GENOME SCAFFOLDING USING POOLED CLONE SEQUENCING

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER ENGINEERING

By

Elif Dal

December, 2014

GENOME SCAFFOLDING USING POOLED CLONE SEQUENCING
By Elif Dal
December, 2014

We certify that I have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Can Alkan(Advisor)

Assist. Prof. Öznur Taştan

Assist. Prof. Özgür Şahin

Approved for the Graduate School of Engineering and Science:

Prof. Levent Onural
Director of the Graduate School

ABSTRACT

GENOME SCAFFOLDING USING POOLED CLONE SEQUENCING

Elif Dal
M.S. in Computer Engineering
Advisor: Assist. Prof. Can Alkan
December, 2014

The DNA sequencing technologies hold great promise in generating information that will guide scientists to learn more about how the genome affects human health, organismal evolution, and genetic relationships between individuals of the same species. The process of generating raw genome sequence data becomes cheaper, faster, but more error prone. Assembly of such data into high-quality, finished genome sequences remains challenging. Many genome assembly tools are available, but they differ in terms of their performance, and in their final output. More importantly, it remains largely unclear how to best assess the quality of assembled genome sequences.

In this thesis, we evaluated the accuracies of several genome scaffolding algorithms using two different types of data generated from the genome of the same human individual: i) whole genome shotgun sequencing (WGS), and ii) pooled clone sequencing (PCS). We observed that, it is possible to obtain less number of scaffolds with longer total assemble length if PCS data is used, compared to using only WGS data. However, the current scaffolding algorithms are developed only for WGS, and PCS-aware scaffolding algorithms remain an open problem.

Keywords: genome assembly and scaffolding, high throughput sequencing, pooled clone sequencing.

ÖZET

HAVUZLANMIŞ KLON DİZİLEME İLE GENOM İSKELELEME

Elif Dal
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Danışmanı: Yrd. Doç. Dr. Can Alkan
Aralık, 2014

DNA dizileme teknolojileri genomların insan hayatını nasıl etkilediği, organizma evrimini ve aynı türler arasındaki genetik ilişki bilgilerini oluşturarak bilim adamlarını yönlendirecek büyük umutlar vaat ediyor. Günümüzde genom dizileme verilerini elde etmek eski teknolojilere göre daha hızlı ve ucuz olmasına rağmen henüz dizileme hataları aşılamamıştır. Bu verilerin birleştirilerek yüksek kalitede bitmiş genom dizilimi elde etmek zorlu bir süreçtir. Günümüzde birçok genom birleştirme algoritmaları mevcuttur fakat performans ve çıktıları açısından farklıdırlar. Daha da önemlisi, birleştirilmiş genom dizilimlerinin kalitesini değerlendirmek büyük ölçüde belirsizdir.

Bu tezde, aynı insan genomundan oluşturulmuş iki farklı tipteki verileri kullanarak bir takım genom iskeleme algoritmalarının hassasiyetini inceledik: (i) genom saçma dizileme (WGS), (ii) havuzlanmş klon dizileme (PCS). Eğer PCS verisi kullanılırsa WGS verisine göre toplam birleştirme uzunluğu daha fazla ve daha az iskele sayısı elde etmenin mümkün olduğunu gözlemledik. Fakat şu anki iskeleme algoritmaları sadece WGS için geliştirilmiştir ve PCS ile iskeleme algoritmaları hala çözülmemiş bir problemdir.

Anahtar sözcükler: genom birleştirme ve iskeleme, yüksek hacimli dizileme, havuzlanmış klon dizileme.

Acknowledgement

First and foremost I offer my sincerest gratitude to my supervisor, Assist. Prof. Can Alkan, who has supported me throughout my thesis with his patience and knowledge. I attribute the level of my Masters degree to his encouragement and effort and without him this thesis would not have been completed or written. One simply could not wish for a better or friendlier supervisor.

Besides my advisor, I would like to thank the rest of my thesis committee: Öznur Taştan and Özgür Şahin for giving their precious time to read and review this thesis.

I would like to thank Scientific and Technical Research Council of Turkey (TUBITAK) for their financial support for this study and MS thesis.

Comments given by Volkan Yazar have been great help in my thesis study.

I am specially thankful to Muhsin Can Orhan, Fatma Balcı, Emir Gülümser, Marzieh Eslami Rasekh and Can Telkenarolu for helping out in one way or another and shearing great time during my master study.

I would like to thank my family (Mehmet, Feriha Burcu Dal, Fatma Türkoğlu and Cemal Yekbaşlı for their continued support. My academic growth has always been supplemented by their wisdom. Their profound love, tremendous support and motivation led me to where I am today. Finally, this thesis is dedicated in memory of my grandmother Fatma Türkoğlu and my dearest family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Completion of the Human Genome Project (HGP) was one of the greatest achievements in all life sciences research. The HGP was started in 1990, and by the year 2000, thanks to the innovations in automated genome sequencing technologies, approximately 85% of the human genome was completed. Today, >97% of the human genome is finished and released as the human reference genome (version GRCh38.p1, October 14, 2014). The HGP has allowed researchers to learn functions of genes and effects of their mutations, and this knowledge will bring an important progress in the fields of medicine and other life sciences.

First genome assembly algorithm was designed in the early 1980s, followed with the development of many different assemblers that make use of different methodologies. With the help of emerging technologies, more powerful computers and the massively parallel "next-generation" sequencing (NGS), scientists are now able to read and assemble genomes faster than ever before.

The assembly process is much like assembling a jigsaw puzzle, trying to find the original places of each puzzle piece with checking each piece next to each other, to see if they fit together. Computationally, it is similar to the *shortest superstring problem*, known to be NP-Complete [7], where approximation algorithms still need to perform billions of suffix-to-prefix comparisons, even with assuming short sequences are error-free. When sequencing errors are considered in genome assembly, each piece of DNA fragments is sequenced several times so the problem becomes much more pronounced.

Creating a digital representation of a genome can be better understood in three main steps: First, the genome (collection of chromosomes) is fragmented into pieces in lab environments, then sequenced using NGS technologies. The sequence raw data is digitized and represented in short strings generated from the DNA alphabet $\Sigma = \{A, C, G, T\}$, which makes it possible to analyze using computers (i.e. *in silico*). Second, the billions of short reads are evaluated to be assembled together to reconstruct the original genome sequence. In this step, contiguous segments (termed "contigs") are obtained using an assembly algorithm. Contigs are long sequences without any information about their order and orientation in the genome. To enhance the assembly to include relative order and orientation of these contigs, scaffolding algorithms are used, to generate genome scaffolds. Finally, genome finishing process begins, which is the most costly part of the constructing a genome reference.

1.1 Motivation

Although most of the human genome is assembled, there is still room for improvements in the human reference genome [8]. Having the correct sequence is exceedingly important because any mistake may change any interpretation of genetic diseases. Scientists need to be sure of the correctness and comprehensiveness of the assembly.

Efforts of assembling large and complex genomes, such as human, gibbon, pine, etc., the assembly always is fragmented into variably sized hundreds of thousands of contigs. This is because of several factors: complexity of the genome (i.e. repeat and duplication content), errors imposed by the sequencing methodology and depth of sequencing coverage. The human reference genome is largely constructed using the Sanger sequencing technology, in a hierarchical manner. Sanger technology is able to generate long reads (700-1000 base pairs) to be sequenced with a very low error rate. However, it is also very costly: the HGP cost over 3 billion dollars to complete. Newer sequencing technologies, commonly referred to as “high throughput sequencing” (HTS), or “next-generation sequencing” (NGS) were first realized in 2005 [9] that evolved very rapidly since. Although most widely used NGS technologies produce short reads (100-150 base pairs) with a higher error rate (0.1-to-1%), the associated costs are substantially less, and they are able to generate billions of reads in a single run. This enables these technologies to provide data at high redundancy, measured as *depth of coverage*, which in turn makes it possible to ameliorate the effect of sequencing errors.

Alongside of difficulties related to technologies such as sequencing errors or high memory usage, type of the genome in interest is very important when considering assembly process. There are two main strategies for genome sequencing and assembly: Whole Genome Shotgun (WGS) and Hierarchical Sequencing. WGS approach breaks all genome into fragments and after they are sequenced, the genome is build up from its billions of reads which is a computationally intensive task, and more prone to errors caused by genome complexity. For example, approximately half of the human genome is composed of repeats, which limits the accuracy of WGS approach [10]. Therefore, the HGP used the hierarchical sequencing approach, where the genome is divided into large, ordered segments (BAC clones) first, then those large segments are shared into smaller fragments then sequenced and assembly process is continued with assembling segments in a hierarchical way. This helped the HGP to better resolve repeats in the genome, but this method is more costly and labor intensive.

One of the difficult problem in genome assembly is resolving repeats and ensuring comprehensiveness. In addition, although hierarchical sequencing yields better results than WGS, its higher cost and increased labor make it impractical to be applied to newly sequenced non-human genomes. A newer technology, called pooled clone sequencing [4] aims to merge the cost efficiency of WGS, with repeat-resolving abilities of clone based hierarchical sequencing. In this thesis. we evaluate the efficacy of various genome scaffolding algorithms when pooled clone sequencing data is available, and compare against assemblies generated with WGS-only data. Here we benchmark four different scaffolding tools

(OPERA [11], SCARPA [12], SSPACE [13], BESST [14]), where we assemble the longest and the shortest human chromosomes (1 and 20), and compare with the assembly generated with ALLPATHS [15]. The pooled clone sequencing dataset we used in this study is generated from the genome of the same individual with the WGS data (NA12878), divided into 288 separate pools that were sequenced using the Illumina technology. We provide the details of the methodology in Chapter 3.

1.2 Thesis Organization

The thesis is organized as follows:

◇ Chapter 2 provides background of the biology and structure of genomes. We also explain sequencing technologies, and discuss their strengths and weaknesses. We provide introduction to the genome assembly problem, and explain various strategies.

◇ Chapter 3 presents the data and methodology that are used in this thesis. We explain the pooled clone sequencing approach in detail. We then offer brief descriptions of the scaffolding tools that we benchmark in this study.

◇ Chapter 4 presents our experimental results in terms of both scaffolding each pool hierarchically, and scaffolding them all. Evaluation of the results is presented.

◇ Chapter 5 concludes the study with our experiences throughout the work done, some lessons-learned and future work.

Chapter 2

Background Information

2.1 Biology

2.1.1 DNA

Cells are fundamental working units of every organism. All necessary information to manage these cells are coded in deoxyribonucleic acid (DNA). The information in DNA is stored as a code made up of four chemical bases: adenine (A), guanine (G), cytosine (C), and thymine (T). The DNA sequence is the particular side-by-side arrangement of bases along the DNA strand (e.g. ATGCAGCTATCCGGA). This order is meaningful for instructions that are required to create an organism with its own features.

DNA bases pair up with each other: A forms double hydrogen bonds with T, and C forms triple hydrogen bonds with G, referred to as base pairs [16]. The bases that form bond with each other are called *complements* of each other. A base is also attached to a phosphate and a sugar molecule in the DNA, and these three molecules are all together called a nucleotide. Nucleotides are placed in both long strands of the DNA in a spiral form which is called double helix.

DNA is arranged well for storing biological information. Both strands of the DNA

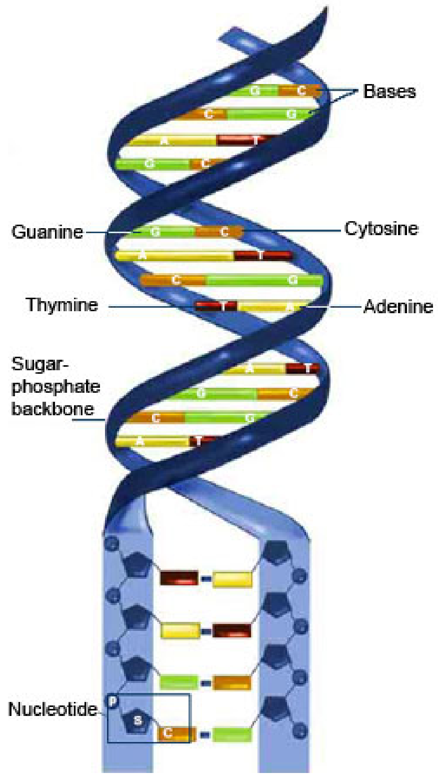


Figure 2.1: DNA structure

store the same biological information. The biological information is encoded in two strands, and these strands are separated during replication. DNA is read always in the same direction; from 5' to 3' ends that refer to the the carbon numbers in the DNA molecule's sugar backbone. This also means that two strands run in opposite (i.e. reverse) directions, therefore one strand is the *reverse complement* of the other.

2.1.2 Genome

Genome is an organism's complete set of DNA. Genomes vary widely in size for different species: A bacterium, which has the smallest genome of a free-living organism contains about 600,000 to 5 million DNA base pairs while mammalian genomes (such as human or mouse) are composed of approximately 3 billion. Except for mature red blood cells and gametes, every single cell in the human body contains a complete copy of the same $\tilde{3}$ billion DNA base pairs, or letters, that make up the human genome.

2.1.3 Gene

DNA in the human genome is arranged into 23 distinct chromosomes. Chromosomes are molecules that contain DNA molecules packed around proteins (called histones) that range in length. Human chromosome length ranges from 45 million (chromosome 22) to 250 million (chromosome 1) base pairs. In the human genome, there are 22 pairs of autosomal chromosomes (chr1 to chr22) and 1 pair of sex chromosomes (chrX+chrX or chrX+chrY). Each chromosome contains many genes. Genes are specific sequences of bases that encode instructions on how to make proteins. In humans, genes vary in size from a few hundred DNA bases to more than 2 million bases. The Human Genome Project (HGP) [17] has estimated that humans have between 20,000 and 25,000 genes [18]. Genes comprise only about 2% of the human genome and the remainder DNA consists of non-coding regions [18].

2.2 DNA Sequencing

DNA sequencing is a technique that is used to determine the nucleotide sequence of the DNA. A DNA sequence contains the most fundamental knowledge of a gene or genome. Instructions for building an organism can be better understood by obtaining the sequence first. DNA sequencing is important for understanding genes, evolution and many unknown questions about living beings.

Sequencing technologies have significantly improved since the first genome was read in 1977 by Sanger. Sanger method is now called as the "First Generation Sequencing", where the latter technologies as Roche 454, Illumina, SOLiD, Ion Torrent, Pacific Biosciences and others are called as Next Generation Sequencing (NGS) technologies, which are mainly divergent because of their sequencing costs

[19, 20]. Today, there are many technologies to sequence a genome that can be used to determine the genome sequence of any species. Although genomes of several species including human, mouse, and many bacterias are sequenced, there are still some incomplete regions in these assemblies, and genomes of many other important species have not yet been sequenced.

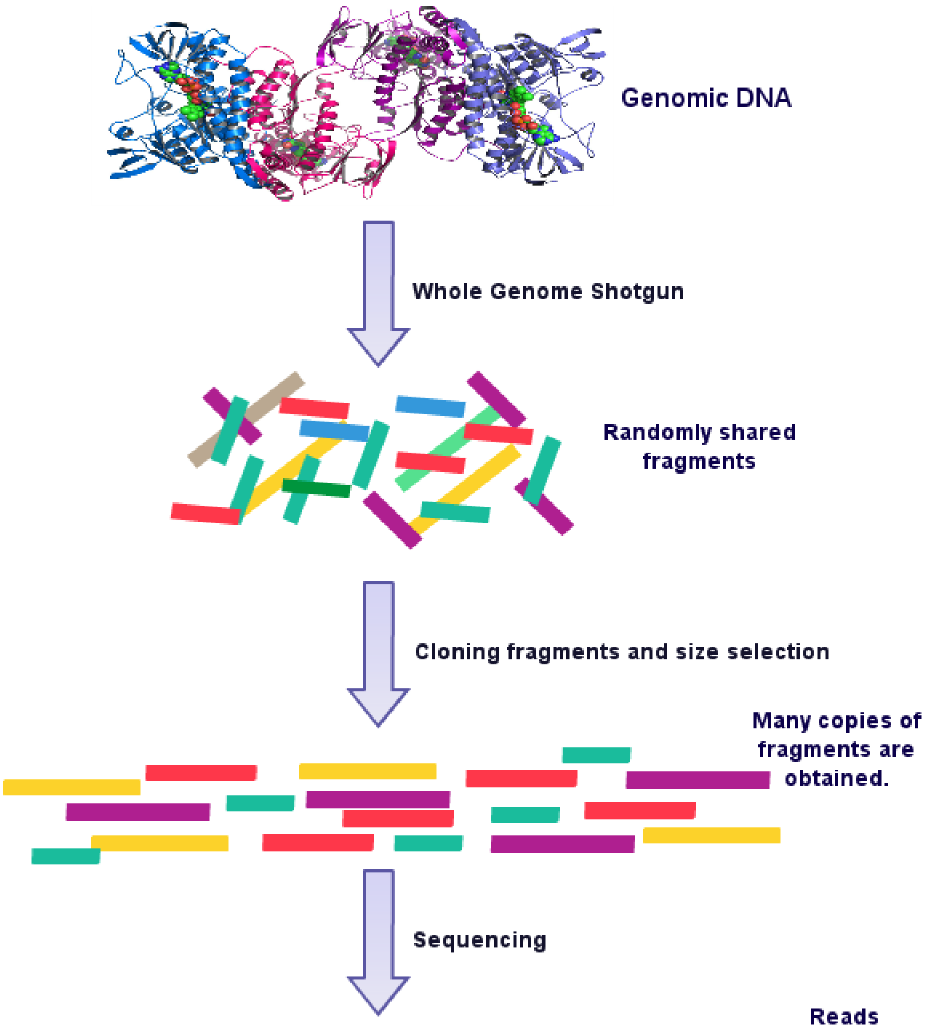
The mostly used technique for DNA sequencing the by whole genome shotgun (WGS) approach. WGS sequencing is done in three main steps. First, DNA is extracted from the cell, then it is sheared into random fragments, and finally fragments are amplified, and DNA sequence of each fragment is read using one of the technologies mentioned above [19]. In the end, a collection of DNA sequences is obtained, and the rest of the work (genome assembly, scaffolding, error correcting) is performed using computers (*in silico*). WGS flowchart is shown in Figure 2.2.

2.2.0.1 Paired-end reads

Each sequence obtained from fragments are called reads. Paired-end reads, also called mate pairs, are produced by sequencing a DNA fragment from both ends. Each sequence obtained from each ends are called reads and their total lengths is usually less than the fragment size because not all the fragments are sequenced, there is a gap which is called insert between two sequenced reads. Insert size differs according to the genome analyses and the methodology used to sequence. Depending on genome being analyzed, optimal insert size should be considered by genome content or genome complexity. Generally, researchers create multiple libraries with different insert sizes in order to get better results. A paired-end read representation is given in Figure 2.3.

2.2.1 First Generation Sequencing Technology

Frederic Sanger developed the first sequencing technique in 1977, called **Sanger sequencing**. In this method, DNA is used as a template to generate a set of


Genomic DNA
Whole Genome Shotgun
Randomly shared fragments
Cloning fragments and size selection
Many copies of fragments are obtained.
Sequencing
Reads

```
@Read1
GTCGTCAGACTGTCGATGAAGCCCTGAAAGACGCGCAGACTCGTATCACCAAGTAATGCTGTGAAATGCCGGATGCGGCGTGAACGCCTTGTCCGGCCTAC
@Read2
GATCCGCGCAGTGCCGATCGCAGAAGCGATGCTGGCGATCGTTTTAATGGATCCCCTGTTACGGCCACGGGCGCAAAATGCCGATGTGAAGACTGATGTTC
@Read3
ACCTTATAGTCACATACCCCAATTGAGAATACCCACATACCCACTGATATTTTCTGTTTGTAGTTTTACCTTTTCCCAAAATGGTATGTGTGTGTGTGTCT
.
```

Figure 2.2: Whole Genome Shotgun Sequencing

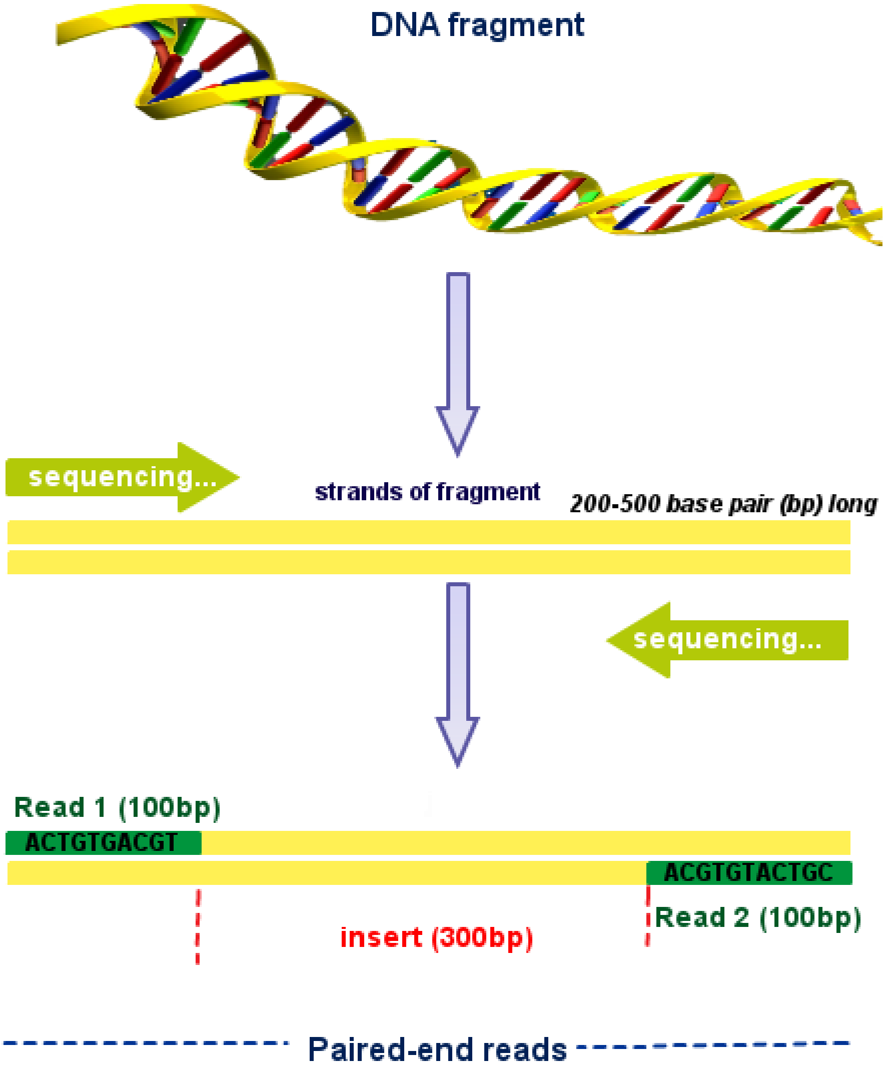


Figure 2.3: Paired-end reads

fragments that differ in length from each other by a single base. The fragments are then separated by size, and the bases at the end are identified, and original sequence of the fragment is obtained.

2.2.1.1 Sanger Sequencing

Sanger Method, is also known as dideoxy chain termination method. First 4 test tubes are labeled with base names as A, T, C and G. DNA is denatured and two strands of the DNA are separated. Single strand formed DNAs are added into each of the test tubes. A primer is added and attached to one of the strands which's 3' end is a dideoxy nucleotide (ddNTP). It is a special kind of nucleotide, used in the Sanger sequencing method. A ddNTP is a kind of nucleotide that is missing the 3-hydroxyl group of its sugar. Because of the structure of DNA, when a ddNTP has been added to a nucleotide chain, any other nucleotide can't be added to a nucleotide chain because of the ddNTP's lack of 3'-hydroxyl group. Therefore, growing chain terminates after a ddNTP is attached to the chain. At this point, last nucleotide, attached to the chain is known because only a specific ddNTP is added to the tube. Most of the time dNTPs (regular nucleotides) are attached to the chain, but whenever a ddNTP is attached the growing chain, it terminates. Many of these reactions are taking place simultaneously in the tubes, but it is random that if any ddNTP's are added to the molecule. All of these reactions produce different length DNA molecules ending with a ddGTP (G), ddATP (A), ddTTP (T) or ddCTP (C). So far, different length of molecules ending with a known bases are obtained. So it is needed to sort these molecules to obtain initial DNA sequence. For this purpose gel electrophoresis is proper to distinguish between DNA molecules of different sizes. Electrophoresis is done and the sequence order is obtained by analyzing the bands in the gel based on the molecular weight. Primer and one of the nucleotides are also recognized by the fluorescent label so the initial DNA molecule sequence can be easily obtained from the gel. In figure 2.4 analysing the sequence from the gel is shown. The Sanger sequencing method is explained in detail [21].

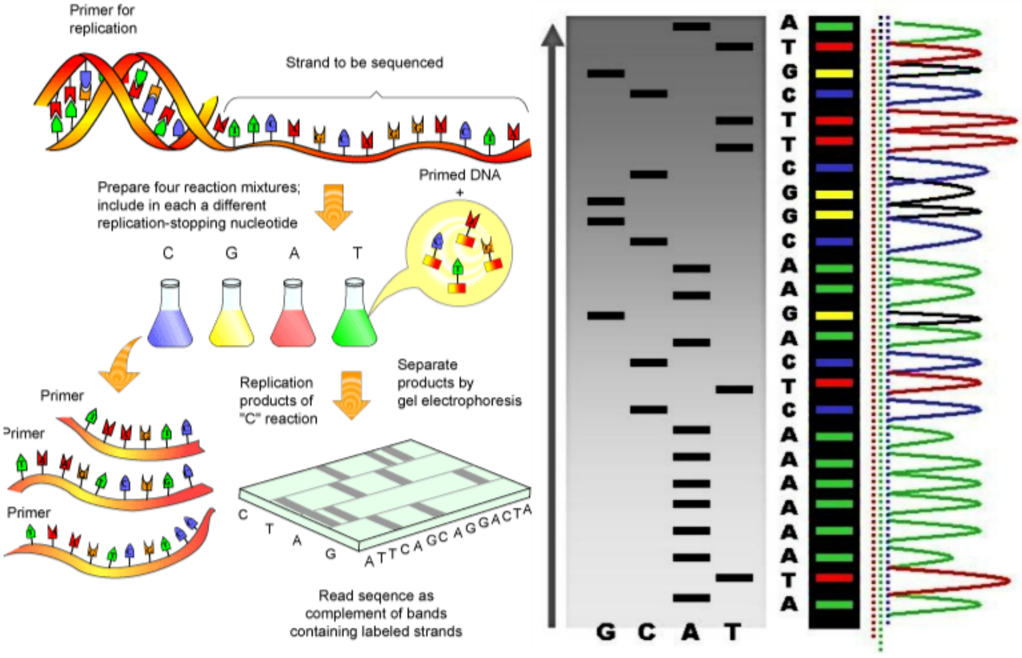

Primer for replication
Strand to be sequenced
Prepare four reaction mixtures; include in each a different replication-stopping nucleotide
Primed DNA
+
C
G
A
T
Primer
Primer
Primer
Replication products of "C" reaction
Separate products by gel electrophoresis
C
T
A
G
ATTCAGCAGGACTA
Read seqence as complement of bands containing labeled strands
G C A T
ATGCTTCGGCAAGACTCAAAAAATA

Figure 2.4: Sanger Sequencing Method

2.2.2 Second Generation Sequencing (Next Generation Sequencing Technology (NGS))

In the beginning of 2005, several NGS technologies were marketed and it brought a new dimension to genomics. It is now possible to sequence a few orders of magnitude more reads in a single run with respect to Sanger sequencing.

The three leading second generation technologies are: **Roche 454**, **Illumina** and **Ion Torrent**. They all have different advantages and disadvantages. Although the technologies have different procedures to sequence DNA fragments, they share the same initial basic steps. DNA is extracted, broken into fragments and the fragments are immobilized to a fixed surface and many sequences are made up in parallel. Thereafter they use their own technologies to decide the order of nucleotides in each sequence. Roche 454 and Illumina produce the sequence as individual nucleotides, where SOLID is using a different procedure that uses 4 colors to represent a change from the previously read nucleotide in a sequence [19].

2.2.2.1 454 Sequencing

The 454 sequencing technology is based on pyrosequencing and uses emulsion-based clonal amplification. 454 sequencing can be undertaken in three steps: First samples are prepared. DNA is broken into 400-600 bp double stranded form of fragments. DNA fragments are attached to special A/B adapters and denatured to single strands. Single stranded DNAs with A and B adapters attached both ends are obtained. In the second step, DNA samples are loaded onto beads. A mixture containing DNA fragments obtained from the first step, beads, PCR reagents and emulsion for reactions is prepared. After sufficient time is passed, reactions are finished and fragment with 100 million of identical copies of it are immobilized on the capture beads. Those beads that hold more than one type of DNA fragment are ready for signal processing. The final step is sequencing the fragments. 454 sequencing process uses sequencing by synthesis approach. In sequencing by synthesis, a single DNA is copied with enzymes and forms double stranded. Starting from one end of the DNA fragment, enzyme adds nucleotides one by one with its matching pair. A plate specific to 454 named PicoTiterPlate is filled with beads on many copies of fragments with their double-helix structures. The four nucleotides A,T,C and G are flowed sequentially on the platform and these nucleotides are incorporated onto the DNA strands and during the incorporation of the nucleotide a light is flashed and during sequencing, these lights are captured [22, 19, 9].

2.2.2.2 Illumina Sequencing

The Illumina sequencing technology is also based on sequencing by synthesis approach like Roche 454 and uses a solid surface for bridging PCR amplication. The reaction occurs on the surface of a flow cell. PCR reaction is performed, and each of the hybridized fragments of DNA are amplified to generate clusters which have the exact copy of the molecule. Then the flow cell is examined under a microscope and when a light flashes the fluorescence, the emission light shows which base was incorporated on each one of those clusters. Although Illumina read length is approximately 100 bases, which is very short, over billion reads are

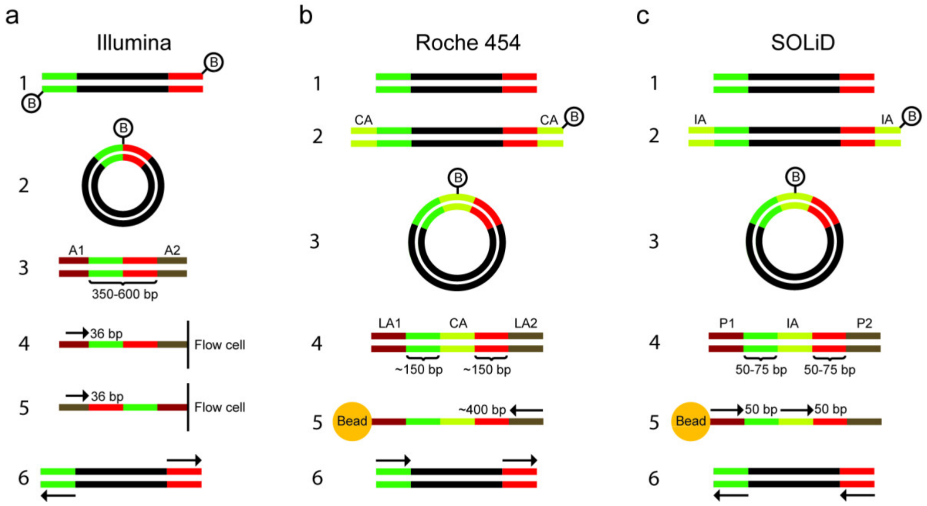


Figure 2.5: Next Generation Sequencing Methods Workflows [1]

generated in a single run with a low cost [19, 23].

2.2.2.3 SOLID sequencing

SOLID sequencing is based on sequencing by ligation and different from Roche454 and Illumina. After preparation either fragment library or mate-pair library, the fragments are attached to magnetic beads and like Illumina and Roche 454, emulsion PCR is performed to amplify the fragments. Sequencing by synthesis is performed by utilizing DNA ligation rather than polymerase. A set of four fluorescently labeled di-base probes compete for ligation to the sequencing primer. Specificity of the di-base probe is achieved by interrogating every 1st and 2nd base in each ligation reaction. Many cycles of ligation and detection are performed where the number of cycles determining the original read length. Then primer is denatured. The platform can include two adjacent primers which decreases error rate. In the end, 25-35 bp length reads are obtained with a 99.99% accuracy and 2-4 GB of DNA sequences are obtained in a single run [22, 24].

2.2.3 Ion Torrent Sequencing

Ion Torrent sequencing, which is also called as Ion semiconductor sequencing, is a method of DNA sequencing based on sequencing by synthesis like Illuminia.

Ion Torrent sequencing detects the protons released as nucleotides are incorporated during synthesis [1]. Shared DNA fragments with specific adapter sequences are linked to and then amplified by PCR. Then, beads are put into proton-sensing wells and sequencing starts from the adapter sequence. While sequencing, each of the four bases is released sequentially. If bases of that type are incorporated, protons are released and a signal is detected related to the amount of incorporated bases [25, 26].

Ion Torrent differentiates from other technologies by it's sequencing protocol. It is based on standard pyrosequencing chemistry, whereby individual bases are introduced one at a time and incorporated by DNA polymerase. Ion Torrent measures protons from the reactions, which makes it relatively inexpensive. Sequencing reactions are relatively fast and error rates are generally not that high (approximately 1%).

2.3 Single Molecule Sequencing

Third generation sequencing (TGS) technologies are able to perform single molecule sequencing without pausing between read steps. It is released in September 2013. The way how NGS and TGS sequence reads separates second and third generation sequencing [27]. The **PacBio RS II** is a Single Molecule, Real-Time DNA Sequencing System that provides the highest consensus accuracy and longest read lengths of any available sequencing technology [28]. It is relatively a new technology and it produces reads with an error rate of 15% but there are error correction tools used to ameliorate the effects of errors.

Different sequencing technologies have different strengths and weaknesses. Roche 454 produces the longer reads than Illumina, but it has also seen that when the

Table 2.1: Characteristics of sequencing technologies [6]

Instrument	Read length	No of Reads	Output (Gb)	Runtime	Advantages	Disadvantages
Sanger	>1000	10^6	0.002	1000h	long reads	long time
Roche45 GS FLX+	700	1x 10^6	0.7	23h	long reads short time	errors expensive
Illumina HiSeq2000	100	3x 10^9	600	11 days	cheap	
SOLiD 5500xl	75	1.5x 10^6	180	14 days	error correction	short reads
Roche 454 GS Junior	400	1x 10^5	0.035	9h	long reads short run	expensive expensive
Illumina MiSeq	150	5x 10^6	1.5	27h	easily used	per base
Pacific Biosciences PacBio RS	>800	1x 10^5	0.1	90min	long reads short run time	high error rate

read length is getting larger, the thoughtput starts to decline errors increase. Illuminia produces high throughput data where 454 Roche and PacBio is low throughput. That is the reason why Illumina is the mostly used sequencing tool among the NGS tools.

2.3.0.0.1 FASTQ Files: After fragments are read, they are stored as reads in formatted text files. The output reads are usually stored in FASTQ files [29]. In FASTQ files, there are four lines for each read: First line begins with a @ character and is followed by a unique sequence identifier. Second line is the raw read sequence consists of only A, T, G, and C letters. Third line begins with a + character and is optionally followed by the same sequence identifier. The fourth line is for quality scores for each letter in the second line. An example of FASTQ file is shown in the Figure 2.6.

```
@NA12878Fospool1pe101C10/1                     @NA12878Fospool1pe101C10/2
GTCGTCAGCCCTGAAAGACGCGCAGACTCGTATCACCAAGCGGA   TATTTTCTTTGCCCTGAAACGCAGGACGCGCAGACTCGTCGCAG
+                                              +
CCCFDFFJGIJIJIJJIJJIGIIIIAGIJJGIJJJJJGFHEHDD   JIJJIJJJGFHEHDDCCCFDJJIGIIIIAGIJJGIJJFFJGIJI
@NA12878Fospool1pe101C11/1                     @NA12878Fospool1pe101C11/2
GTACCAGATGCCAAATTGTAAAGACCATAAAGGCTAGGAAGAGA   GTACATAAAGGACCAGATGCCGCTAGAAATTGTAAAGACAGAGA
+                                              +
@BADDDDFDIIIIEGIIEBCCGIGEGHIGI@HIIIIGHAHIEGF   @GHIGI@HIIIIGHAHIEGFBADDDDFDIIIIEGIIEBCCGIGE
@NA12878Fospool1pe101C12/1                     @NA12878Fospool1pe101C12/2
ATAGTCAAATTGAGAATACCCACATACCCACTGATATTTTCTTT   CTAGAAATTGTCACATACCCACTGATATTTTCTTTATAGTCAAC
+                                              +
CCCFFFIJJJJIJJJJIJJJJJJJJJJJJJJIIIIJHIJIIIIJIJ   CCCJJJJIIIJHIJIFFFIJJJJIJJJJIJJJJJJJJJJJIIJIJ
```

Figure 2.6: Example FASTQ file containing three reads

2.4 Genome Assembly

Once DNA is fragmented into pieces, translated into raw data as reads, data is ready to be assembled. Genome assembly problem aims to reconstruct the whole genome (or chromosome) from sequenced reads.

Genome assembly problem can be better understood with a toy example. Lets assume sequenced reads are GAT, ATT, TTA, TAC, ACA, CAT, CAA. Although read lengths are approximately 100bp, in this example it is given three and assume that the original genome sequence is GATTACATCAA. The problem is how to obtain unknown GATTACATCAA sequence from given reads. It is easily seen that when we order reads like GAT-ATT-TAC-CAT-CAA, output sequence can be obtained, but it is not the only solution. The sequence can also be GAT-TACATTACAA, which is not correct. So optimal solution should be found. This problem is named as shortest substring problem and there are already different algorithms for solving it. Shortest substring finding problem tries to construct a string of minimal length from given a set of S strings, which contains all strings of S as sub-strings. Although genome assembly problem is a super-string problem, finding the minimal length string is not suitable for genome because large genomes contain too much repeated regions so graph based solutions are widely used for assemblers.

Assembling Illumina reads in 100bp length to construct the human genome, which is 3 billion base pair long, is not a trivial task. The biggest problem is handling repeats in the human genome. 50% of human genome is repeated, which means

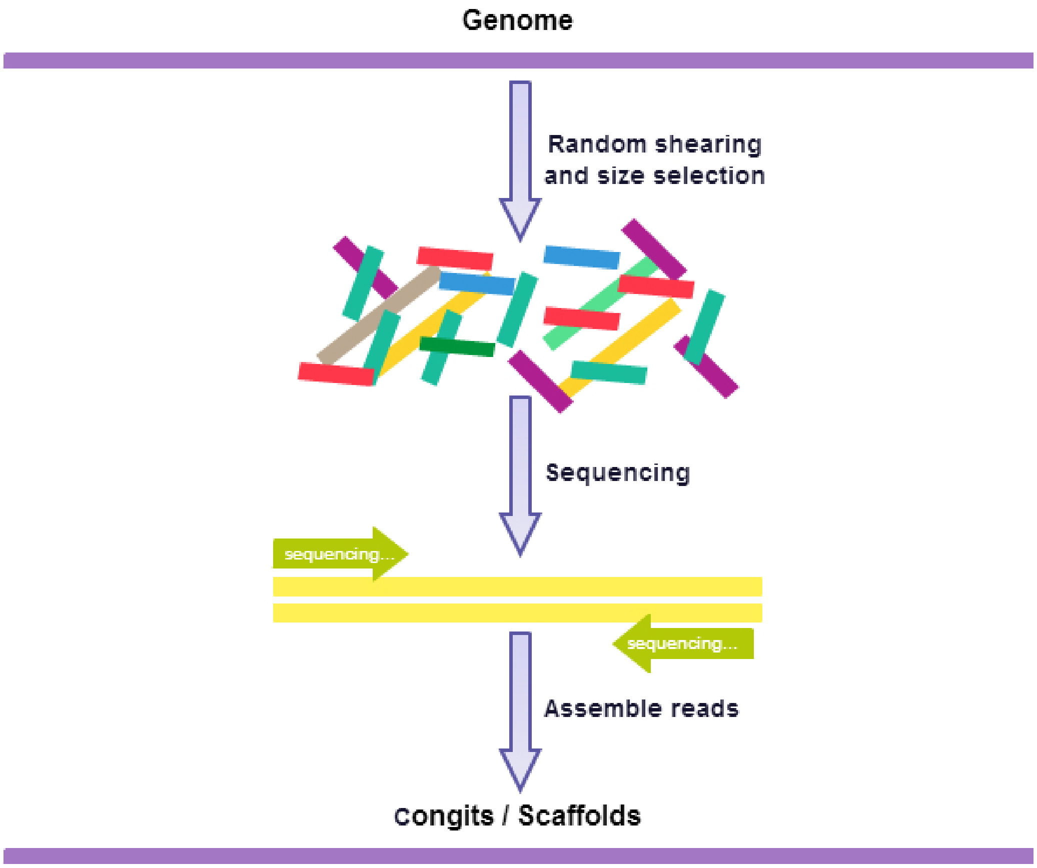


Figure 2.7: Genome Assembly

human DNA contains many identical or near identical sequences inside it. It is difficult to resolve repeats in assembly process, because if a read is sequenced from a repeated part of a genome, without any extra information, it becomes difficult to find which location that read comes from. Most assemblers collapse those reads to one place on the genome, so it causes a huge loss of genomic data.

2.4.0.1 Contig

The term **contig** is the mostly used to indicate any contiguous sequence that has been obtained from overlapped sequenced reads. Contigs are obtained from assemblers using *de novo* methods. Without any reference or extra information, reads are tried to be stitched together to construct contigs. Contigs are usually

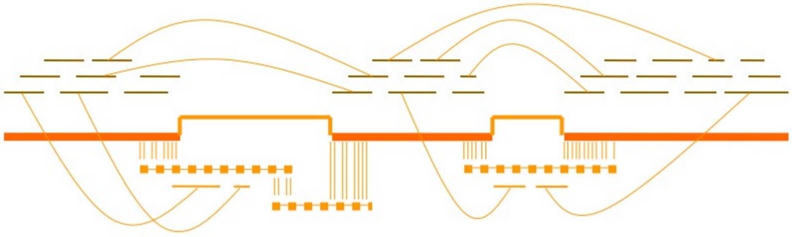


Figure 2.8: A scaffold representation

stored in FASTA format files. FASTA files are like FASTQ files, they contain only a header and the sequence for each contig.

2.4.0.2 Scaffold

Scaffold indicates the sequences constructed from ordering and orienting contigs using mate-pair reads. Scaffolds also contain N, additional to A, C, G, and T that signifies unknown amounts of missing sequence, usually called gaps.

2.4.0.3 Gap

Gaps usually exist between contigs to mark that there should be some sequence, but, because of insufficient coverage this sequence could not be reconstructed. Gaps are represented with Ns.

2.4.0.4 Coverage

Coverage is defined as the ratio of the total length of all the reads to the length of the genome. It can also be defined as the amount of reads that are mapped to a specific letter or sequence on a genome. High coverage is usually required to decide how the genome should be assembled. Low coverage can cause assembly programs to terminate while enlarging contigs, so scaffolding algorithms can continue to resume, but in this case low coverage causes gaps in the assembly.

2.4.0.5 Reference Genome

The idea of determining the whole nucleotide sequence in the human genome was first considered in 1985. Up to now, scientist studied on human reference genome and lastly 38th update was released in 2013. The most recent reference genome, Genome Reference Consortium build 38 (GRCh38), still contains gaps, but more than 92% of the genome is completed and used as a reference for studies [30]. A reference genome sequence is a map that provides the essential coordinate system for annotating the functional regions of the genome and comparing differences between individuals' genomes [31, 18, 32].

2.4.0.6 Alignment

```
  Query 1:AATTGCTGACCTCGATGCA 19
          ||| |||| ||||| ||||
Subject 1:AAT-GCTGTCCTCGCTGCA 18
```

Figure 2.9: Alignment between two sequences

Sequence alignment is a method that finds the similarity between two nucleotide sequences. The sequences having high similarity are considered evolutionary related. An easy way to find sequence alignment is arranging two sequences with inserting gaps in either of them until accessing the most similar sequences with least gaps. An example alignment between the sequences AATTGCTGACCTCGATGCA and AATGCTGTCCTCGCTGCA is shown in Figure 2.9.

To achieve the best alignment manually is often difficult. Two of the first alignment algorithm for computers were NeedlemanWunsch (1970) and Smith-Waterman (1981). Both algorithms use a m × n matrix to calculate the optimal alignment for the sequences, where m and n are the lengths of the two sequences. Both Needleman-Wunsch and Smith-Waterman are dynamic programming approaches to the problem and both find the optimal answer to the problems. Dynamic programming algorithms solve problems in quadratic time, so large amount of sequences can not be solved with these approaches since they are insufficient.

Read alignment is aligning all reads generated by sequencing with each other, or to a reference. After reads are obtained by sequencing, they are aligned with each other, and with following the alignments the original genome tried to be covered. Since NGS technologies produce a massive amounts of copies of reads, alignment work takes time. A sequence with aligned reads on it is shown in Figure 2.10. This process is also called as mapping. For read mapping, I used BWA [33] and Bowtie [34] in my work.

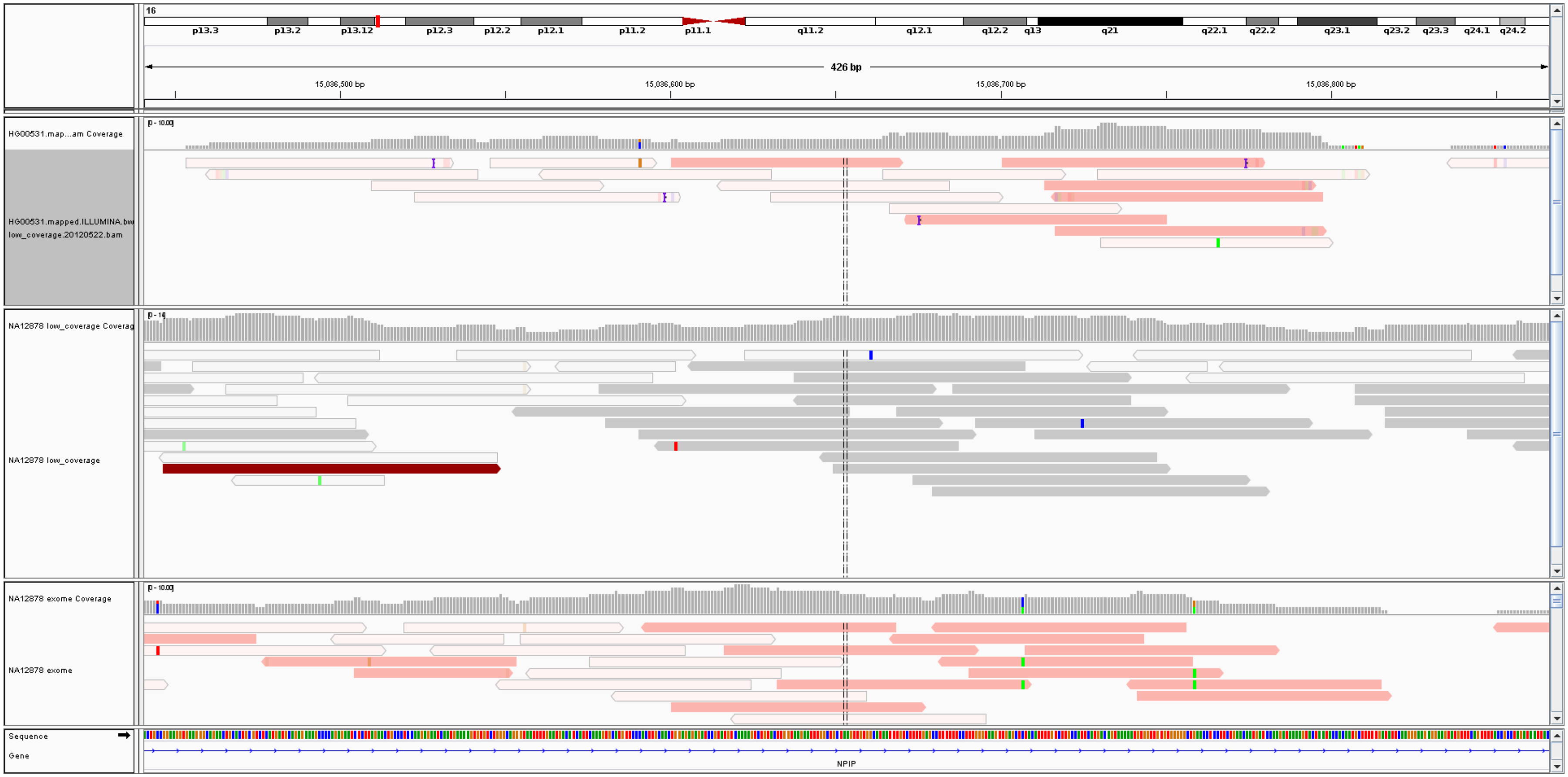


Figure 2.10: Read Alignment Visualization of 3 datasets from 1000Genomes Project [2]

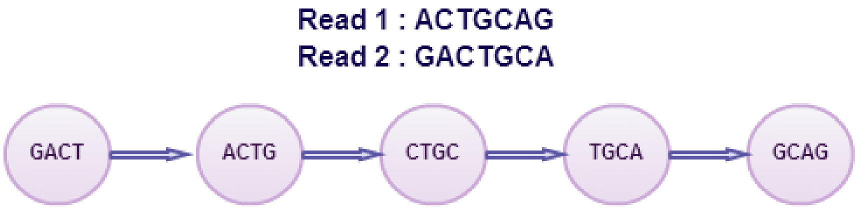
2.4.1 Genome Assembly Algorithms

Genome assembly problem can be solved with following three methodologies with NGS reads. They are greedy algorithms, overlap layout consensus (OLC) methods and de-Bruijn graph based algorithms. They all based on graphs.

A graph is represented with nodes and edges between them. Collection of edges form paths, that are found visiting the nodes in a special order, and all the values of nodes or edges in visiting order is usually meaningful.

An overlap graph represents the reads and their overlaps. Overlaps are computed by sequence alignment, which is computationally hard because the amount of the data (billions of reads). Reads are represented with nodes, and their overlaps between each other are represented with edges in overlap graphs. Contigs are found by following the overlaps of reads and contig sequence are found by traversing the path to sequence with values of nodes and edges.

A de-Bruijn graph is represented independently first with constructing all possible k length sequences for the alphabet of DNA and these fixed-length sequences are assigned to nodes. Edges are represented with suffix-prefix perfect matches of overlaps. A k-mer graph is formed of a de-Bruijn graph. The nodes are all possible k length sequences and edges are representing the overlaps with 1 difference. k-1 bases are overlapped between nodes and nodes can be visited many times. It is advantageous because overlaps are stored only once. For genome assembly with sort reads, the graph represents all reads. Each read is represented by a path in the graph. Overlapped reads share a common path, so read alignment is not required. Although k-mer graphs seem as a good solution for genome assembly, they are highly affected from sequencing errors (1-2%) and repetitive genomes. Sequencing errors cause incorrect overlaps in the graph so resultant contig may not be accurate. Each sequencing error links false nodes and each false node can be matched with another so it causes a false contig path in the assembly. Resolving these problems and assembling a genome is NP-hard problem and therefore there are heuristic algorithms which usually solves problems with approximations by simplifying graphs. An example k-mer representation is shown



in Figure 2.4.1.

2.4.2 Greedy Methods

The greedy algorithms are basic graph algorithms. They simplify the graph inducing by keeping only highest scored overlaps. After any contig extension, they may discard the read which is included to any contig. Greedy algorithms work flow is very simple. First, best overlapped reads (the best alignment between any two reads, they can be perfect matches) are built into a contigs, and these contigs are extended with the next highest scoring overlap to make a new join. Although contigs are extended with highest scoring overlaps, any false join can be followed by next false joins, so in the end, targeted sequence may be false-positive.

2.4.3 Overlap Layout Consensus

OLC algorithms perform well with long reads and small genomes. For NGS reads and large genomes, optimization is needed. Reads are assembled in three steps:

1. In overlap step, pairwise read alignments are computed on k-mer graphs. On pre-computed k-mer graphs, reads are represented and overlaps are found. In this step, an optimal k value should be selected. k should be smaller than read length. k shouldn't be too small because the complexity of the graph increases, and shouldn't be too large because enough overlaps may not be detected. For overlaps, alignment parameters are also important. Identical base pair percentage should be pre-defined. Alignment stringency effects accuracy and length of contigs.

2. In Layout stage, reads linked together in the previous stage are found on the

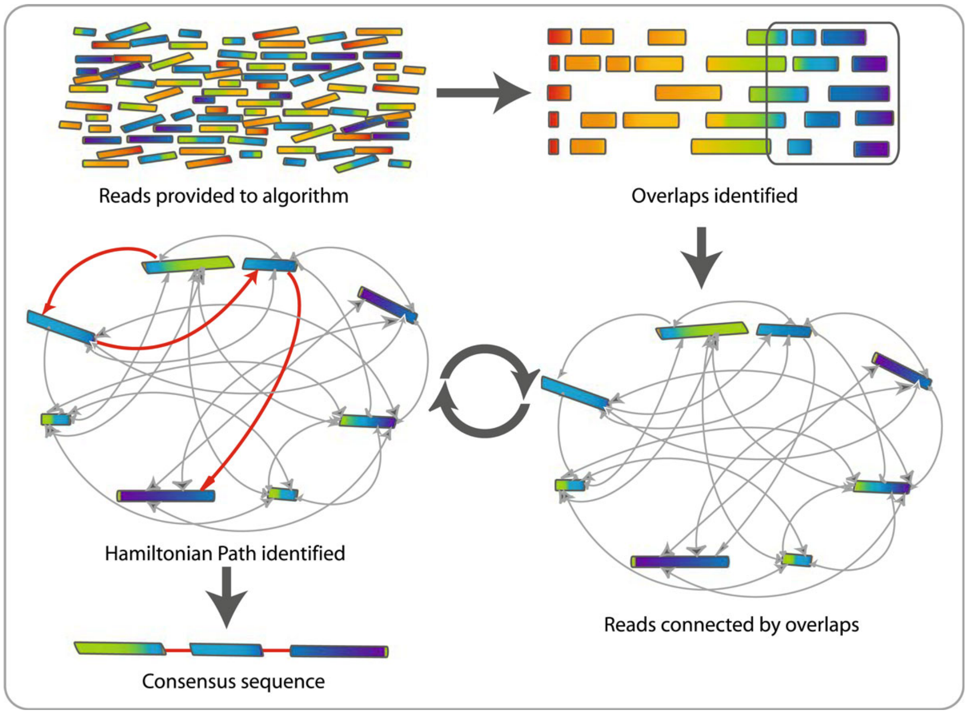

Reads provided to algorithm
Overlaps identified
Hamiltonian Path identified
Reads connected by overlaps
Consensus sequence

Figure 2.11: Overlap Layout Consensus work flow [3]

graph.

3. In consensus stage, contig sequences are obtained by following paths. For gathering sequence from nodes and edges, it is required to determine the best single nucleotide in the multiple sequence alignment. Figure 2.11 (adopted from [3]) shows the algorithm.

2.4.4 De Bruijn Graph Based Methods

De-Bruijn graph based methods are the popular ones in genome assembly. It is appropriate for short reads. It is again based on k-mer graphs but pairwise alignment of all reads are not required. Reads and overlaps between them are (usually) not stored so vast amount of reads are stored in an effective way.

The disadvantage of de-brujin graphs is inputs should be error free [35]. There should be no sequencing errors and all bases in targeted sequence should be

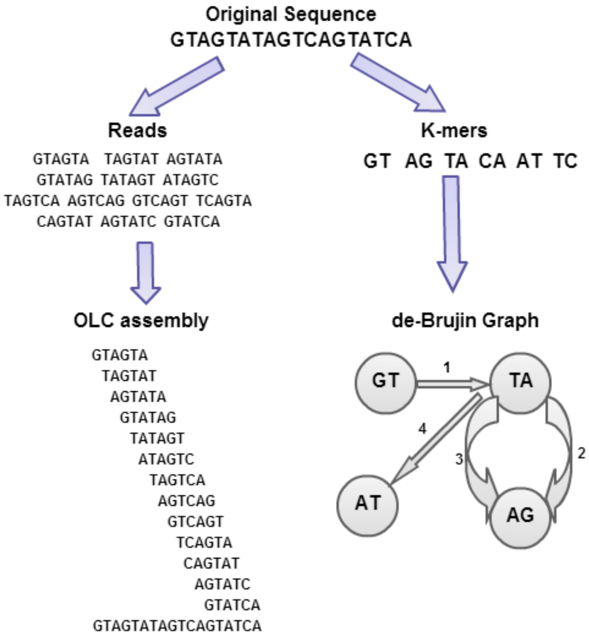


Figure 2.12: De Bruijn graph based algorithm

uniformly covered by reads and perfect match is required between reads. It aims to find a unique path on the graph by visiting all the nodes exactly once (same with finding Eulerian path problem) [36].

2.5 Scaffolding

Scaffolding problem is ordering and orienting of all contigs with given linking information. When the linking data contain errors, scaffolds can be false positive. Both ordering and orienting problems are found NP-hard, so there are approximation algorithms. One of the solution for scaffolding problem is representing contigs with nodes, edges with linking information and finding the path.

2.6 Problems

2.6.1 Repeats & Mis-assemblies

Most of the large genomes (mouse,human, etc) contains many repetitive sequences inside it (50%). Distinguishing these repeats in assemblies are most of the time tricky. If these sequences are not distinguished by assemblers, they are thought as same sequence in the targeted genome and causes breaks and lost of one of the repeated sequences. These lost assemblies are called as mis-assemblies. A mis-assembly is shown in Figure 2.6.1. Targeted genome is shown with its regions in the example, and it has seen that red regions are included twice in the genome. Let's consider this genome is broken into fragments and with sequencing reads are obtained. If reads length is larger than the repeat length, then repeats can be captured because one of the read will be mapped to somewhere in starting with blue, ending with yellow sequence, and one of the read will be mapped to somewhere in starting with yellow ending with green sequence. So, repeats mapped different positions on the targeted genome, which is expected. If repeats are larger than read length, reads from repeat sequences will be considered the same and will be tried to be mapped to the same position. So the resulting sequences will be contig1 and contig2 in the figure shown. So one of the repeat sequence disappears and a false assembly sequence obtained [37].

2.6.2 Segmental Duplications

Segmental duplications (also named low-copy repeats) sequences that range from 1 to 400 KB in length, occur more than one in the genome, and $> 90\%$ identical sequences in the genome [38].

WGS assembly was initially criticized because of its perceived inability to resolve repeat structures within genomes. Here, we quantify the effect of WGS sequence assembly on large, highly similar repeats by comparison of the segmental duplication content of two different human genome assemblies. Our analysis shows that

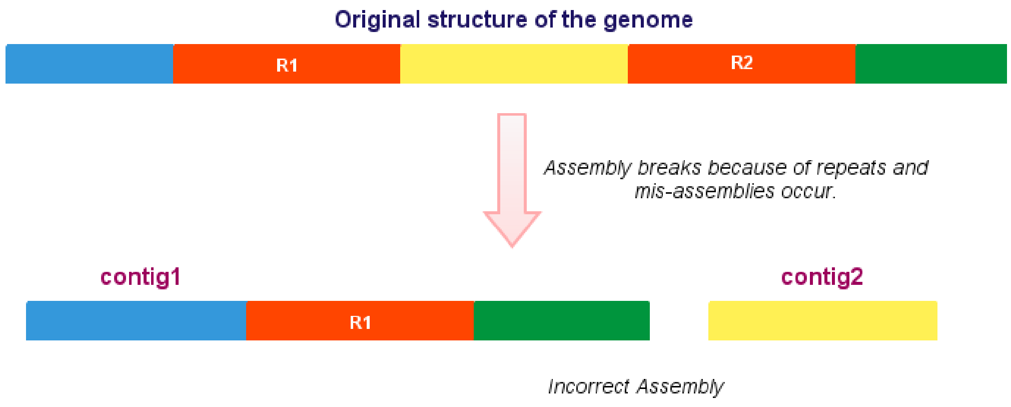

Original structure of the genome
R1
R2
Assembly breaks because of repeats and mis-assemblies occur.
contig1
contig2
R1
Incorrect Assembly

Figure 2.13: Mis-assemblies caused by repeats in genome

large (> 15 kilobases) and highly identical (> 97%) duplications are not adequately resolved by WGS assembly. This leads to significant reduction in genome length and the loss of genes embedded within duplications [39].

2.6.3 Heterozygosity

The two most difficult biological problems affecting assembly are complex genomic architecture seen in large regions with highly homologous duplicated sequences and so much allelic diversity [40]. It's seen that regions of segmental duplication are correlated with copy number variations (CNV). These regions containing large CNV segmental duplications have been misrepresented in the reference genome because it is aimed to be produced in haploid form. Highly identical paralogous and structurally different regions are collapsed into a single sequence during assembly so assembling both strands of the genome is challenging.

There are different methodologies to overcome this problem. One of the methodology considered so far is assembling a genome of hydatidiform mole (HM). A complete hydatidiform mole (CHM) is an abnormal product of conception that is a very early form of a fetal growing with a haploid genome. CHM is an ideal candidate for sequencing and assembling a haploid form of a human genome [40].

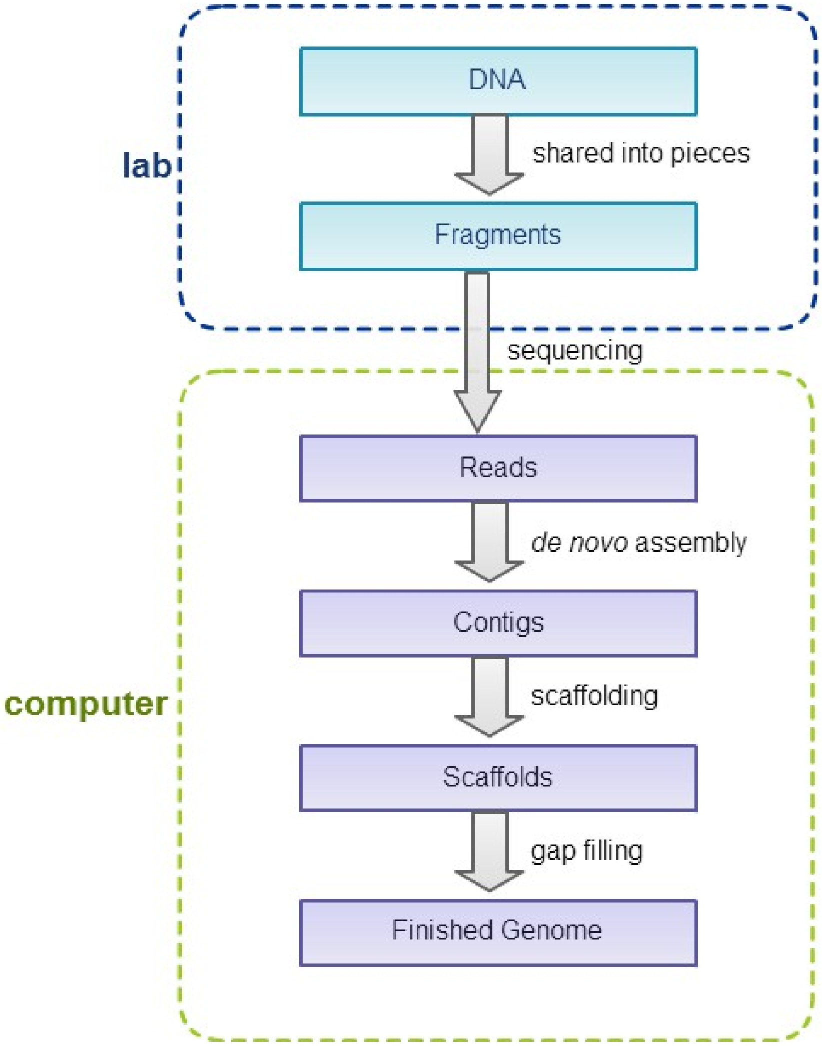


Figure 2.14: Genome Assembly processes

Chapter 3

Data and Methodology

Altough new and emerging NGS technologies have reduced significantly sequencing costs, much work remains to use them effectively for de novo sequencing of complex and highly repetitive genomes such as human genome or polyploid genomes. Here, we report benchmark results of using BAC pooled clone sequencing strategy.

Independent from the platform, two different sequencing strategies are used. Whole genome shotgun (WGS) sequencing is based on random shearing of whole genomic DNA and is preferentially applied to medium sized genomes with limited amounts of repetitive DNA. For plant genomes, WGS by NGS was so far restricted to re-sequencing purposes if a reference sequence was available and de novo sequencing of small and medium sized genomes.

The second, hierarchical sequencing (HS) approach is based on sequencing bacterial artificial chromosomes (BAC) anchored to a physical map (clone-by-clone sequencing). This strategy is more costly than WGS but in return, it is suitable to generate high quality reference sequences even for highly repetitive genomes. The map-based strategy was not only applied to sequencing the human genome but also to plant genomes. Due to its accuracy and reliability, the clone-by-clone strategy was also favored for producing a high-quality reference sequence of the barley genome [41].

Since WGS strategy is cheaper but not good at resolving repeats and HS strategy seems to be better for repeats but expensive, we propose to use a hybrid approach named Pooled Clone Sequencing developed by Kitzman et al [4].

3.1 Pooled Clone Sequencing

For our study, we use the genome of NA12878, an individual from Utah of Northern Europe ancestry. First, genomic DNA is broken into fragments using restriction enzymes and all diploid fragments are placed into an electrophoresis gel. Gel electrophoresis is used to separate the fragments by size and measure them. While heavy DNA fragments (longer sequences) move slowly, small ones travel further easily on the gel. Figure 3.1 shows electrophoresis gel with DNA fragments on it. The positions (on the left) and spacing shows relative sizes in Figure 3.1. After fragments are sorted by their length, the intended sized ones are cut from the gel. In Figure 3.1 lengths are shown as 250kb, 200kb, 150kb, 100kb, 500kb, 10kb and the white space on the gel is containing the fragments with size 150kb length. 150kb band is selected and cut out of the gel from each lane. Next, cloning vectors are prepared such that each 150kb DNA fragment, along with a short known sequence is packed into a bacteriophage virion. In the next step, these bacteriophage viruses are allowed to infect bacterium cells and multiply inside, amplifying our 150kb DNA fragments per cell. After these steps are followed, a single complex clone library of BAC cloning vector is constructed and split into 288 pools each containing 10^5 clones. Since the size of human genome is $3,4 \times 10^9$ by considering 10^5 clones per 150kb fragment, the genome is expected to be covered approximately 3 times by each pool. A vital step in the preparation of those libraries is splitting the initial single clone library of BAC into 288 pools, each of which captures approximately 3% of diploid human genome, so that the odds of getting the same region, given that 70% of human DNA is repetitive sequence, this same region is likely to refer to a repetitive segment of DNA, in the same pool is trivial.

The main aim dividing the genome into 288 pools is avoiding from overlapping repeated sequences and forcing them to be assembled separately. Finally, it's seen that there are variations in the quality of results depending on the tool and data set used. To able to notice the effect of complexity and size of the genome in assembly, chromosome 1 which is the longest and chromosome 20 which is one of the shortest are scaffolded in the study.

3.2 Scaffolding Tools

3.2.1 SSPACE

Although Bambus [42] is the first stand-alone scaffolding algorithm, SSPACE [13] is first scaffolder that use NGS reads. Since scaffolding problem is NP-hard [11] and there are heuristics for it, SSPACE solves the problem starting with largest contig first. After contigs are linked using paired end reads, scaffolds are constructed iteratively by joining contigs if they have enough connections between each other (minimum number of connections = 5) and the distance between contigs ensures the insert sizes of reads. If there are alternative links among the other contigs, then according to a ratio and a threshold, best pairs are chosen and scaffolding process continues until no more contigs joined. If no more contig is found to extend the current scaffold, the current scaffold is finalized. Process continues until all contigs are incorporated into linear scaffolds.

SSPACE has an option for extending contigs by tiling reads across contig ends in library files as a pre-processing step.

SSPACE also allows hierarchical scaffolding for multiple libraries. Libraries with different insert sizes are allowed in the process (starting with small insert libraries).

SSPACE provides the resulting scaffolds in FASTA format. A summary file is also aborted after a scaffold process which has useful statistics such as total number of scaffolds, their average size and N50 values. While running the tool, when it is requested by the user a dot file [43] is also provided for graphical results.

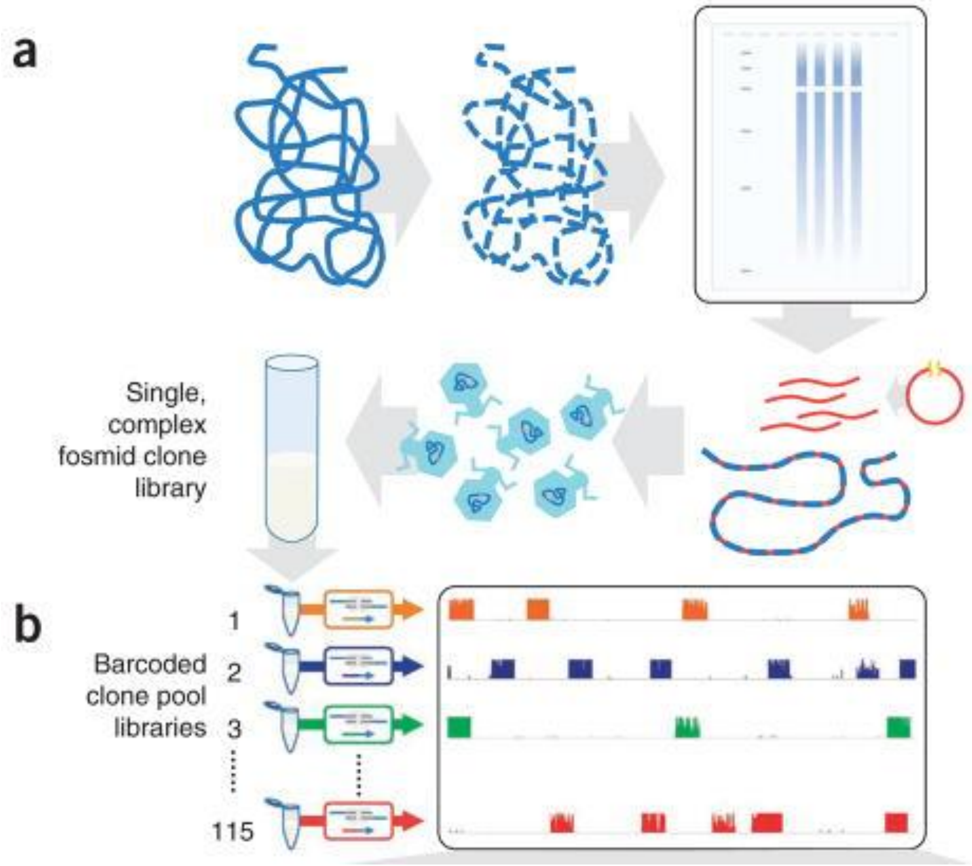


Figure 3.1: Pooled Clone Sequencing. Image adopted from [4]. a) DNA is extracted and size selected. b) DNA is diluted and partitioned into 288 pools

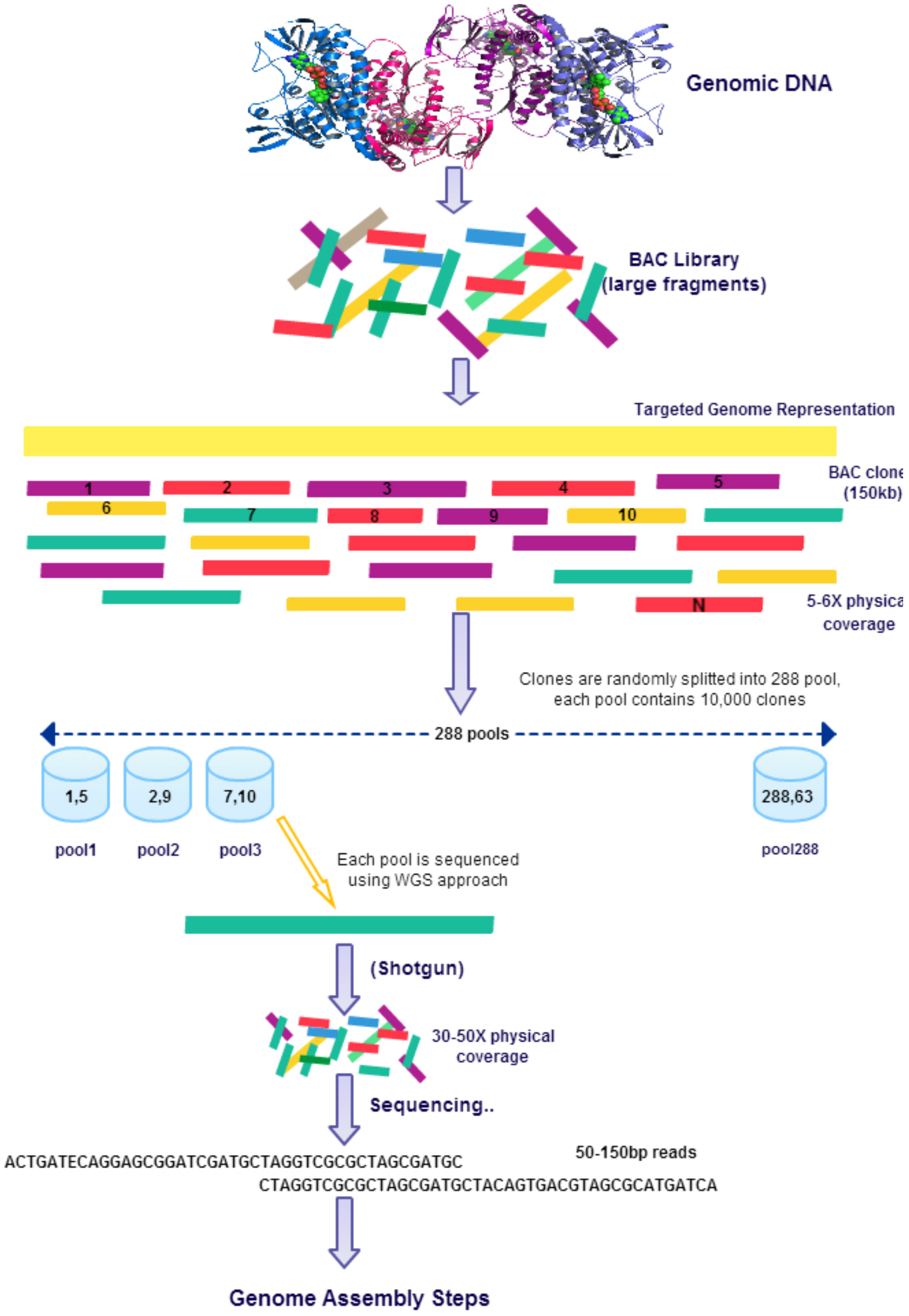


Figure 3.2: Single complex BAC library is constructed and split into 288 pools each containing 10,000 clones. Each pool is sequenced separately using WGS.

parameter	usage
-s contig file	contains contigs in FASTA format
-x [0 or 1] (default x=0)	parameter for whether extend the contigs with reads in lib files before scaffolding.
-k [int] (default k=5)	minimum number of reads to compute scaffolds
-p [0 or 1]	creates dot file for visualization.

Table 3.1: Parameters used for scaffolding with SSPACE

3.2.2 SCARPA

SCARPA [12] is one of the stand-alone scaffolder that uses Linear Programming to find near-optimal scaffolds. The biggest problem with scaffolders is mis-assemblies, and SCARA finalize them during the scaffolding process.

Most of the assemblers and scaffolders are highly affected from chimeric data (erroneous paired end reads). SCARPA defines a bound with some parameters and allows some mismatches during scaffolding. Therefore, some of mis-assemblies can be handled.

SCARPA runs with a contig file in FASTA format and SAM files contains that the mapping positions of one or more paired end libraries. Any software can be used for the mapping stage.

As a pre-processing stage, mapping files are filtered to move ambiguous mappings and some calculations (standard deviation, average insert size, etc) are done before scaffolding stage. If there are any contradictory contigs, they are eliminated. Final stage computes the scaffolds with given links and contigs, so it is very memory and time efficient.

While pre-processing links and contigs, if there is an ambiguity, SCARPA discards contigs rather than read pairs, which makes scaffolds more accurate but causes loss of data. The algorithm mainly tries to convert an arbitrary bi-directed graph to a directed graph by removing minimum number of contigs and nodes.

SCARPA command line as as followings:

For pre-processing:

```
scarpa_process -c contig_file -f reads -i insert_size
```

For scaffolding stage:

```
scarpa -c file -l file -i file -o file
```

parameter	usage
contigs *-c contig_filenames*	Contig file given in FASTA format. (required)
libraries *-l [libraries]*	Library files containing read pairs in fastq format (required)
mappings *-i mappings_filename*	File containing the read mappings. (required)
outputs *-o output_filename*	Directory for outputs. (required)
–min_support N	Minimum required mapped reads to connect two contigs. (default N=2)

Table 3.2: Parameters used for scaffolding with SCAPRA

3.2.3 OPERA

The developers of OPERA [11] aims to find an exact solution for scaffolding instead of heuristics. Since scaffolding problem is NP-hard [11], the exact solution can not calculated efficiently without any constraints. Therefore, they find an optimal solution under specified constraints.

OPERA provides a combinatorial algorithm that guarantees two fundamental issues. First, OPERA aims to use as much of the paired end data as possible, which makes the problem computationally hard to solve. Second, they guarantees the quality of the scaffolds and avoids over collapsing the assembly thus produces larger scaffolds but the more error prone.

OPERA is a graph based algorithm, where contigs are represented as nodes and paired end reads that map to contigs are edges. First, for each contig, two orientations (whether + or -) are assigned, then orientation of contigs are determined by linking paired reads. Using reads and contigs, a scaffold graph is constructed.

Gao et al [11] prove that the scaffolding problem could not be efficiently solved using a scaffold graph without any constraints. A lower bound for initial contig lengths and an upper bound for libraries that contigs can be spanned by a number of paired reads are required. Gao et al argue that, with a fixed number of reads spanning contigs, algorithm can be solved in polynomial time if any discordant edges are removed from the graph. Therefore, OPERA removes the discordant edges from the graph in pre-processing stage.

For repeat resolution, Gao et al show that their algorithm can be extended under some gap length constraints. In the current version of the tool, repeats are resolved using read coverage. Contigs that one covered more than 1.5 times than the genomic mean are filtered before the scaffolding stage.

OPERA is a user-friendly tool. The mapping stage is embedded into the pre-processing stage, which uses bowtie [8] or bwa [33] to map reads. After the pre-processing, perl script is used with reads and contigs, OPERA runs with contigs file and mapping files provided in two ways. It can be either run with a configuration file or parameters can be given directly. There are useful settings that can be included into the configuration file:

3.2.4 BESST

BESST [5] is a scaffolding algorithm that differs from others in estimating gaps in scaffolds. Sahlin et al [5] shows that scaffolding algorithms which developed so far use an inaccurate model for estimating gap size. Sahlin et al shows why maximum likelihood estimators are biased and describes the biases that scaffolding algorithms are facing. BESST provides a model by considering the distribution of reads spanning a gap and derives the ML-based equation previously used by other scaffolders for estimation of gap sizes. That, in fact turned out to be more accurate at such estimators.

Sahlin et al shows that many gaps are poorly estimated, although mapping errors or duplicated errors are removed from input. So BESST provides a model that

parameters	usage
Scaffolding related parameters	
cluster_threshold=k	OPERA discards all clusters less than this value during scaffolding. (default k=5)
abort=true	If running time for specific subgraph is longer than t, it is aborted.
Contig file related parameters	
file_format=fasta	Format of contig file(fasta or statistic, default=fasta)
filter_repeat=no	Eliminates repeated contigs. (yes or no, default=true)
repeat_threshold=1.5	If the coverage of a contig is more than repeat threshold * average coverage, it is considered as a repeated contig.(default threshold=1.5)
contig_size_threshold=m	Contig length threshold (default=500): OPERA will not use the contigs whose length is shorter than this value.
Mapping related parameters	
calculate_ori=no	Should paired-end reads orientations be recalculated or not. (yes or no, default=no)
read_ori=in	Paired end reads orientation (in, out or forward)
map_type=bowtie	Format of mapping file (bowtie or bwa, default=bowtie)
calculate_libs	Recalculate the library information(yes or no, default=yes)
lib_mean=10000	Library mean length
lib_std=1000	Library standard deviation

Table 3.3: Parameters used for scaffolding with OPERA

estimates gaps after contigs are ordered and oriented, and that inconsistent reads are eliminated.

Sahlin et al shows that the approach used by gap estimating scaffolders yield wrong results on the grounds of their assumption for ignoring differences in insert sizes of reads and their using an average insert size during gap gap size estimation. The assumption that the distribution of insert sizes for reads spanning a gap is the same as that of the library is inaccurate. Figure 3.3 exemplifies both negative and positive biases. A negative bias occurs the distribution of read pair insert-length is negatively skewed while a positive bias occurs upon positively skewed insert length distribution.

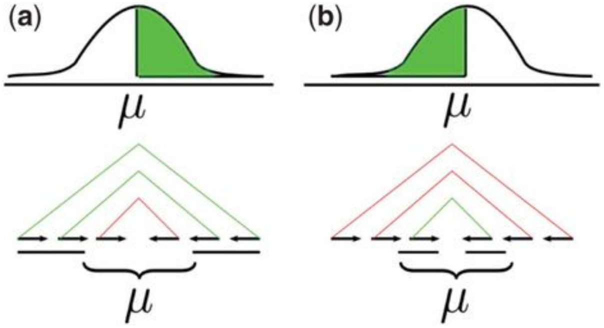


Figure 3.3: Illustrating bias in conditioned read-pair insert-length distribution [5]

parameters	usage
Required	-c (contig file) -f (.bam files) -o (output directory)
-e (optional)	-e [The least amount of links that is needed to create a link]
-z (optional)	-z [Coverage cut-off for repeat handling]
-y (optional)	-y [0 or 1 (Extend scaffolds with smaller contains (default on))].
-q (optional)	-q [flag (Parallelize work load of path finder module in case

Table 3.4: Parameters used for scaffolding with BESST

Chapter 4

Experimental Results

To evaluate current scaffolding algorithms and their performances when using pooled clone sequencing data, we performed two types of experiments. In our first experiment, we merged all pooled clone sequencing reads into a single library and we ran scaffolding algorithms on it.

In the second experiment we ran each and every algorithm on each pool one by one and added the resulting intermediary scaffolds into the subsequent pool on our way for further calculations. The aim of merging all reads and running the algorithms on these data-sets altogether is to eliminate the pooling effect and to benchmark the resulting scaffolds that were obtained by these two types of datasets.

4.1 Scaffolding contigs with merged reads

The results of table 4.1 are obtained by merging all of the pools into a single library of chromosome 1. There exists two algorithms, namely OPERA and SSPACE, that can diminish resulting scaffold numbers while increasing grand total of base pairs. OPERA appears to yield results that are in parallel with our expectations in the sense that it decreased the number of scaffolds most while

Tools	Scaffs	Total bps	ATCG	GC%	Ns	N50	N90
SSPACE	9,891	121,405,472	121,404,200	41.57	1272	28,279	5,757
SCARPA	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
OPERA	9,408	121,412,030	121,400,964	41.57	11,066	28,159	5,757
BESST	7,028	99,697,046	99,595,402	42.12	101,644	32,938	6,708

Table 4.1: Statistics of scaffolding chromosome 1. Scaffs: Number of scaffolds, Total bps: Grand total of bases, ATCG: Grand Total of A,C,T,Gs, GC%: percentage of gc content, Ns: Grand total of Ns, N50: the length of the smallest scaffold S in the sorted list of all scaffolds where the cumulative length from the largest scaffold to scaffold S is at least half of the total length, N90: the length of the smallest scaffold S in the sorted list of all scaffolds where the cumulative length from the largest scaffold to scaffold S is at least 90% of the total length.

leveling up grand total of base pairs. Next, SSPACE increased grand total of base pairs to 121,405,472 while diminishing scaffold number to 9,891. It introduced 1,272 N characters into the assembly and finds an N50 value of 28,279. BESST reduced the number of scaffolds dramatically but appeared to increase the level of redundancy in contig joins. As a result, there happens to be a considerable amount of loss of data. It found a scaffold number of 7,028 while decreasing the grand total to 99,697,046. Even though N50 value reaches 32,938 that might not mean anything significant on the grounds of the loss of substantial amount of data. SCARPA fails to yield reliable results due chiefly to its excessive memory requirements in the analyses of chromosome 1 and our merged read library.

Tools	Scaffs	Total bps	ATCG	GC%	Ns	N50	N90
SSPACE	249	10,019,741	10,019,735	44.75	6	49,769	22,871
SCARPA	248	10,019,787	10,019,735	44.75	52	50,183	22,871
OPERA	248	10,019,760	10,019,686	44.75	74	50,183	23,331
BESST	115	4,683,891	4,683,167	45.44	724	48,117	23,589

Table 4.2: Statistics of scaffolding chromosome 20. Scaffs: Number of scaffolds, Total bps: Grand total of bases, ATCG: Grand Total of A,C,T,Gs, GC%: percentage of gc content, Ns: Grand total of Ns, N50: the length of the smallest scaffold S in the sorted list of all scaffolds where the cumulative length from the largest scaffold to scaffold S is at least half of the total length, N90: the length of the smallest scaffold S in the sorted list of all scaffolds where the cumulative length from the largest scaffold to scaffold S is at least 90% of the total length.

Similarly, table 4.2 presents the results of merging all of the pools into a single library of chromosome 20. As expected, OPERA and SCARPA diminished resulting scaffold numbers while increasing grand total of base pairs. They found a scaffold number of 248 which was initially 250; and increased the N50 value from 49,272 to 50,183. SSPACE found 249 scaffolds diminishing grand total of 9 base pairs which is not significant. BESST reduced the number of scaffolds dramatically, however, it also reduced the grand total of base pairs; and as a result, it seems that a significant amount of data is lost. Also, the N50 value is decreased, suggesting the results are most likely not accurate.

4.2 Scaffolding contigs hierarchically

Tools	Scaffs	Total bps	ATCG	GC%	Ns	N50	N90
SSPACE	9,569	121,501,965	121,491,831	41.57	10,134	29,121	5,936
SCARPA	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
OPERA	9,897	121,406,580	121,403,836	41.57	2,744	28,531	5757
BESST	513	1,564,335	1,564,334	50.66	1	4,520	1,319

Table 4.3: Statistics of scaffolding chromosome 1. Scaffs: Number of scaffolds, Total bps: Grand total of bases, ATCG: Grand Total of A,C,T,Gs, GC%: percentage of gc content, Ns: Grand total of Ns, N50: the length of the smallest scaffold S in the sorted list of all scaffolds where the cumulative length from the largest scaffold to scaffold S is at least half of the total length, N90: the length of the smallest scaffold S in the sorted list of all scaffolds where the cumulative length from the largest scaffold to scaffold S is at least 90% of the total length.

Furthermore, table 4.3 presents the results obtained by running all of the pools in different libraries of chromosome 1. Again, as expected, SSPACE and OPERA diminished resulting scaffold numbers while increasing grand total of base pairs. SSPACE found a scaffold number of 9,569 while increasing the N50 value to 29,121 and the grand total of base pairs to 121,501,965. OPERA produced 9,897 scaffolds with a grand total of 121,406,580 base pairs. BESST reduced the number of scaffolds dramatically, but it also reduced the grand total of base pairs, too; and as a result, significant amount of data seem to be lost; and again the N50 value is decreased so the results are most likely to be inaccurate. As previously mentioned, SCARPA fails to yield reliable results due chiefly to its excessive memory requirements in the analyses of chromosome 1.

Tools	Scaffs	Total bps	ATCG	GC%	Ns	N50	N90
SSPACE	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
SCARPA	247	10,019,775	10,019,735	44.75	40	5,018	23,331
OPERA	250	10,019,740	10,019,735	44.75	5	49,272	22,521
BESST	17	308,948	308,948	47.81	0	22,521	10,538

Table 4.4: Statistics of scaffolding chromosome 20. Scaffs: Number of scaffolds, Total bps: Grand total of bases, ATCG: Grand Total of A,C,T,Gs, GC%: percentage of gc content, Ns: Grand total of Ns, N50: the length of the smallest scaffold S in the sorted list of all scaffolds where the cumulative length from the largest scaffold to scaffold S is at least half of the total length, N90: the length of the smallest scaffold S in the sorted list of all scaffolds where the cumulative length from the largest scaffold to scaffold S is at least 90% of the total length.

The results of table 4.4 are obtained by processing the pools one by one corresponding to single libraries of chromosome 20. Among all, SCARPA is the only algorithm which could perform compellingly - however marginal - whereas 247 out of 250 contigs were correctly assembled and in total, 40 Ns were inserted into the resulting assembly, thus increasing the grand total of base pairs to 10,019,775 and N50 to 51,331, respectively. SSPACE algorithm fails to run for the reads given in different libraries. Likewise, when the default threshold of minimum number of read pairs supporting links was set to 5, OPERA could not perform any scaffolding operations; and no improvement was made by decreasing the threshold value to 2. BESST algorithm succeeds to decrease the number of scaffolds to 17, but the total ground and the N50 are decreased to 308,948 and 22,521, respectively; thus, although the number of scaffolds are reduced dramatically, however, this value is not significant enough to imply the robustness of the algorithm.

4.3 Evaluation

4.3.1 Lost Data

Although we are trying to organize sequences into large scaffolds, we recognized that resulting scaffolds' total base pairs are less than initial assembly's total base pairs. We believe that this is an important error in scaffolding tools. We believe that possible reason for such a situation might be as follows. After scaffolding processes, we expect an increment in the total number of base pairs, or at least, no decrement. Because in the process of scaffolding, contigs are sorted and gaps among contigs are filled with ”N” characters, N being the number of bases in the gap. The main reason for reduction in the base pair number might be the elimination of the contigs that cannot be ordered or oriented.

We noticed that data lost during scaffolding generate a massive problem for BESST. Although scaffold number is very low after contigs are scaffolded using BESST, N50 value is not as high. First, Sahlin et al [5] show that genome scaffolding problem is all about detecting and utilizing the correct links. The algorithm aims to remove unambiguous links first. After reads are mapped to chromosome 20, BESST first filters out 58 links as fishy reads, 1993 links as non-unique reads (at least one read non-unique in read pair) that map to different contigs, and 220 links as duplicated reads. Although number of useful reads (reads mapping to different contigs uniquely) was calculated to be 9908, It ignores 7632 links as reads with too large insert size. And, when those uniquely mapped reads are further processed, we see that only 1276 reads are used for scaffolding steps. Second, after links are found, BESST removes isolated nodes (contigs) which cause the loss of data. In this example, 152 isolated contigs were removed from downstream analysis of scaffolding.

4.3.2 Minimum Number of Read Pairs Supporting Links Between Contigs

By default, minimum number of read pairs supporting links between contigs is set to 5 in all four scaffolding algorithms. Since the coverage of our data is too low, we decided to change it to 2 but no significant results were obtained.

BESST has a different algorithm to link different contigs. It uses number of links supporting a contig (edge) as an indicator of reliability. Firstly, the number of links between two contigs depends on the distance between them. Yet, structural features, such as heterozygosity or repetitive regions, introduce unorthodox clustering patterns where reads are mostly misaligned. BESST assumes that non-structural misalignments such as sequencing errors are negligible compared to structural ones. Counting solely links does not yield reliable results. BESST evaluates edges based on link statistics and tries to make an educated guess of the accuracy of the differential alignment of each read to contigs [5].

SSPACE joins one contig another if and only if their distance satisfies the presumed insert size by user. After contigs are paired, SSPACE starts with the longest contig which satisfies minimum number of links which is 5 by default. It also considers alternative links; a ratio is calculated between alternative links and highest scoring links. If a contig has no links with any others, this contig is excluded and these processes are repeated until all contigs are incorporated into scaffolds.

OPERA is based on an algorithm which finds all possible scaffolds then minimizes the discordant ones with the information provided by the paired end reads. Linking information is used for downstream processes if the scaffold is consistent with user-defined insert size.

SCARPA is also based on an exact approach like OPERA. Number of read pairs that links edges is used to weight the links. It discards the links with a minimum number of linking value lower than the preset threshold (2) during the process.

4.3.3 Coverage

Coverage is defined as the total number of sequenced nucleotides divided by the (estimated) length of the genome. For instance 5× coverage implies that the genome is sequenced 5 times.

Better assembly results are resolved upon higher coverage. Coverage value is essential for genome assemblies to handle sequencing errors. To achieve this, a high depth of coverage is essential, but this time, assembly process will be much more computationally expensive. The coverage needed also depends on the organism, its genome size, and the repetitive content of sequences. Therefore, coverage should be decided accordingly during sequencing.

For de novo assemblies, high coverage is required where reads are obtained using WGS. Our datasets are obtained using WGS technology with a coverage of approximately 3× which was initially thought to be 30×. For scaffolding process, we have BACs with low coverage (5×-6×). The data we used have 5× physical coverage, including 3×-4× coverage for 15kb inserts.

It was shown that 50× coverage was optimal for E.coli genome. In our opinion, since the coverage of our data is too low (3×), we could not obtain reliable results.

4.3.4 Insert Size

De novo assembly and scaffolding processes are substantially affected from read length. Although the same data are simulated using different insert sizes, quite diverging results were obtained in study [44]. Hunt et al shows on simulated data that, while 1% correct joins were found with short reads in assembly when using OPERA, 99% correct joins were seen with long reads. Longer reads usually help in improving contig size and they are also essential for handling repeat clusters. Repeats that extend beyond the insert sizes are hard to resolved by tools.

4.3.5 Important Notes

Hunt et al [44] show that resultant scaffolds are heavily affected by the choice of scaffolder, mapping tool used, insert size and the genome being analyzed. Despite these difficulties, we have tried to boost the precision and contiguity of assembly by splitting reads into pools. We could not obtain consistent results on the grounds of the low coverage. Yet, by using BESST, scaffolding numbers are dramatically reduced as aimed. It is likely because BESST links independent contigs aggressively therefore the low coverage problem of our data could not heavily penalize the scaffolding process. However, with higher coverage BESST may generate chimeric scaffolds.

Chapter 5

Conclusion

Genome assembly problem is typically solved by a two stage process: contig assembly followed by scaffolding. Scaffolds are the main focus of assembly statistics. Obtaining longer scaffolds is important to be able to present large sequences in a genome. Scaffolds are highly prone to errors, especially when generated using short reads or repetitive sequences.

Even small genomes, such as bacteria, contain significant number of repeats; it is computationally hard to assemble the human genome using short reads only. *De novo* assembly with short reads results in a set of contigs with gaps at each repeat region that are longer than read lengths. To bridge these gaps, BAC libraries are very useful when sufficient coverage is obtained. For this reason, we decided to use BAC library that was split into 288 pools, providing 5× physical coverage of the genome.

We experienced that the scaffolders vary in their usability, speed and accuracy. Overall, SSPACE is very useful since it is very easy to install and run. BESST is good at making joins in an aggressive way. OPERA and SCARPA are better when handling mis-assemblies.

Although we tried to improve scaffolds, we recognized that resulting scaffolds total base pairs are less than initial total base pairs. We think that this is an important source of error of scaffolding tools. Possible reason for this might be as follows. After scaffolding processes, we expect an increment of the total number of base pairs or at least no decrement because in the process of scaffolding, contigs are sorted and gaps between different contigs are filled with N characters, N being the number of bases in the gap. The main reason for reduction in the base pair number may be due to the elimination of the contigs that cannot be ordered or oriented.

5.1 Future Work

The study was based on real datasets so it was difficult to check accuracy. Simulated datasets are needed for authentic evaluation processes of the data. We have simulated some data and split them it into 288 pools. We will run the scaffolders on these data, and this time, we will be able to observe the effect of pooled clone strategy on scaffolding.

Since the scaffolders that have been implemented so far have still important limitations, we will implement a new scaffolding tool which will be compatible with pooled clone strategy. It is obvious that pre-processing before scaffolding improves the resultant assembly. Therefore we will start with pre-filtering data first.

We hope to run scaffolders on remaining chromosomes and also on additional sets of genomes. Pooled clone strategy is good at resolving repetitive sequences so plant genomes which harbor massive amount of repetitive sequences might also be assembled with this fashion.

Bibliography

[1] E. C. Berglund, A. Kiialainen, and A.-C. Syvanen, “Investigative genetics,” 2011.

[2] N. Siva, “1000 genomes project,” *Nature biotechnology*, vol. 26, no. 3, pp. 256–256, 2008.

[3] J. Commins, C. Toft, M. A. Fares, *et al.*, “Computational biology methods and their application to the comparative genomics of endocellular symbiotic bacteria of insects,” *Biological procedures online*, vol. 11, no. 1, pp. 52–78, 2009.

[4] J. O. Kitzman, A. P. MacKenzie, A. Adey, J. B. Hiatt, R. P. Patwardhan, P. H. Sudmant, S. B. Ng, C. Alkan, R. Qiu, E. E. Eichler, *et al.*, “Haplotype-resolved genome sequencing of a gujarati indian individual,” *Nature biotechnology*, vol. 29, no. 1, pp. 59–63, 2011.

[5] K. Sahlin, N. Street, J. Lundeberg, and L. Arvestad, “Improved gap size estimation for scaffolding algorithms,” *Bioinformatics*, vol. 28, no. 17, pp. 2215–2222, 2012.

[6] E. C. Berglund, A. Kiialainen, and A.-C. Syvänen, “Next-generation sequencing technologies and applications for human genetic history and forensics,” *Investig Genet*, vol. 2, p. 23, 2011.

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

[8] B. Langmead, C. Trapnell, M. Pop, S. L. Salzberg, *et al.*, “Ultrafast and memory-efficient alignment of short dna sequences to the human genome,” *Genome Biol*, vol. 10, no. 3, p. R25, 2009.

[9] M. Margulies, M. Egholm, W. E. Altman, S. Attiya, J. S. Bader, L. A. Bemben, J. Berka, M. S. Braverman, Y.-J. Chen, Z. Chen, *et al.*, “Genome sequencing in microfabricated high-density picolitre reactors,” *Nature*, vol. 437, no. 7057, pp. 376–380, 2005.

[10] C. Alkan, S. Sajjadian, and E. E. Eichler, “Limitations of next-generation genome sequence assembly,” *Nature methods*, vol. 8, no. 1, pp. 61–65, 2011.

[11] S. Gao, W.-K. Sung, and N. Nagarajan, “Opera: reconstructing optimal genomic scaffolds with high-throughput paired-end sequences,” *Journal of Computational Biology*, vol. 18, no. 11, pp. 1681–1691, 2011.

[12] N. Donmez and M. Brudno, “Scarpa: scaffolding reads with practical algorithms,” *Bioinformatics*, vol. 29, no. 4, pp. 428–434, 2013.

[13] M. Boetzer, C. V. Henkel, H. J. Jansen, D. Butler, and W. Pirovano, “Scaffolding pre-assembled contigs using sspace,” *Bioinformatics*, vol. 27, no. 4, pp. 578–579, 2011.

[14] K. Sahlin, F. Vezzi, B. Nystedt, J. Lundeberg, and L. Arvestad, “Besst-efficient scaffolding of large fragmented assemblies,” *BMC bioinformatics*, vol. 15, no. 1, p. 281, 2014.

[15] S. Gnerre, I. MacCallum, D. Przybylski, F. J. Ribeiro, J. N. Burton, B. J. Walker, T. Sharpe, G. Hall, T. P. Shea, S. Sykes, *et al.*, “High-quality draft assemblies of mammalian genomes from massively parallel sequence data,” *Proceedings of the National Academy of Sciences*, vol. 108, no. 4, pp. 1513–1518, 2011.

[16] N. C. Jones and P. Pevzner, *An introduction to bioinformatics algorithms.* MIT press, 2004.

[17] M. V. Olson, “The human genome project.,” *Proceedings of the National Academy of Sciences*, vol. 90, no. 10, pp. 4338–4344, 1993.

[18] E. S. Lander, L. M. Linton, B. Birren, C. Nusbaum, M. C. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh, *et al.*, "Initial sequencing and analysis of the human genome," *Nature*, vol. 409, no. 6822, pp. 860–921, 2001.

[19] M. L. Metzker, "Sequencing technologies?the next generation," *Nature Reviews Genetics*, vol. 11, no. 1, pp. 31–46, 2009.

[20] E. R. Mardis, "The impact of next-generation sequencing technology on genetics," *Trends in genetics*, vol. 24, no. 3, pp. 133–141, 2008.

[21] F. Sanger, S. Nicklen, and A. R. Coulson, "Dna sequencing with chain-terminating inhibitors," *Proceedings of the National Academy of Sciences*, vol. 74, no. 12, pp. 5463–5467, 1977.

[22] E. R. Mardis, "Next-generation dna sequencing methods," *Annu. Rev. Genomics Hum. Genet.*, vol. 9, pp. 387–402, 2008.

[23] D. R. Bentley, S. Balasubramanian, H. P. Swerdlow, G. P. Smith, J. Milton, C. G. Brown, K. P. Hall, D. J. Evers, C. L. Barnes, H. R. Bignell, *et al.*, "Accurate whole human genome sequencing using reversible terminator chemistry," *Nature*, vol. 456, no. 7218, pp. 53–59, 2008.

[24] D. J. Hedges, T. Guettouche, S. Yang, G. Bademci, A. Diaz, A. Andersen, W. F. Hulme, S. Linker, A. Mehta, Y. J. Edwards, *et al.*, "Comparison of three targeted enrichment strategies on the solid sequencing platform," *PLoS One*, vol. 6, no. 4, p. e18595, 2011.

[25] M. A. Quail, M. Smith, P. Coupland, T. D. Otto, S. R. Harris, T. R. Connor, A. Bertoni, H. P. Swerdlow, and Y. Gu, "A tale of three next generation sequencing platforms: comparison of ion torrent, pacific biosciences and illumina miseq sequencers," *BMC genomics*, vol. 13, no. 1, p. 341, 2012.

[26] B. Merriman, I. Torrent, J. M. Rothberg, R. Team, *et al.*, "Progress in ion torrent semiconductor chip based sequencing," *Electrophoresis*, vol. 33, no. 23, pp. 3397–3417, 2012.

[27] E. E. Schadt, S. Turner, and A. Kasarskis, "A window into third-generation sequencing," *Human molecular genetics*, vol. 19, no. R2, pp. R227–R240, 2010.

[28] S. Koren, M. C. Schatz, B. P. Walenz, J. Martin, J. T. Howard, G. Ganapathy, Z. Wang, D. A. Rasko, W. R. McCombie, E. D. Jarvis, *et al.*, "Hybrid error correction and de novo assembly of single-molecule sequencing reads," *Nature biotechnology*, vol. 30, no. 7, pp. 693–700, 2012.

[29] P. J. Cock, C. J. Fields, N. Goto, M. L. Heuer, and P. M. Rice, "The sanger fastq file format for sequences with quality scores, and the solexa/illumina fastq variants," *Nucleic acids research*, vol. 38, no. 6, pp. 1767–1771, 2010.

[30] D. M. Church, V. A. Schneider, T. Graves, K. Auger, F. Cunningham, N. Bouk, H.-C. Chen, R. Agarwala, W. M. McLaren, G. R. Ritchie, *et al.*, "Modernizing reference genome assemblies," *PLoS biology*, vol. 9, no. 7, p. e1001091, 2011.

[31] T. M. Smith and S. G. Porter, "Development and role of the human reference sequence in personal genomics," *eLS*.

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

[33] H. Li and R. Durbin, "Fast and accurate short read alignment with burrows–wheeler transform," *Bioinformatics*, vol. 25, no. 14, pp. 1754–1760, 2009.

[34] B. Langmead and S. L. Salzberg, "Fast gapped-read alignment with bowtie 2," *Nature methods*, vol. 9, no. 4, pp. 357–359, 2012.

[35] D. R. Zerbino and E. Birney, "Velvet: algorithms for de novo short read assembly using de bruijn graphs," *Genome research*, vol. 18, no. 5, pp. 821–829, 2008.

[36] J. R. Miller, S. Koren, and G. Sutton, "Assembly algorithms for next-generation sequencing data," *Genomics*, vol. 95, no. 6, pp. 315–327, 2010.

[37] L. STEVEN and J. SALZBERG, "Beware of mis?assembled genomes," *Bioinformatics*, vol. 21, no. 4, pp. 320–4, 2005.

[38] A. J. Sharp, D. P. Locke, S. D. McGrath, Z. Cheng, J. A. Bailey, R. U. Vallente, L. M. Pertz, R. A. Clark, S. Schwartz, R. Segraves, *et al.*, "Segmental duplications and copy-number variation in the human genome," *The American Journal of Human Genetics*, vol. 77, no. 1, pp. 78–88, 2005.

[39] X. She, Z. Jiang, R. A. Clark, G. Liu, Z. Cheng, E. Tuzun, D. M. Church, G. Sutton, A. L. Halpern, and E. E. Eichler, "Shotgun sequence assembly and recent segmental duplications within the human genome," *Nature*, vol. 431, no. 7011, pp. 927–930, 2004.

[40] K. M. Steinberg, V. K. Schneider, T. A. Graves-Lindsay, R. S. Fulton, R. Agarwala, J. Huddleston, S. A. Shiryayev, A. Morgulis, U. Surti, W. C. Warren, *et al.*, "Single haplotype assembly of the human genome from a hydatidiform mole," *bioRxiv*, p. 006841, 2014.

[41] B. Steuernagel, S. Taudien, H. Gundlach, M. Seidel, R. Ariyadasa, D. Schulte, A. Petzold, M. Felder, A. Graner, U. Scholz, *et al.*, "De novo 454 sequencing of barcoded bac pools for comprehensive gene survey and genome analysis in the complex genome of barley," *BMC genomics*, vol. 10, no. 1, p. 547, 2009.

[42] M. Pop, D. S. Kosack, and S. L. Salzberg, "Hierarchical scaffolding with bambus," *Genome research*, vol. 14, no. 1, pp. 149–159, 2004.

[43] E. R. Gansner and S. C. North, "An open graph visualization system and its applications to software engineering," *Software Practice and Experience*, vol. 30, no. 11, pp. 1203–1233, 2000.

[44] M. Hunt, C. Newbold, M. Berriman, and T. D. Otto, "A comprehensive evaluation of assembly scaffolding tools," *Genome biology*, vol. 15, no. 3, p. R42, 2014.

Appendix A

Data

Appendix B

Code