FROM MILLIONS TO ONE: THEORETICAL AND CONCRETE APPROACHES TO *DE NOVO* ASSEMBLY USING SHORT READ DNA SEQUENCES

by

ARIELLA SYMA SASSON

A Dissertation submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Computational Biology and Molecular Biophysics

written under the direction of

Anirvan Sengupta

and approved by

_____

_____

_____

_____

_____

New Brunswick, New Jersey

October 2010

ABSTRACT OF THE DISSERTATION

From Millions to One: Theoretical and Concrete Approaches to *De Novo* Assembly Using Short Read DNA Sequences

By ARIELLA SYMA SASSON

Dissertation Director:
Anirvan Sengupta

One of the most significant advances in biology has been the ability to sequence the DNA of organisms. Even in the shadow of the completion of the human genome, intractable regions of the genome remain incomplete. Next generation high-throughput short read sequencing technologies are now available and have the ability to generate millions of short read DNA sequences per run. Although greater coverage depths are possible, *de novo* sequence assembly with these shorter sequences is significantly more complex than resequencing; handling them presents new computational problems and opportunities. Identifying repetitive regions, coping with sequencing errors, and manipulating the millions of short reads simultaneously, are some of the difficulties that must be overcome. As a result of these complexities and working with the short read sequences from the Waksman SOLiD sequencing platform, this work explores the problem of *de novo* assembly. Initially, we develop tools for filtering short read sequence data based on quality scores and find that this procedure is critical for the success of the subsequent *de novo* assembly. Next, we analyze the key phenomena responsible for producing contigs

that are much shorter than the values provided by theoretical estimates. Finally, we explore two different routes to circumventing the difficulty imposed by short contigs. The first involves utilization of information from multiple orthologous genomes in a comparative assembly. In particular, we developed a pipeline for using the reference genome of a close by relative to improve genome assembly. The second approach uses paired read information to build scaffolds that are two orders of magnitude larger than the original contigs. For typical bacterial genomes, less than one hundred of these scaffolds are required to cover the entire genome. The combination of short reads from various platforms, assembly, and recovery pipelines brings mid-sized genomes close to completion. As a result, minimal additional work using conventional sequencing technologies are enough to close the remaining small gaps and return a finished single genome. Current advancements in sequencing technologies leave us hopeful that it would be possible to provide fairly complete assemblies for complex genomes via these technological approaches.

Acknowledgement and Dedication

I would like to thank my advisor, Professor Anirvan Sengupta for the invaluable discussions and guidance in preparing this thesis. In addition, discussions with Dr. Adel Dayarian, Dr. David Sidote, Dr. Mark Diamond and everyone associated with the Waksman Genomics Core Facility for their help and discussions surrounding high throughput sequencing. I would like to thank all my family and friends for their support throughout this endeavor. I especially want to thank my husband, Steven Springer, and sister, Celia Sasson, for their constant help and support; this would not have been possible without them. Finally, I would like to thank the Department of Energy for their funding through Computational Science Graduate Fellowship and the Krell Institute for its administration [DE-FG02-97ER25308].

I would like to dedicate this thesis to my children, Evelyn and Jacob, for keeping me inspired, smiling, and grounded. Finally, I would also like to dedicate this to my parents, Laurette and Jacques Sasson, who instilled in me a lifelong love of learning, for their unwavering support, and teaching me anything can be achieved if done one step at a time.

Table of Contents

Lists of tables

List of illustrations

Introduction

The DNA double helix is nature's elegant solution to the problem of how to store information and how to pass that information from one generation of cells to the next. DNA is classically known to contain the information, which can be transcribed to RNA and then translated into proteins which are necessary for the cell's functioning. Since DNA contains the genetic information that is the basis for all modern living things to function, reproduce, and grow, knowledge of the sequence is essential for a better understanding of the organism of interest. The ability to sequence a genome quickly and cheaply will revolutionize the study of medicine and science. The goal of individual genome sequencing is closer to becoming a reality than ever before. Such an advance could lead to an era of personalized medicine, such as personalized cancer therapy based on the sequencing of the cancer tumor genome in order to maximize effectiveness of the treatment [1].

Already over the past few decades, DNA sequencing has dramatically changed the nature of biological research in many areas such as genetics, evolution, comparative biology, cell biology and developmental biology. Recent technological advances in high throughput sequencing (HTS) reduce the time and cost to sequence DNA. Improvement to the biochemistry involved continually increases the short read length (50 bp -150bp). Therefore, these advances minimized the effort involved in the actual sequencing, while at the same time providing an unprecedented amount of sequence information that needs to be analyzed.

Chapter 1: Sequencing Technologies

1.1 Early Sequencing Techniques:

DNA sequencing is a tool that has been available to scientists for over 30 years. While the initial protocols were developed by Frederick Sanger et al. in 1975, the chain-termination method, also known as the Sanger method did not initially become the sequencing method of choice [2, 3]. At first, Maxam-Gilbert sequencing, also known as chemical sequencing, was more popular even though the sequencing protocols were published two years later [4]. The reason for the initial popularity of Maxam-Gilbert sequencing was the ability to use purified DNA directly while Sanger sequencing required the DNA of interest to be cloned. Maxam-Gilbert sequencing is based on chemical modification of DNA and subsequent cleavage at specific bases [4]. Over time, with the improvements made to the chain termination method, Maxam-Gilbert sequencing fell out of favor allowing for Sanger Sequencing to become the sequencing method of choice.

The Sanger method is still considered the gold standard of DNA sequencing. The Sanger method's strengths are the length (700-1000 bp) and quality of its reads. While the Sanger method originally used radioactivity for identification, current automated methods use fluorescently labeled 2',3'–dideoxynucleotide triphosphates (ddNTPs). Since the ddNTP cannot form a phosphodiester bond with the next dNTP, the DNA chain elongation is terminated. The ratio of a particular ddNTP constitutes only 1% of regular dNTP mix, therefore enabling some DNA polymerization to continue. The polymerase reaction produces a mixture of fluorescent products of various lengths that can be resolved by several methods, namely gel or capillary electrophoresis. Even considering

the various improvements in techniques and automation that have been made over the past three decades, Sanger sequencing is still time consuming and expensive to sequence a large genome. (Table 1.1)

	Sanger	Illumina		SOLiD	
		G.A. IIx	HiSeq 2000	4	4hq
Read Length	Fragment 700-1000 bp	Fragment 35-150 bp Paired End 50 x 50 bp 75 x 75 bp 100 x 100 bp 150 x 150 bp Mate-Pair 36 x 36 bp	Fragment 35-150 bp Paired End 50 x 50 bp 75 x 75 bp 100 x 100 bp 150 x 150 bp Mate-Pair 36 x 36 bp	Fragment 35-50 bp Paired End 35 x 25 bp 50 x 35 bp Mate-Pair 50 x 50 bp	Fragment 35-75 bp Paired End 50 x 35 bp 75 x 35 bp Mate-Pair 75 x 75 bp
Run	96 Mbp (per plate)	90 Gbp	150-200 Gbp	150 Gbp	500 Gbp
Time / Gbp	10,000 days (27.4 years)	0.25 days (2x100 bp)	<1 hour (2x100 bp)	0.5 days (2 x50bp)	<1 hours (2 x75 bp)
Cost/Gbp	~$2,000,000	~$1,000		~$1,000	

Table 1.1: Sequencing Costs – The sequencing cost and run comparison between Sanger Sequencing machines and two of the second generation sequencing platforms

Sanger sequencing, with the advantages of a reduced handling of toxic and radioisotopes, became the preeminent sequencing technique for decades. This tool, coupled with innovations in the genomics field and continually improving computer systems and algorithms, allowed for the initiation of an ambitious project: deciphering the human genome. The primary goal of Human Genome Project's (HPG) was to determine the sequence and to identify all the genes of the human genome (current

estimate is approximately ~25,000 genes). Initially, the goal would be reached using a sequencing plan which was traditional for larger genomes: using Sanger sequences to chromosome walk, also known as primer walking (Figure 1.1). This method progresses through the entire DNA strand, piece by piece, until the sequence of the entire chromosome is known. This project, founded in 1990 by the United States Department of Energy and the National Institute of Health, had a budget of $3 billion and was expected to take 15 years [5].

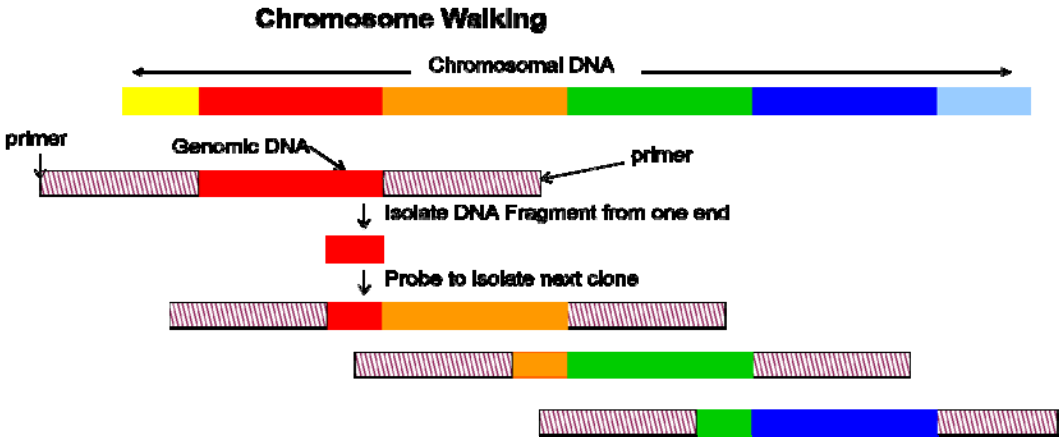


Figure 1.1: Chromosome Walking – This was the original approach of the HGP to sequence the human genome. To start the process, a primer that matches the beginning of the desired DNA sequence is used to generate a short portion of DNA containing the unknown sequence for a portion of the chromosome. This newly generated short DNA strand is then sequenced. The end of this short piece of DNA is used as a primer for the next part of the long DNA. That way the short sequences from the long DNA keeps walking along the sequence, therefore sequencing the entire chromosome.

The competition between Celera Genomics, a private company, and the publically funded Human Genome Project led to great advances in the sequencing community and fundamentally changed the way people thought about sequencing large genomes. One such advance adopted by Celera genomics was a technique that at the time was only attempted on small genomes or fragments of large genomes, namely shotgun sequencing

[6, 7]. Since Sanger sequencing can only be used for relatively short read lengths of 700 to 1,000 bp, longer sequences, such as chromosomes, must be reduced to smaller pieces and then re-assembled in order to get the complete sequence. Whole genome shotgun sequencing is a faster but a significantly more complex and riskier process, since the large quantity of reads and no position identification make assembly computationally challenging (Figure 1.2). For shotgun sequencing [6, 7], the DNA is randomly sheared into many small segments and then sequenced using the Sanger method [2, 3]. With shotgun sequencing, redundancy of the sequence is essential in order to reassemble the sequence. Overlapping ends are used to assemble the final contiguous sequence. As a result, several rounds of fragmentation and sequencing are essential to obtain the necessary redundancy in order to complete assembly. For sequencing the human genome, certain areas of the genome were sequenced up to twelve fold coverage in order to make an accurate assembly.

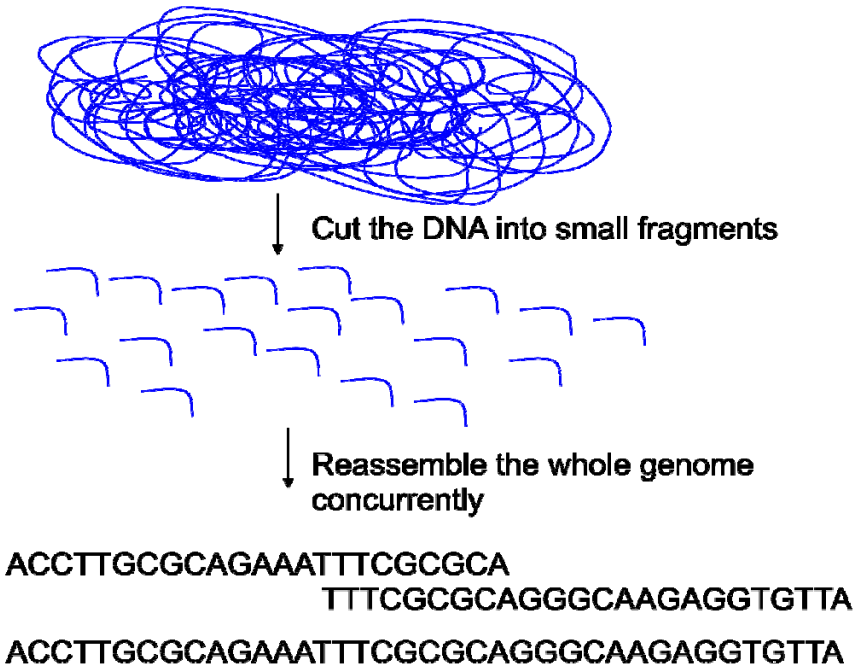


Figure 1.2: Whole Genome Shotgun Sequencing – This approach was used by Celera Genomics to sequencing the human genome. Take many copies of the genome of interest. Cut in randomly into sizes with the ability to be sequenced. Reassemble the genome. With their own sequencing information from step two and the relative positions provided by the daily publishing of the HGP BAC, Celera genomics published their version of the genome in 2001.

Sophisticated computational alignment tools, such as the Celera Assembler, were created to align and assemble the data from both their own sequencing runs as well as the HGP released sequences [8]. Ultimately, the advancements of Celera Genomics resulted in the Human Genome Project changing its methods for reaching their goal, by moving from a chromosome walking approach to hierarchal shotgun sequencing of large fragments of the genome namely the BAC-by-BAC method (Figure 1.3). In the BAC-by-BAC method, the DNA was cut into BAC, bacterial artificial chromosome, sized pieces 150-350 kbp, and each BAC was shotgun sequenced simplifying the assembly slightly. In

the end, Celera Genomic's submitted the first draft of the human genome at a cost of approximately $300 million. (HGP budget was $3 billion) [9, 10].

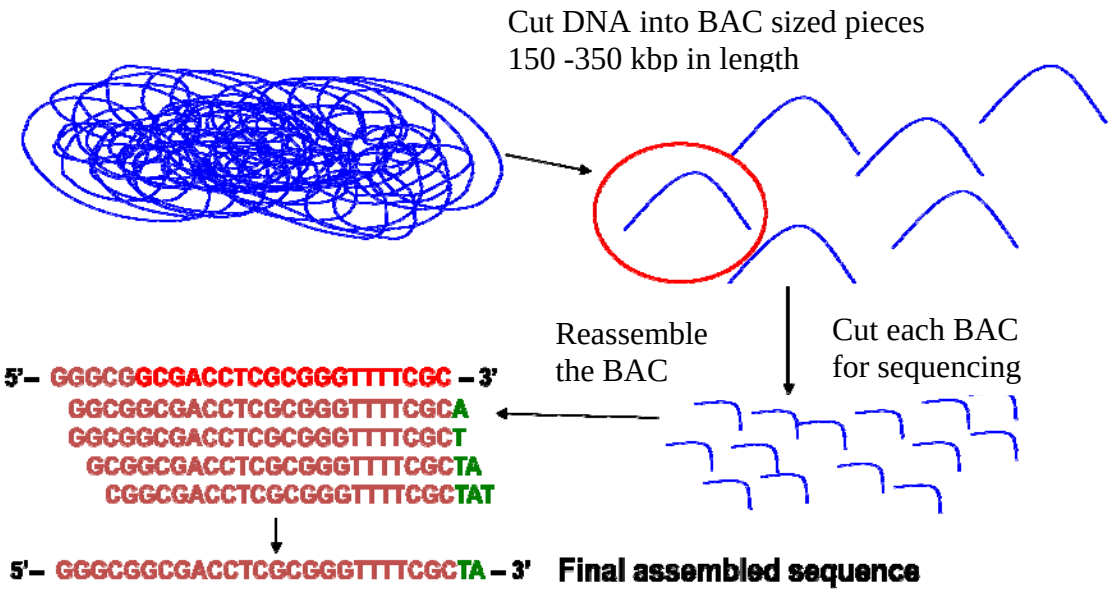


Figure 1.3: BAC-by-BAC Method - This approach was what was ultimately used by the HGP for their sequencing runs. Take many copies of the genome of interest. Cut them into BACs (150-350 kbp). Cut each BAC randomly into sizes with the ability to be Sanger sequenced. Reassemble the BACs. The HGP had a slightly simplified problem of assembling many BACs, but then the genome needed to be reassembled from the BACs.

1.2 Current Sequencing Technology:

Today, even after the human genome project has been labeled as completed, problems still lurk in assembly projects. Intractable regions, or regions of repetitive sequences in the chromosomes that result in gaps in the genome assembly, remain unsequenced. New whole genome sequencing technologies are needed to reach the goal of resequencing a specific human genome for $1000 or less. [11]. For years, the technological advances were limited to improvements on Sanger sequencing and derivatives, such as shotgun sequencing. A revolution in sequencing technology first

appeared in 2005 from 454 Life Sciences, sequencing by synthesis [12], and the multiplex polony sequencing protocols from the Church lab at Harvard Medical School [13]. Both groups developed methods that enabled massively parallel high throughput sequencing. At the time, these technologies enabled fifty fold increases in sequence quantities over the current Sanger sequencers at a much lower cost, approximately 1/9th of the cost of conventional sequencing (Figure 1.4) [13].

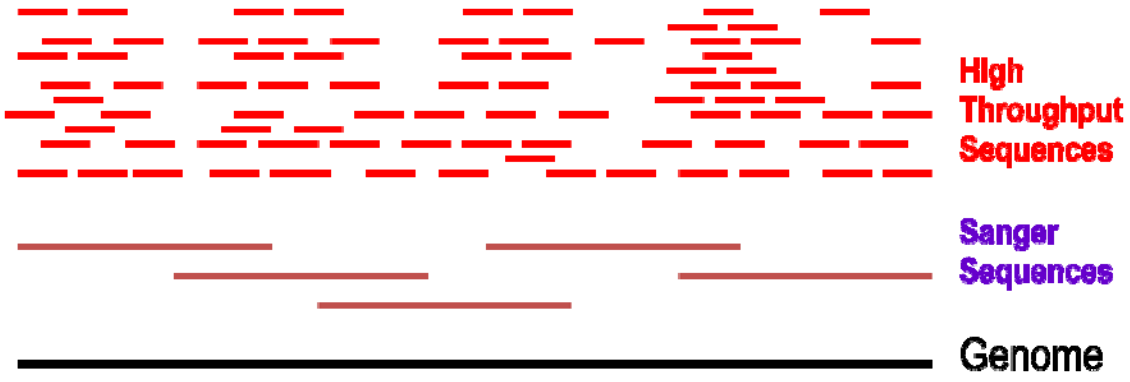


Figure 1.4: Qualitative comparison - This figure is a qualitative comparison between the sequences generated by Sanger and those from the HTS platforms. There is higher abundance and depth of coverage with the short reads, but they are also significantly shorter with little overlap allowed for confidence.

Despite the initial complaints regarding higher error rates, much shorter read lengths (original lengths: 454 100 bp [14], Church Lab 17-18 bp [13]), and the increased computational power needed to handle the data, these technologies are currently available and thriving in labs throughout the world [15]. HTS platforms are now capable of generating far cheaper albeit far shorter reads (50 to 500 bp instead of 800 to 1,000 bp), presenting new computational problems and opportunities. All of the HTS technologies have the same problems and the advantages of the original 454 sequencing technology, when compared to Sanger sequencing: high throughput sequencing, lower costs, and

higher error rates (Table 1.1). When the high throughput sequencing platforms are compared to each other, there are advantages and disadvantages to each. Technological advancements at each company are constantly improving the read lengths, while the error rates are being decreased both directly and indirectly. The HTS platforms are slowly improving error rates directly by improving the biochemistry and the mechanisms for the detection of the nucleotides, and indirectly because the sheer magnitude of coverage of the genome should help to correct the remaining errors. Finally, computational resources are constantly dropping in price and are becoming more available to handle the magnitude of data. As these sequencers become more ubiquitous, computational programs are constantly being developed to aid in the sequencer specific output.

New applications for HTS platforms are constantly being developed. The resequencing, chromatin immunoprecipitation (ChIP-Seq), methylation analysis (Methyl-Seq), gene expression profiling (RNA-Seq, SAGE), and small RNA analysis pipelines traditionally require the alignment of the HTS sequences against the reference genome as an initial step. This alignment reduces the computational burden since the alignment gives confidence in the accuracy of the remaining dataset. All the reads with too many errors would not have aligned and get thrown out prior to continuation down the desired pipeline. With the inherent lack of a reference genome for *de novo* assembly, the problems of how to handle the data in order to optimize an assembly is still not resolved and, as the quantities of data increase, these problems are becoming ever more complicated.

The HTS platforms are comprised of several distinct platforms. These technologies include massively parallel sequencing-by-synthesis (SBS) such as

Roche/454 pyrosequencing and Illumina's "Clonal Single Molecule Array" technology, and sequencing-by-ligation (SBL) such as Applied Biosystems SOLiD Analyzer (Rusk and Kiermer 2008). All of these platforms require the creation of a clone-free library of short DNA molecules with oligonucleotide tags on their ends. There are two key technical benefits of these approaches over Sanger sequencing. The first is that there is no need to clone the DNA and propagate them through bacterial systems. The second is no prior knowledge of the sequence is required for the sequencing; the oligonucleotide tags are independent of the genomic DNA. Each of these platforms can generate approximately one hundred megabases to many gigabases of raw sequence data per run, with costs in the $5,000-$10,000 range per run, depending on the platform.

Sequencing by synthesis, from Illumina, uses a DNA polymerase to identify the bases present in the complementary DNA molecule (Figure 1.5). Reversible terminator methods use reversible versions of dye-terminators. Sequence-by-synthesis adds one nucleotide at a time, detecting fluorescence corresponding to that position. The blocking group is then removed to allow the polymerization of another nucleotide [16]. The current read lengths have been recently increased to 150 bp for short read fragments and 150 x 150 bp ends for paired end reads. Additionally, Illumina has recently added a long-insert mate-pair protocol for their machine. The protocol produces up to 150 x 150 bp pair reads with long linkers two to five kilobases in length. Because of the library preparation's lack of robustness, it is currently not very widely used.

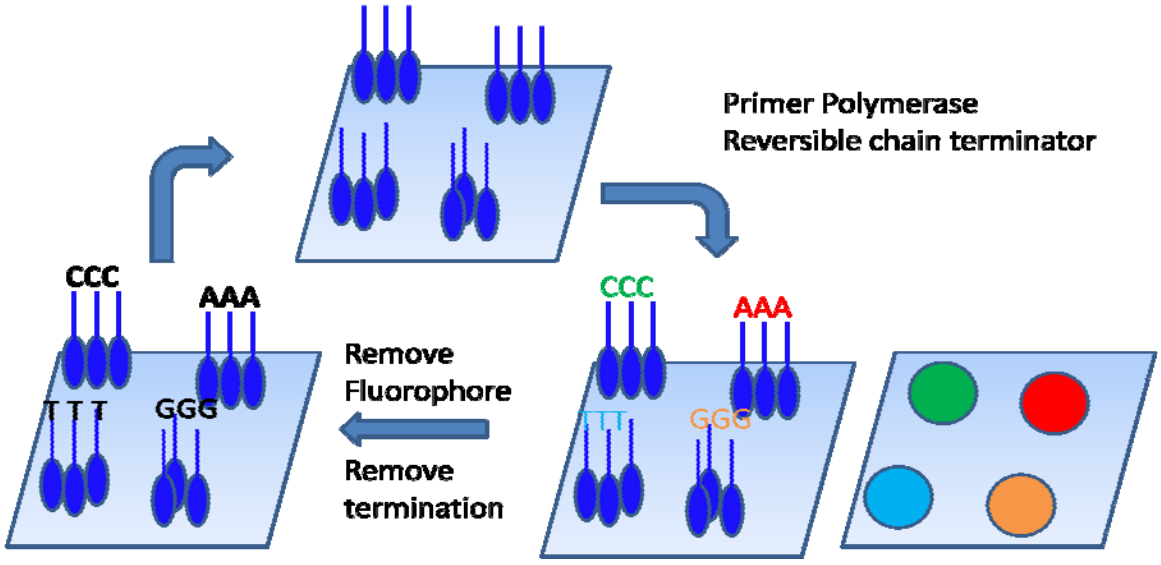


Figure 1.5: Illumina's Clonal Single Molecule Array Technology - In an Illumina flow cell, clusters of sequence are grown. Each cluster should contain one single sequence. Illumina flows reversible dye terminators over these clusters one at a time. The correct base pair is incorporated and with the terminator on the nucleotide sequencing cannot continue on. The florescence is recorded after all four nucleotides are passed. Then the termination is reversed and the cycle begins again until the sequence is finished. (Adapted from [17])

In contrast, pyrosequencing, a specific form of sequence by synthesis from 454 Life sciences, also uses DNA polymerization to add nucleotides (Figure 1.6). Pyrosequencing adds one type of nucleotide at a time, detects the light emitted by the release of attached pyrophosphates, and uses it to quantify the number of nucleotides added to a given location. Increasing the short read length is of the greatest interest to 454 Life Sciences, and they consequently produce the longest short reads of these technologies, 300-500 bp [12, 17].

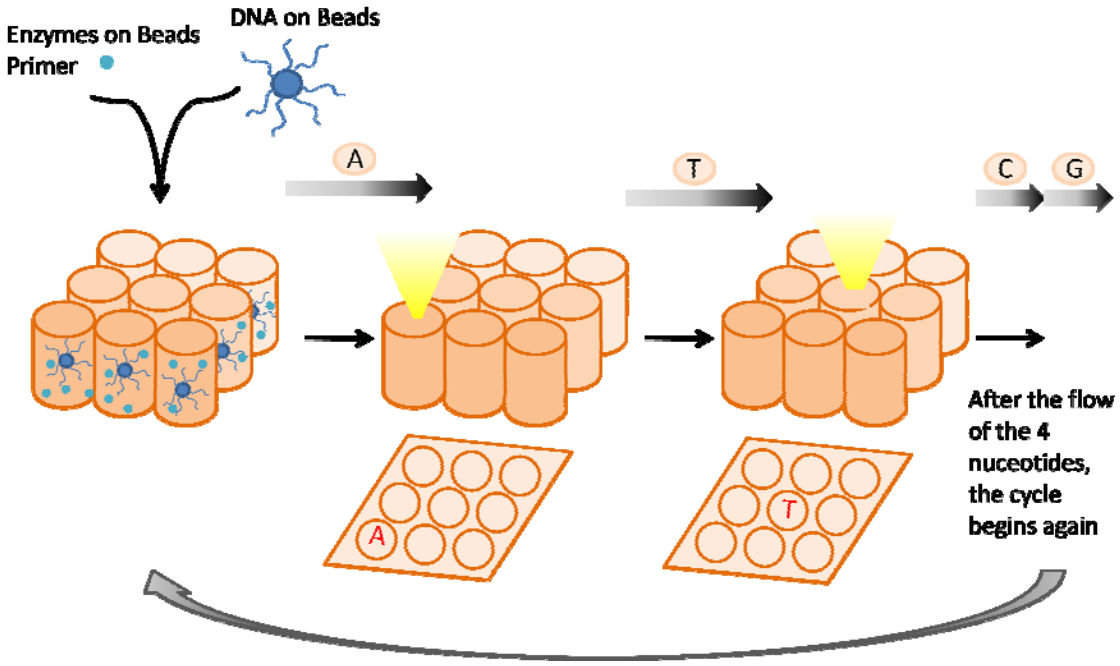


Figure 1.6: Roche/454 Pyrosequencing - In pyrosequencing, a bead containing multiple copies of a single sequence is dropped in a well with enzymes and primer. The machine then flows one nucleotide at a time over the wells. As the correct base is incorporated the well, the energy release is quantized and recorded. The cycle begins again once the last nucleotide is flowed through. Since quantizing long homogeneous stretches is difficult, pyrosequencing is known to have problems in those locations. (Adapted from [17])

Sequencing by ligation, used in Applied Biosystems SOLiD analyzer, is fundamentally different. It uses a DNA ligase enzyme rather than polymerase to identify the target sequence (Figure 1.7). This method uses a pool of all possible oligonucleotides, where positions 4-5 contain 1 of 16 specific dinucleotides. Hybridization and ligation of a specific oligonucleotide that matches that of the template occurs. The preferential ligation by DNA ligase for matching sequences allows for the detection of a specific signal. Other than the biochemistry, the major difference between SOLiD and the other sequencer systems is that it queries 2 bases at once. Thus, there is double confirmation of every base [18]. These sixteen possible dinucleotides are divided into four groups and assigned a unique color (i.e. color 0 represents AA, TT, CC, and

GG) (Figure 1.7C). While only four colors are used, these groups are designed such that every combination of base and color call will uniquely identify the second base. Therefore, each color essentially encodes a transition matrix in the base space. Each SOLiD sequence starts with a reference base (i.e. the last base of the primer usually a T or G) followed by a number of color call. Using the reference base and the first color call, one can find the first base. This base in turn can be combined with the second color call to obtain the second base. Continuing this pattern, one can translate the entire sequence back into DNA-space.

The current maximum read length for SOLiD is 50 bp for both short read fragments and the ends of mate pair reads. Additionally, with the latest upgrade of the machine, paired end capabilities have been added. For the paired end reads, the forward read can be either 35 or 50 bp and the paired end is 25 bp long. In 2008, SOLiD published that their cost of resequencing the human genome is approximately $60,000 [19]. According to their 2010 promotional information, the cost has dropped down to $6,000 for 30x coverage of the human genome [20]. While this price only accounts for the sequencing reagents necessary, this is one step closer to the current goal of the $1,000 human genome.

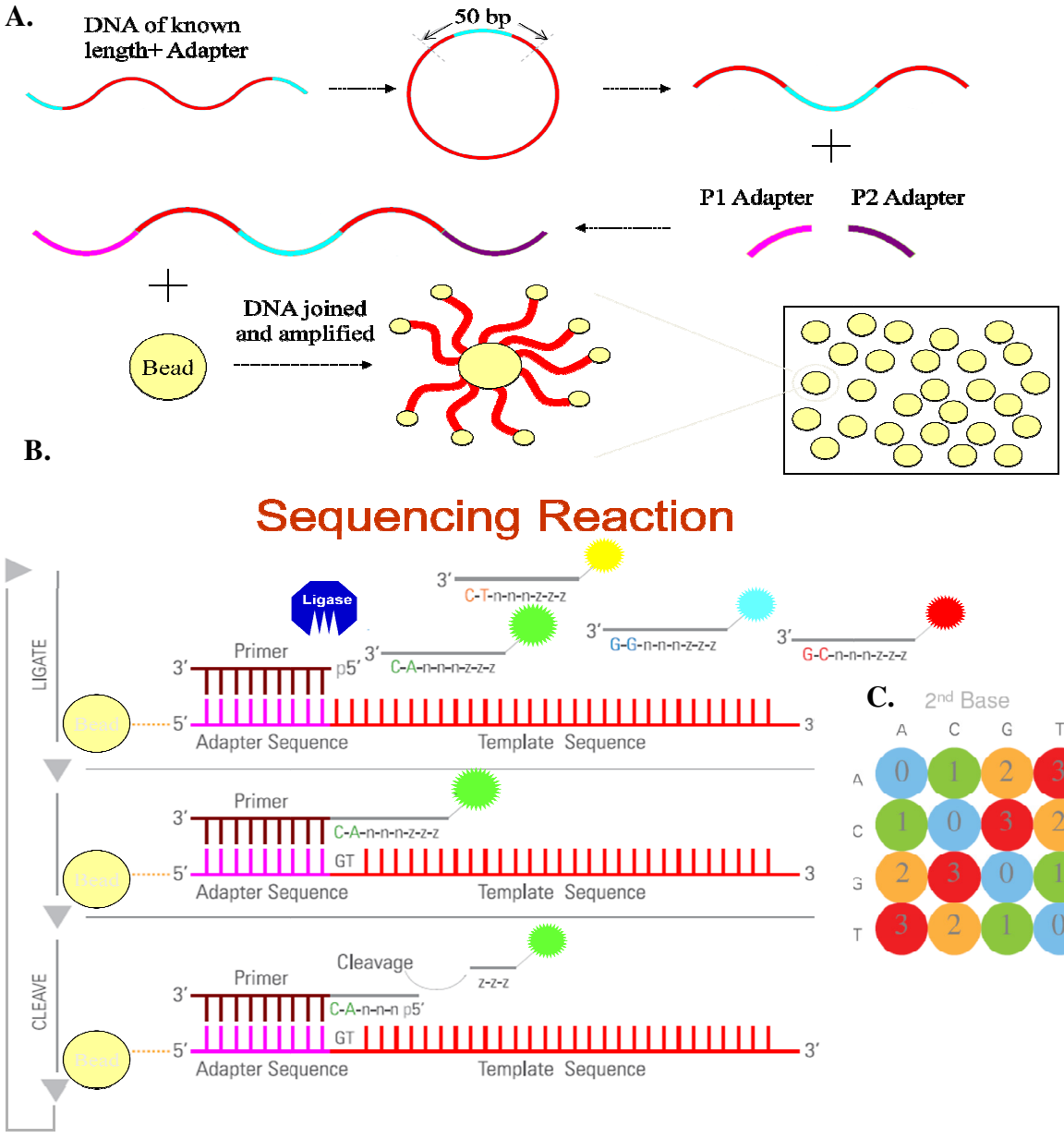


Figure 1.7: Applied Biosystems SOLiD Analyzer Sequencing by Ligation – A. Mate pair protocol and bead creation: The DNA is fragmented and circularized. The circles of appropriate length are selected. Adapters are ligated onto the sequence. The sequence is placed onto a bead and amplified. These beads are placed onto a slide for sequencing. **B. Sequencing:** Primers hybridize to the adapter sequence within the library template. 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. Multiple cycles of ligation, detection and cleavage are performed. **C. Di-base Transitions:** The SOLiD di-base transitions are also known as color space. (Adapted from ABI SOLiD Brochure)

1.3 The Third Generation of Sequencing Technology: Single Molecule Platforms:

The rate at with the second generation sequencing platforms has been increasing its sequences has been dramatic. And it is abundantly clear that this is still the beginning of what these platforms can do. More diverse sequencing experiments are being planned and attempted every day, but all of the methods mentioned above require the creation of a clonal library, a time intensive step that may also result in an unevenly developed library. Should this occur, this could potentially skew the sequencing results. This can cause havoc in *de novo* assembly projects. Sequences could be improperly assembled due to an imbalance in either direction, underrepresented sequences could be mis-categorized as an error and overrepresented sequences could be misconstrued as a repeated region. The goal of the third generations of sequencing technologies seeks to eliminate the need for a clonal library, which would not only avoid this potential pitfall but also increase the scale and reduce the cost of sequencing [21, 22]. The single molecule technologies have long read lengths, much longer than the HTS platforms. In addition, their major advantage is the direct sequencing of both DNA and RNA. This allows for sequencing with potentially smaller amounts of starting material and direct identification of methylation sites.

Some of these single molecule sequencing platforms under development include platforms by Helicos Biosciences, VisiGen Biotechnologies, Pacific Biosciences, Genovoxx, Life Technologies and others. While most of these systems have not been commercialized yet, a few of the Pacific Biosciences Single Molecule Real Time Technology are currently in use in various labs. Through various improvements and by

eliminating need for the clonal library, these companies hope to achieve the elusive goal of the $1,000 genome.

Chapter 2: Assembling the Genome

Assembling a genome *de novo* has often been compared with putting together a large jigsaw puzzle [23, 24]. Repetitive regions of the genome are likened to similarly colored pieces, similar to putting together the pieces of a green grass field. From the very beginning, sequence assembly was a computationally daunting task and it still continues to be. As the sequencing technologies mature and genomes attempted become more complex, the computational resources must become more refined to meet the demand.

2.1 Early Contig Assembly Algorithms:

The first assemblers appeared in the late 1980 and early 1990s. These initial assemblers were simplistic in that they were slightly enhanced sequence alignment programs which took advantage of the long read lengths and lack of complexity of the genome for assembly. With the dream of assembling more complex genome, sophisticated strategies needed to be employed to handle the massive quantities of data, address repetitive regions, and correct for the errors that occur using Sanger sequencing.

Not only did the assembling of the human genome revolutionize how sequencing of novel genomes was done, but it also revolutionized the analysis and assembly of the data. With the challenge of assembling Drosophila melanogaster and the human genome, Kececioglu and Meyers discuss an algorithm to tackle this complex problem [25]. They discuss that the sequence reconstruction problem is a variation of the shortest common superstring problem, therefore sequence reconstruction falls into the class of NP-hard problems for which no efficient computational solution can exist [24]. Sequence reconstruction is further complicated by the presence of sequencing errors and reverse

complements of fragments. In the end, the long Sanger reads as well as use of small mate-pair libraries aided in overcoming some of the complexity of the assembly problem [8, 10]. Their assembly algorithm employs creation of graph where the reads represent nodes and the edges represent the overlaps followed by a graph correcting mechanism to end up with a finished sequence. This algorithm became the core of the Celera Assembler [8] followed shortly thereafter by ARACHNE from Eric Lander's lab at MIT [26]. In addition, the complexities inherent to processing these assemblies for both Celera Genomics and the HGP required utilization of high performance computing in order to achieve the final output [10, 27].

The HGP took a decade to complete using high quality long reads. The assemblers that targeted use of Sanger reads are tuned specifically to take advantage of both the long lengths by allowing long overlaps between the reads and the low error rate. However, with the commercial availability of HTS platforms, hundreds of millions of short reads can be generated in a few days, while adequate computer resources to process them are not always available. The reads produced by HTS platforms require more intricate assemblies than ever before, as the reads are much shorter, and consequently have many ambiguous overlaps, are much more numerous O(100,000,000), and have significantly higher error rates. The assemblers from the past would not produce optimal assemblies on these new types of reads.

2.2 Current Contig Assembly Algorithms:

With this in mind, over the past few years several assemblers have been developed or modified to allow the assembly of short read sequences in order to make

assembly on moderate machines achievable. Most current assemblers have added (or include) algorithms for the assembly of mate pair/paired end short read sequences. All of the assemblers have a problem with long repeated regions and have different error correcting mechanism to avoid misassemblies. Due to the complexities and computational resources involved, most current assemblers target small to midsized genomes (150 kbp – 15 Mbp), although some large draft genomes have been successfully assembled. Li et al used SOAP*denovo* to assemble a draft giant panda genome (estimated size ~2.4 Gbp). While it is a draft, this genome is far from complete with fifty percent of the bases existing in contigs of ~40 kbp or larger, fifty percent of the bases existing in scaffolds of ~1.3 Mbp or larger and ~200,000 gaps [28].

There are two main core algorithms for assembly of sequence *de novo* - 3' kmer extension (Figure 2.1) and Eulerian walk algorithms (Figure 2.2). The 3' kmer extension family of algorithms is more closely related to each other than the Eulerian walk family's relationships. But there are still advantages and disadvantages to each of these three assemblers. The 3' kmer extension assemblers aim to extend a short read into the longest possible assembly through repetitive cycles of finding short read overlaps. The algorithm searches for reads that satisfy the overlap requirement, and if one is found, the read of interest is extended. The core idea or assumption is that if the overlap requirement is met, it is sufficient confirmation that the short reads are actually part of the same assembly.

The 3' kmer extension algorithm cycles through its various steps until all of the short reads have been incorporated into one of the contigs (Figure 2.1). In order to have the ability to search for the necessary reads, all of the short reads and their complements

are read into memory and placed into two searchable data structures. Loading these reads into the data structure allows for a simplified and speedy search for potential extension short reads. Before the cycle can commence, seed sequences are selected for extension from the short read data structure. Next, all the sequences that satisfy the minimal overlap requirements of the program are found. Once found, there are two options for extension depending on the program being used: either the consensus of all the overlapping reads is established and used for extension, or the longest perfect match is identified and the seed is extended by the rest of that particular read. As the reads are incorporated into a contig, they are removed from the data structure and cannot be used for extension elsewhere. The overlap search cycles through many 3' extensions until a failure occurs. If the 3' end cannot be continued, the complement is calculated and extension commences on the opposite end until failure. These steps are repeated with a new seed as necessary until all the short reads have been removed from the data structures.

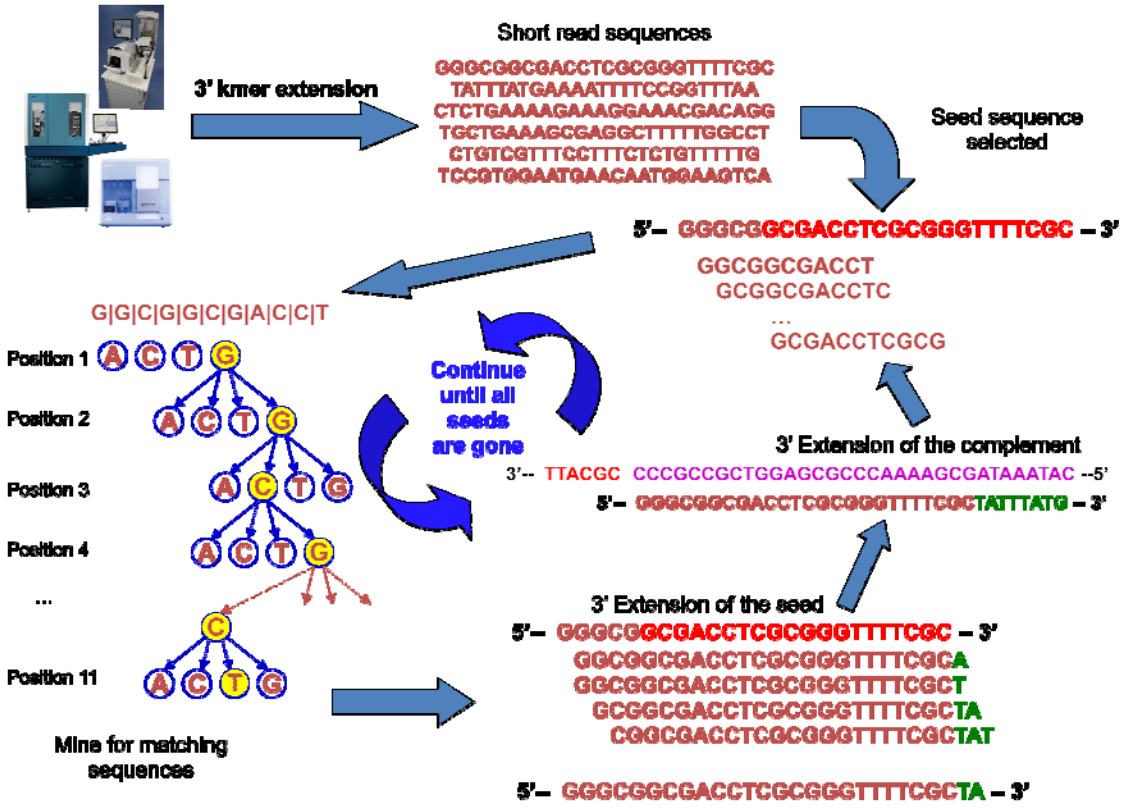


Figure 2.1: 3' Kmer Extension - This is the basic algorithm for all the 3' kmer extension assemblers. Step 1: Take the short reads and place them into a searchable data structure. Step 2: Select a seed sequence. Step 3: Find all sequences that satisfy the minimal overlap requirement (sequence in red). Step 4: Either find consensus for all the reads that match or find the longest perfect match and extend the seed. Continue 3' extension until failure to extend occurs. Step 5: Calculate the complement and start extension of the complement until failure occurs. Repeat steps 2-5 until all seeds have been removed.

In contrast to the core algorithms involved in 3' kmer assembly, the Eulerian walk algorithms initially place the short reads into a graph via various algorithms and then correct the graph until the final contigs emerge. These assemblers are loosely based on the assembly algorithms of the Celera Assembler [25]. While this family of graphing assemblers is not as closely related as the 3' extension algorithms, they do contain the same essential parts: kmer organization, graph creation, graph simplification, error correction, and finally assembly/output (Figure 2.2). The nodes of the graphs can either

be kmers that make up the sequences or the short reads themselves. The node sequences have a length L and are organized into a searchable data structure. For graph creation, the nodes are organized and edges represent overlaps between the sequences within the nodes. A path exists if there is an overlap of L-1 between two kmers, or the overlap criteria is met. For graph simplification, all unique paths are condensed into a continuous sequence. The algorithms will try and correct all erroneous paths and errors that it can identify through both topological features of the graph as well as identifying low coverage paths. Finally, once all that is done, the final assembly is output. Once all errors are corrected, it is presumed that the paths can be walked to find the correct assembly.

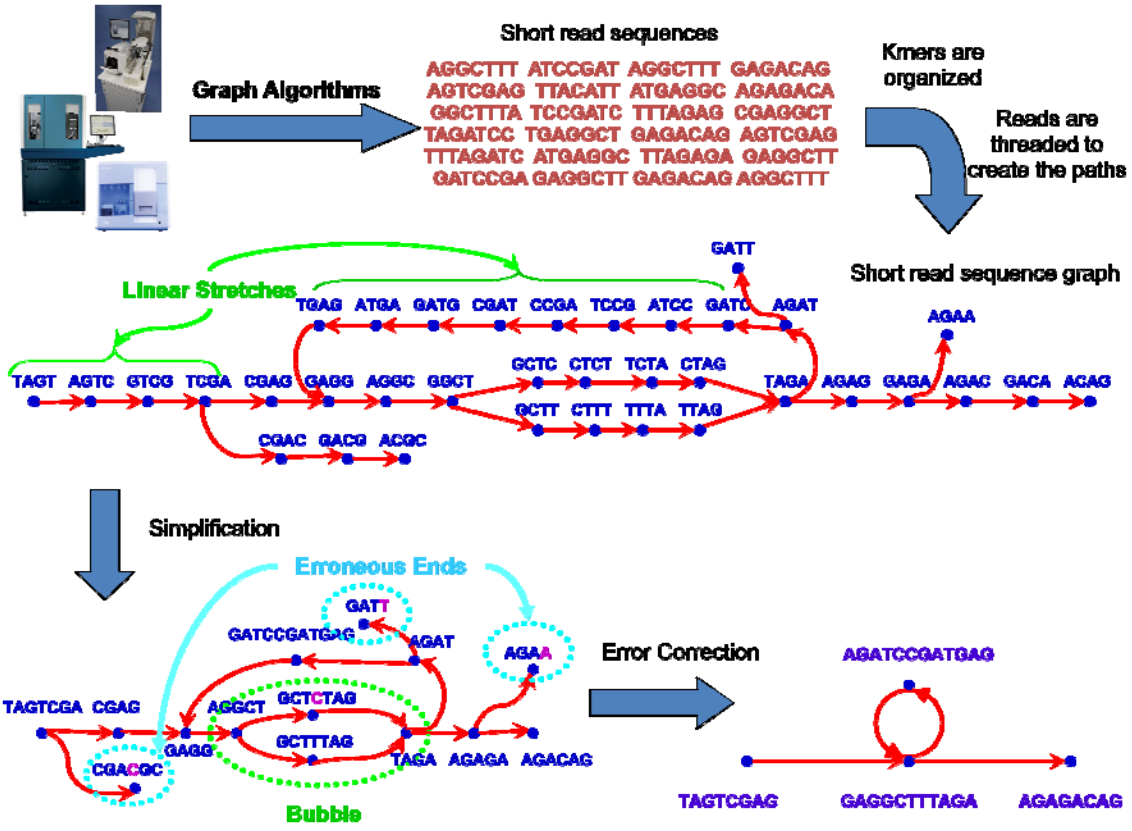


Figure 2.2: The Eulerian walk algorithm - These algorithms vary more greatly then the 3' kmer extension algorithms, but they involve the same basic processes in slightly various ways. Step 1: The short reads sequences and the kmers, length L, that make up the sequences are organized into a searchable data structure. Step 2: Graph Creation - The kmers are organized and the reads are threads through to create paths between the kmers. A path exists if there is an overlap of L-1 between 2 kmers. Step 3: Graph simplification – All unique paths are condensed into 1 sequence. Step 4: Error correction – erroneous paths and errors are corrected. Step 5: Output the assembly.

The 3' kmer extension family is very good at providing large contigs, but is highly sensitive to sequencing errors and slight variations in repetitive regions. These types of sensitivities lead to misassemblies in the genome. Preprocessing of these short read sequences becomes critical. In contrast, the Eulerian walk algorithms have the advantage of being able to "visualize" all the connections between the sequences and use established graphical manipulation algorithms to correct the graph and therefore the assembly. As a result, misassemblies and repetitive regions are easier to pick out, correct

and/or simplify. But creation of the sequence graph and error correcting are not trivial tasks, and require a significant amount of memory. To their advantage, the Eulerian walk algorithms have the advantage of a faster run-time then the 3' kmer graph extension algorithms.

2.3 Current Assembly Algorithms in Detail:

The assemblers are constantly being refined and invented by researchers interested in the assembly problem. Some of the newest innovations to these assemblers are adapting older short read assembly techniques into parallelizable programs. While new assemblers are always appearing they almost always fit into one of the two categories mentioned above. Below is a discussion in detail of some of the most popular assemblers for short read sequences available.

The assemblers that make up the 3' kmer extension family are SSAKE [29], VCAKE [30], and SHARGCS [31]. The difference between these three is in the manner in which they handle error prone short read sequences to optimize the assembly. While the overall algorithm is the same, the extension details are slightly different in all three programs. Both SSAKE and VCAKE find all overlapping sequences, from longest to shortest (where shortest equals a predefined minimum), and then gets a consensus for the extension. SSAKE will extend by the largest number of bases possible in the consensus, but VCAKE will only extend 1 bp at a time before repeating the search for extension. SHARGCS finds the longest perfect read and extends the sequence by the end of the read. Since short read extension is so sensitive to error, in order to improve the assembly both SSAKE and SHARCGS recommend a preprocessing step to remove the low quality

reads. SHARCGS also contains a very interesting feature of being able to combine multiple runs at different parameter settings to aid in confidence in assembly. The reasons for a failed extension are dependent on the stringency of the assembly parameters chosen. Breaks caused by weak stringency/loose assembly parameters are usually a result of ambiguities, while under strong stringency/conservative assembly perameters, these breaks are a result of lack of coverage [31]. Therefore, being able to combine multiple runs is a benefit to the confidence in SHARCGS assembly algorithm. VCAKE contains another unique feature. If sufficient perfect matching reads is not achieved, VCAKE will search for more short reads containing a single mismatch after the 10^{th} base in the short read. Both SHARGCS and VCAKE do not have modules for dealing with mate pair reads. Even though SSAKE has added modules to its 3' kmer extension algorithm to handle mate pair reads, it does not handle them optimally. SSAKE will only extend the scaffold one hop in either direction, which is a severe disadvantage.

In contrast to the core algorithms involved in 3' kmer assembly, the Eulerian walk algorithms all contain the same essential parts: kmer/read organization, graph creation, graph simplification, error correction and finally assembly/output (Figure 2.2). For example, for the generation of the kmer graph and analysis, the short read sequences and the kmers, of length L, that make up the sequences are organized into a searchable data structure. For the graph creation, the kmers are organized and the reads are threaded through to create paths between the kmers. A path exists if there is an overlap of L-1 between two kmers. For graph simplification, all unique paths are condensed into a continuous sequence. The algorithms then will try and correct all erroneous paths and errors that it can identify through both topological features of the graph as well as

identifying low coverage paths. Repetitive regions are identified by cycles in the graph. Once all errors are corrected and repeats are identified, it is presumed that the paths can be walked to find the correct assembly. Finally, once all that is done, the final assembly is outputted.

Velvet [32], EULER-USR [33, 34], Edena [35], SOAP [36], ABySS [37], and ALLPATHS [38] comprise part of the family of graphical algorithms. The principal differences between these assemblers are how they create the graph and then proceed with graph correction and simplification. One caveat is that these assemblers have the potential to be very sensitive to uneven distribution in the sequencing of the genome. This weakness is generally a result of its error correction algorithm, with highly uneven sequencing; true edges can be deleted, reducing the sizes of the contigs.

Velvet builds the a kmer graph using an Idury/Waterman/Pevzner model without doing any initial error correction [39, 40]. Velvet then proceeds to do three consecutive levels of error correcting. The EULER-USR generalizes the Velvet model by allowing for mismatches in the paths of the kmer graph [33, 34]. It then does maximum branching optimization to remove erroneous edges. Finally in EULER-USR, there is a possibility for low coverage areas to be given a second chance to be used in the final assembly, even though they were initially eliminated. Edena [35], like Velvet, does not include initial error preprocessing prior to graph creation; unlike Velvet, Edena uses the overlap-layout-consensus approach to generate the sequence graph where the nodes are the reads and edges exist if the overlap criteria is met [25]. After graph creation but prior to assembly, Edena does graph correction. During assembly it does several phases of error correction to achieve the final graph.

SOAP [36] follows a similar kmer graph creation scheme as Velvet and EULER-USR but restricts the kmer length to odd numbers from thirteen to thirty-one. While larger kmers would allow for higher confidence since the rate of uniqueness would be higher and therefore make a simpler graph, SOAP restricts the length since it would require a higher sequencing depth and longer read length for a successful assembly. In addition, SOAP builds its scaffolds using the smallest insert size first.

ABySS is built to natively use distributed computing [37]. It essentially mimics EULER-USR for a parallel computing environment. ALLPATHS [38] tries to find all paths from one read to a second read covered by other reads, and then it attempts to isolate small parts of the genome to assemble these segments independently. ALLPATHS assembles local sequences first and then aggregates these local assemblies into the master assembly. In addition, ALLPATHS does not return a single assembly; this program returns all possible assemblies including any ambiguities that exist. All but Edena use mate-pair/paired end information to simplify the graph and optimize the assembly.

2.4 Comparing Short Read Assemblers:

In order to compare these assemblers, some common metrics have been developed: the N50, percent of the genome covered through contig alignment, the longest contig assembled and perfect, error-free contig alignment. The N50 contig size is a standard measure used by all of the assemblers to define their success. An N50 contig size means that half of all bases reside in contigs of this size or longer. Another critical consideration is the genome used for the test, error rates, length of both the short reads

and genome, redundancy (fold coverage) and repetition of the genome, simulation and/or real short read data, and the computing power that ran these algorithms. These metrics are very dependent on the genome being assembled. It is very difficult to compare these metrics between sources, since the basis can be vastly different.

In a recent paper, attempts to use most of these assemblers with a variety of parameters were made on the same dataset, facilitating comparison [41]. ALLPATHS could not be used due to a production version not being available. EULER-SR and SHARCGS regularly ran out of RAM (max available 32 GB) during their assembly runs. EULER-USR was not used in this set of assemblies and seems to have corrected the problem of importing large datasets.

The data involved in this experiment came from Illumina sequencing of the 6Mb *Pseudomonas syringae* pv *syringae* B728a genome. The sequencing run produced 3,551,133 mate pairs with 36 bp reads and an average of 400 bp linker lengths. This translates to ~255 Mb (42x) of coverage of the 6Mb genome. For paired end read assembly only SSAKE and Velvet were compared, but initially SSAKE, VCAKE, and Velvet were all compared without the additional mate pair information, essentially treating them as ~7 million independent short read sequences. Velvet assemblies took several minutes to run while VCAKE and SSAKE required days to complete. VCAKE and SSAKE generated assemblies with a large number of errors primarily due to assembling noncontiguous regions into a single contig. Velvet had the optimal balance between length and accuracy (N50 = 6,963 nucleotides) with errors amounting to 0.2% of the assembly. On simulated reads, a test that removed potential errors from the Illumina

sequencing, longer contigs were yielded (N50 = 13,771 nucleotides), but still at minimum 2.5% of the genome was still not resolvable.

For the paired end reads, SSAKE yielded relatively short contigs with high error rates (N50 > 1.5 kb and errors rates > 47%). This was essentially no better than the unpaired assemblies. Velvet, on the other hand, yielded longer contigs (N50 > 15.6 kb) with a higher error rates. Optimization of parameters yielded a better balance, shrinking both the contig size (N50 = 12.5 kb) and the error rate (0.5% of the length of the assembly). In addition to this initial study, they discovered, as expected, fold coverage plays an important role in assembly. However, after 35 fold deep coverage the rate of contig extension declines drastically. Finally, they concluded that the parameters selected for all of the assembly programs play a critical role in the length of the contigs.

2.5 Building Scaffolds with Mate Pair/Paired End Sequences:

The complete draft genomes of Drosophila and Human were not achieved using Sanger fragment reads alone. Small mate-pair libraries were critical in this effort in order to help resolve some of the sequencing ambiguities, such as sequencing repetitive regions [8, 10]. The conclusion was that scaffold assembly is essential for complex genome assembly. From the beginning, the incorporation of information contained in mate pair data has either been addressed concurrently with contig assembly or as an independent scaffolding module [42-44]. Current scaffolding algorithms generally fall into two classes. The de Bruijn graph based assemblers, such as Velvet, utilize mate pairs to improve the walk in the same de Bruijn graph used for contigs assembly. Basically, the paired reads add confidence to the final assembly. The second class formulates the

scaffolding problem in terms of a graph. The vertices are associated with the assembled contigs and edges represent the mate pair linkers [32, 42, 44]. The design of an independent scaffolding module, like Bambus, is seen to allow greater flexibility in the algorithms and additional control over the scaffolding process [44].

Most early scaffolding algorithms follow a greedy approach. Initially, there is some scheme required to order the contigs and pairing information for use [42, 44]. Next, the mate pairs are iteratively incorporated as long as this does not conflict with the previously assembled scaffolds. Essentially, with iteration, only a subset of contigs and their corresponding links are considered. In addition, the ordering scheme, since it is usually based on the number of links between two contigs, could potentially cause improper scaffolding by incorporating repeats/chimeric contigs that have a significant number of links associated with it. Knowing the advantages and disadvantages of short read data, this type of solution faces difficulties [23].

Recently a new scaffolding module has joined the ranks to aid in genome assembly. SOPRA is a scaffolding algorithm that corrects contig assembly and builds scaffolds using statistical optimization [45]. SOPRA was designed to handle the unique challenges inherent in scaffolding using short read data. Unlike previous scaffolding algorithms, SOPRA takes a global approach to linker analysis. SOPRA's goal is to select a sufficiently large subset of mate pair constraints that will achieve a balance between size and quality of the final assembly. In SOPRA, scaffold assembly is presented as an optimization problem for variables associated with the contig connectivity graph. The error-prone nature of HTS data and the fundamental limitations from the shortness of the reads have likely lead to questionable assemblies. SOPRA attempts to circumvent this

problem by treating all the constraints alike in order to solve the optimization problem. The solution itself indicates the problems, chimeric/repetitive contigs, etc., which are subsequently removed. This process iterates until a core set of consistent constraints is reached.

The core algorithm of SOPRA works for both DNA space and color space short read sequences. SOPRA has an additional key feature for the SOLiD HTS platform should it be used as part of its prescribed pipeline. The pipeline includes using S-SOPRA, a color space version of SSAKE or V-SOPRA, the color space version of Velvet, and additional programs for read tracking. If used within these pipelines, SOPRA is able to use a dynamic programming approach to robustly translate the color-space assembly to base-space by keeping track of where the reads assembled and the reference base associated with the read [45].

2.6 Assembly Conclusion

While all of these assemblers and scaffolders are attempting to do the same thing, assemble a genome *de novo*, there is still room for improvement. All of these assemblers are computationally quite taxing on computer systems, and rarely do they output a single genomic sequence from short reads. Both memory costs and run time remain issues. In addition, currently none of these systems can easily handle large repetitive genomes assembly optimally and only Velvet, S-SOPRA, and V-SOPRA can handle the assembly of color space reads of the SOLiD system. In contrast, algorithms are constantly being advanced, new programs are continually being written, computers are consistently getting faster, memory is consistently getting cheaper, and new processor technologies are being

experimented on specifically to optimize assembly. Some assemblers are being written or modified to merge the pipelines together using mate pairs from the beginning to aid in the analysis (Phusion [46] and ALLPATHS [38]). In order to better understand the assemblers' final output, one must explore all aspects of contig building, short reads, and mate pairs.

Chapter 3: Assembly Theory – The Lander and Waterman Calculations

Now that the second generation technologies are available and are able to deliver data at dramatically lower costs and at substantially higher coverage than the previous generation of technologies, the problem remains of how to assemble the short read sequences into a useful DNA sequence. Unfortunately, while these technologies yield high coverage, they also yield very short reads (35-500 bases), a significant limitation for assembly. After understanding the current group of short read assemblers, including their respective strengths and weaknesses, a statistical understanding of genome assembly is needed. In a very influential paper published over a decade ago, Lander and Waterman developed a theory for fingerprinting using restriction fragment lengths [47]. The original paper pointed out that this theory could be applied to shotgun sequencing and therefore able to give some statistical boundary for the data from the HTS platforms. Lander and Waterman's theory provides estimates of the expected size of the assembled contigs given the conditions for detecting overlaps and the depth of sequence coverage.

The following inputs are critical for Lander and Waterman's statistical analysis. A) the DNA sequences: there are N DNA fragments of length L that are randomly placed on a genome of length G; B) the assembly: if two of these fragments overlap by a length greater than or equal to the threshold T, a true overlap would be inferred from the significance of sequence similarity. There are two crucial parameters in this model derived from the inputs above: $c = NL/G$, equivalent to the redundancy of coverage, and $\sigma = 1 - T/L$. One of the significant results of the Lander and Waterman analysis is that the average length of a contig is given by $L[((e^{c\sigma} - 1)/c) + (1 - \sigma)]$.

While Lander and Waterman derived this result in the region where the number of clones N is much smaller than the genome size G, massively parallel sequencing functions in a very different range of parameters. The length of the read L is of the order of 50 bp, versus 1 kbp available by Sanger sequencing, but the number of reads from a single run of a machine, N, is on the order of a few hundred million. With these considerations, unlike in the original Lander and Waterman calculations, N/G is not very small. In addition, T, which is somewhere between 16 and 35 depending upon the genome size, is not much smaller than L. In this limit, the above formula for the average size of the contig has to be replaced by $(e^{c\sigma} - 1)/(1 - e^{-N/G}) + L(1-\sigma)$, which is the same as $(e^{c\sigma} - 1)/(1 - e^{-c/L}) + L(1-\sigma)$. With the current short read sequence parameters, the average size of contigs is significantly smaller than most of the genomes of interest, O(1,000 bp). For example, with 40 fold coverage of the genome, sequences of 25 bp in length and allowing for a 20 bp overlap, the calculations yield a theoretical average of approximately 3,700 bp long. Thus, this is a fundamental roadblock to sequence assembly from traditional short reads.

One way to overcome the problems posed by shorter contigs is to increase the depth of sequencing. Greater coverage depths are affordable (50-200x instead of 2-10x) using the current sequencing technologies, but *de novo* sequence assembly with these simple shorter sequences is significantly more complex, as is suggested by the Lander and Waterman estimates. Since extension is much smaller than the overlap, the contigs do not grow quickly. This is one of the reasons the expected size for the contigs are so low.

With all these considerations, the question then arises: can an accurate genome assembly be computed at acceptable computational costs *de novo*? To answer this question, one must take a detailed look at all the potential factors and how they interplay with each other Some potential factors include errors, repetitive regions, non-uniform distribution of sequencing libraries, and ambiguity in extension,. All of these factors are not accounted for in the Lander and Waterman's calculations.

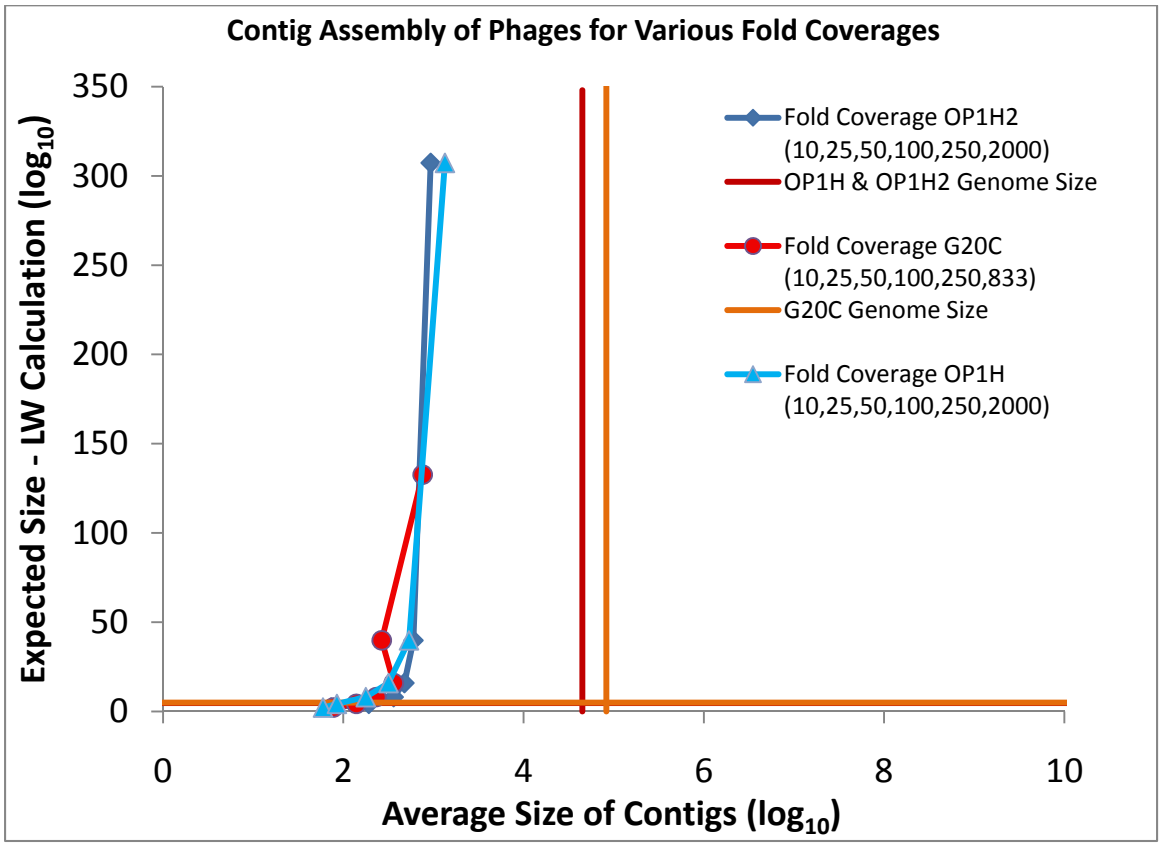


Figure 3.1: Contig Assembly vs. the Ideal - Here shows a comparison of the average contig length from a real assembly versus the expected contig length calculated according to the equations proposed by Lander and Waterman. As one can see, Real assembly is very far from reaching the estimated size.

Chapter 4: High Throughput Sequences and Short Read Issues

From the above discussion, it is clear what the ideal assembly should be. An understanding of the parameters that keep the assembly from reaching its true potential is now needed. The long reads of Sanger sequences were able to overcome some of the complexity involved in large assembly attempts, but they also had some issues. Sanger sequencing usually requires prior knowledge of a small section of sequence that the HTS platforms do not. From this oligonucleotide, sequencing commences. In addition, with Sanger sequencing, DNA is usually cloned before sequencing and therefore, sometimes contains parts of the cloning vector, which can lead to significant errors in *de novo* assembly. This step is eliminated in all HTS platforms. With the advantages of lower cost and library preparation come significant shortcomings: the sequences are significantly shorter and of lesser quality. As the read length decreases, the assemblies become highly fragmented and ultimately lower the quality of the final assembly [48, 49]. Therefore taking into account all issues and trying to compensate for them becomes critical with short read sequence assembly.

Once only achievable at large sequencing centers, these HTS platforms now make sequencing and sequence dependant projects available to many researchers and labs. The sequences from the HTS platforms are very different than Sanger Sequences; the old analysis tools are unable to accurately process this data. Understanding in detail the advantages and disadvantages posed by the sequences from the HTS platforms is a crucial step for optimized analysis of these short read sequences.

4.1 Sequencing Technology Issues

Library Preparation:

HTS platforms contain some elements that can introduce bias at the library level. The two library preparation steps that can potentially introduce bias are: the sonication and polymerase chain reaction (PCR) amplification steps. The sonication shears the DNA into smaller fragments for sequencing. These fragments are then run through a size selection gel in order to isolate sequences of the proper length to sequence. Normally, sonication produces random shearing of the DNA. However, under certain conditions, sonication seems to shear A, T rich regions preferentially [50]. Some researchers have seen evidence of these biases in HTS reads [51]. Preferential shearing can result in non-uniform representation of specific regions of the genome within the size selecting band. The PCR step is used to amplify the DNA to usable quantities for sequencing. It is possible that specific sequence motifs can amplify preferentially and lead to an unbalanced library. Even a slight bias, in either sonication or PCR amplification, can have noticeable effects on an assembly since the biases affect the uniformity of the coverage of the genome. Most assemblers assume a uniform coverage distribution, so large deviations from the estimated coverage can potentially cause faulty assemblies.

Quality:

Another significant consideration for the HTS platforms is the reduction in quality of the sequences produced compared to Sanger sequencing. Like all sequencing platforms, Sanger included, the number of errors grows exponentially toward the end of the read, but with Sanger you can locate long sections (700-1000 bp) that are mostly free from error. In addition, the long overlaps that can be used to align two Sanger reads are

adequate. However, the sequences from the HTS platforms do not allow for long overlaps. In addition, errors can potentially occur throughout the read; although, like Sanger sequences, the number of errors grows exponentially towards the end of the read [52]. While the sequences can be truncated to remove the increased number of errors at the end, the reads still remain short (30-100 bp) and can still contain some error. These errors can lead to misassemblies, early termination of contig assembly, and added complexity in resolving repeated regions. Dealing with these errors is important for resequencing and is essential for *de novo* assembly.

Paired Short Reads:

The availability of large paired read libraries is an important and fairly recent development. While a tiny percentage of the reads used in the drosophila and human projects were paired Sanger reads, they made a noticeable impact in aiding the assembly [8, 10]. The additional information offered, the known approximate distance between the reads, from large mate pair/paired end short read sequence libraries is critical to overcome the high fragmentation found using solely the short read sequences [34, 45].

All three short read sequencing platforms have introduced the ability to generate mate pair and/or paired end reads (Illumina and SOLiD both MP & PE; 454 MP only) (Figure 4.1). Paired end libraries require equivalent amounts of start DNA as fragment libraries and cost the same to prepare. The difference between the two types of paired reads is the library preparation. For paired end reads, there is no circularization step. Essentially, a paired end read is a long fragment where the DNA is sequenced on both ends. The standard paired end read protocols limit the size of the fragment to 500 bp, and therefore, the starting DNA requirement is similar to that of a fragment library

preparation. For mate pair read, the DNA of the desired length is circularized with an adapter in the center. The DNA that is sequenced flanks the internal adapter. The DNA required for the library preparation is a factor of the size of the circularized DNA. The longer the linker, the more DNA is required to prepare the library. Since circularization is not very efficient, up to four times more DNA is required. This is especially true for the long linker lengths. In order to generate mate pairs for sequencing, more DNA is not always possible to obtain and, therefore, can remove mate pairs as a possibility. Plus, the cost involved for preparation and sequencing the mate pair libraries is approximately 60% higher. Thus having long mate pair reads is not always an option for assembly.

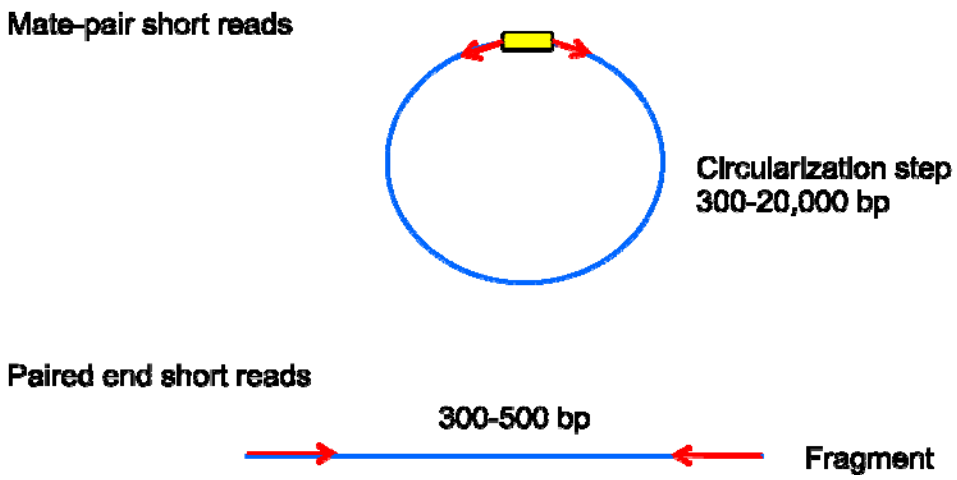


Figure 4.1: Fundamental Difference between Mate Pair and Paired End Reads – Mate pairs require circularization of the DNA and have longer linker lengths. Paired end reads are generated by sequencing two ends of a long fragment.

Each short read found at the ends of the paired read contains the same error profiles as a single, independent short read. But, mate pair reads have an additional factor to consider: relatively high variability in linker length [45]. This is mostly a result of the size selection process; the DNA is moved through a gel and then the desired sized pieces are selected. The longer the linker desired; the closer the DNA fragments run together on the gel. Therefore, even with extreme care, the variability introduced can be high. Smaller linker lengths separate with larger gaps so it is easier to select mostly homogeneous lengths.

Platform Specific:

The problems mentioned above are typical of the HTS platforms in general, but since sequencing protocols on each machine differ, platform specific problems need to be considered as well. For example, the 454 technology does not use chain termination. Therefore, in long homopolymeric regions, stretches of sequence containing greater than six copies of a single nucleotide in a row, the sequencer had difficulty quantizing the length of the region. In the third generation sequencers, the high error rates are kept in check through sequencing the same strand multiple times [23]. Finally with the ABI SOLiD platform there are two considerations. The SOLiD platform can have issues sequencing regions of the genome with high A, T content and, while not a chemistry problem, color space is a factor that must be considered. SOLiD sequencing is reported in color space, a representation of two-base encoding. In resequencing projects, this aids in the differentiation between errors and single nucleotide polymorphisms (SNPs). But in other cases, a naive translation from color space to base space can lead to serious error

amplification [45, 53]. Therefore, it becomes critical for the sequences to remain in color space for assembly in addition to having a robust translation mechanism [45].

4.2 Genomic Technology Issues

The shorter read length produced by the HTS platforms has always proved problematic. As the length of each individual read decreases, the probability of non-uniqueness increases thus resulting in more fragmented assemblies (Table 4.1) [38, 49]. Since the reads cannot bridge long repetitive regions, the short read length is particularly problematic for those regions, and as a result, the true length is difficult to target. These long repeats occur more often as the complexity of the genome increases [54]. Even if perfect reads are used for fragment assembly, between the additional ambiguity and inability to accurately resolve repeats, the assembly ends up being highly fragmented [23, 48, 49].

Kmer Length K	*E. coli*	*S. cerevisiae*	*A. thaliana*	*H. sapiens*
200	0.063	0.26	0.053	0.18
160	0.068	0.31	0.064	0.49
120	0.074	0.39	0.086	1.7
80	0.082	0.49	0.15	7.2
60	0.088	0.58	0.27	18.0
50	0.091	0.63	0.39	32.0
40	0.095	0.69	0.65	78.0
30	0.11	0.77	1.5	330.0
20	0.15	1.0	5.7	2,100.0
10	18.0	63.8	880.0	40,000.0

Table 4.1: Fraction of Kmers with Unique Placement on the Genome - For a given k and a genome, this table shows the fraction of kmers having a unique placement. (Adapted from [38])

4.3 Data Management

With the Illumina HiSeq 2000 and the ABI SOLiD 4 and 4hq Systems producing O(100 million) reads per run, data management and storage has become an increasingly more complex issue. The increase in throughput is what makes certain types of sequencing experiments possible at acceptable costs, but they also threaten to inundate available computing resources. The Illumina HiSeq 2000 produces enough sequence to generate approximately 30 fold coverage for two human genomes with a single run. For Illumina, the images are not analyzed on the machine, they are analyzed on its dedicated cluster. Therefore, bandwidth and transfer speed become critical for the Illumina sequencing run since image files are quite large. Once the images are processed, the actual images are erased and the files are stored in binary to keep the storage footprint small. With a single double slide run of the current ABI SOLiD platform (4), enough sequence is generated to cover the human genome thirty fold. Within the next few releases, ABI projects that the number of beads, and therefore reads, which can be deposited on a slide will increase to ~1 billion beads. This will be achieved by using smaller beads and semi-ordered arrays. Forty-five terabytes of data is expected as the throughput from the machine per month, making accurate analysis and adequate storage capabilities even more critical than before.

As the data becomes more abundant and more dense, standard analysis schemes will be overwhelmed [23]. Efficient ways of storing, transporting, and analyzing these reads will be required. Parallel implementations and specialized hardware, high RAM machines [23, 24] and GPU specific software [55], are beginning to be used to speed up and make analysis possible.

In addition to routine data management problems involved with sequence libraries, *de novo* assembly adds it own computational requirements for analysis. *De novo* assembly is a much more computationally intensive task. Comparative assemblies or analyses that require a reference genome, while not trivial, does significantly simplify the task [23]. Aligning the reads along the genome is inherently an error correcting mechanism; those reads that contain a certain number of errors will not align. With the aligned reads, a wider margin of non-uniform coverage can be used to assemble genomes. These differences in coverage are also critical for other types of sequencing experiments and analysis, ChIP-Seq, methylation analysis, gene expression profiling, small RNA Analysis, to name a few. For true *de novo* assembly, the non uniform distribution becomes a liability: it can potentially cause contigs to break or misassemblies to form [45]. This can potentially be somewhat overcome by using paired reads, which can pose significant algorithmic obstacles for assembly themselves, i.e. keeping track of the mates, and disentanglement of misplaced mate-pair ends [45].

All of these aspects must be taken into account when attempting *de novo* assemblies. There are many different hurdles to overcome in order to optimize assembly. While not simple in the most trivial of cases, understanding the interplay between all these forces becomes a critical and increasingly difficult as the genome becomes more complex. As a result, techniques have to be addressed and improved upon to manipulate these HTS libraries in order to overcome some of these obstacles prior to assembly.

.Chapter 5: Preprocessing Short Read Sequences

Both Illumina and ABI SOLiD claim the accuracy of their sequences are above 99.9%; this number is highly misleading. The 99.9% accuracy rate is based on direct resequencing projects for monoploid organisms with at least some fold coverage, i.e. based only on the aligned reads without including any of the reads that were thrown out. Mismatches/Errors can be identified and dealt with once the short read sequences have been aligned. The maximum number of allowable mismatches/errors is a parameter given to the alignment program. So for a *de novo* project with no orthologous sequence, identifying true reads is almost impossible since sequences cannot be selected for against an orthologous reference genome. Even orthologous genomes will sample only a subset of the true reads due to SNPs, insertions, deletions, rearrangements, and evolutionary divergence of sections in the genome of interest. Since the quality of the library is critical for assembly, how can erroneous reads be removed in order to lower the probability of misassembly? The answer is critical for many of the assemblers mentioned above: preprocessing the library.

All the current HTS technologies produce a sequence and an estimate of the quality of the data. Quality scores are calculated on a per color call/base call basis. These quality scores are calculated by training the sequencing process parameters against several annotated datasets [52]. For example, the ABI SOLiD platform process parameters are image intensity, noise to signal value, and angle. The quality scores are in the form of a phred quality value, essentially the logarithm of the probability that a particular call was inaccurately identified [56]. In essence, the higher the quality score, the higher the confidence in the accuracy of that call. While one can use these scores in

the assemblers, most do not due to the additional complexity involved in the incorporation. Most rely on removal of the low quality reads prior to assembly and/or their error correcting mechanisms. Critical for the ABI SOLiD platform is the lack of prefiltering of low quality reads from the instrument; the SOLiD platform only reports the raw reads with their related quality values. One of the reasons ABI SOLiD allows all reads to pass, where the two-base transition can be identified, even if with poor quality, is that all such reads could be useful. As such, these reads are reported after primary analysis if desired; this then allows the researcher to decide the filtering parameters prior to post analysis. In most cases, the first step for secondary analysis is alignment of the short reads along the reference.

Even though Illumina does an inherent prefiltering step, since assembly is highly sensitive to sequencing errors, it becomes critical to mitigate potential errors prior to assembly. Therefore prefiltering becomes critical for all HTS platforms. There are two types of sequencing errors commonly observed: polyclonal/correlated errors and independent, erroneous color calls [57]. Polyclonal and correlated errors occur when the entire read is of poor quality or missequenced due to a bead level/cluster problem such as in a polyclonal bead/cluster or poor resolution of a particular bead/cluster. A polyclonal bead/cluster occurs when two different templates are amplified on a single bead or in a cluster and then sequenced, resulting in a hybrid sequence that has no match in the true genome. While the original goal was to identify polyclonal beads, there is no guarantee that all the reads identified by this part of the filter are due to polyclonality. A more robust filtering system using the information gathered during the imaging and processing of the sequencing run, i.e. image intensity, noise to signal and angle, could be designed to

distinguish between polyclonal beads versus other types of correlated errors. Single color call/ base call errors are independent and can occur multiple times in the sequence also leading to an inaccurate sequence.

Here, we developed a filtering framework, specific for the ABI SOLiD platform that attempts to optimize the preprocessing step by identification and removal of error prone reads using the quality values (QVs) provided from the SOLiD's primary analysis or the SOLiD Accuracy Enhancement Tool (SAET) modified primary analysis datasets. SAET uses the raw data and the information contained in the two base encoding to correct the miscalls (http://solidsoftwaretools.com/gf/project/saet/). This tool and its use for *de novo* assembly will be further discussed in Chapter 8. This filtering algorithm flexibly targets the two different types of errors that can occur during SOLiD sequencing. The ultimate goal of the preprocessor is to eliminate the low quality reads and pass only the high quality data into downstream applications, thus saving time and resources and improving final output quality. (All further data and discussion in this chapter focus on the SOLiD HTS platform.)

5.1 Tailoring of Error Identification:

To identify sequencing reads with either polyclonal calls or miscalls, we utilized several resequencing datasets where few mismatches were expected between the reference sequence and the sequencing reads. Using these datasets, profiles for both polyclonal/correlated errors and erroneous color calls were determined through various QV analyses. Using the SOLiD mapping pipeline, CoronaLite available from ABI (http://solidsoftwaretools.com/gf/), the reads were matched to the reference sequence, and

similar to other HTS platforms, the number of errors increased toward the 3’ end [34]. Fewer errors occur at the 5’ end of the read irrespective of the genome being sequenced (*E. coli, DH10B*; Arabidopsis, AtCol; or human, Hu3) (Figure 5.2A). The lower error rate between DH10B/AtCol and Hu3 is due to a difference in sequencing chemistry (SOLiD v2 vs. v3) (DH10B data available for download at http://solidsoftwaretools.com/gf/project/ecoli2x50/). From the graph plotting error as a function of QV, it was very clear that color calls with QVs of ten or less had a higher probability of being erroneous (Figure 5.2B). With a QV of ten or less, the SOLiD indicates the probability of error for that color call is 10% or higher. Single color call errors occur randomly throughout the sequence. Polyclonal beads seem to reflect subpar color calls all throughout the read, i.e. lower than expected QV values at the 5’ end of the sequenced read. Analysis of the quality values show early color calls can be highly predictive for the remainder of the read (Figure 5.1). Therefore, the filter polyclonal analysis focuses on the quality of the first ten color calls, requiring that some portion of them be of high quality (QV>=25). Further details of the script are found in the methods subsection.

For both Illumina and SOLiD platform short reads, one way to improve the quality of datasets is to trim the ends of the reads, which essentially removes the error tails [34, 52, 58]. While trimming the tails is effective at removing the most error prone regions, it does affect the length of the read and, therefore, the optimal extension according to Lander and Waterman [47]. Short contigs are a direct factor of the read length and the overlap size. The Illumina platform, while still providing fractured assemblies handles truncation better than the SOLiD platform since reads remain

relatively long after being trimmed. For example, with read lengths of 150 bp even if one removes 33-50% of the read, the Illumina read is still longer than the SOLiD raw read directly off the machine [34, 59]. The error tails between the two platforms are very similar and, as such, one would expect a slightly more fragmented assembly from SOLiD reads. One of the most important reasons to sequence DNA on the SOLID HTS platform is the ability to generate large mate pair libraries with long linker lengths. This will be further discussed in a later chapter.

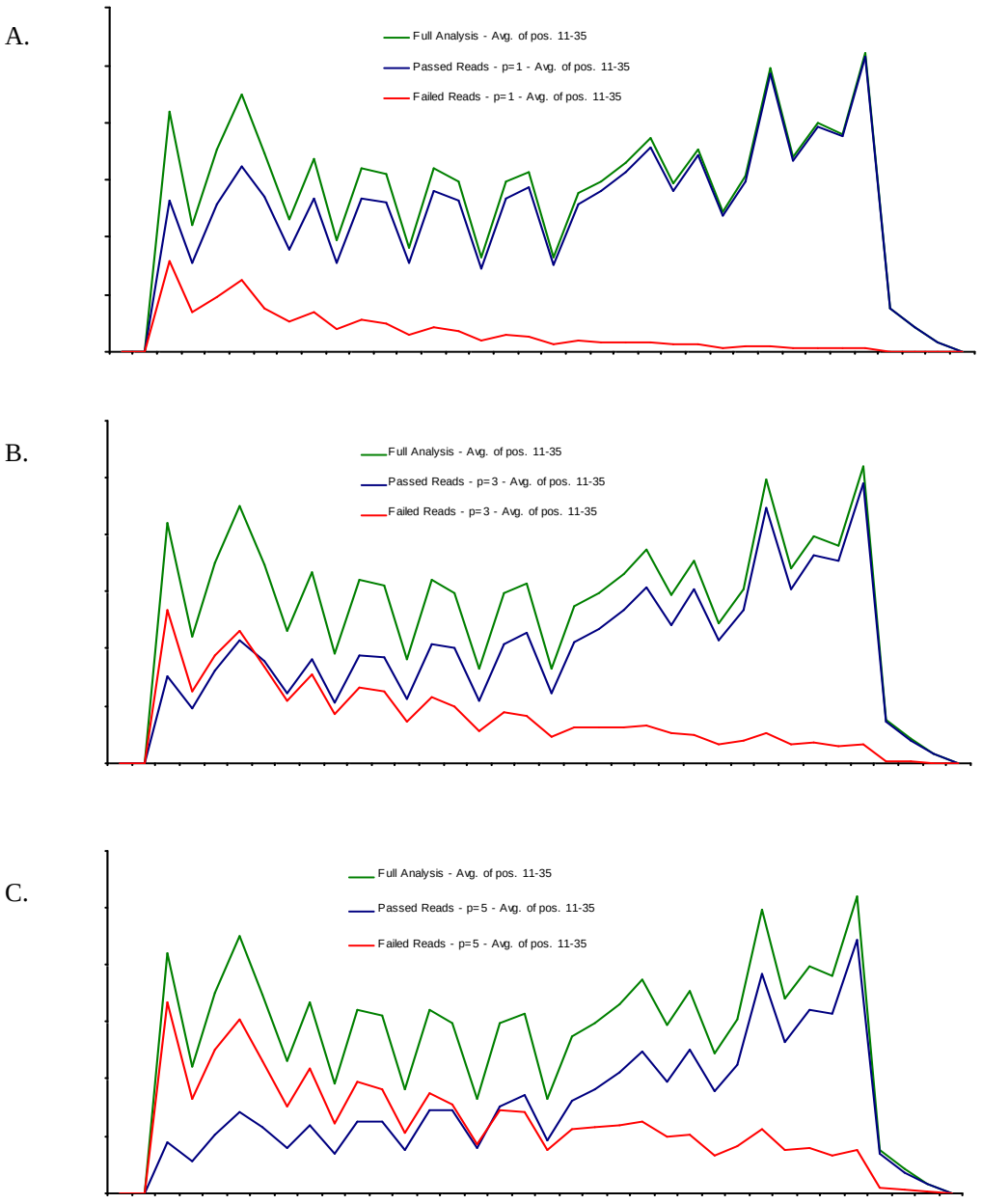


Figure 5.1: Predictive Nature of the First 10 Positions of the Sequenced Read - Graphs A, B, and C contain a quality value analysis on Arabidopsis Columbia accession pre-error and post-error analysis. The following error settings were used for all the above mentioned datasets: **A.** p = 1, p_QV = 25, e = n/a, & e_QV = 10 **B.** p=3, p_QV = 25, e=n/a, & e_QV = 10 and **C**. p=5, p_QV = 25, e=n/a & e_QV = 10. The lines represent the average QV for the positions 11-35. The green line represents for the full, unfiltered data; the blue line represents that sequences that passed the filter; and finally, the red line represents the reads that did not pass the filter requirements.

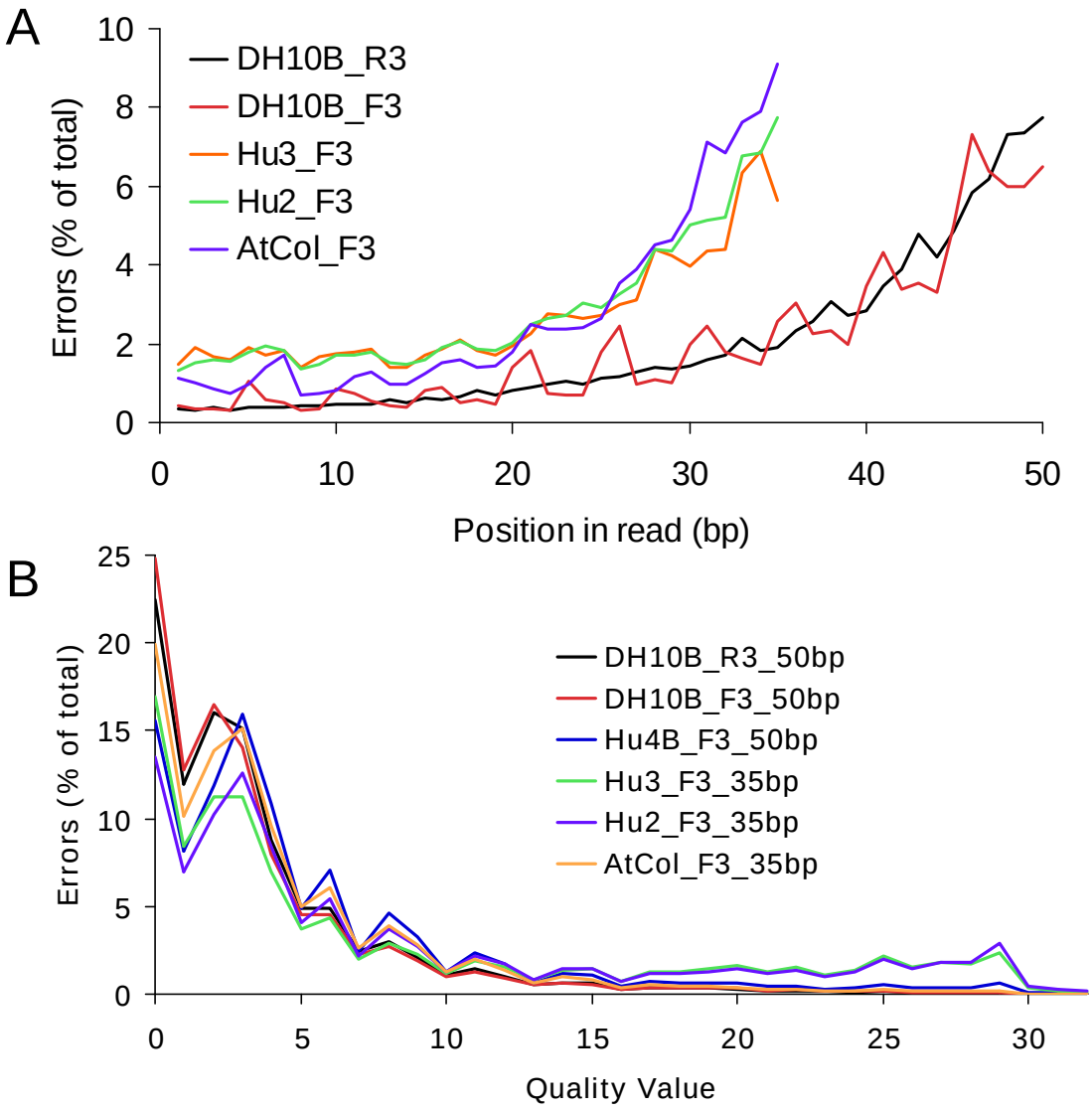


Figure 5.2: The Relationship Between Error and Location in SOLiD HTS reads - A. Location of errors in the SOLiD reads. The errors increase on the 3' end of the read, while the 5' end of the read remains relatively error free. **B**. The QVs of the identified errors from the SOLiD matching pipeline. QVs lower than 10 overwhelmingly correspond to detected errors based on the identification of error by the matching pipeline. DH10B_R3 (E. coli, reverse mate), DH10B_F3 (forward read), Hu3 (Human 3), Hu2, and AtCol_F3 (Arabidopsis thaliana, Columbia)

5.2 Data Analysis:

While filter defaults exist (minimum polyclonal counts: p=1, polyclonal minimum QV: p_QV=25, maximum errors identified: e=3, maximum QV to identify independent errors: e_QV=10), the settings of the parameters should depend upon the error tolerance of the downstream applications, such as mapping, *de novo* assembly, or transcriptome analysis. The user has the ability to define both the counts and the QV that determine the removal of a read from the dataset. More conservative parameters result in smaller datasets, as less data is able to successfully pass the filtering criteria is (Table 5.1, 5.2, 5.3, A.1, A.2, & A.3).

When the filter is applied to a *C. elegans* resequencing dataset, 25 x 25 bp mate pair short reads, using the stringent settings (p=3, p_QV=22, e=3, e_QV=10), the raw reads were reduced from 40 M reads to 5 M. Mapping of these reads increased from 56% to 96%. Of the reads that failed the filter, 38% still mapped with 0-2 mm. However, for both the unfiltered and the failed reads, many reads matched with 1 or 2 mm, while the filtered reads had the highest percentage of reads mapping with 0 mm. These results demonstrate that the filter can effectively identify perfect reads, which would be necessary for applications like *de novo* sequence assembly. While reducing the errors within the dataset is highly critical for a quality assembly, most assemblers contain error identification protocols and will attempt removal even if absent. In addition, extreme reduction of coverage could potentially be more harmful than the presence of few errors. In a practical *de novo* assembly project, we found that the settings should be much more relaxed [60].

This error analysis framework was tested on several datasets from SOLiD v2 and v3 (data not shown). The data that is shown is based on SOLiD v2 output of Arabidopsis fragment data and *C. elegans* mate pair sequencing runs. The Arabidopsis fragment run was a full slide which returned ~193 million reads, and the *C. elegans* mate pair runs were run on a quarter slide each and returned ~40 million sequences for each of the mate pair ends. Details on the composition of the sequences run can be found in Tables 5.1, 5.2, A.1, A.3. The sequencing for these runs followed the protocol as described (Chatterjee, Michael et al., *in review*). For analysis, the mapping for the resequencing portion was done using ABI's open source mapping software CoronaLite to match the reads to the genome. Error profiling on the matching output was done only on the reads which mapped uniquely to the reference genome.

A. thaliana	Read Length	Original # of reads F3	# of Passing Reads	% F3 reads Retained
			No filter	
Columbia Fragment	35 bp	193,121,694	193,121,694	100.0%
			Default (p1 & e3) error analysis	
Columbia Fragment	35 bp	193,121,694	64,294,608	33.3%
			p_1 & e_5 error analysis	
Columbia Fragment	35 bp	193,121,694	86,915,381	45.0%
			p_5 & e_0 error analysis	
Columbia Fragment	35 bp	193,121,694	19,566,547	10.1%

Table 5.1: Detailed Filtering Information-Fragment – Detailed information on the filtering of a fragment library of Arabidopsis Columbia accession pre-error and post-error analysis.

5.3 Downstream Applications - Alignment:

Here are downstream analyses done on a single mate pair dataset filtered at different levels. The library consists of 50 bp long reads generated by the SOLiD platform for the 4.7 Mb genome of *Escherichia coli DH10B* (available for download at http://solidsoftwaretools.com/gf/project/ecoli2x50/). The unaltered library provides ~600 x coverage of the genome.

Mapping was done using the open source software CoronaLite available from ABI (http://solidsoftwaretools.com/gf/). While mapping run times are not dramatic with this library, these times can vary with larger datasets (data not shown). Removing even the worst ~20% (p=1, e=off) of the reads does have an impact on mapping runtimes and the quality of the reads aligned. This very low filter is what we recommend for transcriptome analysis which is very sensitive to the read count. This filter removes the lowest quality reads, with minimal impact to mapping (Table 5.2, A.2).

					Sequence Alignment		
	Filter Criteria			% of			% Aligned
E. coli	Polyclonal	Error		Original	Mapping	%	to Total
DH10B	Count	Count	Reads	Reads	Run Time	Aligned	Reads Aligned
Full F3 Dataset	Off	Off	28,941,110	100.0%	1h 24 min	41.6%	100.0%
Full R3 Dataset	Off	Off	28,883,016	100.0%	1h 29 min	44.9%	100.0%
Filter 1 F3	1	Off	23,491,468	81.2%	1h 17 min	47.9%	93.4%
Filter 1 R3	1	Off	22,002,401	76.2%	1h 21 min	55.7%	94.5%
Filter 2 F3	3	Off	16,278,327	56.2%	1h	56.9%	76.8%
Filter 2 R3	3	Off	16,788,530	58.1%	1h 9 min	62.8%	81.3%
Filter 3 F3	5	Off	10,881,678	37.6%	46 min	63.6%	57.4%
Filter 3 R3	5	Off	12,660,928	43.8%	55 min	67.9%	66.3%
Filter 4 F3	1	5	6,160,991	21.3%	36 min	96.4%	49.3%
Filter 4 R3	1	5	7,612,647	26.4%	44 min	96.5%	56.7%
Filter 5 F3	1	3	4,148,469	14.3%	25 min	98.4%	33.9%
Filter 5 R3	1	3	5,355,243	18.5%	31 min	98.5%	40.7%
Filter 6 F3	3	3	3,925,017	13.6%	22 min	98.3%	32.0%
Filter 6 R3	3	3	5,175,181	17.9%	31 min	98.4%	39.3%
Filter 7 F3	5	0	778,838	2.7%	6 min	99.7%	6.4%
Filter 7 R3	5	0	1,179,762	4.1%	7 min	99.8%	9.1%

Table 5.2: Alignment Results for Different Filtering Criteria - Mapping results for different filtering criteria analyzed by ABI's CoronaLite. For different filtering criteria, we present the number of reads remaining after filtering, mapping runtime and the number of aligned reads. The maximum number of times the reads were allowed to match was 10 and the number of mismatches permitted was 3. As the filter setting becomes more conservative, the dataset gets smaller and the fidelity of the matching gets higher because the quality of the reads improves. While more reads that could potentially match get thrown out, there is an increasing probability that the remaining reads are true to the genome.

5.4 Downstream Applications: *De novo* Assembly:

An assembly was performed using Velvet on the DH10B genome (50 bp reads) (Table 5.3) [61]. Using simple fragment assembly, the impact of filtering can be seen in the trend of the contig N50s. Initially if the dataset is too large and the errors remain unfiltered, the assembly can be poisoned since there exists a larger probability for ambiguity and misassemblies. With too small a library, even if the quality is excellent,

the N50s fall since the reduction is coverage can yield an inability to extend the contigs. There is a tradeoff between coverage and quality in assembling a genome. The optimal choice of parameters, identification of errors and truncation depends on the coverage of the original dataset. (Table 5.3)

	Filter Criteria						
	Polyclonal	Error	Truncation		F3	F3	Total
DH10B	Count	Count	Length	Mate pairs	Orphans	Orphans	Sequences
Full	Off	off	35	28,627,096	314,014	255,920	57,824,126
Filter 1	1	off	35	19,586,554	3,904,914	2,415,847	45,493,869
Filter 2	3	off	35	12,782,983	3,495,344	4,005,547	33,066,857
Filter 3	5	off	35	8,177,848	2,703,830	4,483,080	23,542,606
Filter 4	1	5	35	2,727,735	3,433,256	4,884,912	13,773,638
Filter 5	1	3	35	1,439,840	2,708,629	3,915,403	9,503,712
Filter 6	3	3	35	1,382,001	2,543,016	3,793,180	9,100,198
Filter 7	5	0	35	89,862	688,976	1,089,900	1,958,600

	Fold	Contig
DH10B	Coverage	N50
Full	431	531
Filter 1	339	653
Filter 2	246	939
Filter 3	175	1,270
Filter 4	103	890
Filter 5	71	608
Filter 6	68	605
Filter 7	15	164

Table 5.3: Assembly at Different Filtering Criteria - For different filtering criteria, we show the number of remaining mate-pair reads, the number of remaining F3 and R3 orphans, the estimated fold coverage, and the N50s for contig assembly for DH10B. The trend of assembly can clearly be seen across the reported N50s. As there is refinement of the dataset for the better quality reads, the N50s increase. Once the coverage crosses a critical amount, the N50s decrease and the dataset is reduced to the point where quality restrictions became a detriment.

5.5 Methods:

The filtering Perl script (http://hts.rutgers.edu/filter) was written to overcome memory constraints imposed by uploading all of the reads and their corresponding QVs into memory. The program sacrificed runtime in order to have a small memory footprint. Uploading the data into memory for analysis will only prove more difficult as the quantities of reads sequenced grows with the new generations of the SOLiD platform. The algorithm holds only one read and its corresponding QVs in memory at a time. For mate pairs, one sequence and its corresponding QV is held from each file, F3 and R3, at any given time. Comparisons of bead identifiers allow for mate pairs to be identified, error checked, and then written to the appropriate output file. For a full slide of mate pair reads, 200 million+, the program requires several hours to run (*A. thaliana* fragment data ~200 million reads requires 5 hours to run; spe13D & E mate pair data ~40 million in each tag requires ~2 hours to run).

The filtering framework has several user defined fields which allow for full manipulation and customization of the output. It analyzes both mate pair and fragment SOLiD data. For each, both the .csfasta file and the QV.qual file are required in order for the analysis to proceed since this analysis is mostly based on the quality scores that are outputted by the SOLiD platform.

The output of this error analysis is defined by the user. The user sets up the initial name of the output file, the analysis fills in the rest. Another user defined option is to output the corresponding quality files along with the post analysis .csfasta files. These quality files can potentially be very large, so if unneeded for further analysis, this option can be turned off. For fragment error analysis output, only two sets of files exist: passing

and failing. Mate pair reads, in contrast, have eight different files, four for passing reads and four for failing reads (mate pair F3, mate pair R3, F3 orphan (mate is missing or of bad quality), and R3 orphan (mate is missing or of bad quality)). Both the passing and failing files contain two mate pair files where the mates are identified and output to their respective F3 and R3 file. The ordering of the mates is identical in both files. Additionally there are passing and failing orphan files. Reads that did not have a mate after SOLiD's primary analysis or ones that passed while their mates did not are separated and put into their respective F3 or R3 orphan file. Therefore, if a mate pair preprocessing is done, then eight files are outputted.

Truncation is an additional option and helps minimize errors in the short reads in order to maximize the usable sequence for post analysis. With the release of SOLiD v3, which allows for longer reads(up to 50 bp) truncation down to shorter sizes (30-35 bp) is viable for *de novo* assembly [62]. Since the quality of the reads degrades as the reads get longer on the 3' end, it is possible that reads that would have failed the filtering analysis at full length would pass once they were truncated (Figure 5.2) [62]. Recognition of truncation as a viable option is based on in-depth analyses of resequenced and matched data.

Finally, two additional functionalities were included: 1) removal of any read that contains a missed color call and 2) a quality score analysis for both the original SOLiD dataset and the passing short read sequences. The removal of missed color calls and the quality analysis are both optional and must be turned on if desired. The analysis returns a matrix containing the counts of color calls by position and score. This data can be plotted

in order to better understand the quality distributions of the files being analyzed and to see how the error identification and removal improved the quality of the output.

While some sort of preprocessing has become an essential step in the HTS analysis, choosing the proper filtering and truncation level can be difficult. Understanding the quality of the data and the downstream analysis is critical. While no filtering or low level filtering can be used for any analysis utilizing a reference genome, *de novo* assembly requires a more robust preprocessing step. A delicate balancing act ensues; parameters that are too lax will allow too many errors into the dataset, but parameters that are too stringent will extremely reduce the dataset, which can also prove harmful. Even if one can select the optimal level for preprocessing a particular library, the interaction between the library coverage and the assembly attempt can still have potentially significant effects resulting in contig termination.

Chapter 6: The Problem of Short Contigs

Even the issues of short read assembly did not exist, prefiltering was done to obtain only reads of good quality, and there remained good estimated coverage of the genome, large contigs might still prove elusive. While this problem is specific to fragment library *de novo* assembly, the presence of mate pairs does not compensate for extreme shortness of original contigs. Details will be further discussed in Chapter 8. In the end, these short assembled contigs require some additional information for improvement.

Essential to the assembly process is gaining insight into the potential reasons for the termination of contig extension. In the ideal world, Lander and Waterman's calculations suggested that the entire genomes can be recovered with sufficient coverage [47]. On the other hand, what is seen in practice during real assemblies is the breakage of contigs long before their estimated length is achieved. The termination of extension has two potential causes: poisoning of contig extension and coverage variability issues.

6.1 Poisoning Contig Assembly:

The poisoning of contig assembly is a variation on the idea of misassembly. Misassembly occurs if extension is done with the wrong read. There are two options for misassembly; it either adds a true read from a different part of the genome which can then be grown further to produce a chimeric contig or more likely, the misassembly occurs with an erroneous read killing the extension. While the first option occurs, it erroneously produces long chimeric contigs. In addition, poisoning of the contig assembly is dominated by the second option. Repetitive regions can fall prey to some form of this

poisoning. For example, if contig extension assembles the wrong part of the repeat, the extension is cut short. Errors in the short read datasets persist inspite of these efforts to mitigate them. As previously mentioned, extreme reduction of coverage could potentially be more harmful than the presence of some errors, but the error correcting modules in the assembly algorithms cannot always detect these erroneous reads which cause problems in the assembly. Identification and correction of these misassemblies is only recognizable in resequencing projects and remain camouflaged in *de novo* assemblies.

6.2 Coverage Variability:

High variance in coverage along the genome can cause problems with the assembler. Excessively high coverage is usually caused by repeated regions. Algorithms have difficulty interpreting these regions and usually try and find a simplistic way out of it, i.e. rejection and termination of contig extension. Sometimes, if it is rather easy for the assembler to identify the repetitive region, the assembler makes no attempts to calculate the number of repeats, it defaults to a preprogrammed number. Low coverage can also lead to early termination of contigs. Unlike with the high coverage regions where it is an algorithmic problem, here it is a library problem. The coverage of this region is low or does not exist therefore extension must terminate. This cannot really be corrected by clever improvement of algorithms because the dataset is at fault. High variability of coverage from high throughput sequencing runs seems to be the norm, understanding the reasons for contig termination can hopefully lead to better algorithms targeting assembly.

6.3 Evaluation with Real Data:

Evaluation of contig failure is most easily done using resequencing data. In order to further understand the reasons for breakage, we examined a contig assembly of *E. coli* 50x50 bp SOLiD mate pair dataset using both Velvet [32] and SOPRA modified contigs (Figure 6.1). [45]. This analysis did not use mate pair scaffolding. These studies indicate that most of the contig ends are correlated to local drops in coverage. Evidence of this can be seen in Figures 6.1 and 6.2. The graphs suggest that there is a skew to these breakages. Some of them might be a result of high variation in coverage that the assembler cannot reconcile, but a parameter shift or a different assembler that uses a different algorithm might be able to resolve them. However, regardless of the size of this group, there are regions with ultra low to no coverage that no assembler would be able to resolve.

This analysis only considered perfectly mappable contigs. A small percentage of the reads that did not comply and which could not be mapped might simply contain some error. Others represent false assemblies which would not match the genome. Table 6.1 contains the alignment statistics for this assembly. SOPRA will sometimes trim the contigs when it fits certain criteria. This cutting seems to improve the alignment.

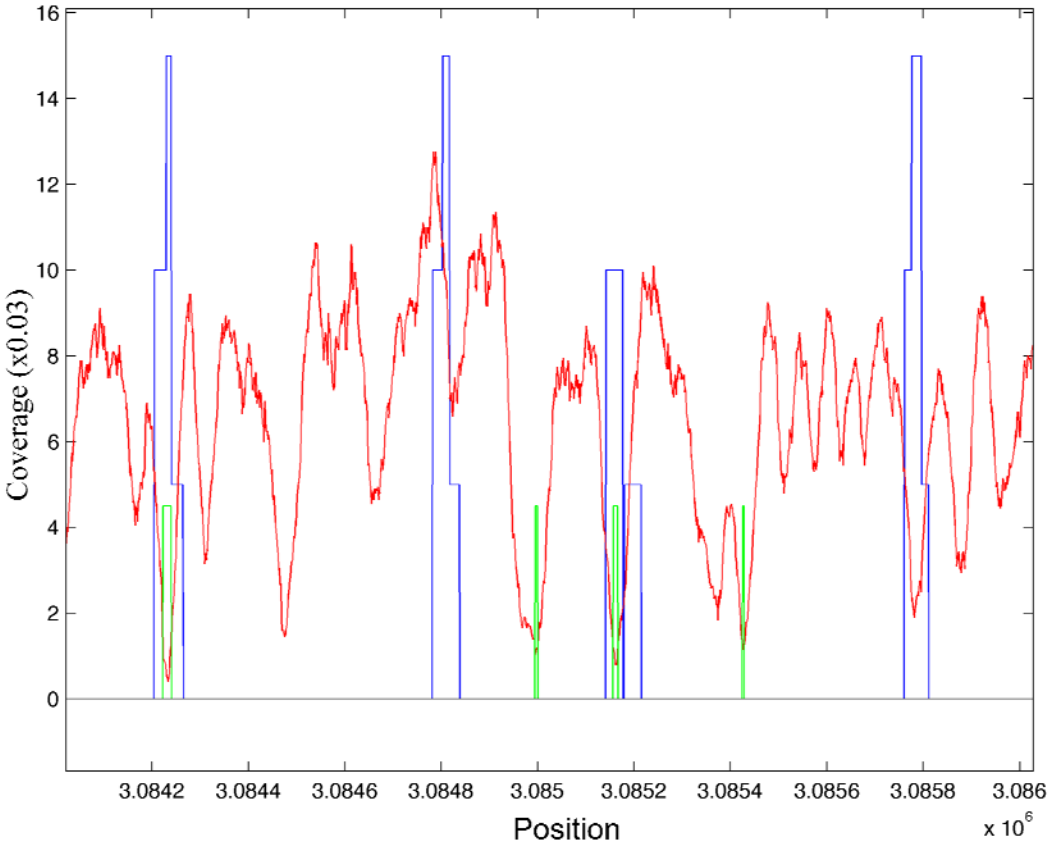


Figure 6.1: Contig Alignment and Coverage for E. coli Assembly – The red line represents the coverage (multiplied by 0.03 for scaling) which was determined by aligning trimmed and filtered reads along the genome using BOWTIE .[63]. The blue lines represent the contig edges. The right edge is represented by the histogram at value 10; the left edge is represented by the histogram at value 5. If there exists an overlap, the blue line in the histogram become the sum of the two (15) (example positions ~3.0842 x 10^6), if not, two independent boxes remain with a gap in the middle (example positions ~3.0852 x 10^6). Finally the green lines represent coverage dips below a certain threshold. While this figure shows multiple causes for contig termination, very often the breaks correlate with low coverage regions (Figure 6.2).

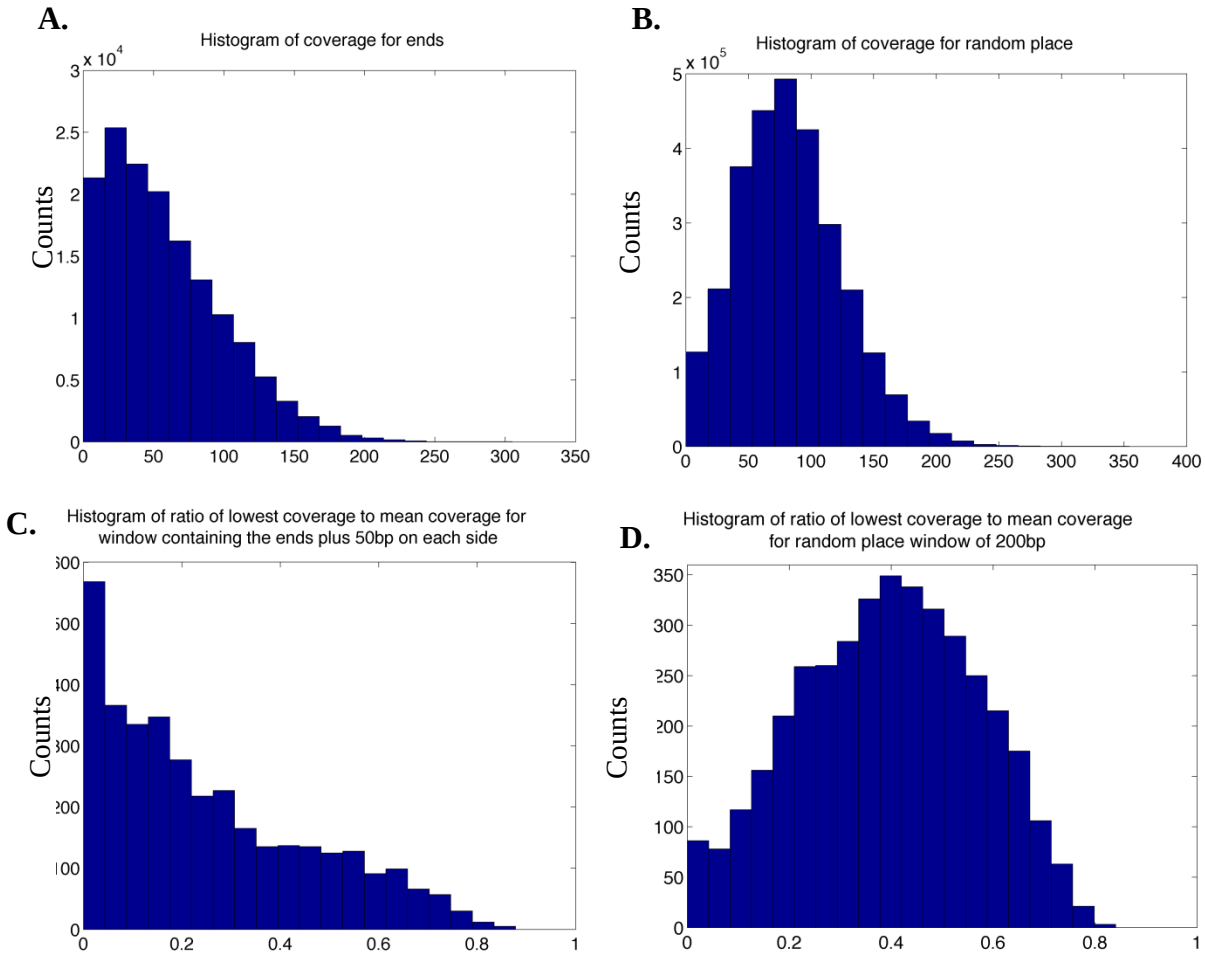


Figure 6.2: Contig breakage – Panels A and C show the statistics from the alignment of contig ends. If the same analysis were run with mapping at random point the graphs would be more similar to Panels B and D. This seems to indicate a strong bias towards breakage at low coverage.

	Alignment					
	Unique		Multiple		Failed	
	# reads	%	# reads	%	# reads	%
SOPRA contig left end	4,428	94.6%	175	3.7%	76	1.6%
SOPRA contig right end	4,334	92.6%	220	4.7%	125	2.7%
Velvet contig left end	4,311	87.7%	209	4.3%	396	8.1%
Velvet contig right end	3,571	72.6%	266	5.4%	1,079	22.0%

Table 6.1: Contig Aligment Statistics for Breakage Analysis – SOPRA will sometimes trim the contigs under to improve assembly. This seems to lead to better alignments against the genome.

A similar analysis was performed on the assembled bacteriophages where an orthologous sequence existed, a known close relative. Specifics of the experiment are discussed in the following chapter where the assembly of the bacteriophages is discussed in detail. The statistics are not as clear cut since full alignment to the genome is not expected. In addition, Blast [64] alignments do not match up exactly to CoronaLite's alignment since Blast allows for indels and only identifies strongly homologous regions. Figure 6.3 represents coverage versus the location of the contigs for a very narrow region. In Figure B.3, the alignment coverage and positioning of the contigs is shown along the entire genome. These two contigs were aligned with the highest Blast score and homology to the reference. From Figure 6.3 it is apparent that the breakage is the result of a dramatic dip in coverage for that region. When looking closely at the contig ends in Figure 6.4, approximately fifty percent terminate under the conditions of low or zero coverage in the window (Table 6.2). For the rest of the locations, there exists high coverage. In some of the cases, there were misassemblies; using the alignments these areas were able to be remedied and in some cases the gaps were able to be bridged. The alignment of contigs only reports the pieces of the contigs that are similar. If a contig end is not homologous, then the contig alignment will not report it. With an orthologous genome being used as a reference, such regions are to be expected. In particular, this is where NCBI Blast aligned these reads according to close homology and in some cases cut off parts of the contig that did not match [64]. In one particular case, one contig was split into three in accordance to what the reference genome suggested. Upon further inspection, it seems to be a rearrangement, and the six contig ends should really be treated as two.

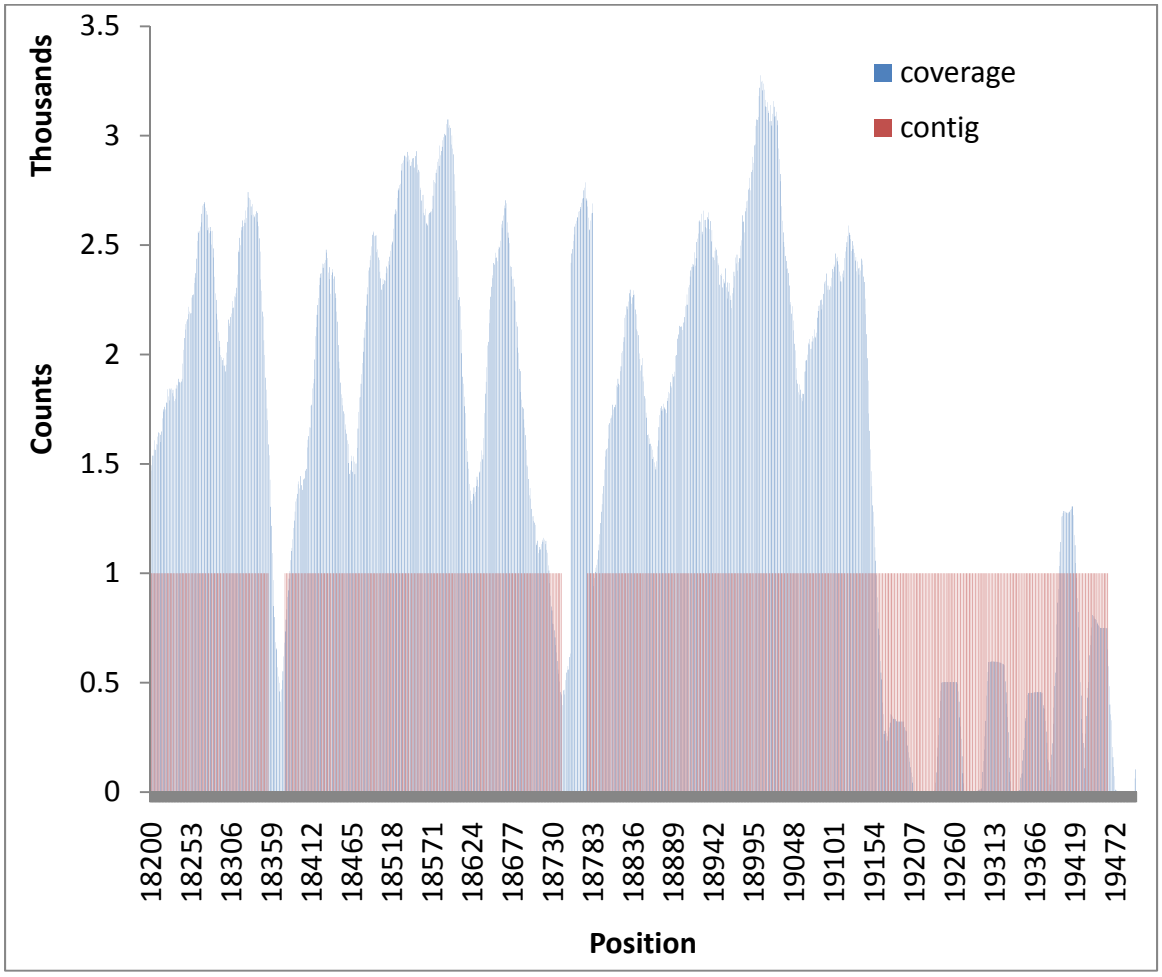


Figure 6.3: Contig Breakage in Bacteriophage OP1H – The blue represents the raw coverage at that particular base in the genome. The red boxes represent where the contigs aligned. The two contigs have very high Blast [64] scores suggesting high homology. The breaks align neatly with sharp dips in coverage.

		Window (30 bp) around position		
		Avg < 150		
	Total	Min < 30	Min = 0	Avg > 500
Contig End	44	24	19	16

Table 6.2: Contig End Mapping with 30 bp Window – Details for levels of contig end mapping for bacteriophage OP1H

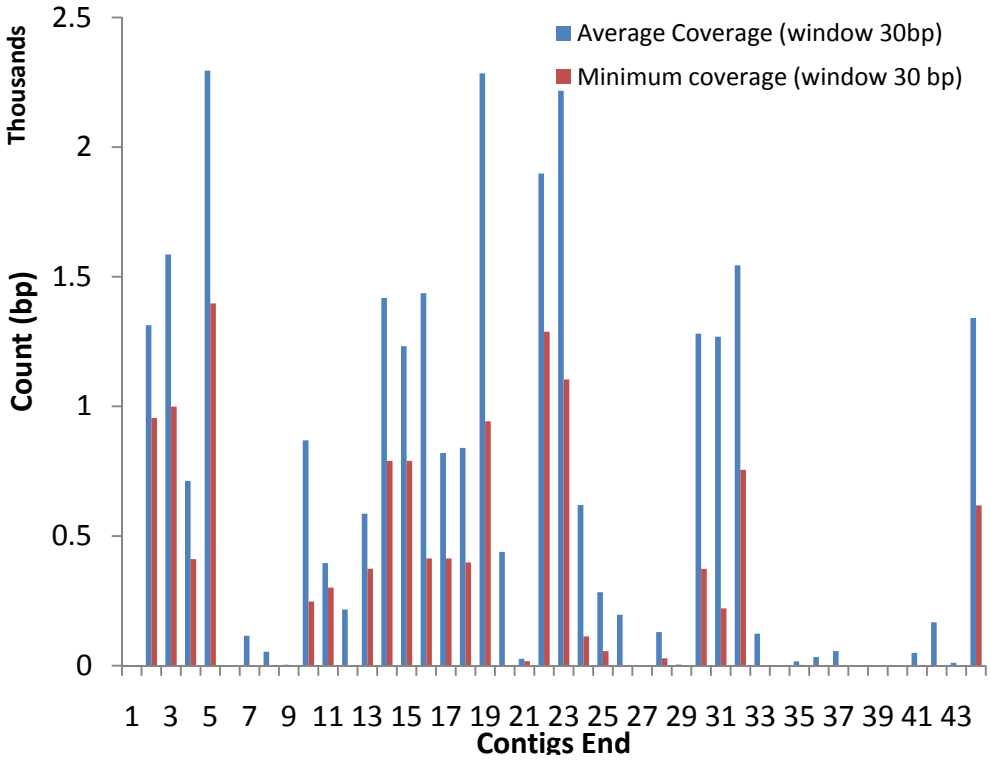


Figure 6.4: Contig End Coverage Map OP1H – The blue represents the average coverage for the 30 bp window and the red represents the minimum coverage within the window.

After a detailed look at to the factors contributing to contig breakage, it has become apparent that the majority of these breaks are a result of high variation in the coverage. It appears that the non-uniform distribution of coverage is an intrinsic problem for most of these HTS libraries. Even if some of these contig ends could be recovered through a variation in parameter of the assembler, some of these regions will cause breakage regardless of the algorithm. As a result other methods, such as comparative assembly and use of mate pair libraries could potentially be used to ameliorate the assembly.

Acknowledgment: The E. coli dataset is the publicly 50x50 bp mate-pair available from ABI SOLiD. The analysis of this dataset was done by Adel Dayarian. The bacteriophage datasets belong to Konstantin Severinov. I assembled and analyzed the bacteriophage datasets.

Chapter 7: Comparative Assembly

Even under the most ideal conditions: high coverage, low repetitive regions, and short genome, one can expect fractured contig assemblies from a fragment assembly with short reads [49]. Even with an optimum library, contig breakage still occurs before the optimal length Figure 3.1. The contigs tend to break partially as a result of misassemblies, but most breaks are caused by high variation in the actual coverage of the genome. Even with these broken assemblies, if there is a reference or related genome, the extension can be recovered to some extent [23, 24]. While the dips in coverage will cause assembly protocols to break due to lack of confidence, comparative assembly methods are able to recover this to some extent. Essentially, the alignment step is able to give the confidence needed to extend the contig through a localized assembly. As can be seen below, while the entire genome might not be recovered, there is a significant improvement using a comparative assembly method (Table 7.3).

Described below is a project whose goal was to assemble seven bacteriophage sequences *de novo* from sequencing runs on the SOLiD HTS platform. The estimated length of these genomes was between 40 – 85 kbp long, they lacked significant repetitive elements, there was sufficient DNA to skip the PCR amplification step in the library preparation; and there was exceptionally high coverage even post filtering. Since these genomes were small and contained very few (if any) repetitive regions, fragment sequencing was initially believed to be sufficient for a full genome assembly. The sequences produced were 50 bp reads with all seven bacteriophages barcoded in one quarter of the slide.

7.1 Analysis of the Library and Preprocessing:

The first step was to assess the quality of the dataset and sub sample the data to understand what might be expected from filtering. Twenty percent of the data, evenly distributed through the dataset, was sampled and analyzed. Figure 7.1 contains the quality profile for phage G20C. This profile was similar for all the bacteriophage libraries sequenced, and is highly representative of the true profile for the entire dataset. The negative one quality value stands for a miscall, the color could not be discerned during processing. All miscalls were removed during filtering since the miscall would add ambiguity to the translation process.

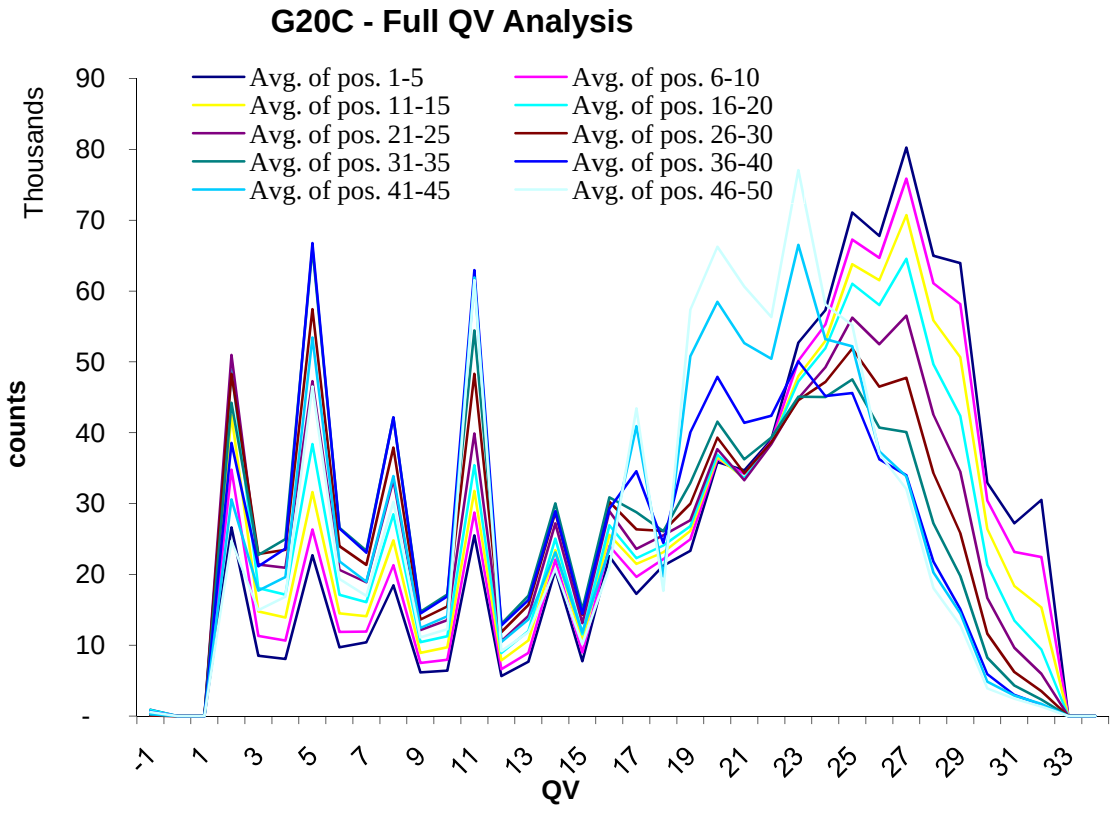


Figure 7.1: Full G20C Bacteriophage Quality Profile – This QV profile is based on a random sampling of twenty percent of the data. The analysis keeps track of the positions of every QV to assess the quality of the run. This is highly representative of the entire dataset.

While many filtering and truncation parameters were run, the final filter used was a mean filter where the average QV had to be twenty or higher (Figure 7.2). The severe truncation was only possible because of the exceedingly high coverage. This allowed the removal of error tails and high confidence in the remaining reads. Table 7.1 contains information on the filtering and coverage of the genome. Notice bacteriophages G17 and XP12 are not in line with the other sequenced bacteriophages. Usually with barcoding, libraries are of approximately equal sizes. The unusual overabundance of G17 and under-abundance of XP12, could mean that there were problems with those particular libraries. While assembly was continued until the end, their final assemblies remain in question; further analysis is required on those two genomes.

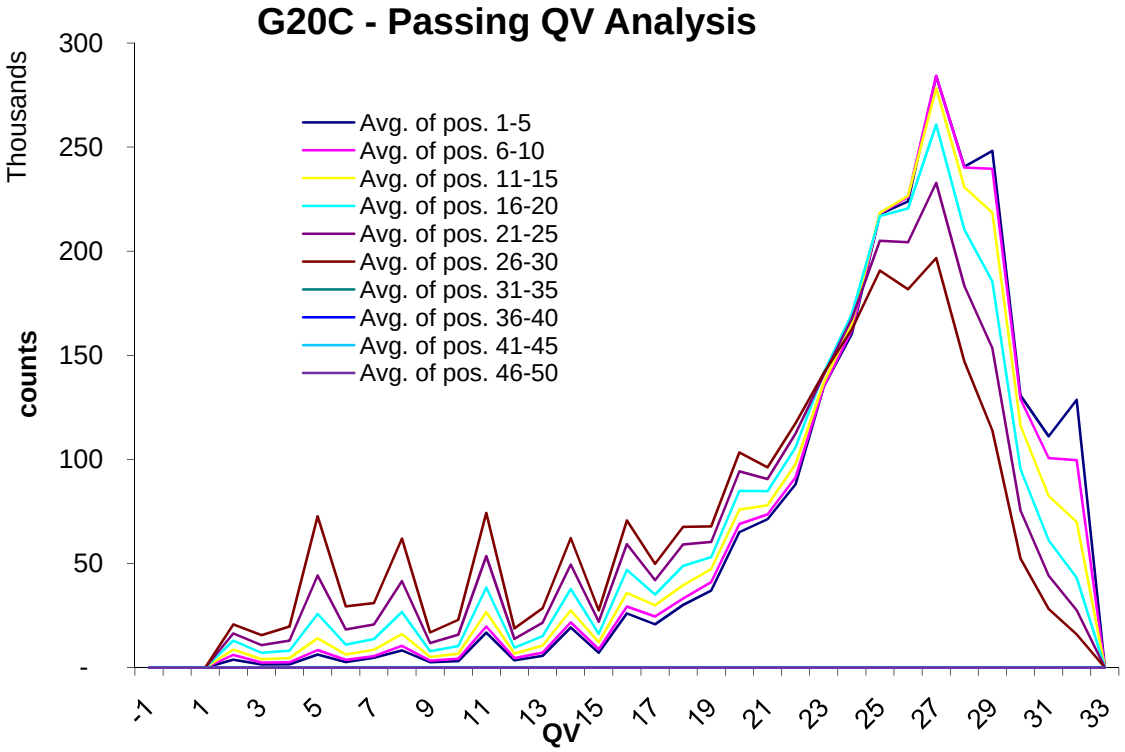


Figure 7.2: Passing G20C Bacteriophage Quality Profile – This QV profile for all the reads that passed the filter. This shows a significant improvement in the quality of the passing reads over the original dataset. The periodic bumps on the tail correlate with the SOLiD sequencing cycles.

column	(1)	(2)	=(2)/(1)	=(2)*30 /45000	=(2)*30 /80000
	Total reads	Filtered &		45 kb	80 kb
	Original Solid	Truncated	% of Total	Est. Fold	Est. Fold
Phage	Output (50 bp)	Remaining Reads	Reads	Coverage	Coverage
G17T	47,515,184	18,369,653	38.7%	12,246	6,889
G18	8,608,948	3,726,996	43.3%	2,485	1,398
G20C	4,638,580	2,305,365	49.7%	1,537	865
G20T	4,170,365	1,931,185	46.3%	1,287	724
OP1H	7,832,974	3,515,413	44.9%	2,174	1,223
OP1H2	6,486,125	3,260,577	50.3%	2,344	1,318
XP12	1,173,552	463,209	39.5%	309	174

Table 7.1: Filtering for Bacteriophage Assembly – This table lists the number of reads which pass the final filter prior to the assembly attempt. The filter parameter used was the mean QV of the read must be greater or equal to twenty and the reads were truncated to thirty bp long.

7.2 *De novo* Contig Assembly:

As was stated above, assembly is very sensitive to the parameters assigned. So for every assembly, multiple parameters were used to fine-tune the assembly process [59]. The contig assembler used was Velvet [32] with the translation to base space done by the SOPRA translation module [45]. The critical parameters used in this assembly were kmer length and the coverage cutoffs under which the connection between two kmers is considered erroneous. In addition, for high amounts of coverage, if a significant number of errors remain in the dataset, assemblers can become confused and truncate the contigs from their maximal assemblies. As a result, a bootstrapping test was done which targeted approximately 100 fold coverage of the genome with 10 replicates each for attempted assembly (truncation 30, mean filter m=20). The contigs produced were slightly smaller than with use of the full dataset, so in the end the full dataset was used

for contig assembly. Since the truncation removes most of the error tail, there is high confidence in the remaining 30 bp. In this case, the high coverage did not hurt the assemblies.

As Figure 3.1 suggested, the assembly should have rapidly been assembled with the coverage allotted. Running many assemblies under various combinations of filters and parameters allowed for experience with the parameter-space and was useful in narrowing the parameters toward those used in the final assembly. As seen in Table 7.2, the N50s are significantly shorter than the full length of the genome, and without mate pairs there is no other additional information from the HTS library that can be applied to improve the assembly.

Phage	Kmer Length	Coverage Cutoff	N50 (bp)	Maximum Contig (bp)	Total Number of Bases
G17T	23	100	353	2,753	211,365
G18	21	75	1,413	6,973	62,668
G20C	19	50	1,307	4,180	74,587
G20T	19	50	947	2,898	62,923
OP1H	23	75	3,217	6,457	41,640
OP1H2	19	75	2,202	5,171	40,149
XP12	21	50	305	1,832	19,250

Table 7.2: Bacteriophage Assembly Statistics – This table lists the statistics from the *de novo* assembly run.

7.3 Improving the Assembly: Comparative Assembly:

Using a highly homologous genome, a close relative, for a comparative assembly is not a novel idea [23, 24]. Using the reference aids in reducing the complexity of the assembly. While there have been attempts to do a comparative assembly with sequence [65], no real effort has been done with solely short reads. Salzberg et al did use a mixed *de novo* approach to assemble Illumina short reads, but it relied heavily on the comparative information for assembly [66]. From the beginning, their pipeline requires a lot of initial information, i.e. multiple reference sequences from close relatives, and many assumptions to build the gene and protein models for the use of their gene-boosting technique. While more data can only aid in the process of assembly, sometimes such large amounts of information and inferences are not possible. If there is only some minimum amount of data, some version of Gnerre et al. tool to use with HTS assembly pipelines would be quite useful. Described below is an algorithm to do just that (Figure 7.3).

The method was developed with a focus on improving the assemblies of the bacteriophages. For three of these bacteriophages, a close relative's genome was available for comparison, OP1 and XP10 for the assembled phage genome OP1H and OP1H2, and P23_45 for G20C. G17T, G18, G20T, while having no close relative's sequence, are closely related to each other, and XP12 has no known close relatives. Since these three known genomes were available, not only can confidence be drawn on the contigs assembled in the *de novo* assembly, the HTS library could be aligned using this known sequence as a reference, and finally the contigs could be extended using an overlap and extension scheme for homologous regions (Figure 7.3).

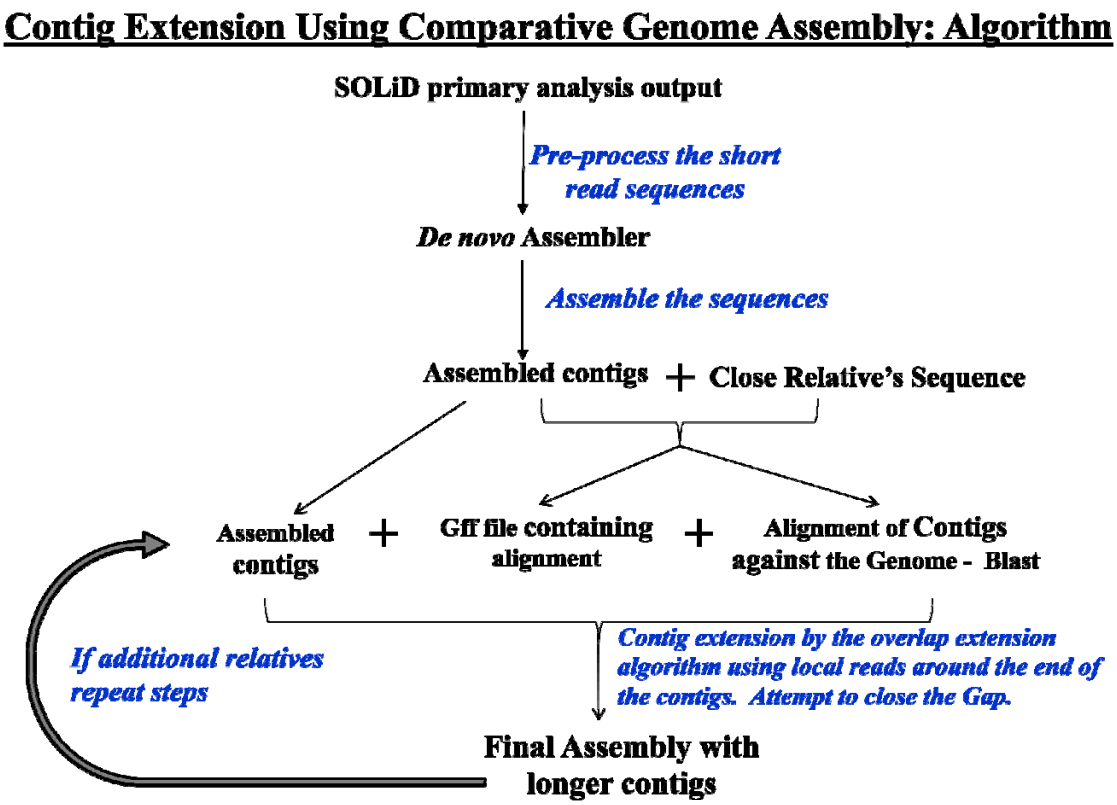


Figure 7.3: Comparative Assembly Pipeline – This figure represents a schematic of the algorithm used for comparative assembly discussed in this chapter.

The first step was to Blast the assembled contigs against the sequences of the known genome using NCBI nucleotide Blast [64] to get a sense of the quality of the assembly, the start and stop positions, and directionality of the contigs relative to the reference genome. Concurrently, an alignment of the reads used for assembly was done using ABI's CoronaLite (http://solidsoftwaretools.com/gf/) (Table B.1). While not as fast as some of the aligners available, CoronaLite does have the advantage of identifying and characterizing single color call errors. It is trivial to use this information to correct these single color call errors. The gff3 file was created from this alignment. The sequences corrected and translated into DNA space and then downloaded for contig extension. Initially all three genomes were extended by hand using a SSAKE like

method, overlap and extension [29]. Using this method, significant improvements were made as can be seen in Table 7.3 and in the visualizations of the improvements in Figures 7.4, B.2, and B.3. Since there were two genomes for use with OP1H and OP1H2, this process was repeated twice. The first cycle used alignments against the OP1 genome to obtain assembly v1, and then again with XP10 to obtain the final assembly.

	De novo Assembly		Comparative Assembly		%
		Maximum		Maximum	Improvement
Phage	N50 (bp)	Contig (bp)	N50 (bp)	Contig (bp)	N50
G20C	1,307	4,180	8,266	13,118	532.4%
OP1H	3,217	6,457	9,192	11,900	185.7%
OP1H2	2,202	5,171	4,411	7,468	100.3%

Table 7.3: Comparative Assembly Improvements – For all three bacteriophages significant improvements were made to the N50s and maximal contigs but using the comparative assembly algorithm described above.

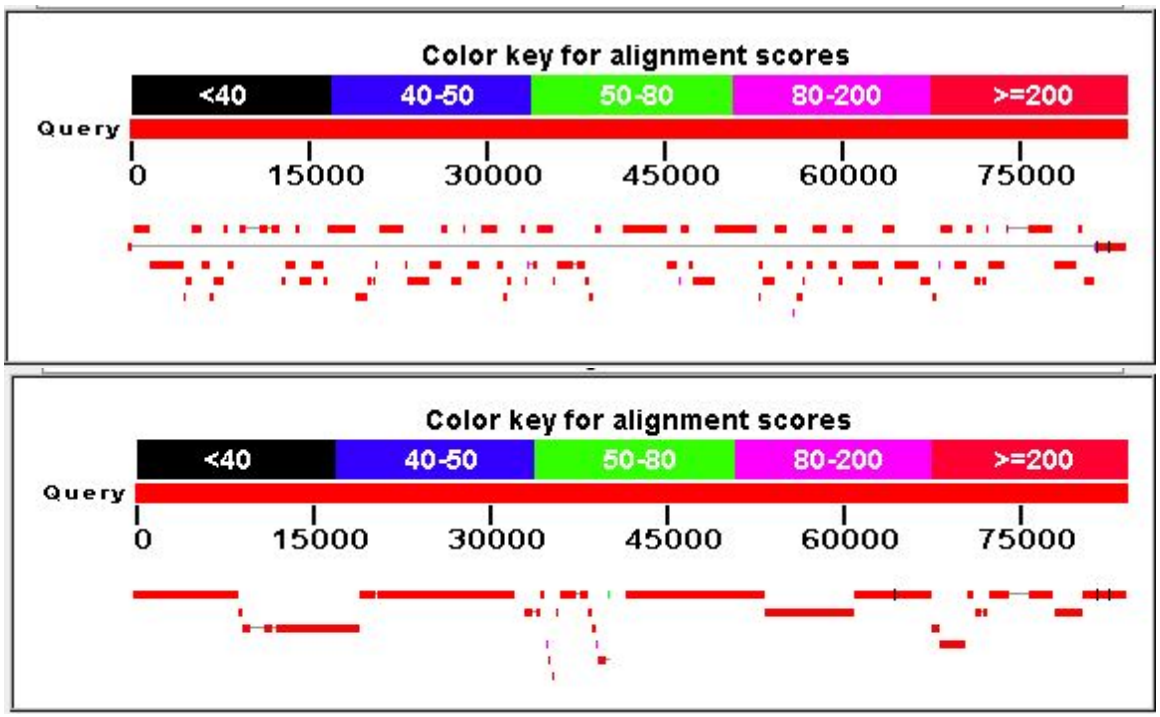


Figure 7.4: Visualizations of the Comparative Assembly of G20C – Here is a visualization of the improvements made to the assembly. The top panel contains the contigs aligned to the reference P23_45 post *de novo* assembly. The bottom panel contains the final assembly contigs aligned to the genome. As can be seen there is a significant improvement in the length and ordering of the reads. This can be seen for all the comparative assemblies done (Figures B.5 and B.6).

7.4 *De novo* Pipeline Addition: Comparative Assembly:

With the success discovered during the extension and the improvement of the bacteriophage assembly, automating the process as described in Figure 7.3 seemed a reasonable undertaking. As a result CoAX (Comparative Assembly eXtender) was developed to take in the modified gff file, the Blast alignment [64], and a fasta file containing the assembled contigs and extend the contigs by less stringent local assemblies. The Perl program uses the start and stop sites of the alignments to produce micro assembly cycles using a SSAKE-like program [29]. This mitigates the propensity for misassemblies by giving the micro-assembly cycle only reads that have aligned in that

local region. In addition, since alignment gives locality, looser overlap and extension parameters can be used. If extension can bridge the interval between the two contigs, then the contigs are merged. If not, the remaining reads are used to attempt a loose assembly of the gap. If these assemblies are greater than one and a half times the reads length, then they are kept otherwise they are thrown out. If some of the assembled contigs align to non-homologous regions, the contigs are not cut based on the Blast alignment prior to extension [64]. That particular end of the contig remains whole, this will maintain the information to be further corrected during the annotation step or with additional sequencing. The final contigs are given in the order that they are aligned to the genome. The script, while not as robust as a hand assembly, i.e. if overlaps are shorter than then some given threshold, they are kept separate in order to remain confident in the automation. The final assemblies produced are very close to the hand curated assemblies.

While comparative assembly using a closely related genome did an impressive job improving the assembly, it is not always possible to do. The comparative assembly pipeline may even be able to improve the assembly even more if several close relatives are available. If there are no close relatives, there are still potentially ways to improve the comparative assembly possibilities. There exists the possibility to map the contigs based on protein homology and then use a protein-boosted approach for improvement. Since the phages are gene dense, the partial overlap of contigs with these proteins might allow for the creation of a type of scaffold reflecting a likelihood of proximity. Another option is to use overlap extension across many species, some more closely related than others with some sort of weighting mechanism based on evolutionary distance for confidence. The simplest possibility is the use of paired read libraries from the

beginning. This information can aid in relative placement based solely on the genome in question. There are many considerations before deciding to sequence a paired read library, cost, starting material etc. The question then remains, how good is the improvement if paired reads are used?

Chapter 8: When Mate Pairs Can Rescue the Assembly

As can be seen from the bacteriophage assembly described above, super high coverage should be able to rescue the assembly, but it does not. The other mechanism for rescuing and improving the assembly is by using paired reads [34, 42, 45, 59]. Some sort of paired read is what made assemblies of large genomes possible [8, 10, 28]. Specifically, large HTS projects are only just getting underway. In the Panda genome assembly project, paired end reads were incorporated almost from the beginning. Li et al used 37 paired end read libraries targeting 73 fold coverage of the panda genome [28]. The final high N50s reflect their integral use of the paired reads from the beginning. But in their initial round of assembly, they focused on creating confident contigs. Using just the short reads from the 500 bp linker paired end library which approximated thirty nine fold coverage of the genome, their initial contig N50s were 1.5 kbp. This is in line with what was produced for the bacteriophages discussed above. Assembly then progressed using cycles of assembly with additional libraries varying lengths of linkers. In addition, comparison and confidence was gain from comparisons with the annotated dog genome. The final assembly with an N50 of approximately 40,000 kbp and approximately 200,000 gaps was achieved [28].

All the current next-generation sequencing platforms contain protocols to generate some type of paired read, mate pairs or paired end. Paired reads have an additional piece of information, the approximate distance between the reads. The difference between the two types of paired reads is in the library preparation. Essentially, mate pairs require circularization of the DNA (linkers 300-10,000 bp). For paired ends, the reads come from sequencing two ends of a long fragment and as a limitation of this

protocol have much narrower range of short linkers (300-500 bp) (Figure 4.1). There are complications with targeting the extremely long linker lengths in mate pairs, such as significantly more starting material required. Illumina and SOLiD have recently introduced the other paired read kit into its sequencing protocols (Illumina has added mate pairs and SOLiD has added paired end). Paired end is a significant addition to the SOLiD arsenal because the costs for sequencing and the starting material remain in line with a fragment assembly, but make available the linker information for analysis. For mate pairs, the long linker length is SOLiD's biggest asset for assembly. They allow for facilitated recovery from long repetitive regions. Without them, some assemblies can become stuck with very few options for recovery. The Illumina mate pair protocols are currently less robust though there are attempts at improving it in various sequencing centers.

While very useful, these linker length do contain variability that is relatively high [45]. This variability is a function of size selection. Running the long lengths on the gel allows for a narrower window between the bands, so one essentially targets a band of linkers or a distribution around a mean rather than a particular linker size. Balancing between the advantages and parameters available for narrowing down the linker length variance would require new sequencing experiments to specifically tackle this problem.

8.1 Ideal Case:

In order to better understand the interplay between parameters, such as coverage, read length, and how mate pair can recover the assembly, a simulation was designed and implemented using Matlab R2009b to mimic *de novo* assembly of a genome. The

simulation allows for various combinations of genome length, read length, overlap criteria for contig extension, various combination and sizes of mate pair libraries, variability in linker length to understand scaffold disentanglement, misassembly rates and breakage rates. The *de novo* assembly algorithm can be visualized in Figure 8.1.

While other aspects were added to try and understand the interactions between contig assembly and scaffolding, these simulations were able to establish a baseline for scaffolding, i.e. the optimal scaffold assembly. The simulation was able to recreate the baseline calculated by Lander and Waterman during contig extension, and then push it onto scaffold building. Under ideal conditions and starting at very low coverage, at least one scaffold exists that can successfully span the genome (Figure 8.2A). Therefore, even if contig assembly fails and has short N50s the scaffold spanning the genome can be assembled even if there are large gaps. Even with the high variability of linkers, the ideal scaffolding was able to span the genome.

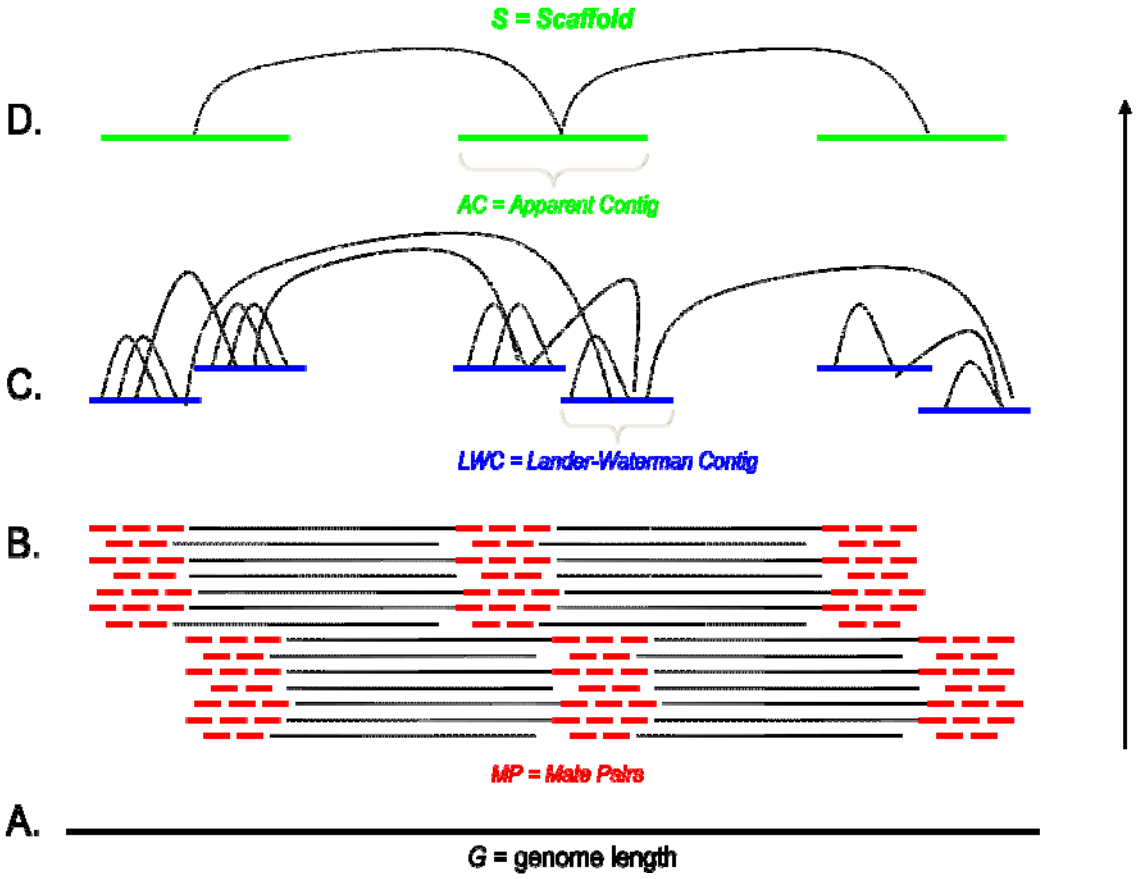


Figure 8.1: Representation and terminology of the assembly algorithm - **A.** The line represents the length of the genome. Due to computational concerns, the simulations were run on a genome with a single chromosome with a length of 10 Mbp or less. **B.** The red lines represent the short read ends of the mate pair read. The black lines represent the linker connection between the short read ends. **C.** The blue lines represent how the short read sequences are aggregated into standard contigs using a large overlap threshold for confidence. To go from B to C the linkers are tracked but are not used to assemble the Lander-Waterman contigs. **D.** The linker information from the mate pair reads is used to separate the independent connected components from each other. Then with these connected components if two Lander-Waterman contigs meet the overlap criteria the Lander-Waterman contigs are grown into an apparent contig. If not, they independently become apparent contigs joined by a linker. The minimization of the apparent contigs yields the final scaffold outcome.

Figure 8.2 shows how the coverage and read length can affect the contig assembly and scaffolding. Saturation of the genome by the scaffold is achieved very quickly, with full coverage of the genome not far behind. Therefore, Figure 8.2 B focuses on the fold coverages prior to the saturation of the 10Mbp genome. Independent contig assembly only approaches that limit when the reads are long and overlaps are low. In practice, this is unrealistic due to the ambiguity of short overlaps as discussed above. But, Figure 8.2A demonstrates how critical the ratio of overlap to read length is critical for contig extension. Figure 8.2B shows the relationship between the contig assembly and scaffolding results. Each of the lines represents different fold coverages of the genome. The enhancement factor calculated for each of the various linker lengths (x axis) is the quotient of the coverage of the longest scaffold (final output of the mate-pair assembly) by the N50s of the Lander-Waterman contigs (output from the first part of assembly simulation which mimics fragment assembly). In this figure, one see only a portion of the data since coverage of the genome rapidly approaches saturation, full coverage with no gaps. Additional runs are currently continuing to further explore scaffold building.

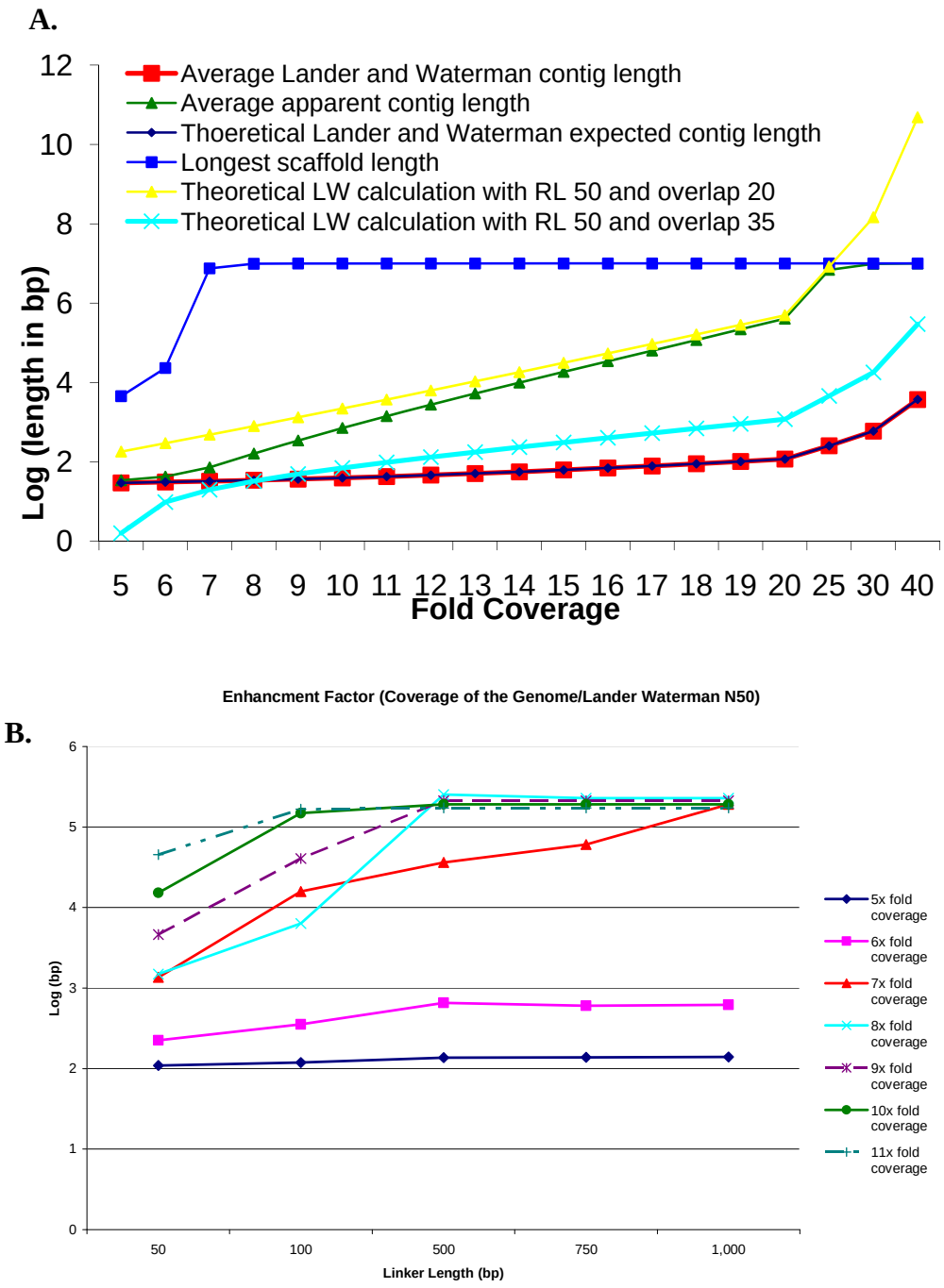


Figure 8.2: Simulations Mimicking *De novo* Assembly Output – A. Fold Coverage effects – From the simulations, a comparison was able to be made between fold coverage, contig growth, and scaffolding. **B. The Enhancement Factor -** The simulations output a list of traditional Lander-Waterman contigs and the scaffold coverage of the genome. The Enhancement Factor was calculated by taking the product of the quotient of the coverage of the longest scaffold (in bp) by the N50 of the Lander-Waterman contigs. The length of the simulated genome was 10 Mbp. Saturation was achieved quickly, so only a subset of the fold coverages was used.

8.2 Scaffolding Algorithms:

Scaffolding algorithms have been essential to all major sequencing efforts to date [8, 10, 28, 42, 44]. The early scaffolders that were discussed by Huson et al. and Pop et al. work by using greedy algorithms. These greedy algorithms slowly add paired reads until it reaches the final scaffold. The goal of these algorithms is to add the reads so that a consistent scaffold is maintained at all times. The problem occurs if one erroneous paired read exists and is added to the scaffold since it has no conflicts. When the true paired read is selected for addition, since it poses an inconsistency, the new one will be thrown out incorrectly. Essentially if there is no conflict with the previous scaffold it is added; from there, when conflicts arise, the already assembled scaffold takes precedence (Figure 8.3). If there are no inconsistencies, the greedy scaffolding algorithm works very well and the entire scaffold is appropriately returned. With small paired read libraries, this can be an effective algorithm because the mate pairs are in a vastly smaller proportion to the reads. With HTS platform, this method is not as effective because of the production of large mate pair libraries causing the high potential for conflicts.

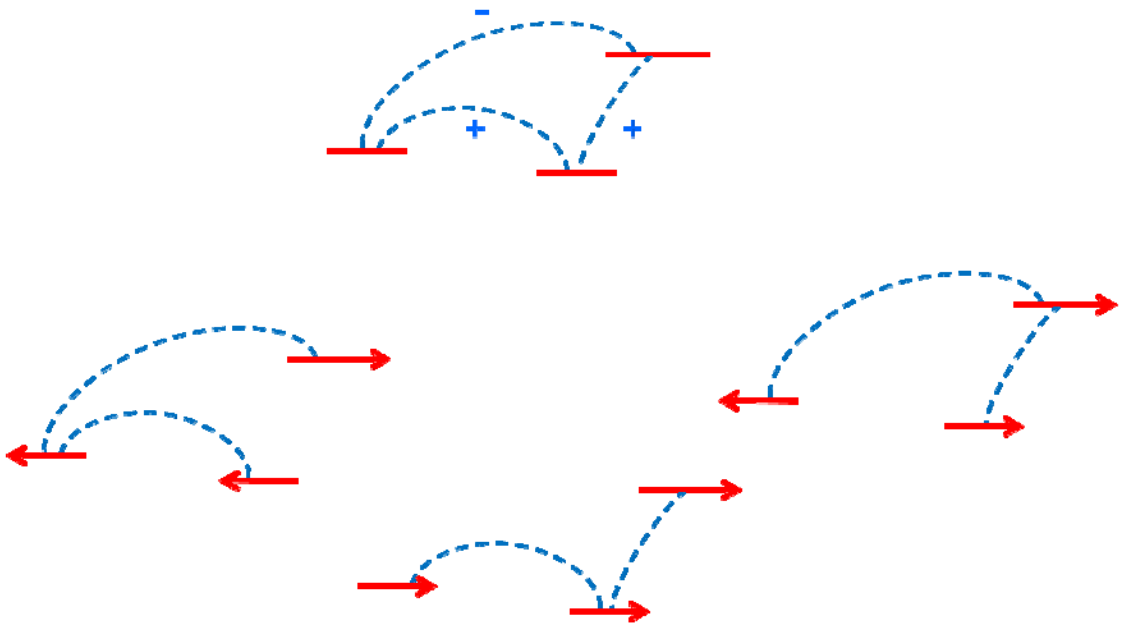


Figure 8.3: Greedy Scaffolding Algorithms – The greedy algorithms introduce a single linker at a time. As long the new linker does not conflict with the previous set, then it is added. If there exists an inconsistency, the order that the linkers are added can have a significant impact on the orientation.

A different approach was targeted in the SOPRA scaffolding program [45]. Unlike the greedy algorithms, SOPRA's algorithms are designed to deal with the challenges of scaffolding with HTS short read data using a global approach to linker analysis. SOPRA attempts to balance size and quality of the final assembly by selecting a subset of mate pairs. SOPRA sets the up the mate pair scaffolding assembly as an optimization problem of the contig connectivity graph where vertices represent contigs and edges represent the links between them. Since contig assembly can be affected by chimeric reads, repetitive regions, and misassemblies, by optimizing over the entire graph at once, these problematic regions are easily identified and removed. This process continues until a consistent set of constraints is reached. This global approach is more effective with paired reads from HTS platforms than the greedy algorithms when one

considers the size of the paired read libraries and the propensity for contig assembly to have issues. While the scaffolds assembled by SOPRA are sometimes smaller than other scaffolders, there is higher confidence in the scaffolds assembled [45]. While significant improvements can be made over contig assembly, there are limitations to scaffolding that are tied to contig assembly. If the contig assembly is falls below a certain size, the scaffold assembly tends to follow (Table 8.2).

8.3 *De novo* Mate Pair Assembly:

In order to better understand mate pair assembly and scaffolding, a project was undertaken to assemble Desulfoluna spongifila, strain AA1. First, the reads were prefiltered using the mean filter set, i.e. all reads must have an average QV score of twenty (or nineteen in 2 datasets) to pass. In initial runs, the reads were trimmed to forty base pairs long. Two assemblers were used for comparison of mate pair assembly. Velvet [32] was used as both a mate pair assembler as well as a contig assembler for SOPRA's scaffolding and translation module [45]. The critical parameters, as with the bacteriophages, were kmer length, the coverage cutoffs, and the minimum coverage of the links.

Many different assemblies were tried with various truncation rates as well as modifying the reads using the SAET program. The goal of SAET is for pre-assembly error correction of the SOLiD reads taking advantage of the high throughput and two-base encoding of the data. This tool does a spectral analysis similar to that described in EULER-SR [33]. In contrast to ABI's claims and increase in the alignment of reads seen in the Waksman Genomics Core Facility here at Rutgers, the desulfoluna dataset

produced consistently shorter N50s with use of this tool(Table 8.1). While the parameter space of the tool was not explored, the documentation on SAET claims improvement, obviously it is not as simple as it would seem. Contact with ABI's bioinformaticians was made but the issue has not been resolve.

		F3	R3	Total	4.5 mb	Contig	Scaffold
	MP	orphans	orphans	Reads	cov	N50	n50
m19	26,391,621	7,811,582	8,859,360	69,454,184	617.37	1,237	44,808
m20	23,204,752	8,199,267	9,139,891	63,748,662	566.65	1,222	58,544
SAET m19	25,754,796	8,079,668	8,837,637	68,426,897	608.24	712	30,867
SAET m20	22,647,596	8,433,022	9,096,313	62,824,527	558.44	1,003	4,510

Table 8.1: Different Parameters and the Desulfoluna Assembly - This show the filtering criteria (mean QV >= 19 or 20), resulting coverage and resulting best assemblies under the best set of assembly parameters for those datasets. The SAET results do not line up with the ABI claims that SAET improves assembly with naïve use. These reads have been truncated to 40 bp.

To improve assembly from the figures initially calculated, the reads were additional truncated; the final assembly length was thirty-two, and then filtered for mean quality of greater or equal to twenty. In the end, these modifications produced the a final assembly that produced scaffold N50s of greater than 200,000. Table 8.2 contains some of the assembly and scaffolding statistics under some of the different conditions used. Truncation to 40 bp resulted in terrible contig assembly. The following scaffold assembly is not particularly good, falling far below the final scaffolding N50s. This is mostly caused low quality bases that were permitted to remain in the tail of some of the reads. Interestingly, the contig assembly under the finalized parameters is no better than that of the bacteriophages described above, but the mate pair library was able to rescue the assembly growing the scaffolds N50s to approximately two hundred thousand. The

best scaffold spans close to half a million base pairs of the desulfoluna genome. While Velvet [32] builds very long scaffold, experience has shown they can contain significant errors [45, 59]. Further analysis is required to better understand the final scaffolding of velvet under these two sets of parameters. Unfortunately, there is no known relative able to be used to extend the contigs, give additional confidence in the assembly, or help fill in the gaps as was done with the bacteriophages. Annotation is the next step and will begin shortly.

column	(1)	(2)	(3)	(4)	(5)	=[(2)+(3)]/(1)
	Total reads Original Solid (50 bp)	Filter Parameter	Truncation Parameter	Filtered & Truncated (32 bp)		% of Total
Desulfoluna	Output	Mean	Mean	MP	Orphans	Reads
F3	66,311,716	20	32	26,777,606	7,643,461	51.91%
R3	66,254,565	20	32	26,777,606	8,909,976	53.86%
Sum Reads	**132,566,281**	**20**	**32**	**53,555,212**	**16,553,437**	**52.89%**
F3	66,311,716	20	40	23,204,752	8,199,267	47.36%
R3	66,254,565	20	40	23,204,752	9,139,891	48.82%
Sum Reads	**132,566,281**	**20**	**40**	**46,409,504**	**17,339,158**	**48.09%**

column	=[(2)+(3)]*32/				
	Truncation Parameter	4.5 Mbp Est. Fold	Contig N50	Scaffold N50	Scaffold N50
Desulfoluna	Mean	Coverage	Velvet	Velvet	SOPRA
Sum Reads	**32**	**499**	1,467	169,097	204,606
Sum Reads	**40**	**453**	269	59,187	5,899

Table 8.2: Desulfoluna Assembly - This show the filtering results for the final assembly (mean QV >= 20), A comparison can be drawn between contig assembly and scaffolding. With good contig assembly the scaffolding can reduce the highly fructuous nature of contig assembly. With poor contig assembly, the scaffolding also remains subpar.

Even with the use of mate pairs, room for improvement still exists. One possibility includes the use of multiple mate pair libraries with varying linker lengths to overcome the high complexity of large genomes. These multi mate pair libraries would also help to target and build the repetitive regions accurately (modification of Phusion for short reads [46]). The Panda draft genome was able to be assembled by this technique using thirty-seven paired end libraries with linkers varying between 500 bp – 10 kbp. While there is still room for improvement in the Panda genome, a significant portion was able to be assembled. To minimize these gaps left by the assembly using mate pairs, if a closely related genome exists, another possibility is to use some variation of comparative assembly to extend the contigs within the scaffolds. In the end, researchers want to obtain a single continuous genome from the sequencing and assembly process. To reach that goal using HTS platforms, there is still room to improve techniques, algorithms, and pipelines. But understand the interplay between all the parameters, is a good place to begin.

Acknowledgment: The Desulfoluna spongifila, strain AA1 library belongs to Joachim Messing at the Waksman Institute and the strain was provided by Max Haggblom. Discussions with Adel Dayarian on SOPRA and assembly helped improve the final assembly to the state discussed in this chapter.

Epilogue

Short read sequences from the HTS platforms significantly complicate the *de novo* assembly process. There are lots of problematic factors to consider and attempt to mitigate when assembling the genome. On the other hand, the HTS platforms do make sequencing and assembly possible for independent labs. This adds methods to their toolbox that were not available before unless studying a model genome. In the end, *de novo* assembly seems to be evolving. Utilizing reads from several platforms, seems to be the direction many researchers are taking to mitigate the problems of a particular platform and assemble complex genomes [67]. While this adds a level computational complexity, dealing with various read lengths and mate pair libraries, the results are expected to be impressive. One type of assembly pipeline might call for Illumina reads for contig assembly, then aligning the SOLiD mate pair ends to the assembled contigs and using both the paired end and mate pair data to finalize assembly. The long linkers can provide confidence in the repetitive regions and the short ones aid for local assembly. This is only one option, for large genomes large sequencing consortiums have been formed with the goal of combining various platforms for exactly this purpose. The project targets sequencing on multiple platforms with high coverage in various labs to combine the information for a draft genome. Some examples include the Strawberry and Cranberry consortiums.

With the single molecule technologies just on the horizon, this added sequencing protocol might allow for an explosion in *de novo* assembly and resequencing projects since they target very long read lengths and there is removal of the PCR amplification step. This might allow for sequencing of regions that have proved difficult for Sanger

and HTS platforms. With the continual improvement of error rates on these third generation sequencing platforms, the combination of the long reads and the ease of creating paired read libraries from the HTS platforms might prove to be the ideal pipeline for *de novo* assembly.

In the end, the knowledge gained by having a myriad of genomes at the disposal of the scientific community is priceless. Better techniques on all the platforms, computational options and algorithms, and interest are expanding on a regular basis. Therefore, as time progresses, ways of dealing with these short reads ultimate improves along the way. This makes the possibility of assembling genomes with short reads difficult but a reality nevertheless.

References

1. Hayden EC: **Personalized cancer therapy gets closer**. *Nature* 2009, **458**:131-132.
2. Sanger F, Coulson AR: **A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase**. *J Mol Biol* 1975, **94**(3):441–448.
3. Sanger F, Nicklen S, Coulson AR: **DNA sequencing with chain-terminating inhibitors**. *PNAS* 1977, **74**(12):5463-5467.
4. Maxam AM, Gilbert W: **A new method for sequencing DNA**. *Proc Natl Acad Sci* 1977, **74**(2): 560-564.
5. **Human Genome Project official information page** [http://www.ornl.gov/sci/techresources/Human_Genome/home.shtml]
6. Anderson S: **Shotgun DNA sequencing using cloned DNase I-generated fragments**. *Nucleic Acids Research* 1981 **9**(13):3015–3027.
7. Staden R: **A strategy of DNA sequencing employing computer programs**. *Nucleic Acids Research* 1979, **6**(7):2601-2610.
8. Myers E, Sutton G, Delcher A, Dew I, Fasulo D, Flanigan M, Kravitz S, Mobarry C, Reinert K, Remington K: **A whole-genome assembly of Drosophila**. *Science* 2000, **287**(5461):2196 - 2204.
9. Marshall E: **HUMAN GENOME: A High-Stakes Gamble on Genome Sequencing**. *Science* 1999, **284**(5422):1906 - 1909.
10. Venter JC, Adams MD, Myers EW, Li PW, Mural RJ, Sutton GG, Smith HO, Yandell M, Evans CA, Holt RA *et al*: **The Sequence of the Human Genome**. *Science* 2001, **291**(5507):1304-1351.
11. Service RF: **Gene Sequencing: The Race for the $1000 Genome**. *Science* 2006, **311**(5767):1544-1546.
12. Margulies M, Egholm M, Altman WE, Attiya S, Bader JS, Bemben LA, Berka J, Braverman MS, Chen Y-J, Chen Z *et al*: **Genome sequencing in microfabricated high-density picolitre reactors**. *Nature* 2005, **437**:376-380.
13. Shendure J, Porreca GJ, Reppas NB, Lin X, McCutcheon JP, Rosenbaum AM, Wang MD, Zhang K, Mitra RD, Church GM: **Accurate Multiplex Polony Sequencing of an Evolved Bacterial Genome**. *Science* 2005, **309**:1728-1732.
14. Rothberg JM, Leamon JH: **The development and impact of 454 sequencing**. *Nature Biotechnology* 2008, **26**(10):1117-11124.
15. Chi KR: **The year of sequencing**. *Nature Methods* 2008, **5**(1):11-14.
16. Bentley DR, Balasubramanian S, Swerdlow HP, Smith GP, Milton J, Brown CG, Hall KP, Evers DJ, Barnes CL, Bignell HR *et al*: **Accurate whole human genome sequencing using reversible terminator chemistry**. *Nature* 2008, **456**:53-59.
17. Rusk N, Kiermer V: **Primer: Sequencing--the next generation**. *Nature Methods* 2008, **5**(1):15.
18. Smith DR, Quinlan AR, Peckham HE, Makowsky K, Tao W, Woolf B, Shen L, Donahue WF, Tusneem N, Stromberg MP *et al*: **Rapid whole-genome mutational profiling using next-generation sequencing technologies**. *Genome Res* 2008, **18**:1638-1642.
19. **Applied Biosystems Surpasses Industry Milestone in Lowering the Cost of Sequencing Human Genome** [http://phx.corporate-ir.net/phoenix.zhtml?c=61498&p=irol-abiNewsArticle&ID=1207606&highlight=]
20. **Brochure: SOLiD™ System Sequencing: See the Difference** [http://www3.appliedbiosystems.com/cms/groups/mcb_marketing/documents/generaldocuments/cms_061241.pdf]
21. Edwards D, Batley J: **Plant genome sequencing: applications for crop improvement.** *Plant Biotechnol J* 2010, **8**(1):2-9.
22. Imelfort M, Edwards D: **De novo sequencing of plant genomes using second-generation technologies**. *Brief Bioinform* 2009, **10**:609-618.
23. Pop M: **Genome assembly reborn: recent computational challenges**. *Brief Bioinform* 2009, **10**(4):354 - 366.
24. Pop M, Salzberg SL: **Bioinformatics challenges of new sequencing technology**. *Trends in Genetics* 2008, **24**(3):142-149.
25. Kececioglu J, Myers E: **Combinatorial Algorithms for DNA-Sequence Assembly**. *Algorithmica* 1995, **13**(1-2):7 - 51.

26. Batzoglou S, Jaffe DB, Stanley K, Butler J, Gnerre S, Mauceli E, Berger B, Mesirov JP, Lander ES: **ARACHNE: a whole-genome shotgun assembler**. *Genome Res* 2002, **12**(1):177–189.
27. Myers EW, Sutton GG, Delcher AL, Dew IM, Fasulo DP, Flanigan MJ, Kravitz SA, Mobarry CM, Reinert KH, Remington KA *et al*: **A whole-genome assembly of Drosophila.** *Science* 2000, **287**(5461):2196-2204.
28. Li R, Fan W, Tian G, Zhu H, He L, Cai J, Huang Q, Cai Q, Li B, Bai Y *et al*: **The sequence and de novo assembly of the giant panda genome**. *Nature* 2010, **463**(7279):311-317.
29. Warren R, Sutton G, Jones S, Holt R: **Assembling millions of short DNA sequences using SSAKE**. *Bioinformatics* 2007, **23**(4):500 - 501.
30. Jeck W, Reinhardt J, Baltrus D, Hickenbotham M, Magrini V, Mardis E, Dangl J, Jones C: **Extending assembly of short DNA sequences to handle error**. *Bioinformatics* 2007, **23**(21):2942 - 2944.
31. Dohm JC, Lottaz C, Borodina T, Himmelbauer H: **SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing**. *Genome Res* 2007, **17**:1697-1706.
32. Zerbino D, Birney E: **Velvet: algorithms for de novo short read assembly using de Bruijn graphs**. *Genome Res* 2008, **18**(5):821 - 829.
33. Chaisson M, Pevzner P: **Short read fragment assembly of bacterial genomes**. *Genome Res* 2008, **18**(2):324 - 330.
34. Chaisson MJP, Brinza D, Pevzner PA: **De novo fragment assembly with short mate-paired reads: Does read length matter?** *Genome Res* 2009 **19**(2):336-346.
35. Hernandez D, Francois P, Farinelli L, Osteras M, Schrenzel J: **De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer**. *Genome Res* 2008, **18**(5):802 - 809.
36. Li R, Li Y, Kristiansen K, Wang J: **SOAP: short oligonucleotide alignment program**. *Bioinformatics* 2008, **24**:713-714
37. Simpson JT, Wong K, Jackman SD, Schein JE, Jones SJ, Birol I: **ABySS: a parallel assembler for short read sequence data**. *Genome Res* 2009, **19**(6):1117-1123.
38. Butler J, MacCallum I, Kleber M, Shlyakhter I, Belmonte M, Lander E, Nusbaum C, Jaffe D: **ALLPATHS: de novo assembly of whole-genome shotgun microreads**. *Genome Res* 2008, **18**(5):810 - 820.
39. Idury RM, Waterman MS: **A new algorithm for DNA sequence assembly**. *J Computat Biol* 1995, **2**:291-306.
40. Pevzner PA, Tang HX, Waterman MS: **An Eulerian path approach to DNA fragment assembly**. *PNAS* 2001, **98**:9748-9753.
41. Farrer RA, Kemen E, Jones JDG, Studholme DJ: ***De novo* assembly of *Pseudomonas syringae* pv *syringae* B728a genome using Illumina/Solexa short sequence reads**. *FEMS Microbiology Letters* 2009, **291**(1):103-111.
42. Huson D, Reinert K, Myers E: **The greedy path-merging algorithm for Contig Scaffolding**. *Journal of the Acm* 2002, **49**(5):603 - 615.
43. Pevzner P, Tang H: **Fragment assembly with double-barreled data**. *Bioinformatics* 2001, **17**(Suppl 1):S225 - 233.
44. Pop M, Kosack D, Salzberg S: **Hierarchical scaffolding with Bambus**. *Genome Research* 2004, **14**(1):149 - 159.
45. Dayarian A, Michael T, Sengupta A: **SOPRA: Scaffolding algorithm for paired reads via statistical optimization**. *BMC Bioinformatics* 2010, **11**(1):345.
46. Mullikin JC, Ning Z: **The Phusion Assembler**. *Genome Res* 2003, **13**:81-90.
47. Lander ES, Waterman MS: **Genomic Mapping by Fingerprinting Random Clone: A Mathematical Analysis**. *Genomics* 1988, **2**:231-239.
48. Chaisson M, Pevzner P, Tang H: **Fragment assembly with short reads**. *Bioinformatics* 2004, **20**(13):2067-2074.
49. Whiteford N, Haslam N, Weber G, Prugel-Bennett A, Essex JW, Roach PL, Bradley M, Neylon C: **An analysis of the feasibility of short read sequencing**. *Nucl Acids Res* 2005, **33**(19):e171-.
50. Deininger PL: **Random subcloning of sonicated DNA: Application to shotgun DNA sequence analysis**. *Analytical Biochemistry* 1983, **129**(1):216-223.
51. Locke G, Tolkunov D, Moqtaderi Z, Struhl K, Morozov AV: **High-throughput sequencing reveals a simple model of nucleosome energetics**. 2010:arXiv:1003.4044 (In Review).

52. Sasson A, Michael T: **Filtering error from SOLiD Output**. *Bioinformatics* 2010, **26**(6):849 - 850.
53. McKernan K, Peckham H, Costa G, McLaughlin S, Fu Y, Tsung E, Clouser C, Duncan C, Ichikawa J, Lee C: **Sequence and structural variation in a human genome uncovered by short-read, massively parallel ligation sequencing using two-base encoding**. *Genome Res* 2009, **19**(9):1527 - 1541.
54. Lynch M, Conery JS: **The Origins of Genome Complexity**. *Science* 2003, **302**(5649):1401-1404.
55. Schatz MC, Trapnell C, Delcher AL, Varshney A: **High-throughput sequence alignment using Graphics Processing Units**. *BMC Bioinformatics* 2007, **8**:474-483.
56. **Dibase sequencing allows accurate SNP detection at moderate and low coverage with diBayes algorithm** [http://www3.appliedbiosystems.com/cms/groups/mcb_marketing/documents/generaldocuments/cms_065554.pdf]
57. Valouev A, Ichikawa J, Tonthat T, al. e: **A high-resolution, nucleosome position map of C. elegans reveals a lack of universal sequence-dictated positioning**. *Genome Res* 2008(18):1051-1063.
58. Marioni JC MC, Mane SM, Stephens M, Gilad Y.: **RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays.** *Genome Res* 2008, **18**(9):1509-1517.
59. Farrer R, Kemen E, Jones J, Studholme D: **De novo assembly of the Pseudomonas syringae pv. syringae B728a genome using Illumina/Solexa short sequence reads**. *FEMS Microbiol Lett* 2009, **291**(1):103 - 111.
60. Dayarian A, Michael T, Sengupta AM: **SOPRA: an algorithm for de novo assembly of paired reads**. *In Review* 2009.
61. Zerbino DR, Birney E: **Velvet: Algorithms for de novo short read assembly using de Bruijn graphs**. *Genome Res* 2008, **18**:821-829.
62. Chaisson MJP, Brinza D, Pevzner PA: **De novo fragment assembly with short mate-paired reads: Does read length matter?** *Genome Res* 2008(online printing).
63. Langmead B, Trapnell C, Pop M, Salzberg SL: **Ultrafast and memory-efficient alignment of short DNA sequences to the human genome** *Genome Biol* 2009, **10**:R25.
64. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ: **Basic local alignment search tool**. *J Mol Biol* 1990, **215**:403-410.
65. Gnerre S, Lander ES, Lindblad-Toh K, Jaffe DB: **Assisted assembly: how to improve a *de novo* genome assembly by using related species**. *Genome Biology* 2009, **10**:R88.
66. Salzberg S, Sommer D, Puiu D, Lee V: **Gene-boosted assembly of a novel bacterial genome from very short reads**. *PLoS Comput Biol* 2008, **4**(9):e1000186.
67. Nagarajan H, Butler JE, Klimes A, Qiu Y, Zengler K, Ward J, Young ND, Methe BA, Palsson BØ, Lovley DR *et al*: **De Novo Assembly of the Complete Genome of an Enhanced Electricity-Producing Variant of Geobacter sulfurreducens Using Only Short Reads**. *PLoS One* 2010, **5**(6):e10922.

Appendix A

Additional Preprocessing Details

C. elegans	Original # of reads F3	Original # of reads R3	No Filter				
			# of mate pairs	# F3 orphans	# R3 orphans	% F3 reads Retained	% R3 reads Retained
Quad 1 mate pair	40,038,192	39,860,989	39,724,467	313,725	136,522	100.0%	100.0%
Quad 2 mate pair	42,701,697	41,850,915	41,736,336	965,361	114,579	100.0%	100.0%
			Default (p1 & e3) error analysis				
Quad 1 mate pair	40,038,192	39,860,989	7,943,943	4,675,817	7,301,058	31.5%	38.2%
Quad 2 mate pair	42,701,697	41,850,915	8,793,085	5,752,471	7,595,014	34.1%	39.2%
			P_1 & e_5 error analysis				
Quad 1 mate pair	40,038,192	39,860,989	11,226,961	4,872,740	7,354,185	40.2%	46.6%
Quad 2 mate pair	42,701,697	41,850,915	12,422,596	5,959,005	7,442,328	43.0%	47.5%
			P_5 & e_0 error analysis				
Quad 1 mate pair	40,038,192	39,860,989	1,679,549	2,654,397	4,472,967	10.8%	15.4%
Quad 2 mate pair	42,701,697	41,850,915	1,783,402	3,268,438	4,692,045	11.8%	15.5%

Table A.1: Filtering with Mate Pairs - Detailed information on *C. elegans* quad 1 and 2 mate pair (25 bp per end) run pre-error and post-error analysis. The following error settings were used for all the above mentioned datasets: 1) off, 2) polyclonal count of 1 error count of 5, 3) default settings (polyclonal count of 1 error count of 3) and 4) polyclonal count of 5 and error count of 0. From these four different settings the dataset reduction due to the stringency of the filter can be seen.

Arabadopsis thaliana matching results under various filtering conditions

	Mismatches				
A. thaliana	**0 mismatches**	**1 mismatch**	**2 mismatches**	**3 mismatches**	**Total**
Full F3 Dataset	5,152,120	2,921,867	2,202,437	1,774,558	**12,050,982**
Full R3 Dataset	5,972,709	3,087,024	2,201,168	1,696,394	**12,957,295**
Filter 1 F3	4,949,159	2,716,941	2,003,830	1,581,827	**11,251,757**
Filter 1 R3	5,803,223	2,905,530	2,019,155	1,518,166	**12,246,074**
Filter 2 F3	4,347,800	2,198,867	1,544,475	1,168,644	**9,259,786**
Filter 2 R3	5,284,683	2,455,279	1,626,819	1,173,662	**10,540,443**
Filter 3 F3	3,471,080	1,597,701	1,070,999	779,719	**6,919,499**
Filter 3 R3	4,551,952	1,947,822	1,235,931	861,210	**8,596,915**
Filter 4 F3	3,938,893	1,274,652	517,065	209,746	**5,940,356**
Filter 4 R3	4,876,081	1,569,031	640,999	257,440	**7,343,551**
Filter 5 F3	3,117,071	680,877	212,333	70,483	**4,080,764**
Filter 5 R3	4,013,937	899,266	274,467	84,844	**5,272,514**
Filter 6 F3	2,943,329	644,441	203,326	68,089	**3,859,185**
Filter 6 R3	3,872,569	869,735	267,875	83,389	**5,093,568**
Filter 7 F3	740,913	18,322	14,544	3,037	**776,816**
Filter 7 R3	1,135,346	22,082	17,128	2,725	**1,177,281**

Table A.2: Mismatching Details - The mapping results for different filtering criteria analyzed by ABI's CoronaLite. For different filtering criteria, the number of aligned reads is presented for different number of allowed mismatches. The number of perfect matches stays relatively high with very little impact to mapping until the final filter which has a dramatic reduction in coverage.

Filering Program Inputs

Input Name	Required Options and Defaults	Description
-i or –input_type	No – default is set for a fragment run	The type of data - identifies mate pair or fragment analysis. If the -i option is left blank or excluded a Fragment analysis will be preformed, and if -r and -s options are used in such a case, they will be ignored. For mate pair analysis, the mate pair option must be selected. [F, Frag, Fragment, f, frag, fragment, mp, MP, mate-pair, Mate-pair]
-f or –f3	Yes	csfasta file for analysis - this is the name of the file coming from the SOLiD primary analysis. If this is analysis for mate pairs the F3 file set must be passed in here and the mates under the –r option.
-g or –f3QV	No	Corresponding quality file for the –f option. If this is left blank, the name of the corresponding QV file must match and be in the same location as the csfasta file except the ending will replace the .csfasta with _QV.qual.
-r or –r3	No	csfasta file for mate pair analysis - this is the name of the mates' csfasta file to the f3 option. If mate pair analysis is turned on then this field becomes required.
-s or –r3QV	No	Corresponding quality file for the –r option. If this is left blank, the name of the corresponding QV file must match and be in the same location as the csfasta file except the ending will replace the .csfasta with _QV.qual.
-x or –poly_analysis	No – default is on	Polyclonal analysis on/off - Polyclonal analysis looks only at the first 10 color calls. It asks that within those first 10 calls there must be a certain number (p|p_cnt) of qv scores that exceed the qv score of interest (q|p_qv). [on, yes, y, off, no, n]
-p or –p_cnt	No – default is 1	Polyclonal analysis count required - This is the count required for the polyclonal analysis. This number must be between [0-10]. Zero is equivalent to having the polyclonal analysis turned off.

-q or –p_qv	No – default is a quality score of 25	Polyclonal analysis minimum QV score - This is the minimum score for the polyclonal analysis. This must be a number between 0-50. Please be aware that scores above 34 are exceedingly rare.
-y or –error_analysis	No – default is on	error analysis on/off - Error analysis looks at the entire read for quality scores that fall below the error score passed (e_sc). These calls are counted, and the total number of these erroneous calls must be under the err_cnt passed. [on, yes, y, off, no, n]
-e or –e_cnt	No – default is 3	error analysis maximum error count allowed - This is the maximum number of errors allowed per read. If the number is greater than the read length it is equivalent to having the error analysis turned off.
-d or –e_sc	No – default is a quality score of 10	error analysis maximum QV score - This is the maximum score for error analysis. This must be a number between 0-50. Please be aware that scores above 34 are exceedingly rare.
-n or –neg_qv	No – default is off	Removal of all reads containing a missed color call on/off – If removal of all reads containing missing color calls (identified by negative quality scores), this flag must be turned on. [on, yes, y, off, no, n]
-t or –trunk	No – default is off	truncation of reads on/off - If truncation of reads is desired, this flag must be turned on. If turned on, option u must be used as well. [on, yes, y, off, no, n]
-u or –tr_len	No	length of desired read after truncation - This is the length of the sequence desired, any color calls after this length are removed. This option must be filled in if truncation is turned on and be an integer greater than 0.
-a or –qv_analysis	No	Analysis of the quality values for all of the inputted reads and the passing reads. Analysis returns a file with a matrix of a count of scores by position.
-o or –output	Yes	output file name - this is the beginning of the name for the output information. The endings are filled in as needed.
-v or -ouput_qv	No – default is on	Output matching QV files on/off – this will print the matching QV files to the outputted csfasta files. [on,off]

Table A.3: Table of inputs – Inputs and switches into the error analysis framework.

Appendix B

Visualizations of the Phage Dataset and Assembly

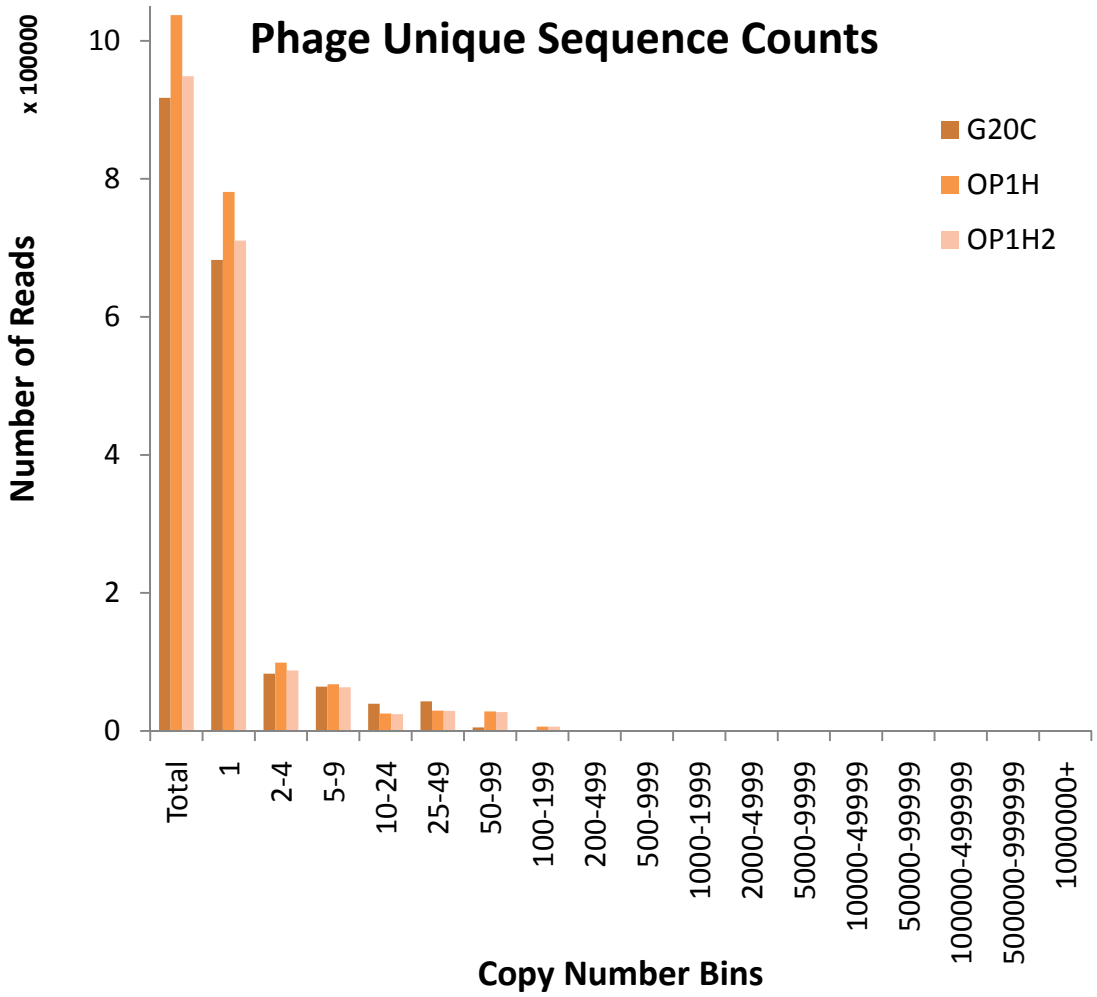


Figure B.1: Unique Reads - With such a high final coverage, approximately 1000 fold, excluding XP12 and G17 from in depth consideration, one would expect relatively high coverage of reads, high counts of the same read. The reality is that the majority of reads are unique seeming to indicate a relatively high error rate. One would expect a long tail, but this result seems relatively surprising [49].

Bacteriophage Alignment Results

column	(1)	(2)	(3)	(4)	(5)
		Uniquely mapped Reads			
Phage-Ref	Filtered & Truncated Reads	0 Mismatches	1 Mismatches	2 Mismatches	3 Mismatches
OP1H-OP1	3,515,413	230,205	87,323	291,589	126,361
OP1H2-OP1	3,260,577	269,348	100,158	298,836	129,805
OP1H-XP10	3,515,413	377,477	135,481	301,426	131,899
OP1H2-XP10	3,260,577	709,454	248,804	352,468	150,902
G20C-P23_45	2,305,365	376,594	168,509	401,476	189,087

column	(6)	(7)	(8)	(9)
			Uniquely mapped Reads	
Phage-Ref	4 Mismatches	5 Mismatches	6 Mismatches	Total Mismatches
OP1H-OP1	251,402	137,982	227,391	**4,106,225**
OP1H2-OP1	248,956	120,237	229,798	**4,113,006**
OP1H-XP10	228,664	120,700	191,306	**3,800,022**
OP1H2-XP10	193,380	101,816	145,242	**3,560,498**
G20C-P23_45	254,881	124,369	132,036	**3,972,307**

Phage-Ref	Total # of Matched Including multi mat.	% uniquely Mapped Vs Non Unique	% uniquely Mapped vs Total Filtered	# of reads per Start Pos	Avg # of reads per Start Pos	# of Valid Adj Errors	# of Bases not covered
OP1H-OP1	1,357,149	99.64%	**38.47%**	43,449	31.12	1,516,362	9,318
OP1H2-OP1	1,412,723	98.90%	**42.85%**	42,973	32.74	1,488,894	9,342
OP1H-XP10	1,496,208	99.38%	**42.30%**	48,740	30.51	1,304,566	7,770
OP1H2-XP10	1,915,746	99.29%	**58.34%**	55,208	34.45	1,036,930	7,821
G20C-P23_45	1,648,508	99.91%	**71.44%**	122,374	13.46	1,270,160	8,702

Table B.1: Phage Alignment Results – These are the alignment results for all the bacteriophages used in the comparative assembly step.

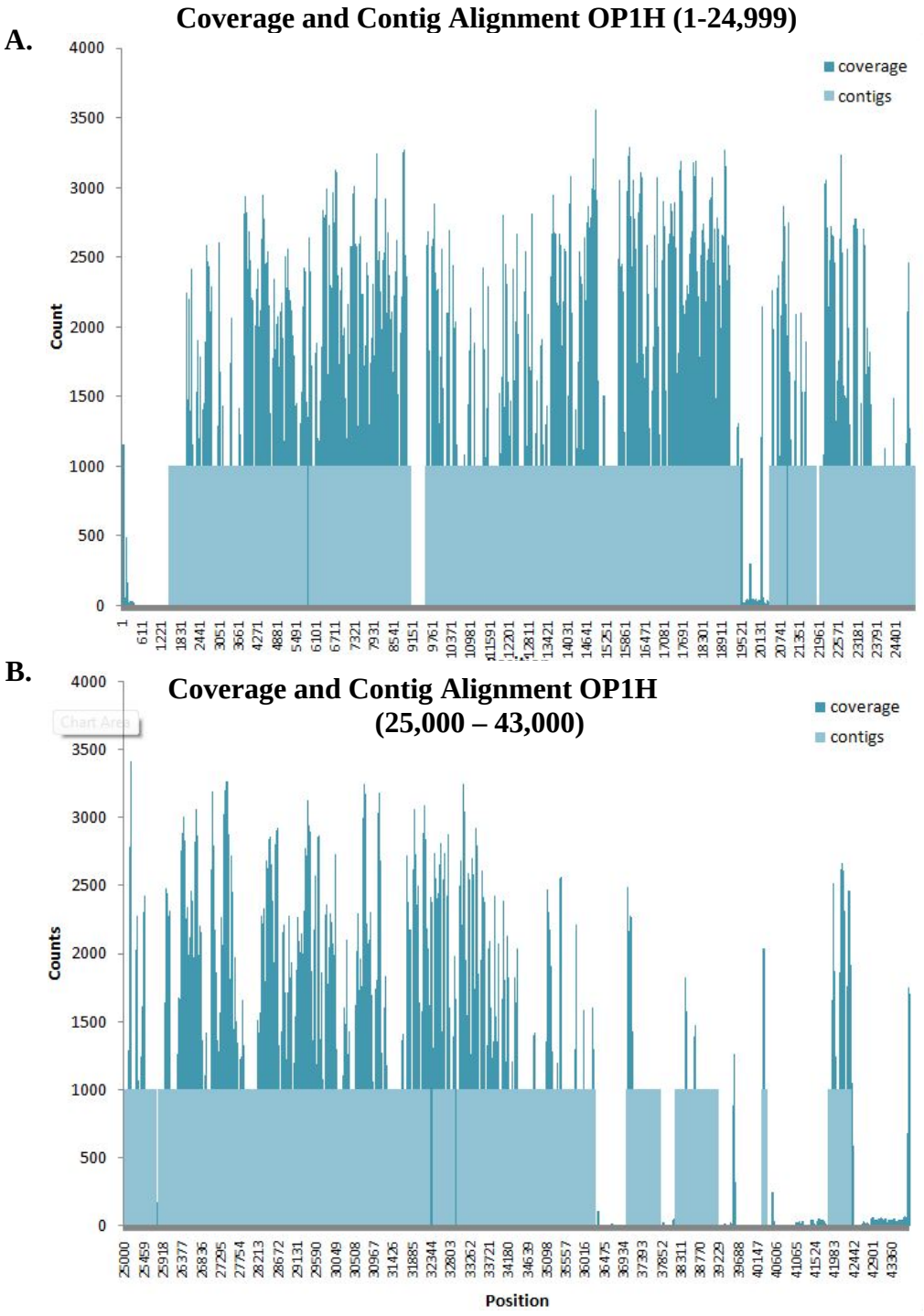
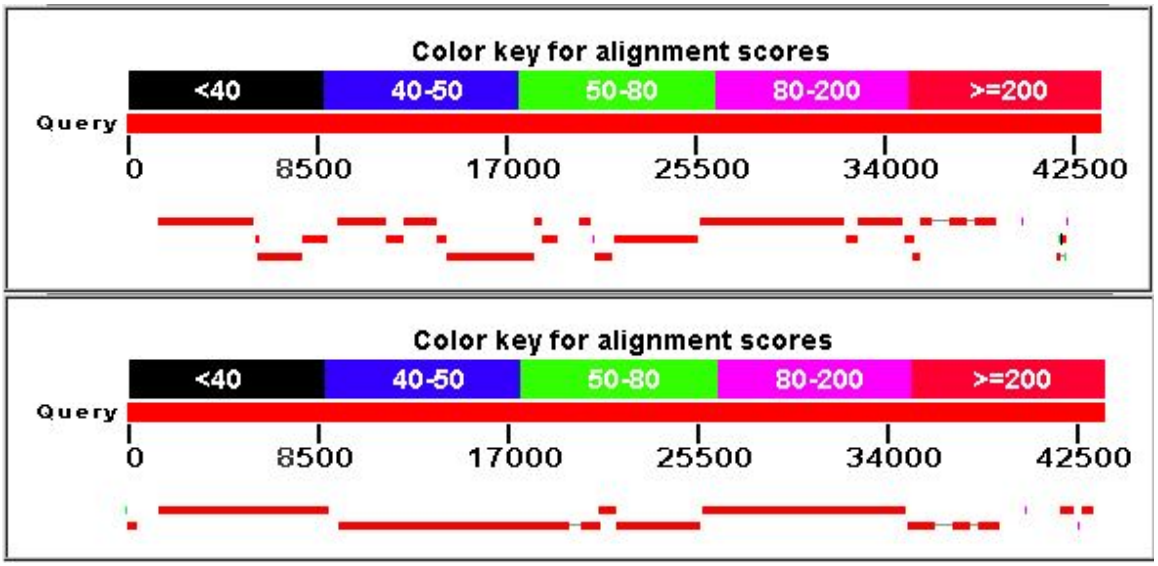


Figure B.2: Coverage and Contig Alignment OP1H – This is the same data as in Figure 6.4 but along the entire genome. The dark blue histogram is raw coverage generated from aligning using CoronaLite (http://solidsoftwaretools.com/gf/) and the light blue are the contig alignments done using Blast [64]. A. Contains positions 1-24,999 B. Contains positions 25,000 – 44,000.

Visualizations of Comparative Genome Assembly OP1H

A. OP1



B. XP10

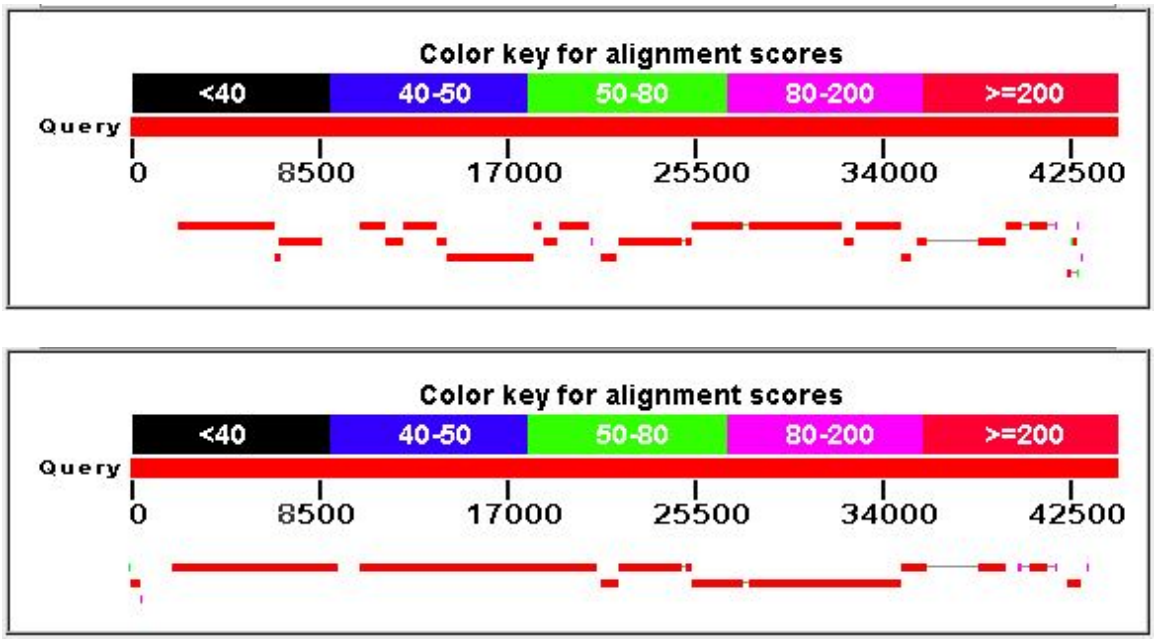
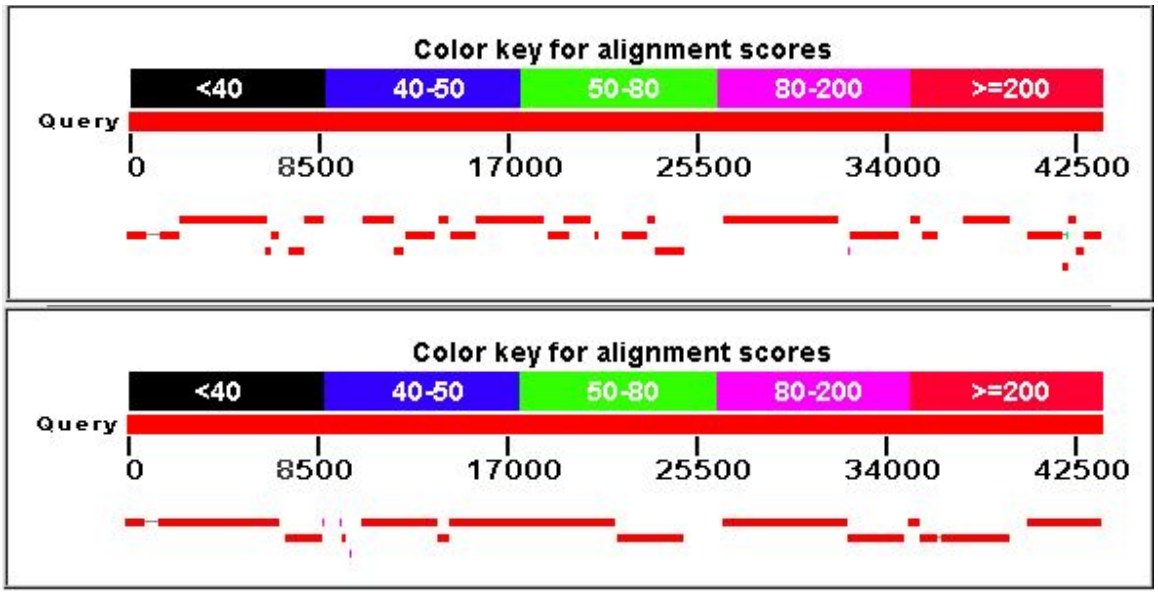


Figure B.3: Visualizations of Comparative Genome Assembly OP1H - Here is a visualization of the improvements made to the assembly. A. The top panel contains the contigs aligned to the reference OP1 post *de novo* assembly. The bottom panel contains the final assembly contigs aligned to the same reference. B. The top panel contains the contigs aligned to the reference XP10 post *de novo* assembly. The bottom panel contains the final assembly contigs aligned to the same reference. As can be seen for both of these, there is a significant improvement in the length and ordering of the reads.

Visualizations of Comparative Genome Assembly OP1H2

A. OP1



B. XP10

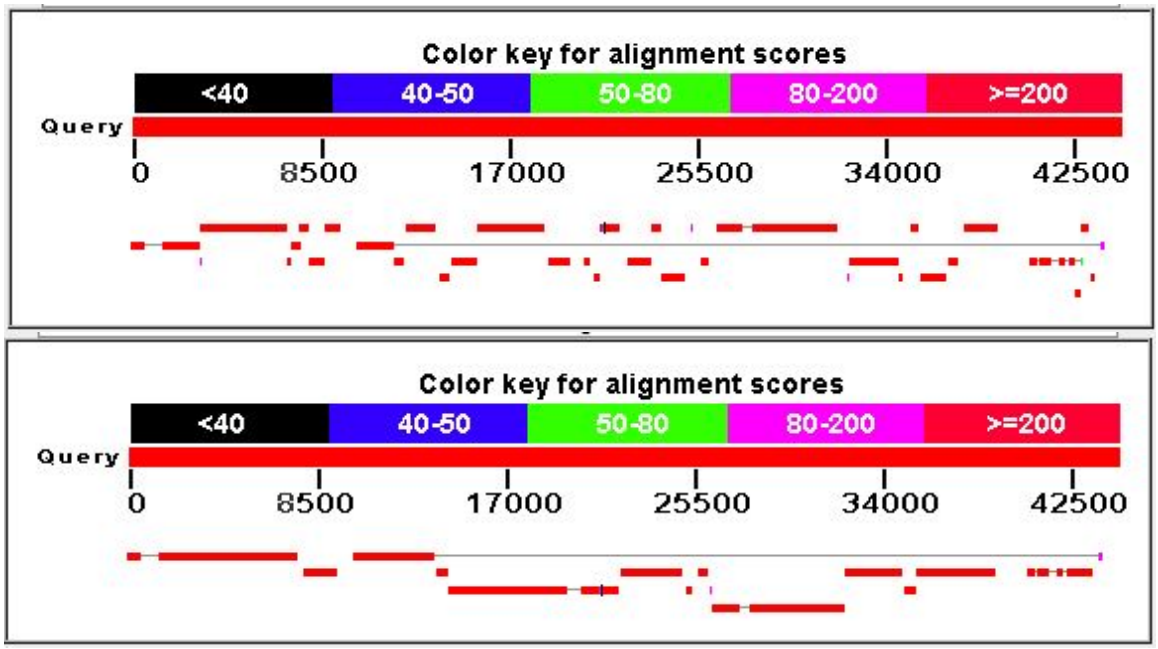


Figure B.4: Visualizations of Comparative Genome Assembly OP1H2 - Here is a visualization of the improvements made to the assembly. A. The top panel contains the contigs aligned to the reference OP1 post *de novo* assembly. The bottom panel contains the final assembly contigs aligned to the same reference. B. The top panel contains the contigs aligned to the reference XP10 post *de novo* assembly. The bottom panel contains the final assembly contigs aligned to the same reference. As can be seen for both of these, there is a significant improvement in the length and ordering of the reads.

Appendix C

Visualizations of the Desulfoluna Dataset

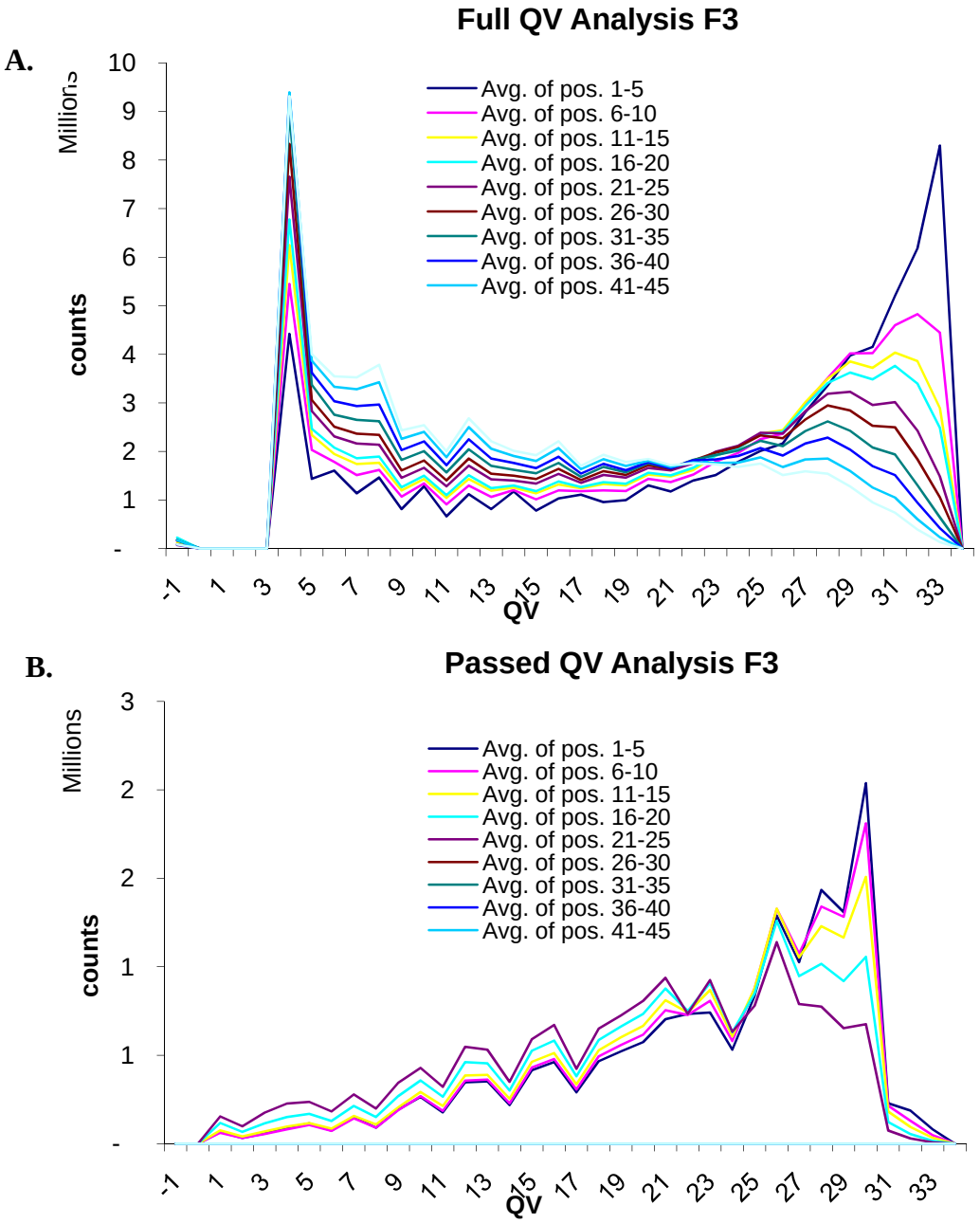


Figure C.1: Desulfoluna Quality Profile F3 – A. This QV profile represents the QV profile for the entire F3 dataset. B. This is the QV profile for all the F3 reads that passed the filter (mean QV >= 20). This shows a significant improvement in the quality of the passing reads over the original dataset.

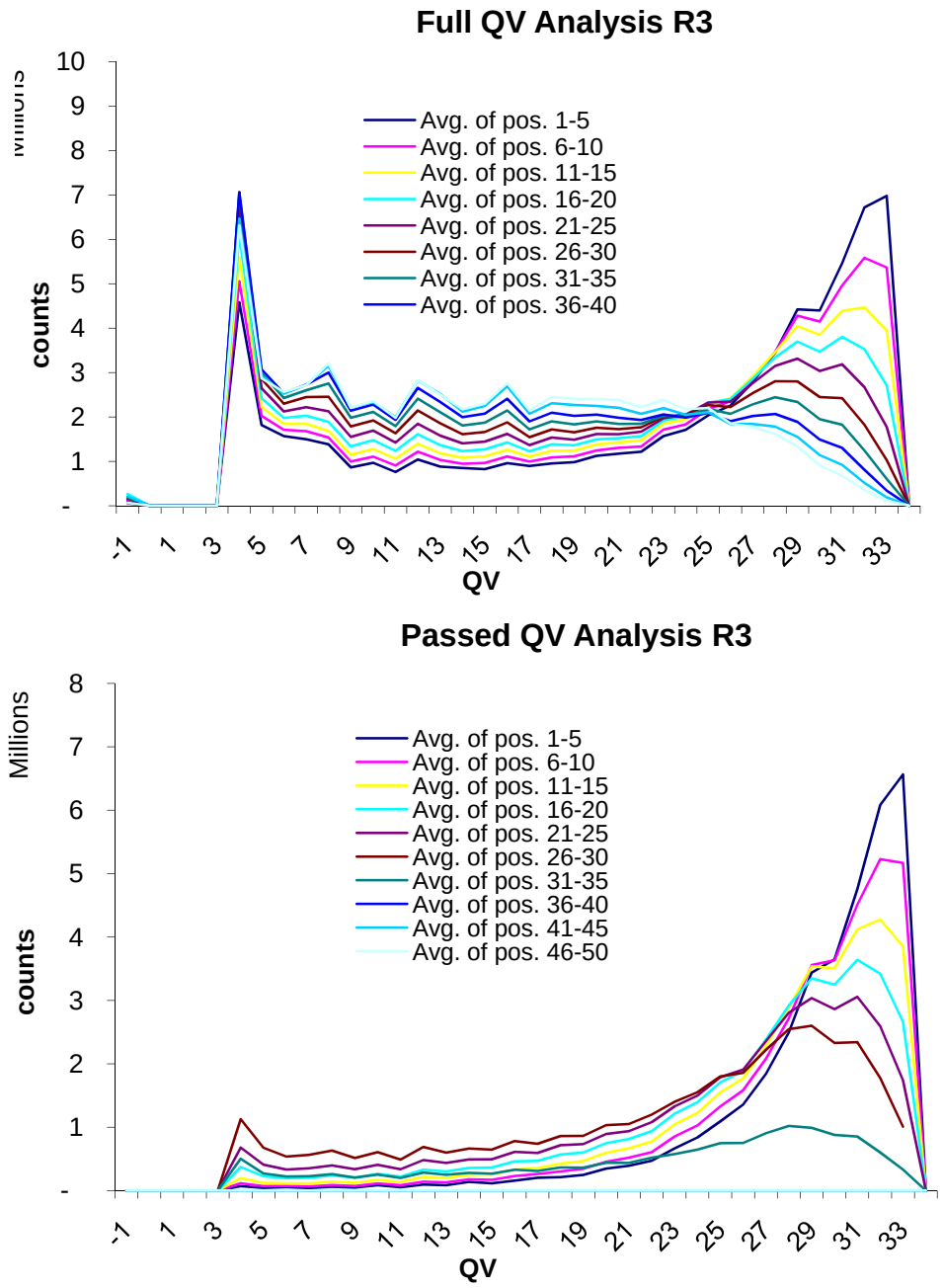


Figure C.2: Desulfoluna Quality Profile R3 - A. This QV profile represents the QV profile for the entire R3 dataset. B. This is the QV profile for all the R3 reads that passed the filter (mean QV >= 20). This shows a significant improvement in the quality of the passing reads over the original dataset.

Appendix D

Library Preparation

Library preparation and SOLiD sequencing was done by Randy Kerstetter and Mark Diamond at the Waksman Core Facility, Waksman Institute, Rutgers University.

The following figures contain the workflows followed to prepare the bacteriophage and Desulfoluna spongifila, strain AA1 libraries.

Barcoded Fragment Library Preparation:

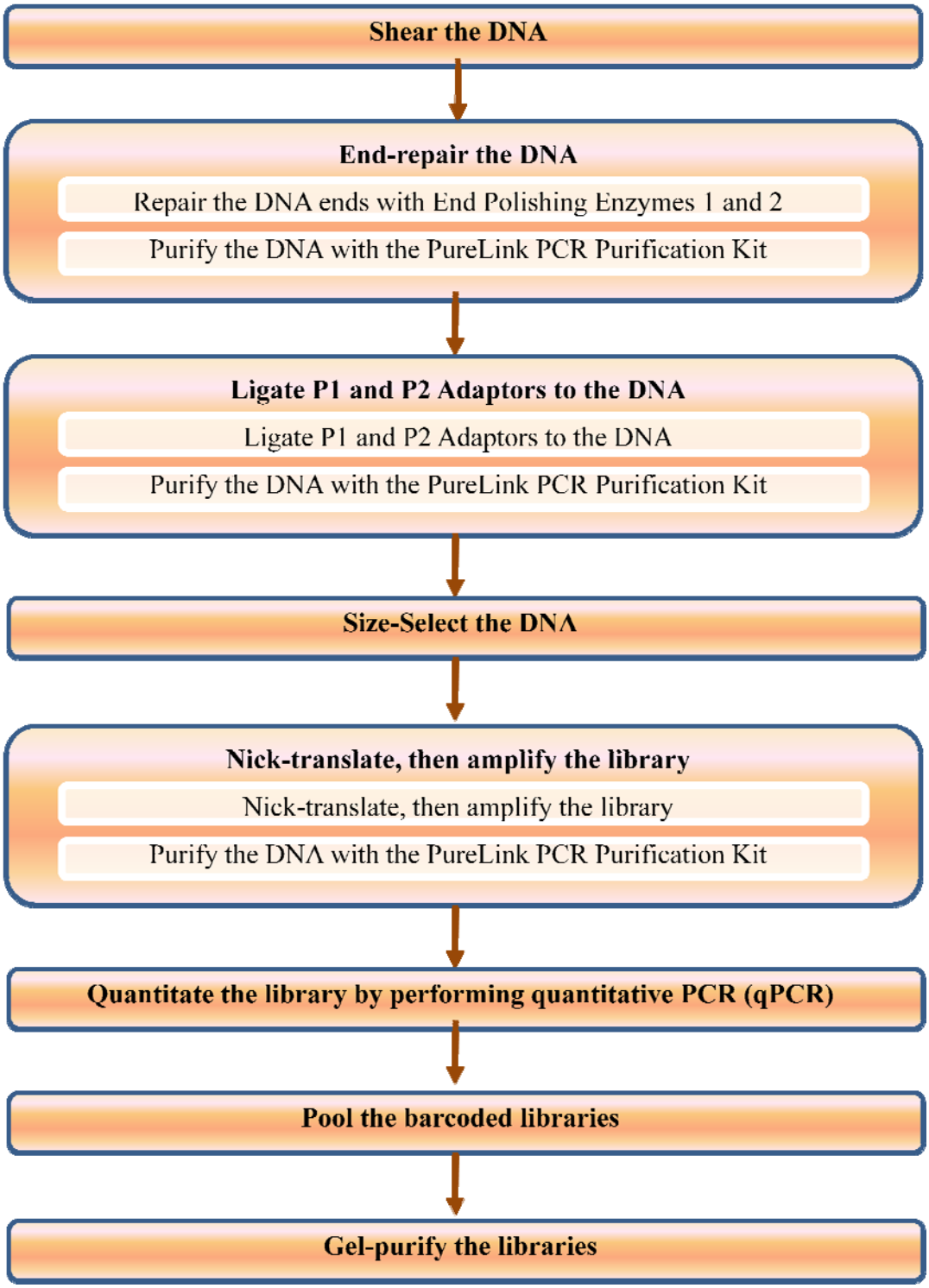
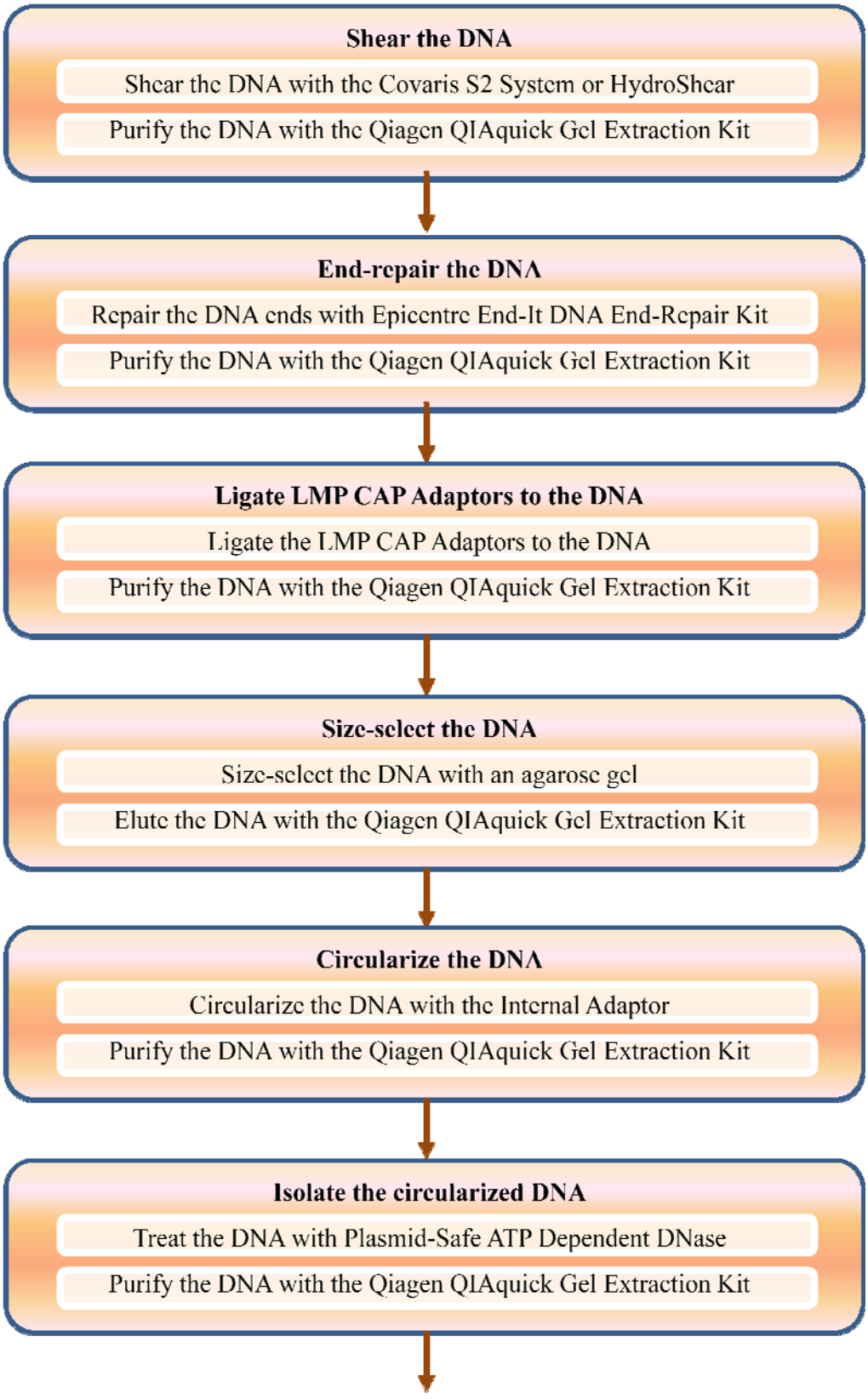
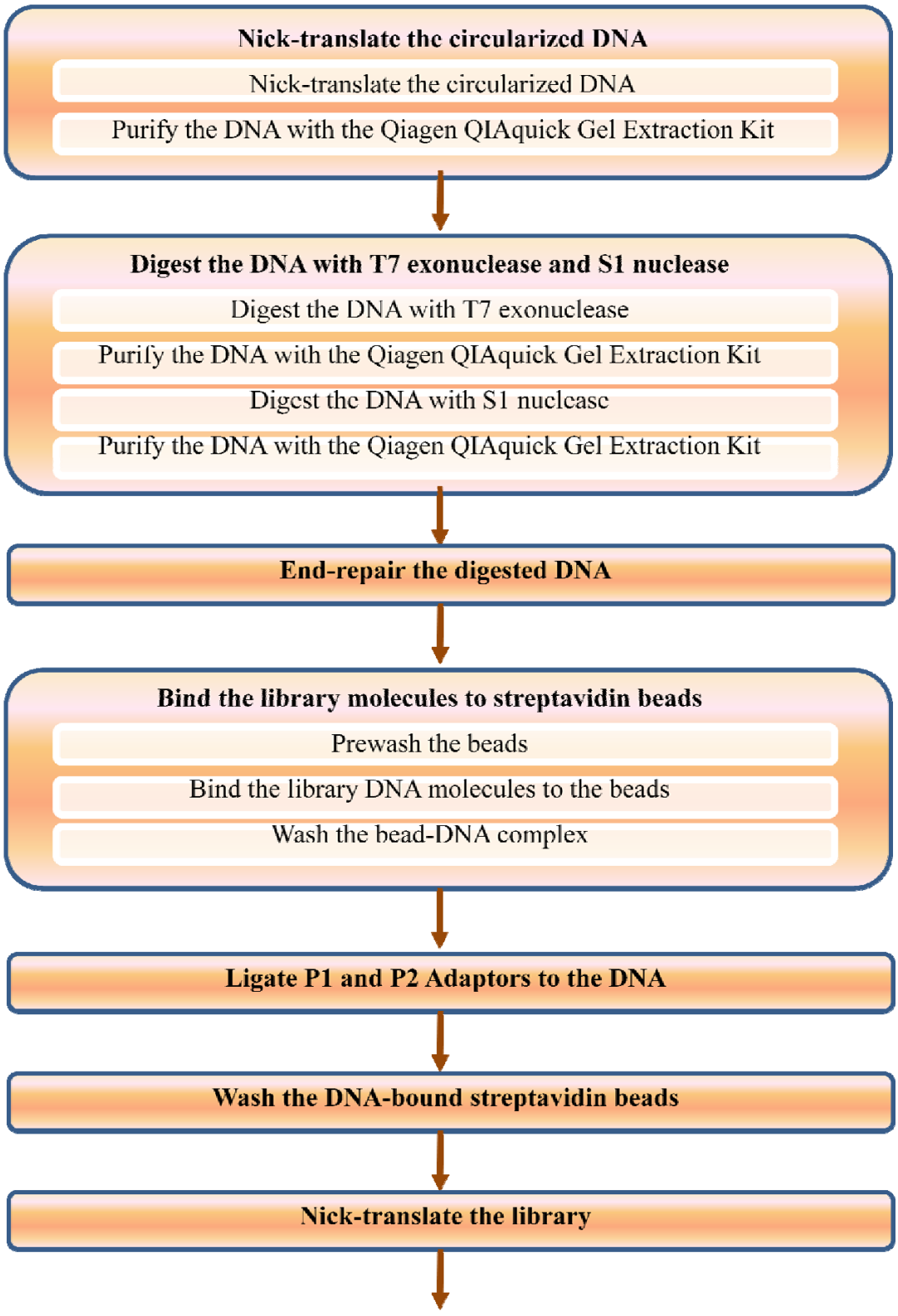


Figure D.1: Barcoded Fragment Library Preparation – This is the workflow followed to prepare the bacteriophage libraries.

Mate-Pair Library Preparation:




Nick-translate the circularized DNA
Nick-translate the circularized DNA
Purify the DNA with the Qiagen QIAquick Gel Extraction Kit
Digest the DNA with T7 exonuclease and S1 nuclease
Digest the DNA with T7 exonuclease
Purify the DNA with the Qiagen QIAquick Gel Extraction Kit
Digest the DNA with S1 nuclease
Purify the DNA with the Qiagen QIAquick Gel Extraction Kit
End-repair the digested DNA
Bind the library molecules to streptavidin beads
Prewash the beads
Bind the library DNA molecules to the beads
Wash the bead-DNA complex
Ligate P1 and P2 Adaptors to the DNA
Wash the DNA-bound streptavidin beads
Nick-translate the library

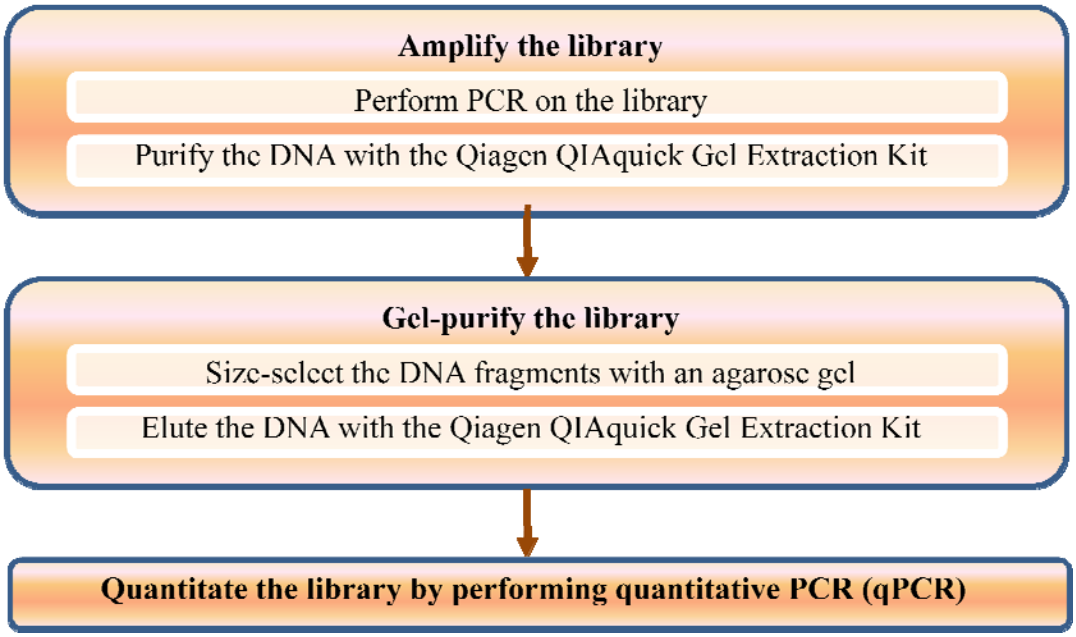


Figure D.2: Mate-pair Library Preparation – This is the workflow followed to prepare the Desulfoluna spongifila, strain AA1 library.

Curriculum Vitae

Ariella Syma Sasson

January 2001 Bachelor of Arts from Rutgers College, Rutgers The State University of New Jersey, New Brunswick, NJ

Majors: Mathematics, Computer Science, and Biomathematics

January 2001 – May 2001

Software Developer, Connotate Technologies, New Brunswick, NJ

December 2001 - August 2006

Actuarial Analyst, Chubb Group of Insurance Companies, Warren, NJ

October 2010

Ph.D. Computational Biology and Molecular Biophysics

Publications:

W. Michael Brown, Ariella Sasson, Donald R. Bellew, Lucy A. Hunsaker, Shawn Martin, Andrei Leitao, Lorraine M. Deck, David L. Vander Jagt, Tudor I. Oprea. "Efficient Calculation of Molecular Properties from Simulation Using Kernel Molecular Dynamics." Journal of Chemical Information and Modeling 2008 48 (8), 1626-1637.

Ariella Sasson and Todd P. Michael. “Filtering error from SOLiD Output.” Bioinofrmatics 2010 26 (6), 849-850.