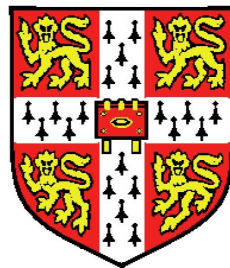


Efficient sequence assembly and variant calling using compressed data structures



Jared Thomas Simpson
Queens' College
Wellcome Trust Sanger Institute
University of Cambridge

This dissertation is submitted for the degree of

Doctor of Philosophy

September 2012

I took a lengthy path to reach this point and my family supported me the entire way. I dedicate this work to them - thank you Mom, Dad, Calley and Kim.

Declaration

This dissertation describes work carried out from May 2009 to July 2012 under the supervision of Dr Richard Durbin at the Wellcome Trust Sanger Institute, while a member of Queens' College, Cambridge. This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text. The content in Chapter 2 was published in [Simpson and Durbin \[2010\]](#). The content of Chapter 3 was published in [Simpson and Durbin \[2012\]](#).

This thesis does not exceed the length limit of 60,000 words as specified by the Biology Degree Committee.

Jared Thomas Simpson

August 29, 2012

Acknowledgements

My decision to undertake a Ph.D. was influenced by many of my friends and colleagues. I owe thanks to my close friends Gwynn Elfring and Scott Drader, who have been a constant source of support and advice. My friend Christian Steidl introduced me to Computational Biology. This introduction led me to work on the sequence assembly problem, which directly led to this work.

My Ph.D. benefitted from many people at the Sanger Institute, most importantly my supervisor Richard Durbin. Without Richard's guidance and insight this work would be greatly diminished. I am very appreciative of the opportunity to work with Richard and proud of the work we accomplished over the last four years. I had many great discussions within Richard's group in journal clubs, group meetings, lunches and over coffee. These discussions introduced me to many areas of genetics and provided a great environment for developing as a scientist. A strength of the Ph.D. program at Sanger is the opportunity to build collaborations across many different research areas. Within Richard's group I enjoyed collaborating with Leopold Parts, Aylwyn Scally and Kees Albers. I also appreciate the time I spent working in David Adams' lab and the Cancer Genome Project under Peter Campbell. I owe thanks to the Wellcome Trust for financially supporting my Ph.D. and Annabel Smith and Christina Hedberg-Delouka for making sure the graduate program at Sanger runs smoothly.

Summary

De novo genome assembly is one of the most computationally demanding problems in genomics. In this thesis, I describe a collection of novel algorithms for performing *de novo* assembly using compressed data structures. First, I describe an algorithm to directly construct the assembly string graph - a model of overlap-based sequence assembly - using the compressed FM-index data structure. Previous algorithms for constructing the string graph required the intermediate step of building a full overlap graph, then removing transitive edges from the graph. My novel FM-index based algorithm does not require this time-consuming intermediate step. This algorithm allows fast and memory efficient overlap-based assembly. In Chapter 3, I extend my FM-index algorithms to build a space-efficient assembler for real sequencing data by designing error correction, read merging and scaffolding algorithms. Using these efficient algorithms I am able to reduce the memory requirement for assembling a human genome to 54GB.

In Chapter 4, I address the problem of detecting DNA sequence differences between two related genomes - the *variant calling* problem. Traditional approaches to variant calling align short sequence reads to a reference genome. While this approach is effective for simple differences, like isolated SNPs, it is more difficult to find complex changes like the insertion or deletion of sequence. My approach is based on analyzing the structure of an assembly graph built from the sequence data from multiple individuals. In Chapter 5, I apply this approach to real sequencing problems, including finding *de novo* mutations in the child of two parents, somatically acquired mutations in cancer and polymorphic variants present in a large human population.

Contents

Contents	v
List of Figures	ix
1 Introduction	1
1.1 DNA Sequencing	2
1.1.1 High Throughput Sequencing	4
1.2 Sequence Assembly	6
1.2.1 A Practical Overview of Assembly	9
1.2.2 The Topology of Assembly Graphs	10
1.2.2.1 Graph Tips	10
1.2.2.2 Graph Bubbles	11
1.2.2.3 Repeats	11
1.2.2.4 Unipaths	12
1.2.2.5 Assembly Software	13
1.3 Resequencing and Variant Calling	14
1.4 Compressed Data Structures	15
1.5 Overview of this work	17
2 The FM-Index and Genome Assembly	19
2.1 Introduction	19
2.1.1 Publication Note	19
2.2 Definitions and Notation	19
2.2.1 Genomes and Sequence Reads	20
2.3 Assembly Graphs	21

2.3.1	Overlap Graphs	21
2.3.2	de Bruijn Graphs	23
2.3.3	The String Graph	24
2.4	The Suffix Array, BWT and FM-Index	28
2.4.1	The Generalized Suffix Array	31
2.5	Direct Construction of the String Graph	31
2.5.1	Building an FM-index from a set of sequence reads	31
2.5.2	Overlap detection using the FM-Index	32
2.5.3	Detecting irreducible overlaps	34
2.5.4	Results	38
2.6	Representing a de Bruijn Graph using the FM-Index	41
3	The SGA Assembler	44
3.1	Introduction	44
3.1.1	Publication Note	44
3.1.2	Algorithm Overview	45
3.2	SGA Algorithms	46
3.2.1	Construction of the FM-index for large read sets	46
3.2.2	k -mer based error correction algorithm	47
3.2.3	Overlap based error correction	49
3.2.3.1	Finding Inexact Overlaps with the FM-Index	50
3.2.3.2	Overlap Based Error Correction Algorithm	52
3.2.4	Read filtering	53
3.2.5	Read merging and assembly algorithm	53
3.2.6	Paired end reads/Scaffolding	55
3.2.7	Implementation Details	57
3.2.7.1	FM-Index Implementation	57
3.2.7.2	Program Design, Implementation and Libraries	58
3.3	Results	59
3.3.1	Index construction results	60
3.3.2	<i>C. elegans</i> Assembly	61
3.3.2.1	Substring coverage	62
3.3.2.2	Assembly Contiguity	63

3.3.2.3	Assembly Completeness	64
3.3.2.4	Assembly Accuracy	65
3.3.2.5	Computational Requirements	66
3.3.3	Human Genome Assembly	66
3.3.4	The Assemblathon	70
3.3.5	<i>Schizosaccharomyces pombe</i> assemblies	71
4	Algorithms for Variant Detection from an Assembly Graph	74
4.1	Introduction	74
4.1.1	Collaboration Note	75
4.2	Algorithms	76
4.2.1	Motivating Example	77
4.2.2	Discovering Candidate Variants	78
4.2.3	de Bruijn graph haplotype generation	80
4.2.4	String graph haplotype generation	83
4.2.5	Haplotype quality control	86
4.3	Probabilistic realignment	86
4.3.1	Extracting Haplotype Reads from the FM-Index	87
4.3.2	Probabilistic read-haplotype alignment	88
4.3.3	Annotating variants in the candidate haplotypes	88
4.3.4	Aligning haplotypes to a reference genome	89
4.3.5	Comparative variant-calling	89
4.3.6	Population calling	91
4.4	Discussion	92
5	Assembly-Based Variant Calling Results	94
5.1	Introduction	94
5.1.1	Implementation Note	95
5.2	The power to detect variants using unique k -mers	95
5.3	Simulated single-genome variants calls	96
5.3.1	Computation Requirements	99
5.4	Simulated genome comparison	99
5.4.1	Computation Requirements	101

CONTENTS

5.5	Reference-based Substitution Calls	101
5.6	Estimating the background error rate for comparative variant calling	105
5.7	Calling <i>de novo</i> mutations in a trio	106
5.8	Cancer mutations	109
5.8.1	Analysis Notes	115
5.9	Low-Coverage Population Calls	115
5.9.0.1	Computation Requirements	117
5.10	Discussion	118
6	Conclusions	119
	References	121

List of Figures

1.1	A simple tip in an assembly graph. The red vertices contain sequencing errors - due to these errors the sequence of this branch diverges from the rest of the graph (grey vertices). The arrows on the terminal grey vertices are to indicate the graph continues off-page.	10
1.2	A bubble in the graph showing the distinctive divergence/collapsing signature.	11
1.3	A simple repeat in the assembly graph. The red nodes represent sequences present multiple times in the genome.	12
1.4	A unipath graph constructed from the graph depicted in figure 1.3. The unambiguously connected vertices have been merged together.	13
2.1	Diagram of a simple assembly graph. Three overlapping reads (R_1, R_2, R_3) are shown in panel A. Panel B shows the graph constructed from the overlaps between the reads. The arrowheads pointing into the nodes depict an edge of type P and arrowheads pointing away from the nodes depict edges of type S. For example the edge between R_1 and R_2 is a <i>SP</i> -edge. The edge $R_1 \leftrightarrow R_3$ is transitive. Removing this edge will turn the graph into a string graph.	27
2.2	The running time of the direct and exhaustive overlap algorithms for simulated <i>E. coli</i> data with sequence depth from 5X to 100X.	39
3.1	Schematic of the flow of data through SGA.	45

LIST OF FIGURES

3.2	<i>k</i> -mer occurrence histogram for simulated perfect data (left) and simulated data with 1% uniform base calling errors (right). The y-axis records the number of times a <i>k</i> -mer with frequency <i>x</i> occurs in samples of the data set. For example, there are 57,059 <i>k</i> -mers seen 20 times in the perfect data set. The histogram was calculated by sampling 10,000 random reads.	48
3.3	Reference string coverage analysis for the <i>C. elegans</i> N2 assembly. For string lengths from 50bp up to 5,000bp, 10,000 strings were sampled from the consensus-corrected <i>C. elegans</i> reference genome. The proportion of the strings found in the SGA, Velvet, ABySS and SOAPdenovo assemblies is plotted.	63
3.4	The number of bases of the <i>C. elegans</i> reference genome covered as a function of minimum contig alignment length.	64
3.5	The amount of the human reference genome covered by a contig as a function of the minimum contig alignment length. For each length <i>L</i> on the x-axis, contig alignments less than <i>L</i> bp in length were filtered out and the amount of the reference genome covered by the remaining alignments was calculated.	69
3.6	The relationship between sequence coverage and contig N50 for the <i>S. pombe</i> data set. The plot in the left panel displays the complete data set. The plot in the right panel only shows strains that have <100X coverage.	72
3.7	The relationship between sequence coverage and CPU time for the <i>S. pombe</i> data set.	73
4.1	A bubble in a de Bruijn graph built from G_v and G_c . The grey <i>k</i> -mers (labelled K_1 and K_2) are shared between G_v and G_c and are the entry/exit points of the bubble. The red and blue vertices represented <i>k</i> -mers unique to G_v and G_c , respectively.	77
5.1	The <i>k</i> -mer detectability of point mutations introduced into the human reference genome. The black line indicates the proportion of introduced variants that are detectable at a given <i>k</i> . The red line indicates the proportion of variants that form clean bubbles.	96

LIST OF FIGURES

5.2	Sensitivity (left panel) and precision (right panel) of reference-based calls on simulated data. Note the different range of the y-axis in each panel.	98
5.3	Sensitivity (left panel) and precision (right panel) of the simulated genome comparison. Note the different range of the y-axis in each panel.	101
5.4	The left panel plots the proportion of mapping-based SNP calls using GATK that were found by the de Bruijn graph and string graph callers as a function of k . In the right panel, the proportion of SNP calls that are found in dbSNP v1.32 is plotted.	103
5.5	The proportion of mapping-based SNPs found (left) and the proportion of our SNP calls contained in dbSNP v1.32 (right) for the downsampled data set.	104
5.6	The allele frequency distribution for substitution calls made by the string graph caller	113
5.7	The allele frequency calculated by the string graph caller (x-axis) and CGP (y-axis) for calls made a common sites	114
5.8	The allele frequency distribution for SNP and Indel calls on the AFR continental group of the 1000 Genomes Project	116
5.9	The distribution of insertion (positive) and deletion (negative) lengths for the 1000 Genomes data set. The data set consists of 35,846 assembly indel calls (black points) and 64,319 mapping calls from Phase 1 of the 1000 Genomes Project (red points). Events larger than 50bp were excluded from this plot.	117