A METHOD FOR REFERENCE-FREE GENOME ASSEMBLY QUALITY ASSESSMENT

by

JOSHUA BURKHART

A THESIS

Presented to the Department of Computer and Information Science
and the Graduate School of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Master of Science

June 2013

THESIS APPROVAL PAGE

Student: Joshua Burkhart

Title: A Method for Reference-Free Genome Assembly Quality Assessment

This thesis has been accepted and approved in partial fulfillment of the requirements for the Master of Science degree in the Department of Computer and Information Science by:

John S. Conery	Chairperson

and

Kimberly Andrews Espy	Vice President for Research and Innovation; Dean of the Graduate School

Original approval signatures are on file with the University of Oregon Graduate School.

Degree awarded June 2013

THESIS ABSTRACT

Joshua Burkhart

Master of Science

Department of Computer and Information Science

June 2013

Title: A Method for Reference-Free Genome Assembly Quality Assessment

How to assess the quality of a genome assembly without the help of a reference sequence is an open question. Only a few techniques are currently used in the literature and each has obvious bias. An additional method, using restriction enzyme associated DNA (RAD) marker alignment, is proposed here. With high enough density, this method should be able to assess the quality of *de novo* assemblies without the biases of current methods.

With the growing ambition to sequence new genomes and the accelerating ability to do so cost effectively, methods to assess the quality of reference-free genome assemblies will become increasingly important. In addition to the existing methods of known sequence alignment, RAD marker alignment may contribute to this effort.

CURRICULUM VITAE

NAME OF AUTHOR: Joshua Burkhart

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon, Eugene
Grand Valley State University, Allendale, Michigan

DEGREES AWARDED:

Master of Science, Computer and Information Science, 2013
University of Oregon

Bachelor of Science, Computer Science, 2010
Grand Valley State University, Michigan

AREAS OF SPECIAL INTEREST:

Scientific Computing
Data Mining
Information Theory
Bioinformatics

PROFESSIONAL EXPERIENCE:

Graduate Teaching Fellow, University of Oregon, March 2012 - June 2013

TechStudent-PostGraduate, Raytheon, June 2012 - September 2012

Software Engineer, Blue Medora, January 2011 - December 2011

GRANTS, AWARDS, AND HONORS:

Grand Valley State University Upsilon Pi Epsilon Hon. Soc., Vice President, 2010

Grand Valley State University, Student Senator, 2010

ACKNOWLEDGMENTS

I wish to express sincere appreciation to Dr. John Conery, Dr. William Bradshaw, Dr. Christina Holzapfel, Rudy Borowczak, and Alida Gerritsen for their assistance in the preparation of this manuscript. In addition, special thanks are due to Dr. Matt Streisfeld and Dr. Jay Sobel for first introducing me to to the science and wonder of evolution and genomic research.

To my parents, Dr. Gary Burkhart and Marilyn Burkhart, who have inspired me to think critically about my intuitions. And to my two dear friends, Alex Boorsma and Kimberly Baker, who have provided emotional support during my academic pursuits.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

BACKGROUND

Whole genome sequencing enumerates the nucleotides found in the chromosomal DNA of an organism and provides a most intimate view of an individual. Today, genome sequencing contributes to our understanding of biology in many ways including medical diagnoses, gene network discovery, and evolutionary adaptation.

Sequencing even a single organism can assist in medical diagnoses and understanding gene networks by both identifying genetic markers known to be linked with certain conditions and identifying expressed and unexpressed regions of DNA along with the areas that may contribute to those regions' promotion or repression.

Comparative genomics focuses on how different genome sequences relate to each other and necessarily involves sequencing more than one organism. It can be used to find inter-species and intra-species differences. Common differences among genomes include inversions, repeats, deletions, and transpositions. It can be especially interesting if genotypic differences can be well correlated with phenotypic variation.

The brief history of genome sequencing began when Watson, J. and Crick, F. published their seminal paper[1] on the structure of DNA in 1953. Early advancements in the field include Maxam-Gilbert sequencing[2] and Sanger Sequencing[3], both published in 1977. They provided methods by which DNA could be sequenced with high accuracy, though at a

cost that was prohibitive for large projects. A revolutionary technique termed *random* sequencing or *shotgun* sequencing was used as early as 1995[4] to sequence the *Haemophilus influenzae Rd.* genome and was later used to sequence the human genome[5]. Shotgun sequencing was later parallelized[6] allowing for large amounts of DNA to be sequenced at low cost. This parallelization of DNA sequencing is sometimes called *high throughput* sequencing or *next-generation* sequencing (NGS). A further extension of this technique allowed for multiple libraries to be sequenced simultaneously using unique identifying sequences or *barcodes*[37]. The state of the art is exemplified by industry leaders like Life Technologies SOLiD Next-Generation Sequencing, 454 Sequencing GS Systems, and Illumina HiSeq Systems.

NGS has many sequencing applications. One of these is genome sequencing. At a high level of abstraction, a typical NGS genome sequencing project is outlined below.

1. Cells from organism(s) of interest are collected.
2. DNA is separated from other cellular components.
3. DNA is amplified using a cloning process like the polymerase chain reaction[39] (PCR)
4. DNA is fragmented or *sheared* randomly either by chemical or mechanical processes
5. DNA fragments of a specific size are collected, forming a *library*
6. The library is loaded into a sequencing machine

7. Each DNA fragment in the library has up to 100 base pairs of one or both of its ends sequenced, resulting in *reads*
8. The sequencing machine produces files containing reads to be used for further analysis

Because the NGS process yields only short disconnected sequences, several additional stages must be completed before meaningful insights can be made.

Firstly, by removing or *filtering* some of the reads produced by the sequencing process, one may be able to detect and address several issues that may hinder the remainder of the genome sequencing process. Several features should be filtered out from a set of reads prior to further analysis.

1. Sequencing machines are prone to misidentifying base pairs due to one or more of several phenomena, described well by Ledergerber and Dessimoz[7], resulting in what is termed a *miscall* or *mismatch*.

2. Due to the nature of the algorithms used in the later stage of genome assembly, it is difficult to determine the length and correct placement of highly repetitive sequences like *AAAAAAAA...* or *AGTAGTAGTAGT...* These sequences should be discarded.

3. Sequence contamination is the inclusion of 'unlikely' sequence in a genome. It has been reported that several published genomes have been found to contain sequence data probably describing another organism in the same experimental environment, such as that of human laboratory equipment operators[8].

Modern DNA sequencers record both the base pair and a quality score associated with each base pair and report it using the FASTQ file format[9]. Quality scores give a measure of assurance of each call and are typically generated using Phred[10].

Using filtering programs that average base pair quality scores along each read, portions of reads can be discarded or *trimmed* once predefined confidence thresholds are reached[11].

A k-mer is an ordered subset of necessarily adjacent base pairs with a length of some natural number, k, base pairs found to be in some larger sequence. By enumerating all the k-mers of length k, the amount of unique information in a genome can be measured using the distribution of resultant k-mers. By varying k, it is as if viewing a sequence through different lenses, each providing a slightly altered picture of the information contained therein (See APPENDIX: FIGURES: Fig. 1).

A k-mer *frequency* is the number of times a specific k-mer is seen in a larger sequence (See APPENDIX: FIGURES: Fig. 2).

Filters can use k-mer distributions to keep only those k-mers with maximum or minimum frequencies. Highly frequent k-mers may indicate repetitive sequence, contamination, or other over-represented DNA due to errors during the cloning process and can be discarded. Additionally, rare k-mers may indicate sequencer error and can often be discarded.

Typically, k-mer frequency is plotted against the number of k-mers with each frequency, or k-mer *count* in graphs (See APPENDIX: FIGURES: Fig. 3, 4, 5). Notice how scale changes affect each view.

Genome assembly is the process by which short sequences are connected or *assembled* into longer contiguous sequences termed *contigs*. There are several algorithms that perform this task but most genome assembly software today uses algorithms based on one of either de Bruijn graphs or string graphs.

A popular algorithm used for genome assembly relies on de Bruijn graphs which consider (k-1)-mers as vertices and k-mers as edges in a directed graph and search for an Eulerian path in order to reconstruct a source sequence (See APPENDIX: FIGURES: Fig. 12). For a more complete description, see [12].

Alternatively, algorithms using string graphs consider reads as *strings* or *curves* and attempt to find the best intersection between these curves. For a more complete description, see [13].

The goal of genome assembly is to accurately reproduce the underlying contiguous sequences present in a genome. Theoretically, each contig should represent a chromosome. When sequencing an organism for the first time, a reference genome is not available. This is called reference-free or *de novo* genome sequencing. The accuracy of such an assembly is difficult to assess. Several methods have been proposed but no single method consistently assures the highest accuracy[14].

An early method used to gauge assembly quality was to quantify connectivity. NG50 is the number of contigs into which the first 50% of the base pairs in the estimated genome assemble when ordered from largest to shortest. N50 is the number of contigs into which the first 50% of the base pairs in contigs assemble when ordered from largest to shortest (See APPENDIX: FIGURES: Fig. 6). N50 Length is the size of the middle contig (See APPENDIX: FIGURES: Fig. 7). Metrics like this can be taken with other percentages too: 75%, 90%, etc. Together, these are termed "NX" statistics.

In addition to NX statistics, the mean contig length and longest contig length are simple ways to determine how well an assembly's contigs are connected, though they do not address how well any of the sequence produced represents the actual genome.

One thought is to compare the quality of genome assemblies produced by different genome assembly software pipelines, select the best one, and trust its output. An advantage to this

method is that testing is theoretically easy. A synthetic genome can be created as was done in Assemblathon 1[15], whose sequence is known. It can then be artificially broken into reads, used as input to the pipelines, and aligned with the contigs produced by the pipelines to test for accuracy. Unfortunately, this method has not been shown to produce a clear winner and the most accurate genome assembly pipeline remains undecided[14, 15, 16].

Another thought is to align some known sequence to a genome assembly and call the assembly "accurate" based on whether or not a high percentage of the known sequence can be matched to corresponding sequence in the assembly.

Known sequence can be obtained using Expressed Sequence Tags[17] (EST's) and aligned to a genome assembly to test for congruence. These are produced by sequencing complementary DNA (cDNA) created using RNA expressed from genic regions. A drawback of EST alignment is representation bias. Genic regions are not evenly distributed around a genome; extragenetic factors introduce positional bias, such as epigenetic gene silencing[38] which forces EST's to cluster around less tightly packed *euchromatic* regions of a chromosome[18]. This can leave more tightly packed *heterochromatic* regions of chromosomes untested, thus whole-genome assembly quality uncertain.

Another method of using known sequence to assess genome assembly quality is to align sequences believed to be conserved in an organism to its genome assembly[14]. This requires both making *a priori* assumptions about the structure of an organism's genome and

suffers from the same uneven distribution bias as EST alignment.

A novel method to assess the quality of genome assemblies is to consider the alignment of RAD markers[19]. Because some restriction enzymes have been shown to digest DNA indiscriminately[40], regardless of gene density or epigenetic structure, resultant RAD markers are theoretically more evenly distributed around a genome than either EST's or known conserved sequence. Additionally, several RAD treatments may be applied to a genome to increase marker density. Each restriction enzyme cut site should result in the creation of two RAD markers, so by aligning restriction enzyme cut sites to a genome assembly and comparing the ratio of cut site alignments to RAD marker alignments, the percent RAD marker alignment can be computed.

$$t/2c = a$$

where:

t = number of RAD markers that align to genome assembly

c = number of restriction-enzyme cut sites that align to genome assembly

a = RAD marker alignment ratio

(A high ratio indicates a genome assembly expected to be of high quality.)

CHAPTER II

PROJECT INTRODUCTION

An example of a project using genome assembly is the ongoing effort to sequence several *Wyeomyia smithii* populations found along the eastern seaboard of North America in order to study the genetic basis for several varying characters including photoperiodism and propensity to feed on blood.

CHAPTER III

PROCEDURE

Following DNA collection from groups of organisms representative of target *W. smithii* populations, reads were obtained from an Illumina HiSeq 2000 in FASTQ format with Illumina descriptors[9].

The FASTQ files underwent an initial assessment so as to estimate storage, memory, processor, and bandwidth usage.

Initially, read filtering was performed using a custom perl script that counted quality scores until a predefined threshold was reached. Several more sophisticated methods of read filtering have been published and this method was ultimately abandoned.

The kmer_filter program, one of the components of the Stacks pipeline[20], filters reads using k-mer distributions with maximum and minimum frequency thresholds optimized using k-mer distribution visual representations (See APPENDIX: FIGURES: Fig. 3, 4, 5).

A project called Quake[21] is a package developed for use especially with Illumina machine output and uses read quality values in addition to known error rates in order to estimate miscalls and correct them when possible.

A program called Diginorm[22] is a package that normalizes k-mer coverage, narrowing

the range of k-mer frequencies, and has the effect of greatly decreasing the amount of data in read files (over 50% reductions were seen during this project) without greatly reducing the amount of information.

The effects of filtering can be quantified in several ways. First, the number of reads retained can be counted and compared to the number of reads prior to filtering. If a high percentage of reads are left, the filter may not have been effective. Second, the resultant k-mer distribution can be plotted and inspected for interesting features[23]. Third, executing a genome assembly using the filtered reads may indicate what filtering works well. In fact, much time was spent "bouncing" between read filtering and genome assembly in an attempt to optimize the results of both processes.

Using the results of several genome assembler competitions[14, 15, 16] as a guide, several attractive assemblers were selected at the outset of this project.

The Broad Institute's ALLPATHS-LG assembler[42] was considered due to its high ratings in Assemblathon 1[15], 2[14], and GAGE[16] but required DNA fragments of varying length and was thus unfit for the available dataset which consisted entirely of (common length) short reads.

SOAPdenovo2[43] had shown good results in Assemblathon 1[15], 2[14], and GAGE[16] but numerous parameters and sparse documentation made it nebulous and a poor candidate as a tool for unfamiliar users.

MSR-CA, an early version of MaSuRCA[44]. It was shown to be a competitive candidate in GAGE[16], but, due to its early stage of development at the outset of this project, only registered users had access to it. Thus it was not available for installation on the ACISS[26] system.

The String Graph Assembler[24] is an assembler that relies on a string graph algorithm based on overlaps. It was used following the procedure described in the source repository[25]. The reported memory usage was low, even for large genomes in comparison to other assemblers and it was a high scorer in Assemblathon 1[15], 2[14], and GAGE[16]. Unfortunately, the longest available execution queue available on the ACISS cluster computer is 336 hours[27]. SGA was unable to complete an assembly in this amount of time. Additionally, SGA was found to be poorly documented and require unrealistic execution times for large genomes.

Velvet[28] was a mediocre scorer in Assemblathon 1[15] and GAGE[16] but was well documented, could finish executions within the time limits of ACISS queues, and could accept a single length DNA fragment library as input. Velvet is a memory-intensive program and only special "fat" nodes with 384 GB RAM[27] could complete some

genome assemblies. Additionally, not all Velvet executions finished in a reasonable amount of time. Velvet tended to have unpredictable run times and parameter value testing and optimization was necessary to produce assemblies (See APPENDIX: FIGURES: Fig. 13). Initial parameter estimates were made with the help of a ruby script[47] that reported average nucleotide coverage and expected k-mer coverage for several values of k. To further assist in optimizing Velvet parameters, an R script was written[45] that displayed node coverage, as explained in the Velvet manual[46]. In summary, Velvet was a good choice for the ACISS computing environment as the high memory resources required were available and the time required for some other assemblers was not.

Several methods were used to compare genome assemblies. Connectivity statistics were computed using a custom ruby script[29] and RAD marker alignment scores were computed using a package called Radiqual[30]. Review the Radiqual README.md document for a brief description and usage[31] (See APPENDIX: FIGURES: Fig. 8, 9, 10, 11).

CHAPTER IV

FUTURE DIRECTIONS

To fully validate the ability of RAD marker alignment to discern accurate assemblies from inaccurate ones further research is required. Genome assembly alignment to synthetic or finished reference genomes should be used as bases for quality and compared to EST, known conserved region, and RAD marker alignment for those assemblies. A relatively strong correlation between observed accuracy and that predicted by RAD alignment would confirm its usefulness.

The variation of features found in k-mer distributions deserves further investigation. It is interesting that Fig. 5 shows 69-mers with an opposite concavity to that of 51-mers at low frequencies and that the k-mer distributions in Fig. 3 have similar inflection points.

As we seek to better understand the origin and nature of life on this planet, projects conducting *de novo* genome sequencing are becoming more numerous. A few examples are below.

1. The 1000 Genomes Project[32] has a goal of sequencing 1000 human genomes in order to discover low-frequency genetic variation.

2. The Genome 10K[33] is a project whose goal is to assemble 10,000 vertebrate species in order to make discoveries about genetic diversity.

3. The 1000 plant genomes initiative[34] attempts to generate sequence for 1000 plant species.

With the assemblies of multiple organisms, many interesting discoveries can be made. Software like MUMmer[48] can be used to align genomes to one another to search for large-scale differences(See APPENDIX: FIGURES: Fig. 14, 15) and small-scale differences(See APPENDIX: FIGURES: Fig. 16) that could explain the origins of disease, gene network development, or speciation events.

Genome sequencing is a field of active research and technological development. Promising technologies include Illumina's 150-250 base pair insert sequence size library protocol[35], allowing even short reads to be more connective then the current 100 base pair size. Oxford Nanopore Technologies has also developed biosensors that may one day be used to provide even longer reads[36].

With the growing ambition to sequence new genomes and the accelerating ability to do so cost effectively, methods to assess the quality of reference-free genome assemblies will become increasingly important. In addition to the existing methods of EST and conserved sequence alignment, RAD marker alignment may contribute to this effort.

APPENDIX:

FIGURES

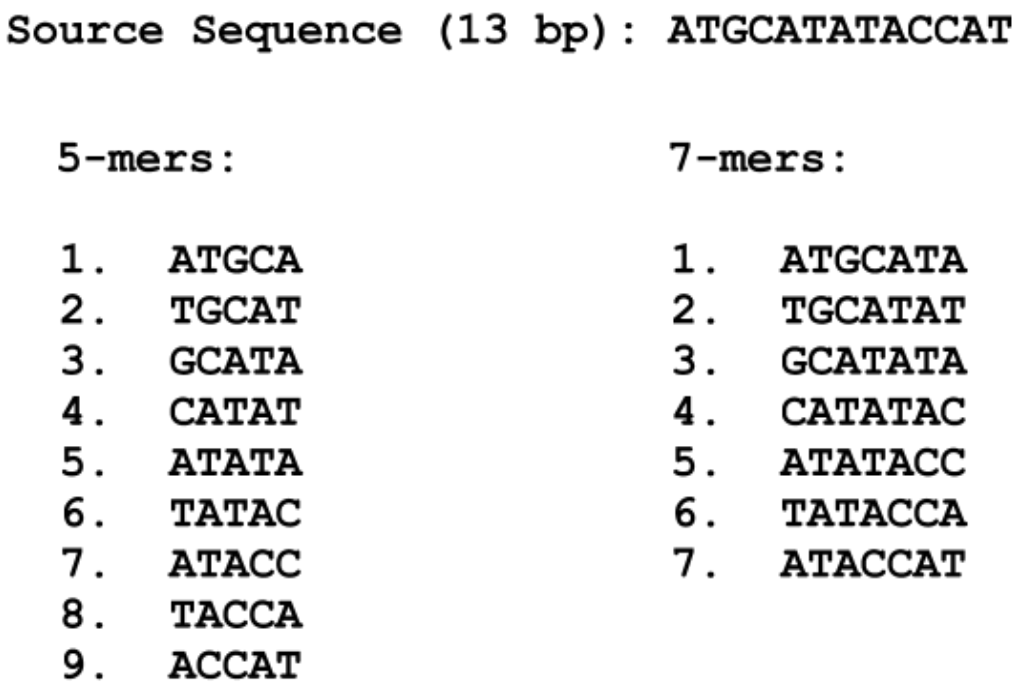

Fig. 1. A sequence with 13 base pairs along with its 5-mers and 7-mers. Each of the 5-mers and 7-mers appears only once in the source sequence.

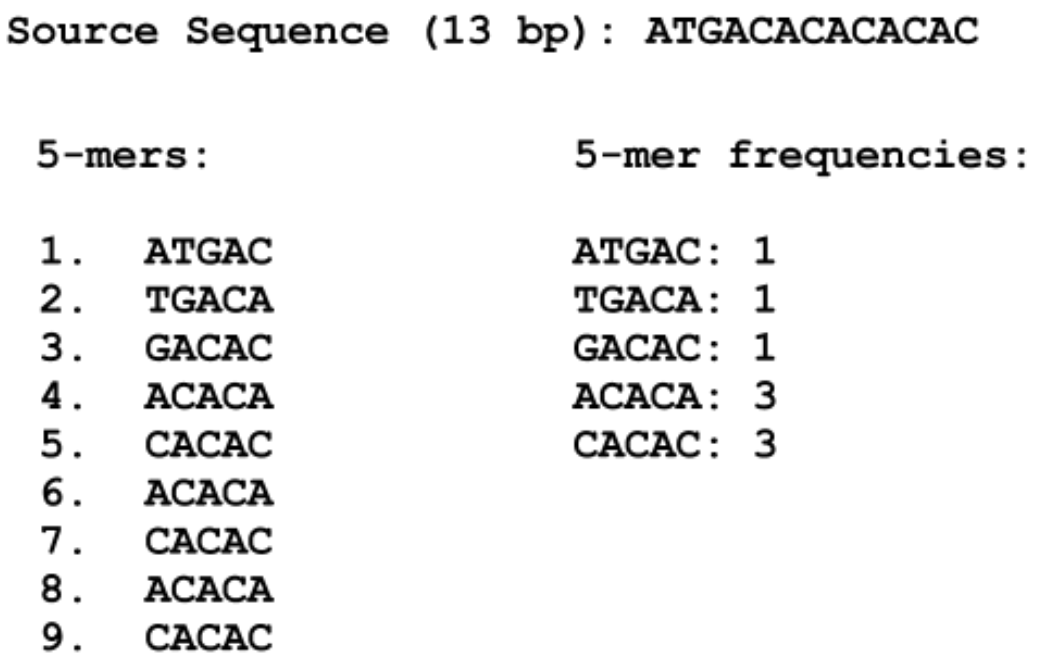

Fig. 2. A sequence of 13 base pairs along with its 5-mers and the frequencies of each 5-mer. Some of the 5-mers appear only once in the source sequence but (4) and (5) each appear three times.

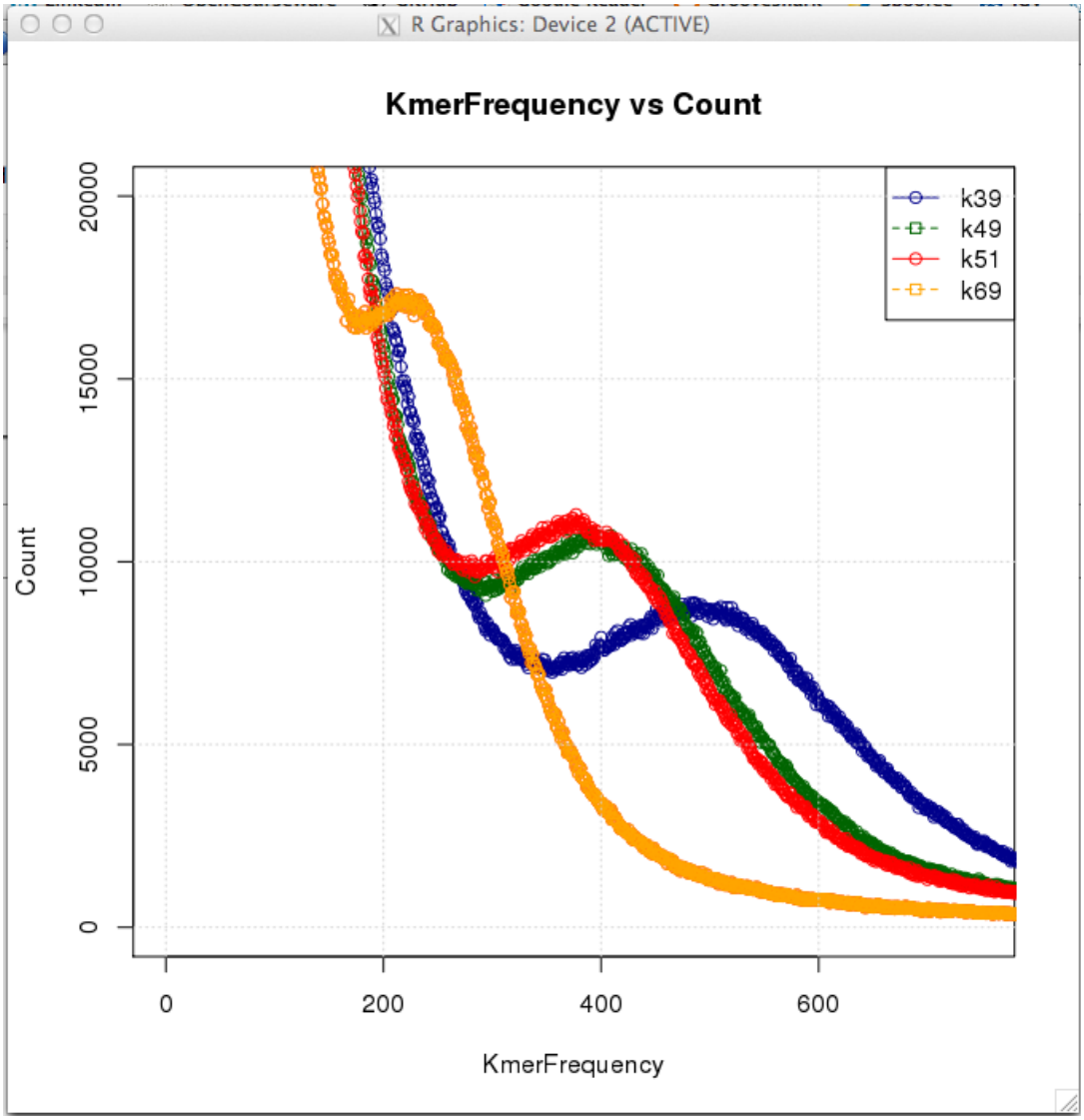

Fig. 3. Plotting several k-mer distributions together may lead to insights about the information contained in a source sequence.

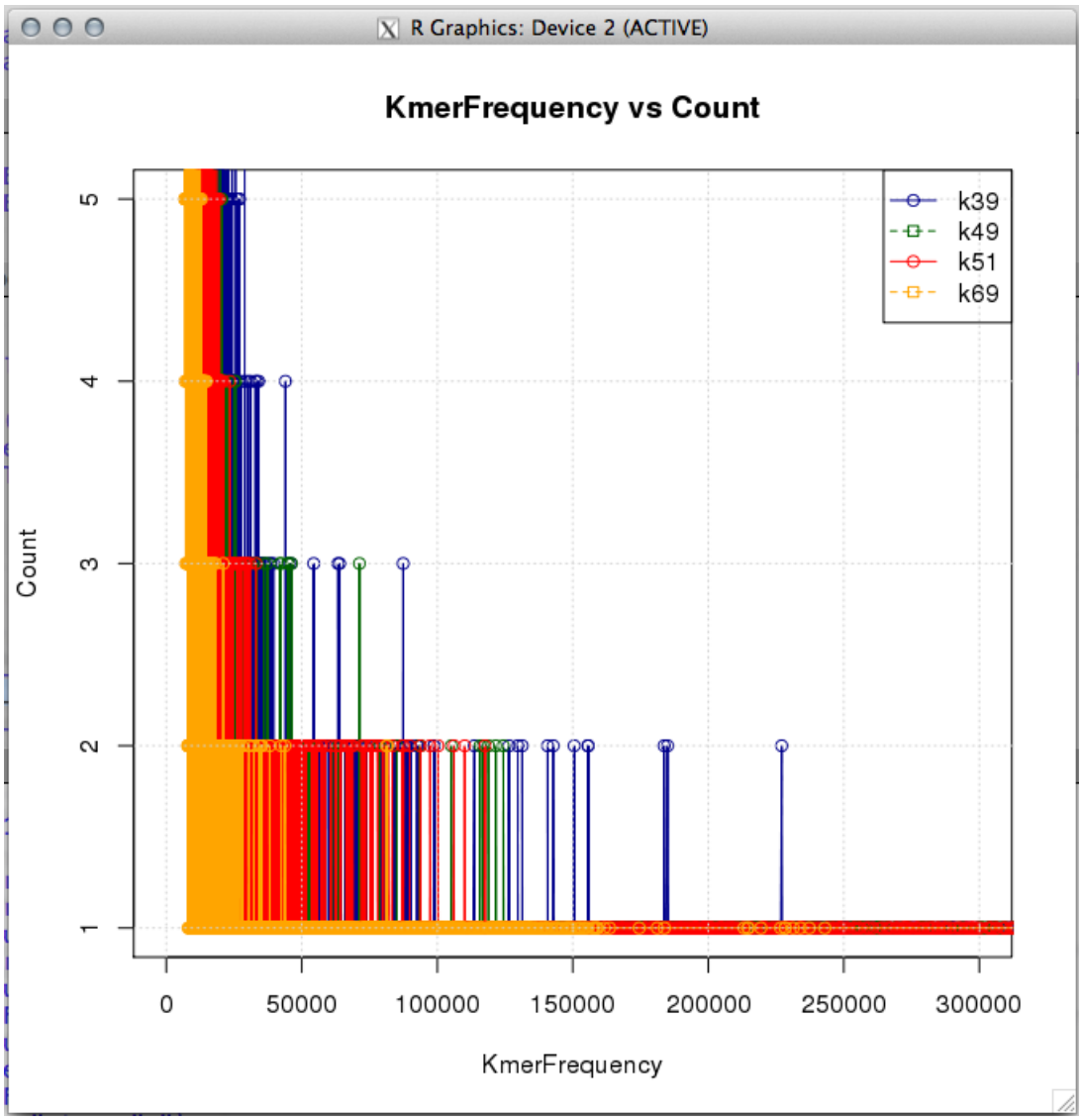

Fig. 4. Many k-mers are found with frequencies below 150,000.

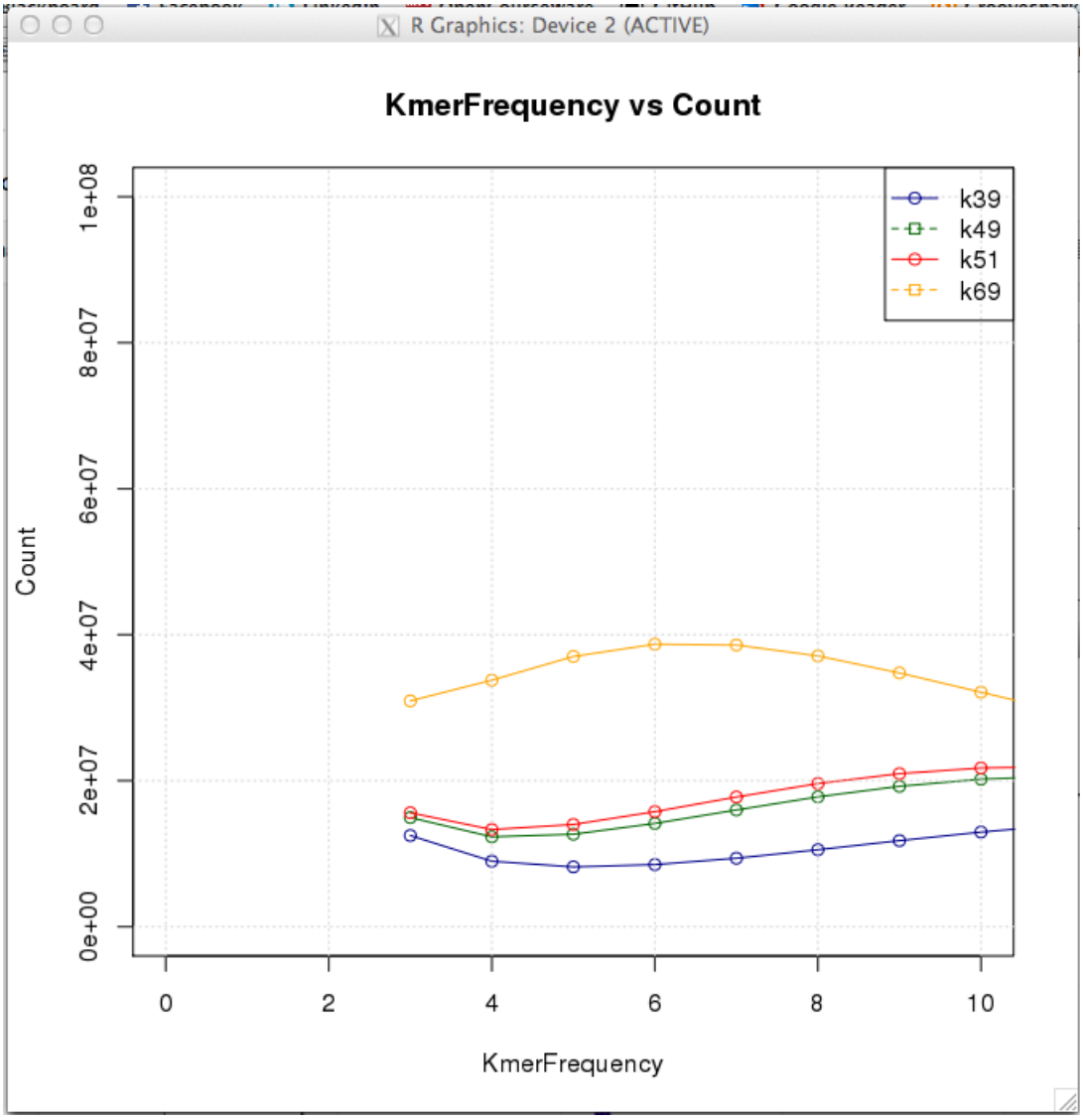

Fig. 5. Every k-mer found in the source sequence appears at least three times. Also, notice the distribution of 69-mers behaves differently from the other k-mer distributions plotted here.

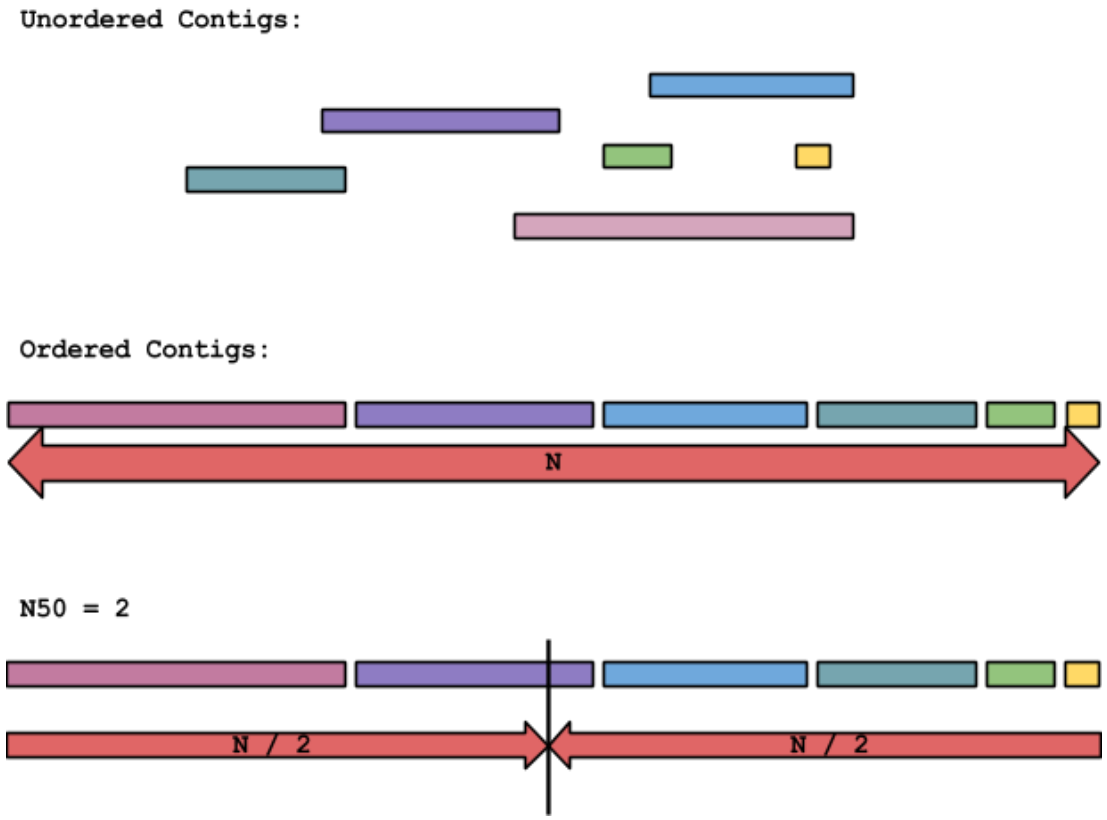

Fig. 6. Color-coded contigs (top) in no particular order, representing the way they are output from a genome assembler. The sum of the lengths of these contigs is taken as N, and the contigs are ordered from longest to shortest (middle). By halving N (bottom), a position along the ordered contigs is selected, indicated here by a black line. The number of contigs encountered as this position is approached, from largest to shortest, is the N50. In this figure we see N50 = 2.

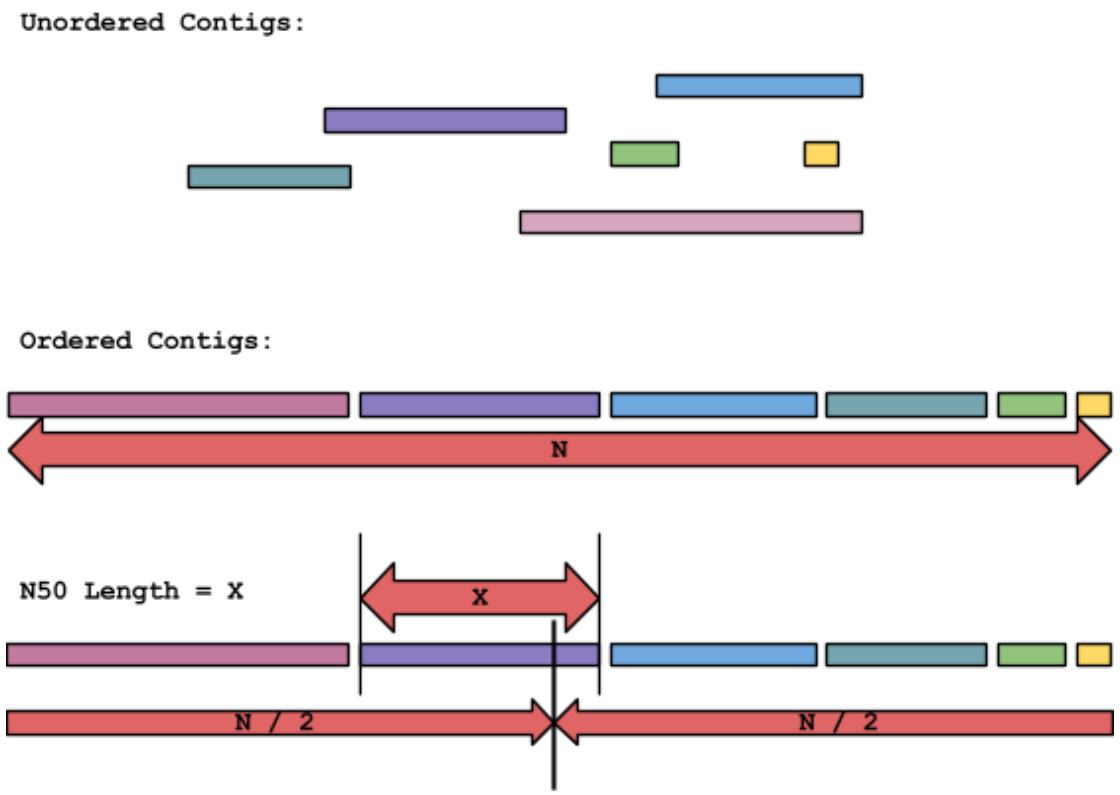

Fig. 7. Similar to Fig. 6, above, an ordered list of contigs is required to calculate N50 Length. Instead of counting the number of contigs encountered as the N/2 position is approached, the length of the contig containing the position is reported as the N50 Length.

Fig. 8. Two RAD markers aligning to the genome assembly around a single cut site.

Fig. 9. Two cut sites near each other and two RAD markers aligning to the genome assembly.

Fig. 10. One RAD marker aligning to the genome assembly around a single cut site.

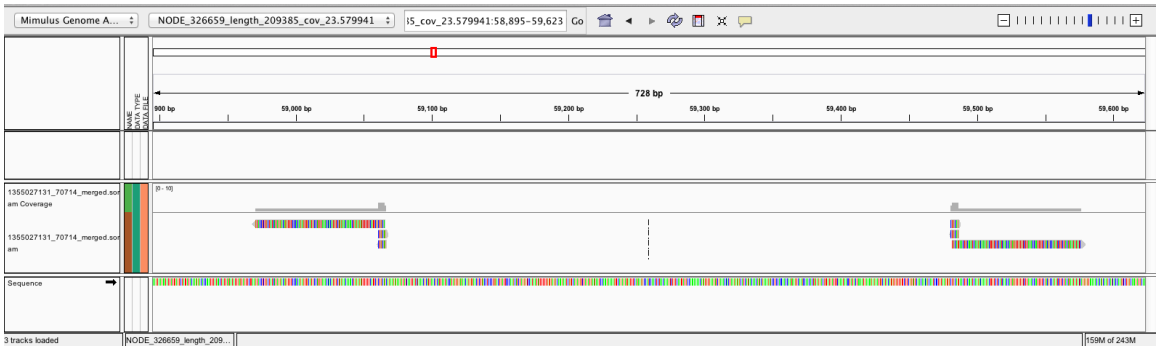

Fig. 11. Two cut sites somewhat separated and two RAD markers aligning to the genome assembly around those cut sites.

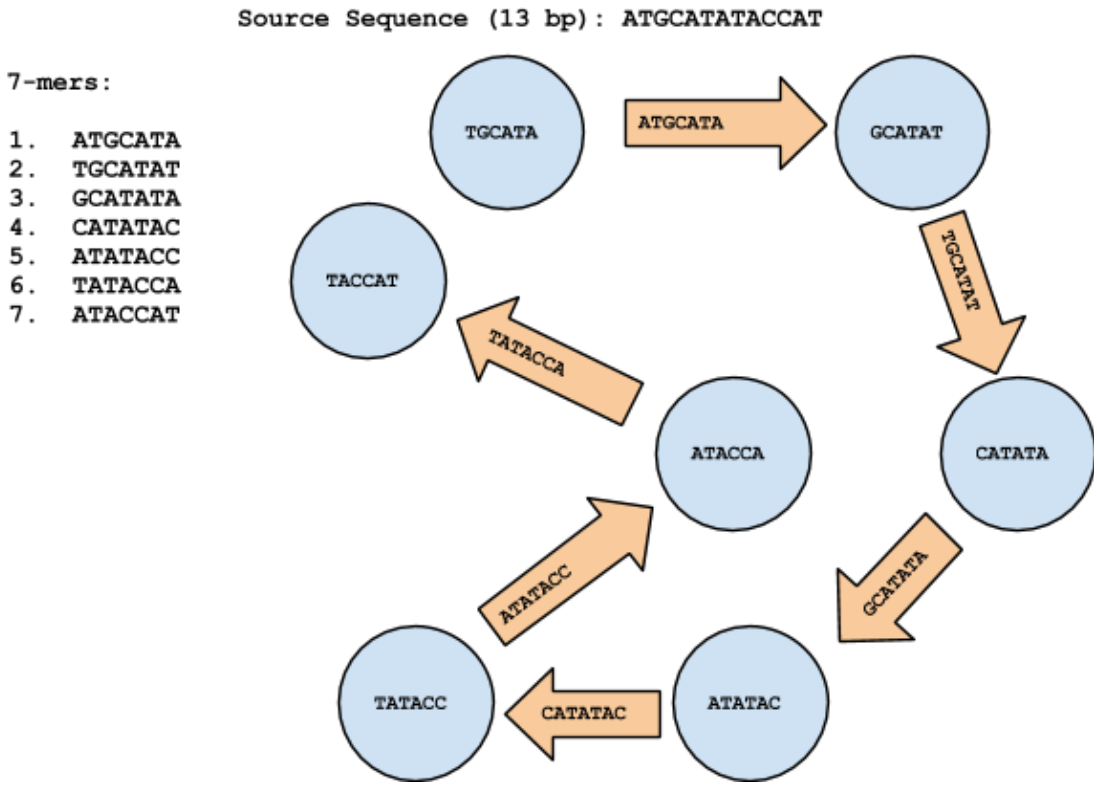

Fig. 12. By constructing a digraph using the (k-1)-mers as nodes, indicated here as blue circles, and the k-mers as directed edges, indicated here as orange arrows, it is possible to reconstruct the source sequence by finding an Eulerian path. Circular source sequences, such as bacterial chromosomes, are reconstructed using Eulerian cycles instead.

	m	max_k_freq	min_k_freq	k	insert_length	min_config_length	expected_cov	actual_cov	NS0_for_configs_>500	Max_config_length	Num_configs_>500	Total_bases_in_configs_>500	radIqual_#cut	radIqual_#reds	radIqual_actAvgs
		250,000	0	39	500	100	50.00								
		250,000	1	39	500	100	50.00								
		250,000	2	39	500	100	50.00								
		250,000	4	39	500	100	50.00								
		250,000	100	39	500	100	50.00								
	-1 max_0 min k39	250,000	400	39	500	100	50.00								
		2,480,000	0	39	500	100	50.00								
		2,480,000	1	39	500	100	50.00								
		2,480,000	2	39	500	100	50.00								
		2,480,000	4	39	500	100	50.00								
		2,480,000	100	39	500	100	50.00								
		2,480,000	400	39	500	100	50.00								
		150,000	0	49	500	100	42.00								
		150,000	1	49	500	100	42.00								
		150,000	2	49	500	100	42.00								
		150,000	4	49	500	100	42.00								
		150,000	100	49	500	100	42.00								
		150,000	400	49	500	100	42.00								
	501K max_2 min k49	150,000	400	49	500	100	42.00								
	1187K max_0 min k49	1,187,000	0	49	500	100	42.00	<won't finish>	23,730	184,967	25,575	201,949,337			
	1187K max_1 min k49	1,187,000	1	49	500	100	42.00	41.00	27,342	182,867	23,406	202,406,766			
	1187K max_2 min k49	1,187,000	2	49	500	100	42.00	41.00	27,969	223,082	23,186	202,664,710			
	1187K max_4 min k49	1,187,000	4	49	500	100	42.00	42.00	24,504	195,199	24,744	202,300,041			
		1,187,000	100	49	500	100	42.00								
		1,187,000	400	49	500	100	42.00								
	-1 max -1 min k51	-1	-1	51	500	100	40.00	<won't finish>	<won't finish>	<won't finish>					
		150,000	0	51	500	100	40.00								
		150,000	1	51	500	100	40.00								
		150,000	2	51	500	100	40.00								
		150,000	4	51	500	100	40.00								
		150,000	100	51	500	100	40.00								
		150,000	400	51	500	100	40.00								
	200K max k51	200,000	0	51	500	100	40.00	38.00	2,570	38,254					
	716K max_0 min k51	716,000	0	51	500	100	40.00	35.00	13,700	106,977	41,104	208,171,553	83418	25115	0.150537754352077
	716K max_1 min k51	716,000	1	51	500	100	40.00	35.00	15,698	111,780	38,077	208,515,604	83282	25723	0.151437081865824
	716K max_2 min k51	716,000	2	51	500	100	40.00	36.00	15,604	123,536	36,101	208,605,663	83012	25236	0.151834898859977
	716K max_4 min k51	716,000	4	51	500	100	40.00	35.00	14,343	108,025	39,499	208,445,691	83078	25224	0.151809143214981
		716,000	100	51	500	100	40.00								
		716,000	400	51	500	100	40.00								
	295 max_1 min k67	295,000	1	67	500	100	28.00	24.00	24,560	187,899	27,300	223,365,300	86878	25932	0.149243787121711
	375 max_1 min k67	375,000	1	67	500	100	28.00	24.00	24,802	206,497	27,522	223,549,523	86984	25911	0.148975438112322
	150K max_0 min k69	150,000	0	69	500	100	25.00						86834	25977	0.149419445676856
	150K max_1 min k69	150,000	1	69	500	100	25.00	22.00	31,193	209,453	23,129	223,634,860	86822	25985	0.149449251203861
	150K max_2 min k69	150,000	2	69	500	100	25.00	22.00	21,127	148,989	29,384	223,579,081	87542	25979	0.148380720677251
	150K max_4 min k69	150,000	4	69	500	100	25.00	22.00	19,123	159,278	30,895	223,137,451	87564	25963	0.148251594366991
		150,000	100	69	500	100	25.00								
		150,000	400	69	500	100	25.00								
	2954 max_1 min k69	295,000	1	69	500	100	25.00	22.00	22,111	165,547	29,481	224,388,438	87706	26006	0.148256875712033
	335K max_0 min k69	335,000	0	69	500	100	25.00	22.00	20,761	162,333	30,786	224,747,512	87806	25946	0.147746787887854
	335K max_1 min k69	335,000	1	69	500	100	25.00	24.00	29,407	228,737	24,642	224,721,502	86878	25921	0.149098447290630
	335K max_2 min k69	335,000	2	69	500	100	25.00	23.00	25,349	192,720	26,665	224,504,065	87168	25990	0.149079937591777
	335K max_4 min k69	335,000	4	69	500	100	25.00	22.00	18,924	127,177	31,679	223,982,961	87468	25981	0.149412764448883
		335,000	100	69	500	100	25.00								
		335,000	400	69	500	100	25.00								
	375 max_1 min k69	375,000	1	69	500	100	25.00	21.00	16,111	118,742	36,329	224,477,891	86726	25992	0.147393619370324
	200K max_0 min k71	200,000	0	71	500	100	24.00	19.00	2,116	30,913	114,853	192,453,543	86184	24817	0.143976940204877
	295 max_1 min k71	295,000	1	71	500	100	24.00	21.00	23,831	169,806	27,867	225,394,184	87510	25903	0.148514456490980
	375 max_1 min k71	375,000	1	71	500	100	24.00	21.00	23,906	160,541	27,986	225,625,200	87418	25899	0.148647975723853
	400K max_1 min k71	400,000	1	71	500	100	24.00	20.00	18,056	144,196	33,520	226,673,420	88192	25967	0.147298550317795

Fig. 13. Sample execution records are shown here. Notice the wide variation in results given slight changes in parameters like *min_k_freq* and *expected_cov*.

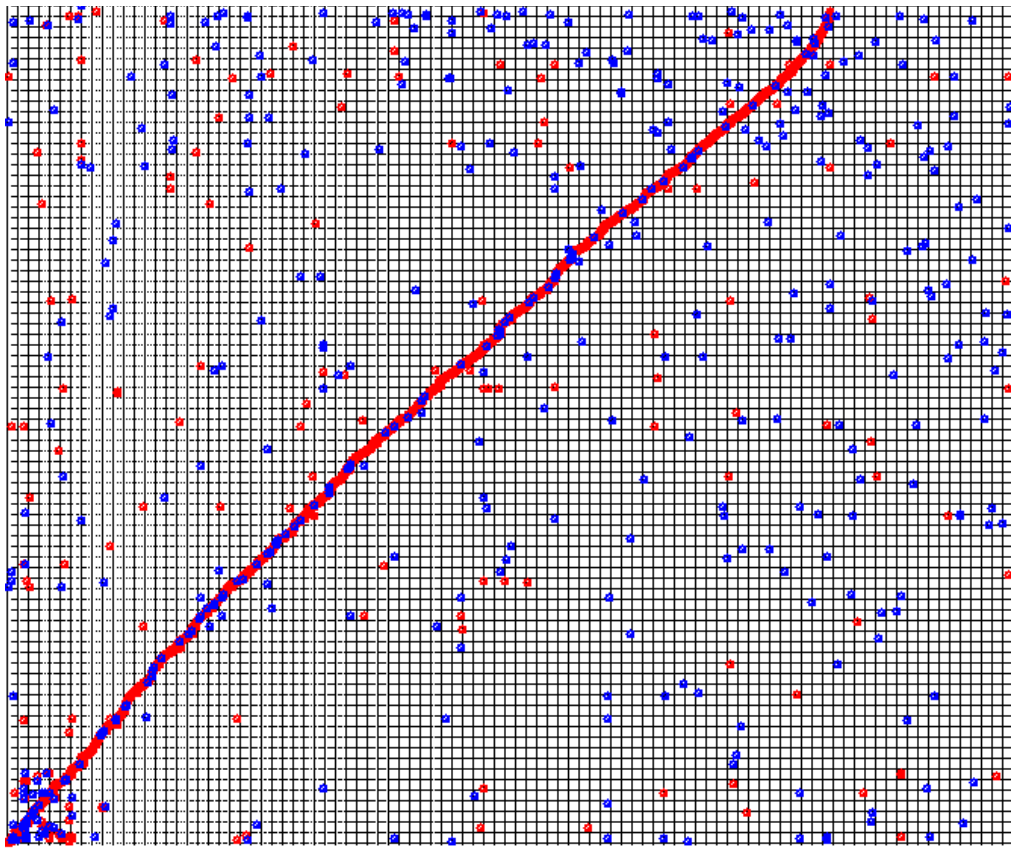

Fig. 14. This figure shows how two genome assemblies align to each other. Notice much of the sequence aligns well. Forward alignments are in red, reverse alignments are in blue.

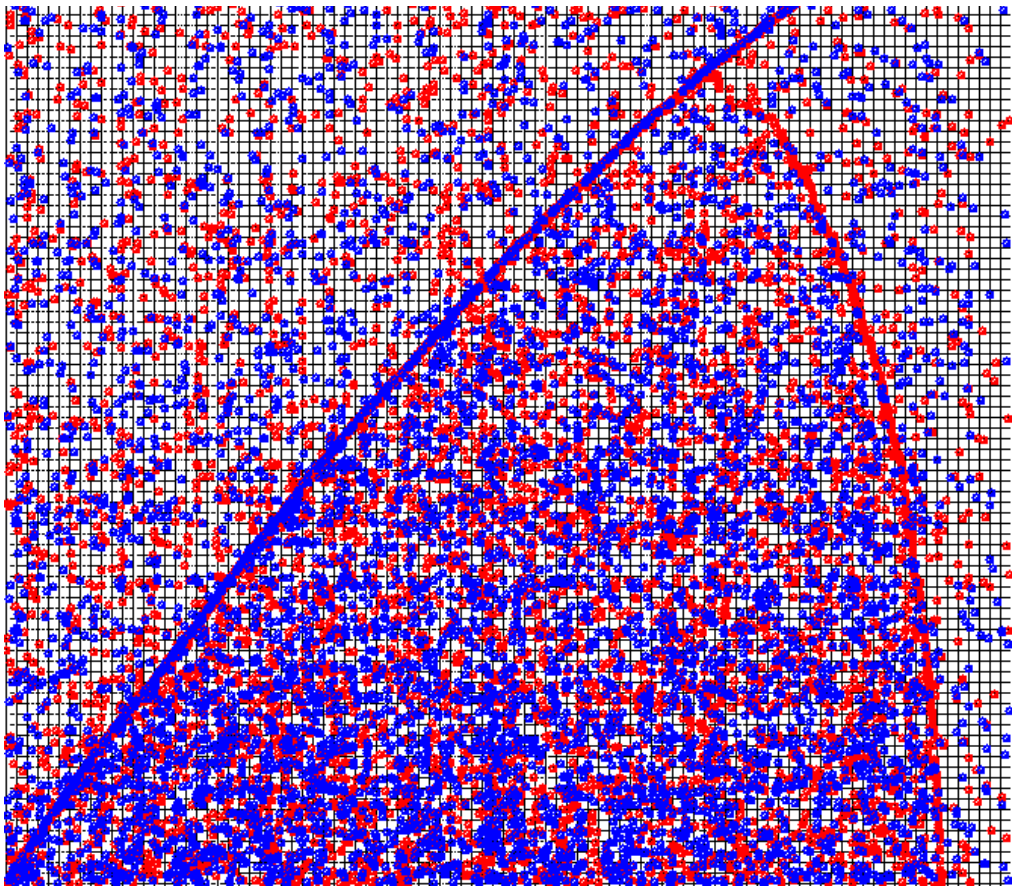

Fig. 15. This figure shows how two genomes assemblies' predicted protein products align to each other. Notice the interesting difference in concavity when comparing the forward alignments, in blue, and the reverse alignments, in red.

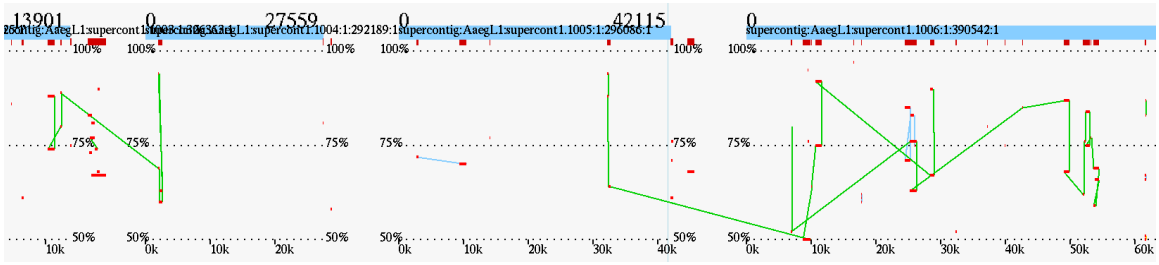

Fig. 16. This figure shows contigs from a reference genome assembly, in blue, at top, and contigs from a query genome assembly, in green, at bottom. Notice the complex alignments some contigs from the query have with those from the reference.

REFERENCES CITED

[1] Watson, J. D., Crick, F. H. C., A Structure for Deoxyribose Nucleic Acid, Nature, Vol. 171, No. 4356, pg. 737-738, 1953.

[2] Maxam, A. M., Gilbert, W., A New Method for Sequencing DNA, PNAS, Vol. 74, No. 2, pg. 560-564, 1977.

[3] Sanger, F., Nicklen, S., Coulson, A. R., DNA Sequencing with Chain-Terminating Inhibitors, PNAS, Vol. 74, No. 12, pg. 5463-5467, 1977.

[4] Flieshmann, R. D., et. al., Whole-Genome Random Sequencing and Assembly of Haemophilus influenzae Rd., Science., Vol. 269, No. 5223, pg. 496-512, 1995.

[5] Venter, J. C., et. al., The Sequence of the Human Genome, Science, Vol. 291, No. 5507, pg. 1304-1351, 2001.

[6] Brenner, S., et. al., Gene Expression Analysis by Massively Parallel Signature Sequencing (MPSS) on Microbead Arrays, Nature Biotechnology, Vol. 18, pg. 630-634, 2000.

[7] Ledergerber, C., Dessimoz, C., Base-Calling for Next-Generation Sequencing Platforms, Briefings in Bioinformatics, Vol. 12, No. 5, pg. 489-497, 2011.

[8] Longo, M. S., Abundant Human DNA Contamination Identified in Non-Primate Genome Databases, PLoS ONE, Vol. 6, No. 2, 2011.

[9] Cock, P. J. A., et. al., The Sanger FASTQ File Format for Sequences with Quality Scores, and the Solexa/Illumina FASTQ Variants, Nucleic Acids Research, Vol. 38, No. 6, pg. 1767-1771, 2009.

[10] Ewing, B., et. al., Base-Calling of Automated Sequencer Traces Using Phred. I. Accuracy Assessment, Genome Research, Vol. 8, No. 3, pg. 175-185, 1998.

[11] Chou, H., Holmes, M. H., DNA Sequence Quality Trimming and Vector Removal, Bioinformatics, Vol. 17, No. 12, pg. 1093-1104, 2001.

[12] Compeau, P. E. C., et. al., How to Apply de Bruijn Graphs to Genome Assembly, Nature Biotechnology, Vol. 29, pg. 987-991, 2011.

[13] Myers, E. W., The Fragment Assembly String Graph, Bioinformatics, Vol 21, Sup. No. 2, pg. ii79-ii85, 2005.

[14] Bradnam, K. R., et. al., Assemblathon 2: Evaluating de novo Methods of GEnome Assembly in Three Vertebrate Species, pre-print available at http://arxiv.org/pdf/1301.5406v1.

[15] Earl, D. A., et. al., Assemblathon 1: A Competitive Assessment of de novo Short Read Assembly Methods, Genome Research, published online, 2011.

[16] Salzberg, S. L., GAGE, A Critical Evaluation of Genome Assemblies and Assembly Algorithms, Genome Research, published online, 2011.

[17] Parkinson, J., Blaxter, M., Expressed Sequence Tags: An Overview, Methods Mol Biol, Vol. 533, pg. 1-12, 2009.

[18] Stralfors, A., Ekwall, K., Heterochromatin and Euchromatin--Organization, Boundaries, and Gene Regulation, Encyclopedia of Molecular Cell Biology and Molecular Medicine, Wiley-VCH Verlag GmbH & Co., 2011.

[19] Miller, M. R., et. al., Rapid and Cost-Effective Polymorphism Identification and Genotyping using Restriction Site Associated DNA (RAD) Markers, Genome Research, Vol. 17, pg. 240-248, 2007.

[20] Catchen, J., et. al., Stacks: An Analysis Tool Set for Population Genomics, Molecular Ecology, Vol. 22, No. 11, pg. 3124-3140, 2013.

[21] Kelly, D. R., et. al., Quake: Quality-Aware Detection and Correction of Sequencing Errors, Genome Biology, Vol. 11, R116, 2010.

[22] Brown, T. C., A Reference-Free Algorithm for Computational Normalization of Shotgun Sequencing Data, pre-print available at http://arxiv.org/abs/1203.4802.

[23] Chor, B., et. al., Genomic DNA k-mer Spectra: Models and Modalities, Genome Biology, Vol. 10, R108, 2009.

[24] Simpson, J. T., Durbin, R., Efficient de novo Assembly of Large Genomes using Compressed Data Structures, Genome Research, published online, 2011.

[25] https://github.com/jts/sga/blob/master/src/examples/sga-celegans.sh

[26] MRI-R2: Acquisition of an Applied Computational Instrument for Scientific Synthesis (ACISS), Grant #: OCI-0960354.

[27] https://aciss.uoregon.edu/wiki/Submission_queues

[28] Zerbino, D. R., Birney, E., Velvet: Algorithms for de novo Short Read Assembly using

de Bruijn Graphs, Genome Research, Vol. 18, No. 5, pg. 821-829, 2008.

[29] https://github.com/joshuaburkhart/bio/blob/master/param_ests.rb

[30] https://github.com/joshuaburkhart/RadiQual

[31] https://github.com/joshuaburkhart/RadiQual/blob/master/README.md

[32] http://www.1000genomes.org/

[33] https://genome10k.soe.ucsc.edu/

[34] http://www.onekp.com/

[35] http://investor.illumina.com/phoenix.zhtml?c=121127&p=irol-newsArticle&ID=1252407

[36] http://www.nanoporetech.com/technology/introduction-to-nanopore-sensing/introduction-to-nanopore-sensing

[37] Wong, K. H., et. al., Multiplex Illumina Sequencing Using DNA barcoding, Current Protocols in Molecular Biology, Ch. 7, Unit 7.11, 2013.

[38] Jaenisch, R., Bird, A., Epigenetic Regulation of Gene Expression: How the Genome Integrates Intrinsic and Environmental Signals, Nature Genetics Supplement, Vol. 33, 2003.

[39] Kramer, M. F., Coen, D. M., The Polymerase Chain Reaction, Current Protocols in Molecular Biology, Wiley Interscience, Ch. 15, Unit 15.0.1-15.0.3, 2009.

[40] Tiwari, P. K., Lakhotia, S. C., Restriction Enzyme Digestion of Heterochromatin in Drosophila Nasuta, Journal of Biosciences, Vol. 16, No. 4, pg. 187-197, 1991.

[41] ten Bosch, J. R., Grody, W. W., Keeping Up With the Next Generation: Massively Parallel Sequencing in Clinical Diagnostics, Journal of Molecular Diagnostics, Vol. 10, No. 6, 2008.

[42] Gnerre, S., et. al., High-Quality Draft Assemblies of Mammalian Genomes from Massively Parallel Sequence Data, PNAS, Vol. 108, No. 4, pg. 1513-1518, 2010.

[43] Luo, R., et. al., SOAPdenovo2: An Empirically Improved Memory-Efficient Short-Read de novo Assembler, GigaScience, Vol. 1, No. 18, 2012.

[44] Zimin, A. V., et. al., The MaSuRCA Genome Assembler, Submitted to Genome

Biology, 2013.

[45] https://github.com/joshuaburkhart/bio/blob/master/plot_coverage.R

[46] http://www.ebi.ac.uk/~zerbino/velvet/Manual.pdf

[47] https://github.com/joshuaburkhart/bio/blob/master/param_ests.rb

[48] Kurtz, S., et. al., Versatile and Open Software for Comparing Large Genomes, Genome Biology, Vol. 5, R12, 2004.