion sequences: their genomic impact and diversity. *FEMS microbiology reviews*, 2014.

[20] Erwin L van Dijk, Yan Jaszczyszyn, and Claude Thermes. Library preparation methods for next-generation sequencing: Tone down the bias. *Experimental cell research*, 322(1):12–20, 2014.

[21] David Williams, William L Trimble, Meghan Shilts, Folker Meyer, and Howard Ochman. Rapid quantification of sequence repeats to resolve the size, structure and contents of bacterial genomes. *BMC genomics*, 14(1):537, 2013.

[22] Fei Yao, Pramila N Ariyaratne, Axel M Hillmer, Wah Heng Lee, Guoliang Li, Audrey SM Teo, Xing Yi Woo, Zhenshui Zhang, Jieqi P Chen, Wan Ting Poh, et al. Long span dna paired-end-tag (dna-pet) sequencing strategy for the interrogation of genomic structural mutations and fusion-point-guided reconstruction of amplicons. *PloS one*, 7(9):e46152, 2012.

Appendix A

Sl. No.	Common gap start	Common gap end	Gap length	Repeat no.	Repeat length	Total no. of occurrences
1	686847	687952	1105	1	1195	10
2	2063396	2064502	1106	1		
3	2099003	2100075	1072	1		
4	2286081	2287175	1094	1		
5	3362230	3363321	1091	1		
6	19778	20552	774	2	714	6
7	278300	279097	797	2		
8	289622	290430	808	2		
9	3580076	3580691	615	2		
10	380241	381549	1335	3	127	5
11	1465357	1466542	1185	3		

12	2063413	2064482	1069	3		
13	3182839	3184063	1224	3		
14	314366	315515	1149	4	1255	5
15	390776	391922	1146	4		
16	565767	566922	1155	4		
17	223422	228430	5008	5 & 6	1732 & 2342	7
18	2723019	2727988	4969	5 & 6		
19	3419969	3425484	5515	5 & 6		
20	3937908	3943263	5355	5 & 6		
21	4162695	4168253	5558	5 & 6		
22	4205434	4208706	3272	5 & 6		
23	728319	732041	3722	7	3588	2
24	1630023	1634007	3984	8	2440	2
25	1194942	1210147	15205	9 & 10	125 & 367	2
26	3466473	3467878	1405	11	1106	2

Table 1: Gaps common to all 4 assemblers (IDBA, SOAP de novo, SPA-des and PE-Assembler). In most cases, the assemblers collapse all occurrences of the repeat into one occurrence.

Repeat No.	Repeat BLAST result	Remark
1	Escherichia coli str. K-12 substr MG1655 beta-galactosidase (lacZ) gene, complete cds; insertion sequence IS5 transposase (insH) gene, complete cds; and lactose permease (lacY) gene, partial sequence	2 occurrences are fully covered by 2 different assemblers. 3 occurrences are partially covered. This is the transposon repeat whose case study was performed
2	Escherichia coli insertion sequence IS30B, complete sequence; insertion sequence IS1B InsA (insA) and InsB (insB) genes, complete cds; and unknown genes	
3	E. coli galE gene with inserted IS2 element	
4	E.coli insertion sequence IS3	

5 & 6	Both repeats give the result ”E. coli ribosomal operon rrnB encoding the 16S ribosomal RNA. Also transfer RNA specific for Glu, 23S ribosomal RNA and two unidentified open reading frames. This sequence was obtained from the transducing phage lambda-rif-d 18 (BAMHI fragment)”	The two repeat blocks are separated by 380bp. The order of the repeats is inverted in two of the occurrences
7	Escherichia coli Rhs core protein and RhsC accessory element-encoded genes, complete cds	
8	Escherichia coli C321.deltaA, complete sequence	
9 & 10	The 367bp repeat gives the result ”E. coli K12 DNA fragment for invertible-P region of the excisable element e14”	The two repeat blocks are separated by 751bp. The 125bp repeat gives no special results
11	E.coli str operon with fusA and tufA genes coding for elongation factors G and Tu	

Table 2: BLAST results for the repeat regions. The results show that several of the repeats are caused by transposable elements. Repeat 8 shows that the sample cells have sequences from other strains such as C321.deltaA